Nilssen, Mikael
Helland, Andrè

Distributed Instrument Cluster

Bachelor's project in Computer Science
Supervisor: Styve, Arne
May 2021

Bachelor's project

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

NTNU
Kunnskap for en bedre verden

Nilssen, Mikael
Helland, Andrè

Distributed Instrument Cluster

Bachelor's project in Computer Science
Supervisor: Styve, Arne
May 2021

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

NTNU
Kunnskap for en bedre verden

NTNU
Kunnskap for en bedre verden

Department of Computer Science

IE303612 - Bachelor Thesis

Distributed Instrument Cluster

Authors:
Mikael Nilssen, Andre Helland

Supervisor:
Arne Styve

20th May 2021

Contents

Preface

This report is the final submission of the a bachelor thesis by computer science 2 students at NTNU Campus Ålesund.

During the project we developed a prototype for a remote control system with a Web user interface, that targets maritime instruments like Radar, or GPS. Maritime devices are normally locked so software can not be installed on them like normal computers, and this requires a system that can control the device externally, in our case hardware connected via USB. The system we have developed can in theory control any computer that supports the USB HID interface used by our hardware.

This project is part of a DIKU financed research project.

Acknowledgments

We would like to thank our supervisor Arne Styve for helping and guiding us through this project. We would also like to thank Norvald Kjerstad the product owner and also Lars Ole Hurlen our designated contact during the project for being a big help, being heavily involved in the development process, as well as going out of his way to teach us valuable development skills. Thank you Furuno for allowing us to test on your devices.

Abstract

The goal of this project was to create a web based user interface that can remote control maritime devices to provide realistic data for teachers to use in classrooms and courses when training students and personnel in the use of these devices. Existing solutions for remote control of systems exists with technology like VNC, but many maritime devices are proprietary and do not allow for the installation of any software, requiring for an external solution.

Our solution to this problem was the implementation of an interface using hardware that can emulate a mouse and keyboard via USB. The main challenges in the implementation was the low bandwidth the devices would have available to them, as well as the latency needed when a user is directly controlling the remote device, if the response time is to large the experience for users will be terrible. The video would also need to be compressed to reduce the data used by devices, so the system could be economically viable when using 4g data.

Our solution implements many technologies like Blazor and ASP.NET. Our implementation of compressed video uses MJPEG compression to reduce bandwidth usage. To capture user input we use events in Blazor together with mouse locking so you can directly control the remote device with your own mouse and keyboard through the website.

The result of the thesis was a prototype of a system for remote controlling devices with a mouse and keyboard, or a virtual keyboard. Supporting multiple video devices on the remote device. The solution uses a external control method that bypasses DRM.

Terminology

- **Git** - Version Control System
- **Github** - Online implementation of GIT
- **Blazor** - Framework for C# Web Development via Web assembly
- **4G** - Broadband cellular network for mobile network
- **Latency** - How long it takes before an action is preformed and the goal is achieved
- **RTC** - Real Time Connection
- **WebRTC** - Googles implementation of a real time web protocol
- **HLS** - HTTP Live Streaming
- **CODEC** - Encodes and decodes data, often used when referring to compression
- **Bitrate** - Number of bits that are conveyed or processed per unit of time
- **Emulate** - Reproduce the function or action of (a different computer, software system, etc.)
- **Digital rights management (DRM)** - A set of access control technologies for restricting the use of proprietary hardware and copyrighted works
- **User Story** - Description of a feature in a development project
- **CPU** - Central Processing Unit
- **GPU** - Graphics Processing Unit
- **RAM** - Random-access memory is a form of computer memory that can be read and changed in any order, typically used to store working data and machine code
- **API** - Application Interface
- **HTTP** - Hypertext Transfer Protocol
- **HTTPS** - Hypertext Transfer Protocol Secure
- **TLS** - Transport Layer Security
- **TCP** - Transmission Control Protocol
- **UDP** - User Datagram Protocol
- **W3C** - World Wide Web Consortium
- **P2P** - Peer To peer
- **DOM** - Document Object Model, The structure of a html/xml document
- **IOT** - Internet Of Things
- **HTML** - The HyperText Markup Language, or HTML is the standard markup language for documents designed to be displayed in a web browser
- **CSS** - The language we use to style an HTML document
- **JS** - JavaScript is the programming language of the Web
- **Firefox** - Firefox Browser, also known as Mozilla Firefox or simply Firefox, is a free and open-source web browser
- **Chrome** - Google Chrome is a cross-platform web browser developed by Google

- **IntelliSense** - Intelligent code completion is a context-aware code completion feature in some programming environments that speeds up the process of coding applications by reducing typos and other common mistakes
- **UI** - User Interface
- **Lag** - A period of time between one event and another
- **FPS** - Frames Per Second
- **FHD** - Full HD, 1920 pixels displayed across the screen horizontally and 1080 pixels down the screen vertically
- **URL** - A Uniform Resource Locator (URL), colloquially termed a web address, is a reference to a web resource that specifies its location on a computer network
- **IP address** - A numerical label assigned to each device connected to a computer network that uses the Internet Protocol
- **Loopback** - The routing of electronic signals or digital data streams back to their source without intentional processing or modification. It is primarily a means of testing the communications infrastructure
- **OBS** - OBS (Open Broadcaster Software) is free and open source software for video recording and live streaming
- **VLC** - A free and open source cross-platform multimedia player
- **CI/CD** - Continuous Integration and Continuous Development
- **UML** - Unified Modeling Language
- **HID** - Human interface device
- **MVC** - Model View Controller
- **DBMS** - Database management System
- **Parser** - Formal analysis by a computer of a sentence or other string of words into its constituents

List of Figures

List of Tables

1 Introduction

The goal of the project is to create a system that can remote control maritime devices via an online web user interface so that instructors that require data from maritime devices can have access to real instruments and data when training students. Solutions for this problem exists in the form of VNC and other remote control software, however for maritime devices that are proprietary and have various content locks where you can not download software to or access the system, it requires an external solution. The main goal of this project is to research the possibilities of creating such a system and testing if it is possible to remote control these devices reliably. The solution was left mostly up to us in terms of technology, however the main control device was suggested and already tested when we began.

The challenges for this project involves research around which devices can be controlled. Research regarding latency of a remotely located device, and reducing the packet sizes to a level which makes this a viable solution for low bandwidth remote locations, as well as having a responsive and user friendly system with low levels of latency.

This report will contain information about our solutions for solving various problems regarding video compression, latency, and what technologies were used to achieve this. The final result of the thesis of will be detailed as well as a discussion around decisions that were made.

1.1 Scope

This is a project that will be continuously developed on by others for extended functionality. A complete application was not expected as the result of this thesis. The maritime devices we focused our development around was the RPU-025 Furuno radar.

2 Theory

In this chapter we will explain some of the literature and techniques/technologies you need to know about to understand the rest of the thesis.

2.1 Video Compression

2.1.1 JPEG

JPEG or JPG is one of the most used compression formats for storing and transmitting digital images.

JPEG compression is a lossy compression method meaning data from the image file is being removed to reduce the size of the file. Lossy compression methods introduce artifacts, and the severity of these artifacts can be adjusted using a quality variable when encoding ranging from 1 to 100. The quality variable adjusts how much data the compression algorithm has to spend, higher quality results in larger files but less artifacts.

Humans perceive artifacts differently, because of this JPEG compression is strategic about what data it should keep and how much it can compress it. One example is how JPEG compresses color information, the human eye can see more detail in brightness compared to color and hue, the compression algorithm will then dedicate more data to brightness over color and hue when compressing an image.

Because JPEG is an old compression codec it is widely used and has good support. Due to this it is common for computer chip makers to add circuitry to CPU's and GPU's that are dedicated to decoding and encoding of JPEG images as fast and power efficiently as possible, this is also known as hardware acceleration. This allows for a wide variety of devices to take use of the image format and make low powered devices able to handle larger files than they would have been without hardware acceleration.

Wikipedia (i) *Leveraging the Hardware JPEG Decoder and NVIDIA nvJPEG Library on NVIDIA A100 GPUs*

2.1.2 MJPEG

MJPEG or motion JPEG uses the JPEG compression previously mentioned but applies it on the individual frames in a video file. This per frame compression method is also known as intraframe compression. Because JPEG compression is an image compression algorithm it can only use information on a per image basis. This means that MJPEG can not use more advanced compression methods involving the change from one frame to another. Every frame has to be compressed as if the information in the frame has never before been seen. The benefit of this is that motion intensive videos don't suffer from the same compression artifacts more advanced interframe video compression methods do.

This limitation makes the compression ratio of MJPEG comparably low to other more advanced compression methods. The simplicity however makes the latency of the compression method low as it does not rely on a frame buffer. This reduced latency can help improve user experience if MJPEG is used to display video of a device being controlled.

MJPEG uses the same compression algorithm as JPEG images so the wide support of JPEG is inherited because MJPEG can make use of the same hardware acceleration as JPEG. This again allows for lower power devices to play MJPEG they might not have been able to with hardware acceleration.

Wikipedia (l)

2.1.3 H264

The H.264 codec is a widely used and mature video compression codec. This makes it widely supported and relatively easy to implement compared to less mature codecs.

H.264 is an interframe compression codec meaning it uses so called P and I frames to produce a video. The codec uses P frames to give a new frame, then uses an I frame containing information about the difference between the two frames to reconstruct the 2nd frame in a video sequence. This method uses substantially less data compared to sending a completely new frame each time. Interframe compression codecs strength is shown best when compressing a video with little movement. If a video with a moving foreground subject has a mono-colored background with no movement, the compression algorithm can encode an I frame detailing that the background did not change and only data about how the foreground subject moved is required to specify how the image changed.

The H.264 codec has more compression methods like B slices, multiple reference frames, chroma subsampling, entropy coding. x264 is a specific algorithm implementation of the h.264 codec and has wide support for different chroma subsampling levels and bit depth of colors.

Wikipedia (a)

2.1.4 AV1

As of writing AV1 video compression is the most advanced and recent video compression codec. It is able to achieve 50.3% higher data compression compared to x264. This level of compression is very helpful in low bandwidth situations like transferring a video feed from sea over satellite connection.

Due to its recent development AV1 is not widely supported and only new devices have hardware acceleration support. As of writing the only consumer devices capable of hardware accelerated decoding is RDNA2 devices from AMD and 3000 series GPU's from Nvidia. Only professional grade equipment is capable of hardware accelerated video encoding. Without hardware accelerated encoding achieving real time encoding speed for use in live streaming a control feed of a maritime device becomes significantly harder and computationally expensive.

Unlike H.264, AV1 is royalty free making adoption of the codec faster and more appealing for open-source projects. This makes AV1 a likely candidate to be the standard video codec of the future. Adaptation of the codec is already faster than H.264 when it was initially released. As chip makers make hardware decoders and encoders more available the codec will likely become more widely used and surpass H.264. However as of writing, this is not the situation and implementing AV1 into the solution might not be as smooth as using more widely used H.264.

Wikipedia (b) *AMD RDNA™ 2 Graphics Architecture* *Video Encode and Decode GPU Support Matrix*

2.2 Video Streaming

2.2.1 Protocols

Streaming video starts at the encoding step and compresses the video down to a smaller more manageable size. To transport that compressed data over a network reliably a protocol is necessary. There are many different protocols for transferring video streams over a network and they all have different pros and cons.

Real-time Transport Protocol/Real Time Streaming Protocol (RTP/RTSP):

The protocol is commonly used in security cameras and for transport of video over local networks. RTSP rarely has playback support on consumer player and devices making it impractical for use as a protocol for delivering video to end users. RTSP has a comparably low latency of 2 seconds and can be made lower with tuning.

HLS:

The protocol is developed by Apple and has wide support on end user devices for playback. It is the most used protocol for end user delivery of video streams. The protocol supports adaptive bitrate allowing clients to seamlessly switch to higher or lower bitrate streams depending on network bandwidth. The protocol has latency at around 6-30 seconds but can be tuned to 2 seconds using Low-Latency HLS.

WebRTC:

WebRTC is a near real time streaming protocol with latency below 500 milliseconds. The protocol has wide playback support on end user devices. The protocol was designed as a peer to peer protocol making it scale poorly, this makes it sub optimal for one to many video streaming.

Streaming Protocols: Everything You Need to Know (Update) Wikipedia (p)

2.2.2 FFMPEG

FFMPEG is a open source command line program made for handling multimedia. FFMPEG supports many different codecs and protocols making it used by many. It is the core of the popular and free video play VLC and is used in the video processing pipeline at YouTube.

FFMPEG has built in API's for many hardware encoder and decoders making it support Nvidias NVENC, Intel Quick Sync and many more making it capable of easily decoding and encoding videos fast and efficiently.

Wikipedia (f)

2.3 Serial Interfacing

2.3.1 Serial port

A serial port is an interface where information is transferred in or out one bit at a time. Interfaces such as Ethernet and USB also send data as a serial stream but typically only RS-232 compliant hardware is referred to as serial port.

Serial ports are mostly obsolete on modern computers but were common on computers in the early 2000's and earlier. Serial ports are still in use today on configuration ports for routers and in servers. Modern PCs can still connect to serial devices using a USB to serial adapter.

Wikipedia (s) Wikipedia (c)

2.3.2 Crestron Cable

Crestron's CBL-USB-RS232MK-6 cable is a serial to USB adapter allowing for serial byte commands to emulate keyboard strokes and mouse commands. The cable allows for make and break (up and down) byte commands for individual keys making it possible to preform all key combinations imaginable. The cable also supports commands for mouse movements with a command for different magnitude allowing for large movements done quickly.

Due to the emulation nature of the device any machine it is used on will just see the cable as a normal human interface device or HID. To the target machine the cable is indistinguishable from a real HID this allows for bypassing of any DRM system put in place on the machine.

CBL-USB-RS232KM-6

2.4 Network Protocols

2.4.1 TCP

Transmission Control Protocol or TCP is a transportation layer internet protocol, and is one of the main protocols. TCP is an ordered, reliable transfer protocol with error checking. This ensures that all data sent will arrive on the other side, and can be put back together in the order it was sent in if they arrived at different times, and if the data is wrong it can be detected with a check-sum calculation. TCP is also connection based, having to connect to a remote endpoint before it can send data.

rfc675

2.4.2 UDP

User Datagram Protocol or UDP is a like TCP a transportation layer internet protocol, and is also one of the main protocols. UDP is connection less and does not have any ordering or reliability built in. UDP does also have checksum for error correction. UDP is better for Real-Time applications since it does not spend time on Congestion control and waiting for packets to arrive, or requesting lost packets like in TCP. UDP is therefore in theory better for applications with real-time video streaming in terms of latency, if the video is not recorded. UDP is often blocked by firewalls on many networks and on windows, which makes TCP necessary for connections in some circumstances.

rfc768

2.4.3 ICMP

Internet Control Message Protocol is a network layer protocol used communicate between networking devices. It sends error messages that occur when communicating with other devices on other IP addresses, like when the requested address can not be reached. It is used when echoing pings to check if an IP Endpoint is available.

rfc1812

2.5 Networking

2.5.1 Async Socket

C# Synchronous sockets will suspend applications while waiting for network operations to complete, locking the thread from completing other work. C# Asynchronous sockets will continue execution of the current thread, and assign a new thread from a thread pool to wait for a callback, allowing the calling thread to continue running.

Microsoft (a)

2.5.2 Firewall

A Firewall is a system that monitors incoming and outgoing network traffic. Firewalls operate on rules that can be setup to let in packets from specific addresses or other attributes, like allowing only certain ports or protocols to enter. Firewalls do packet filtering inspecting packets that travel between points in the network, dropping the ones that are unwanted. Firewalls can be run as separate network devices, or as services in an operative system. Windows has its own firewall as an example. Next-generation firewalls can have more advanced features like filtering web requests to identify unwanted content.

2.6 Web Technologies

2.6.1 HTTP

HTTP is an application layer protocol in the OSI Model. Http is the backbone of the modern web and has gotten multiple updates over the years, Http/1, Http/2, Http/3 are all implementations of the http protocol, and not every website support all http implementations.

Http/1: Http/1 is the baseline http implementation that is used for transferring hypertext documents over the internet.
Http/2: Http/2 was primarily developed by google and added many new features to the http protocol such as compression, multiplexing and prioritization. The main difference between Http 1 and 2 is that http/2 is not plain text, and is encapsulated in binary, which allows for other transport methods that are more efficient. In Http/2 the server can push data to the the requesting client which removes the need for more request cycles that cause delay when transferring. TLS Encryption is not inherently mandatory in the Http/2 protocol, but all the major browsers have stated they will only support TLS on http/2 which makes TLS semi-mandatory.
Http/3: Http/3 is a new major version for http. The main difference in the http/3 is that the underlying transport protocol TCP, is being changed for UDP, which allows for improvements in various streams that are delayed by the inherent packet security of TCP, reducing latency and overhead. The guaranteed transfer of packets is handled outside of the transport protocol.

The main effect this has on web development is that in the near future Http based streaming and latency will improve with the introduction of Http/3. Http/3 is currently not supported by all the major browser providers. Http/3 implementation on an application like the one we are developing could provide reduced latency for streaming.

Wikipedia (g)

2.6.2 HTTPS

HTTPS is a secure version of http that runs over TLS. TLS is a cyrptographic protocol that encrypts the data travelling over a connection with the use of a shared Certificate. Having an encrypted connection prevents man in the middle attacks, and eavesdropping and tampering on the data of your connection. To establish a secure connection the server must have a valid Certificate, that is signed by a trusted third party verifying that the connection is safe. The web is slowly transforming to Https only, so the implementation of systems like our Mjpeg server, which is HTTP only are blocked as insecure.

Wikipedia (h) Wikipedia (v)

2.6.3 Websocket

A Web-socket is the full-duplex connection type of the web, allowing for communication from end to end without the request based half-duplex of a http connection. Full-duplex means that data can travel both ways at the same time, as opposed to a half-duplex system where both sides can also send data, but only one at a time. Web-sockets are not http connections, but are made to work with the http ports of 80 and 443, when creating a web-socket connection the http connection will request a protocol upgrade and signal switching of protocols. Web-sockets also have a secure version that runs over TLS. Web-sockets are useful in applications that demand real-time communication between server and client because it removes the overhead from http. In our application the real time nature of web-sockets were useful in reducing latency sending data to the server.

Wikipedia (x)

2.6.4 Web Assembly

Web Assembly or Wasm is a relatively new open standard that first appeared in 2017, and was added to the W3C as a web standard in 2019, on the same level as Java Script, HTML, and CSS. Web assembly defines a portable binary-code format for high performance applications on web pages. The standard is not exclusive for web pages and can in theory run anywhere, this makes it very versatile in terms of porting code bases to various platforms and turning web apps into standalone apps for desktop or phone. Web assembly can also support any language that has a compiler which compiles to web assembly.
Web Assembly has some limitations:

1. No direct access to the DOM, Interactions with the DOM have to go through JavaScript Interop.
2. No Multi-threading support.
3. No Garbage Collector.
4. Security Flaws.

There are plans for adding both threading support and garbage collection to web assembly. The delay in threading support currently is due to the security flaws possible where threading could circumvent Spectre and Meltdown prevention in the browser. There are also concerns around the prevention of ad-blocking and tracking prevention due to the obscured code.

WebAssembly

2.6.5 Web RTC

Web Real-Time Communication is an open-source project that aims to provide real-time communication to web pages. WebRTC provides P2P video and audio communication between Web browsers of high-quality using simple browser API's. WebRTC has very low latency in their End-to-end communications and is regarded as one of the best protocols for web when it comes to latency. The latency of WebRTC is measurable to around 500ms depending on conditions, it is ideal for ultra-low latency streaming of video. However WebRTC uses UDP, which will be blocked by many firewalls, this makes utilization of Turn Servers that broadcast as TCP to the clients a requirement for clients that can not have a UDP connection which will have an impact on the latency. WebRTC is a uni-cast protocol, that means it communicates in one-to-one connections which is not ideal when creating a broadcasting system, but it can still be used.

WebRTC Wikipedia (w)

2.6.6 Signal R

Signal R is a software library developed by Microsoft allowing for asynchronous notifications to be sent to client-side applications from the server-side. Signal R utilizes web-sockets but can fall back to http if it is not supported. Signal R aims to provide Real-time communication between web applications.

Wikipedia (t)

2.7 User Experience

2.7.1 Latency

Latency is the delay between an action, and the cause and effects of the action, or some other system change. In a live video streaming application the word latency is in reference to how long it takes for the image to update on the user end, in relation to when the the picture was taken on the other side.

Latency is very important when it comes to user experience when the user controls the device directly, if the time between the user preforming an action and the feedback displaying on the screen is large, the product will not be pleasant to use, sometimes the latency can be so big that the application becomes unusable. The ideal latency for anything to be human controlled, is less than 50 ms, because that ensures that it is below what humans regularly experience. This is obtainable in regular networking applications when the distance travelled is not to large, but if you need to compress and decompress video in your application, it adds a layer of latency that will take the overall latency above this threshold in most circumstances. A balance between usability and lowered file sizes is central in a system like the one we are creating.

Wikipedia (k)

2.7.2 User Interface Design

When designing a user interface it is important to consider various things for it to be effective and easy to use. You can use Design Principles like Don Norman's Principles of design, to guide users through the usage of the interface and avoiding hard to use systems.

Don Norman's six principles of design are:

1. Visibility
   The user should be able to find features and recognise what the features do easily.
2. Feedback
   When Interacting with something there should be clear a response, or indication that the interaction had an effect, like sounds or visual change.
3. Affordance
   Affordance is the relation between how something looks and how its used. Buttons should look like they can be clicked. Features should look like what people expect them to look like so they are easily found.
4. Mapping
   The relation between how you control something and the effect it has. A scroll bar that scrolls the page up or down, the way you drag the bar should reflect the response. E.G Dragging up should result in upward motion.
5. Constraints
   Features should be have limited functionality, and be easy to use, if a single part of the application is to advanced it will have a negative effect on the user experience.
6. Consistency
   When preforming an action repeatedly the result should stay the same, and not change.

Enginess

2.8 Agile Development

2.8.1 Scrum

Scrum is an agile development method, that is meant to improve the development process, allowing for changes in priorities and close interaction with the product owner for a better result.

The scrum development process consists of a repeated process of a sprint meeting, followed by a sprint review, and then a retrospective meeting, before you repeat the process. Some development teams also have daily meetings where they update each other so everyone has a view of the current state of the product.

The sprint planning meeting is where the product owner defines what he wants to accomplish. The Development team and the product owner work together in defining user stories for the sprint and backlog, the product owner decides what he wants to prioritize, and the Development team decides how long each Story will take to implement. At the end of the meeting the sprint will be defined with a goal for what should be completed after the sprint.

The sprint review is a meeting with the stakeholder/product owner at the end of each sprint where the team shows what they have accomplished, The focus is mainly on what was completed, but what to do about the incomplete work is discussed.

In Retrospective meetings the development team should reflect on the sprint they had and what they accomplished, typically define what went well and what did not go well, and make suggestions for improvement.

Agile development is very common for software development projects due to the difficulty in predicting future challenges and needs when planning, the iterative process allows for changes in priority and goals at a fast pace. Software development projects also have difficulties when communicating between the product owners that do not have technical insights into how development and software functions, consistent involvement during the sprint iterations, allow the product owner to better specify their needs, and to be more connected to the product in general, which normally leads to better results for the project.

Wikipedia (r)

2.8.2 Scrum-Roles

Scrum Master: The Scrum master is responsible for making sure the scrum procedures are followed, and that the team is functioning optimally, and remain focused on the tasks given to them. The Scrum master is not a team leader, but more of a helping role that can help with all tasks that are related to the scrum procedure.

Product Owner: The product owner is responsible for representing the product stakeholder during the meetings as well as managing the backlog of tasks for the developers to complete, and giving priority to them, assuring that the customers will be satisfied.

Developers: The developers work on the product and develop value. The term developer does not only mean software developer, but all participants that add value to the project.

Wikipedia (r)

2.8.3 Time estimation and Story points

In development projects and scrum there are two main ways used to measure the progress of a project, Time estimation and Story points. Time estimation is just a measure of how many work hours you think a task will take to complete. This method is very hard to get accurate measurements with, which is why the more abstract method of story points allocating is more common. Story points are arbitrary points allocated as a team, the arbitrary points tend to be better estimations than time after a team has worked together for a while.

2.8.4 Extreme programming

Extreme Programming is an agile software development methodology that focuses on software quality and responsiveness to changing customer requirements. This means that like in other agile methodology extreme programming facilitates iterative releases. Extreme programming also has other elements like pair programming, which is supposed to increase code quality by having 2 people code together where one codes, and the other spectates. Extreme programming also includes Extensive code reviews and unit testing, and not implementing code until it is actually needed for a feature requested by the client. Extreme programming also has many principles like striving for simplicity which goes against code reusability, and embraces reworking code if the requirements from the customer changes, which is quite different from normal coding principles where code reusability is important.

Extreme programming

2.9 Database

2.9.1 Sql

Relational Database management systems are databases that store data in managed tables of columns and rows where data is mapped with Primary keys representing a specific row, each row is called a record. Each record can reference the Primary Keys of other records in other tables as their own foreign keys creating a relationships between the data in the database. This allows for searching and mapping of different tables allowing you to connect the data in many useful ways, filtering based on different attributes in many tables. Indexing of the data in the database can increase the search time of the database, at the cost of the extra storage space used by the index, allowing for optimizations of tables that are accessed often.

Wikipedia (q)

2.9.2 noSql

NoSql or Non-relational databases refers to database implementations that do not fit into the category of relational database. The various noSql databases may store databases, but they normally are linear and for this reason they are very easy to scale. NoSql databases also use a key to represent each data input for identification. NoSql databases are normally faster than relational databases but do not offer the same level of consistency.

Wikipedia (m)

2.9.3 Entity Framework Core

Entity Framework Core is an Object-relational mapper or ORM that allows dotNet developers to use objects when doing data access. Entity Framework Core has three main usage types:

1. **Code first** Code first is where you already have a model set you want Entity Framework to construct a new database from for you automatically.
2. **Database First** You generate an EDM or Entity data model from an already existing database.

3. **Model First** You create and model the database using visual tooling and create your own EDM which will be used by Entity framework to generate a database.

Microsoft (b)

2.10 Security

2.10.1 DOS

DOS Denial of service or DOS, is a cyber attack strategy where a client machine attempts to flood a host machine with incoming connections making it overload from the incoming requests.

DDOS DDOS or Distributed Denial of service is a method to deny access or take down remote servers. DDOS is just a DOS attack from many machines simultaneously. The purpose of making the attack come from multiple machines is to make it impossible to just block the connection like you can with a DOS attack from a single machine.

Wikipedia (d)

2.10.2 JWT

Json Web Token or JWT is a web standard for communicating claims between web systems. The token is sometimes encrypted with a public/private key asymmetrically, specifically a signature so the verification can be handled by comparing the private key hash of the json token. This allows for systems where a server can send a signed token with the claim ”you are admin” or something similar to a client, and the client can use the token to prove that they are an admin in later procedures.

Wikipedia (j)

2.11 Licensing

2.11.1 Software Licensing

A Software license is a legal instrument governing the use or redistribution of software. There are many types of software licensing, typically software licensing for proprietary software grants an end-user permission to use a product where normally the usage of the product would count as an infringement on the owners exclusive copyright. Typically this comes in the form of Eula's or End-user license agreements where the user just has to accept the agreement to be allowed to use the software. Proprietary license agreements do typically not allow distribution or copying of the product.

Wikipedia (u)

2.11.2 Open Source

Open source software is distributed via open source licensing that have very limited restrictions. Typically open source is made freely available for both redistribution and modification, with source code and blueprints freely available for anyone to use. Open source projects can also have restriction where anyone that uses the source code of the project to create extensions or further develop features, must keep the code they create open and free for everyone to use.
Open source projects often have a focus on mass collaboration, where anyone that can add value to the project can do so. This allows for innovation in the software development industries. Due to the

open nature of open source any goal oriented individuals or organisations can expand feature sets for existing software as they need it, without having to create proprietary software from scratch.

Wikipedia (u) Wikipedia (n)

2.12 .Net

In this section we will explain what .Net or dotNet is, and how it works.

dotNet is an open source software framework developed by Microsoft. dotNet has had many versions with varying levels of compatibility and features. dotNet is structured so that the programming language you are using is compiled to a common intermediate language which will then run on the dotNet runtime.

2.12.1 Versions

.Net Framework dotNet Framework is the oldest original release of the dotnet framework. dotNet Framework does not support cross-platform development and was developed for Windows only. The Final release of dotNet Framework was released in 2019.

.Net Standard dotNet standard is a specification of multiple dotNet API's that is meant to be uniformly available to multiple dotNet implementations and provide a unified ecosystem for dotNet.

.Net Core dotNet Core was developed as the successor to dotNet Framework and supports cross-platform development. dotNet Core 3.1 is the latest dotNet core LTS release. LTS releases of dotNet get 3 years of updates.

.Net 5 dotNet 5 is the newest version of dotnet core as of writing. Using dotNet 5 allows you to use the newest c# 9 language standard, however dotNet 5 is not a LTS release. All dotNet releases from dotNet 5 and forwards aim to combine the dotNet standard into the base dotNet implementations, removing the need for dotNet standard.

2.13 Code Standard

2.13.1 Style guide

A style guide, or naming convention is the way you are supposed to write code to make it easily readable. C# has its own coding conventions defined by Microsoft that developers of the language should use for easy readability. We follow most of these conventions, however both of the people working on this project were trained in Java Coding conventions and prefer them, so we do not follow the method naming convention which is supposed to be PascalCase, and not camelCase like we are using, this was not intentional but because we used resharper it automatically updated its standard to camelCase.

2.13.2 Linter

A Linter is a static code analysis tool that can detect errors, bugs, and stylistic errors and warn the programmer.

3 Methods And Materials

In this chapter we will look at the libraries, tools, frameworks, hardware, and development techniques that were used during this project.

3.1 Planning

User Interface Planning At the start of our project we made simple sketches to get a general idea of what we wanted the user interface to look like, and have something to get confirmation on from our thesis contact. We made simple drawing of each part of the application of what we thought the application would look like. An example of the what sketches look like is shown in figure 1, which is an example of the page where video is received and displayed on the website.

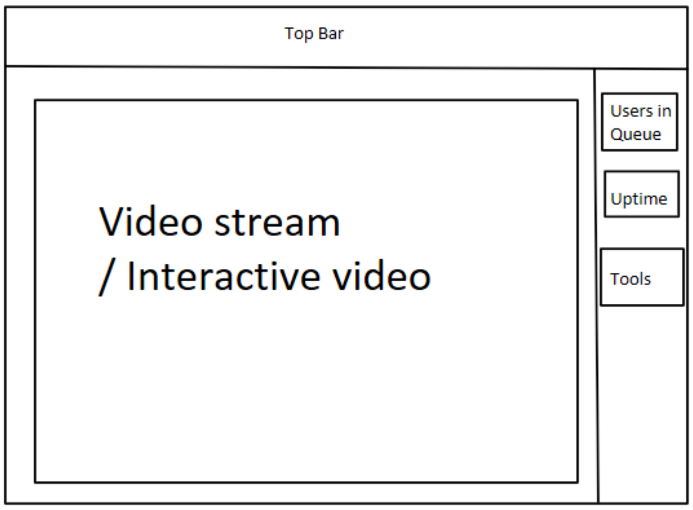


Figure 1: Image of UI planing example (image 1/10)

Technology Research We also had a general idea of what technologies we were gonna use in our project. We researched Signal R and WebRTC because we knew they were low latency or real-time web solutions. Signal R was the main reason we went for .Net for our development ecosystem because we thought it would give us a simple way to implement low latency communication with the front-end. We also did research on what protocols had the lowest latency's outside of the ones we already knew.

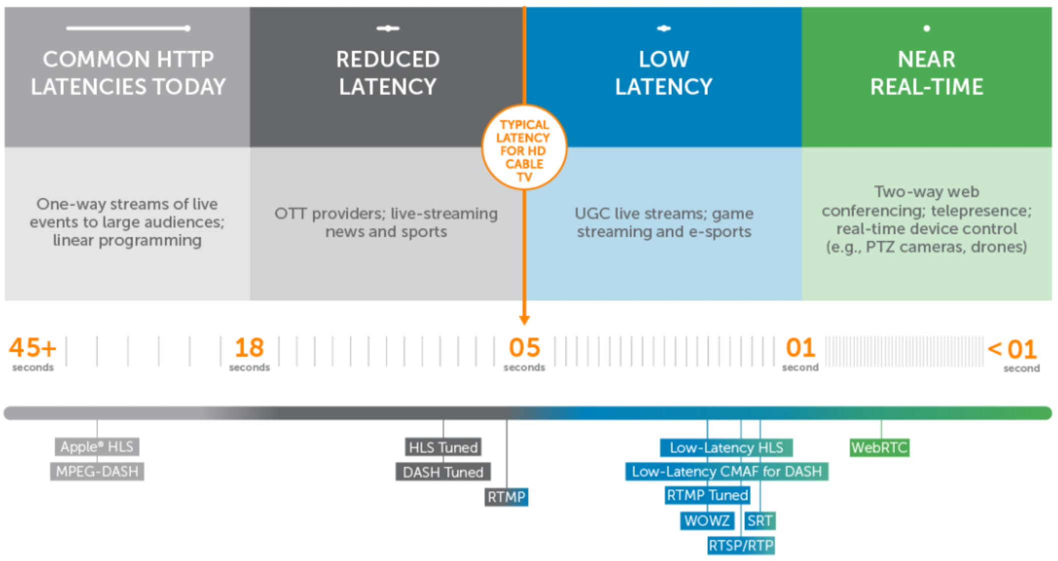


Figure 2: Diagram showing various protocols latency

Streaming Protocols: Everything You Need to Know (Update)

Agile Development We used scrum to plan issues for our sprints as explained in chapter 2.8.1. We had sprints with a duration of 1 week, with bi weekly meetings with our supervisor. We preformed sprint retrospective meetings reflecting on what we had done well and done bad after each sprint. For time management we decided to use Time-estimation for tracking and managing issues. We had sprint planning meetings where we planned what should be accomplished during the sprint.

3.2 Project Architecture

We made a plan for how the various instruments would communicate with each other and what data they would need to transfer. At the start of the project we decided we would go for a system where everything connected to a central backend.

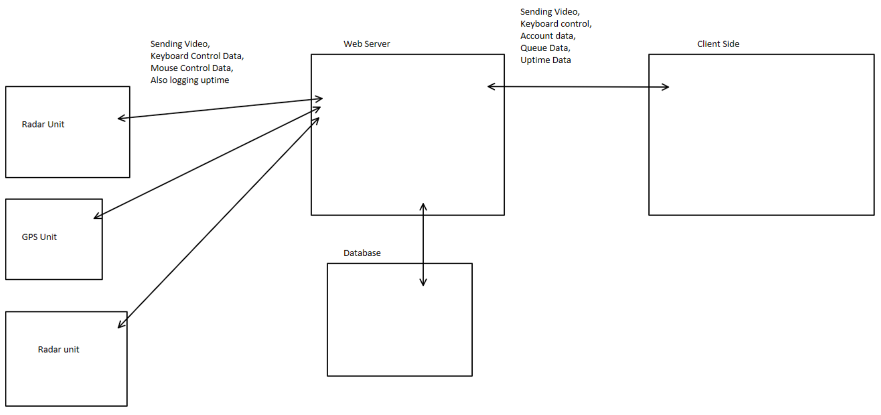


Figure 3: Sketch for planned server-client structure

3.3 Libraries And Frameworks

In this section we will look at the various libraries and frameworks we used during our project and the features they provide.

3.3.1 ASP.NET Core

ASP.NET core is a open source web framework developed by Microsoft to extend the .Net features to the web. ASP.NET supports both MVC, and API development, and has also added support for web assembly with Blazor. The following are development features in ASP.NET Core that we used in our project.

Middleware Middleware in aspnet is used to manage incoming http connections and preform actions on them as they pass along the appropriate pipeline. When an incoming connection is received it will be passed down from middleware to middleware until processed. Middelware is used to check if the connection is authorised and authenticated, then deny access to features. It can redirect incoming http connections to https or other routing actions. In our application we use the baseline middleware for redirection and check if the incoming connection is on the URL path for web-sockets before establishing the connection. Middleware also handles the mapping of endpoints in the application, like the various API paths that can exist in a ASP.NET controller.

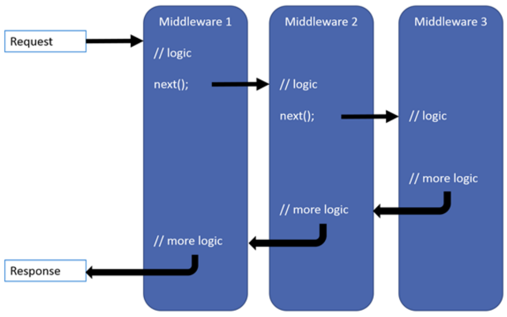


Figure 4: Image of an ASP.NET Middleware pipeline

Dependency Injection Dependency Injection is a design pattern used to implement Inversion of control. Inversion of control is a software engineering principle where you transfer the responsibility of the program flow to the framework you are using. This allows for better decoupling and Code quality in programs. There are three main types of Dependency injection

1. Constructor injection where dependencies are passed through the constructor of an object.
2. Setter injection where the class exposes a setter for a field in itself for the injector to inject.
3. Interface injection, an interface that will inject dependencies to classes passed to it. Interface injection also uses Setter injection.

Wikipedia (e)

Signal R Signal R is an open source software library developed by Microsoft for ASP.NET that aims to provide real-time communication. Signal R utilizes web-sockets to push data to connected clients allowing for fast and efficient communication. In web apps it is used to only update small parts of the web page instead of sending all the data it needs for the page repeatedly. Signal R also allows for the creation of hubs where clients can connect and communicate with one another.

Controller endpoints Controller endpoints are API implementations that allow you to specify HTTP Get/Put/Push requests in ASP.NET. Controllers are normally used in MVC or Model View Controller, which is a development method for separating user interface from logic in an ASP.NET Web application. Controllers are also used when creating an API that does not have a UI implementation, like in our project.

3.3.2 Blazor

Blazor is a relatively new open source web framework that was released in 2018 and is developed by Microsoft. It allows you to make web apps with C# in the browser. Blazor uses a component system where c#, HTML, and CSS is packaged into razor files. These razor files allow you to alter the HTML and CSS with c# code, as well as having it combined for use anywhere in the application, allowing for component nesting where a component can have other components inside them. Parameters for components can be set from the outside, and data can be cascaded through the entire component layer, making blazor excellent for clean code and code reuse with its versatile and easily modified components. Blazor has two main implementations that aim to provide different sets of pros and cons.

Blazor Server-Side Blazor Server-side apps are hosted in an ASP.NET Core server and all the logic is processed on the server and sent to the client. The client downloads the User Interface and gets updates over a Signal R connection. The clients are thin-clients because most of the processing load is handled by the server. This makes Blazor Server-side compatible with most browsers and devices, as well as requiring very little compute power from clients making it excellent for slower devices.

Blazor Client-Side Blazor Client-side apps are Web Assembly applications that get downloaded and run in the clients browser independently. This allows for less resources to be taken by any servers by requiring the client's computer to do processing instead, as well as creating completely offline applications that only need to be downloaded once and never need to communicate with a server. This makes it very good for reducing server load and makes it incredibly good for scaling.

ASP.NET hosted Our implementation of blazor uses a "third" method which is blazor web assembly, but hosted with a ASP.NET core backend. This is called ASP.NET hosted, and it is a middle ground of the client and server options.

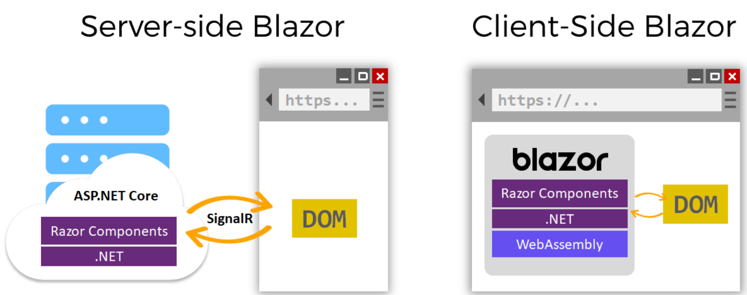


Figure 5: Blazor Hosting Models

3.3.3 OpenCv

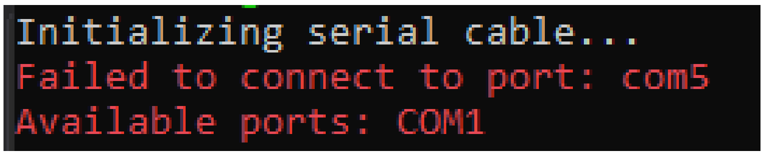
OpenCV stands for Open Source Computer Vision Library and is a open source library with focus on computer vision. The library has more than 2500 optimized algorithms for computer vision and machine learning.

OpenCV was used to access video devices on the remote system. Using OpenCV gave us access to media API's in Windows, OpenCV supports Linux API's too making it possible to run the code on a Linux system. In our specific instance the software was designed to be hosted on a Windows machine so the DSHOW API was used to access video devices. Frames are accessed one at a time using the built in "Read()" method in OpenCV, the frame is then encoded and compressed into a JPEG using the jpeg encoder in OpenCV.

3.3.4 Serial IO

Serial IO is a open source nuget package library developed by Microsoft for interfacing with serial ports on windows.

The library was used to send byte commands to the serial to USB Crestron cable to emulate mouse and keyboard activity. The library is capable of listing all available serial ports or what is called COM ports in windows. This feature is used to print available ports if an invalid one is specified in the config file.



```
Initializing serial cable...
Failed to connect to port: com5
Available ports: COM1
```

Figure 6: List of valid ports

3.4 Hardware

3.4.1 Crestron Cable

To control the remote device the Crestron CBL-USB-RS232MK-6 cable was used. The cable bypasses DRM restrictions on the controlled machine by emulating a HID. In figure 7 you can see on the left the controlling machine connected to the Crestron cable using a USB to serial adapter. On the right is the controlled machine representing a stand in for an actual maritime device and it is connected to the USB end of the Crestron cable.

To make the Crestron cable reproduce key and mouse commands given to it an API for the device had to be made. The Crestron cable works on a byte command basis where each keyboard key has a make and break byte command. The make command corresponds to a down key press and the break releasing the key. The solution for making a simple API for the device was to store all the byte commands for all the supported keys in a CSV file and read it into the program on start up. The values for the make and break commands of a key can then be stored in a HashMap allowing for instant look up (O(1) time complexity).

Serial ports have usually quite slow transfer speeds all depending on the so called baud rate which is just a measurement of how much data can be transferred per unit of time similar to bitrate. The Crestron cable operates at a reasonably high baud rate but is way slower than the speed of a modern CPU. If the CPU were to send data at the speed it wanted the Crestron cable would receive to much data at a time and be unable to reproduce the given command. To prevent this the Crestron returns bytes when it is ready to execute the next command. Because the serial cable is slow the Crestron API developed has a queue of bytes to execute. Then a thread dedicated to taking one command at a time from that queue and waiting for a response is used to send commands to the cable preventing waits on the main program thread. This dedicated thread also ensures that the Crestron commands are sent to the cable as quickly as possible after receiving the response byte from the cable.

Due to the limited execution speed of the cable or baud rate the cable comes with two magnitudes for mouse movements. The mouse commands have 4 directions up, down, left and right and moving

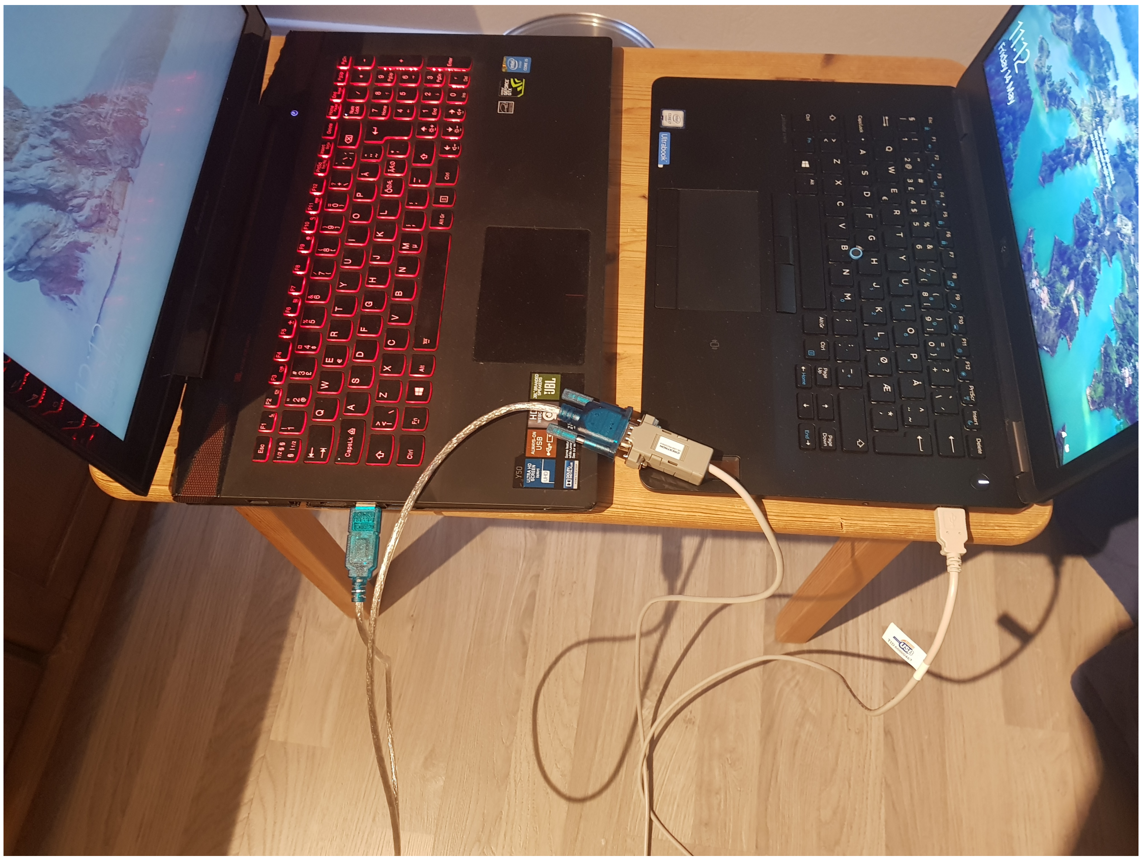
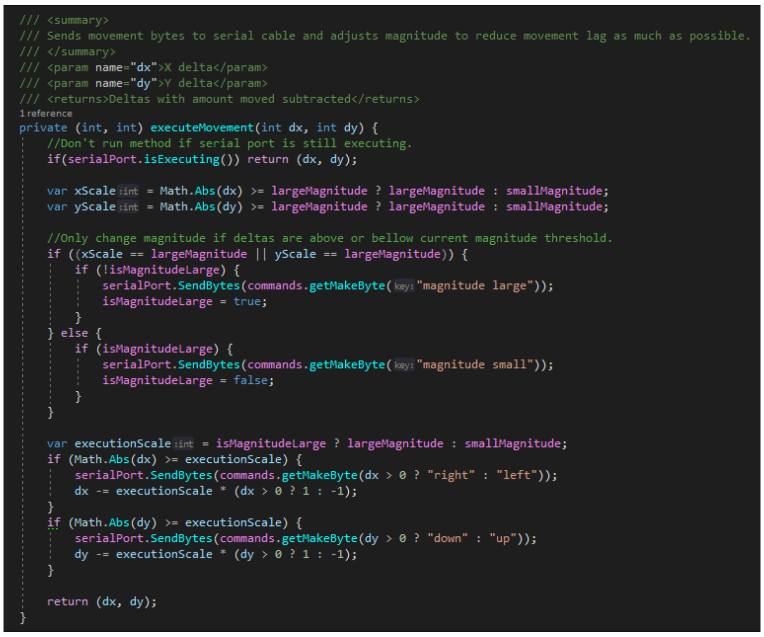


Figure 7: How Crestron cable is used

in one of them using the small magnitude only moves the cursor one pixel at a time. With the execution speed of the Crestron this is to slow for replicating human movement speeds. While the large magnitude moves the cursor approximately 20 pixels. This large magnitude mode comes with the disadvantage of being inaccurate, so a mix of the two magnitudes has to be used. The challenge of mixing the large and small magnitude into the most efficient combination of movements was solved using a dedicated algorithm.

The cursor movement algorithm takes x and y values in, x representing up and down and y left and right. Because the Crestron cable is slow at executing commands it is possible for the algorithm to receive new movement values as it is executing. To include those new values the algorithm continually sums up the desired movement and compares it with the actual distance moved. The algorithm then checks if the sum of the movement is greater than 20 and engages large magnitude mode. If the sum of the movement is less than 20 precision is required and the last distance is moved using the small magnitude mode.



```
/// <summary>
/// Sends movement bytes to serial cable and adjusts magnitude to reduce movement lag as much as possible.
/// </summary>
/// <param name="dx">X delta</param>
/// <param name="dy">Y delta</param>
/// <returns>Deltas with amount moved subtracted</returns>
1 reference
private (int, int) executeMovement(int dx, int dy) {
    //Don't run method if serial port is still executing.
    if(serialPort.isExecuting()) return (dx, dy);

    var xScale:int = Math.Abs(dx) >= largeMagnitude ? largeMagnitude : smallMagnitude;
    var yScale:int = Math.Abs(dy) >= largeMagnitude ? largeMagnitude : smallMagnitude;

    //Only change magnitude if deltas are above or bellow current magnitude threshold.
    if ((xScale == largeMagnitude || yScale == largeMagnitude)) {
        if (!isMagnitudeLarge) {
            serialPort.SendBytes(commands.getMakeByte(key:"magnitude large"));
            isMagnitudeLarge = true;
        }
    } else {
        if (isMagnitudeLarge) {
            serialPort.SendBytes(commands.getMakeByte(key:"magnitude small"));
            isMagnitudeLarge = false;
        }
    }

    var executionScale:int = isMagnitudeLarge ? largeMagnitude : smallMagnitude;
    if (Math.Abs(dx) >= executionScale) {
        serialPort.SendBytes(commands.getMakeByte(dx > 0 ? "right" : "left"));
        dx -= executionScale * (dx > 0 ? 1 : -1);
    }
    if (Math.Abs(dy) >= executionScale) {
        serialPort.SendBytes(commands.getMakeByte(dy > 0 ? "down" : "up"));
        dy -= executionScale * (dy > 0 ? 1 : -1);
    }

    return (dx, dy);
}
```

Figure 8: Method for determining what commands to send based on deltas

3.4.2 Video Capture Card

To get the video output of the remote device a capture card capable of capturing the display output type is used. The StarTech USB3HDCAP capture card was used during development to get video output from the controlled laptop. As demonstrated in figure 9 the video output from the controlled laptop is sent to the capture card using HDMI. The capture card is then connected to the controlling laptop using USB.

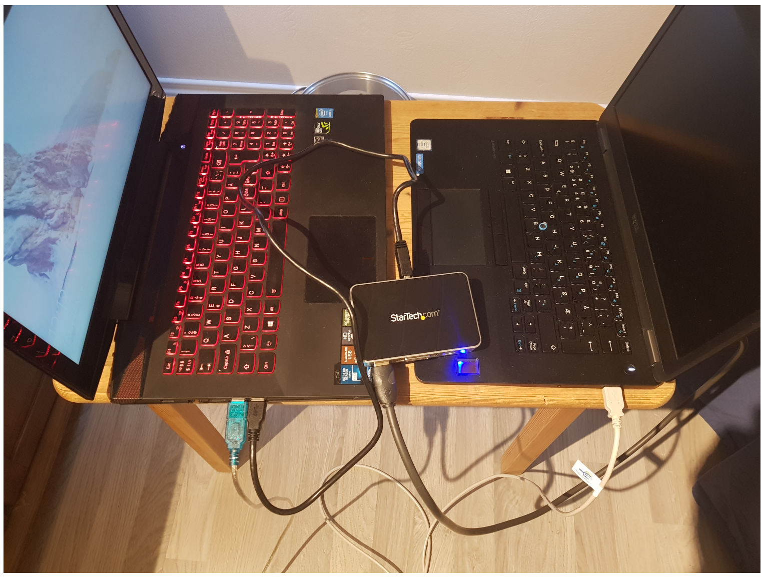


Figure 9: How capture card is used

Windows does not automatically install the required driver for the capture card. This requires a manual install of the drivers from StarTech's website to make the capture card work. The capture card also only has Windows drivers making it only usable on a Windows machine.

The capture card only simulates a display, so to the controlled laptop it is as if another display was connected. To make the capture card capture what is on the laptop screen the output behaviour in Windows must be set to mirror the display. Similarly on the radar devices we tested on some did not output video to the capture card because to it another monitor was connected and it was not configured to mirror the output. The simple solution is to use a HDMI splitter from the main video output and connect on to the actual monitor and the other to the capture card. We did not test this solution as we only had time to test the solution on real maritime equipment once and it was that one time the issue became apparent.

3.4.3 WebCam

Webcams are used to capture the outside environment or to capture maritime devices with no conventional capturable output. The webcam can be pointed at the screen of the device allowing for remote viewing. The output of the webcam is accessed the same way the output from the capture card is because to the software it is just another video device. Multiple webcams are handled by opening a new MJPEG stream for each new device.

3.5 Video Streaming

3.5.1 MJPEG

Streaming of MJPEG was done by manually coding a HTTP rest API that would respond to HTTP requests. The streaming class hosts a server listening for HTTP request on a specified port (this allows for multiple streams on the same IP address). When the class receives a request the client is added to a client list and all new MJPEG frames will be sent to that client as long as it is connected to the server. Because the MJPEG server is hosted at the location of the remote device opening ports for clients to connect to the video feed in necessary. Ports were forwarded using settings in home routers when testing. MJPEG server code is based on code by Ragheed Al-Tayeb.

The stream is accessed by entering the URL and port of the device the stream is hosted on. To embed the stream into our website a HTML image element was used with the source set to the URL and port where the stream is hosted.

Motion JPEG Streaming Server

3.5.2 MJPEG Testing

Testing bandwidth usage of the MJPEG server was done on a variety of settings, see Appendix C for settings used. The testing methodology was to test the steaming under the worst possible conditions bandwidth wise by playing a video of static noise. Playing static noise should yield the worst compression results possible for the settings used and thus show the theoretical maximum bandwidth those settings will ever use. The bandwidth usage was observed using resource monitor in windows and seeing under the network section how much bandwidth the process "Remote_Server" was using. Disclaimer windows reports network usage in MiB/s or just B/sec. To convert B/sec to Mbps the number reported in resource monitor was divided by 1 million and converted from MiB to Mbits using google.

Video used when testing

Testing latency of the system was preformed by only including the latency from the code and browser. Excluding network delay gives more consistent results when testing and reduces unwanted variables. The remote server and web server were both hosted on the same machine connected to a different machine being controlled and web site accessed using the same machine. This should eliminate all network latency and all network traffic should go through loopback. Other preconditions for the test is all machines involved in testing do not suffer from any performance bottlenecking meaning CPU usage is at a reasonable level, there is enough RAM, etc. The machine used to preform the test should be able to produce 60 fps or higher.

To measure the latency the screen of the test machine used for testing was recorded using OBS at 60fps! Then the device was controlled through the website, on the controlled device notepad was opened. Using the virtual keyboard on the website the numbers 1,2 … 9,0 was pressed. The screen recording was then opened in VLC. Latency was measured by increments through individual frames using the E key. When a virtual key was pressed the number of frames between the press and seeing the number appear in notepad was counted. The number of frames for each key is counted and averaged. Millisecond latency is calculated using this equation: $ms = (1000/fps) * (f_{\text{avg}})$. Fps is the frame rate of the recording and f_{avg} is the average number of frames between virtual key being pressed and seeing the number in notepad.

3.5.3 FFMPEG

FFMEG was used for testing video streaming of h264 encoded video. Different video streaming protocols were tested with different tuning.

Codec used in testing was the libx264 codec encoding video from StarTech USB3HDCAP capture card at 60 frames per second. Streaming protocols tested: udp, rtmp, rtp, mpegts, rtsp and rtsp_transport. Stream modifications used to try and reduce latency: zerolatency tune, ultrafast preset and reduced buffer size.

For receiving the video stream FFPLAY was used with settings compatible with the protocol used when testing. Most of the settings tried is from this streaming guide from FFMPEG.

StreamingGuide

3.6 Development Tools

In this section we will look at some of the tools used for developing our application.

3.6.1 Jira

Jira is an issue tracking software that is used to help development teams manage work, and tracking progress. Jira uses scrum and also has charts for helping you visualize progress based on what was completed in the current sprint.

3.6.2 Visual Studio

Visual Studio is an integrated development environment developed by Microsoft. Visual studio structures code and projects in solutions that contain many projects. A project is a specific application or library, for example a console application would be its own project in the solution. Visual studio is built to be extendable where it is not made for any single language, but uses packages or extensions to add support for various programming languages. Anyone can create an extension, for visual studio making it modular and customizable. Microsoft has added support for their own languages, c# and c++, as well as database integration, and project templates for ASP.NET and Blazor projects.

Visual Studio was almost exclusively used for all programming and managing project file structure in conjunction with ReSharper for better IntelliSense. Building and running developed code was done in Visual Studio. Installing Nuget packages was also done through Visual Studio. Debugging code and profiling performance was done in Visual Studio.

3.6.3 Netlimiter4

Netlimiter4 is an Internet traffic management tool for windows that allows you to set bandwidth quotas, or throttle the internet speeds of specific programs on a specific program on the computer. We used netlimiter4 to simulate low speed environments like something running over 3g or 4g cellular networks, as well as tracking the bandwidth used by our systems, as those are quite important aspects of our project.

3.6.4 Git

Git is a version control system for tracking file changes and coordinating work between multiple collaborating programmers. Git functions by having both remote repositories of the code, as well as storing things in repositories locally. This makes git a distributed system where users can push and pull changes from what is called the remote or origin of the system. In our project we use git as our version control, allowing us to do refactoring in separate branches, and working on the same code base at the same time.

3.6.5 Github

GitHub is a hosting service for git repositories. GitHub uses git for its version control and source code management, but it also has its own features like access control and continuous integration. GitHub actions help with Continuous integration and continuous development, actions allow you to test build and setup pipelines for your repositories.

3.6.6 NuGet

NuGet is a package managment system developed by Microsoft, originally as an extension for Visual studio. NuGet is no longer exclusive to Visual studio and has support for independent usage. The package manager is directly integrated into visual studio making it very easy to import packages into your projects, as well as storing information about the dependencies in the project files so it will be automatically pulled by other developers using your projects.

3.6.7 Resharper

Resharper is a code analysis tool and linter developed by JetBrains. Resharper is used to detect code style, and suggest name changes to follow the style guide selected in the options of the program. Refactoring is also made very simple in Resharper where you can change names in the code and it will automatically update across all files that use this name. Rehsarper also suggests code improvements and simplifications for code that follows simple patterns like if statements or switch cases.

3.6.8 Visual Paradigm

Visual Paradigm is a UML case tool used to create diagrams. Visual Paradigm is not only used for UML, it also has support for diagrams relating to business logic, Entity relationship diagrams, scrum diagrams, networking diagrams, and much more. In our project we used visual paradigm to create our UML diagrams, and network diagrams.

3.7 User Input

3.7.1 Pointer Lock

To capture data from the mouse, and removing it from the screen so you could only see the mouse of the device you are controlling we used the Pointer Lock API. The API allows for capturing data about the mouse and its movements.

mozzila

3.7.2 Keyboard events

To capture keyboard input we used the built in event classes in Blazor called KeyboardEventArgs, which allow you to detect keyboard input in the browser from the users keyboard.

Microsoft (c)

3.8 DevOps

DevOps is the combination of Software Development (Dev) and IT operations (Ops), DevOps is used to shorten development Lifecycles and provide Continuous delivery and integration. DevOps is complementary to agile development methods allowing you to create versions of your software for each sprint.

3.8.1 Docker

Docker is software for OS-Level virtualization, which means the main operating system allows multiple kernel level isolated user-space instances. Docker works by creating containers that are isolated from each other and bundle all the dependencies and software in the container so it can run on any machine that supports docker and virtualization. Containerizing your projects also removes many issues with setup for other users that don't know what technologies and dependencies the project uses. The Cloud and hosting in the cloud is becoming more popular and cloud service providers like Amazon Web Services and Azure support hosting of docker containers. In our project we have containerized our ASP.NET server allowing for easy hosting and setup, but we have not containerized the remote server due to the limitations in Usb communication to the inside of a container.

Wikipedia (o)

3.8.2 Dockerhub

Dockerhub provides online repositories for Docker containers, allowing for sharing and distribution of docker images. Dockerhub allows you to create your own repositories for projects and setting it up with automatic building of images, Dockerhub can automatically build directly from GitHub or Bit-bucket, or it can automatically build from releases. Dockerhub also supports automatic testing, image security scanning, and webhooks. Our repository is setup so it will build the latest release of the master branch and the development branch of our GitHub repository continuously as they get new commits, as well as building version releases with a tag matching the layout v1.0.0.

3.9 Testing

3.9.1 Unit Testing

Unit Testing is the testing of a specific unit or class to make sure its procedures function as intended. Unit testing have many benefits when making software, as it helps reduced unintended behavior in code, and reduces the places where you need to look when an error occurs in your software. We used unit testing to test our communication classes making sure their base functionalities work as intended, as most of our implementation run on top of these classes and it could be the root cause of many errors.

3.9.2 Integration Testing

Integration testing is testing of a group of units together and checking if they interact as intended. In our system we test the interaction between different parts of our network system and how they interact, and that the data received and sent all happen as intended.

3.9.3 Acceptance Testing

User Acceptance Testing is a test usually preformed by the client or product owner that is used to verify that the the features the client expects are working and present in the system. A User Acceptance test typically contain multiple test cases that can be preformed by the user and verified if the expected and correct outcomes were achieved, allowing an objective way for the user to test the system, and see what the developer intended the functionally to do in a written document. Our acceptance test has tests for the user interface and latency functionalities of the web application.

3.9.4 Internal Testing

Internal testing aims to test internal systems where all details are visible. Internal testing tests systems that are not interesting for the client or customer to know about or see, but still have an impact on the project. In our project internal testing checks for compression results.

3.9.5 Latency Testing

We preformed a latency test meant to simulate realistic usage of the system at various bandwidth speeds below what is needed for the connection. The compression and frame rate used was at a constant, 40% quality, with 30 fps from a Full HD Video Capture card, when transferring the full data and quality of this video, with the crestron the connection will use 2.6 MB/s for 3 video streams. The person controlling the system remotely and measuring the latency was about 20km away from the server. This test was performed to see how the system would react if the speed needed to transfer the required video data was too low for the system. The latency is measured from when the user sends input to when the resulting action showed for the user, the test also had 2 spectators, meaning the video data used was tripled on the current implementation of the system.

3.10 License Choice

3.10.1 MIT

We chose the MIT License because it was the most permissive software license we could find and the project is publicly funded so there should be no copyright or restriction on the software created as specified in section 2.11.2. The MIT License permits usage, copy, modification, merging, publishing, distributing, sublicensing, and selling of the source code, making it essentially free for everyone to use.

3.11 Programming Languages

3.11.1 C#

C# was used for most of the code in the system. The Asp.Net Server is written in c# and all logic on the client except the pointer Lock API logic, is written in C#.

3.11.2 HTML

Hyper Text Markup Language is the most used language for structuring the layout of a web page. For this project the HTML that was built into Blazor was used which comes with the ability to generate HTML using code. This code generated HTML was used to display list items and keys in the virtual keyboard. HTML was used in conjunction with C# code, CSS and JS.

3.11.3 CSS

Cascading Style Sheet was used to style the HTML used on the website. It was used for user feedback, using changing pointer styles to indicate to the user intractability and color change on buttons to give user feedback on actions.

Blazor comes with bootstrap included with CSS which we made use of using many of the premade style classes included with bootstrap. The included open iconic icons were used for home button, options button, account button, etc.

3.11.4 JS

JavaScript is the most common language used to script logic into web pages and has for most of web history been the only option. New languages like web assembly has allowed for assembly code to run in web browser making nearly any programming language able to run in a browser.

For most of the project any web page logic was coded in C# but certain browser API's are only available in JS making the use of the language necessary. The one instance where JS had to be used was when using the pointer lock API available in Firefox and Chrome. To control the device using a mouse there was a need to lock the pointer in place preventing mouse movement form escaping the browser. This was achieved using the pointer lock API and was called using JS.

4 Results

In this chapter we will talk about what we have created and achieved in this project.

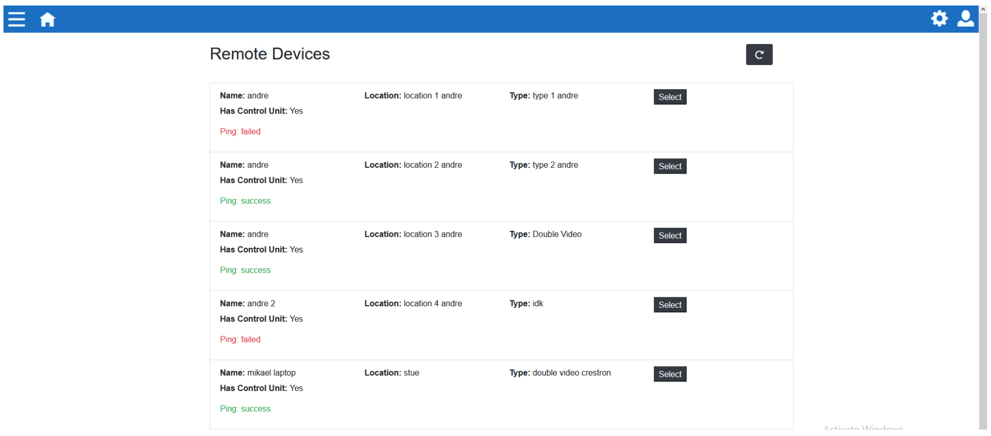
4.1 Source Code

The source code repository for this project can be found here: https://github.com/Mikael-og-Andre/Distributed-Instrument-Cluster

4.2 Features

4.2.1 Application pages

The User Interface currently consists of two main pages. The remote device selection page with a list of remote devices, and the device control page where you control the device. We also created a login and registration pages, but they are not coupled with the backend so they are disconnected. Both the Login and Registration pages have fully functional form validation with error messages. UI development had focus on the Design principles specified in section 2.8 in order to make the UI as user friendly as possible. The top bar is currently not connected to anything, and will redirect you to the remote device selection page.



Remote Devices

Name: andre **Has Control Unit:** Yes Ping: failed	**Location:** location 1 andre	**Type:** type 1 andre	Select
Name: andre **Has Control Unit:** Yes Ping: success	**Location:** location 2 andre	**Type:** type 2 andre	Select
Name: andre **Has Control Unit:** Yes Ping: success	**Location:** location 3 andre	**Type:** Double Video	Select
Name: andre 2 **Has Control Unit:** Yes Ping: failed	**Location:** location 4 andre	**Type:** idk	Select
Name: mikael laptop **Has Control Unit:** Yes Ping: success	**Location:** stue	**Type:** double video crestron	Select

Figure 10: Image of device selection page UI

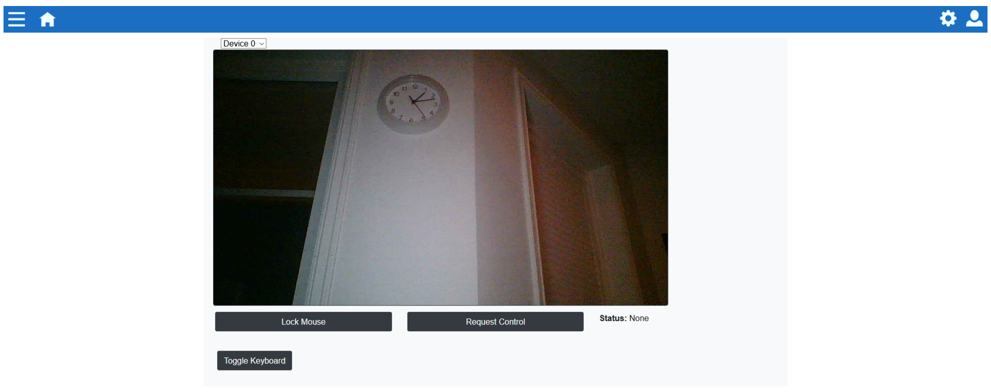
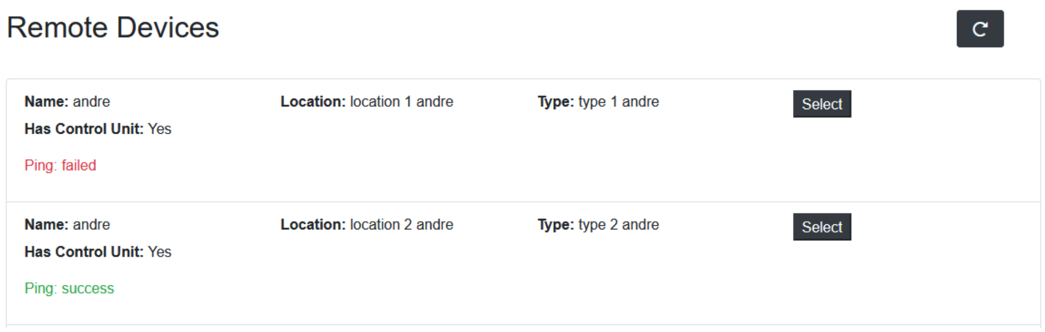


Figure 11: Image of device control page UI

4.2.2 Device Selection

The Web application will receive data about what remote servers exist from the ASP.NET server and will display them as a list of devices, with information about the name, location, and type, and if the ASP.NET Server was able to reach the server endpoint with an ICMP Ping request, as explained in section 2.4. You can also get a refreshed list from the server with the refresh button, showing a loading text while the data is being fetched. An example of 2 devices is shown in figure 12. If you want to control or view video for the device you can press the select button, and it will bring you to the Video and control page for the device.


Remote Devices

Name: andre
Has Control Unit: Yes
Ping: failed
Location: location 1 andre
Type: type 1 andre
Select

Name: andre
Has Control Unit: Yes
Ping: success
Location: location 2 andre
Type: type 2 andre
Select

Figure 12: Image of device selection

4.2.3 Multi Video

When on a video and control page there is a drop-down selector allowing you to swap between the different video streams, the selected Video device will be displayed on the screen, or an error message telling you the stream is unreachable. The multiple videos streams are intended so you can hookup a camera or something similar as well as the maritime device to the remote server so you can directly look at the weather or another thing in the environment, like a boat that is giving you the data on the other screen. The app theoretically supports as many video devices you are able to connect to the remote server. An example of selecting a video stream is shown in figure 13.

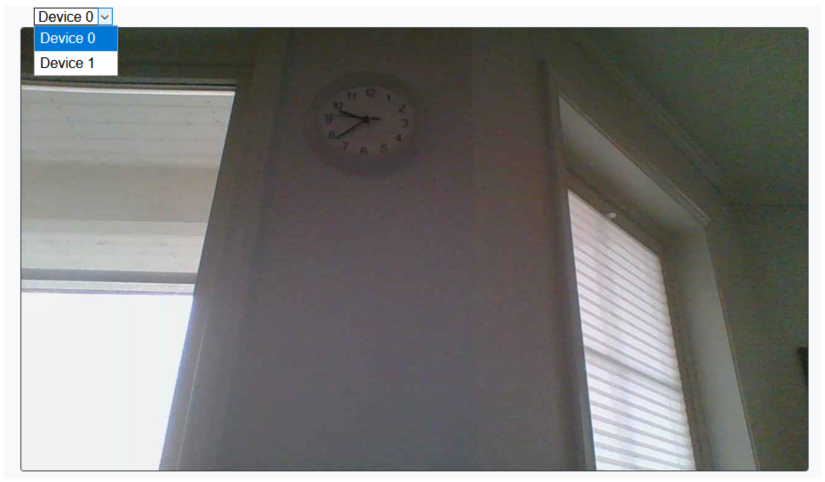


Figure 13: Example of Video Selection

4.2.4 Device control

On the video and control page you can connect to the Remote Server's Crestron by clicking the request control button. A status of the connection will be displayed to the right of the button, updating the user about the current state of the connection. The status will show what position in the queue the user is currently in for control of the unit, it also displays if the device is being requested, and if you are currently in control. The status will reflect the states shown in the Crestron Web-socket state diagram shown in figure 21. The Lock Mouse button will call the Mouse pointer Lock API and lock the mouse from moving. Mouse movements made while the mouse is locked will all be detected and sent as commands to the Remote server. Locking also captures keyboard key presses, allowing you to remote control the device using normal mouse movement and keyboard key presses. You can unlock the mouse by pressing ESC. An example of the displayed status is shown in figure 14. Several keys will remove the mouse lock API, so you can not use the computer's own keyboard for making special key strokes like using the ESC key, for this use the virtual keyboard as it does not rely on the lock API.

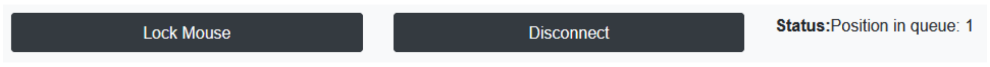


Figure 14: Example of a Control status message

4.2.5 Virtual Keyboard

The virtual keyboard allows for key combinations that would not be possible to do in a browser to be sent to the remote device. The well know alt+F4 would not be possible due to the browser closing on the client side when preforming it. Ctrl+alt+del would also not be possible assuming the client is connected to the website on a windows machine. Using the virtual keyboard and right clicking to hold buttons down any combination of key presses is possible. The virtual keyboard is able to execute all keys on a full 104 key keyboard. Due to limitations with the Creston cable one key on the Nordic keyboard layout is not possible to reproduce, the IntlBackslash key ("<" next to left shift). This is because the Crestron cable only emulates the keycodes for key on American keyboard.

Because the Crestron cable contains the layout made for the virtual keyboard mimics the American one but can be easily changed by loading a different json file containing values for a Nordic layout. Infact when keys like "[" (see figure 15) pressed the Nordic letter "å" appears on the controlled windows machine because the keyboard layout on it is set to Nordic. To the machine the bytes for "[" is interpreted as an "å" so a Nordic keyboard layout can be displayed on the website assuming layout of the target machine is set to Nordic. This was not done because it was assumed the layout in the radar machines would not be Nordic and could not be tested due to Covid-19 limiting the testing possibilities on real maritime equipment.

Customization The virtual keyboard is highly customizable because the layout of the keyboard is stored in a json file. The json file can be easily modified to show a different keyboard layout. The json file is stored in the root of the web servers file structure and is accessed using simple http get request. Those requests can be modified to specify a keyboard layout allowing for easy switching of keyboard layouts from a code logic perspective.

4.2.6 Device Support

The current implementation of the Remote Server relies on the crestron cable for sending Mouse and Keyboard commands. The use of the crestron cable makes the system able to control any system that supports the USB HID 1.1 that the crestron cable uses. This makes the list of supported devices, almost any modern Windows and Linux computer, as long as it does not restrict the USB communication with the crestron cabel. Devices that are only video based and do not have port for video capture can also be setup with normal cameras for recording the screen.

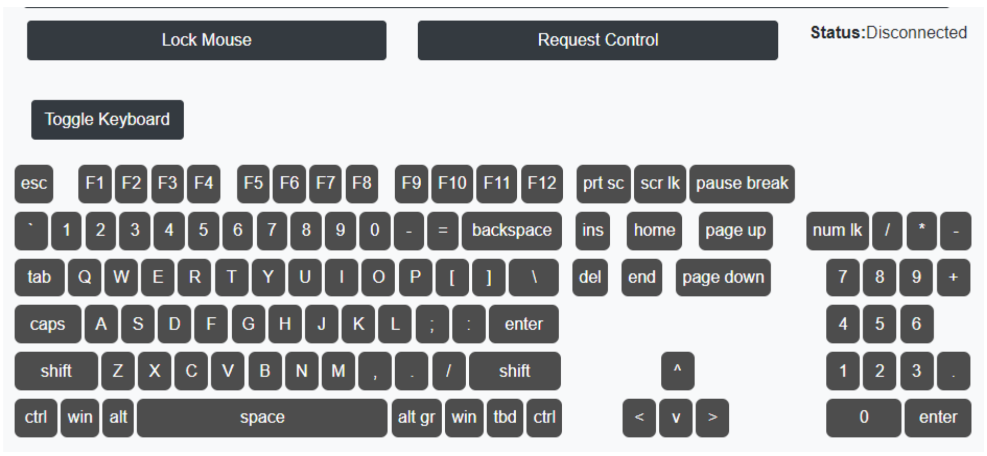

Lock Mouse
Request Control
Status:Disconnected
Toggle Keyboard
esc F1 F2 F3 F4 F5 F6 F7 F8 F9 F10 F11 F12 prt sc scr lk pause break
` 1 2 3 4 5 6 7 8 9 0 - = backspace ins home page up num lk / * -
tab Q W E R T Y U I O P [ ] \ del end page down 7 8 9 +
caps A S D F G H J K L ; : enter 4 5 6
shift Z X C V B N M , . / shift ^ 1 2 3 .
ctrl win alt space alt gr win tbd ctrl < v > 0 enter

Figure 15: Virtual keyboard on website

4.3 Development

In the later stages of development we started having our backlog be prioritized by our product owner contact, allowing him to make decisions about what should and should not be prioritized. We also implemented a CI/CD pipeline for docker. We started utilizing extreme programming principles like pair programming, see section 2.8.4, to increase the effectiveness of our team development process.

4.4 Remote Server

4.4.1 Bandwidth usage

The current solution is able to hit the bandwidth target of what a 4G connection is capable of. However this is under ideal land based 4G connection speeds. Parameters can be tweaked to hit lower bandwidth but comes at the cost of video quality and responsiveness. The result of different parameters can be seen in figure 16. The test was preformed using a video of static noise on the controlled machine something that in theory should be the worst possible conditions for the JPEG compression algorithm to compress. The testing done seem to indicate that this is the way to use as much bandwidth as possible so the results of the test should be worst case and real conditions should use less bandwidth.

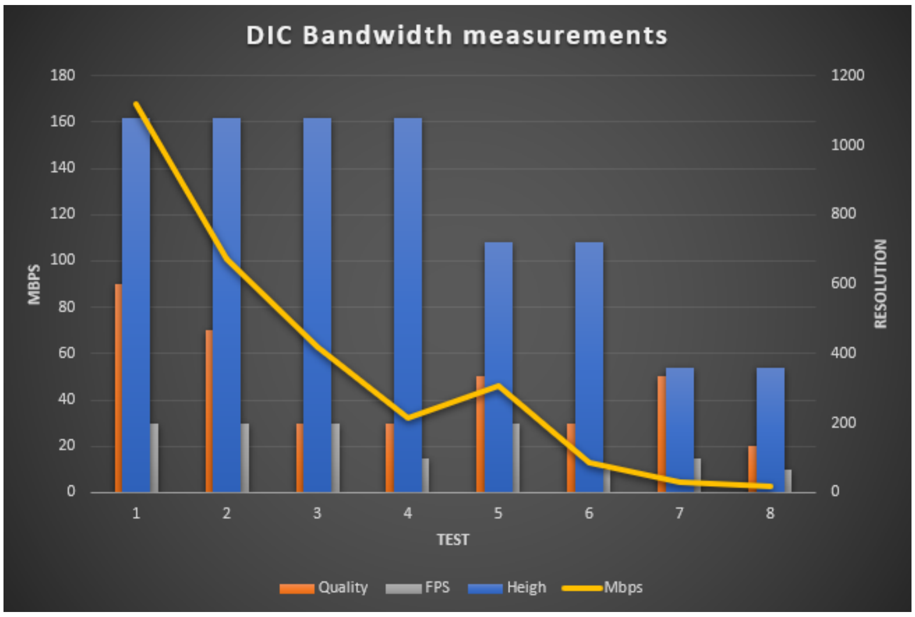


Figure 16: Results from bandwidth test

Height is the pixel height of a 16:9 ratio frame, quality is the quality on the JPEG compression (higher is better), fps is frames per second or the sampling rate, Mbps shows the Mega bits per second used of network bandwidth.

Test	Width	Heigh	Quality	FPS	Mbps
Test1	1920	1080	90	30	168
Test2	1920	1080	70	30	101
Test3	1920	1080	30	30	63
Test4	1920	1080	30	15	32
Test5	1280	720	50	30	46
Test6	1280	720	30	10	13
Test7	480	360	50	15	4.2
Test8	480	360	20	10	2.5

Table 1: Table of bandwidth test results

4.4.2 Stability

We preformed a test run of a the remote server code, having it run for as long as possible with continuous connections and disconnections from the control system and video access from our testing. As of writing it had been running for 1 week continuously with netlimiter4 running throttling the Network speed to 1MB/S.

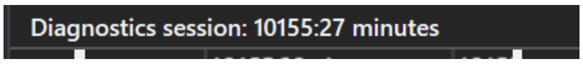

Diagnostics session: 10155:27 minutes

Figure 17: Screenshot of debug session timer

The remote server was running in the visual studio 2019 debug environment while testing. The computer used during the test was a Laptop running windows 10 PRO, with an i7-6600u CPU and integrated graphics with 16GB of ram. The video devices used during the testing was a Logitech webcam, and the laptops integrated webcam. The crestron cable was not used during this test, but the output and functionality was verified with an interface simulating a crestron.

4.4.3 Compression

Compressing the MJPEG stream at 40 quality seemed to yield a good balance between bandwidth usage and compression artifacts. The compression settings can be adjusted to achieve better quality with higher resolutions and higher fps, or the resolution, quality and fps can be reduced to get lower bandwidth usage. Shown in figure 18 is the uncompressed ground truth FHD image and in figure 19 is the image compressed to FHD using quality 40.

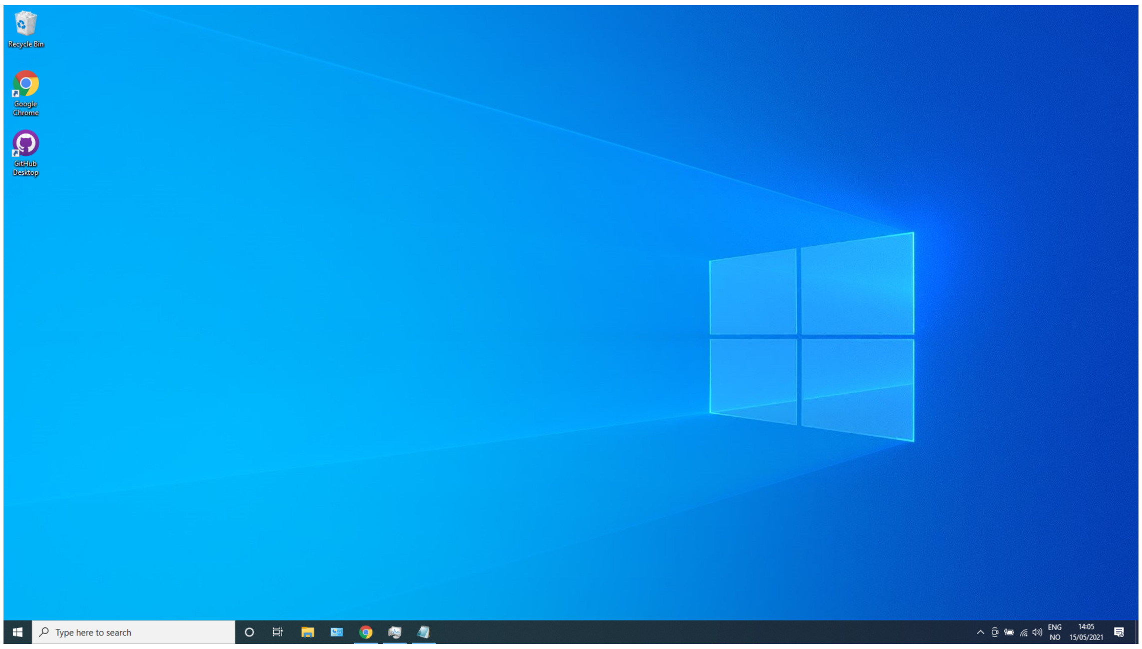


Figure 18: Desktop screenshot uncompressed

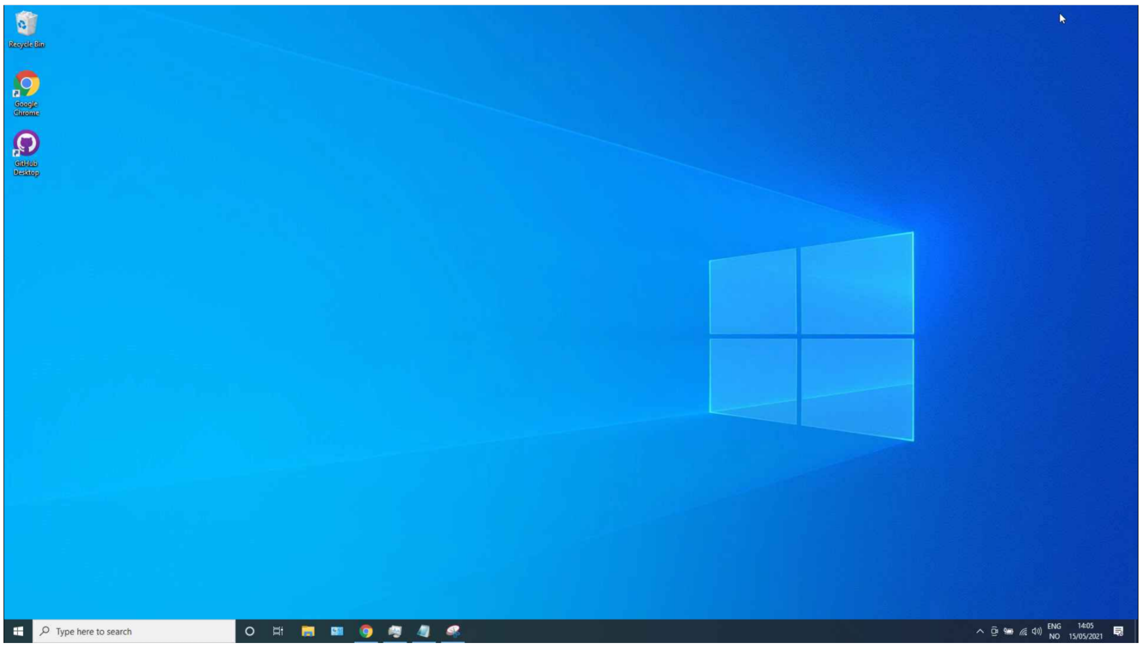


Figure 19: Desktop screenshot compressed at 40 quality

To highlight where the compression algorithm is making compromises a zoomed in view of the recycle bin icon is show in figure 20 showing the uncompressed image on the left and the compressed on the right. It is easy to see the degradation in quality particularly around text, which is something JPEG compression performs poorly on.

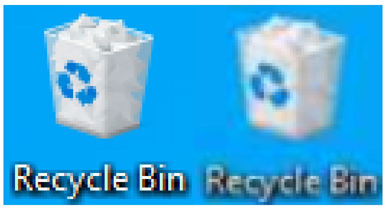


Figure 20: Zoomed in view of recycle bin uncompressed and compressed at 40 quality

4.5 ASP.NET Server

4.5.1 Control Management

The ASP.NET server manages the queue for users trying to control the Crestron on the remote server, so that only one user can send commands at a time, the server will disconnect any user that does not send any commands in 2 minutes time and gives the control to the next user in the queue. The implementation happens over web-socket so the server can not tell if the user disconnects if the user side web-socket does not specifically send a disconnect message to the server, making the time based disconnect crucial. The server will also manage the connection to the remote server, if the connection goes down, it will try to reconnect when necessary, and disconnect users if it is not possible to reach the remote server. The current implementation is stable and handle many users, see example of state diagram in figure 21. The implementation was created with web-sockets due to the reduced latency, and duplex communication, see section 2.6.3.

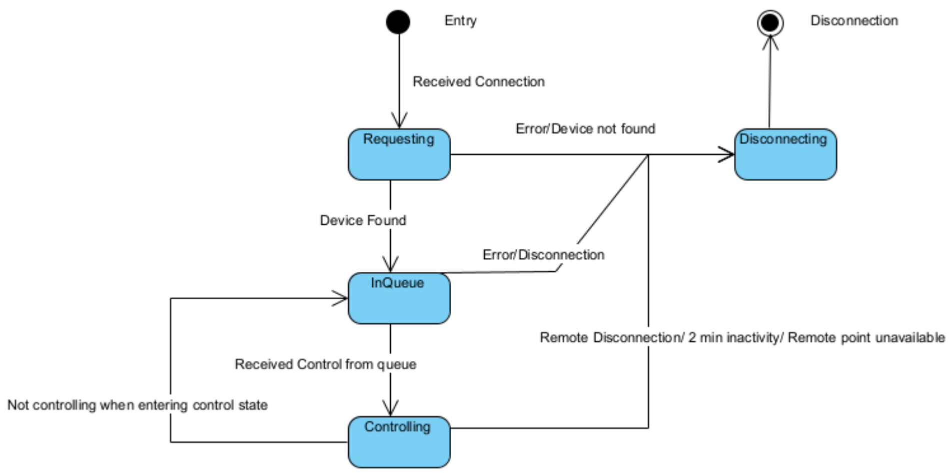

Entry
Disconnection
Received Connection
Requesting
Error/Device not found
Disconnecting
Device Found
Error/Disconnection
InQueue
Remote Disconnection/ 2 min inactivity/ Remote point unavailable
Received Control from queue
Not controlling when entering control state
Controlling

Figure 21: State Diagram for Crestron Websocket connections

4.5.2 Remote Devices

Remote Devices information is defined on the ASP.NET Server in a Json file and sent to all connecting clients when they go to the Remote Device selection page. Remote Devices stored and loaded from a Json file, where Ip, and other metadata is defined about the remote device, including what the video port for the remote device is. The ports of the video streams are incrementally generated from what we call the base port, if the base port is 8080, and there are 5 devices, the 5th device will be on port 8084, because there are 4 additional devices, 8080 + 4 = 8084. If the device does not have a Crestron connection it can be defined here, and if it does have a Crestron the port must also be specified.

4.6 Architecture

The final architecture of the product is not what was initially planned, see section 3.2 figure 3. The final architecture has no outgoing connections from the remote server, and due to time restrictions we could not implement a broadcasting feature for the video stream on the asp.net backend so the video is fetched directly from the remote device, see figure 22. All keyboard and mouse commands are passed through the server allowing for Control management.

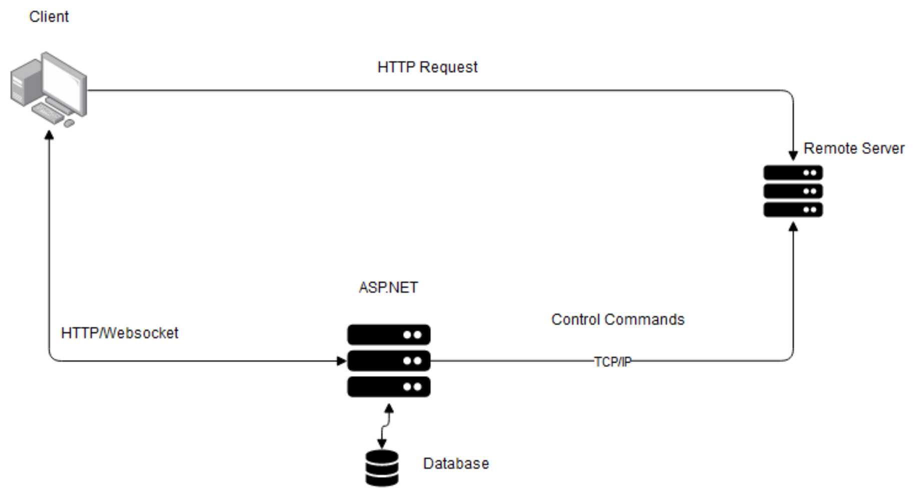


Figure 22: End Architecture

Deployment The current deployment has the remote server connected to the device being controlled with a crestron cable, and one or more video capture devices. The ASP.NET server will load a list of all remote servers, and will receive requests for control from the client with a websocket and manage who has control of the device currently, and pass the commands sent from that connection to the remote device via TCP/IP. The client will directly connect to the remote server for video, but will connect to the ASP.NET server for requesting and controlling the device. The client will also receive data about what devices exist from the ASP.NET server.

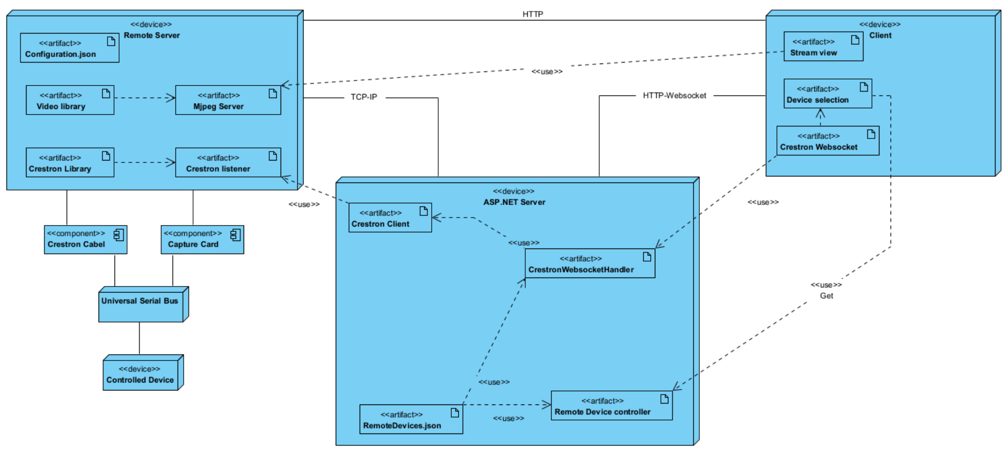

<<device>> Remote Server
<<artifact>> Configuration.json
<<artifact>> Video library
<<artifact>> Mjpeg Server
<<artifact>> Crestron Library
<<artifact>> Crestron listener
<<component>> Crestron Cabel
<<component>> Capture Card
Universal Serial Bus
<<device>> Controlled Device
HTTP
<<use>>
TCP-IP
HTTP-Websocket
<<device>> Client
<<artifact>> Stream view
<<artifact>> Device selection
<<artifact>> Crestron Websocket
<<device>> ASP.NET Server
<<artifact>> Crestron Client
<<artifact>> CrestronWebsocketHandler
<<artifact>> RemoteDevices.json
<<artifact>> Remote Device controller
<<use>> Get

Figure 23: Deployment Diagram

4.7 Video Streaming

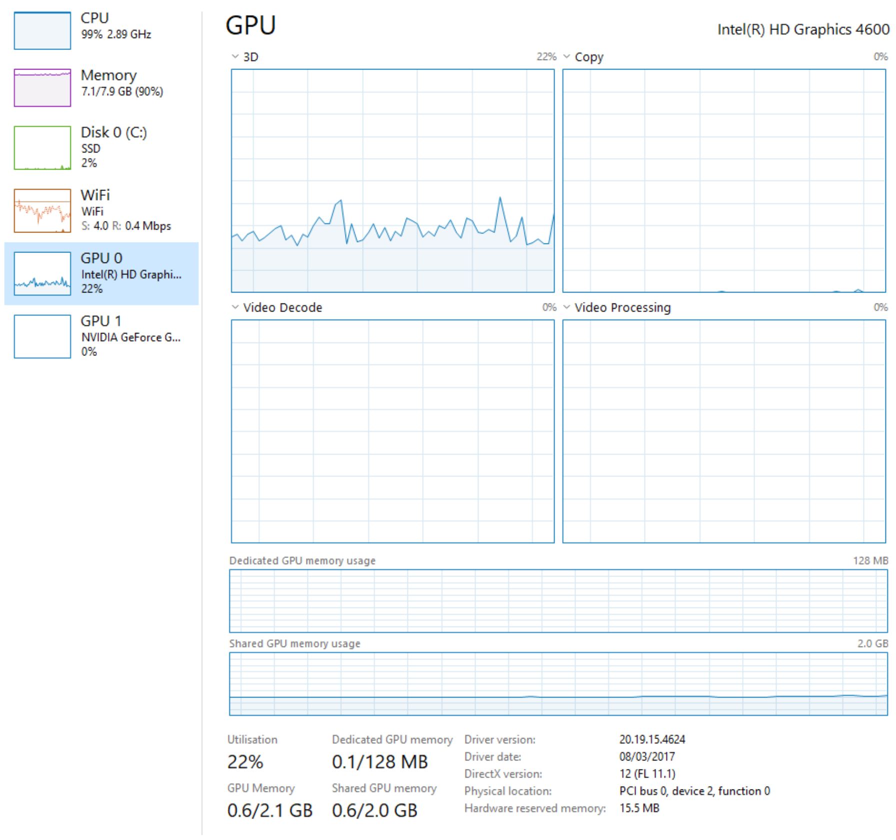
4.7.1 MJPEG

The MJPEG streamer can reliably handle multiple clients and continue streaming MJPEG in conditions with limited bandwidth. The behaviour observed when the stream is transmitting over a limited connection is a reduction in frame rate.

The end-to-end latency or delay of the stream is measured to be on average less than 200ms and maximum 300ms. This is just the delay of the codebase (including Crestron cable delay) and browser and does not include network delay. Testing methodology described in section 3.5.2.

The bandwidth usage is described in section 4.4.1, most of the bandwidth is just the JPEG file and any overhead by the MJPEG streamer is insignificant as it just includes HTTP header data.

Because the streaming server is a REST server the browser decides how to decode the response data. When using chrome to view the stream it can be observed in task manager (windows) that GPU usage increases showing that the decoding work is being put onto the GPU, see figure 24. Only 3D usage is showing however and not video decode suggesting hardware acceleration is not being used, but similar behaviour can be observed when watching YouTube. Task manager usually does not show very accurate GPU data anyways so the data should be taken with some scepticism.



Utilisation 22%
Dedicated GPU memory 0.1/128 MB
GPU Memory 0.6/2.1 GB
Shared GPU memory 0.6/2.0 GB

Driver version: 20.19.15.4624
Driver date: 08/03/2017
DirectX version: 12 (FL 11.1)
Physical location: PCI bus 0, device 2, function 0
Hardware reserved memory: 15.5 MB

Figure 24: GPU usage in windows task manager

4.7.2 FFMPEG

As mentioned in 3.5.3 FFMPEG was used to test different video streaming protocols. These tests were not performed scientifically so the findings are only anecdotal experiences from when the testing was performed as it was only part of the development process to assess the usability of FFMPEG.

As mentioned in 3.5.3 different protocols were tested and the result from all of them was latency of 3 seconds or more. Applying different tunes to the streaming protocols was attempted to reach a latency low enough for real time control but the lowest latency achieved with tuning was around 1 second. This is too slow for real time control and made us abandon FFMPEG and continue work on streaming MJPEG.

These finding match the numbers given by Wowza for latencies on different protocols seen in figure 25 and the theory in section 2.2.1.

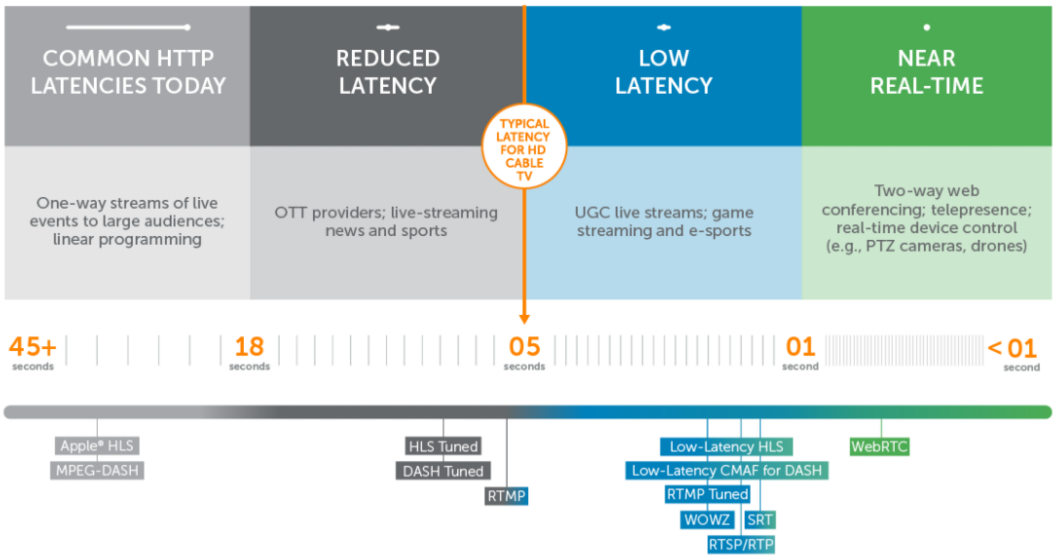

COMMON HTTP LATENCIES TODAY
REDUCED LATENCY
LOW LATENCY
NEAR REAL-TIME
TYPICAL LATENCY FOR HD CABLE TV
One-way streams of live events to large audiences; linear programming
OTT providers; live-streaming news and sports
UGC live streams; game streaming and e-sports
Two-way web conferencing; telepresence; real-time device control (e.g., PTZ cameras, drones)
45+ seconds
18 seconds
05 seconds
01 second
< 01 second
Apple® HLS
MPEG-DASH
HLS Tuned
DASH Tuned
RTMP
Low-Latency HLS
Low-Latency CMAF for DASH
RTMP Tuned
WOWZ
SRT
RTSP/RTP
WebRTC

Figure 25: Diagram showing various protocols latency

4.8 Libraries

4.8.1 Video Library

To handle all video and multimedia related tasks a video library was developed. The library is capable of interfacing with any video device connected to the machine and capture video frames from it. The library can compress the captured frames to different JPEG qualities and down scale the frames to a lower resolution for lower bandwidth usage. The library can also host a MJPEG server where clients can connect to receive a MJPEG stream. The library also contains the beginning of a FFMPEG wrapper for launching of FFMPEG processes to stream video using h.264 over RTSP or any other configuration FFMPEG is capable of.

4.8.2 Crestron Library

To interface with the Crestron cable over serial cable the Crestron library was developed. For sending of serial bytes to the Crestron cable a serial interface was developed. This interface can be reused for sending bytes a Arduino as discussed in section 6.5. The library contains a command parser allowing for simple use of the Crestron cable by for example sending "make c", resulting in the Crestron cable producing a down press of the c key. The library looks up the command in a hash map making the look up O(1) time complexity, reducing latency as much as possible. The parser in the library also has a method of computing the sequence of commands necessary to move the cursor a desired amount. The method combines faster large magnitude moves with small once for quickly and precisely reaching its desired amount of movement.

4.8.3 Socket library

We implemented a static library with methods for sending data with sockets asynchronously and synchronously. The library is meant to let the other classes of the system have access to the same code for sending and receiving messages via TCP/IP sockets.

4.8.4 Server Library

The server library was originally an implementation of all communication in the system. The current server library specifies abstract base classes for inheritance in other parts of the system, allowing for easy implementation of new communication classes. The system created by the server library will have the inheritor of the listener class specify how incoming connections should be handled, creating a new connection class that inherits from the base connection class, allowing for easy setup of network communication, all sending and receiving logic can be customized in the connection inheritor, this system also allows for polymorphism between connection objects if the system has a need for different implementations of connections. The server library has base classes for asynchronous and synchronous systems.

4.9 Code Quality

The code in the project was created with focus on readability, because we knew that it would be developed further by other people in the future.

Style guide The code was written using a generated style guide in Resharper. Resharper will automatically suggest and highlight naming violations in the code with it's linter, this makes the naming of variables, classes, and methods consistent and easy to read across all aspect of the project. see section 2.13 for details about style guides and linters.

XML Comments Classes and methods have XML comments, allowing the IDE to display what the class with intelliSense, as well as allowing for automatic generation of documentation. Comments also make it easier for developers to see what the code does.

Dependency Injection ASP.NET and Blazor both use dependency injection to initialize objects. Blazor uses dependency injection for navigation and logging, as well as initializing HTTP Clients across the project. The ASP.NET server uses dependency injection when distributing the Remote Device Manager that is used to track all the remote devices in the system, allowing for easy access to the remote devices in any future implementations.

Blazor Components All the main functionality in the blazor web assembly application is built with components, allowing for easy restructuring and reuse of various implementations. Each component's code is separated and inherited by the razor component allowing for easy reuse of code when creating new UI for the frontend.

4.10 Maritime Device Compatibility

At NMK Ålesund we tested an early version of our system for compatibility with various maritime devices. The results of the test are displayed in Table 2.

Device Name	OS	Crestron Test	Video Test	Additional info
TECDIS	Windows XP	Success	Success	
FMD-3200	Linux	Success	Success	Main monitor capture only
RPU-025	Linux	Failed	Success	accepts normal HP mouse not crestron cable

Table 2: Table of Maritime device test results

4.11 Latency Test

This section shows the results of the test discussed in section 3.9.5.

Throttle Level	Average latency keyboard	Average latency mouse
No throttle	634ms	660ms
1024 KB/s	1940ms	2040ms
640 KB/s	5500ms	6120ms

Table 3: Latency test

The results form the test show that the latency will get increasingly worse as the bandwidth gets lower as expected because the system does not automatically adjust compression rate based on the current bandwidth available.

4.12 Limitations

Browser The frontend blazor application is made to be used on Chrome and Firefox due to the use of the Pointer Lock API which is mainly supported by google chrome and Firefox, we have not tested the system on other browsers. In order for the browser to load the video in google chrome, the browser must be in allow insecure content, allowing for mixed HTTP and HTTPS, see figure 27.

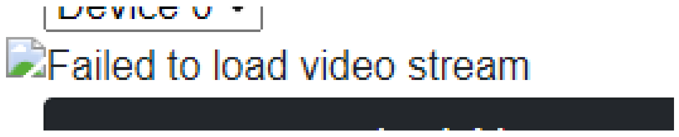

Failed to load video stream

Figure 26: Not reachable video stream

Insecure content
Allow

Figure 27: Image of allowing Insecure content

4.13 Sources of error

Remote Server testing All testing of the Remote server was preformed on windows 10 pro, due to drivers of our capture card not existing on Linux. dotNet 5 is cross-platform so the remote server can run on Linux devices in theory, but we have not tested for this.

Internet Both developers of this solution have relatively high internet speeds, so the latency we detect might be different from if the ASP.NET server is hosted on a lower speed network.

4.14 Security Issues

MJPEG The REST HTTP server used for MJPEG has no validation and can currently be accessed by anyone. Due to the remote server being designed to be used with a mobile data plan, having the connection be open to anyone is not good design, and is a direct design flaw due to the pivot in architecture happening 2 weeks before we were done with the project. It is highly recommended to have the remote server be accessed with a VPN, and not having it be open to the internet if on a data plan.

Remote Control There is currently no separation between different ASP.NET servers, so anyone with technically knowledge could launch a version of the User interface to control any device they know the IP address for due to this being an open source project. Remote devices should have a form of verifying and denying control connections.

IP address The Ip address of remote servers is sent to the client so they can directly connect to the remote device. This is bad design and should be removed as soon as a broadcasting functionality is implemented on the ASP.NET server.

DOS We have not tested how the Remote Server would react to a denial of server attack, and if it is open to the internet this could be a potential problem, see section 2.10.1.

5 Discussion

In this Chapter we will discuss and reflect on changes in priority, issues with development, possible errors in estimation, as well as reflect on our own performance during development.

5.1 Results compared to Product Requirements

The product requirements identified for our product at the beginning of the development and documented in the pre-project report were as follows.

1. The product should be easily extendable to control a wide variety of maritime instruments.
2. The product should reflect the real look of the device to maximize learning value for students and trainees.
3. The product should be able to control a Furuno Radar
4. The product should have a user friendly User Interface

Furuno Radar The latest iteration of the source code fulfills all of the requirements above, except for being able to control the Furuno radar. This makes the project by straight definitions a failure, however the reason for the crestron cable not being able to interface with the Furuno radar is a known issue found by separate research and has to do with the HID of the crestron cable being rejected by the Furuno radar. Due to the corona virus pandemic, we could not readily test our solutions for controlling and interfacing with the various maritime instruments. There were strict restrictions on entering the premises of Furuno during the pandemic. When testing we did also plug in a normal HP usb mouse, and it was able to control the Furuno radar, so a solution to interfacing with the Furuno radar is highly possible, but was not feasible to develop during our project due to the restrictions. There is currently a plan to develop a system that converts the Crestron HID to a downgraded version as an outside solution to the problem, this should be able to control the system via the crestron cable. According to our product owner this means we have created a way for them to control the Furuno radar, fulfilling the requirement.

Device control When it comes to being able to extend and control a wide variety of maritime instruments, currently we have implementations that could support any number of screens for a single instrument, as well as other video devices like cameras, if a system does not have HDMI ports you can still film the screen. The crestron cable we are using to send keyboard and mouse commands works with any normal Linux and Windows system, making it in theory able to control any computer that does have DRM protection,

User Interface The User Interface was designed with Don Normans design principals in mind and has feedback, with limited functionality, we also tried to simulate the look and layout of YouTube to have something that is recognisable to users, because it is the biggest video platform currently. We did not have time for any official user testing, but unscientific and opinionated testing was performed with non-technical people and they were reliably able to navigate the user interface without guidance. We also tested the system with the product owner, he was easily able to navigate the website when we showed him the demo. We gave him vague tasks and he was able to find all parts of the user interface reliably with little guidance.

Maximize learning value The video can be captured with a high quality video capture card and passed into the system, where you can specify how much the image should be compressed, because the system directly captures the screen it feels like you are controlling the system directly, and the image quality can be set to what you need for an accurate representation of the device. The frame rate can also be specified, allowing you to balance the quality and frame rate to customize what result you get in terms of video quality and bandwidth usage.

Additional Nice to have requests Some additional features were discussed about what they wanted the product to support.

1. On/Off Device control
2. Record Video
3. View recorded Video
4. Single person at a time control
5. Low Bandwidth
6. Account Authorization
7. Up-time Tracking

Out of these requested features we have implemented the single person control, and focused on low bandwidth, we also had a plan for implementing time-tracking and on/off control, but due to time restrictions and sudden decision to restructure the entire project structure we could not finish. An implementation of a database, and data access system with authorization using JWT has already been started but will also not be finished in time, details about the implementations of these features will be described in chapter 6.

5.2 Development requests from product owner conctact

5.2.1 Crestron cable

The crestron cable was a solution for remote controlling that had been tested by the product owners before we started the project, and it was requested that we develop a product using the crestron cabel.

5.2.2 MJPEG

MJPEG was a suggestion for compression a compression to use by our project contact. In a rebuttal implementing h.264 was suggested and due to the project contacts willingness to go with what we suggested we were free to choose.

5.2.3 User acceptance Test

A request for a user acceptance test was made by the product owner contact. The User acceptance, see appendix D, is a test detailing how a user can interact with the system, and what the expected result from the test case is. The user acceptance test allows for an objective view of state of the project from the perspective of a user.

5.2.4 Internal acceptance test

A request for an Internal acceptance test was made by the product owner contact. The Internal acceptance, see appendix C, is a test detailing how the internals of the system should perform, and what the expected result from the test case is. The internal acceptance test allows for an objective look at statistics about how the current implementation of backend features that are less relevant to a user perform.

5.3 Live streaming Vs Real-Time Streaming

Most streaming technologies are made for live streaming, typically you would consider live-streaming to be live-video that is streamed to the screen, there is wide support for this and ways to provide this service. Most of these live streams are delayed by up to 30 seconds. What our application needs is called Real-time, this is a term is used in stock trading systems where latency is extremely important, if you want to directly control a device like we are attempting, you should ideally not have more than 50ms in latency however this unrealistic, because in addition to transfer speeds, we also require compression. Compression also takes time so it adds to the latency, making real-time even harder if you also want to compress the data being sent. The real-time streaming is very hard compared to live streaming, and a target of less than 3 seconds places the expectations in an obtainable goal.

5.4 Enhancements

5.4.1 Asynchronous Networking

We refactored the networking to be asynchronous from the original synchronous sockets. The reason for the asynchronous refactoring was general code quality improvement, as well as it being standard for web servers in ASP.NET to be coded with asynchronous sockets, see section 2.5.1. Multiple users can interface with the asp.net server at once allowing the server to delegate resources to current network operations as well handling asynchronously database access and calculations at the same time, making for a more stable and efficient server.

5.4.2 Architecture Change

At the end of the project, we prioritized reconstructing the network architecture, the initial iterations of the project all had an architecture where the remote server would connect to the ASP.NET server. See section 3.2 for an example of the original plan for the project architecture, and figure 22 for the final iteration. The original architecture was planned around the premise that the web server would be the single point of access removing the need for storing remote server information on the server. This would also not expose the remote servers to the internet, making it very secure and immune to DOS attacks, see section 2.10.1, however it made it necessary for the device to continuously communicate with the server, which is wasted bandwidth usage. There were also stability issues with the connections and disconnecting remote devices in the old architecture. The new architecture has the web server connect to the remote device when they are requested by a client, the crestron implementation is still the same on the frontend but now only connects to the remote server when a user requests control, and manages the connection when needed. The video is now transferred via direct http connection, which is a security flaw, as well as non optimal for bandwidth usage. The video stream should be rebroadcast from the web server making it so that the remote server never needs to send data to more than one client. Due to the very late decision of the architecture change we did not have time to develop any security or plan for DDOS protection. The new architecture is very stable compared to the old architecture.

5.5 Reflection

5.5.1 Development

Time estimation At the start of the development process we decided to use Time estimation instead of story point estimation because we thought it would be useful. However this was a massive mistake, making it hard to effectively make estimations and the estimation did not get better over time.

Iterative Releases Our supervisor about halfway through the project suggested that we should make releases of our software. We made releases in Jira, but did not setup a development pipeline with docker hub until the late stages of development. This was because we did not have the proper knowledge required at the time of how ci/cd functioned to actually know this is what our supervisor wanted. If we had setup this from the beginning it would have made the development and testing much smoother and effective.

New Technologies The way we planned our project was very optimistic due to the fact that almost all of the technologies were new to us. We had experience programming in the languages that were used, but Blazor, ASP.NET, were new technologies that we had to learn from the bottom. We had also never used Dependency Injection so that added to the learning curve. We had done network programming before, but never asynchronously, and not with a focus on latency. We also had no experience with creating video streaming, other than sending basic PNG images over TCP connections.

Development Process This was the first big development project we preformed as students, our combined team efforts were not constant, and generally there was always one person with more work than the other, this was due to our inexperience in developing code in teams, as we had only done this with Scrum one other time. Our usage of agile methods were not good, and generally flawed, our user stories were badly written. In the later stages of the project we were reliably able to code together and perform Pair programming, which is a Extreme programming principal described in section 2.8.4. This greatly improved our development speed, and we were actually able to combine our efforts in a much better way. If we had done this from the beginning the of the project the work we would have been able to complete would have be much greater. At the late stages of development we also started using scrum in a more correct way, our user stories were correctly formatted, and we started using releases and CI/CD pipelines.

5.5.2 Over engineering

Our project went through many iterations where we ended up fighting the networking structure of our project, we spent a lot of time trying to improve the system by changing the networking and creating features that were intended as improvements, but would get removed when we had to pivot the project to another path. The project originally had features where the connections would send access tokens to confirm their identity. The system also supported multiple crestron connections at one point. Many of these features were removed as we changed the architecture, making the work essentially wasted. We should have prioritized getting the minimal functionality instead of over engineering systems. We started focusing on this later in the development cycle as we learnt how much simpler everything got when you created the baseline first and then optimized and extended the features.

5.5.3 Non suitable Technologies

During the development process we spent a lot of time researching technologies that ended up not being used in the project.

Signal R Signal R was one of the main reasons for why we decided to pick Blazor and ASP.NET for our project, because as stated in section 2.6.6 it aims to provide real-time communication, so we figured it would be good for communicating the crestron commands from the frontend to the backend, however the hub based system it uses did not fit the use case so we abandoned Signal R implementation quite early in the development process and looked for other options.

5.5.4 Latency

One of the biggest challenges we met during the development process was latency. The early iterations of our system had a latency of up to 5 seconds when controlling, would get continuously worse as the system was used. The reason we had to spend so much time on latency is because there is built in latency in many part of the system that we can not control at all. The latency of transferring data over the network, latency from sending data through the crestron cable, the latency of the remote device you are controlling, all get combined. On top of this innate latency, the video needs to be compressed to lower bandwidth costs, and it being able to support low speeds of 4g connections, this adds an additional layer of latency. More latency is added when the commands being sent to the remote device are passed through the server, to the remote server, before the change happens on the remote device. This makes the system have many layers of latency on top of each other, making it very important for us to create a system with as low latency as possible in the parts we can actually have an impact on.

5.5.5 Resources

During the development of this project, for the vast majority we only had access to 1 crestron cable and 1 capture card. This hindered the development of the product significantly, because realistically only half the development team could actually test the code they made on the real system. Many bugs and unintended errors occurred because one of the developers in the team tested their solutions without the crestron cable on a mock interface.

5.5.6 Global Pandemic

This project was conducted during the global corona virus or Covid-19 pandemic, placing many restrictions on our interaction with other groups involved in the DIKU research project. We could not perform as many tests as we needed on the maritime devices our product was meant to control, because there were understandably heavy restrictions at companies for interaction with people outside the company, especially interactions with students which are globally seen as a more exposed group due to heavy socializing. The first test of our API for interacting with the actual maritime devices happened weeks into the development process. Ideally this should have happened much earlier for us to be aware of how the interactions would work. The remote development process did not effect us as group in particular.

5.6 Technology Reflection

5.6.1 Front-end Framework

We looked at several frontend frameworks for our implementation, since we had no requirements in technology from the product owner.

1. **Blazor** Blazor is a Web Assembly based frontend framework from Microsoft for developing component based web applications in c#.
2. **ReactJs** ReactJS is frontend Javascript library developed by Facebook for creating component based User Interfaces. React Js was a consideration for a frontend system, because it is one of the most popular Web User Interface libraries.
3. **AngularJS** AngularJS is a javascript web framework mainly developed by Google. AngularJS mainly focuses on creating single page applications.

Blazor was the framework we ended up using due to the easy integration of Signal R and allowing us to use the same language in the frontend and backend. We went with a combination of the client-side and server-side blazor systems called ASP.NET Hosted Client-side Blazor, giving us a frontend WebAssembly client and an Asp.Net Server for hosting Data access, authorization and similar things. We also had an internal interest in learning blazor development so it was also a personal choice.

The best choice At the end of the project it is easy to say that picking blazor probably was not the best choice, Blazor is very new compared to the other options and there was little support for web-protocols like WebRTC and we would have to create our own Wrapper for any protocol we wanted to test them because they were made for JavaScript and not c#. Blazor was not inherently a bad choice, but if we had went with angularJs we would have had a way easier time finding example implementations and support for various Web API's, without having to create Wrapper classes to call the Javascript Interop from Web-Assembly. We also picked blazor because our plan was to utilize Signal R as it is meant to be a low latency real-time communication protocol, but it did not suit the use case we were aiming for and we quickly swapped it out for normal web-sockets.

5.6.2 Backend Frameworks

For the backend framework considered we did not really consider any other options than ASP.NET because we decided to use the ASP.NET Hosted as a combination with the Web Assembly. We have had no issues with this, and ASP.NET has been great for our project. When it comes to the combination of the two project, we should probably have created a separate ASP.NET core API project instead of using an ASP.NET hosted project, allowing for the two projects to be more decoupled, allowing for future developers to swap frontend or backend systems if necessary.

5.6.3 OpenCV

OpenCV might seem like an odd fit in the current solution as its role is only interfacing with the hardware and encoding the JPEG's. OpenCV contains a lot of machine vision algorithms that are not used in the current solution and is admittedly a bit overkill. The original reason for picking OpenCV was because it was assumed it would have all the things needed for video streaming. Because the OpenCV library is built upon FFMPEG it was assumed it would be possible to interface with it through OpenCV.

The open licence nature of it also made it compelling and was something we might have put a little too much focus on in the beginning trying to avoid the GNU license. Additionally, we already had some experience with OpenCV making it less time consuming to implement as some of the learning required to use it had already been done. There was also doubts as to how usable the latency would be in the final solution so some of the machine vision could have been used to make a system that was aware of where the cursor was to compensate for some of the lag.

The way OpenCV gives access to devices is also not ideal, making you pick the index of the device you want and no way to list the available devices or device names even though the DSHOW API in Windows do provide device names. The JPEG compression algorithm in OpenCV also do not seem to take advantage of hardware acceleration making the compression of JPEG's significantly impact CPU usage.

Overall OpenCV does the job and with little stability issues and is very conveniently able to rescale the resolution of the video and adjust the compression quality of JPEG's.

5.7 Ideal Solution

The current solution is far from perfect and is mostly just a result of circumstances, meaning the way the whole system works is a result of what we got working at the time we had to complete the project (bachelor deadline). As a result certain aspects of the solution is done in an imperfect way, particularly the video streaming system.

5.7.1 Networking

Throughout the development process a lot of different methods of streaming video was attempted. The main issue was receiving the video using the socket in .NET, it seems like it is not built for reliable transfer of large amounts of data and fails when data is larger than one packet. This could probably have been avoided if a protocol built for large data transfer had been used but due to time constraint this was not explored further.

5.7.2 Video Distribution

The way video is streamed in the current solution is using a MJPEG server hosted at the remote device and not at the location of the web server. This is an issue particularly if the remote device is stationed on a boat at sea and the video data has to go through 4g satellites. Because the server is hosted at the remote device as soon as two clients connect to that device the bandwidth usage is immediately doubled because the MJPEG server has to provide video for both clients. The intended use for the MJPEG server was for it to be hosted with the web server where bandwidth is not constrained and function as a redistributor of the video feed coming from the remote device.

5.7.3 Video Compression

Because the video feed is using MJPEG for compression, the compression ratio is not great and bandwidth usage is high which is not ideal when using satellite 4g. Ideally the remote server would compress the video into AV1 or h.265 and send that to the web server. Currently AV1 hardware encoding is not available on commercial equipment so expensive equipment would be necessary. H.265 however does have hardware encoders in commercial GPU's but has it's limitation in the form of software locks limiting the concurrent amount of streams to some number (in NVIDIA cards specifically). This limitation is not present in Nvidia's professional grade Quadro cards so it is possible to pay your way around the problem.

The h.265 or AV1 video stream coming from the remote server is not ideal for clients as these codecs are a bit recent and support is not as wide spread as h.264. Therefore the video stream can be encoded to h.264 and distributed to all the connected clients because when the video stream has reached main land bandwidth is less of an issue. At this point the video stream can also be encoded at different bitrates and using protocols like HLS clients can dynamically pick the video stream with the bitrate it is able to play. HLS is not ideal for controlling due to the protocol only being able to reach latencies around 2 seconds when using the low latency version but is fine for clients spectating as latency is only an issue when controlling.

5.7.4 Streaming Video

All these video streams need a protocol to reliably transfer it over internet and all the traditional streaming protocols are to slow for real time control. Infact this was a huge issue when researching and developing, there is no clear distinction between streaming, live streaming and real time streaming online so when looking for proposed solutions online many of the results would be of just streaming video from a file source. Then there would be live streaming with massive delay not suitable for real time control. It seems a "last minute" discovery made after the development time was used up, UltraGrid was discovered an ultra-low latency streaming protocol with latency as low as 50ms. It has not yet been tested so the claims have not been verified by us and it might not be a good fit for the solution, but it looks promising. It is a shame it was discovered so late as it seems to be a solid streaming protocol and a good fit.

Another possible streaming "protocol" that could be used is webRTC and this was discovered in the beginning of the development process. The reason we did not try to implement webRTC again was the limited time and the fact that it is peer to peer and does not work well for scaling to multiple clients. In hindsight we should have probably gone with webRTC as a method of streaming video from the remote device to the controlling client and encode the video and streamed it using HLS to spectating clients because to them latency as low as 1 seconds is not important. Because webRTC supports h.264 we would have gotten better compression ratios compared to MJPEG. Compression ratios would also be better for spectating clients because h.264 compression works better with a larger frame buffer.

One upside of not having re-encoding done at the web server is that it scales better with more remote devices specifically. All the bandwidth from many different devices don't have to go through one choke point. This is a minor upside however as modern server infrastructure is built to handle high traffic workloads. Hosting the web server in a data centre using a cloud hosting service like azure puts the burden of handling the load on the service provider. Having all the video traffic go through the data centre might increase hosting cost to an unacceptable amount in which case the more decentralized way might again be better.

5.7.5 Security

The current solution also has some security issues as the remote server does not correctly verify the authenticity of the connecting web server. This allows for anyone to host the web server and take control of the remote device through their own web server. The reason for this security flaw is again the limited time we had and is just what we ended up with when the development time was spent up. The security issues can easily be fixed by running the remote server on a VPN routed to the webserver network making it unexposed to the internet and allowing the web server to connect to the remote device.

5.7.6 Input Capture

The virtual keyboard is not ideal and preferably all keys could be pressed without needing a virtual keyboard. Due to the solution being web based that inherently limits the capability to press certain keys. From experience with unity and making web games there seems to be a more aggressive input capture method game engines like unity employees when building the game as a web app. Using that same API or method would be more ideal for input capture as it from experience allows for esc key capture at least, something the current method is not. The “Unity” method is still not able to stop alt+f4 and windows key shortcuts so a virtual keyboard would be necessary anyways. So it is a bit nit-picky but it would be nice for just out right support for more keys making the use of the virtual keyboard less necessary.

5.7.7 Video Recording

The current solution is missing a few feature that was specified as nice addition, one of them being the ability to record the video feed of a remote device and view it on the website later. If the proposed re encoding solution mentioned above had been implemented it would be easy to pass the video feed into a FFMPEG process and set it to write to a file. Then make that file accessible on the web site with a simple HTTP get command. Again, this was not implemented due to time constraints.

5.7.8 Stability

A lot of time in the late stages of development was spent on fixing bugs and stabilising the code but there are still some issues. We would have liked to have made the code even more stable and bug free considering the remote deployment nature of the system. It is not too much of a concern that the system has some bugs that might require a restart because the machine hosting the remote server can simply be connected to using Windows remote desktop. As long as the code does not crash the host system restating it like this is feasible although not ideal. A common piece of technology in remote systems and systems designed for high up-time and stability uses something called a watch-dog. In short it monitors the system looking out for anything unnormal and can restart the system if it does not respond or it observes anomalous behaviour. This was not implemented due to time constrain and a bit of a lax attitude towards extreme 100% up-time by the product owner, indicating that occasional down time and manual restarts are not that big of an issue.

5.7.9 UI Improvement

The current solution has a very small viewing window for the remote device and this is again due to time limitations. Ideally the view would by default be a bit bigger and there would be a option for full screen. The issue can be worked around by using the magnification feature present in most browsers. The stream can also be made full screen by right clicking and opening image in new tab.

Some radar devices have multiple screen outputs so to effectively use those devices a setup with multiple capture cards and video streams would be needed. This is technically feasible to do on the current system and is just a matter of a UI change on the website to support it. Multiple screen views could be managed using pop out players that many browsers come with support for. This feature gives the client a window of the video stream they are free to move and position however they want on their machine.

5.7.10 Ideal Example

To sum up, the ideal solution would hardware encode a more advanced video codec like h.265. Streams that do not need low latency encode with a larger frame buffer giving better compression ratios. The video feed would then be transmitted using a low latency protocol like webRTC or UltraGrid. Then at the server the video feed would be re encoded and distributed to all clients watching that device. The controlling client is the only one needing low latency so that client would receive a webRTC or UltraGrid video feed. The other spectating clients can receive a HLS stream with a larger frame buffer for better compression ratios. If recording is enabled the video feed would be sent to a FFMPEG process writing the video file to the server storage where it can later be access from the website. Instead of using pointer lock to capture client input something similar to what Unity uses would be used to capture more keys. And a full screen mode would be available. The remote server would have a watchdog to insure uptime and the system would generally be very stable. Remote devices should authenticate connecting servers. The UI and UX should be improved with a friendlier design and also nicer looking.

6 Planned Features

In this chapter we will discuss the plans and proposals we have for features that were not finished in time, and why we think they should be implemented.

6.1 Broadcasting MJPEG

MJPEG broadcasting refers to sending video data to the asp.net server, so it can be rebroadcast to all the clients wanting the video stream that is being sent. This is a feature that is required for reducing bandwidth consumption on the remote server, with he current implementation the usage will be the same if there is only 1 client wanting the video, but each new viewer will establish a new connection that will be served by the remote server, increasing the bandwidth usage. If the video data was rebroadcast you could avoid the remote server that is presumably using limited and expensive bandwidth from broadcasting to all clients, requiring only one connection.
Our proposed implementation of MJPEG broadcasting is creating a Stream manager on the backend, which will have access to all remote device information and can establish connections to different remote device video ports to get the stream. This manager can then have an interface that can be accessed from a controller. The client can call the get request with data containing what device they wish to have a stream from, and the controller can then call the interface of the stream manager and request the image data and pass it on to the client. It is important that the stream manager knows if no users are currently watching the stream so it can close the connection to the remote server, this is optimal for reducing data usage.

6.2 HTTPS MJPEG

HTTPS MJPEG refers to making the MJPEG stream that sends video data to the client HTTPS. The current HTTP implementation will get blocked by many browsers and counts as insecure content, the web is moving towards HTTPS only support, so a HTTPS solution would be a smart feature when it comes to future proofing, it also removes the need to enable unsafe mode on Google Chrome.
Our proposed implementation for HTTPS MJPEG is to have the original http stream remain as http, but have rebroadcasting as mentioned earlier in the chapter, rebroadcast over HTTPS from the backend when implementing a controller or other solution, for the broadcast of video, see section 2.6.2.

6.3 Spectator Mode

Spectator mode refers to a different video stream being displayed to users that are not controlling the device directly. The current implementation of video has all users on the low latency MJPEG, but this is not optimal, the implementation of MJPEG was to make remote controlling usable, but it is not necessary to have low latency if you just want to view the video. Therefore you can have a different video stream that uses lower bandwidth, and has much more delay.

6.3.1 H264

H.264 refers to the video compression codec, see section 2.1.3. While testing streaming of h.264 it was discovered that common video streaming protocols like RTSP was too slow for use in real time control situations. H.264 streaming can be used in scenarios where low latency is not as important as mentioned in section 6.3. The higher latency streaming can also be used on cameras used for viewing the weather conditions on the site of the maritime equipment. H.264 encoding is preferable because it has better compression ratios than MJPEG making it spend less bandwidth for the same amount of perceived quality.

Our proposed solution for implementing h.264 video streaming with high latency is to continue development of the FFMPEG wrapper located in the video library. Using that to start a FFMPEG streaming process with default settings and using RTSP for streaming. The stream can then be captured by the web server using a FFMPEG process there too and set it to redistribute in a web friendly protocol like HLS.

Alternatively FFServer could be used for redistributing seeing as that is what it is intended for. FFServer is Linux only however making it more cumbersome to setup. We have not look as deeply into FFServer as FFMPEG and it seems advisable to go with FFMPEG considering the solution is not going to do any mass media distribution and FFMPEG is probably enough for basic video distribution.

6.4 Video

6.4.1 Recording

Video recording refers to recording and storing the video feed coming from a remote device locally on the server to later be retrieved for viewing. The current system can accommodate this feature with a few tweaks.

Our proposed solution is to take the source of the video stream URL and give it to a FFMPEG process running on the server. The FFMPEG process can then set a folder on the server as the target output and encode and store the video file there. In the current solution with the remote server hosting the MJPEG server the source URL would be the remote server IP address and MJPEG port. If the streaming of video is reworked to have the web server redistribute the video than that rebroadcasted stream can be used. Using the rebroadcasted stream prevents two clients being connected to the remote server and doubling the bandwidth usage.

6.4.2 Video viewing

Video viewing refers to viewing of the stored video recordings on the web server through a web interface.

Since the current system is very open and is intended to be that way our proposed solution for accessing the video files on the server is to keep them in the root folder and setting the source of a video element to that of the desired video.

If the system needs to restrict certain videos a more advanced solution with specific video streaming is needed. Our proposed solution is to use Blazors file streaming ability to give the client a stream of the specific file allowing for it to be accessed as if it was stored locally.

6.5 On/Off control

On/Off refers to the implementation of a system that can turn off devices like the Furuno Radar unit. There is currently no support for this in the application.
Our proposed solution is to extend the Crestron command class that is currently used for sending commands to the Crestron. The class is serialized to json and sent to the remote server for parsing. This allows for easy extensions of functionality, where you can have a message type enum in the class to specific what type of action you want to preform, and have a check on the remote server to see what action is meant to be preformed. This would allow you to specify in the control message that a system should be turned off or on.
The plan for turning the devices on or of was with a electrical relay unit connected to an arduino. Using an arduino would allow you to plug it in to the remote server and use the library we created for sending commands to the Crestron cable, to send data to the arduino.

6.6 Uptime tracking

Up-time tracking refers to having statistics about when a remote server is running and reachable. With the current implementation you can not see how long a device has been up, all you get information on is if the device was reached with ICMP, see section 2.4. This is not optimal, as you would likely want to see how long the device has been running.

Our proposed implementation of uptime tracking is to store the current time when a successful ping is sent to the remote server, this can either be uploaded to the database if the implementation is meant to be statistics over a long period of time, or it can be stored in a variable for future more hasty use. The implementation might want to ping devices on its own and not just when users interact with the system, requiring implementation of a Remote Device monitoring system service, than can ping and track the status of various devices. A service is a background task in asp.net specifically.

6.7 Database

A database is required for account authorization and storage, and implementing a data access controller for updating the Remote Devices, removing the need to load data with Json at startup. The database could also be used to store remote server up-time stats.

6.7.1 Entity Framework Core

Our proposed solution for an implementation of a database was to use entity framework core with he code first approach, see section 2.9.3, and have entity framework connect to the DBMS, and create a database based on the code. SQL Server is the DBMS we planned to use, however any DBMS with entity framework core support can be used.

6.8 Authorization

6.8.1 Client

Authorization on the client refers to being able to lock certain systems behind an authorization wall, where only people with the correct authorization level that have been authenticated by the backend ASP.NET server can access it. Authorization will allow us to lock the control of the system to only people with higher level access, as facilitating support for other features such as being able to limit who can access what data from the server in a hypothetical future system where recorded videos were stored as a user specific resource.

An implementation of a client side authorization system using ASP.NET Core authorization and JWT, see section 2.10.2, is currently partially finished, the logic for storing and updating JWT tokens on the frontend web assembly app is implemented, but not tested with a backend. The authorization level will be used to lock the device control component and virtual keyboard so only authorized users can access this feature.

6.8.2 Server

Authorization and Authentication on the server side refers to login and logout functionalities, allowing for user accounts to be stored with specific roles and claims, and having system for confirming identities on the backend. There is currently no implementation of authentication or authorization on the backend.

Our proposal for the implementation of backend authorization and authentication is to develop a complementary system to the one mentioned in section 6.8.1, this refers to the implementation of a JWT Claims system where a user can get a token with their claims or authority level when logging in with their account details. JWT authorization allows the server to easily identify the users because the token data is sent with every HTTP request in the header, and is verified by authorization middleware in ASP.NET. This system can be created by using the ASP.NET core authorization package, and having already implemented a database system for storing user accounts.

6.9 UI improvement

Top Bar The top bar is currently not connected to anything, the hamburger menu on the left of the top bar is intended to serve as a navigation menu when more pages are implemented. The Home button is functioning as intended. The profile button is intended to be coupled with the authorization system. Allowing users to see their account details, and authorization levels. The settings button is an artifact from early planning, and there aren't specific plans for implementations of it, but general things like password change should be added to this as a drop-down menu.

7 Conclusion

The result of our thesis is prototype of a remote control system that is able to control a wide variety of maritime systems from a Web user interface with direct mouse and keyboard inputs or a virtual implementation of a keyboard, the prototype also supports many video streams being present on the same device. The user interface of our system should be able to provide a realistic look of the system preserving learning value of the data. our solution is confirmed to work for all the maritime devices we tested except for one, and this device has a future solution that will allow our system to control it, via downgrade in HID version. The solution is not limited to maritime system, any system that accepts the USB HID can be controlled with this implementation.

We also have an outline of potential improvements that can be made to the system with proposed implementations that will improve bandwidth and usability. As well as research regarding streaming protocols that can be used to extend functionality of the system in the future.

Our experience in trying to create an optimal solution for this problem is that creating a remote control system with low bandwidth, low latency, high video quality, on a connection with bad internet speeds, is very hard to achieve. Especially when using a method of controlling the system externally adding more latency to the system, which is already filled with innate latency. Real-time control of remote devices with high video quality at a level where it is usable for simple commands is very possible, but the latency will never be at level where it is seamless with current available technology. This therefore makes it necessary to compromise since the ideals of the system all contradict each other, latency and compression, compression and video quality, are all things that retract from each other.

We also found that most documentation for streaming application revolves around live streaming with large delays and there is limited research on real-time streaming over the internet.

During this project we learnt a lot about almost all aspect of development, how to structure the development process. How to prioritize and optimize work when working towards a goal. Some of the most challenging aspects was time estimation, and creating user stories. We also learnt many new technologies like Blazor and ASP.NET, as well as a very large amount about how internet communication works, various protocols and how to use them. Restrictions that are present with current technologies. We also had little to no experience with Video encoding which we now have a general view of how to do.

The final verdict of the product we developed is that it was a successful prototype that fulfilled the base target of what was requested, but that lacked some extra functionality that was not implemented due to time restrictions.

Bibliography

AMD RDNA™ 2 Graphics Architecture (). URL: https://www.amd.com/en/technologies/rdna-2.

CBL-USB-RS232KM-6 (). URL: https://www.crestron.com/Products/Interconnects,-Interfaces-Infrastructure/Interconnects/Virtual-Control-Surface-Cables/CBL-USB-RS232KM-6.

Enginess (). 'The 6 Principles Of Design, a la Donald Norman'. In: URL: https://medium.com/@sachinrekhi/don-normans-principles-of-interaction-design-51025a2c0f33.

Extreme programming (). URL: https://en.wikipedia.org/wiki/Extreme_programming.

Leveraging the Hardware JPEG Decoder and NVIDIA nvJPEG Library on NVIDIA A100 GPUs (). URL: https://developer.nvidia.com/blog/leveraging-hardware-jpeg-decoder-and-nvjpeg-on-a100/.

Microsoft ([a]). *Asynchronous Socket*. URL: https://docs.microsoft.com/en-us/dotnet/framework/network-programming/using-an-asynchronous-client-socket.

— ([b]). *Entity Framework Core*. URL: https://docs.microsoft.com/en-us/ef/core/.

— ([c]). *KeyboardEventArgs*. URL: https://docs.microsoft.com/en-us/dotnet/api/microsoft.aspnetcore.components.web.keyboardeventargs?view=aspnetcore-5.0.

Motion JPEG Streaming Server (). URL: https://www.codeproject.com/Articles/371955/Motion-JPEG-Streaming-Server.

mozzila (). *Pointer Lock API*. URL: https://developer.mozilla.org/en-US/docs/Web/API/Pointer_Lock_API.

rfc1812 (). *Internet Control Message Protocol*. URL: https://datatracker.ietf.org/doc/html/rfc1812.

rfc675 (). *Transmission Control Protocol*. URL: https://datatracker.ietf.org/doc/html/rfc675.

rfc768 (). *User Datagram Protocol*. URL: https://datatracker.ietf.org/doc/html/rfc768.

Streaming Protocols: Everything You Need to Know (Update) (). URL: https://www.wowza.com/blog/streaming-protocols.

StreamingGuide (). URL: https://trac.ffmpeg.org/wiki/StreamingGuide.

Video Encode and Decode GPU Support Matrix (). URL: https://developer.nvidia.com/video-encode-and-decode-gpu-support-matrix-new.

Video used when testing (). URL: https://www.youtube.com/watch?v=ubFq-wV3Eic.

WebAssembly (). *WebAssembly*. URL: https://webassembly.org/.

WebRTC (). *WebRTC*. URL: https://webrtc.org/.

Wikipedia ([a]). *Advanced Video Coding*. URL: https://en.wikipedia.org/wiki/Advanced_Video_Coding.

— ([b]). *AV1*. URL: https://en.wikipedia.org/wiki/AV1.

— ([c]). *COM (hardware interface)*. URL: https://en.wikipedia.org/wiki/COM_(hardware_interface).

— ([d]). *Denial of service Wiki*. URL: https://en.wikipedia.org/wiki/Denial-of-service_attack.

— ([e]). *Dependency Injection*. URL: https://en.wikipedia.org/wiki/Dependency_injection.

— ([f]). *FFmpeg*. URL: https://en.wikipedia.org/wiki/FFmpeg.

— ([g]). *Hypertext Transfer Protocol*. URL: https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol.

— ([h]). *Hypertext Transfer Protocol secure*. URL: https://en.wikipedia.org/wiki/HTTPS.

— ([i]). *JPEG*. URL: https://en.wikipedia.org/wiki/JPEG.

— ([j]). *Json Web Token*. URL: https://en.wikipedia.org/wiki/JSON_Web_Token.

— ([k]). *Latency (engineering)*. URL: https://en.wikipedia.org/wiki/Latency_(engineering).

— ([l]). *Motion JPEG*. URL: https://en.wikipedia.org/wiki/Motion_JPEG.

— ([m]). *NoSQL*. URL: https://en.wikipedia.org/wiki/NoSQL.

— ([n]). *Open Source wikipedia*. URL: https://en.wikipedia.org/wiki/Open_source.

— ([o]). *Os Level Virtualization*. URL: https://en.wikipedia.org/wiki/OS-level_virtualization.

— ([p]). *Real Time Streaming Protocol*. URL: https://en.wikipedia.org/wiki/Real_Time_Streaming_Protocol#Server.

— ([q]). *Relational database*. URL: https://en.wikipedia.org/wiki/Relational_database.

— ([r]). *Scrum (software development)*. URL: https://en.wikipedia.org/wiki/Scrum_(software_development).

— ([s]). *Serial port*. URL: https://en.wikipedia.org/wiki/Serial_port.

— ([t]). *Signal R*. URL: https://en.wikipedia.org/wiki/SignalR.

— ([u]). *Software license*. URL: https://en.wikipedia.org/wiki/Software_license.

— ([v]). *Transport Layer Security*. URL: https://en.wikipedia.org/wiki/Transport_Layer_Security.

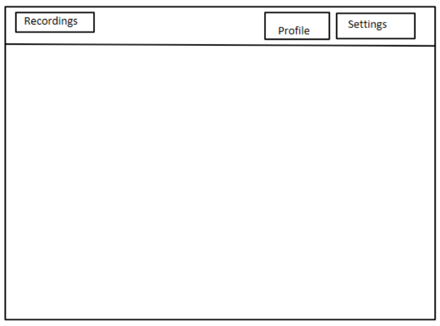
— ([w]). *WebRTC Wikipedia*. URL: https://en.wikipedia.org/wiki/WebRTC.

Wikipedia ([x]). *Websockets*. URL: https://en.wikipedia.org/wiki/WebSocket.

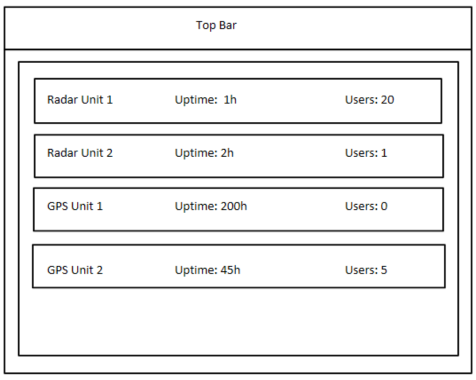
Appendix

A Appendix Diagrams

A UI Examples

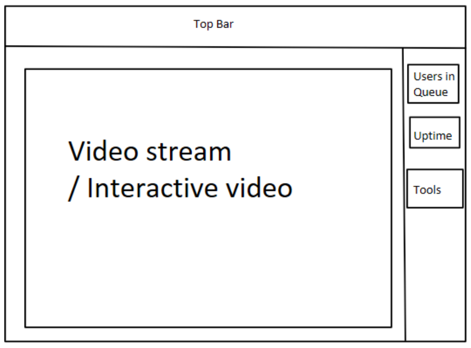


top Bar

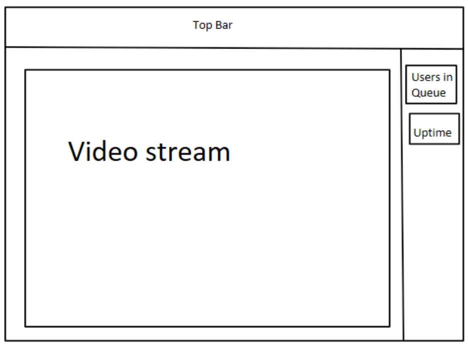


Radar Unit 1	Uptime: 1h	Users: 20
Radar Unit 2	Uptime: 2h	Users: 1
GPS Unit 1	Uptime: 200h	Users: 0
GPS Unit 2	Uptime: 45h	Users: 5

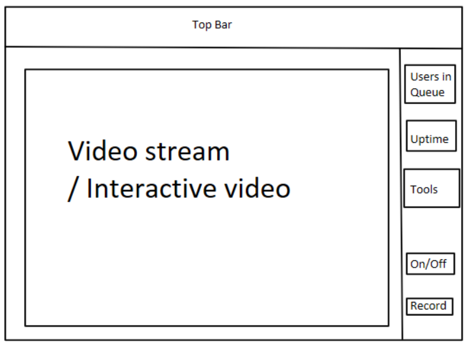
Device selection



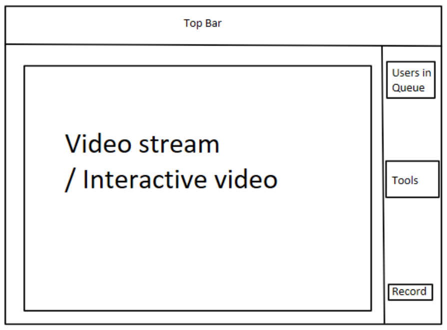
Video View 1



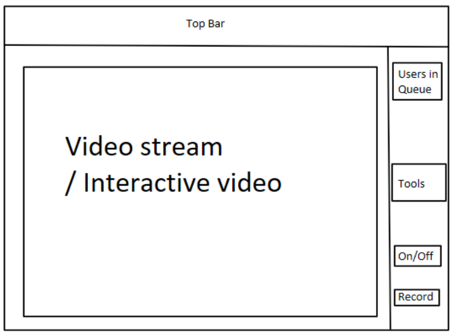
Video view 2



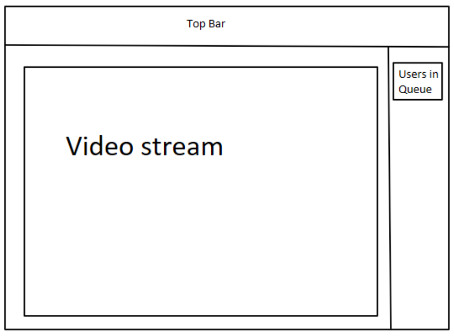
Video View 3



Video view admin

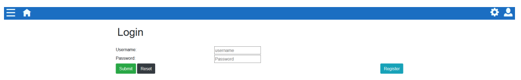


Video view



Video view 4

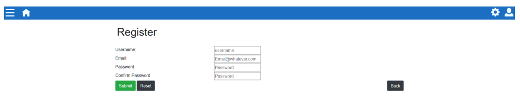
B Login and registration UI


Login

Username: username

Password: Password

Login Page


Register

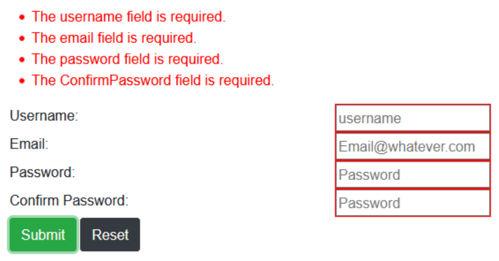
Username: username

Email: Email@whatever.com

Password: Password

Confirm Password: Password

Registration Page


- The username field is required.
- The email field is required.
- The password field is required.
- The ConfirmPassword field is required.

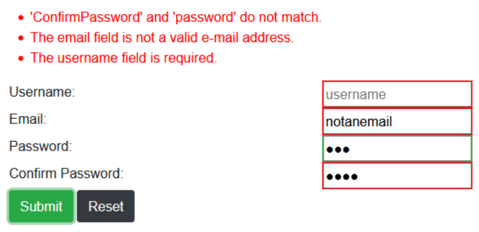
Username: username

Email: Email@whatever.com

Password: Password

Confirm Password: Password

Validation 1


- 'ConfirmPassword' and 'password' do not match.
- The email field is not a valid e-mail address.
- The username field is required.

Username: username

Email: notanemail

Password: ●●●

Confirm Password: ●●●●

Validation 2

B Appendix Preproject report

FORPROSJEKT - RAPPORT

FOR BACHELOROPPGAVE

NTNU
Kunnskap for en bedre verden

TITTEL: **Distributed Instrument Cluster**			
KANDIDATNUMMER(E): **Mikael Nilssen** **Andrè Helland**			
DATO:	EMNEKODE: * **IE303612**	EMNE: **Bacheloroppgave (Data)**	DOKUMENT TILGANG: - Åpen
STUDIUM: **DATAINGENIØR**		ANT SIDER/VEDLEGG: /	BIBL. NR: - Ikke i bruk -
OPPDRAGSGIVER(E)/VEILEDER(E): Norvald Kjerstad og Arne Styve			

OPPGAVE/SAMMENDRAG:

Vi skal planlegge og utvikle programvare for å koble diverse Maritime instrumenter til ett web basert brukergrensesnitt. De viktigste funksjonene skal være fjernstyrt radar kontroll, Video opptak, tilskuer modus, og verktøy valg. Alt skal kunne styres fra ett web brukergrensesnitt. Mulige andre verktøy enn radar, er ECDIPS, AIS, GPS, men vi skal fokusere på radar kontroll.

INNHOLD

1 INNLEDNING

Dette er en forprosjektrapport for DIKU finansiert forskningsprosjekt for bachelor oppgaven til Mikael Nilssen og André Helland ved NTNU i Ålesund. Rapporten skal inneholde forutsetninger og planlegging, arbeidsmetoder, og målsetninger ved gjennomføring av prosjektet.

Oppdragsgiveren har spesifisert at hoved målet med oppdraget er å lage ett verktøy som kan hjelpe maritime studenter med å lære å jobbe med maritime instrumenter som radar og GPS fra klasserom og kurs.

Vi skal derfor forske på måter å fjernstyre disse systemene, bestemme om det er mulig, Forske på måter å redusere nettbruk av disses systemene så de kan være plassert på fjerne lokasjoner med kun tilgang til 4g internett. Lage systemer som har mulighet for å bli pakket sammen med en størst mulig rekke av maritime systemer uten å måtte tilpasse programvaren for hver enhet.

2 BEGREPER

Radar – Instrument som bruker radiobølger for å finne objekter i nær områder
GPS- Global positioning System. Bruker satellitter for å finne posisjoner
4G- Broadband cellular network for mobile nettverk
VS- Visual Studio, Et Integrated Development environment eller en IDE, for kode
C# - Microsoft sit object oriented programmeringspråk. Som kjøerer på .Net runtime
.**Net** – En runtime som bruker bytecode eller ett mellomspråk før det kompilere til maskinkode
Docker- Viritualiserings verktøy for å lett kjøre programvare på en rekke forskjellige systemer og i skyen.
Linux- Operating system
Github- Versjonskontroll system for kode
Jira- Agile Development Tools, som hjelper utviklere samarbeide og planlegge arbeidet sit.
RTSP- Streaming protocol som brukes for å sende live video over nettet
WebRTC- Streaming protocol som brukes for å sende live video over nettet
FFMPEG- Video and audio handling library
Blazor- C# basert Web Rammeverk
UI- User Interface (brukergrensesnitt)
Crestron – Selskap som lager spesial kabel nødvendig for å få emulere radar tastaturet.
Furuno – Produsent av radar og radar systemer
Emulere – Prøve å gjenskape karakteristikker med andre verktøy
MVP – Minimum valid product
WebRTC, RTSP – Real time connection web protokoller
Fullstack – Alle sider av utvikling, Både server og brukergrensesnitt
Latency – Hvor lang tid det tar for mål og oppnås, ofte i referanse til sending av data

3 PROSJEKTORGANISASJON

3.1 Prosjektgruppe

Studentnummer(e)	Navn
498752 498754	Mikael Nilssen André Helland

3.1.1 Oppgaver for prosjektgruppen – organisering
Gruppen skal samarbeide på alle delmoduler i prosjektet. Ingen hierarki, flat gruppe struktur.

3.1.2 Oppgaver for gruppe medlemmer
Rapport skriving

Dokumentasjon

Programmering

Modellering (database/UI)

Testing QA

3.2 Styringsgruppe (veileder og kontaktperson oppdragsgiver)
Veileder Arne Styve

Kontaktperson Lars Ole Hurlen

Oppdragsgiver Norvald Kjerstad

4 AVTALER

4.1 Avtale med oppdragsgiver
Oppdragsgiver Norvald Kjerstad har referert til samarbeid med Lars Ole Hurlen som skal være mellommann og involvert samarbeids partner med gruppen.

4.2 Arbeidssted og ressurser
Arbeid skal hovedsakelig utføres fjernt med nettbasert kommunikasjon mellom gruppe medlemmer, veileder og oppdragsgiver.
Gruppemedlemmer har avtalt møte en gang i uken for å rapportere tilstand, og fremgang til prosjektet.
Gruppen skal også ha møte med veileder annen hver uke for å få eksterne kommentarer.
Oppdragsgiver blir kontaktet fortløpende med lav terskel.

4.3 Gruppenormer – samarbeidsregler – holdninger

Gruppen har ingen krav til arbeidstider, eller nødvendig møte tider, annet enn etter hver sprint.
Problemer kan bli tatt opp når som helst grunnet nettbaserte møter.
Arbeidsetikken til gruppen vil være basert på milepæler i prosjektet og gruppemedlemmer vil prøve å holde ett jevnt og likt arbeids nivå med hverandre.
Samarbeidsforholdene til gruppen vil være basert på standard IT prosedyrer med agile Development, med Project management programvare for å holde følge på fremgang og hvem som jobber med hva.
Gruppemedlemmer skal også operere med samme dokumentasjons kvalitet og test nivå.

5 PROSJEKTBESKRIVELSE

5.1 Problemstilling - målsetting - hensikt

Effektmål:
Hjelpe maritime studenter lære og forstå maritime instrumenter.
Muliggjøre opptak av instrumenter så undervisere kan klargjøre eksempel.

Resultatmål:
Oppnå effektive resultater angående hvor mye data programmet bruker.
Enkelt utvidet og bra dokumentert programvare.
Fjernstyring av radar enhet.
Emulert tastatur.
Mulighet for enkel utvidelse av enheter.
Flere bruker skal kunne se på samme instrument.
Logging av oppetid til instrumenter.

Prosessmål:
Bruke minst mulig tid for å nå MVP.
Få erfaring innen Fullstack utvikling.

5.2 Krav til løsning eller prosjektresultat – spesifikasjon

Produktet skal enkelt kunne utvides til å styre en rekke maritime instrumenter.

Produktet skal reflektere utseende til det ekte instrumentet for maks læringsverdi.

Produktet skal kunne styre Furuno Radar.

Produktet skal ha ett brukervennlig brukergrensesnitt.

Produktet sine spesifikke krav er løse, og det er ansett å være ferdig når vi når en MVP, men det er mange ekstra funksjoner som er ønsket av oppdragiver utenfor det som er krav.

5.3 Planlagt framgangsmåte(r) for utviklingsarbeidet – metode(r)

Vi skal bruke Agile utviklingsmetoder, med 1 ukers sprint perioder. Dette er vanlig innen utviklings miljøer. Sprint med lengde på 1 uke lar oss lett finne blokkerende problemer, og endring i retning for prosjektet. Det er veldig lett og komme på villspor i hva som faktisk er nødvendig å lage i et utviklings prosjekt, Agile lar deg ha endringer ofte, som tilpasses til situasjonen som er hele tiden. Programmet vi bruker er Jira, og vi bruker spesifikt scrum til å fordele arbeid, og estimere tidsbruk på hver oppgave.

5.4 Informasjonsinnsamling – utført og planlagt

Prosjektet vil hovedsakelig handle om implementasjon av Radar enheten sine funksjoner, hvor den ikke har nåværende web basert brukergrensesnitt. Deretter skal vi gjøre det mulig å legge til andre web baserte enheter til samme nettside. Så langt har vi funnet at alle enheter som har ett web brukergrensesnitt allerede vil være enkelt og legge til i våre systemer.

Vi har undersøkt mye angående video og direkte sendte internettpakker. Nåværende virker WebRTC som det mest lovende, men vi undersøker også RTSP og andre protokoller. Vi har undersøkt VLCLib, FFMPEG, og andre video encoding programvare for å mest effektivt sende video mellom enheter og serveren.

Videre i prosjektet må vi utføre tester og målinger av muligheten og effektiviteten av disse protokollene, og hvor stor latency det skaper for produktet. Mye av informasjons innsamlingen vi skje senere i prosjektet grunnet at alle detaljene på ett så løst prosjekt ikke kan fastslås så tidlig.

5.5 Vurdering – analyse av risiko

Trussel	Sannsynlighet	Effekt	Risikoreduksjon	Risikonivå
Kovid / generell sykdom	Medium	Høy	Mask, håndvask, antibac, antisosial.	Middels
Kan ikke fullføre MVP	Lav	Høy	Riktig arbeidsmoral	Lav
Ikke mulig å implementere tiltenkt kontrollnivå	Middels	Høy	USB-emulator, USB-fangsenhet	Middels
Kan ikke oppfylle båndbredde krav	Middels	Middels	Plan B (lav rammeløsning)	Middels
Gruppemedlem lider arbeidstretthet	Høy	Høy	Skam og skyld (arbeidsmoral)	Middels
Store tilbakeslag og pivoter	Middels	Middels	God planlegging	Middels
Programvarefeil	Høy	Lav	God kodekvalitet, Dokumentasjon,	Middels

			Enhetstesting	
System ustabilitet	Lav	Medium	Linux, Watchdog timer	Lav
Kan ikke generaliseres for alle maskinvarekomponenter	Lav	Medium	God kodekvalitet, Samhold, Lav kobling	Medium
Tap av prosjektfiler	Lav	medium	Github, lokalt lagret kopi, RAID 5 backup.	Lav

-

5.6 Hovedaktiviteter i videre arbeid

Nr	**Hovedaktivitet**	**Ansvar**	**Tid/omfang**
A0	Research	AH/MN	∞
A1	Crestron Demo	MN	24
A11	Crestron Library	AH/MN	48
A2	Transcoding Demo	AH	24
A21	Transcoding Library	AH/MN	48
A3	Web Server	AH/MN	70
A31	Database	MN/AH	48
A32	UI	AH/MN	50
A33	WebRTC	AH/MN	24
A4	Rapport	AH/MN	70

5.7 Framdriftsplan – styring av prosjektet

5.7.1 Hovedplan

- Research protokoller for video strømming og andre teknologier nødvendig for prosjektet.
- Utvikle biblioteker for emulering, video strømming, databasehåndtering, etc.
- Programmere radar skjermopptak og tastatur emulering.
- Programmere server.
- Koble systemet.
- Implementere ekstra egenskaper.
- Skrive sluttrapport.

5.7.2 Styringshjelpemidler

I dette prosjektet bruker vi JIRA hvor vi vil estimere alt vi skal lage samt logge tid og ha en oversikt over framgang. Ved hjelp av burndown-chart kan vi unngå at prosjekt størrelsen blir for stor. Alle møter dokumenteres også i JIRA.

5.7.3 Utviklingshjelpemidler

Programmering i C# skal utføres ved hjelp av VS. Blazor rammeverket skal benyttes for å gjøre C# koden om til en nettside. Docker skal benyttes for å utplassere server koden skal kjøre i Azure. Operativ systemet Linux skal benyttes for å ha et pålitelig system med høy oppetid.

5.7.4 Intern kontroll – evaluering

Interne evalueringer vil skje ukentlig hvor vi diskuterer framgang og endringer i retning som skal forbedre prosjekt resultat. Vi diskuter også om noen har gjort for lite i forhold til andre gruppe medlemmer.

5.8 Beslutninger – beslutningsprosess

Beslutninger vil bli tatt basert på konstruert krav spesifikasjon laget med kommentarer fra oppdragsgiver.
Beslutninger vil alltid prøve å nå best mulig end produkt.
Grunnet to likeverdige gruppemedlemmer blir interne konflikter løst med hjelp av veileder.

6 DOKUMENTASJON

6.1 Rapporter og tekniske dokumenter

Arbeids log annenhver uke.

Sprint retrospektive møte notater.

Veileder møte notater.

Bachelor rapport.

Research dokumenter.

Framdriftsrapport.

Kravspesifikasjon.

UML Diagram.

UI Diagram.

Instruksjonsmanual.

7 PLANLAGTE MØTER OG RAPPORTER

7.1 Møter

7.1.1 Møter med styringsgruppen

Gruppen skal møte med veileder annenhver uke, hvor tidspunkt kan variere.
Hensikten med møtet er for veileder å gi tilbakemelding på gruppens framgang og gi veiledning.

7.1.2 Prosjektmøter

Prosjektgruppen skal ha minst ett møte på slutten av hver sprint. Sprint varigheten er en uke og startes og avsluttes på mandager. Hensikten med møtet er å delegere arbeid og få oversikt av hvilken deler av prosjektet som går bra og hvor det må gjøres endringen for å forbedre resultater.

7.2 *Periodiske rapporter*

7.2.1 Framdriftsrapporter (inkl. milepæl)

Arbeids rapport/arbeids log skal leveres annenhver uke for å dokumentere utført arbeid og vise framgang.

8 PLANLAGT AVVIKSBEHANDLING

Prosjekt planen har avsatt tid på slutten for uforventede tilbakeslag så det er rom for å bruke ekstra tid på det som overtrer tidsfrister. Mye av funksjonene til prosjektet er ekstra funksjoner og kan derfor droppes. Hvis video innkoding ikke klarer å bli implementert må skjermdelingen bli utført ved å sende bilder uten bruk av video kompresjons teknologi.

9 UTSTYRSBEHOV/FORUTSETNINGER FOR GJENNOMFØRING

Hardware:

- Crestron-kabel
- USB til serie adapter
- Video capture card
- Web kamera
- Furuno radar system
- Datamaskin (koblet til radar system)

Software:

- Jira
- Azure
- Microsoft Word
- Microsoft PowerPoint
- Visual Studio
- Docker
- Linux
- Blazor
- mySQL
- C#
- .NET
- JavaScript

C Appendix Internal Test Procedure

Internal Test Procedure for Distributed Instrument Cluster

Items used for the test.

Minimum 2 Windows Computers

1 Crestron cable

1 Video Capture device (Capture card)

Get the latest commit from Github.

https://github.com/Mikael-og-Andre/Distributed-Instrument-Cluster/tree/Master

Visual Studio 2019 – (Used in development)

Web browser - Google Chrome or Firefox (These browsers were used in testing, and support Pointer lock APIs in the browser)

Video link: https://www.youtube.com/watch?v=ubFq-wV3Eic

OBS Open Broadcaster Software.

VLC.

Contents

Device start up tests

Test	Remote device start up 1
Test Description	Launch main program for remote device and see if program connected
Negative/Positive	Negative
Precondition	config.json exists in same folder as program. Crestron cable is connected to a running machine.
Test steps	Set “portName” variable in config.json file to “com”. Launch main program.
Expected	CLI warning: “Failed to connect to port: com” Then a list of available com ports.
Status	
Actual result	

Test	Remote device start up 2
Test Description	Launch main program for remote device and see if program connected
Negative/Positive	Positive
Precondition	Previous conditions.
Test steps	Set “portName” variable in config.json file to one of the com ports listed from the previous test. Launch main program.
Expected	CLI message “Successfully connected to port: {port selected}”
Status	
Actual result	

Test	Remote device start up 3
Test Description	Launch main program for remote device and see if program connected
Negative/Positive	Negative
Precondition	Previous conditions.
Test steps	Set “deviceIndex” variable in config.json file to 100. Launch main program.
Expected	CLI message: “Initializing video device100... No output from video device or no device found”
Status	
Actual result	

Test	Remote device start up 4
Test Description	Launch main program for remote device and see if program connected
Negative/Positive	Positive
Precondition	Previous conditions. Capture card is connected to a running machine. And machine is configured to output video to capture device.
Test steps	Set “deviceIndex” variable in config.json file to an integer corresponding to the capture card. Launch main program.

Expected	CLI message: “Initializing video device{device index}... Detecting frames from video device”
Status	
Actual result	

Web server tests

Test	Server test 1
Test Description	Test web server start up.
Negative/Positive	Negative
Precondition	remoteDevices.json contains no rememote devices.
Test steps	Launch web server, go to web site.
Expected	Web site says, “No device connected”.
Status	
Actual result	

Test	Server test 2
Test Description	Test web server start up.
Negative/Positive	Positive
Precondition	remoteDevices.json contains one remote device: "ip": "127.0.0.1", "name": "test", "location": "test", "type": "test", "VideoDevices": 1, "VideoBasePort": 8080, "hasCrestron": true, "CrestronBasePort": 6981 Remote_Server.exe is running with correct settings on test machine.
Test steps	Launch web server, go to web site.
Expected	Web shows device with information corresponding to json data.
Status	
Actual result	

Bandwidth tests

Test	Bandwidth test 1
Test Description	See bandwidth usage.
Negative/Positive	Positive
Precondition	Previous conditions. Connected device is capable of outputting full HD (1920:1080) at 30 fps or higher.
Test steps	config.json settings: “width” : 1920, “height” : 1080, “quality” : 90, “fps” : 30. Launch program. Open video on pc being captured, hd setting, full screen. Keep remote control website open to maintain video feed.
Expected	In windows Resource Monitor under Networking “Remote_Server.exe” has Total (B/sec) less than 20MiB/sec (168Mbps)
Status	
Actual result	

Test	Bandwidth test 2
Test Description	See bandwidth usage.
Negative/Positive	Positive
Precondition	Previous conditions.
Test steps	config.json settings: “width” : 1920, “height” : 1080, “quality” : 70, “fps” : 30. Launch program. Open video on pc being captured, hd setting, full screen. Keep remote control website open to maintain video feed.
Expected	In windows Resource Monitor under Networking “Remote_Server.exe” has Total (B/sec) less than 12MiB/sec (101Mbps)
Status	
Actual result	

Test	Bandwidth test 3
Test Description	See bandwidth usage.
Negative/Positive	Positive
Precondition	Previous conditions.
Test steps	config.json settings: “width” : 1920, “height” : 1080, “quality” : 30, “fps” : 30.

	Launch program. Open video on pc being captured, hd setting, full screen. Keep remote control website open to maintain video feed.
Expected	In windows Resource Monitor under Networking “Remote_Server.exe” has Total (B/sec) less than 7,5MiB/sec (63Mbps)
Status	
Actual result	

Test	Bandwidth test 4
Test Description	See bandwidth usage.
Negative/Positive	Positive
Precondition	Previous conditions.
Test steps	config.json settings: “width” : 1920, “height” : 1080, “quality” : 30, “fps” : 15. Launch program. Open video on pc being captured, hd setting, full screen. Keep remote control website open to maintain video feed.
Expected	In windows Resource Monitor under Networking “Remote_Server.exe” has Total (B/sec) less than 3,8MiB/sec (32Mbps)
Status	
Actual result	

Test	Bandwidth test 5
Test Description	See bandwidth usage.
Negative/Positive	Positive
Precondition	Previous conditions.
Test steps	config.json settings: “width” : 1280, “height” : 720, “quality” : 50, “fps” : 30. Launch program. Open video on pc being captured, hd setting, full screen. Keep remote control website open to maintain video feed.
Expected	In windows Resource Monitor under Networking “Remote_Server.exe” has Total (B/sec) less than 5,5MiB/sec (46Mbps)
Status	
Actual result	

Test	Bandwidth test 6
Test Description	See bandwidth usage.
Negative/Positive	Positive
Precondition	Previous conditions.

Test steps	config.json settings: “width” : 1280, “height” : 720, “quality” : 30, “fps” : 10. Launch program. Open video on pc being captured, hd setting, full screen. Keep remote control website open to maintain video feed.
Expected	In windows Resource Monitor under Networking “Remote_Server.exe” has Total (B/sec) less than 1,5MiB/sec (13Mbps)
Status	
Actual result	

Test	Bandwidth test 7
Test Description	See bandwidth usage.
Negative/Positive	Positive
Precondition	Previous conditions.
Test steps	config.json settings: “width” : 480, “height” : 360, “quality” : 50, “fps” : 15. Launch program. Open video on pc being captured, hd setting, full screen. Keep remote control website open to maintain video feed.
Expected	In windows Resource Monitor under Networking “Remote_Server.exe” has Total (B/sec) less than 0,5MiB/sec (4,2Mbps)
Status	
Actual result	

Test	Bandwidth test 8
Test Description	See bandwidth usage.
Negative/Positive	Positive
Precondition	Previous conditions.
Test steps	config.json settings: “width” : 480, “height” : 360, “quality” : 20, “fps” : 10. Launch program. Open video on pc being captured, hd setting, full screen. Keep remote control website open to maintain video feed.
Expected	In windows Resource Monitor under Networking “Remote_Server.exe” has Total (B/sec) less than 0,3MiB/sec (2,5Mbps)
Status	
Actual result	

Delay tests

Test	System delay (MJPEG)
Test Description	Test for delay on whole code base. Web site control to remote device and video of that control input back to user on website. Using MJPEG video streaming.
Negative/Positive	Positive
Precondition	Previous conditions. config.json settings: “width” : 1920, “height” : 1080, “quality” : 40, “fps” : 30. Server and remote device code running on same machine (no network delay). Using MJPEG video streaming. Machine running test can run everything with moderate CPU usage (less than 80%). Machine is capable of producing and recording 60 fps or more.
Test steps	Launch Blazor server and Remote_Server. Open website using Firefox or Chrome. Start recording screen using OBS at 60fps or higher! Take control of device on website, open notepad (on remote device) and type numbers 0-9 using virtual keyboard. Stop recording and open it in VLC, increment over frames using e key. Look at virtual keyboard for the first signs of a number key being pressed. Count the number of frames from first signs of a key press to when it appears in notepad on the website. Delay in ms = (1000/fps)*(number of frames between console and note pad), e.g. (1000/60)*9 = 150ms. Average the delay from all 10 numbers.
Expected	Average delay less than 200ms and max delay less than 300ms.
Status	
Actual result	

Test	System delay (H.264)
Test Description	Test for delay on whole code base. Web site control to remote device and video of that control input back to user on website. Using H.264 video streaming.
Negative/Positive	Positive
Precondition	TBD
Test steps	TBD
Expected	TBD
Status	
Actual result	

Stability tests

Test	Stability test 1
Test Description	Test long duration stability of web and remote server
Negative/Positive	Positive
Precondition	Correctly configured web and remote server.
Test steps	Launch web and remote server. Run both for 1h+.
Expected	No issues, no crashes, no memory leak, no “unusual” exception.
Status	
Actual result	

Tests performed:

Tests passed:

Signature:

Date:

D Appendix User Acceptance Test Procedure

User Acceptance Test
Distributed Instrument Cluster

Contents

Items used for the test.

Minimum 2 Computers

The remote device for the test should be a windows pc ideally.

1 Crestron cable

2 Video Capture device (Webcam or Capture card)

Get the latest commit from Github.

https://github.com/Mikael-og-Andre/Distributed-Instrument-Cluster/tree/Master

Visual Studio 2019 – (Used in development)

Web browser - Google Chrome or Firefox (These browsers were used in testing, and support Pointer lock APIs in the browser)

Scenario 1 Guest Login

Precondition: Backend web server is running

Test	Guest Login 001
Test Name	Enter Website
Test Description	Enter the URL into your Browser with /Login, and press enter.
Expected Result	The website page loads with the login fields and buttons in less than 10 seconds.
Actual Result	
Input Data	
Passed/Failed	

Test	Guest Login 002
Test Name	Login as Guest
Test Description	Press the login/enter as guest button on the bottom of the screen.
Expected Result	You are transferred to the device selection page.
Actual Result	
Input Data	
Passed/Failed	

Test	Guest Login 003
Test Name	Check profile Information
Test Description	In the top bar of the device selection page, left click the profile button, it looks like a face/human
Expected Result	A drop-down menu should appear showing your information as guest
Actual Result	
Input Data	
Passed/Failed	

Scenario 2 Select a device

Precondition: Logged in as a guest, on the device selection screen, and a remote device is connected to the server

Test	Select Device 001
Test Name	Refresh Connection List
Test Description	In the upper middle of the screen, there is a button that looks like 2 arrows, left click it to refresh the list of connections.
Expected Result	The page should display a loading animation, the loading should not last longer than 10 seconds.
Actual Result	
Input Data	
Passed/Failed	

Test	Select Device 002
Test Name	Check for a connected device
Test Description	If a device has connected, there should be a list of all connected devices in the middle of the device selection screen. confirm that the device shows up on the screen
Expected Result	A device should show on the screen with, the name, location, and type, and a button that says select, and if the connection has a Crestron, as well as the ping result.
Actual Result	
Input Data	
Passed/Failed	

Scenario 3 View video as guest

Precondition: Video device is outputting frames to the server, selected a Device, logged in as Guest

Test	View video as guest 001
Test Name	Functioning video
Test Description	In the middle of the screen there should now be a video stream
Expected Result	Video is showing on the webpage
Actual Result	
Input Data	
Passed/Failed	

Test	View video as guest 002
Test Name	No admin controls

Test Description	Check if there is a control bar below the video and no button that says Toggle Keyboard
Expected Result	There should not be a control bar when in guest mode.
Actual Result	
Input Data	
Passed/Failed	

Test	View video as guest 003
Test Name	Check delay
Test Description	Either use the direct control if you are capturing video from a pc, or wave your hand in front of the webcam, and then look at how long it takes to update.
Expected Result	The video delay should be less than 15 seconds
Actual Result	
Input Data	
Passed/Failed	

Test	View video as guest 004
Test Name	Swap Video Device
Test Description	In the top left above the video display, there should be a Selector dropdown saying, Device 0, left click it and select device 1.
Expected Result	The video stream should change to another video stream, e.g., another picture.
Actual Result	
Input Data	
Passed/Failed	

Scenario 4 Register account

Precondition: starting at the login page

Test	Register Account 001
Test Name	Go to registration page
Test Description	At the bottom of the screen left click on the button that says Submit

Expected Result	The page should change to the registration page. "/Register"
Actual Result	
Input Data	
Passed/Failed	

Test	Register Account 002
Test Name	Check email validation
Test Description	In the email field, write "hello", then left click the submit button
Expected Result	At the top of the screen, below the Register title, there is a red notification saying, "The email field is not a valid e-mail address."
Actual Result	
Input Data	
Passed/Failed	

Test	Register Account 003
Test Name	Test notifications
Test Description	In the text field called Email enter an email in the format "word@word.com", and left click the submit button
Expected Result	A red notification appears at the top of the screen with the following text. "the username field is required." "the password field is required." "the Confirm Password field is required"
Actual Result	
Input Data	
Passed/Failed	

Test	Register Account 004
Test Name	Fill in a username
Test Description	Enter a username into the username field, left click the submit button
Expected Result	At the top of the screen the notifications read "The password is required" "The confirm password field is required".
Actual Result	
Input Data	
Passed/Failed	

Test	Register Account 005
Test Name	Hidden Password
Test Description	Write hello in the password field
Expected Result	The text in the password field is obscured
Actual Result	
Input Data	
Passed/Failed	

Test	Register Account 006
Test Name	Test password check
Test Description	In the password field enter a password, now enter a different Password in the confirm password field. Left click the submit button.
Expected Result	A red notification appears at the top of the screen saying “confirm password and password do not match”
Actual Result	
Input Data	
Passed/Failed	

Test	Register Account 007
Test Name	Reset button
Test Description	Left click the reset button
Expected Result	All the fields have the text you put in removed
Actual Result	
Input Data	
Passed/Failed	

Test	Register Account 008
Test Name	Register an account
Test Description	Write a username in the username field, a valid email in the email field, write a password in the password field, write the same password in the confirm password field. Left click the submit button.
Expected Result	The registration should be successful, and you should be sent to the login page again.
Actual Result	

Input Data	
Passed/Failed	

Scenario 5 Account Login

Precondition: registered an account successfully, starting at the login page

Test	Account Login 001
Test Name	Empty Login
Test Description	Left click the submit button at the bottom of the screen with an empty username and password field
Expected Result	A Red notification appears at the top of the screen. “The username field is required.” “The password field is required.”
Actual Result	
Input Data	
Passed/Failed	

Test	Account Login 002
Test Name	Empty password
Test Description	Enter the username in the username field
Expected Result	A red notification appears. “The password field is required”
Actual Result	
Input Data	
Passed/Failed	

Test	Account Login 003
Test Name	Wrong password
Test Description	Enter the correct username in the username field, and enter a different password from when you registered
Expected Result	A red notification should appear denying you access
Actual Result	
Input Data	
Passed/Failed	

Test	Account Login 004
Test Name	Test a user
Test Description	Enter the username and password from the registration stage
Expected Result	You should be moved to the device selection page
Actual Result	
Input Data	
Passed/Failed	

Test	Account Login 005
Test Name	Check account information
Test Description	In the top right of your screen, left click the profile icon. Looks like face/human
Expected Result	A dropdown menu should appear below the button, showing that you are logged in with the email from the previous step
Actual Result	
Input Data	
Passed/Failed	

Scenario 6 Control Device

Precondition: Starting after a device is already selected, logged in to an account, remote device selected is a computer with capture card, and Crestron connected

Test	Control Device 001
Test Name	Connect to device
Test Description	Locate the button on the page that has the text “Request Control” and left click it.
Expected Result	After pressing the text “status: none” should update to display “status: Connecting”, “status: Requesting device”, “status: In queue pos 1” “status: Controlling”, You should reach the controlling state in less than 10 seconds is no other device is connected. And the status does not read “Status: Disconnected”.
Actual Result	
Input Data	
Passed/Failed	

Test	Control Device 002
Test Name	Lock mouse to device
Test Description	Locate the button that has the text “Lock mouse” and left click it
Expected Result	After pressing your mouse should disappear from the screen, and a popup at the top of the screen should read “press escape to unlock”
Actual Result	
Input Data	
Passed/Failed	

Test	Control Device 003
Test Name	Check mouse movement
Test Description	Drag your mouse to the right.
Expected Result	The video should update showing your mouse move to the right after max 3 seconds
Actual Result	
Input Data	
Passed/Failed	

Test	Control Device 004
Test Name	Check left click
Test Description	Hover the mouse cursor over desktop icon element and press the right mouse button once.
Expected Result	The element should respond to being clicked, with a highlight.
Actual Result	
Input Data	
Passed/Failed	

Test	Control Device 005
Test Name	Check right click
Test Description	Hover the mouse on the desktop background and press the right mouse button.
Expected Result	A dropdown menu appears.
Actual Result	
Input Data	
Passed/Failed	

Test	Control Device 006
Test Name	Open notepad on the remote computer
Test Description	Open a notepad or other application where you can type text and highlight it so you can start writing in the application.
Expected Result	A text application should be open on the remote device.
Actual Result	
Input Data	
Passed/Failed	

Test	Control Device 007
Test Name	Check normal letters
Test Description	On your keyboard tap the following letters q,w,e,r,t,y,u,I,o,p,a,s,d,f,g,h,j,k,l,z,x,c,v,b,n,m
Expected Result	The letters q,w,e,r,t,y,u,I,o,p,a,s,d,f,g,h,j,k,l,z,x,c,v,b,n,m have been entered in the text document on the remote device.
Actual Result	
Input Data	
Passed/Failed	

Test	Control Device 008

Test Name	Check shift key
Test Description	Hold down the shift key on your keyboard. Then press the key a. Release the shift key.
Expected Result	The letter A has appeared in the document in uppercase.
Actual Result	
Input Data	
Passed/Failed	

Test	Control Device 009
Test Name	Check backspace key
Test Description	Press and release the backspace key.
Expected Result	The A letter should be deleted from the text document.
Actual Result	
Input Data	
Passed/Failed	

Test	Control Device 010
Test Name	Check holding backspace
Test Description	Hold down the backspace key for 4 seconds
Expected Result	The letters will be erased from the text document.
Actual Result	
Input Data	
Passed/Failed	

Test	Control Device 011

Test Name	Check special keys
Test Description	Press and release the windows button
Expected Result	The windows menu should open.
Actual Result	
Input Data	
Passed/Failed	

Test	Control Device 012
Test Name	Check special key combos
Test Description	Press ctrl, alt, delete, at the same time
Expected Result	The device should enter a menu allowing you to logout or enter task manager, if you are on windows.
Actual Result	
Input Data	
Passed/Failed	

Scenario 7 Website keyboard

Precondition: Starting after a device is already selected, logged in to an account, remote device selected is a computer with capture card, and Crestron connected

Test	Website keyboard 001
Test Name	Connect to device
Test Description	Locate the button on the page that has the text “Request Control” and left click it.
Expected Result	After pressing the text “status: none” should update to display “status: Connecting”, “status: Requesting device”, “status: In quue pos 1” and after less than 5 seconds it should display “status: controlling”
Actual Result	
Input Data	
Passed/Failed	

Test	Website keyboard 002
Test Name	Open website keyboard
Test Description	Locate the button that has the text “Keyboard” and left click it
Expected Result	After pressing your mouse should disappear from the screen, and a popup at the top of the screen should read “press escape to unlock”
Actual Result	
Input Data	
Passed/Failed	

Test	Website keyboard 003
Test Name	Open notepad on remote computer
Test Description	Open a notepad or other application where you can type text and highlight it so you can start writing in the application.
Expected Result	A text application should be open on the remote device.
Actual Result	
Input Data	
Passed/Failed	

Test	Website keyboard 004
Test Name	Check normal letters
Test Description	On the website keyboard tap the following letters q,w,e,r,t,y,u,I,o,p,a,s,d,f,g,h,j,k,l,z,x,c,v,b,n,m
Expected Result	The letters q,w,e,r,t,y,u,I,o,p,a,s,d,f,g,h,j,k,l,z,x,c,v,b,n,m have been entered in the text document on the remote device, and in the google chrome console as “a” followed by “a break”. If they appeared in uppercase, press the caps lock key, and repeat the step.
Actual Result	
Input Data	

Passed/Failed	

Test	Website keyboard 005
Test Name	Check shift key, with hold down key support
Test Description	Hold down the shift key on the website keyboard, you “hold down a key” by right clicking it with your mouse, the button should get a darker shade to show it as clicked. Then press the key a. Release the shift key.
Expected Result	The letter A has appeared in the document in uppercase. In the google chrome console, there is a log saying “shift” and “shift break”
Actual Result	
Input Data	
Passed/Failed	

Test	Website keyboard 006
Test Name	Check backspace key
Test Description	Press and release the backspace key on the website keyboard.
Expected Result	The A letter should be deleted from the text document. In the google chrome console there is a backspace log
Actual Result	
Input Data	
Passed/Failed	

Test	Website keyboard 007
Test Name	Check holding down backspace
Test Description	Hold down the backspace key for 4 seconds on the website keyboard
Expected Result	The letters will be erased from the text document. In the google chrome console, A “backspace” log appears when you press the key down, and a “backspace break” appears when you release the key
Actual Result	

Input Data	
Passed/Failed	

Test	Website keyboard 008
Test Name	Check special key combos
Test Description	Right click ctrl and alt, then left click delete, by using the right click functionality specified earlier.
Expected Result	The device should enter a menu allowing you to logout or enter task manager, if you are on windows.
Actual Result	
Input Data	
Passed/Failed	

Tests performed:

Tests passed:

Signature:

Date:

E Appendix Meeting Notes

2021-01-13 First meeting with arne styve

Date

13 Jan 2021

Attendees

- Mikael Nilssen
- Andre Helland
- Arne Styve (adm)

Goals

- Discuss meeting plans
- Get a heading for the project

Discussion items

Kontak person: Lars ole hurlen, norvald kjerstad

Avvik:

Missed deadlines

Bugs

Missed scope

Finne ut om api og programmeringsmuligheter.

Arbeidskontrakt:

Spør norvald

“Hvor synes du vi skal begynne?”

Møte oppdrags giver.

“Hvor ofte skal vi logge?”

Velg selv.

“Hvor ofte vil du ha møte?”

Anna hver fredg

Action items

- [x] Mikael Nilssen Contact Norvald Kjerstad and schedule a meeting
- [x] Arne Styve (adm) lager confluence og kaller inn til møte

2021-01-18 First meeting with Norvald Kjerstad

Date

18 Jan 2021

Attendees

- Mikael Nilssen
- Andre Helland
- Norvald kjerstad

Goals

- Gather information about requirments and needs for the project
- Specify potential ui wishes

Discussion items

Furuno Contact:

Bjørnli julibø - 92083010

Lars:

Lars.ole.hurlen@mrfylke.no

Topics covered.

Administering - Find at later date further in the project.

UI - Want the ui to be most simillar to the hardware side as possible, due to it being used in a learning environment.

Replay - Wanted feature of having the ability to record and replay a radar scanning.

Student collisons - Queue feature or something simillar to handle multiple students accessing the same hardware.

Budget- further discussion with lars.

Radar warmup- when turned on the radar has a Warmup period of 3 min

Controls - THere is a custom keyboard for the radar. potental of creating a web version.

Action items

- ✓ Mikael Nilssen Contact Lars Ole Hurlen

2021-01-19 First Meeting With Lars Ole

Date

19 Jan 2021

Attendees

- Mikael Nilssen
- Andre Helland
- Lars ole Hurlen

Goals

- Gather information about requirments and needs for the project

Discussion items

Topics covered:

Keyboard emulation – emulate byte code for keyboard using serial to USB adapter cable.

Capture monitor – use capture card to capture video output from radar pc.

Video quality – prioritize video quality over frame rate.

Security – network security has low priority (no sensitive information).

Hosting location – cloud hosting (AWS).

Reliability – system needs high reliability making windows unsuited as a hosting OS.

Spectating – potential future addition allowing for spectating others using the radar system on the web site.

Action items

- [x] Start first sprint

2021-01-25 Meeting Lars Ole Check up 1

Date

25 Jan 2021

Attendees

- Andre Helland
- Mikael Nilssen
- Lars Ole

Goals

- Check up on proejct progress.
- Discuss license

Discussion items

- Employment contract and source code license - Use any Open Source License (MIT)
- Version control (github?)
- Work packages and sub-goals
- Presentation and preliminary project
    o Possibly contribute images from our simulator and our instrument if you feel it can help communicate the goal of the project

He was not displeased.

UDP does not work over internet.

Action items

- ✓ Meeting with Lars Ole 12:00 26/01/2021

2021-01-29 Arne Styve Check up 1

Date

29 Jan 2021

Attendees

- Mikael Nilssen
- Andre Helland
- Arne Styve (adm)

Goals

- Get input on how progress and documentation is working

Discussion items

Item	Notes
Epic	• Use epics as big features eks "Remote Radar Control"
Story and task use	• Use features and subtasks
Forprosjekt rapport + slutt innlevering?	• Will be in the final hand in
Jira time estimate?	• Arne gives admin access
Mixed language?	• Ok mixing (not in the same document)
Work log?	• Ok with just jira
Road map	• more subgoals • less Software spesific bars • Add subgoals for project end • Padding

Action items

- ✓ Arne Styve (adm) Grant Admin Access / Fix time logging
- ✓ Arne Styve (adm) Fix editable backlog
- ✓ Arne Styve (adm) Give feedback on Forprosjektsrapport / Powerpoint
- ✓ Mikael Nilssen Andre Helland Fix backlog, Fix Veikart

2021-02-8 Meeting Lars Ole check up 2

Date

09 Feb 2021

Attendees

- Andre Helland
- Mikael Nilssen
- Lars Ole

Goals

- Test crestron code on radar system.

2021-02-12 Arne Check up 2

Date

12 Feb 2021

Attendees

- Andre Helland
- Mikael Nilssen
- Arne Styve

Goals

- Get input on licenses
- Get a checkup

Discussion items

License:

http://www.aforgenet.com/framework/license.html

https://ffmpeg.xabe.net/license.html

Scrum:

Use case diagram style issues and epics.

Action items

- [x] Mikael Nilssen Andre Helland Fix Backlog Again, use verbs

2021-02-26 Arne Checkup 3

Date

26 Feb 2021

Attendees

- Mikael Nilssen
- Andre Helland
- Arne Styve

Goals

- Status update

Discussion items

Thread use is fine, consumer provider is okay.

Showed demo

Actions should be used in retrospective meeting for what to improve on

Action items

- [ ]

2021-03-07 Lars Ole Demo View and code review

Date

07 Mar 2021

Attendees

- Mikael Nilssen
- Andre Helland

Goals

Get insight via code review

Show demo to lars ole

Discussion items

Time	Item	Notes
10 min	Research MJEPG	better resource managment
10 min	Folder structure	• Src, Demo, Unit
5 min	Test lav båndbredde	Test hva som kjer om klient har lav båndbredde
5 min	Test CPU-Tid	
5 min	Style guide	▪ Reharper or another styleguide
5 min	Use SOLID	
5 min	Swap Field Comments	Use field comments that connect to the code
5 min	Documentation Generation	
5 min	Separate inline code from .razor	
5 min	Use Packaging, and reuse principles	
5 min	Connect Todo items to Jira issues	

Action items

- [x] Mikael Nilssen Andre Helland Restructure folders

2021-03-12 Arne Styve Checkup 4

Date

12 Mar 2021

Attendees

- Mikael Nilssen
- Arne Styve
- Andre Helland

Goals

- Get insight into progress, and get comments on bachelor

Discussion items

- Skriv større møte referat
- Bruk versjoner
- Git Tags for å hvis hvilke commit som tilhører forskjellige mål i versjoner og releases
- Lag releases hver sprint fremover
- Dokumenter At det kommer til å være mindre arbeid PGA Eksamen i kommende sprint
- Snakket om gitflow og branch work

Action items

- [x] Arne Styve Gi Tilgang til Versjoner
- [x] Mikael Nilssen Andre Helland Tag Git Branch
- [x] Mikael Nilssen Andre Helland Lag eksempel bachelor template

2021-03-15 Pre Sprint 9 Meeting

Date

15 Mar 2021

Attendees

- Mikael Nilssen
- Andre Helland

Goals

- Disscuss how to handle reduced work due to Upcoming Exam on the 25th of march

Discussion items

Expected work on the upcoming 2 sprints will be heavily reduced, we are assuming 2-3 days of work per sprint, with the rest being used for studying for the upcoming exam,

We did try to overload Sprint 8 inorder to get ahead on progress and reduce impact. but we also decided to refactor some thigns so we ended up kinda even.

We will focus on getting some cahnges ready for the furuno meeting for testing.

Action items

- [ ]

2021-03-19 Meeting notes Furuno Testing

Date

19 Mar 2021

Attendees

- Mikael Nilssen
- Andre Helland

Goals

- Test Capture card and video device

Discussion items

Wanted features discussed:

Allowed packetloss - Aka dont require the system to receive all frames

Potential MultiScreen support

Stop Start Remotely

Many systems have internal delays already

Wanted a demo to show on their own website

Resulst from testing

TECDIS - Windows xp

Test:

Keyboard Mouse - Success

Video - Success

FMD-3200 - Linux

Test:

Keyboard mouse - Success

Video - Success*

Had to use the main monitors video deivce for the capture to function

RPU-025 - Linux

Test:

Video - Success

Keyboard mouse - Failed

The usb port did accept a normal HP mouse, but the crestron cable could not access

Action items

- [ ]

2021-03-26 Arne Checkup 5

Date

26 Mar 2021

Attendees

- Mikael Nilssen
- Andre Helland
- Arne Styve

Goals

- Discuss product delivery/hand over
- Ask about usability testing

Discussion items

Time	Item	Notes
	• Product delivery/hand over	• Brukerveiledning • Skermbilder og forklaringer
5min	• Usability testing	• Web based, will allow for testing during covid • Make good questions, specific questions • Make haste in delivery

Action items

- [x] Mikael Nilssen Andre Helland Setup proper readme

2021-04-09 Arne Checkup 6

Date

09 Apr 2021

Attendees

- Mikael Nilssen
- Andre Helland
- Arne Styve

Goals

- Get Input on Bachelor Template
- Ask what should be in the method and what should be theory section

Discussion items

Describe Pedagogy with learning instruments and how it might improve things

Programming lanugage and .net moved to methods

Move More relevant things to our specific project up in the list content

Conclusion short and concise

Result describe acheivements

Disscussion Describe reasonings and pointers given by lars ole, and decisions he recommanded or pre decided

Action items

- [ ]

2021-04-12 Sprint checkup w/ Lars Ole

Date

12 Apr 2021

Attendees

- Andre Helland
- Mikael Nilssen
- Lars Ole Hurlen

Goals

- Plan Sprint With Lars ole

Discussion items

Priority this sprint, Create Procedure testing

Hardware encoding for video compression (bachelor text).

Del løsning specification i bachelor.

Test: Factory acceptance test, internal acceptance test, customer acceptance test.

Fokus på hvordan man kan bytte ut "komponenter i et prosjekt".

Action items

- [ ]

2021-04-19 Sprint Checkup w/ lars ole 2

Date

19 Apr 2021

Attendees

- Mikael Nilssen
- Andre Helland
- Lars ole hurlen

Goals

- Get feedback on previous sprint
- Set new priorities

Discussion items

Reformat Test procedures to one table per test.

Priority install windows on laptops.

Development setup description.

Prioritize stability over preformance.

Feature Broadcast pausing.

Feature priority Web keyboard.

Feature priority h264.

Action items

- [ ]

2021-04-23 Arne Styve checkup 7

Date

23 Apr 2021

Attendees

- Mikael Nilssen
- Arne Styve
- Andre Helland

Goals

- What uml diagrams should we use
- Where to include Story point miss

Discussion items

What Diagrams should we have?

Class Diagram for business logic

Sequence Diagram

Component Diagram

Deployment Diagram

Use case Diagram

State diagram

Use visual Paradigm.

Use of time estimation versus storypoints should be included in the method section, result section, and discussion section.

In the results define why using storypoints could have been better, also refere to a research paper about the topic if found.

Use HTTPS only.

Action items

- [ ]

2021-04-26 Checkup meeting w/ lars ole 3

Date

26 Apr 2021

Attendees

- Mikael Nilssen
- Andre Helland
- Lars ole

Goals

- Set new priorities

Discussion items

Prioritize finishing Networking.

Estimation updating cahnge scope during work.

Log more frequently.

Add Serialization for future arduino device control.

Add Database features

Action items

- [ ]

2021-05-03 Sprint checkup 4 w/ Lars Ole , Arne Styve

Date

03 May 2021

Attendees

- Mikael Nilssen
- Arne Styve
- Andre Helland
- Lars ole Hurlen

Goals

- Priority check
- HTTPS
- Show progress

Discussion items

Prioritize stability

Switch remote device to independent server

Action items

- [ ]

2021-05-07 Arne checkup 8

Date

07 May 2021

Attendees

- Andre Helland
- @Arne Styve

Goals

- Discuss thesis writing.

Discussion items

Auto generated class diagram needs cleaning.

Rapport mal.

Deleivery (disscuss future work med Lars Ole og Norvald potentialy).

Action items

- [ ]

2021-05-10 Sprint checkup 5 w/ Lars Ole

Date

10 May 2021

Attendees

- Andre Helland
- Mikael Nilssen
- Lars Ole

Goals

- Show progress.
- Discuss what to work on.

Discussion items

Progress report and results:

Very good results,

New setup at fagskolen for testing will arrive after the bachelor date.

Work to complete, Bachelor, and some kind of crud for the remote device management

Action items

- [ ]

2021-05-17 Planning for Norvald meeting

Date

17 May 2021

Attendees

- Mikael Nilssen
- Andre Helland

Goals

- Have plan

Discussion items

Show him a demo of the project.

Discuss handover and manual's.

Get opinon of result, is it a success?

Discuss further development.

Action items

- [ ]

2021-05-18 Show Case w/ Norvald and lars ole

Date

18 May 2021

Attendees

- Mikael Nilssen
- Andre Helland

Goals

- Show product to norvald
- performe basic test

Discussion items

Was the project a success: Yes

Usability of the project is good, he was able to navigate the solution reliably.

Have a good user documentation in the result table export as pdf separately.

Action items

- [ ]

F Appendix Sprint Reports

Sprint Report

Switch report

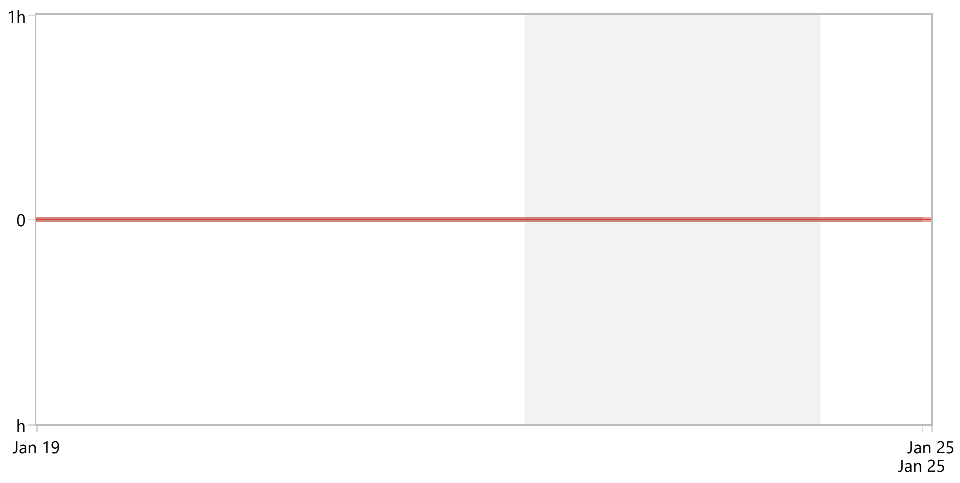
INSTCLUS Sprint 1

COMPLETED Progress on Forprosjektrapport and powerpoint

Details View linked pages

Started: 19/Jan/21 4:36 PM by Mikael Nilssen (planned - 18/Jan/21 4:33 PM)

Ended: 25/Jan/21 5:59 PM by Mikael Nilssen (planned - 25/Jan/21 4:33 PM)



Status Report

* Issue added to sprint after start time

Completed Issues
View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (-)
INSTCLUST-1	Product Requirment	Story	Medium	DONE	-
INSTCLUST-20	Project planning / gannt chart	Story	High	DONE	-
INSTCLUST-29	UI Diagrams	Story	Medium	DONE	-

Issues Not Completed
View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (-)
INSTCLUST-2	Forprosjektsrapport	Task	Highest	TO DO	-
INSTCLUST-3 *	Research topics	Task	Medium	TO DO	-
INSTCLUST-23	Create Library diagram UML	Task	Low	TO DO	-

Sprint Report

Switch report

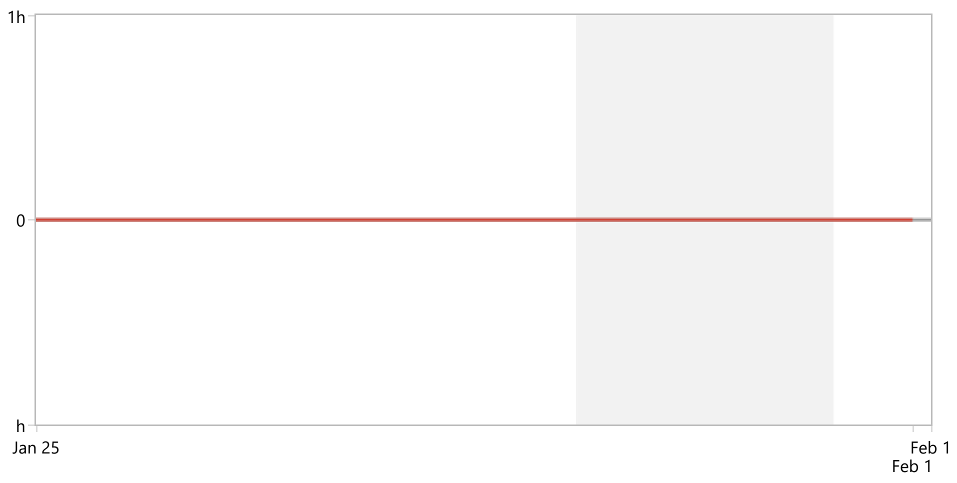
INSTCLUS Sprint 2

COMPLETED Complete Planning Stage

Details View linked pages

Started: 25/Jan/21 7:24 PM by Mikael Nilssen (planned - 25/Jan/21 6:05 PM)

Ended: 01/Feb/21 2:38 PM by Mikael Nilssen (planned - 01/Feb/21 6:05 PM)



Status Report

Completed Issues

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (-)
INSTCLUST-2	Forprosjektsrapport	✓ Task	Highest	DONE	-
INSTCLUST-3	Research topics	✓ Task	Medium	DONE	-
INSTCLUST-39	Powerpoint presentasjon about project	✓ Task	Highest	DONE	-
INSTCLUST-41	System Architecture Diagrams/Research	✓ Task	Medium	DONE	-
INSTCLUST-42	Risk assessment	✓ Task	Medium	DONE	-
INSTCLUST-43	Research FFMPEG	✓ Task	Medium	DONE	-

Issues Not Completed

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (-)
INSTCLUST-23	Create Library diagram UML	✓ Task	Low	TO DO	-

Sprint Report

Switch report

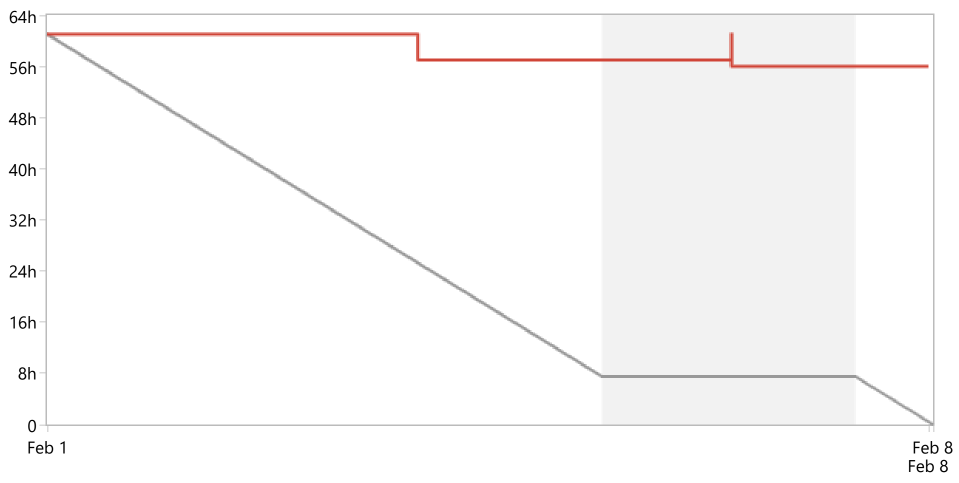
INSTCLUS Sprint 3

COMPLETED Start Coding and get a rythm

Details View linked pages

Started: 01/Feb/21 2:47 PM by Mikael Nilssen (planned - 01/Feb/21 2:43 PM)

Ended: 08/Feb/21 1:51 PM by Mikael Nilssen (planned - 08/Feb/21 2:43 PM)



Status Report

Completed Issues

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (5h)
INSTCLUST-74	Research Keyboard Emulator	Task	Medium	DONE	5h

Issues Not Completed

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (1w 2d)
INSTCLUST-23	Create Library diagram UML	Task	Low	TO DO	2h
INSTCLUST-46	Implement Crestron Library - Control Keyboard on Remtoe device	Story	High	TO DO	3d
INSTCLUST-56	Implement Video Connection Library - Read Video	Story	High	TO DO	3d
INSTCLUST-72	Research Relu unit	Task	Medium	TO DO	2h
INSTCLUST-73	Research Capture card	Task	Medium	TO DO	4h

Sprint Report

Switch report

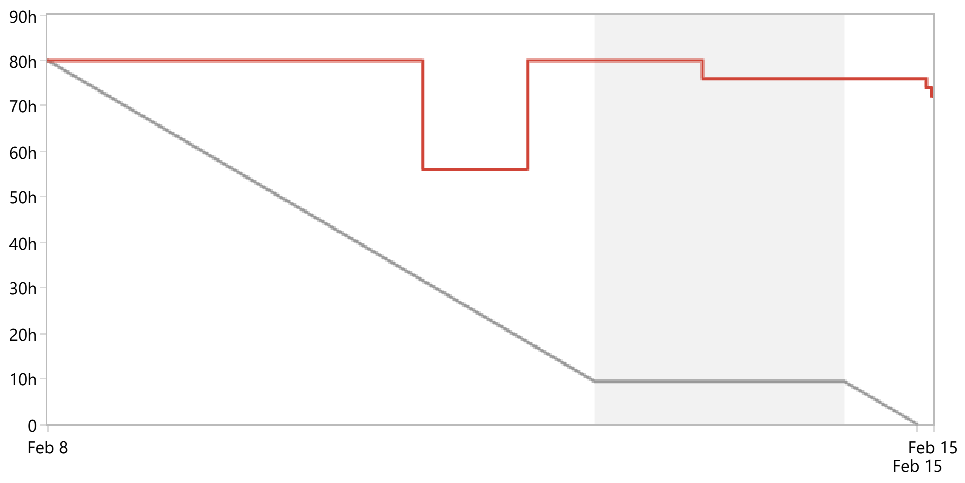
INSTCLUS Sprint 4

COMPLETED Complete Crestron Features And Working Video Connection Library

Details View linked pages

Started: 08/Feb/21 2:50 PM by Mikael Nilssen (planned - 08/Feb/21 2:00 PM)

Ended: 15/Feb/21 5:02 PM by Mikael Nilssen (planned - 15/Feb/21 2:00 PM)



Status Report

Completed Issues

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (4d)
INSTCLUST-23	Create Library diagram UML	Task	Low	DONE	2h
INSTCLUST-46	Implement Crestron Library - Control Keyboard on Remtoe device	Story	High	DONE	3d
INSTCLUST-72	Research Relu unit	Task	Medium	DONE	2h
INSTCLUST-73	Research Capture card	Task	Medium	DONE	4h

Issues Not Completed

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (1w 1d → 1w 4d)
INSTCLUST-53	Implement Basic Communication Library - Send Data	Story	Medium	TO DO	2d → 1w
INSTCLUST-56	Implement Video Connection Library - Read Video	Story	High	TO DO	3d
INSTCLUST-76	Have meeting with furuno	Task	Low	TO DO	1d

Sprint Report   Switch report

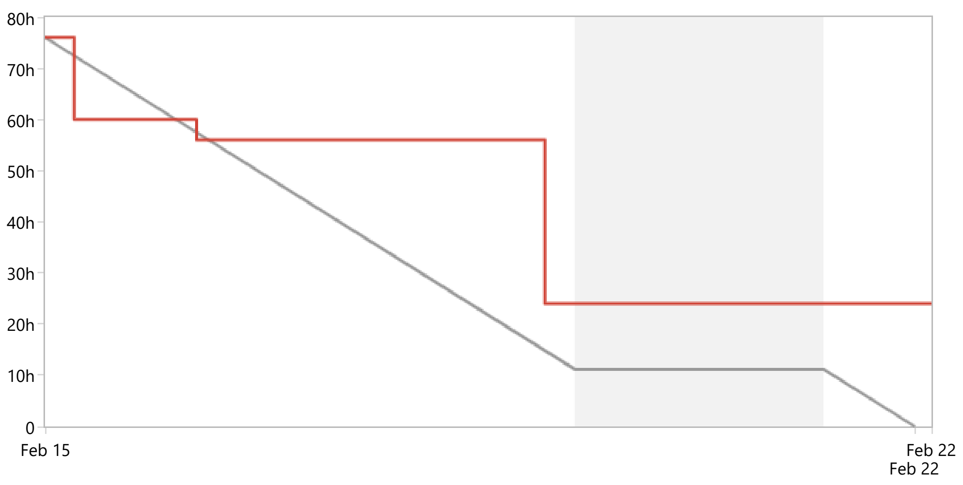
INSTCLUS Sprint 5

COMPLETED

Details   View linked pages

Started: 15/Feb/21 5:53 PM by Mikael Nilssen (planned - 15/Feb/21 5:39 PM)

Ended: 22/Feb/21 8:49 PM by Mikael Nilssen (planned - 22/Feb/21 5:39 PM)



Status Report

Completed Issues

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (1w 1d 4h)
INSTCLUST-53	Implement Basic Communication Library - Send Data	Story	Medium	DONE	4d
INSTCLUST-56	Implement Video Connection Library - Read Video	Story	High	DONE	2d
INSTCLUST-78	Refactor serial interface	Story	High	DONE	4h

Issues Not Completed

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (3d)
INSTCLUST-77	Create Main program for hardware side	Story	High	TO DO	1d
INSTCLUST-79	Parse user commands	Story	High	IN PROGRESS	1d
INSTCLUST-80	Create Simple test website	Story	High	TO DO	1d

Sprint Report

Switch report

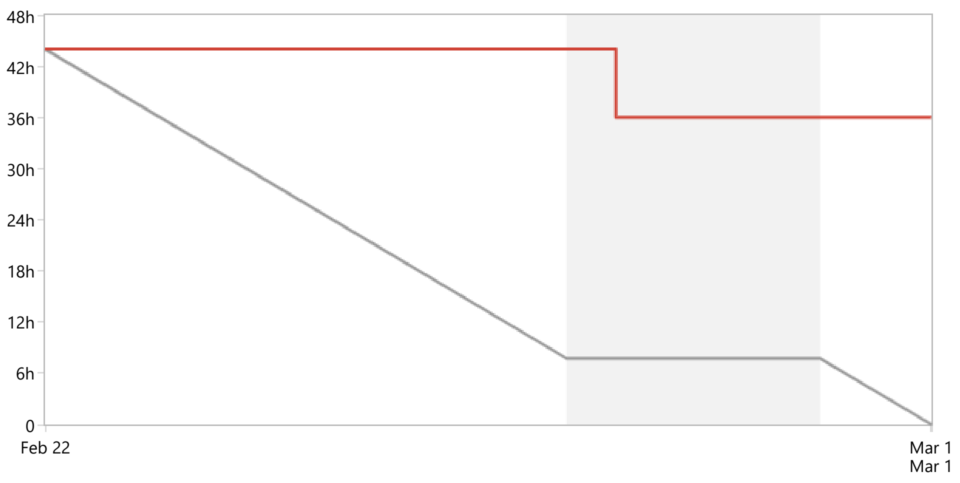
INSTCLUS Sprint 6

COMPLETED Get a demo website

Details View linked pages

Started: 22/Feb/21 9:11 PM by Mikael Nilssen (planned - 22/Feb/21 9:06 PM)

Ended: 01/Mar/21 9:01 PM by Mikael Nilssen (planned - 01/Mar/21 9:06 PM)



Status Report

Issues Not Completed

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (4d 4h)
INSTCLUST-77	Create Main program for hardware side	Story	High	IN PROGRESS	1d
INSTCLUST-79	Parse user commands	Story	High	IN PROGRESS	1d
INSTCLUST-80	Create Simple test website	Story	High	TO DO	2d
INSTCLUST-90	Setup Docker Support	Task	Low	TO DO	4h

Issues Removed From Sprint

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (1d)
INSTCLUST-76	Have meeting with furuno	Task	Low	TO DO	1d

Sprint Report   Switch report

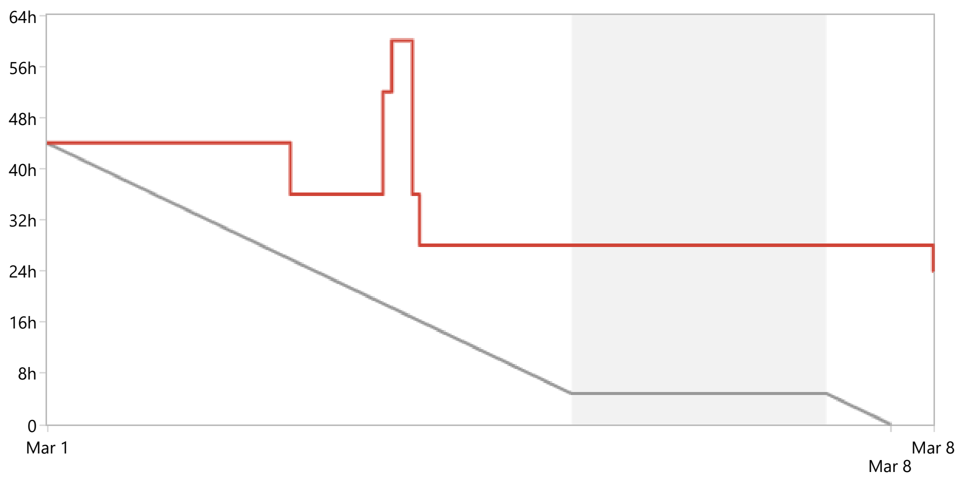
INSTCLUS Sprint 7

COMPLETED   Finish MVP Website, and start on further features

Details   View linked pages

Started: 01/Mar/21 9:12 PM by Mikael Nilssen (planned - 01/Mar/21 12:10 PM)

Ended: 08/Mar/21 8:10 PM by Mikael Nilssen (planned - 08/Mar/21 12:10 PM)



Status Report

* Issue added to sprint after start time

Completed Issues

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (4d 4h)
INSTCLUST-77	Create Main program for hardware side	Story	High	DONE	1d
INSTCLUST-79	Parse user commands	Story	High	DONE	1d
INSTCLUST-80	Create Simple test website	Story	High	DONE	2d
INSTCLUST-90	Setup Docker Support	Task	Low	DONE	4h

Issues Not Completed

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (2d → 3d)
INSTCLUST-69 *	Implement Backend Web Server - Make Transfere of video / Controls to Backend	Story	Medium	TO DO	2d → 3d

Issues Removed From Sprint

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (1d)
INSTCLUST-76	Have meeting with furuno	Task	Low	TO DO	1d

Sprint Report

Switch report

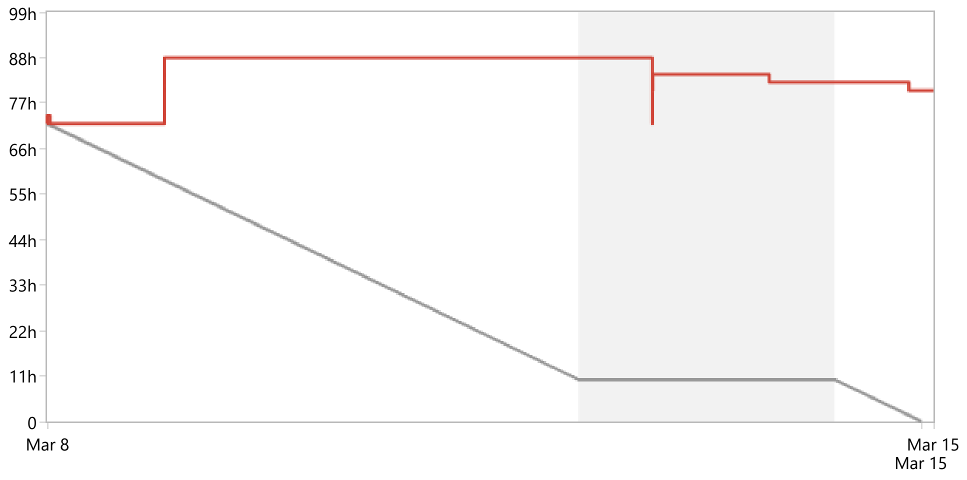
INSTCLUS Sprint 8

COMPLETED Extend Features and Reduce Current Tech debt. Polish the Minimum MVP

Details View linked pages

Started: 08/Mar/21 8:27 PM by Mikael Nilssen (planned - 08/Mar/21 4:18 PM)

Ended: 15/Mar/21 6:29 PM by Mikael Nilssen (planned - 15/Mar/21 4:18 PM)



Status Report

* Issue added to sprint after start time

Completed Issues
View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (- → 2d 6h)
INSTCLUST-106 *	Restructure Project folders	Story	Medium	DONE	- → 2h
INSTCLUST-107 *	Refactor Instument Communicator library	Story	Medium	DONE	- → 2d
INSTCLUST-109 *	Unit Test Server Library	Story	Medium	DONE	- → 2h
INSTCLUST-110 *	Unit Test Networking Library	Story	Medium	DONE	- → 2h

Issues Not Completed
View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (1w 4d → 2w)
INSTCLUST-52	Implement Better MJPEG	Story	Highest	TO DO	2d
INSTCLUST-68	Implement Web Assembly Video UI - Make web interface for Video	Story	Medium	TO DO	2d
INSTCLUST-69	Implement Backend Web Server - Make Transfere of video / Controls to Backend	Story	Medium	TO DO	3d
INSTCLUST-103	Implement Configuration File reading for setup	Story	Medium	IN PROGRESS	1d
INSTCLUST-104	Reduce Tech Debt by completing todos	Story	Low	IN PROGRESS	1d
INSTCLUST-108 *	Refactor Blazor Backend Listener Services	Story	Medium	IN PROGRESS	- → 1d

Sprint Report

Switch report

INSTCLUS Sprint 9

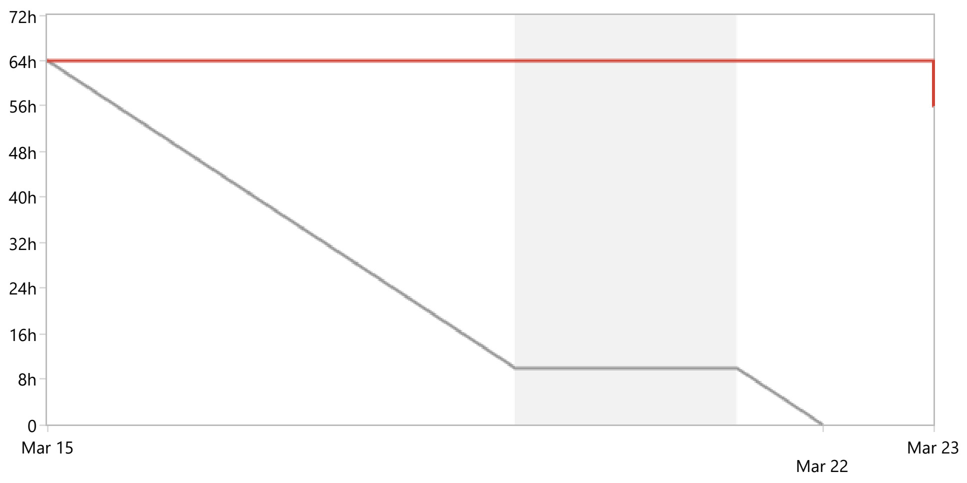
...

COMPLETED   Prepare for meeting and showcase with furuno

Details   View linked pages

Started: 15/Mar/21 6:39 PM by Mikael Nilssen (planned - 15/Mar/21 6:37 PM)

Ended: 23/Mar/21 6:39 PM by Mikael Nilssen (planned - 22/Mar/21 6:37 PM)



Status Report

Completed Issues

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (1d)
INSTCLUST-76	Have meeting with furuno	☑ Task	Low	**DONE**	1d

Issues Not Completed

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (1w 2d)
INSTCLUST-68	Implement Web Assembly Video UI - Make web interface for Video	Story	= Medium	**TO DO**	2d
INSTCLUST-69	Implement Backend Web Server - Make Transfere of video / Controls to Backend	Story	= Medium	**TO DO**	3d
INSTCLUST-103	Implement Configuration File reading for setup	Story	= Medium	**IN PROGRESS**	1d
INSTCLUST-108	Refactor Blazor Backend Listener Services	Story	= Medium	**IN PROGRESS**	1d

Sprint Report Switch report

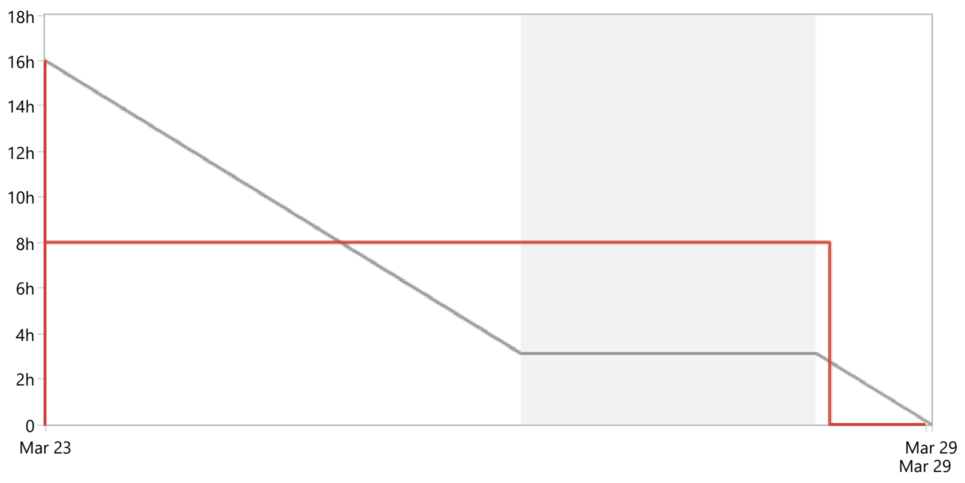
INSTCLUS Sprint 10

COMPLETED   Start process of Compression and optimization

Details View linked pages

Started: 23/Mar/21 6:47 PM by Mikael Nilssen (planned - 22/Mar/21 6:45 PM)

Ended: 29/Mar/21 5:44 PM by Mikael Nilssen (planned - 29/Mar/21 6:45 PM)



Status Report

* Issue added to sprint after start time

Completed Issues

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (3d)
INSTCLUST-68	Implement Web Assembly Video UI - Make web interface for Video	Story	Medium	DONE	2d
INSTCLUST-108 *	Refactor Blazor Backend Listener Services	Story	Medium	DONE	1d

Sprint Report

Switch report

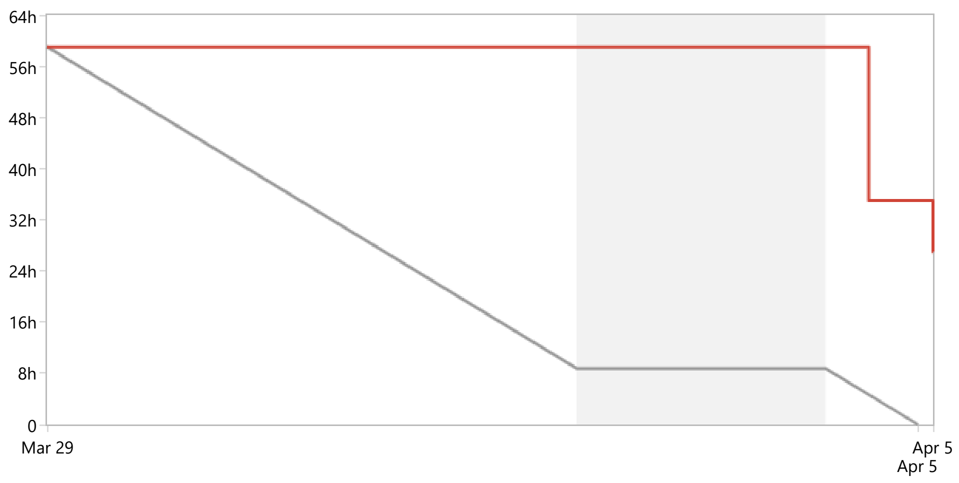
INSTCLUS Sprint 11

COMPLETED   Implement Compression

Details   View linked pages

Started: 29/Mar/21 5:56 PM by Mikael Nilssen (planned - 29/Mar/21 5:50 PM)

Ended: 05/Apr/21 8:43 PM by Mikael Nilssen (planned - 05/Apr/21 5:50 PM)



Status Report

Completed Issues

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (4d)
INSTCLUST-69	Implement Backend Web Server - Make Transfere of video / Controls to Backend	Story	Medium	DONE	3d
INSTCLUST-103	Implement Configuration File reading for setup	Story	Medium	DONE	1d

Issues Not Completed

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (3d 3h)
INSTCLUST-52	Implement Better MJPEG	Story	Highest	TO DO	2d
INSTCLUST-104	Reduce Tech Debt by completing todos	Story	Low	IN PROGRESS	1d
INSTCLUST-111	Create Template for Bachelor	Task	Medium	TO DO	3h

Sprint Report   Switch report

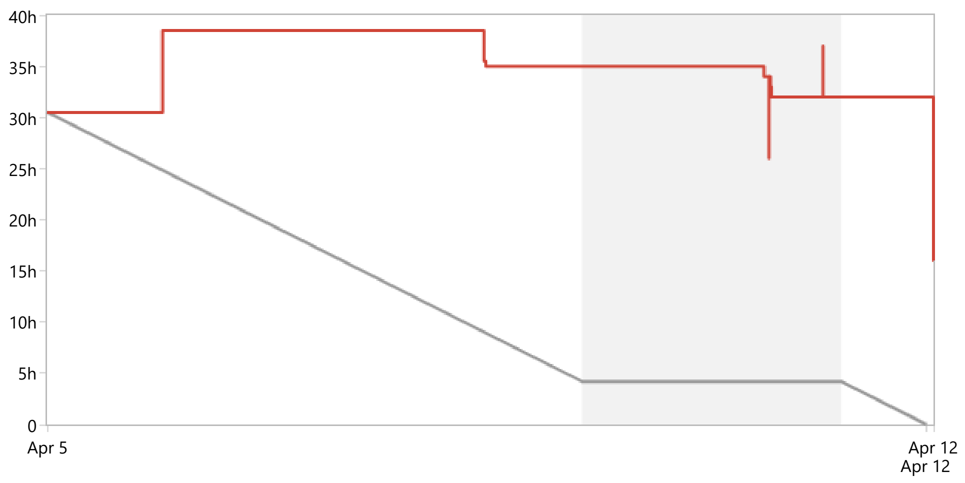
INSTCLUS Sprint 12

COMPLETED   Finish Compression and video work, And work on Bachelor

Details   View linked pages

Started: 05/Apr/21 8:57 PM by Mikael Nilssen (planned - 05/Apr/21 3:49 PM)

Ended: 12/Apr/21 5:06 PM by Mikael Nilssen (planned - 12/Apr/21 3:49 PM)



Status Report

* Issue added to sprint after start time

Completed Issues
View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (3d 3h 30m → 3d 2h 30m)
INSTCLUST-52	Implement Better MJPEG	Story	Highest	DONE	2d
INSTCLUST-111	Create Template for Bachelor	Task	Medium	DONE	3h
INSTCLUST-114	Write Basic Introduction For bachelor	Task	Medium	DONE	2h → 1h
INSTCLUST-115	Write Basic Terminology page for bachelor	Task	Medium	DONE	1h
INSTCLUST-116	Write Frontpage of bachelor	Task	Medium	DONE	30m
INSTCLUST-125 *	Refactor Server and Networking Library And Web Backend to use byte arrays	Task	Medium	DONE	5h

Issues Not Completed
View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (1d → 2d)
INSTCLUST-104	Reduce Tech Debt by completing todos	Story	Low	IN PROGRESS	1d
INSTCLUST-124 *	Finish incomplete work on whole solution.	Story	Highest	IN PROGRESS	- → 1d

Sprint Report

Switch report

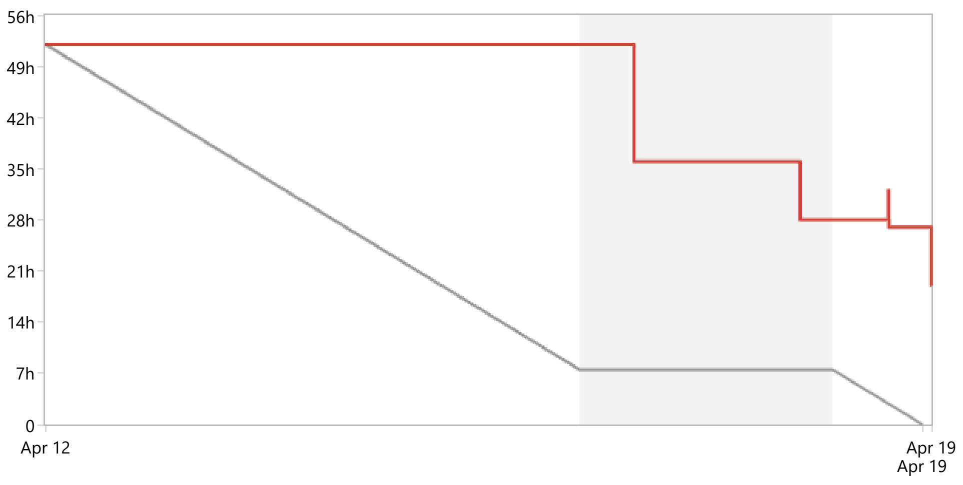
INSTCLUS Sprint 13

COMPLETED Complete Procedures

Details View linked pages

Started: 12/Apr/21 6:34 PM by Mikael Nilssen (planned - 12/Apr/21 5:06 PM)

Ended: 19/Apr/21 6:46 PM by Mikael Nilssen (planned - 19/Apr/21 5:06 PM)



Status Report

* Issue added to sprint after start time

Completed Issues

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (4d 4h)
INSTCLUST-104	Reduce Tech Debt by completing todos	Story	Low	DONE	1d
INSTCLUST-124	Finish incomplete work on whole solution.	Story	Highest	DONE	1d
INSTCLUST-128	As a User i want to have a test procedure so that i can ensure the quality and features of the software	Story	Medium	DONE	2d
INSTCLUST-133 *	As a user i have a user interface for login and registration	Story	Medium	DONE	4h

Issues Not Completed

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (2d 3h)
INSTCLUST-118	Write Basic Theory for bachelor	Task	Medium	TO DO	3h
INSTCLUST-127	As a Developer i want to have an internal test procedure so that i can measure the quality and output of the software	Story	Medium	IN PROGRESS	2d

Issues Removed From Sprint

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (1h)
INSTCLUST-117	Write Summary basic for bachelor	Task	Medium	TO DO	1h

Sprint Report

Switch report

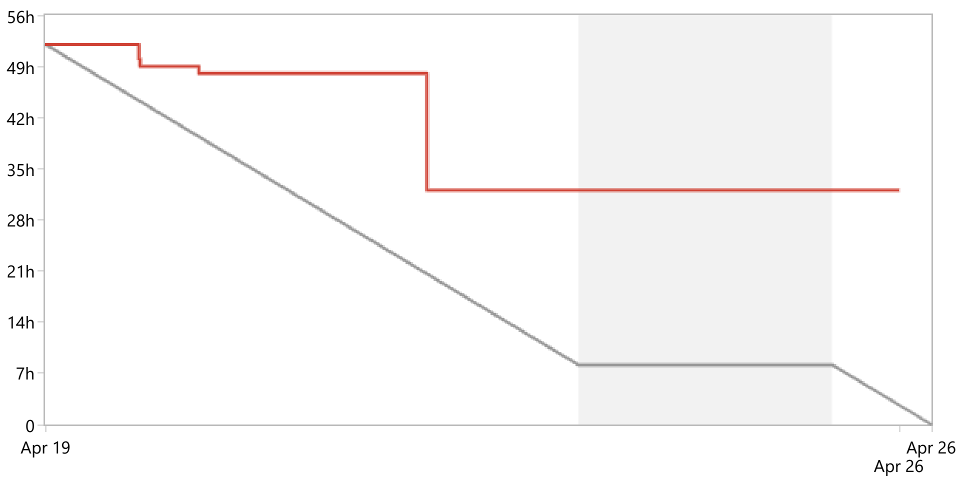
INSTCLUS Sprint 14

COMPLETED Stability in remote program

Details View linked pages

Started: 19/Apr/21 7:09 PM by Mikael Nilssen (planned - 19/Apr/21 6:46 PM)

Ended: 26/Apr/21 12:44 PM by Mikael Nilssen (planned - 26/Apr/21 6:46 PM)



Status Report

Completed Issues

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (2d 4h)
INSTCLUST-127	As a Developer i want to have an internal test procedure so that i can measure the quality and output of the software	Story	Medium	DONE	2d
INSTCLUST-137	Create A Development Setup Description	Task	Medium	DONE	2h
INSTCLUST-138	Reformat User acceptance tests	Task	Medium	DONE	1h
INSTCLUST-144	Fix bug in video stream where the dynamic url has an extra slash	Task	Medium	DONE	1h

Issues Not Completed

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (4d)
INSTCLUST-123	As a user i should have a virtual keyboard	Story	Medium	IN PROGRESS	1d
INSTCLUST-141	As a product owner i should have h264 streaming	Story	Medium	TO DO	1d
INSTCLUST-145	As a product owner my product should be stable	Story	Medium	TO DO	2d

Sprint Report   Switch report

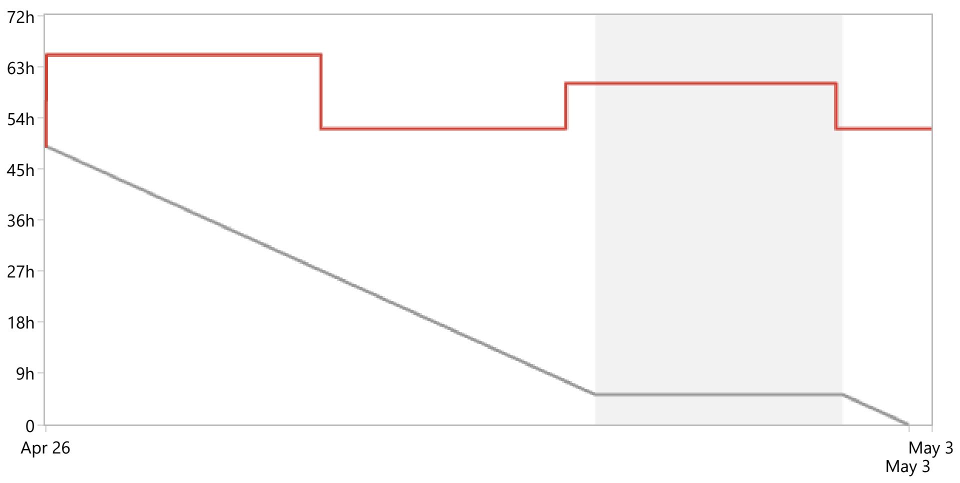
INSTCLUS Sprint 15

COMPLETED   Complete Networking and Virtual Keyboard, Finish Login and auth features

Details   View linked pages

Started: 26/Apr/21 1:07 PM   by Mikael Nilssen (planned - 26/Apr/21 12:45 PM)

Ended: 03/May/21 5:14 PM by Mikael Nilssen (planned - 03/May/21 12:45 PM)



Status Report

* Issue added to sprint after start time

Completed Issues
View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (1d 5h → 2d 5h)
INSTCLUST-123	As a user i should have a virtual keyboard	Story	Medium	DONE	5h → 1d 5h
INSTCLUST-145	As a product owner my product should be stable	Story	Medium	DONE	1d

Issues Not Completed
View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (1w 1d 4h)
INSTCLUST-95	As a user i should be able to login	Story	Medium	TO DO	1d
INSTCLUST-96	As a connecting device i should be able to identify myself	Story	High	TO DO	1d
INSTCLUST-97	As a user i should be able to see my account details	Story	Medium	TO DO	4h
INSTCLUST-141	As a product owner i should have h264 streaming	Story	Medium	IN PROGRESS	1d
INSTCLUST-150	Common for 95 and 96	Story	High	TO DO	1d
INSTCLUST-154 *	As a user i should be able to turn devices on and off	Story	Medium	TO DO	1d
INSTCLUST-155 *	Make MJPEG stream HTTPS	Task	Medium	IN PROGRESS	1d

Sprint Report

Switch report

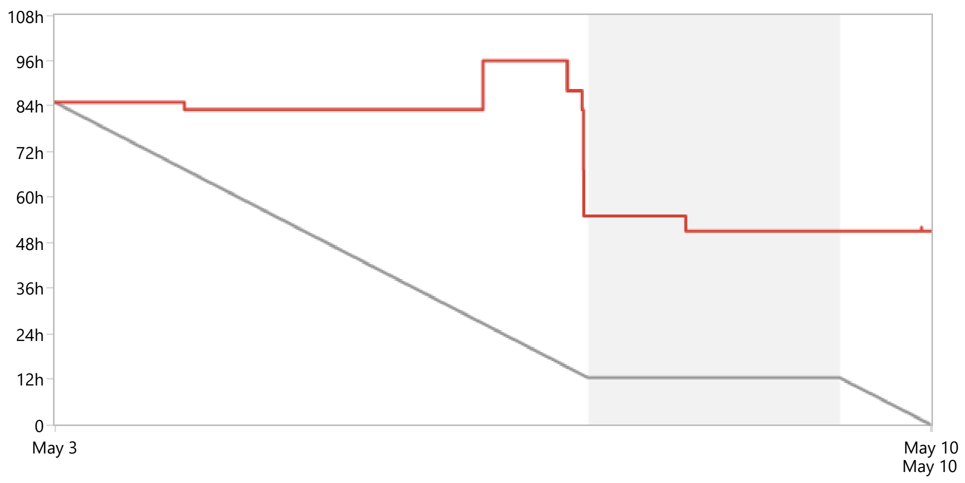
INSTCLUS Sprint 16

COMPLETED Finish The most possbile

Details View linked pages

Started: 03/May/21 6:12 PM by Mikael Nilssen (planned - 03/May/21 5:15 PM)

Ended: 10/May/21 5:26 PM by Mikael Nilssen (planned - 10/May/21 5:15 PM)



Status Report

* Issue added to sprint after start time

Completed Issues

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (3d 2h → 2d 4h)
INSTCLUST-156	BUG Delay when controlling	Task	Medium	DONE	1d
INSTCLUST-158	Refactor Config Json files	Task	Medium	DONE	2h
INSTCLUST-159	Remote device should be independent	Story	High	DONE	2d → 4h
INSTCLUST-160 *	Docker Container Runnable	Task	Medium	DONE	- → 5h
INSTCLUST-161 *	Run Speed limites stability test	Task	Medium	DONE	- → 1h

Issues Not Completed

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (1w 3h → 1w 1d 3h)
INSTCLUST-95	As a user i should be able to login	Story	Medium	TO DO	4h
INSTCLUST-96	As a connecting device i should be able to identify myself	Story	High	TO DO	4h
INSTCLUST-97	As a user i should be able to see my account details	Story	Medium	TO DO	4h
INSTCLUST-117	Write Summary basic for bachelor	Task	Medium	TO DO	4h
INSTCLUST-118	Write Basic Theory for bachelor	Task	Medium	IN PROGRESS	3h → 2d
INSTCLUST-150	Common for 95 and 96	Story	High	TO DO	1d → 3h
INSTCLUST-157	Write Planned feature descriptions	Task	Medium	TO DO	2d

Issues Removed From Sprint
View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (2d)
INSTCLUST-119	Write Basic Method for bachelor	✓ Task	= Medium	TO DO	2d

Sprint Report

Switch report

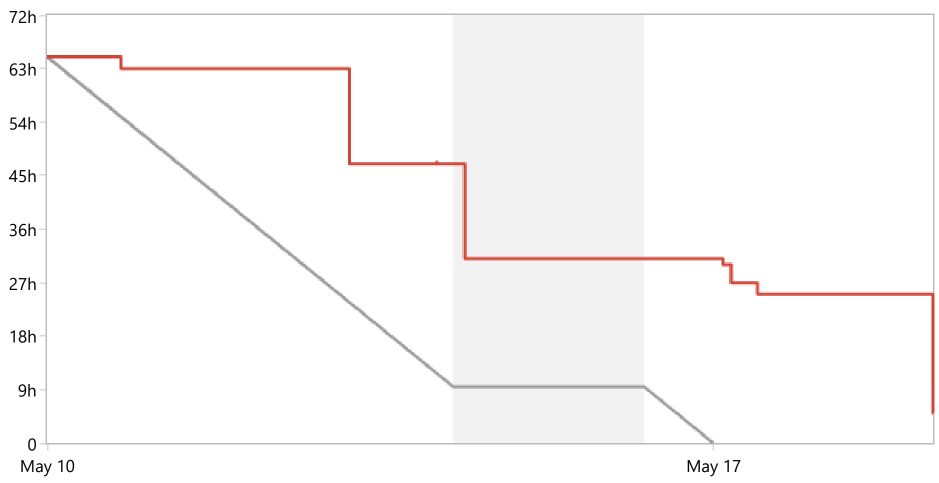
INSTCLUS Sprint 17

ACTIVE Last spurt

Details View linked pages

Started: 10/May/21 5:42 PM by Mikael Nilssen (planned - 10/May/21 5:27 PM)

Ended: (planned - 17/May/21 5:27 PM)



Status Report

* Issue added to sprint after start time

Completed Issues

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (1w 2d 4h → 1w 2d 4h 20m)
INSTCLUST-117	Write Summary basic for bachelor	✓ Task	= Medium	DONE	4h
INSTCLUST-118	Write Basic Theory for bachelor	✓ Task	= Medium	DONE	2d
INSTCLUST-119	Write Basic Method for bachelor	✓ Task	= Medium	DONE	2d
INSTCLUST-157	Write Planned feature descriptions	✓ Task	= Medium	DONE	2d
INSTCLUST-162	Create Readme	✓ Task	= Medium	DONE	2h
INSTCLUST-163	Update Acceptance Test	✓ Task	= Medium	DONE	1h
INSTCLUST-164	Update Internal Test	✓ Task	= Medium	DONE	3h
INSTCLUST-166	Network Topology Diagram	✓ Task	= Medium	DONE	2h
INSTCLUST-171 *	Json RemoteDevice Loading	✓ Task	= Medium	DONE	- → 20m
INSTCLUST-172 *	Write Results Chapter	✓ Task	= Medium	DONE	-
INSTCLUST-173 *	Write Discussion Chapter	✓ Task	= Medium	DONE	-
INSTCLUST-174 *	Write Conclusion Chapter	✓ Task	= Medium	DONE	-

Issues Not Completed

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (5h)
INSTCLUST-165	Create UML Diagrams	✓ Task	= Medium	IN PROGRESS	5h
INSTCLUST-175 *	Add all appendecies	✓ Task	= Medium	IN PROGRESS	-

G Appendix Epic Burndown

Epic Burndown

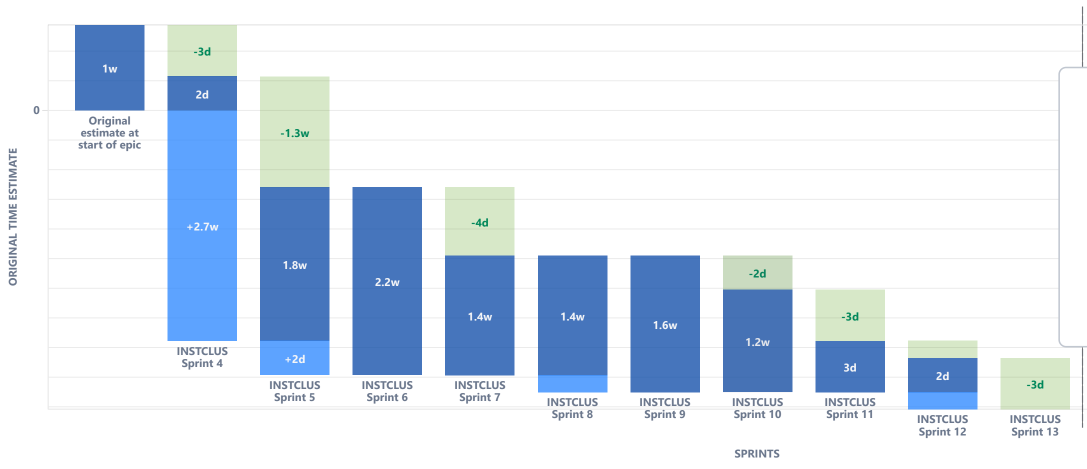
Switch report

INSTCLUST-44: User must be able to control...

View linked pages

0% unestimated issues **4w 2d 4h of 4w 2d 4h** completed (original time estimate)

🟩	Work completed	🟦	Work remaining
⬜	Work forecast	🟦	Work added



All issues are done!

Viewing options

☐ Align sprints at the base of the chart

Use this view to see trends in the scope for the epic.

Epic

INSTCLUST-44 User must be able to control and view remote devices

Completed Issues

INSTCLUS Sprint 13 12/Apr/21 - 19/Apr/21
View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (3d)
INSTCLUST-52	Implement Better MJPEG	Story	Highest	DONE	2d
INSTCLUST-124	Finish incomplete work on whole solution.	Story	Highest	DONE	1d

INSTCLUS Sprint 12 05/Apr/21 - 12/Apr/21
View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (1d)
INSTCLUST-103	Implement Configuration File reading for setup	Story	Medium	DONE	1d

INSTCLUS Sprint 11 29/Mar/21 - 05/Apr/21
View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (3d)
INSTCLUST-69	Implement Backend Web Server - Make Transfere of video /	Story	Medium	DONE	3d

	Controls to Backend				

INSTCLUS Sprint 10 22/Mar/21 - 29/Mar/21
View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (2d)
INSTCLUST-68	Implement Web Assembly Video UI - Make web interface for Video	Story	Medium	DONE	2d

INSTCLUS Sprint 9 15/Mar/21 - 23/Mar/21

No issues completed this sprint for current epic.

INSTCLUS Sprint 8 08/Mar/21 - 15/Mar/21

No issues completed this sprint for current epic.

INSTCLUS Sprint 7 01/Mar/21 - 08/Mar/21
View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (4d)
INSTCLUST-79	Parse user commands	Story	High	DONE	1d
INSTCLUST-80	Create Simple test website	Story	High	DONE	2d
INSTCLUST-77	Create Main program for hardware side	Story	High	DONE	1d

INSTCLUS Sprint 6 22/Feb/21 - 01/Mar/21

No issues completed this sprint for current epic.

INSTCLUS Sprint 5 15/Feb/21 - 22/Feb/21
View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (1w 1d 4h)
INSTCLUST-56	Implement Video Connection Library - Read Video	Story	High	DONE	2d
INSTCLUST-78	Refactor serial interface	Story	High	DONE	4h
INSTCLUST-53	Implement Basic Communication Library - Send Data	Story	Medium	DONE	4d

INSTCLUS Sprint 4 08/Feb/21 - 15/Feb/21
View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (3d)
INSTCLUST-46	Implement Crestron Library - Control Keyboard on Remtoe device	Story	High	DONE	3d

Epic Burndown

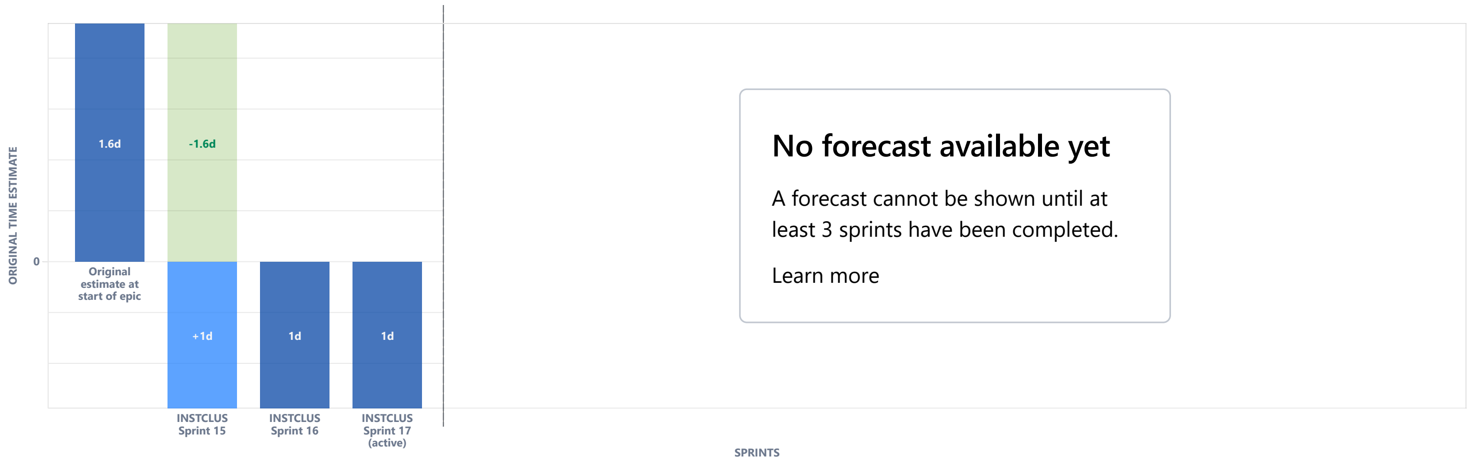
Switch report

INSTCLUST-139: Website Control

View linked pages

🟩 Work completed   🟦 Work remaining
⬜ Work forecast   🟦 Work added

0% unestimated issues   **1d 5h of 2d 5h** completed (original time estimate)



No forecast available yet

A forecast cannot be shown until at least 3 sprints have been completed.

Learn more

Viewing options

☐ Align sprints at the base of the chart

Use this view to see trends in the scope for the epic.

Epic

INSTCLUST-139 Create a keyboard on the website that can use features blocked by the pointer lock api, like the windows key

Completed Issues

INSTCLUS Sprint 17 10/May/21 - 17/May/21

No issues completed this sprint for current epic.

INSTCLUS Sprint 16 03/May/21 - 10/May/21

No issues completed this sprint for current epic.

INSTCLUS Sprint 15 26/Apr/21 - 03/May/21
View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (1d 5h)
INSTCLUST-123	As a user i should have a virtual keyboard	Story	Medium	DONE	1d 5h

Incomplete Issues

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (1d)
INSTCLUST-154	As a user i should be able to turn devices on and off	Story	Medium	TO DO	1d

Epic Burndown

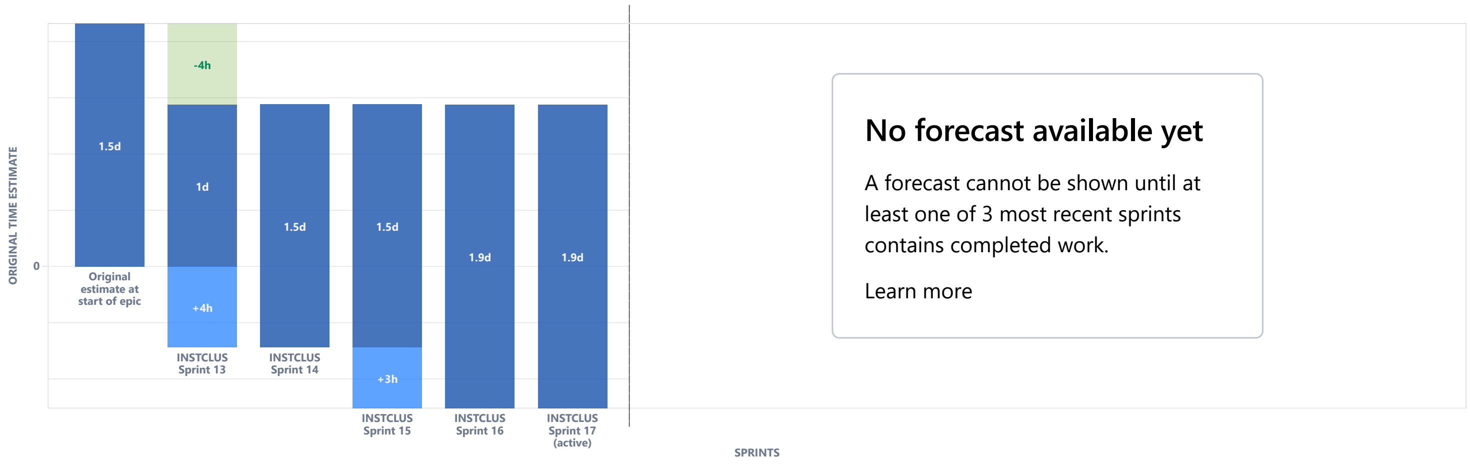
Switch report

INSTCLUST-45: User Login

View linked pages

0% unestimated issues **4h of 2d 3h** completed (original time estimate)

🟩 Work completed	🟦 Work remaining
⬜ Work forecast	🟦 Work added



No forecast available yet

A forecast cannot be shown until at least one of 3 most recent sprints contains completed work.

Learn more

Viewing options

☐ Align sprints at the base of the chart

Use this view to see trends in the scope for the epic.

Epic

INSTCLUST-45 Queue feature that removes collisons of controls. Login system with database. Locking features behind admin walls

Completed Issues

INSTCLUS Sprint 17 10/May/21 - 17/May/21

No issues completed this sprint for current epic.

INSTCLUS Sprint 16 03/May/21 - 10/May/21

No issues completed this sprint for current epic.

INSTCLUS Sprint 15 26/Apr/21 - 03/May/21

No issues completed this sprint for current epic.

INSTCLUS Sprint 14 19/Apr/21 - 26/Apr/21

No issues completed this sprint for current epic.

INSTCLUS Sprint 13 12/Apr/21 - 19/Apr/21

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (4h)
INSTCLUST-133	As a user i have a user interface for login and registration	Story	Medium	DONE	4h

Incomplete Issues

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (1d 7h)
INSTCLUST-96	As a connecting device i should be able to identify myself	Story	High	TO DO	4h
INSTCLUST-95	As a user i should be able to login	Story	Medium	TO DO	4h
INSTCLUST-97	As a user i should be able to see my account details	Story	Medium	TO DO	4h
INSTCLUST-150	Common for 95 and 96	Story	High	TO DO	3h

Epic Burndown

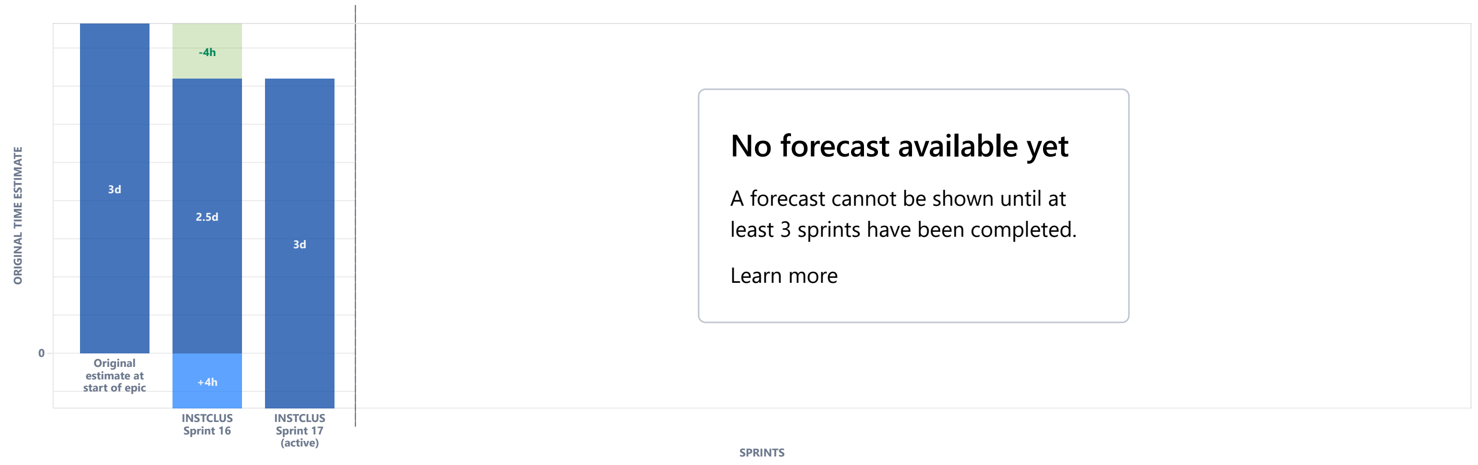
Switch report

INSTCLUST-140: Improved Bandwidth

View linked pages

🟩 Work completed 🟦 Work remaining
⬜ Work forecast 🟦 Work added

33% unestimated issues **4h of 3d 4h** completed (original time estimate)



No forecast available yet

A forecast cannot be shown until at least 3 sprints have been completed.

Learn more

Viewing options

☐ Align sprints at the base of the chart

Use this view to see trends in the scope for the epic.

Epic

INSTCLUST-140 Improve compression and network communication

Completed Issues

INSTCLUS Sprint 17 10/May/21 - 17/May/21

No issues completed this sprint for current epic.

INSTCLUS Sprint 16 03/May/21 - 10/May/21
View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (4h)
INSTCLUST-159	Remote device should be independent	Story	High	DONE	4h

Incomplete Issues

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (3d)
INSTCLUST-141	As a product owner i should have h264 streaming	Story	Medium	IN PROGRESS	3d
INSTCLUST-169	As a product owner i should have lower bandwidth	Story	Medium	TO DO	-

Epic Burndown

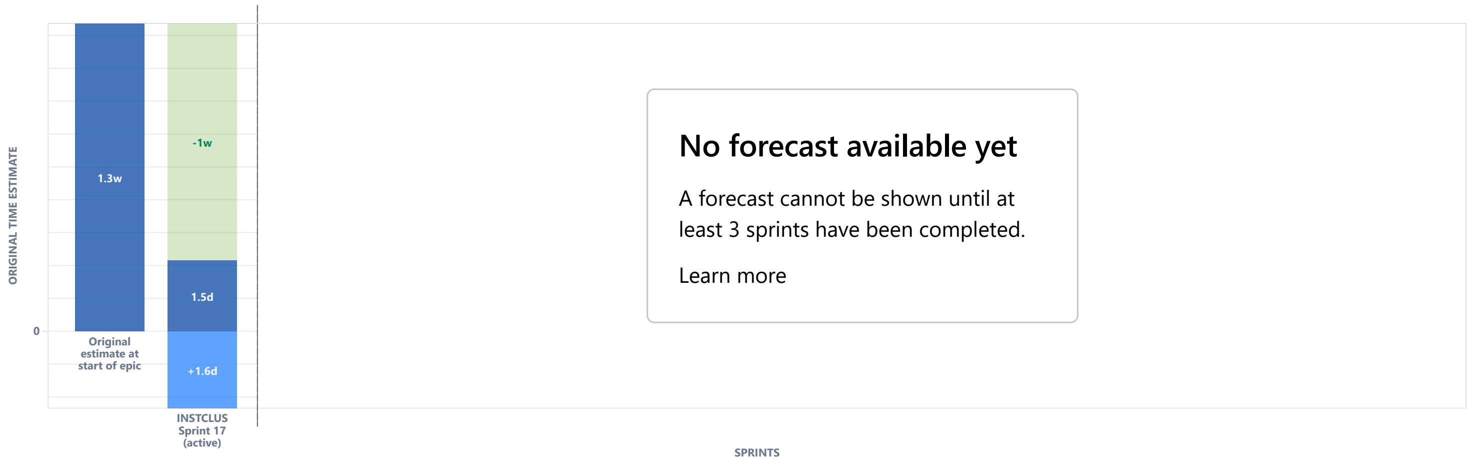
Switch report

INSTCLUST-129: Bachelor Thesis

View linked pages

🟩 Work completed	🟦 Work remaining
⬜ Work forecast	🟦 Work added

31% unestimated issues **1w of 1w 3d 1h** completed (original time estimate)



No forecast available yet

A forecast cannot be shown until at least 3 sprints have been completed.

Learn more

Viewing options

☐ Align sprints at the base of the chart

Use this view to see trends in the scope for the epic.

Epic

INSTCLUST-129 Write Bachelor

Completed Issues

INSTCLUS Sprint 17 10/May/21 - 17/May/21

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (1w)
INSTCLUST-166	Network Topology Diagram	✓ Task	= Medium	DONE	2h
INSTCLUST-118	Write Basic Theory for bachelor	✓ Task	= Medium	DONE	2d
INSTCLUST-119	Write Basic Method for bachelor	✓ Task	= Medium	DONE	2d
INSTCLUST-163	Update Acceptance Test	✓ Task	= Medium	DONE	1h
INSTCLUST-164	Update Internal Test	✓ Task	= Medium	DONE	3h
INSTCLUST-172	Write Results Chapter	✓ Task	= Medium	DONE	-
INSTCLUST-162	Create Readme	✓ Task	= Medium	DONE	2h

Incomplete Issues

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (3d 1h)

INSTCLUST-117	Write Summary basic for bachelor	✓ Task	= Medium	TO DO	4h
INSTCLUST-157	Write Planned feature descriptions	✓ Task	= Medium	IN PROGRESS	2d
INSTCLUST-165	Create UML Diagrams	✓ Task	= Medium	IN PROGRESS	5h
INSTCLUST-173	Write Discussion Chapter	✓ Task	= Medium	IN PROGRESS	-
INSTCLUST-174	Write Conclusion Chapter	✓ Task	= Medium	TO DO	-
INSTCLUST-175	Add all appendecies	✓ Task	= Medium	IN PROGRESS	-

Epic Burndown

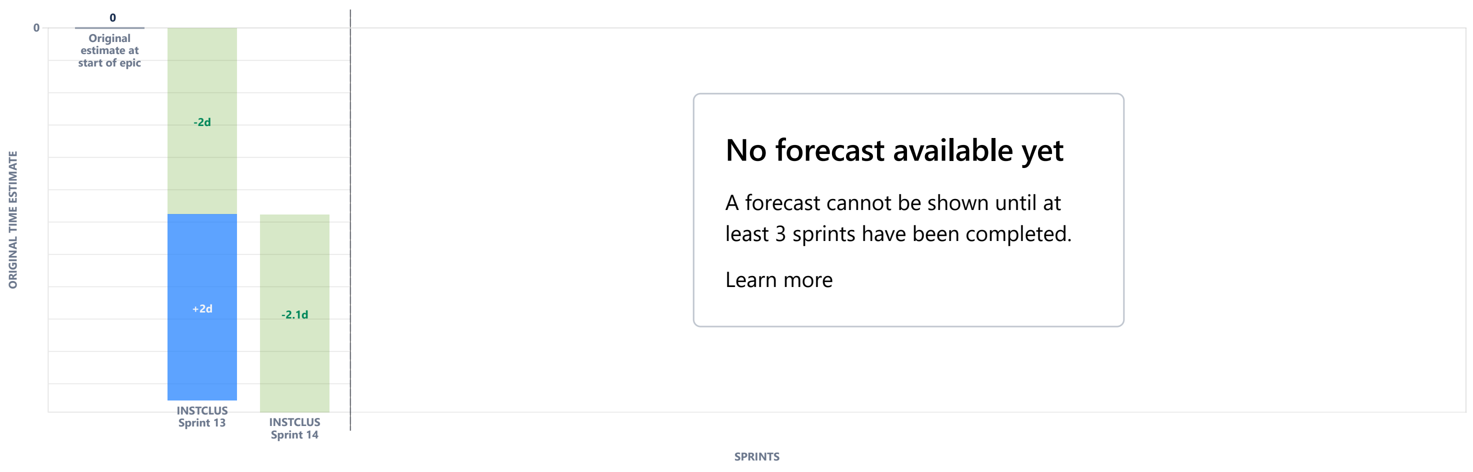
Switch report ⌄

INSTCLUST-126: Test Procedures

View linked pages

🟩 Work completed ⬜ Work forecast 🟦 Work remaining 🟦 Work added

0% unestimated issues **4d 1h of 4d 1h** completed (original time estimate)



No forecast available yet

A forecast cannot be shown until at least 3 sprints have been completed.

Learn more

Viewing options

☐ Align sprints at the base of the chart

Use this view to see trends in the scope for the epic.

Epic

INSTCLUST-126 Internal test procedure for quality testing, User acceptance test documenting functionality and Customer acceptance

Completed Issues

INSTCLUS Sprint 14 19/Apr/21 - 26/Apr/21
View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (2d 1h)
INSTCLUST-138	Reformat User acceptance tests	☑ Task	= Medium	DONE	1h
INSTCLUST-127	As a Developer i want to have an internal test procedure so that i can measure the quality and output of the software	Story	= Medium	DONE	2d

INSTCLUS Sprint 13 12/Apr/21 - 19/Apr/21
View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (2d)
INSTCLUST-128	As a User i want to have a test procedure so that i can ensure the quality and features of the software	Story	= Medium	DONE	2d

Epic Burndown

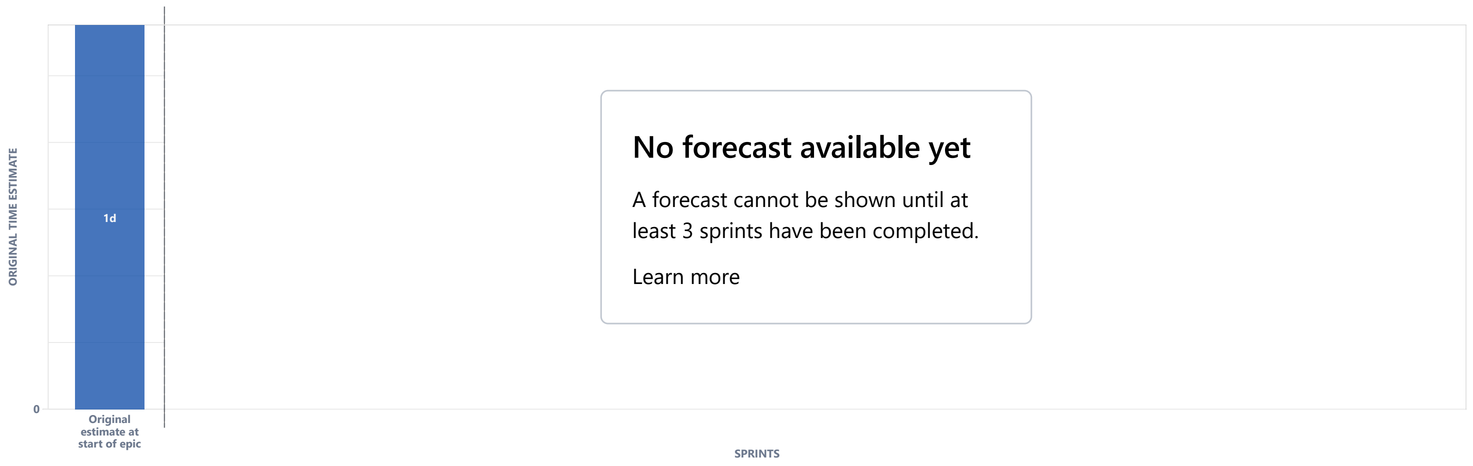
Switch report

INSTCLUST-170: Video

View linked pages

67% unestimated issues **0 of 1d** completed (original time estimate)

🟩 Work completed	🟦 Work remaining
⬜ Work forecast	🟦 Work added



No forecast available yet

A forecast cannot be shown until at least 3 sprints have been completed.

Learn more

Viewing options

☐ Align sprints at the base of the chart

Use this view to see trends in the scope for the epic.

Epic

INSTCLUST-170 Video recodring

Incomplete Issues

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (1d)
INSTCLUST-155	Make MJPEG stream HTTPS	Task	Medium	IN PROGRESS	1d
INSTCLUST-168	As a user i should be able to watch recoreded video	Story	Medium	TO DO	-
INSTCLUST-167	As a user i should be able to record Video	Story	Medium	TO DO	-

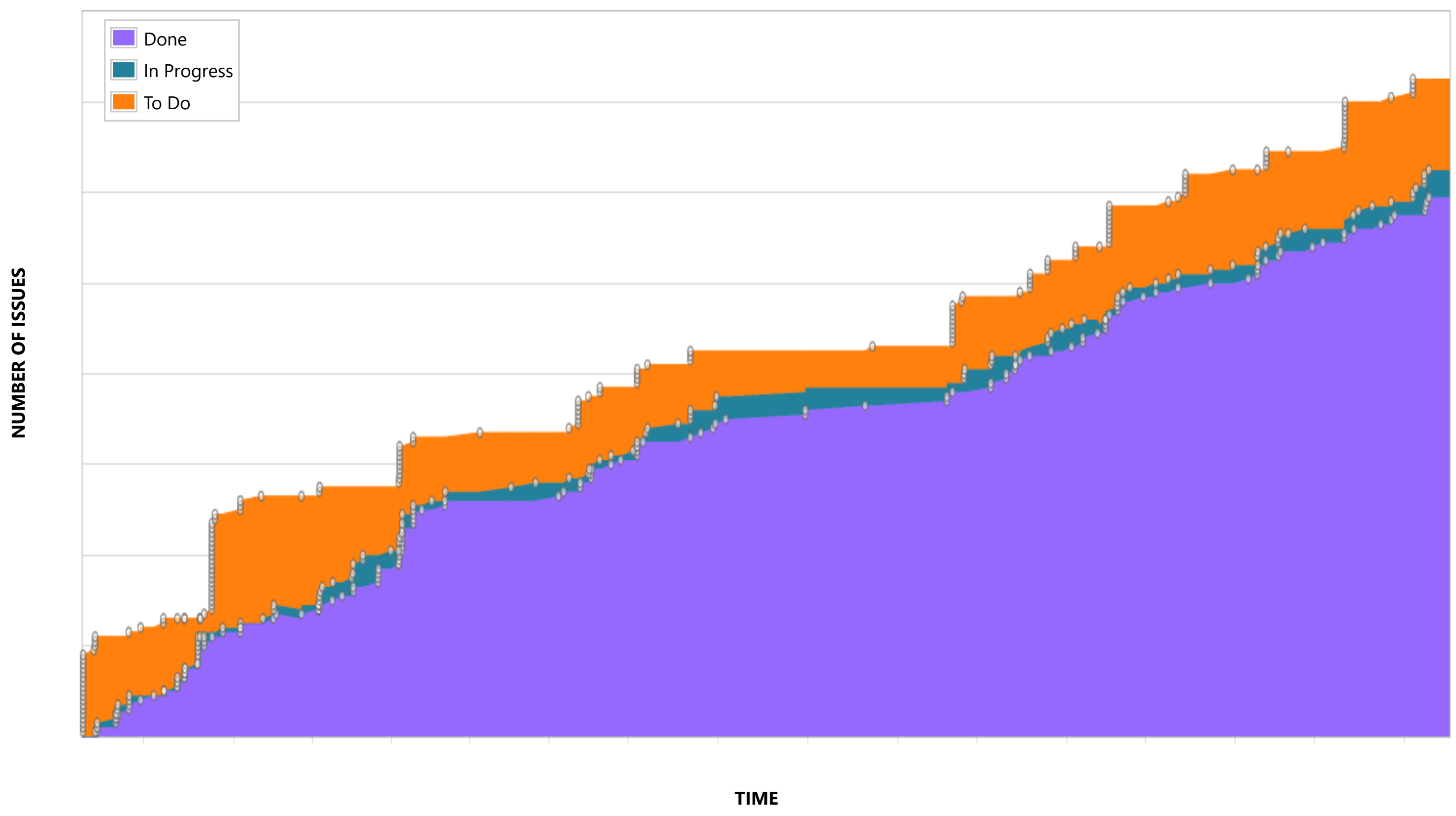
H Appendix Cumulative Flow Diagram

Cumulative Flow Diagram

Switch report

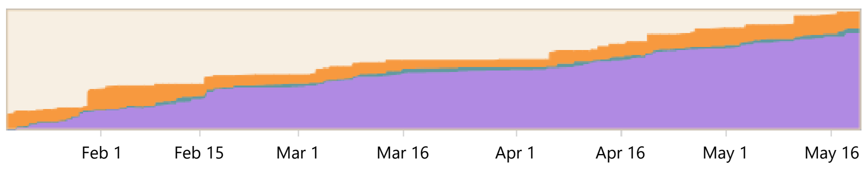
18/Jan/21 to 20/May/21 (All Time)

Refine report



Overview

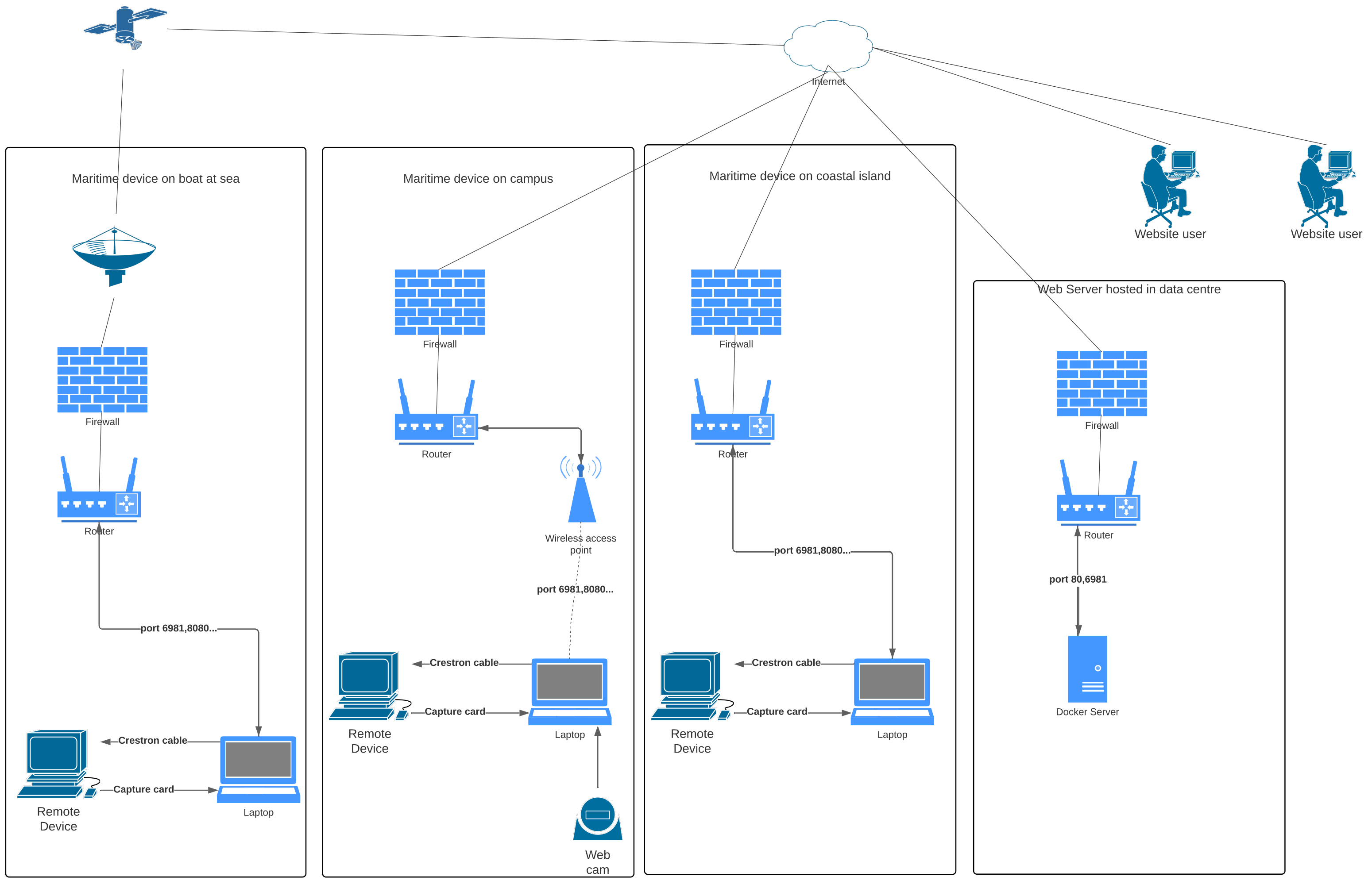
Click and drag cursor across chart or chart overview to select date range (double-click overview to reset).



I Appendix Network Topology Example

Distributed Instrument Cluster Network Topology Example

Andre Helland | May 11, 2021



Mikael Nilssen, Andrè Helland

NTNU
Kunnskap for en bedre verden