Eirik Thue

Bachelor's thesis

AutoPacker - An open-source project

May 2021

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

Bachelor’s thesis

2021

NTNU

Norwegian University of
Science and Technology

Eirik Thue

AutoPacker - An open-source project

Bachelor's thesis
May 2021

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

Obligatorisk egenerklæring/gruppeerklæring

Den enkelte student er selv ansvarlig for å sette seg inn i hva som er lovlige hjelpemidler, retningslinjer for bruk av disse og regler om kildebruk. Erklæringen skal bevisstgjøre studentene på deres ansvar og hvilke konsekvenser fusk kan medføre. Manglende erklæring fritar ikke studentene fra sitt ansvar.

Du/dere fyller ut erklæringen ved å klikke i ruten til høyre for den enkelte del 1-6:		
1.	**Jeg/vi erklærer herved at min/vår besvarelse er mitt/vårt eget arbeid, og at jeg/vi ikke har brukt andre kilder eller har mottatt annen hjelp enn det som er nevnt i besvarelsen.**	☒
2.	**Jeg/vi erklærer videre at denne besvarelsen:** • ikke har vært brukt til annen eksamen ved annen avdeling/universitet/høgskole innenlands eller utenlands. • ikke refererer til andres arbeid uten at det er oppgitt. • ikke refererer til eget tidligere arbeid uten at det er oppgitt. • har alle referansene oppgitt i litteraturlisten. • ikke er en kopi, duplikat eller avskrift av andres arbeid eller besvarelse.	☒
3.	Jeg/vi er kjent med at brudd på ovennevnte er å betrakte som fusk og kan medføre annullering av eksamen og utestengelse fra universiteter og høgskoler i Norge, jf. **Universitets- og høgskoleloven** §§4-7 og 4-8 og **Forskrift om eksamen** §§14 og 15.	☒
4.	Jeg/vi er kjent med at alle innleverte oppgaver kan bli plagiatkontrollert i Ephorus, se **Retningslinjer for elektronisk innlevering og publisering av studiepoenggivende studentoppgaver**	☒
5.	Jeg/vi er kjent med at høgskolen vil behandle alle saker hvor det forligger mistanke om fusk etter **høgskolens studieforskrift §31**	☒
6.	**Jeg/vi har satt oss inn i regler og retningslinjer i bruk av kilder og referanser på biblioteket sine nettsider**	☒

Publiseringsavtale

Studiepoeng: 20

Veileder: Girts Strazdins

Fullmakt til elektronisk publisering av oppgaven

Forfatter(ne) har opphavsrett til oppgaven. Det betyr blant annet enerett til å gjøre verket tilgjengelig for allmennheten (Åndsverkloven §2).
Alle oppgaver som fyller kriteriene vil bli registrert og publisert i Brage HiM med forfatter(ne)s godkjennelse.
Oppgaver som er unntatt offentlighet eller båndlagt vil ikke bli publisert.

Jeg/vi gir herved NTNU i Ålesund en vederlagsfri rett til å gjøre oppgaven tilgjengelig for elektronisk publisering: [✓ ja] nei

Er oppgaven båndlagt (konfidensiell)? ja [✓ nei]
(Båndleggingsavtale må fylles ut)
- Hvis ja:
Kan oppgaven publiseres når båndleggingsperioden er over? ja nei

Er oppgaven unntatt offentlighet? ja [✓ nei]
(inneholder taushetsbelagt informasjon. Jfr. Offl. §13/Fvl. §13)

Dato: 20.05.2021

Preface

In this thesis described AutoPacker as an open-source project, which contain the experiences of working in an open-source project, the research needed to enter an already established project and the development that goes with it. The project initially started in 2020 where the foundation to the project lies.
This thesis is written at NTNU Ålesund as part of a requirement fulfil the requirement for a computer engineering bachelor’s degree.

Abstract

AutoPacker is a program that configure a virtualized environment and make you able to setup this locally or a remote server. These projects can be submitted to an organization, which can contain many projects, an organization can be created by anyone. The creator of an organization can chooses who get to join and submit something to the organization. Anyone in an organization can utilize the projects that are already submitted, and they can add it to their own server while also control what the users roles are or if they are to be removed from the organization.

Acknowledgement

I want to thank Girts Strazdins for the help as supervisor on this project

I also want to give a thanks to Aron Mar Nicholasson form Avento for helping as mentor for this project.

Contents

Terminology

Agile - Agile Development is a way of working in iterations, where each iteration is a set period between 1 to 4. In the end each iteration you have a meeting about this iteration and what to do in the next one.

API - Application Programming Interface, is a way to communicate and extract data.

Back-end – Is the part of the system the user does not see, but it is code that handle logic, user's data and how/where to save it.

CD - Continuous deployment is a way to deploy

Docker – Is a program that handles docker containers

Docker Container – Docker container is an instance of a docker image.

Docker Image – Is an executable package, this can be any software.

Front-end – Is the part of the program the users can directly interact with.

GDPR - General Data Protection Regulation, new rules for handling user data that came in 2018.

GUI - Graphical User Interface, how the softer look like

HTML - Hypertext Markup Language, is the language most websites are structured upon.

HTTP - Hypertext Transfer Protocol, is a protocol to share webpages

HTTPS - Hypertext Transfer Protocol Secure, is a further develop HTTP that is more secure so one can share sensitive data between client and server.

Issues – A issue can be a lot of different things, it is a work order on something that is categorized usually as a bug or an enhancement for the project.

JDK - Java Development Kit, a tool to develop Java applications.

JSON - JavaScript Object Notation, is a minimal language to share information between client and server.

JWT - JSON Web Token, is a way to keep hold of user's settings and how long they should be logged in.

Open Source – A project where whomever can work on and improved since the code is open to the public.

Server – A computer that give information to other computers

Sprint – A limited period usually ranging from a week to a month where you work on specific task that you laid out for that sprint.

SQL - Structured Query Language, a programmable language for speaking with databases.

SSH - Secure Shell, is a network protocol that is making it secure to communicate without someone listening in.

SSL - Secure Sockets Layer, is a certificate for

TCP - Transmission Control Protocol, is one of the common ways servers can talk to each other

Figures

1. Introduction

1.1 Background

The reason why I choose this task is because I wanted to work in an open-source environment, and I knew about the project already had knowledge about how it operated. I was presented with a handful of roads to go down, where the tool was lacking features, my mentor wanted to improve upon. In the end I settled on working with the organization part of project where there were limited features to

1.1 Scope and objective

The scope of this is to be able to enter an open-source project and research their technology so it can be further developing the product in the organization area.
Document how to work in an open-source project, what is the challenges and how is what is the procedure working with open-source.
In the end of the project a user should be able to create their own organization, control roles member has, accept/decline new member, accept/decline project submissions and be able to use those submissions by any user of an organization and add it to your own server.
Main objects would be:

- Research the technologies tied to the project
- Research and document how it is to work in an open-source project
- To make organizations fully functional within AutoPacker so anyone can utilize it

2. Theoretical Basis

2.1 Privacy

2.1.1 GDPR

GDPR stands for General Data Protection Regulation and is a law became applicable 25th of May 2018 for the European Union. The goal for this regulation is to ensure protection for privacy and personal information to citizen within the EU. This enforces businesses to follow specific rules of how they should handle information and it gives more rights for the user what should happen with their data. A user could ask for a business to delete all their data and the business would have to due to the GDPR
Norway is not part of the EU but have embedded it into their Personal Data Act. [1] [2]

2.1.2 Personal Data Act

The Personal Data Act is a Norwegian law regarding handling, using and storing data. This makes sure that citizens data is handle well and give them their right to privacy.

Norwegian citizens and businesses follow this specific law, but also the GDPR. [3] [4]

2.2 Security

2.2.1 Hash

Hash is often used as a one-way operation method. It takes the raw data adding it to the hash function and give you a hash sum, this sum has is mainly to hide the original value you started with. Your hash sum is usually a fixed sum that changes when the data changes, just a just a little change in the data and you get a different result. As an example, if your password list is leaked for your website, the password is not clear text and if you used a good one-way hash function it is most likely the intruders cannot reverse it into the original value. [5] [6]

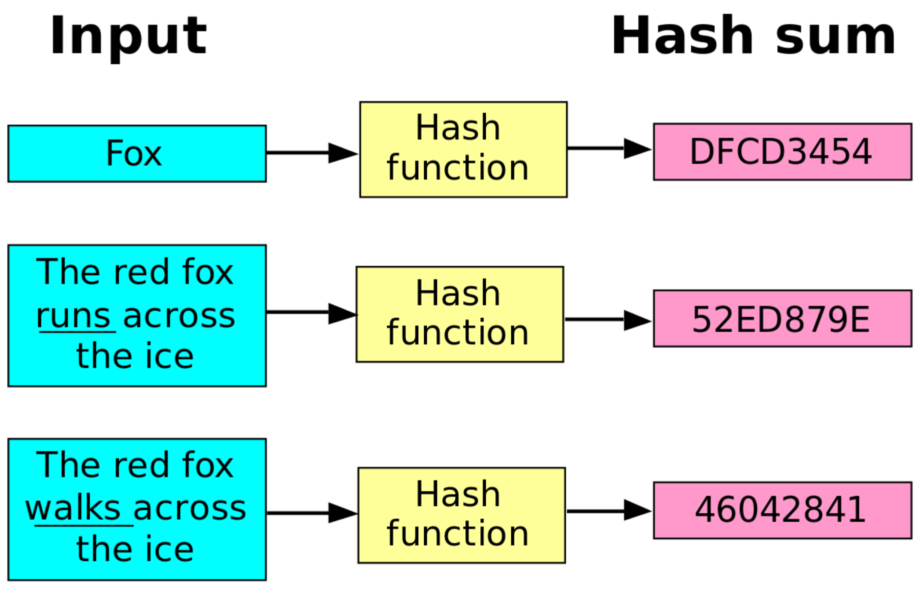


Figure 1: Hashing explanation [7]

2.2.2 Web Token

Web tokens are usually used for places you need to sign in and keep you sign in while you are on the page. It contains the user's authentication values, but they are usually hashed to secure the users so if they land in someone else's hands, it still will not compromise the user's logins, however it still could be used to gain access of an account. [8]

JSON Web Token is one that is used a lot of instances where you need either authorization or API calls. It is composed of what they call "Header", "Payload" and "Signature". [9] [10]

- Headers contain the algorithm and what type of token it is
- Payloads contain the data, example: Username/Password for login
- Signatures contain verification for the integrity of the message

2.2.3 What is HTTPS and common attacks?

It stands for Hypertext Transfer Protocol Secure, that make you able to communicate on the internet without someone being able to listen in. This is done through a layer that is called "Transport Layer Secure" with encrypt your data, so it is harder for someone to pick up the communication that goes between a client and a server. [11]

Some typical attacks that happen would be: [12]

- **Man in the Middle Attacks** usually ends up in the secure TLS gets exploited and someone gets access to listen in to the data that gets sent and received. Worst case it ends up with the attacker modifying the information, so you receive or give away information that the attack wants you to send or see.

- **SQL Injection** Is the form of using SQL queries where they are not intended. If your webpage has an unsecure text submit field there can be possible to send a SQL query to extract usernames or passwords.

- **Denial of service Attacks** is carried out by a machine that try to send as many requests as it can to a server. If the server has no filter for the request, it might think all the request is valid request and try to handle all of them. If the server gets too many of these, it will eventually shut down due to overworking itself. Meaning the request coming from computers that want to use the service will not get access.

- **Cross-Site Scripting** happens when an attacker can inject a script into your website. The usage of this for the attacker can be different from case to case, but usually the scripts are used to capture user's input. Like usernames, passwords, personal information, etc.

2.3 REST API

2.3.1 What is REST?

REST also known as RESTful API, stands for Representational State Transfer, which enable web services to interact. [13] [14]For the API to be considered RESTful however it needs to meet certain criteria:

- **Client-Server Architecture** needs to be built up of server, clients and resources. To change anything, it needs to go through a HTTP request.

- **Stateless** which means every request is separated and unconnected. The API is not saving any prior or future calls and the calls coming in is not depending on another call to go through. All the session data is only locally on the client computer.

- **Cacheable data** that can streamline and make the client to server interactions faster, this way the API do not need to recreate data every time a call is made.

- **Uniform interface**, this includes various components:
  - Resources that can be requested are identifiable

  - Resources can be manipulated by the client
  - Message's clients receive are self-descriptive so client can understand the process
  - Hypermedia, meaning a user can see all the actions they can do in a system
- **Layered system**, the client should not know if they are connected directly or just intermediary to the API. If there are intermediary servers or services, they should not feel different than being connected directly.

2.4 Concepts in programming

2.4.1 Cohesion and Coupling

They are some key principals in Object Oriented Programming. [15]

Cohesion is referring to what a class should do. Low cohesion means that the class is unfocused on what it should do and have various actions. High cohesion means the class is focused on what its purpose is and all the methods within the class fits what the intention to the class.

Coupling is referring how tightly the different classes are together. High coupling is not what you want, that means that the classes are highly dependent on each other to work. If you change one class, it might break the whole system. Low coupling is the opposite where they depend less on each other and make it easy to change one file, so you do not need to change the whole system

Usually what you would want in a system is high cohesion and low coupling

2.4.2 Object Oriented Programming

Is the way of constructing objects which is an instance of a class. There are many popular program languages built on this foundation of thinking, some of them are:

- Java
- C++
- Phyton

The classes will be defining what type of attributes any object can have, but the attribute value is specific to the object itself. Let us say you have a class car, where one of the attributes to the class is color. If you have multiple objects (instances), one car can have the color attribute white and the other black. So, it is operated as two different entities, as a black and a white car [16].

2.4.3 Version Control Systems

Version control is a system which is made to keep track of all the changes made to your code. It keeps a record of any changes made so you can always reference back to it, while making sure that what changes was made by who. Version control also makes it easy for developers to work together on a project, even within the same file. If both developers are submitting their changes and there are any conflicts, the control helps to ensure it can be solved by the developer in terms of what to keep or what to not keep [17, 18]

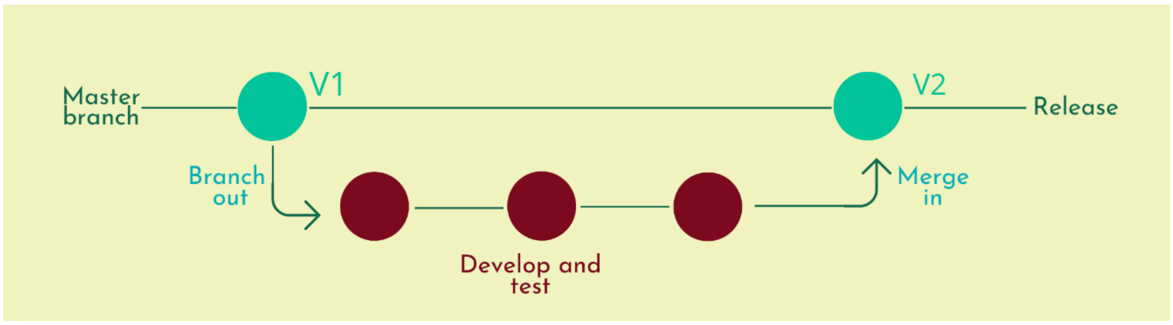


Figure 2: Version Control [19]

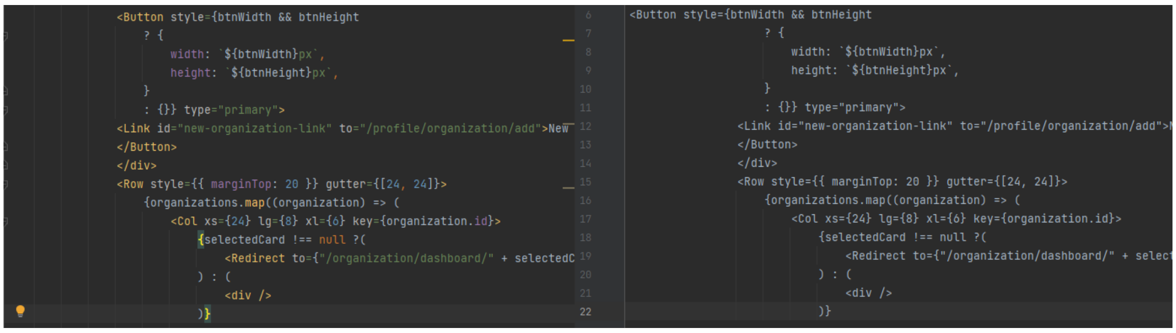
Above you can see the way the version control is using the master branch, copies it from that and then develops further until the developer are done. Then the changes are merged back into master for a new version of the project to emerge.

2.5 Programming

2.5.1 IDE

An IDE stands for integrated development environment and it is essentially a text editor, with extra set of features specific for programming [20]. The main features would be:

- **Auto Completion** on what you are writing, if you are trying to do a function call the IDE would know what you are trying to write before you have. So, you can work faster with the Auto completion of the sentence.
- **Syntax Highlighting** each IDE is often built for one or more languages and will detect which language it is and give a specific color to certain part of the code to make it more readable. As an example, you can find on the picture under, part of the project. On the left side there is a file with syntax highlight and right side without.



```
<Button style={btnWidth && btnHeight
    ? {
        width: `${btnWidth}px`,
        height: `${btnHeight}px`,
    }
    : {}} type="primary">
<Link id="new-organization-link" to="/profile/organization/add">New
</Button>
</div>
<Row style={{ marginTop: 20 }} gutter={[24, 24]}>
    {organizations.map((organization) => (
        <Col xs={24} lg={8} xl={6} key={organization.id}>
            {selectedCard !== null ?(
                <Redirect to={"/organization/dashboard/" + selectedC
            ) : (
                <div />
            )}
```

```
<Button style={btnWidth && btnHeight
                    ? {
                        width: `${btnWidth}px`,
                        height: `${btnHeight}px`,
                    }
                    : {}} type="primary">
                <Link id="new-organization-link" to="/profile/organization/add">
                </Button>
                </div>
                <Row style={{ marginTop: 20 }} gutter={[24, 24]}>
                    {organizations.map((organization) => (
                        <Col xs={24} lg={8} xl={6} key={organization.id}>
                            {selectedCard !== null ?(
                                <Redirect to={"/organization/dashboard/" + select
                            ) : (
                                <div />
                            )}
```

Figure 3: Syntax highlight side to side [21]

- **Building & Compiling** your programs for you, when you are done with the code it is just text. Before it gets built and compiled of the IDE and made into an executable program.
- **Debugging** is one of the most powerful tools in the whole program. Debugging is a way of either showing you where you have error in your code or follow the data that goes through your system to figure out where a problem might arise.

2.5.2 Java

Java is object-oriented program language, where everything is built up on all the attributes for a given object is inherent form the class. Java can run on all devices that support JVM, which stands for Java Virtual Machine. When you want to start any Java program, it is started within the JVM which runs on your operating system. This ensure that Java can run on most common systems today, the JVM can varies in different programs. But most are forked off the OpenJDK's Hotspot JVM. One of the strongest points within the JVM is the garbage collector, which makes it possible not worry of an overflooded ram. When you make instances of different classes it will take up space, but when you close those instances the garbage collector will make it [22, 23]

2.5.3 Spring

Spring is a framework is a controller container which have most common API tools, it is built for the Java platform. It contains Spring Boot that is minimal tool that is friendly for people to setup and contain most tools you would need for setting up your own API for a webservice.

2.5.4 SQL

Structured Query Language or SQL for short is the common standard for interacting with databases. It is mainly used for creating, deleting or gathering data in tables and have been marked as an ISO standard. [24, 25]
Common syntax usages:

- **Use** – To set the database you would use to find a table. In case there are multiple databases.
- **Select –** Which help you select what data be returned from a query.
- **From** – Defines which tables this data comes from.
- **Where –** Gives the query a specific requirement for what data it want back.
- **Update –** Tell the query to modify already existing data.
- **Set** – Helps to select which columns to update.
- **Delete –** Tell the query which tables to delete.

2.5.5 Maven

Maven is a powerful managing tool for projects, it works around POM which stand for Project Object Model. That means everything you need for a project you will add to a XML file, where you tell the Maven what version and dependencies your program needs, and Maven gather those programs automatically for you and help build the project files for mainly Java applications. [26]

2.5.6 Yarn

Yarn is one of the most popular package manager tools, it saves what Yarn calls a manifest "package.json" which contains all the versions and dependencies for JavaScript applications. [27]

2.5.7 Continuous Integration

CI or Continues Integration refers to the automatic implementation of changes that happens to the repository. The reason this have become a popular way of programming is that the changes get automatically built and tested. So, if you have problems in your code, they will be detected faster than if you tested the system after many iterations of updates. Then doing it manually which takes way longer than an automatic system. *[28]*

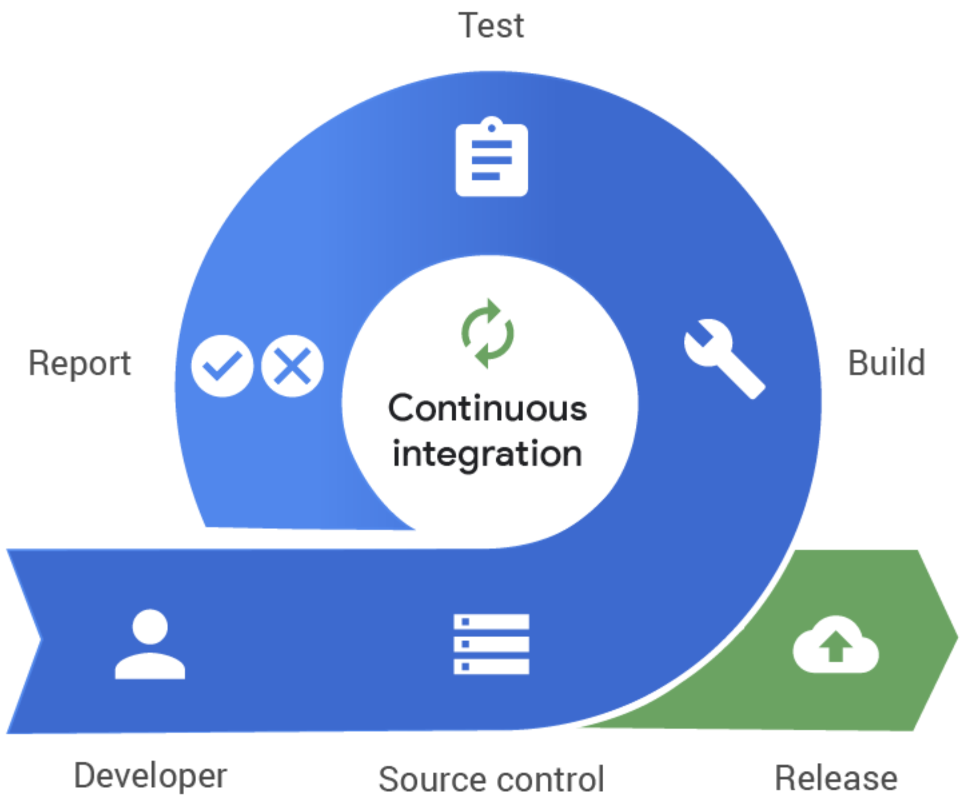


Figure 4: CI Cycle Explanation [29]

2.5.8 Continuous Deployment

CD or Continuous Deployment differ from CI in the way it automatically deploys the software after it is built. Then in the test environment bugs and other problems become apparent. It is not the same as CI, but in practice it achieves the same goal, but in another way of doing so. [30]

2.6 Docker

Docker have become widely popular since when it started up back in 2014, the reason for this popularity is the virtualization technology. It the container structure that docker is building is making it faster than regular VM (Virtual machine) [31, 32]

2.6.1 Containers

The reason containers are faster than regular VM is due to how it is layered, where you would make all the containers being able to access the same bins or libraries. This way of structuring as well cut out the middleman of having a hypervisor that connect to a guest OS for each virtualization like in the picture under. This way it become a more portable platform where containers can be independent of the platform and will not experience problems that are OS specific. [31]

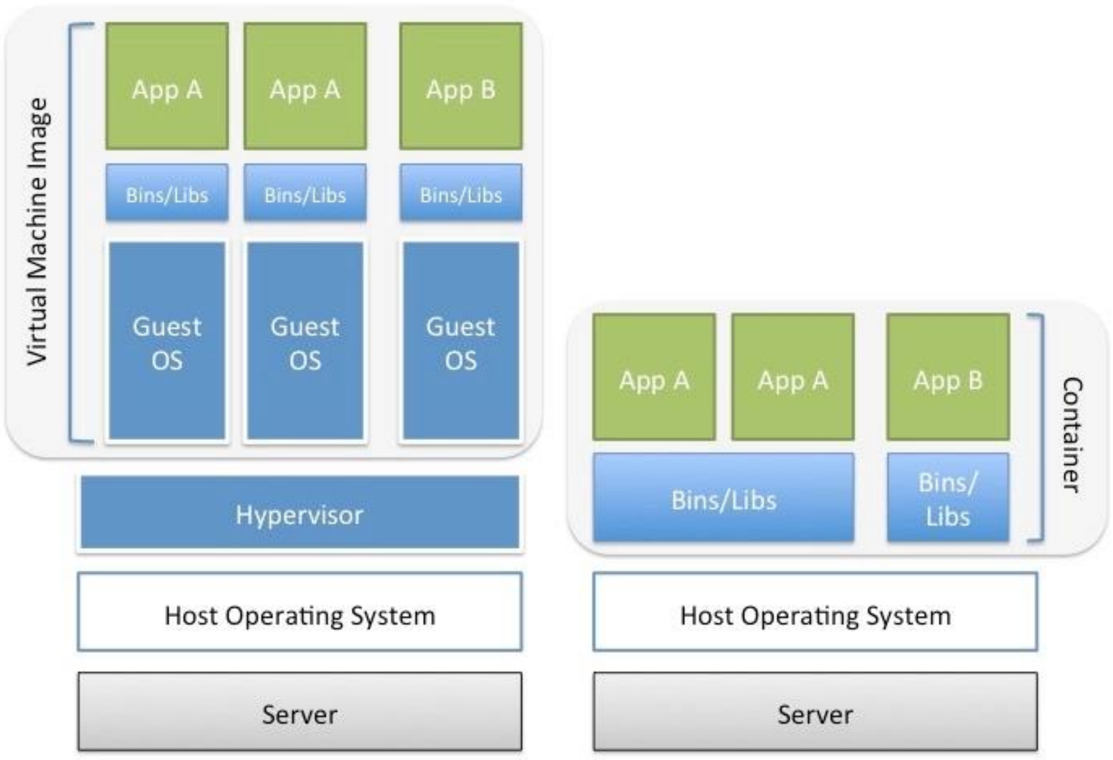


Figure 5: Docker containers [33]

2.6.2 Docker Mount and Registry

Docker volume is usually the preferable way of mounting docker containers to your computer, you can choose an option called blind mount but that is an older way of structuring docker components.

The difference is that in blind mounts, a directory on the computer will be used as the placement and the absolute path of where the containers are mounted will be used. [34] Where volumes would create another directory within the docker environment and will only behave within that. [35]

Within these dictionaries the docker images are executed, but to what images are stored where you use Docker Registry, which is a stateless, scalable, distribution software. This will control where everything gets stored, making you get more control of your own distribution pipeline and make storage and distribution tied into your workflow. For every time new code is pushed, this registry would register this as a new commit and keep a record of the changes. [36]

2.7 Management

2.7.1 Agile

In software development, Agile workflow means that you work with the costumer and meet every few weeks. A meeting that is often referred to as an iteration, where they go through the same phases. There are 6 in total:

- Requirements
- Design
- Development
- Testing
- Deployment
- Review

All this happens during the iteration, the "Requirements" and the "Review" is the meeting in the beginning/end of an iteration. Agile rely on not making too big steps, due to the nature of the methodology which want to be always done with what is worked on during an iteration. So next iteration you can keep developing on top of what you have already produced. [37, 38]

2.7.2 Scrum

Scrum utilizes Agile and builds on it, everyone to four weeks you make a backlog of problems or new features you should work on during the next sprint. Usually, you utilize a scrum board where you have all your backlog collected in different categories. Were you place new backlog in a "To do" category, when you start working on a specific backlog you move it to different categories to show where you are with the progress of the backlog. The type of categories is often up to the group, but usually it can be setup like this:

- To do
- In progress
- Done – Sprint 1, 2, 3. (And further depending on how long the project is)

This way you can keep track on what is in progress, what is done and what you got left to do. One can add different subcategories on the way depending on the teams need, as example:

- "Verification" if your group verifies updates.
- "Testing" if you test for every update.
- "Deployment" if the updates are not going automatically and you have a dedicated person to push it.

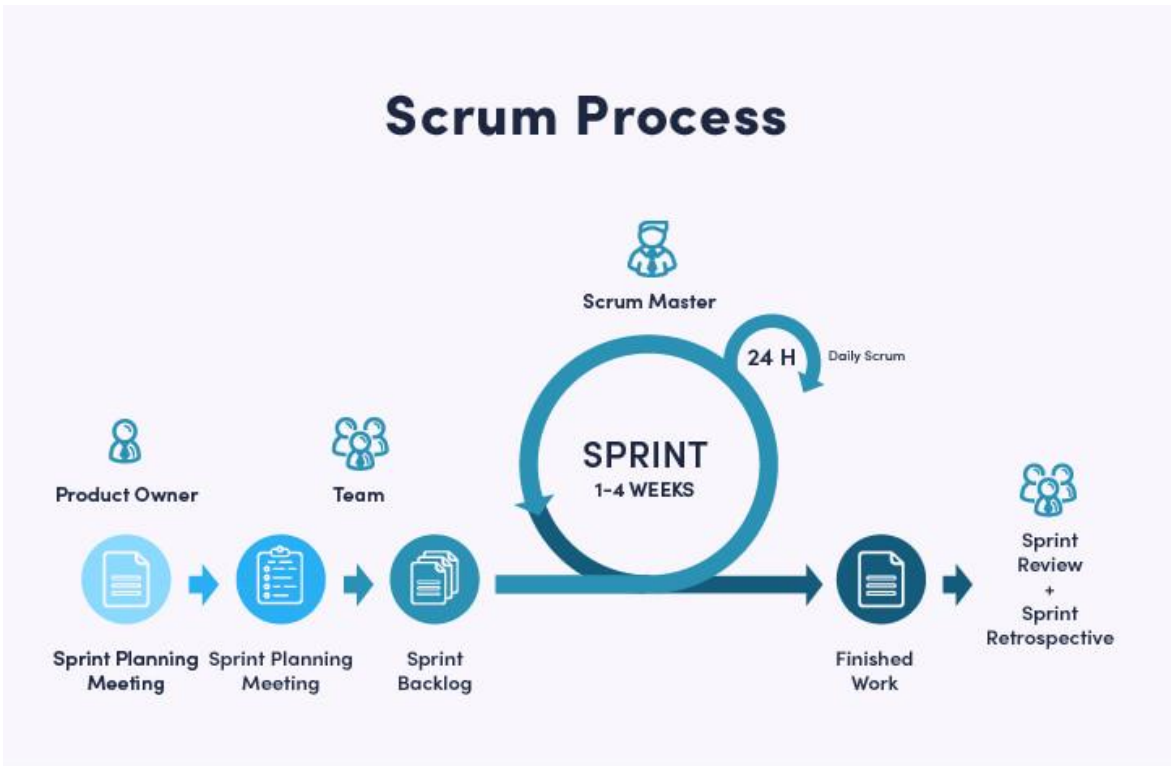


Figure 6: Scrum Process [39]

2.8 Open-Source

2.8.1 What is open source?

Open-source development refer to the open access to the source code of a software, where modification or re-distribution of the software goes under an open-source license. This means that people or companies can modify and improve the code for their own needs. [40] [41]

2.8.2 Differences

The main difference between an in-house team and open source comes down to how it is operated. The development process may vary, but it looks like this usually:

- Fork: Implies the code is copied for inspection by an interested individual.
- Modify: The person having interests in the project can choose to modify the source doe due to the open-source license that allow them.
- Release: After it is modified this costume software that was altered can be shared by the individual. Often back to the same community as their own contribution, when it is reviewed it is merged back to the code based it was forked from.

This loop of fork, modify, release can be reproduced anyone that want to work with a project. [42, 43]

3. Material and Methods

3.1 Project Organization

3.1.1 The Team

The team is based on myself Eirik Thue that is writing this bachelor thesis, one supervisor Girts Strazdins and one mentor Aron Mar Nicholasson.

3.1.2 Scrum Workflow

The projects workflow that we choose in the beginning of the project is Scrum form of agile development. We decided 2 weeks per sprint and from there adjusted the workload based on what needed to be done as next step in the progress of moving into the project. To track issues, we used GitHub's project feature, where you can have a scrum board with your issues and expanding the scrum board as you go.

3.1.3 Communication

The groups communication was mainly based on Discord or GitHub Discussions for general topics around the project or sharing resources. Meetings were done though Microsoft Teams on the same day every two weeks for an hour to decide what to work on next sprint.

3.1.4 Project Management

GitHub Projects – I have used GitHub projects to keep track of issues during each sprint and organizing what have been done.

Confluence – Confluence gives a lot of different templates and storage options for wireframes, diagrams, etc. Most type of documents and I have used confluence to store it.

Draw.io – Draw.io is an easy and free to use wireframe program that have been used to make the different wireframes for each individual changes of the website.

3.2 Programming

Due to the project were already existing before the work started on it, it is only natural to keep using the same languages and frameworks that already is in place.

3.2.1 JavaScript

React.js is the language most of the front-end development for this project is based on and I am further developing within the same language. React is becoming very popular within web development, making it only natural to keep using this language.

3.2.2 Java

Java programming in this project is on the back end, it was built around Spring Boot framework which is the API solution that is being used for this project.

3.3 Environment & Frameworks

3.3.1 GitHub

GitHub is a version control tool where a user or a team of users can build a repository, in the repository the code will be uploaded and altered over time. This can be done in a private or a public repository, since this is an open-source project the repository for this project is open. When the repository is open to anyone, there will be some hierarchy where when changes are made, the moderators of the repository can take the changes in effect.

You can add a CI/CD solution as well to GitHub, making it able to deploy any changes that is made directly. [44]

3.3.2 MySQL

MySQL is an open-source RDBMS for the ones that do not like acronyms, that stands for relational database management system. It is one of the most common and used database systems out there. [45, 46, 47]

3.3.3 IntelliJ

IntelliJ was used for both back & front-end development in this project, it is a JetBrains IDE that support various languages like Java and JS, making it a solid choice for further developing this project. [48]

3.3.4 Spring Boot

Spring Boot is an open-source framework that uses “Inversion of control” that helps the developer early on to configure the business logic for the API. [49] The way of setting up spring boot is often done in a “Controller”, “Service” and “Repository” partition. [50]

It would be used in this way:

- **Controller class** is the first stop for data when the API is called. Here you will obtain the data from the call, authenticate user that it is allowed and have quality testing (Checking if the call is empty). If the API call is valid the data is sent to the service class.
- **Service class** is the business logic of the system, where the data is handled. Depending on the API call, it can be altering existing data, adding new data or deleting data. Usually, this class interact a lot with different repository classes.
- **Repository** is an interface that is tasked with talking to the database, here you either save or gather information to or from the database.

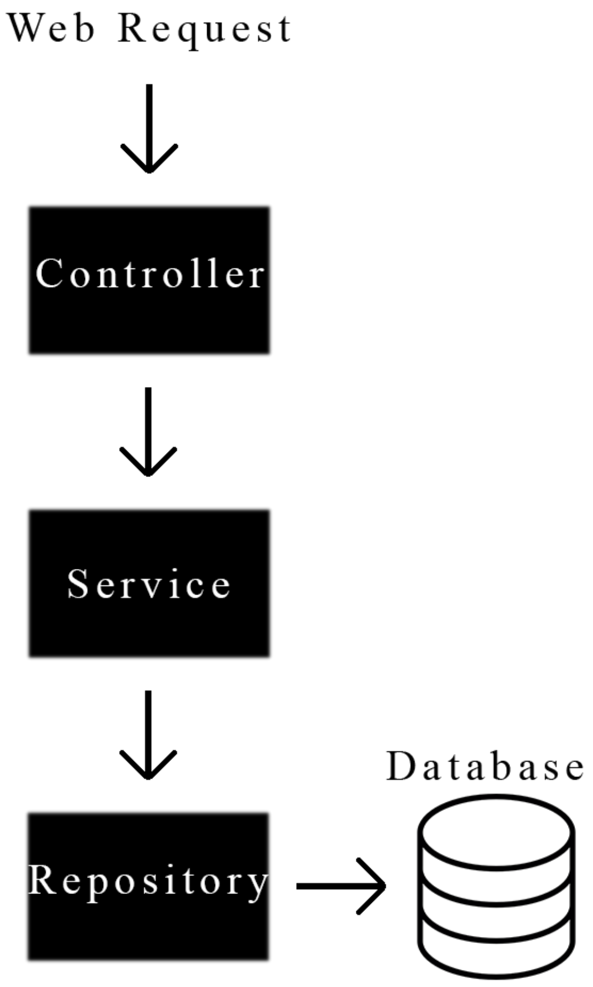


Figure 7: API explanation

3.3.5 Postman

Postman is a powerful API tool, it helps the developer testing or documenting their API when they are developing them. The tool makes you able to make HTML request with JSON bodies, making you able to use the APIs without going through the website and testing each API call. Instead, you pre-configure all the calls with the necessary JSON body and authorization, when this is done you can just run the test every time you want to try if the API works. [51, 52]

3.3.6 React

React is web framework that based to make single page applications (SPA), the way SPA works is different from traditional webpages. In traditional pages every time you do some sort of request, you will be refreshing the whole site and new content will be loaded in.

For react this work differently, react have a dynamic way of dealing with it where only the relevant content is updated. It gives you control over DOM elements; Therefore, you can control what is added to the page next. [53]

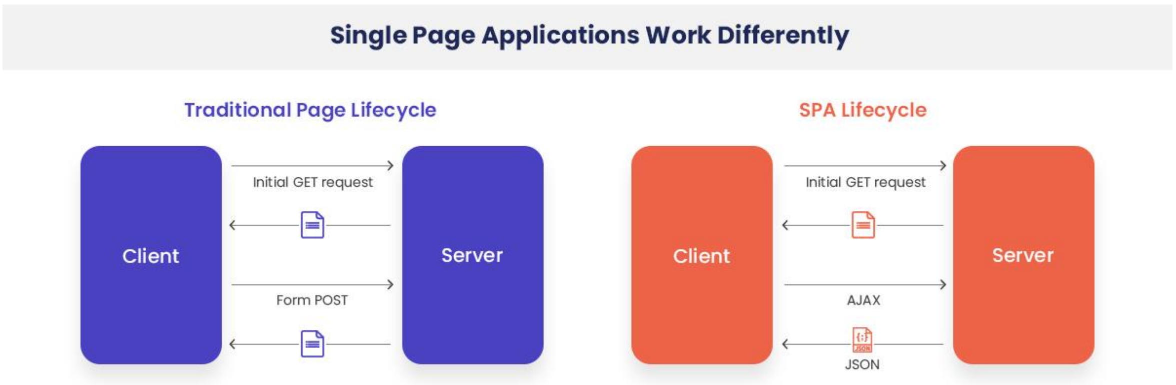


Figure 8: Traditional Page vs SPA [54]

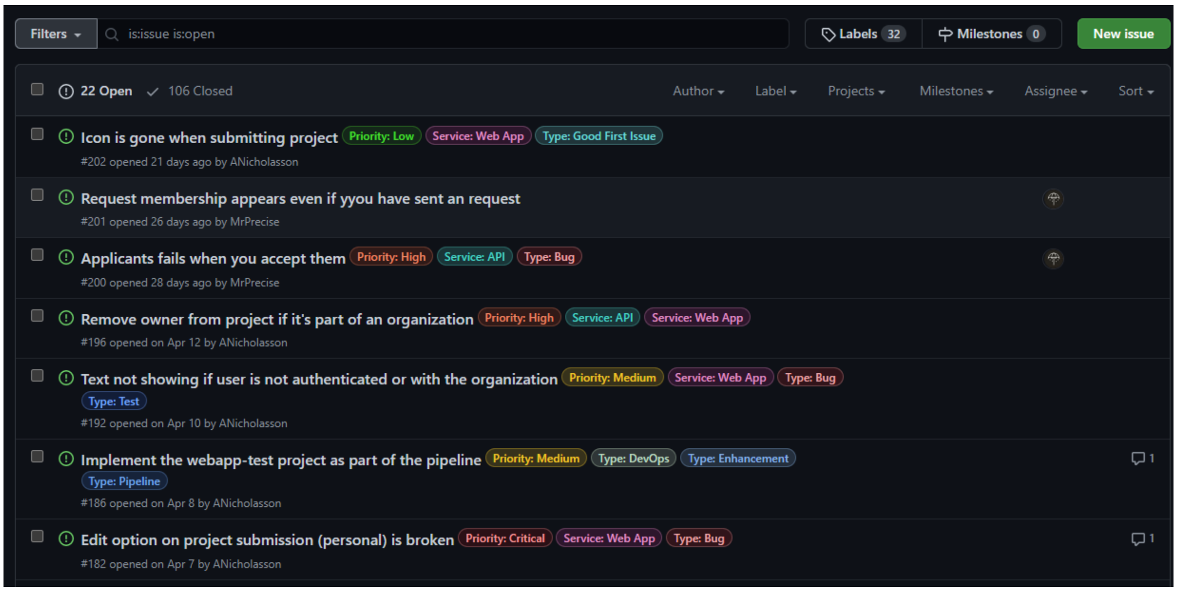
React is buildup of what we call components, props and states. A component is a function that can be used anywhere, it is often just an empty shell before they get handed their props. Props stand for properties and will give component their data, this can be transferred from component to component. Last we got state, which is what controls your components, where your components are and what they look like at any time. [55, 56]

3.4 Open-Source in GitHub

3.4.1 What is an issue?

When you are working in an open-source environment you might not have access directly to anyone that works beside you on the project. Giving you a great tool called issues, it is a backlog of any project. Any operation that is performed on a project is an issue, weather it is a bug or an enhancement to the project. In these issues you suggest improvements or problems/vulnerabilities with the current version of a software.

It gets sorted into a list of issues where anyone can contribute by commenting or even assign the issue to themselves and solve it. [57]



Icon is gone when submitting project — Priority: Low, Service: Web App, Type: Good First Issue
#202 opened 21 days ago by ANicholasson

Request membership appears even if yyou have sent an request
#201 opened 26 days ago by MrPrecise

Applicants fails when you accept them — Priority: High, Service: API, Type: Bug
#200 opened 28 days ago by MrPrecise

Remove owner from project if it's part of an organization — Priority: High, Service: API, Service: Web App
#196 opened on Apr 12 by ANicholasson

Text not showing if user is not authenticated or with the organization — Priority: Medium, Service: Web App, Type: Bug, Type: Test
#192 opened on Apr 10 by ANicholasson

Implement the webapp-test project as part of the pipeline — Priority: Medium, Type: DevOps, Type: Enhancement, Type: Pipeline — 1
#186 opened on Apr 8 by ANicholasson

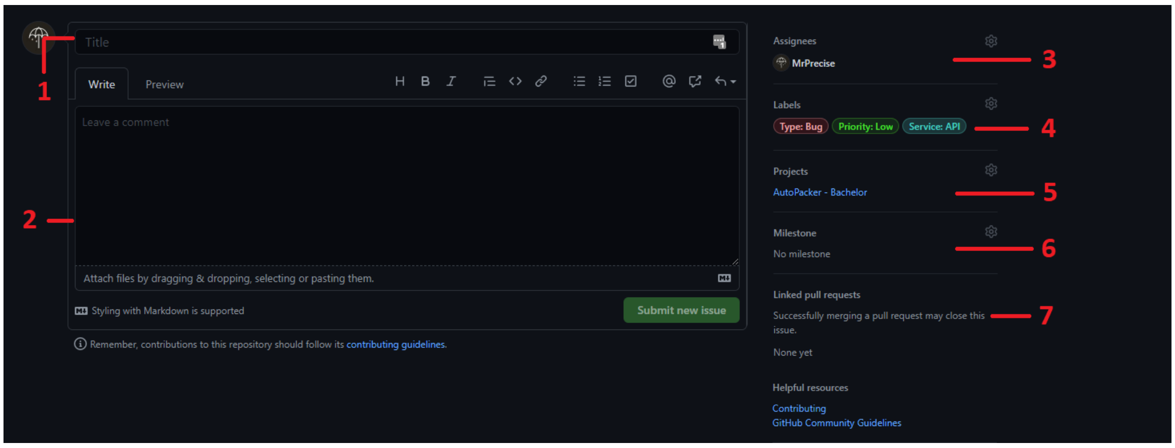
Edit option on project submission (personal) is broken — Priority: Critical, Service: Web App, Type: Bug — 1
#182 opened on Apr 7 by ANicholasson

Figure 9: Issue, what is it?

3.4.2 How to write an issue?

When you have found a bug or something you think can be improved, the first thing you should do to improve the project is to write an issue. The way you do this in the best practice, is in the following fashion (Numbered picture to follow under) [57]:

1. **Title:** The title should be encapsulating the problem or the enhancement that needs to take place without having to delve into the issue in depth. A reader should be able to understand what it is about only from reading the title.
2. **Description:** Is the main body of an issue, here you will describe what the problem or enhancement is and in some cases suggestions on how to solve the problem if you know a way it can be solved. The description field in GitHub is an advanced text editor with markdown capabilities, where you can reference other issues, mentions users, make text fields or share pictures.
3. **Assignees:** Can be non or as many people you want to this issue. If none are selected, then it will be in a backlog for a later date when someone else picks it up.
4. **Labels:** Is custom made label you add your issue, they are usually specific to the type of project you are working on to signal what type of issue and where in the project is this issue located. However, sticking to some generic ones are useful as well. Like in the figure 10 under, you will see we have priority "Low", we also have one for "Medium" and "High" as well to distinguish what need faster attention than others.
5. **Projects:** Projects are not the same as a milestone, where they do the same thing of being a collection of issues. Projects will have a bigger scope than a milestone and containing many milestones within a project.
6. **Milestone:** Is used for corresponding to individual goals or features within a project. Where you have a collection of issues needed for fulfilled for that feature or goal to be done. You use milestones to track the progress.
7. **Linked pull request:** This is often added later when the job is performed, where you could look back to a pull request from the assignee on the issue. To see what was done and review the code for adjustments or giving feedback on the work that was done


Title

Leave a comment

Attach files by dragging & dropping, selecting or pasting them.

Styling with Markdown is supported

Remember, contributions to this repository should follow its contributing guidelines.

Assignees
MrPrecise

Labels
Type: Bug
Priority: Low
Service: API

Projects
AutoPacker - Bachelor

Milestone
No milestone

Linked pull requests
Successfully merging a pull request may close this issue.
None yet

Helpful resources
Contributing
GitHub Community Guidelines

Figure 10: How to write an issue

4. Results

4.1 Contribution to open source

4.1.1 Process of contributing

When it comes to process of contributing it start with looking at the product itself, while the most obvious implementations you would want to do is new features that can be added to make the product more versatile. But also bug and things that is not working should be made issues of, when a potential case appears one should make an issue on it.

4.1.2 Issues

During the project I created 43 issues regarding the project, this have been general bug fixes, but mainly focus on the organization part of the project where my main responsibility lied.

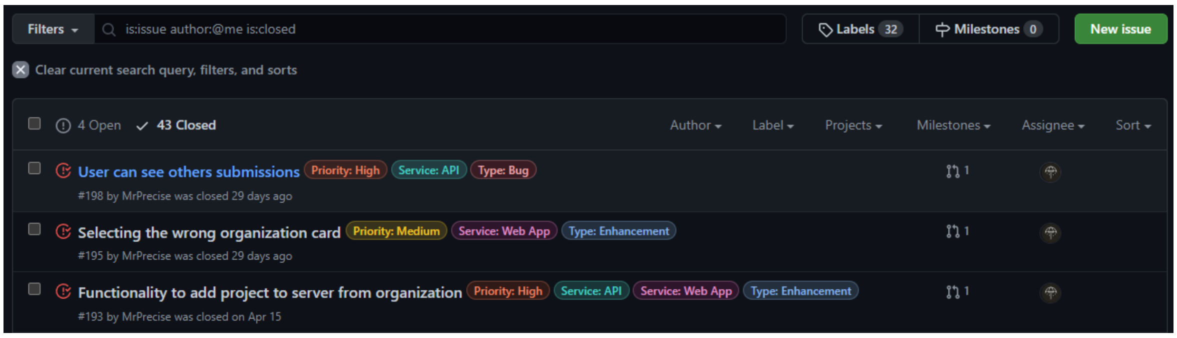

Filters is:issue author:@me is:closed Labels 32 Milestones 0 New issue
Clear current search query, filters, and sorts
4 Open ✓ 43 Closed Author Label Projects Milestones Assignee Sort
User can see others submissions Priority: High Service: API Type: Bug
#198 by MrPrecise was closed 29 days ago
Selecting the wrong organization card Priority: Medium Service: Web App Type: Enhancement
#195 by MrPrecise was closed 29 days ago
Functionality to add project to server from organization Priority: High Service: API Service: Web App Type: Enhancement
#193 by MrPrecise was closed on Apr 15

Figure 11: Issue list

I have also closed 85 issues for the project during the whole bachelor period, this shows of course assigned issues that was given to me from other contributors. Not only the issues I was creating hence the bigger list of issues completed.

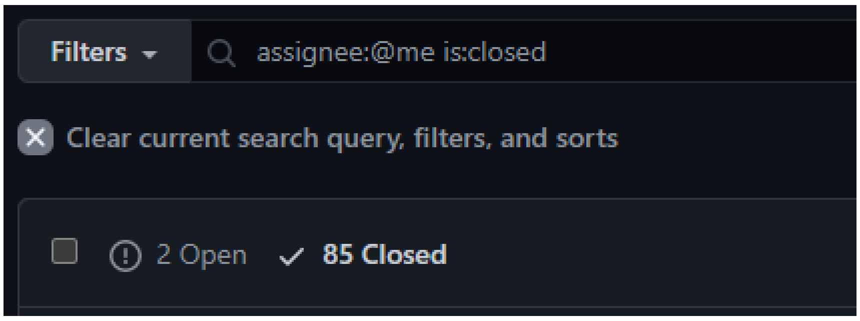
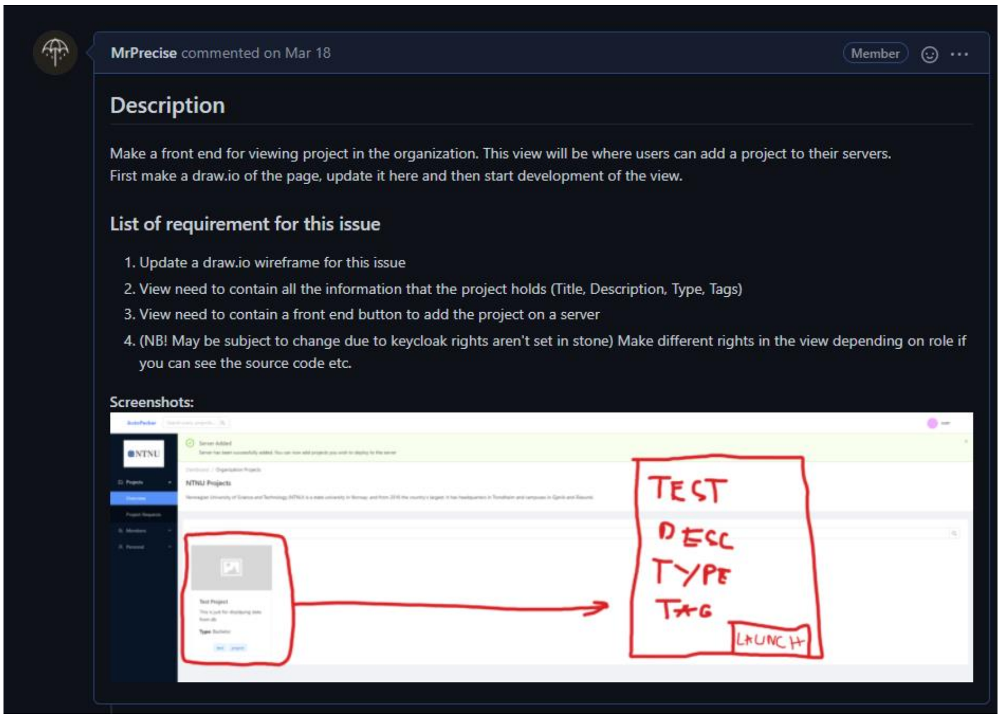

Filters assignee:@me is:closed
Clear current search query, filters, and sorts
2 Open ✓ 85 Closed

Figure 12: Total closed issues

Going from issues to implementations are not always straight forward, many occasions you would most likely add a type of wireframe to show the changes that are going to take place so if others are working on the same issue, you are. There will be no surprises of how thing will turn out, here is an example of that, I made a wireframe to this that can be found in appendix.



MrPrecise commented on Mar 18

Member

Description

Make a front end for viewing project in the organization. This view will be where users can add a project to their servers. First make a draw.io of the page, update it here and then start development of the view.

List of requirement for this issue

1. Update a draw.io wireframe for this issue
2. View need to contain all the information that the project holds (Title, Description, Type, Tags)
3. View need to contain a front end button to add the project on a server
4. (NB! May be subject to change due to keycloak rights aren't set in stone) Make different rights in the view depending on role if you can see the source code etc.

Screenshots:

Figure 13: Example issue

4.1.2 Work done

According to GitHub I added 4778 and removed 1030 during this project, due to this being already an established project before I started to work on it. A lot of my work revolved around refactoring code that was already in place, making the numbers may be lower than a newly established project.

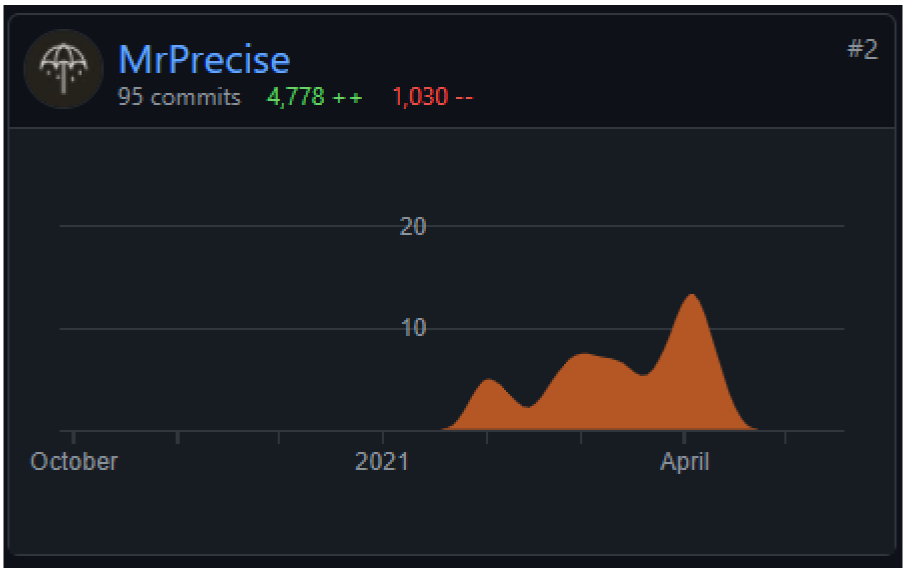


Figure 14: Work done during bachelor

4.2 System Architecture

4.2.1 System design

The general system designs of the system have become less complicated over time, when I started working on it was a microservice where we had a General, File, Organization and User API.

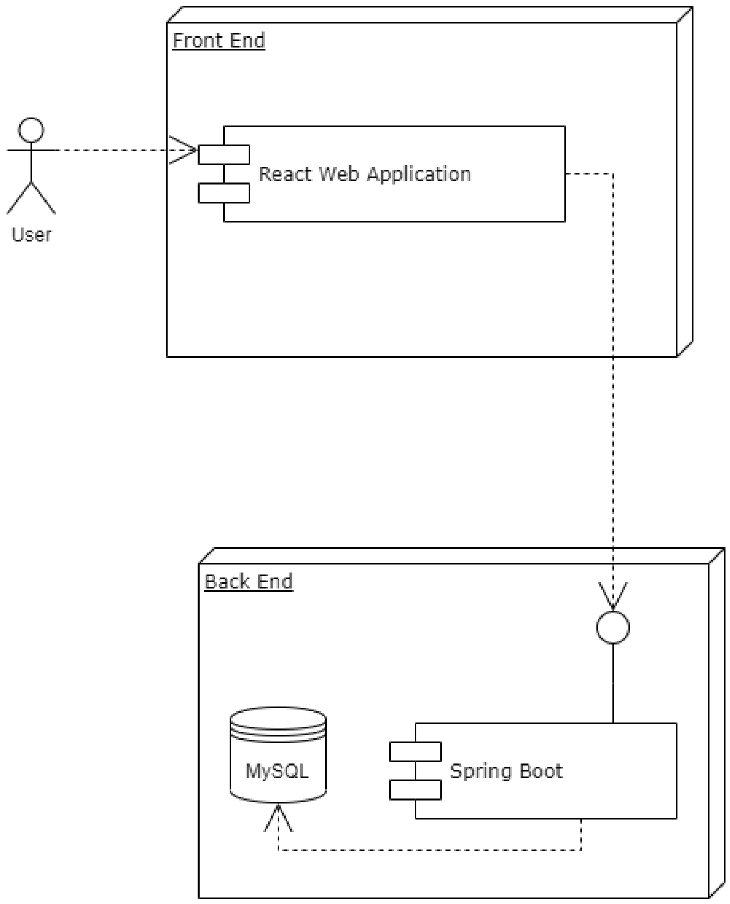


Figure 15: System Design

As you can see above the system have become way less complicated and straight forward in the path the data takes. We took all the microservice APIs and merged them to one, now we only have one back-end API we need to call. My focus was pivoting towards the organization part of the site, although I helped in the process of the merging of the APIs.

4.3 Database

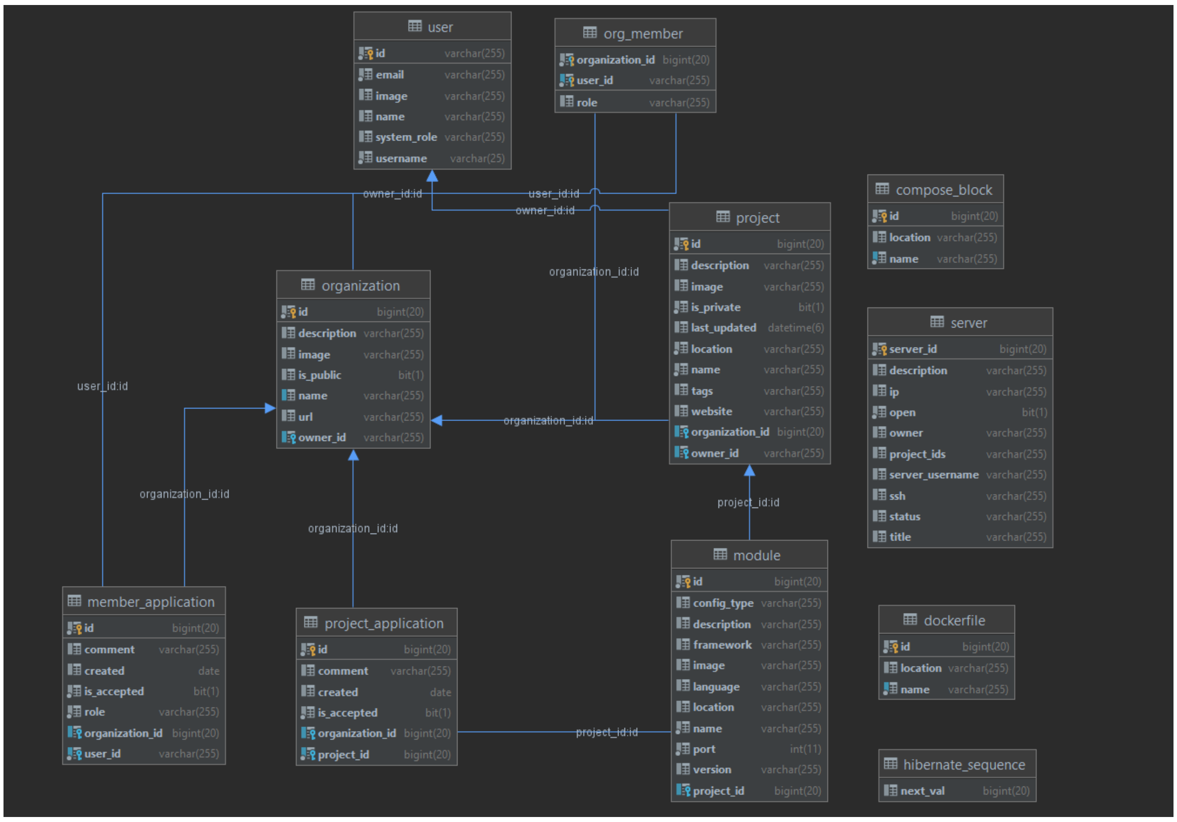


Figure 16: Database overview

I will go over the ones I used in my project under.

4.3.1 User Table

- Id varchar (255): Primary Key. A unique value to identify the user
- Email varchar (255): Holds the users email that was used to registered on the page
- Image varchar (255): Hold the location of the user's personal image
- Name varchar (255): Users real name
- System_role varchar (255): This is for if you are a system admin for the whole system or not, which would make it able to give you access to different part of the system a normal user should not use.
- Username varchar (255): User another unique

Not too many fields were change here but the system_role, it gave us greater control over who was an admin of the site so they could get greater access and made it easier to extinguish roles for the whole system and different organizations.

4.3.2 Organization Table

- Id varchar (255): Primary Key. A unique value to identify the organization
- Description varchar (255): Holds the description value to the organization
- Image varchar (255): Hold the location of the organization's logo
- Name varchar (255): Name of the organization owner's name
- URL varchar (255): Holds URL to relevant pages for the organization
- Owner_id varchar (255): Foreign key. Reference to org_member table

The biggest change to organization making the way of reusing the information from other tables to the base information for an organization.

4.3.3 Org_member Table

- Organization_id bigint(20): Primary Key / Foreign Key to organization table id.
- User_id varchar (255): Primary Key / Foreign Key to user table id.
- Role varchar (255): Holds the role within a given organization.

Org_member is a whole new table made entirely to hold the different roles different users have in different organizations.

4.3.4 Member_applications Table

- Id bigint (20): Primary Key. A unique value to identify the application.
- Comment varchar (255): Value containing the comment made for the application.
- Created date: The date of creation.
- Is_accepted bit (1): True / False statement if the member is accepted.
- Organization_id bigint (20): Foreign Key to organization table id.
- User_id varchar (255): Foreign Key to user table id.

Member applications is to see request for joining an organization.

4.3.5 Project Table

- Id varchar (255): Primary Key. A unique value to identify the project.
- Description varchar (255): Holds the description value to the project.
- Image varchar (255): Hold the location of a picture for the project.
- Is_private bit (1): True / False statement to hide or show the project.
- Last_updated datetime (6): Show date and time when the project was last updated.
- Location varchar (255): Shows where on server the project is located.
- Name varchar (255): Holds the value for the name of the project.
- Tags varchar (255): Holds the tags for the project.
- Website varchar (255): Holds any links connected to the project.
- Organization_id bigint (20): Foreign Key. Holds the id for the organization that owns the project.
- Owner_id varchar (255): Foreign Key. Holds the id for the user that owns the project.

Project table have changed a bit from the original where all project information is stored here and there are no individual table for organization project. This way we added organization_id as a foreign key so when a project is submitted to an organization, you can add an organization value to change ownership instead of a whole different table for the same information.

4.3.6 Module Table

- Id varchar (255): Primary Key. A unique value to identify the module.
- Description varchar (255): Holds the framework value to the module.
- Framework varchar (255): Holds the description value to the module.

- Image varchar (255): Hold the location of a picture for the module
- Language varchar (255): Holds the language value to the module.
- Location varchar (255): Holds the location to the module.
- Name varchar (255): Holds the name to the module.
- Port int (11): Holds the port for the module
- Version varchar (255): Holds version number of the module
- Project_id: Foreign Key. Value to the project id.

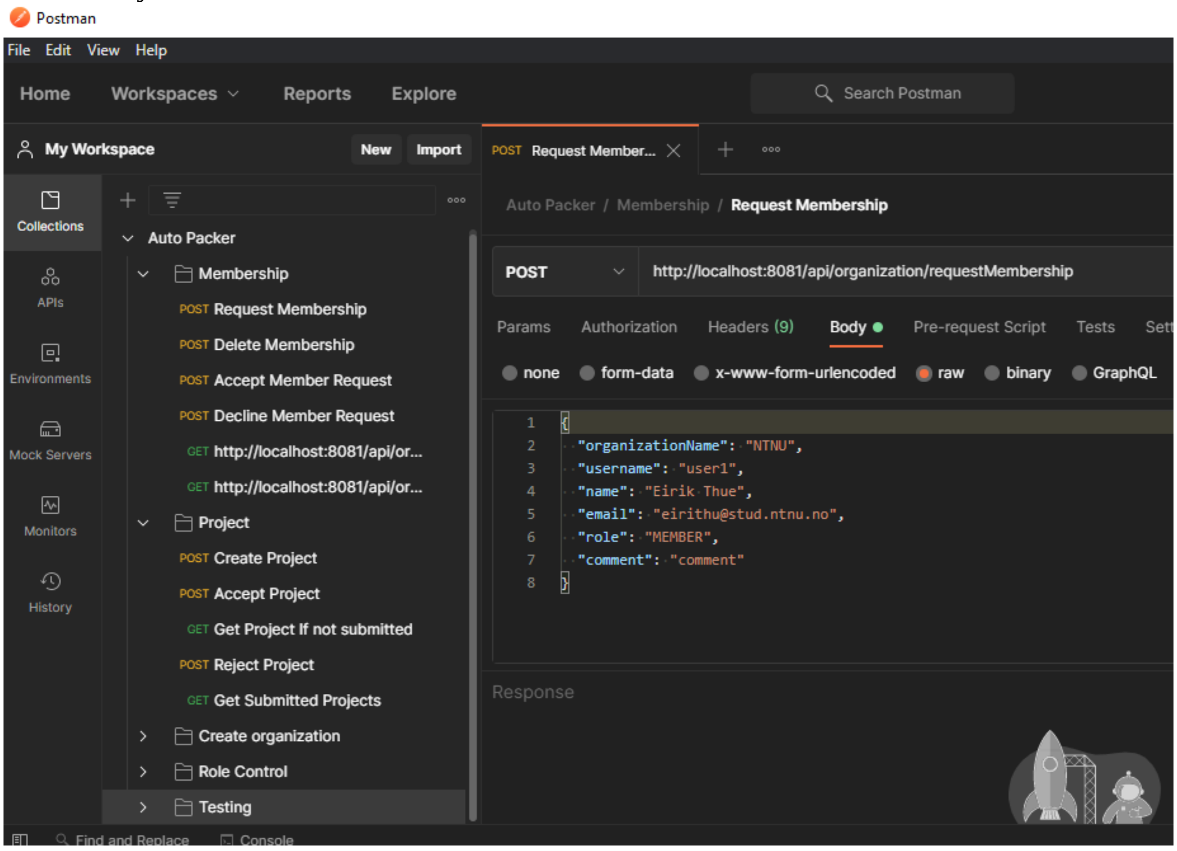
Module table stand mostly unchanged but are mentioned because it was needed to perform project in organizations.

4.4 Backend

The backend has a layered approach where controller will have to the verify that the information that is coming in is correct, the service layer handing the data and repository layer that saves it.

4.4.1 Postman

When I was building the backend, I was taking good usage out of Postman. It helps you with various task that you would have to have a front end to test. If we add the correct JSON body that the



POST http://localhost:8081/api/organization/requestMembership

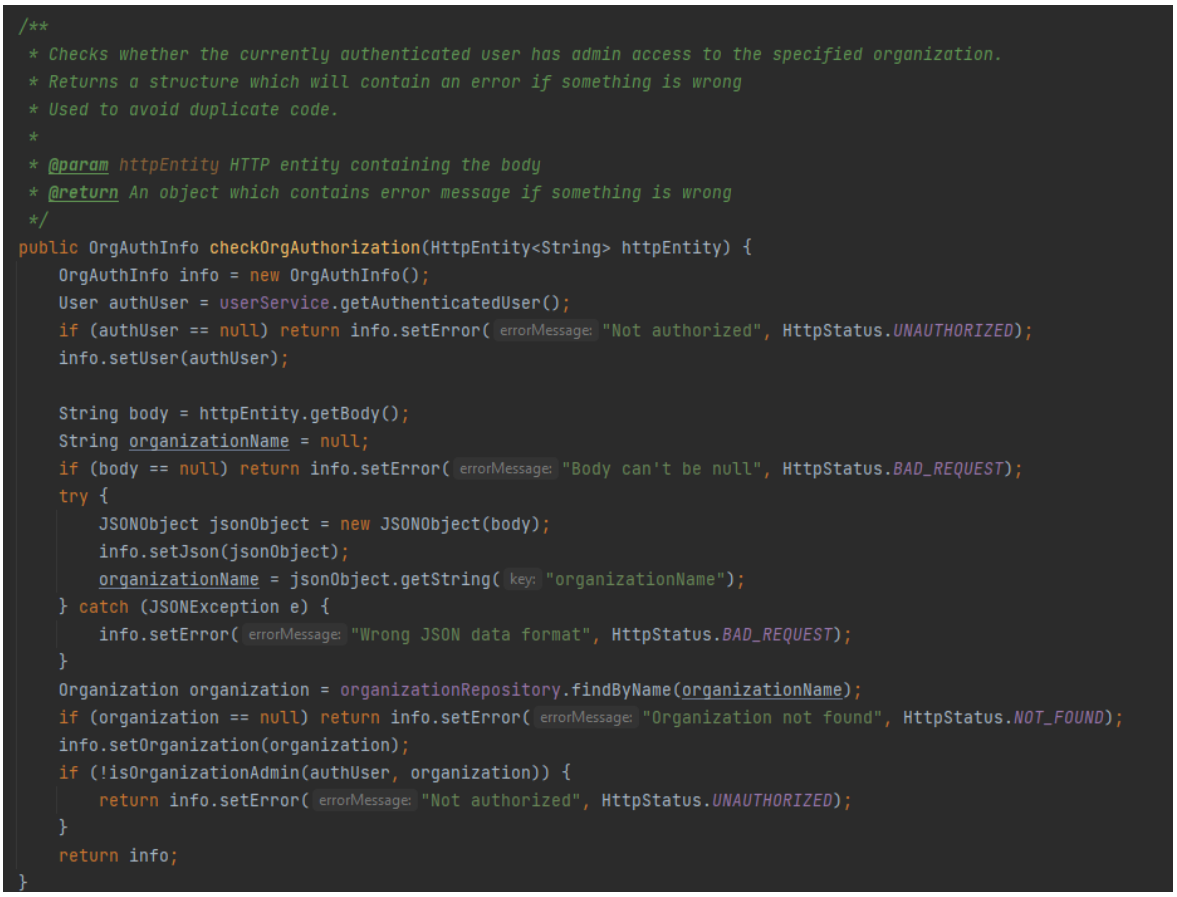
```
{
  "organizationName": "NTNU",
  "username": "user1",
  "name": "Eirik Thue",
  "email": "eirithu@stud.ntnu.no",
  "role": "MEMBER",
  "comment": "comment"
}
```

Figure 17: Postman test

4.4.2 Security Design

The website got their security for the people logging in, but the organization part did not have any way of distinguishing any of their users. We made use of what user sent the http request, if the user is not there or the body is null it will be dismissed right out of the gate. If the data is correct the formula will check if the user have admin access to the given organization.

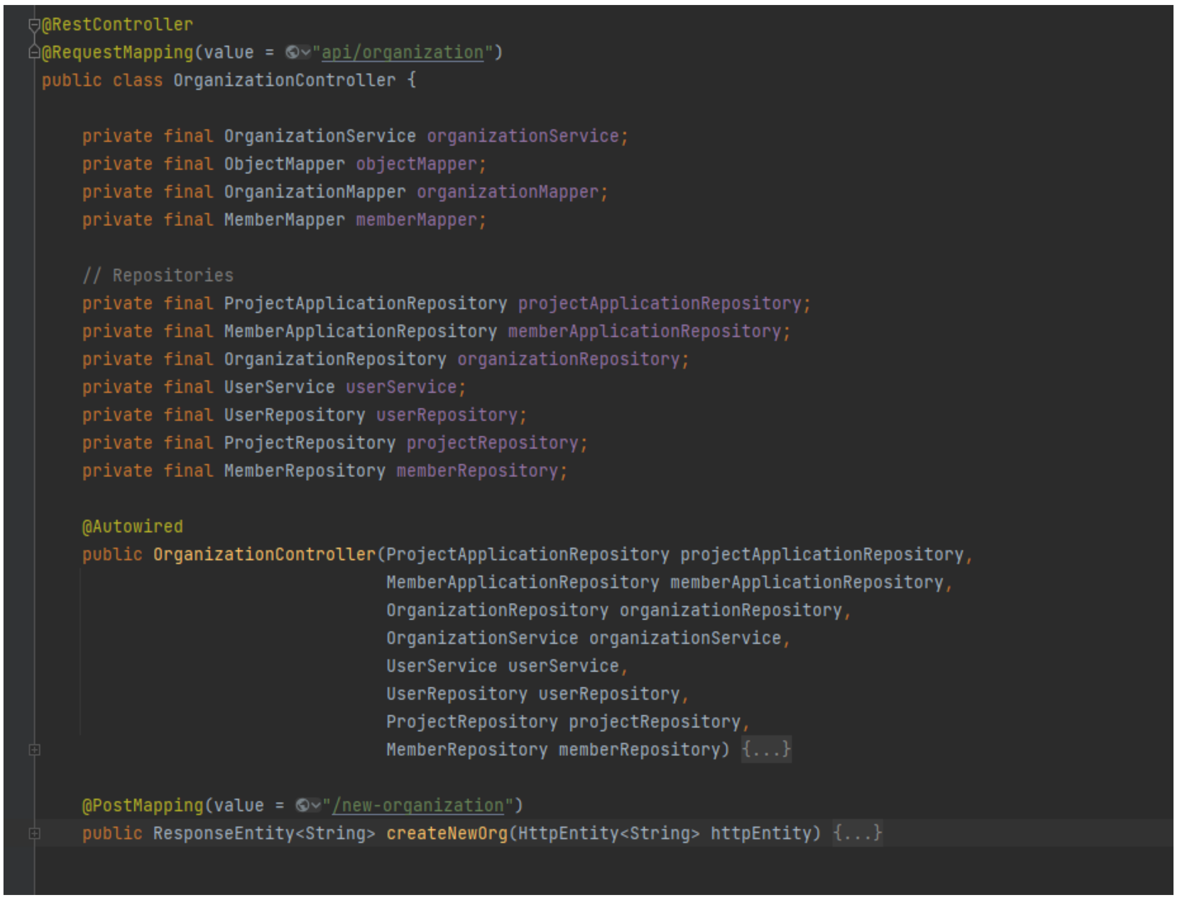
This formula will be used for most the instances where organization members need admin access.



```java
/**
 * Checks whether the currently authenticated user has admin access to the specified organization.
 * Returns a structure which will contain an error if something is wrong
 * Used to avoid duplicate code.
 *
 * @param httpEntity HTTP entity containing the body
 * @return An object which contains error message if something is wrong
 */
public OrgAuthInfo checkOrgAuthorization(HttpEntity<String> httpEntity) {
    OrgAuthInfo info = new OrgAuthInfo();
    User authUser = userService.getAuthenticatedUser();
    if (authUser == null) return info.setError( errorMessage: "Not authorized", HttpStatus.UNAUTHORIZED);
    info.setUser(authUser);

    String body = httpEntity.getBody();
    String organizationName = null;
    if (body == null) return info.setError( errorMessage: "Body can't be null", HttpStatus.BAD_REQUEST);
    try {
        JSONObject jsonObject = new JSONObject(body);
        info.setJson(jsonObject);
        organizationName = jsonObject.getString( key: "organizationName");
    } catch (JSONException e) {
        info.setError( errorMessage: "Wrong JSON data format", HttpStatus.BAD_REQUEST);
    }
    Organization organization = organizationRepository.findByName(organizationName);
    if (organization == null) return info.setError( errorMessage: "Organization not found", HttpStatus.NOT_FOUND);
    info.setOrganization(organization);
    if (!isOrganizationAdmin(authUser, organization)) {
        return info.setError( errorMessage: "Not authorized", HttpStatus.UNAUTHORIZED);
    }
    return info;
}
```

Figure 18: Security in backend

4.4.3 Organization Controller Layer

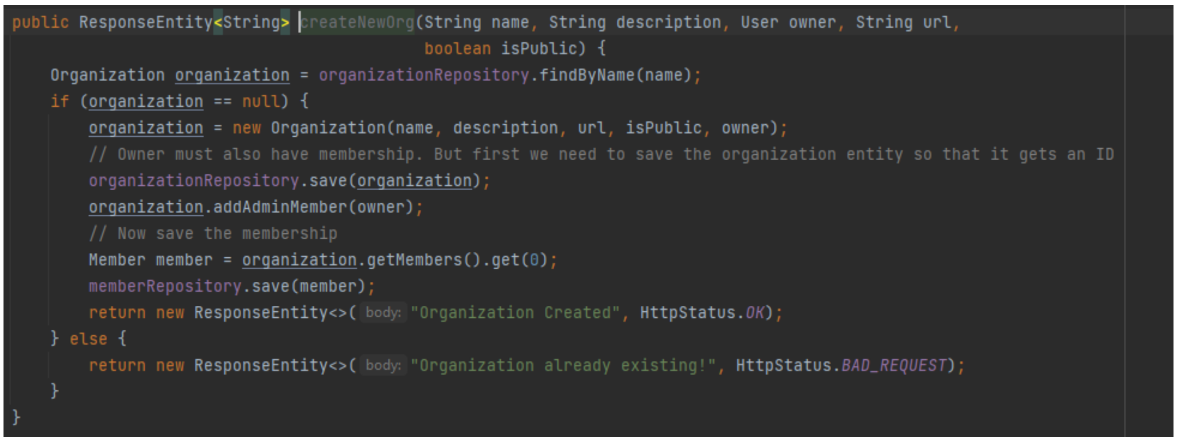


```java
@RestController
@RequestMapping(value = "api/organization")
public class OrganizationController {

    private final OrganizationService organizationService;
    private final ObjectMapper objectMapper;
    private final OrganizationMapper organizationMapper;
    private final MemberMapper memberMapper;

    // Repositories
    private final ProjectApplicationRepository projectApplicationRepository;
    private final MemberApplicationRepository memberApplicationRepository;
    private final OrganizationRepository organizationRepository;
    private final UserService userService;
    private final UserRepository userRepository;
    private final ProjectRepository projectRepository;
    private final MemberRepository memberRepository;

    @Autowired
    public OrganizationController(ProjectApplicationRepository projectApplicationRepository,
                                  MemberApplicationRepository memberApplicationRepository,
                                  OrganizationRepository organizationRepository,
                                  OrganizationService organizationService,
                                  UserService userService,
                                  UserRepository userRepository,
                                  ProjectRepository projectRepository,
                                  MemberRepository memberRepository) {...}

    @PostMapping(value = "/new-organization")
    public ResponseEntity<String> createNewOrg(HttpEntity<String> httpEntity) {...}
```

Figure 19: Organization controller layer

The way the controller is setup is that every resource contains their own mapping, as you can see above as an example. “API/organization/new-organization” is the path, this is true for all our resources. We use @Autowire for our controller to be able to collaborate with the other Repository/Service classes.

4.4.4 Organization Service Layer (Business logic)

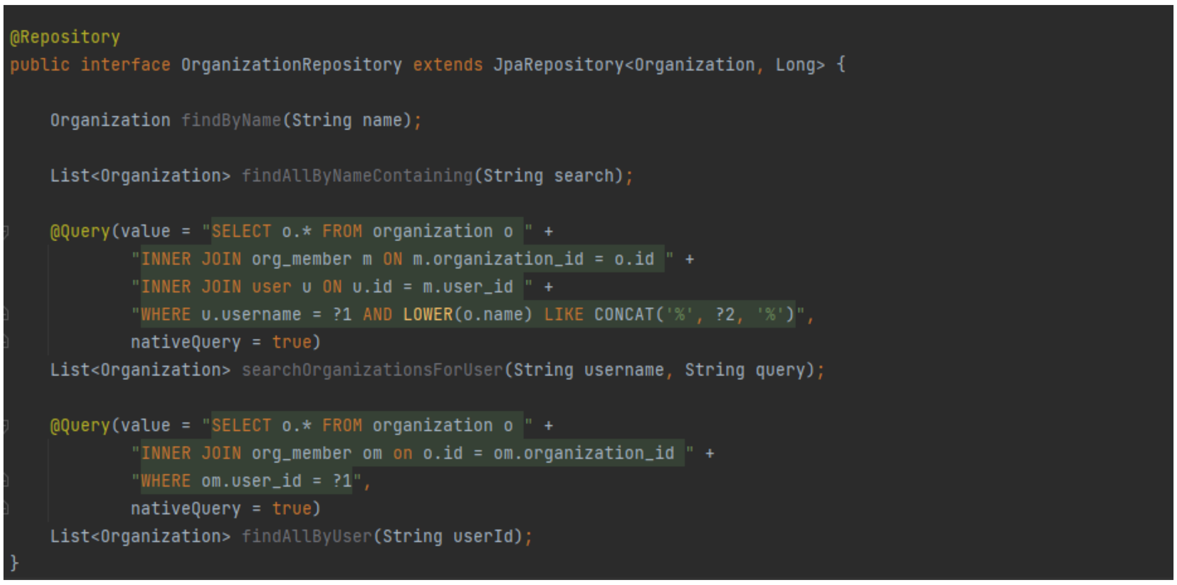


```java
public ResponseEntity<String> createNewOrg(String name, String description, User owner, String url,
                                           boolean isPublic) {
    Organization organization = organizationRepository.findByName(name);
    if (organization == null) {
        organization = new Organization(name, description, url, isPublic, owner);
        // Owner must also have membership. But first we need to save the organization entity so that it gets an ID
        organizationRepository.save(organization);
        organization.addAdminMember(owner);
        // Now save the membership
        Member member = organization.getMembers().get(0);
        memberRepository.save(member);
        return new ResponseEntity<>( body: "Organization Created", HttpStatus.OK);
    } else {
        return new ResponseEntity<>( body: "Organization already existing!", HttpStatus.BAD_REQUEST);
    }
}
```

Figure 20: Organization service layer

The business logic for all the different calls check, as in the case above if the organization is already created. If it is created, you would get a bad request, if it is not, you can go ahead, and it gets created with the user as their admin. For all the request you send either a HttpStatus.OK or HttpStatus.Bad_Request back to the user depending on the outcome.

4.4.5 Organization Repository Layer



```
@Repository
public interface OrganizationRepository extends JpaRepository<Organization, Long> {

    Organization findByName(String name);

    List<Organization> findAllByNameContaining(String search);

    @Query(value = "SELECT o.* FROM organization o " +
            "INNER JOIN org_member m ON m.organization_id = o.id " +
            "INNER JOIN user u ON u.id = m.user_id " +
            "WHERE u.username = ?1 AND LOWER(o.name) LIKE CONCAT('%', ?2, '%')",
            nativeQuery = true)
    List<Organization> searchOrganizationsForUser(String username, String query);

    @Query(value = "SELECT o.* FROM organization o " +
            "INNER JOIN org_member om on o.id = om.organization_id " +
            "WHERE om.user_id = ?1",
            nativeQuery = true)
    List<Organization> findAllByUser(String userId);
}
```

Figure 21: Repository Layer

For the repository layer we used JPA to save and retrieve the data from the database. We the reason JPA is that it gives us a way of writing our own queries when it comes to retrieving data, you can speed up the process with being able to be writing your own queries.
Giving us the control, we wanted of the data we retrieved from the database as you can see in the picture above.

4.5 Front end

4.5.1 Graphical Design

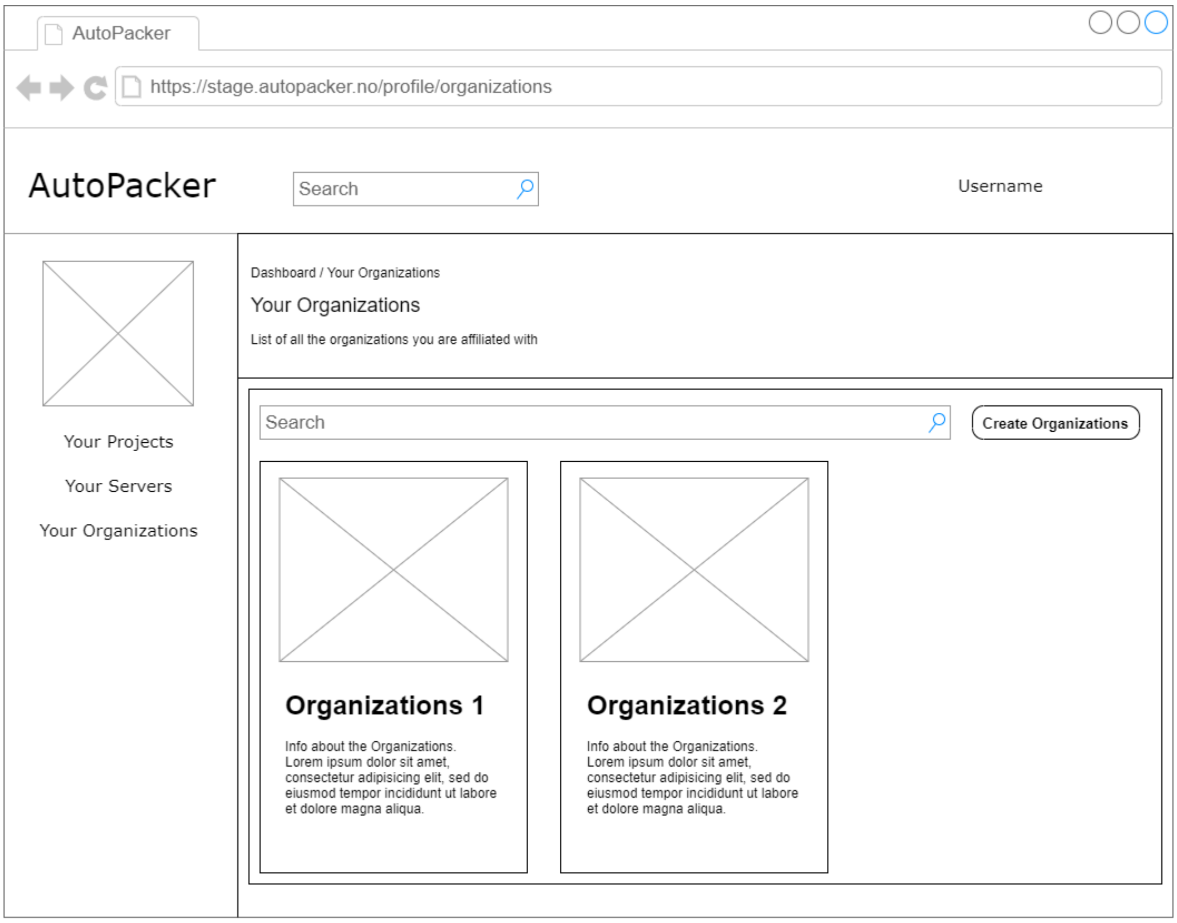
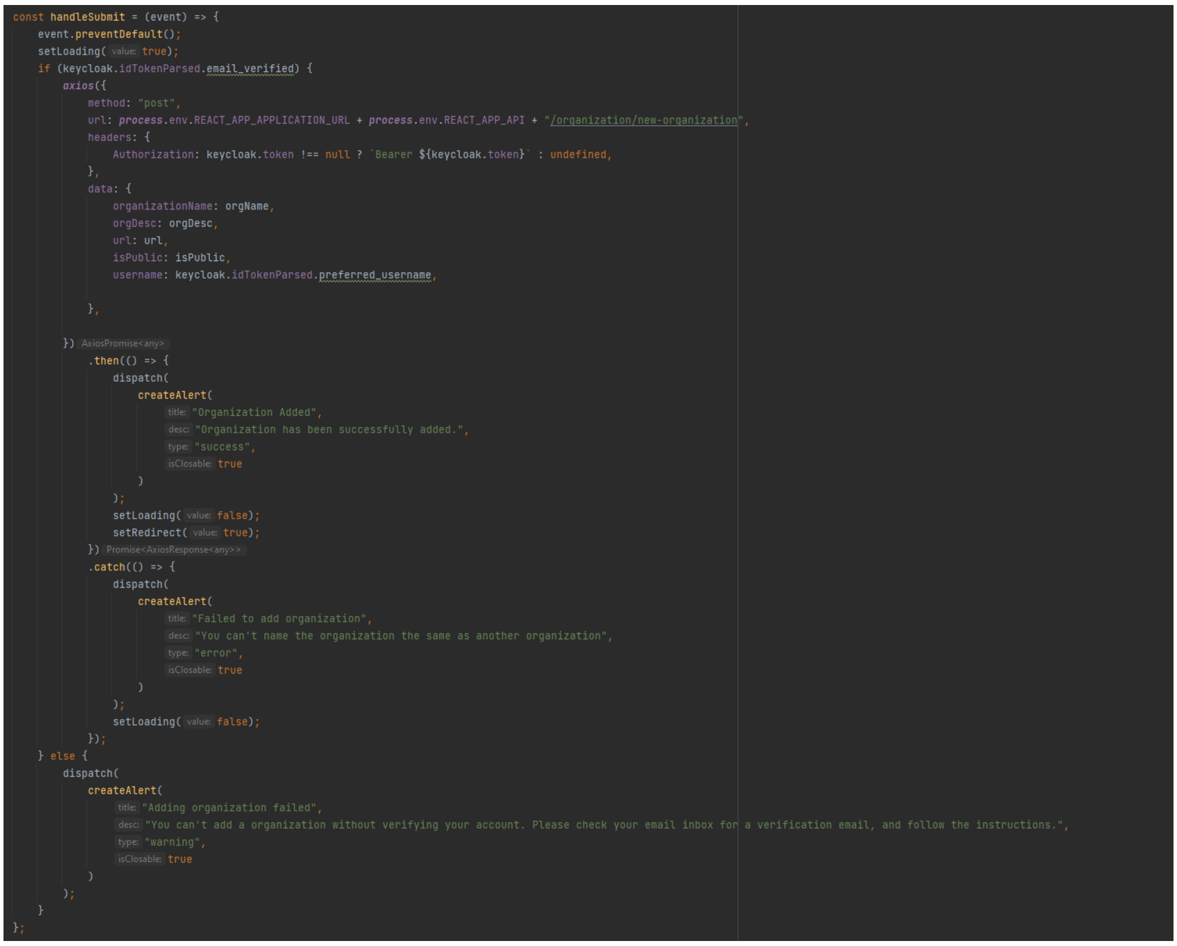

AutoPacker
https://stage.autopacker.no/profile/organizations
AutoPacker
Search
Username
Your Projects
Your Servers
Your Organizations
Dashboard / Your Organizations
Your Organizations
List of all the organizations you are affiliated with
Search
Create Organizations
Organizations 1
Info about the Organizations. Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
Organizations 2
Info about the Organizations. Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Figure 22: Graphical design wireframe

Due to the nature of the project being already being established, I continued with the same style of resources the project was founded up on which was Ant Design and heavily invested in React.js which will continue to be the main component in the front end of the website.

The design philosophy that is used, is taking the information and adding them to cards. These cards contain a small amount of the information, usually the name, description and picture like in the wireframe above. This is the core design choices for displaying information in quantity in Auto Packer.

4.5.2 API Communication



```javascript
const handleSubmit = (event) => {
    event.preventDefault();
    setLoading( value: true);
    if (keycloak.idTokenParsed.email_verified) {
        axios({
            method: "post",
            url: process.env.REACT_APP_APPLICATION_URL + process.env.REACT_APP_API + "/organization/new-organization",
            headers: {
                Authorization: keycloak.token !== null ? `Bearer ${keycloak.token}` : undefined,
            },
            data: {
                organizationName: orgName,
                orgDesc: orgDesc,
                url: url,
                isPublic: isPublic,
                username: keycloak.idTokenParsed.preferred_username,

            },

        }) AxiosPromise<any>
            .then(() => {
                dispatch(
                    createAlert(
                        title: "Organization Added",
                        desc: "Organization has been successfully added.",
                        type: "success",
                        isClosable: true
                    )
                );
                setLoading( value: false);
                setRedirect( value: true);
            }) Promise<AxiosResponse<any>>
            .catch(() => {
                dispatch(
                    createAlert(
                        title: "Failed to add organization",
                        desc: "You can't name the organization the same as another organization",
                        type: "error",
                        isClosable: true
                    )
                );
                setLoading( value: false);
            });
    } else {
        dispatch(
            createAlert(
                title: "Adding organization failed",
                desc: "You can't add a organization without verifying your account. Please check your email inbox for a verification email, and follow the instructions.",
                type: "warning",
                isClosable: true
            )
        );
    }
};
```

Figure 23: API Communication

Every API call in the front end is built up like in the picture above. This one is under a handle submit so it does not create a new organization without an event happens.
The request checks if the account that makes the request is verified, if it is verified it would build up the request. Giving it a method (In this case is a post), URL to contact the right API path, header taking a keycloak token for verification and the data.

If the creation of an organization is a success the user will be prompted with a success alert, otherwise a failed one based on what happen.

4.5.3 Organization Views

4.5.3.1 Your Organizations

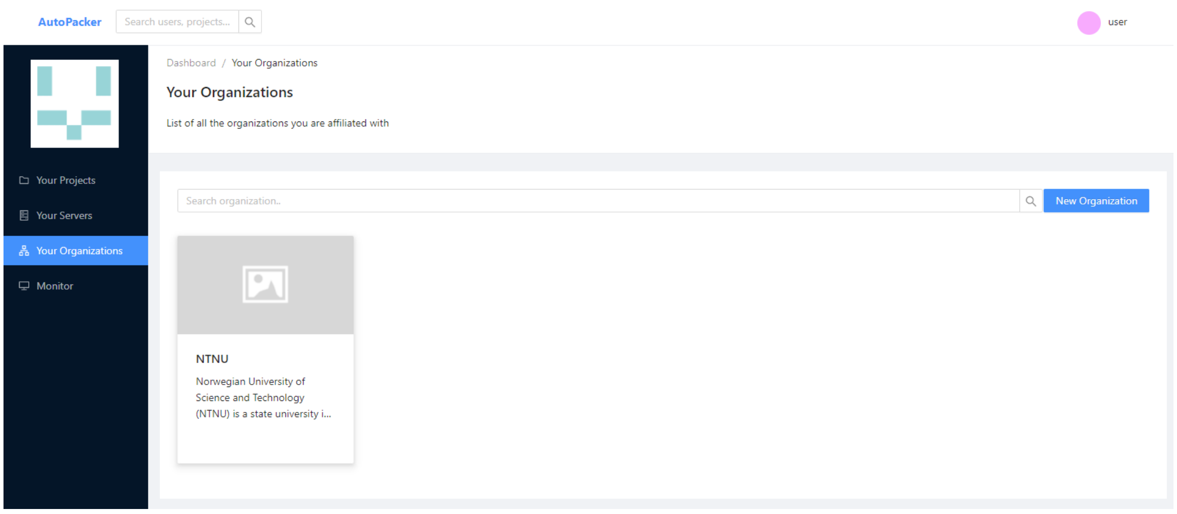

AutoPacker
Search users, projects...
user
Your Projects
Your Servers
Your Organizations
Monitor
Dashboard / Your Organizations
Your Organizations
List of all the organizations you are affiliated with
Search organization..
New Organization
NTNU
Norwegian University of Science and Technology (NTNU) is a state university i...

Figure 24: Your organization

This is the dashboard where you will see all the organizations you are apart of, up to the right you will see that you can create a new organization based on your own needs.

4.5.3.2 Create Organization

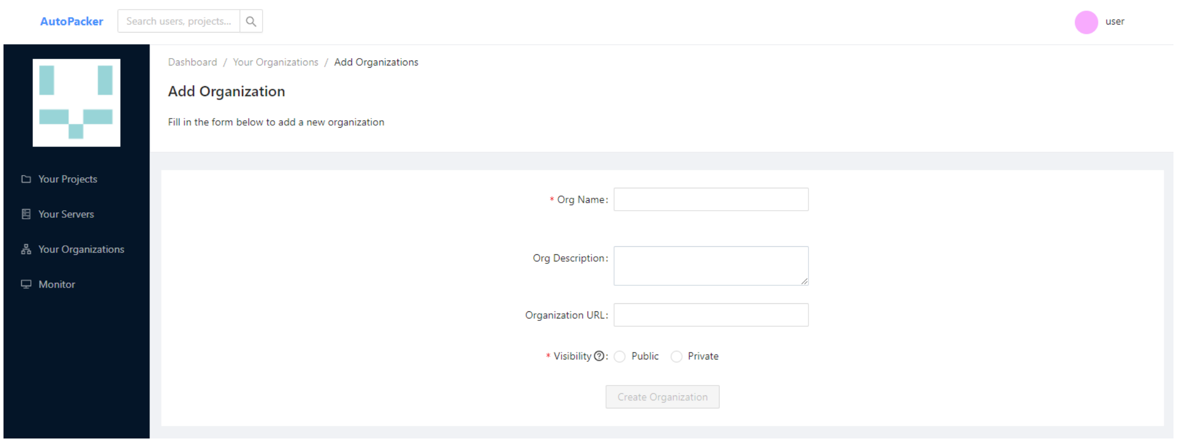
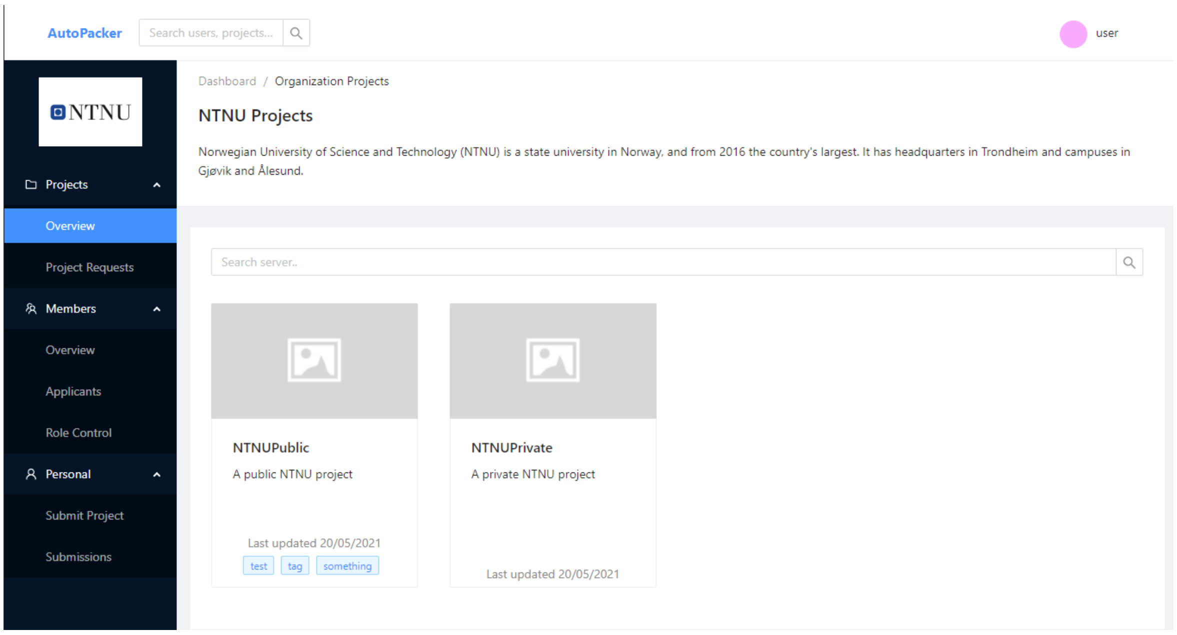


Figure 25: Create your own organization

In this picture you can see the form you need to fill out create your own organization, name and visibility is mandatory while description and URL is optional if you want to provide any extra information.

4.5.3.3 Organization Dashboard



NTNU Projects

Norwegian University of Science and Technology (NTNU) is a state university in Norway, and from 2016 the country's largest. It has headquarters in Trondheim and campuses in Gjøvik and Ålesund.

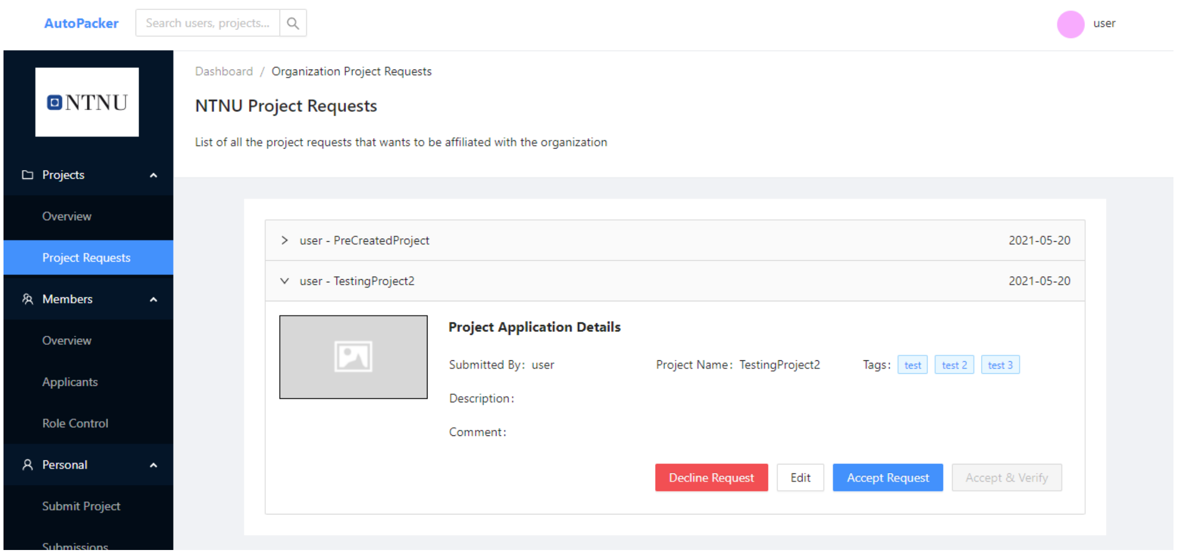
NTNUPublic
A public NTNU project
Last updated 20/05/2021
test tag something

NTNUPrivate
A private NTNU project
Last updated 20/05/2021

Figure 26: Organization dashboard

When you have joined or setup your own organization you will be greeted to your organization overview with the projects that is tied to your organization.

4.5.3.4 Accepting Project Request



NTNU Project Requests

List of all the project requests that wants to be affiliated with the organization

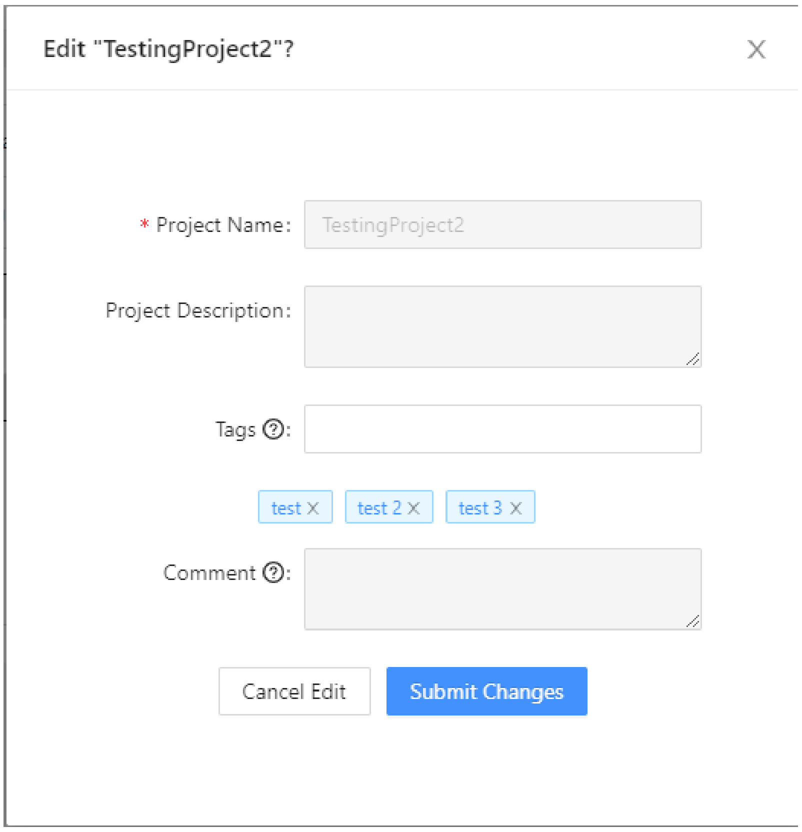
user - PreCreatedProject 2021-05-20
user - TestingProject2 2021-05-20

Project Application Details
Submitted By: user
Project Name: TestingProject2
Tags: test test 2 test 3
Description:
Comment:

Figure 27: Accept project request

Now if you have Admin access to the organization, you can see the Project Request part of the organization page. Here you can Accept/Decline different applications for people to submit their projects. When an application is accepted the organization is taking over the ownership of the project.

4.5.3.5 Edit Project Request (Admin View)


Edit "TestingProject2"?

* Project Name: TestingProject2

Project Description:

Tags ?:

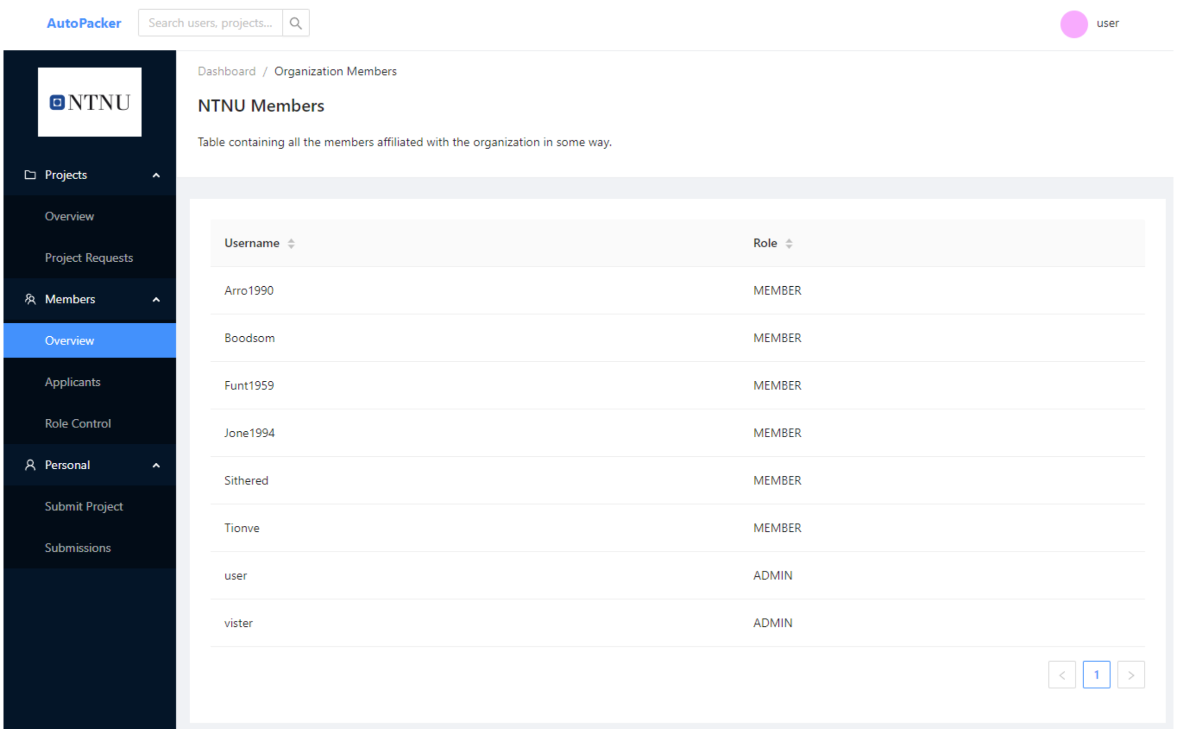
test × test 2 × test 3 ×

Comment ?:

Figure 28: Admin edit project

If an admin wants to add or change the tags on a project to correct it. They can do it in the form above.

4.5.3.6 Member overview



NTNU Members

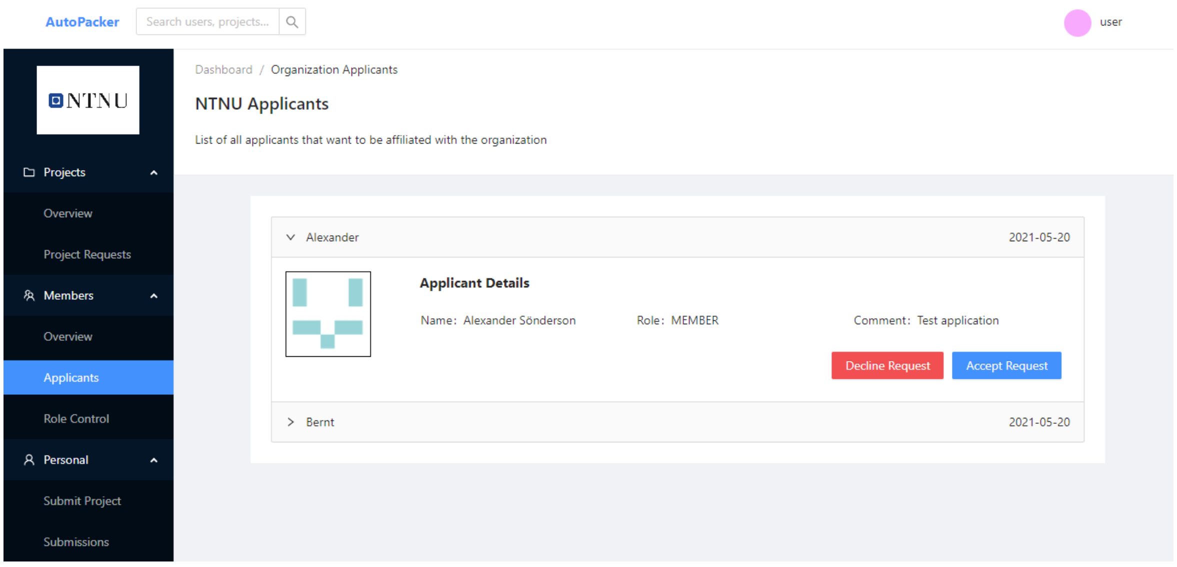
Table containing all the members affiliated with the organization in some way.

Username	Role
Arro1990	MEMBER
Boodsom	MEMBER
Funt1959	MEMBER
Jone1994	MEMBER
Sithered	MEMBER
Tionve	MEMBER
user	ADMIN
vister	ADMIN

Figure 29: Member overview

All the users in an organization can see what users there are and what roles they have within the organization.

4.5.3.7 Member applications



NTNU Applicants

List of all applicants that want to be affiliated with the organization

Alexander 2021-05-20

Applicant Details

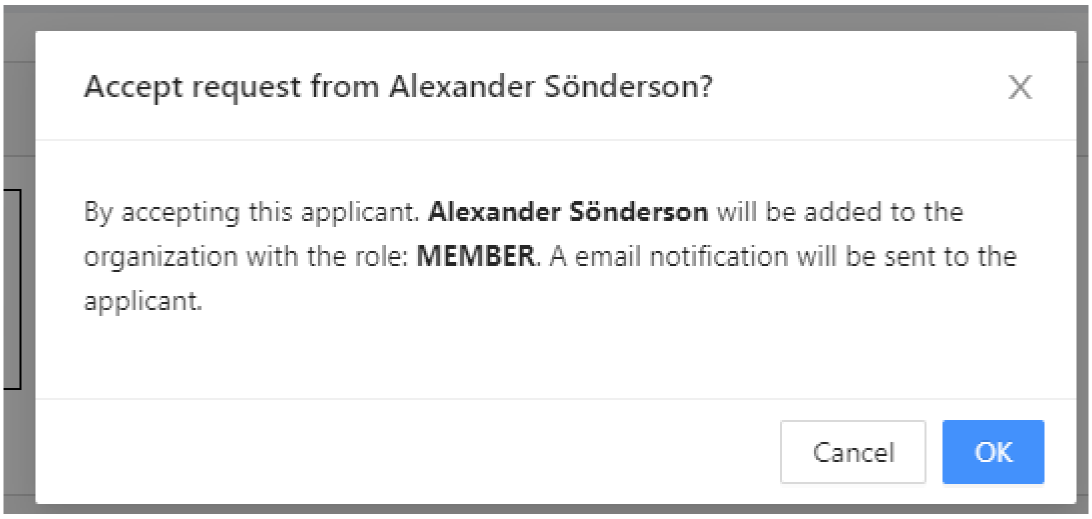
Name: Alexander Sönderson Role: MEMBER Comment: Test application

Bernt 2021-05-20

Figure 30: Member applications

If you have admin rights in the organization, you can see the applicants screen and decline or accept applicants. If you click accept you can see the picture below to confirm if you click decline you will not get a prompt.

4.5.3.8 Member application (If Accepted)



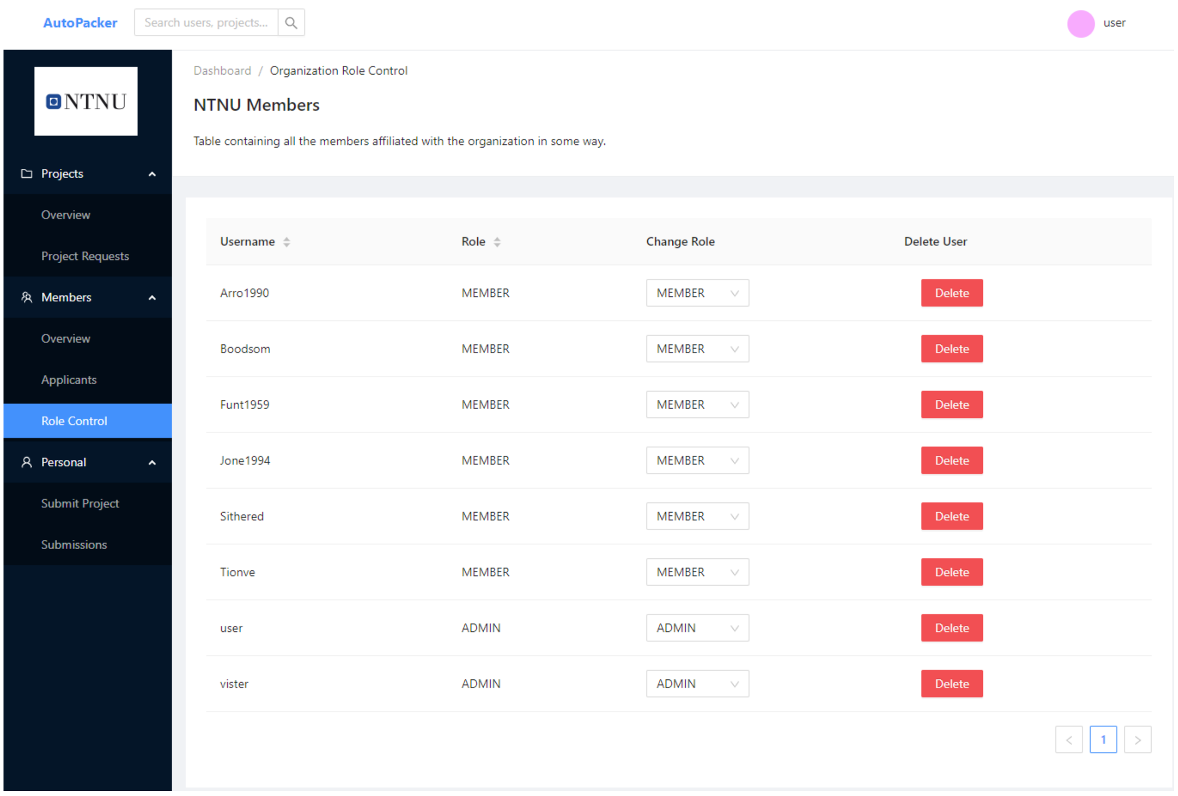
Accept request from Alexander Sönderson?

By accepting this applicant. **Alexander Sönderson** will be added to the organization with the role: **MEMBER**. A email notification will be sent to the applicant.

Figure 31: Member application if accept

If accept is clicked, you must confirm that you accept this person into your organization.

4.5.3.9 Role Control



NTNU Members

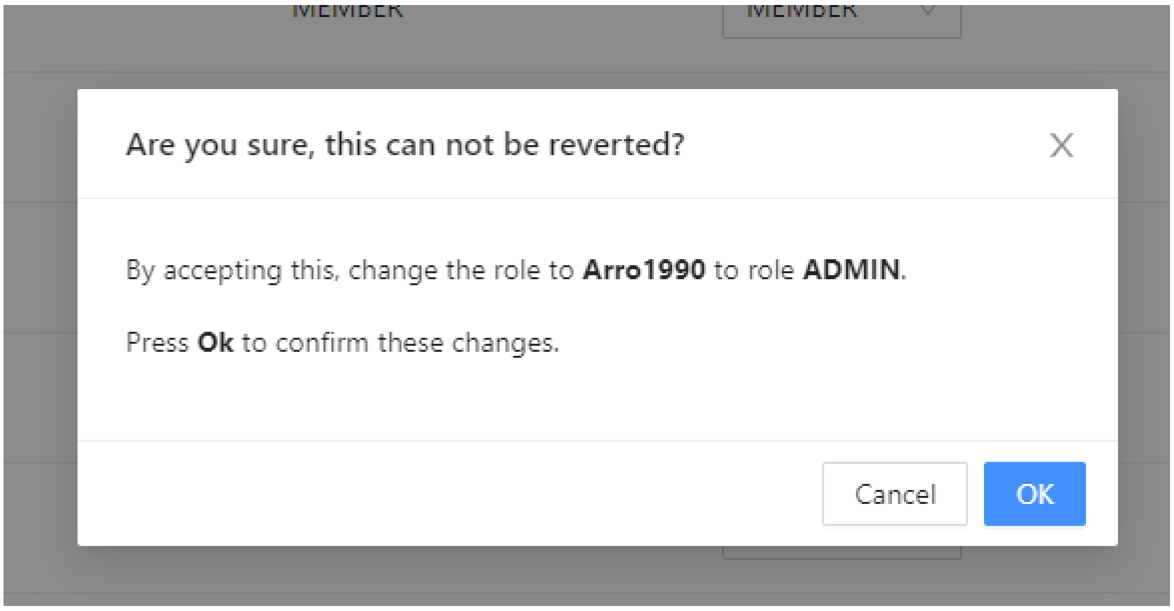
Table containing all the members affiliated with the organization in some way.

Username	Role	Change Role	Delete User
Arro1990	MEMBER	MEMBER	Delete
Boodsom	MEMBER	MEMBER	Delete
Funt1959	MEMBER	MEMBER	Delete
Jone1994	MEMBER	MEMBER	Delete
Sithered	MEMBER	MEMBER	Delete
Tionve	MEMBER	MEMBER	Delete
user	ADMIN	ADMIN	Delete
vister	ADMIN	ADMIN	Delete

Figure 32: Role Control

Role control can be only seen by admin in the organization, it is a view that makes it able to change roles or delete members from your organization.

4.5.3.10 Confirm role change



Are you sure, this can not be reverted?

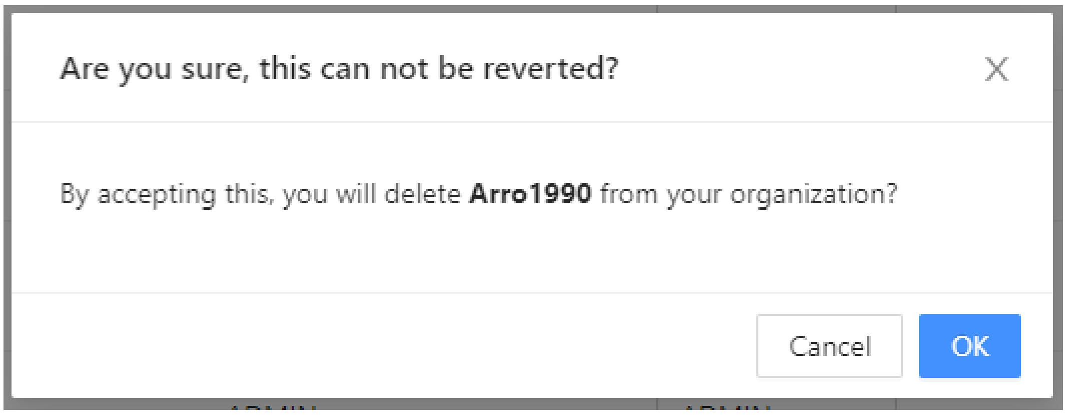
By accepting this, change the role to **Arro1990** to role **ADMIN**.

Press **Ok** to confirm these changes.

Figure 33: Confirm roles

If you want to change roles of a user you will get a prompt on accepting the change, so no mistakes are made.

4.5.3.11 Confirm deletion



Are you sure, this can not be reverted?

By accepting this, you will delete **Arro1990** from your organization?

Figure 34: Confirm deletion

When trying to delete someone from your organization you will get a prompt you have to confirm to avoid mistakenly delete someone from your organization.

4.5.3.12 Submitting project

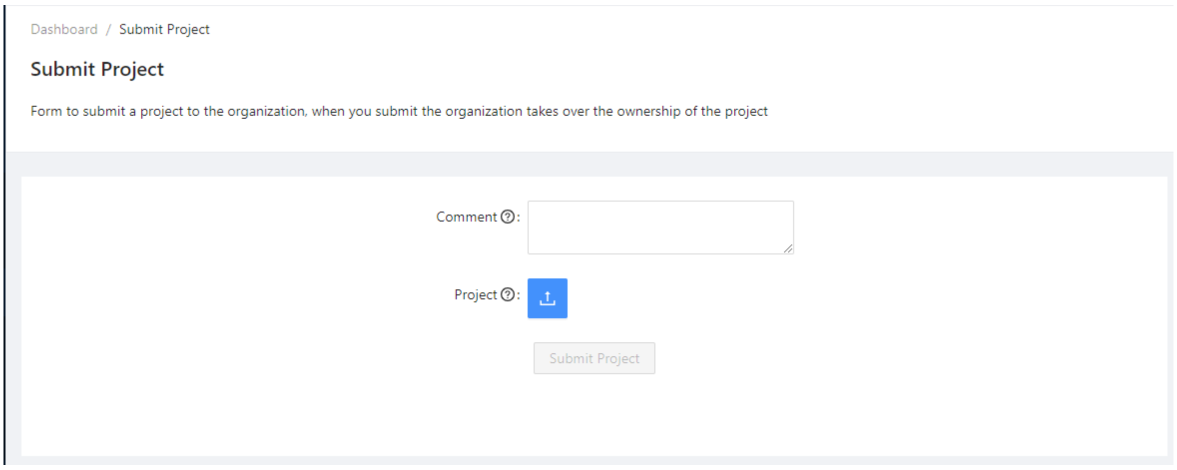

Dashboard / Submit Project
Submit Project
Form to submit a project to the organization, when you submit the organization takes over the ownership of the project
Comment ⓘ:
Project ⓘ:
Submit Project

Figure 35: Submit project

In this view you can select one of your personal projects and submit them to the organization, this will remove your access to the project from personal view. But you can access it through the organization.

4.5.3.13 Choosing project

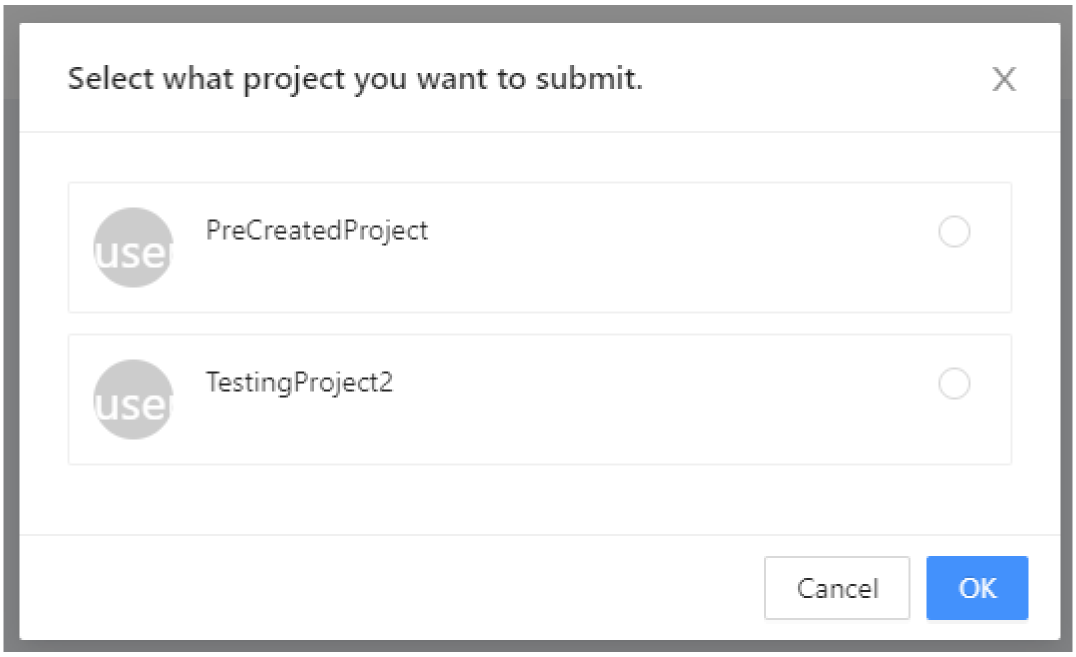
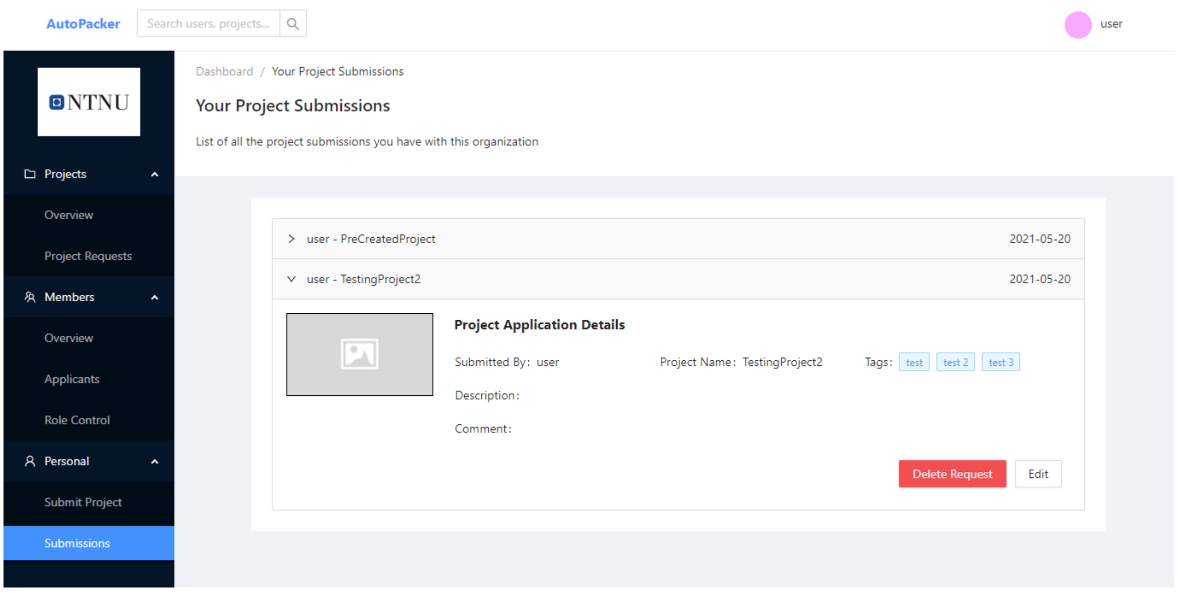


Figure 36: Choosing the project

Before submitting you must choose which project to submit, this list is gathered from your own personal project list.

4.5.3.14 Project submission list



Your Project Submissions

List of all the project submissions you have with this organization

user - PreCreatedProject 2021-05-20

user - TestingProject2 2021-05-20

Project Application Details

Submitted By: user Project Name: TestingProject2 Tags: test test 2 test 3

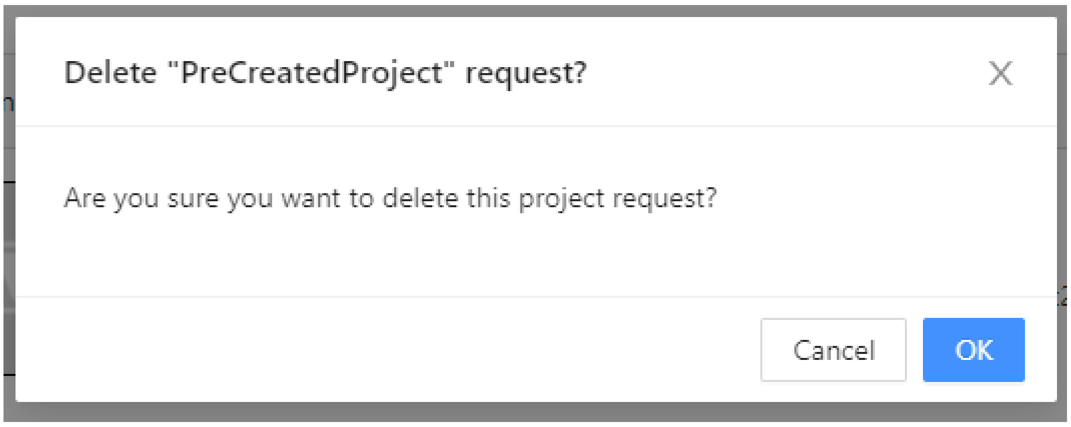
Description:

Comment:

Figure 37: Submission list

All the projects will be added to a submission list before they get submitted to the organization. Here you can delete your request or change it if you forget to add something.

4.5.3.15 Delete project request


Delete "PreCreatedProject" request?

Are you sure you want to delete this project request?

Figure 38: Delete project request

By selecting delete request you will get a prompt telling you to confirm the deletion.

4.5.3.16 Edit project request

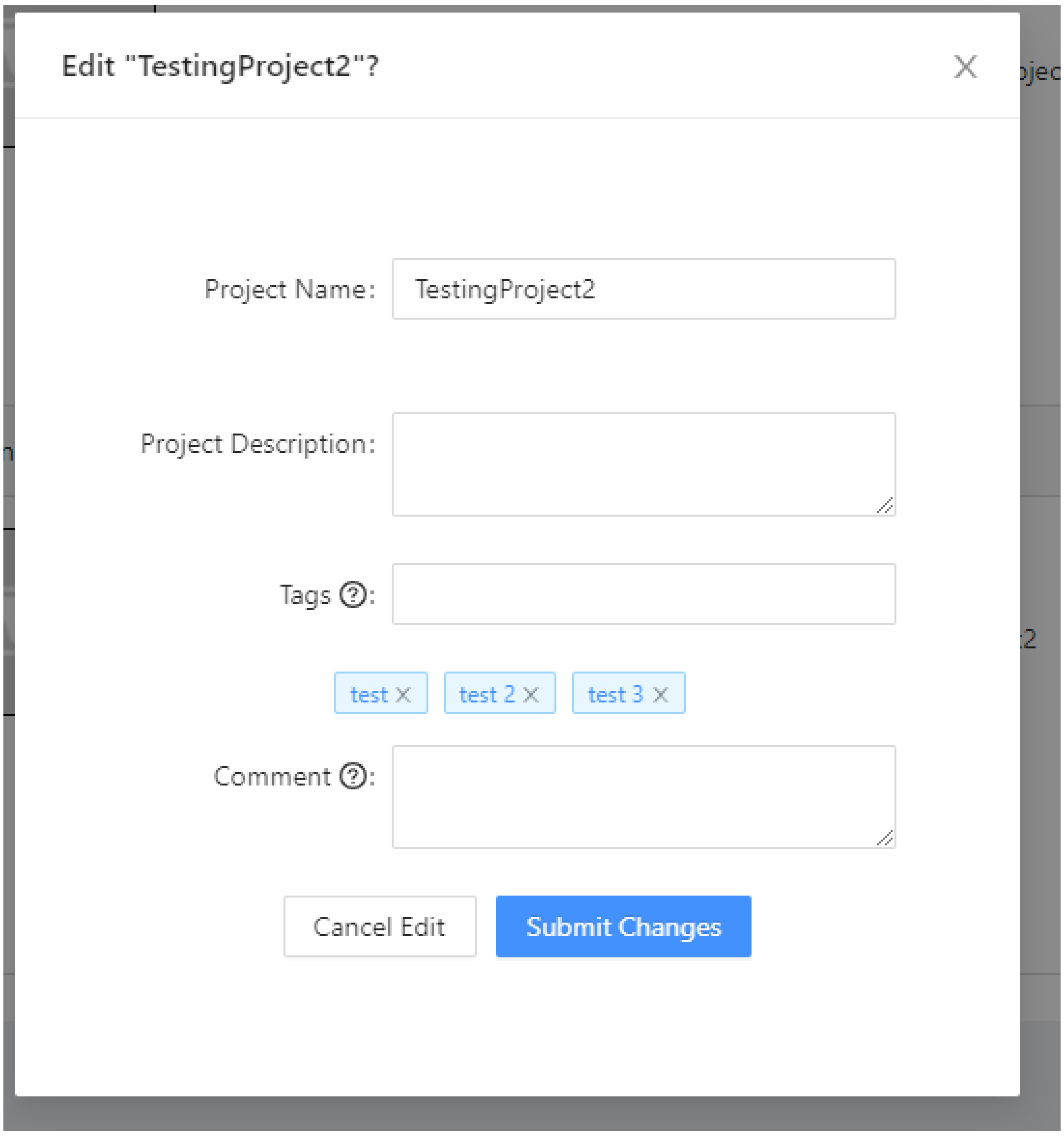


Figure 39: Edit project request

If you want to edit your project information before it is submitted you have chance to do so, hit submit to save the changes.

4.5.3.17 Organization Projects

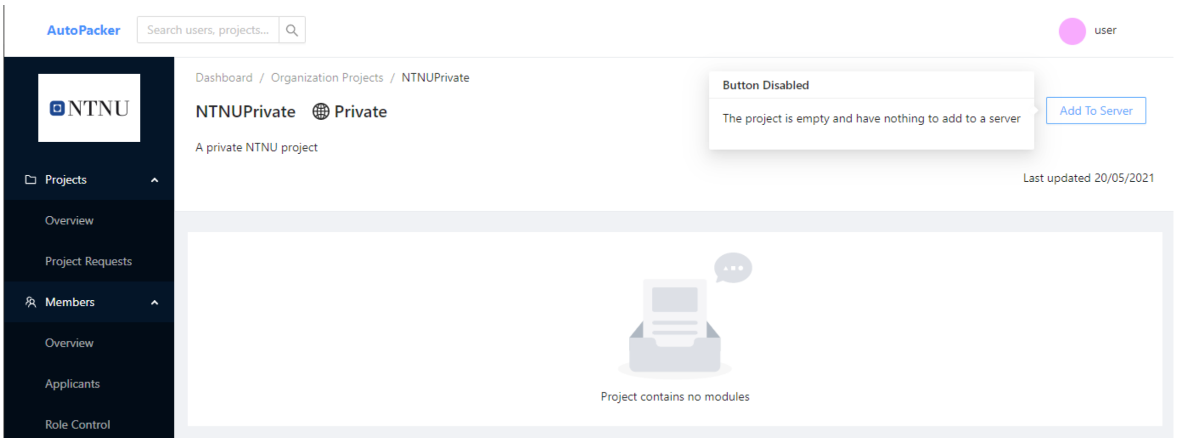


Figure 40: Organization projects

Organization projects can be viewed and added to your own server. If the project has no modules, you would get a warning and get told the project is empty.

4.5.3.18 Add project to server from organization

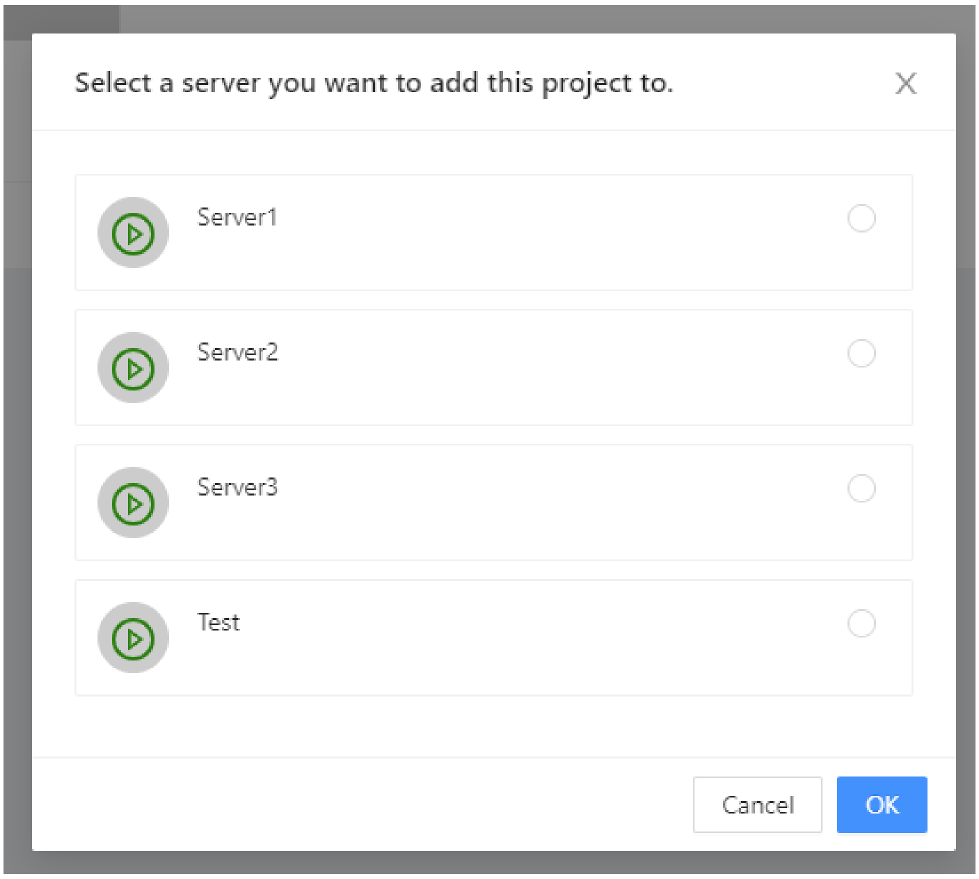


Figure 41: Add project to server

If the project has modules, you can click add to server, then you are presented with a list of the servers you have setup in the “Your Servers” sections of the website. When selecting the right server, it is getting added.

5. Reflection

5.1 Scrum Workflow

The beginning of the project went great; however, it took a turn for the worse with the constant COVID-19 restrictions and lockdowns. It started with stabile motivation and a willing to learn but being in the project alone made prioritization and strict workflow made the content produced less than what I would have liked.
All sprints were fulfilled to some capacity, but not having the social interactions as working in teams usually have was sorely missed.

5.2 Front-end

The section of further develop the application based on Ant Design, since it was already put in place for me and being the choice of the previous developers. I did not need to choose just further develop the product.

Overall, I am happy with how it turned out, there have been some changes to the UI. Although the main changes have happened in back end rather to front-end, due to there was some proof of concept for the organization part already there. So minimal changes had to take place to make the organization fully working.

What we ended up with is an organization section for Auto Packer where anyone can create their own organization, they become admin of the organization when they create it and can invite anyone to join. You have control over who can join, you can accept or reject members based on what they ended up with in their request to join. When they joined you can control their role and if you want to delete them from the organization, in any case someone is not fit to be there or have left the organization. Projects have been fully integrated with organizations, so they have ownership, and any user can add the project from their organizations to their servers.

There are some lacking features that I wanted to implement but did not have time for like:

- Management tool for project
- Setting view for organization project
- More flexible roles
- Communication outlet within the organization

With that in mind, what was done is aimed on a complete package that works, rather than too ambitious goals that is not achievable in the limited time of this project and I did not know much about React to begin with. Makes me think I got far in this project and delivered a working package.

5.3 Back end

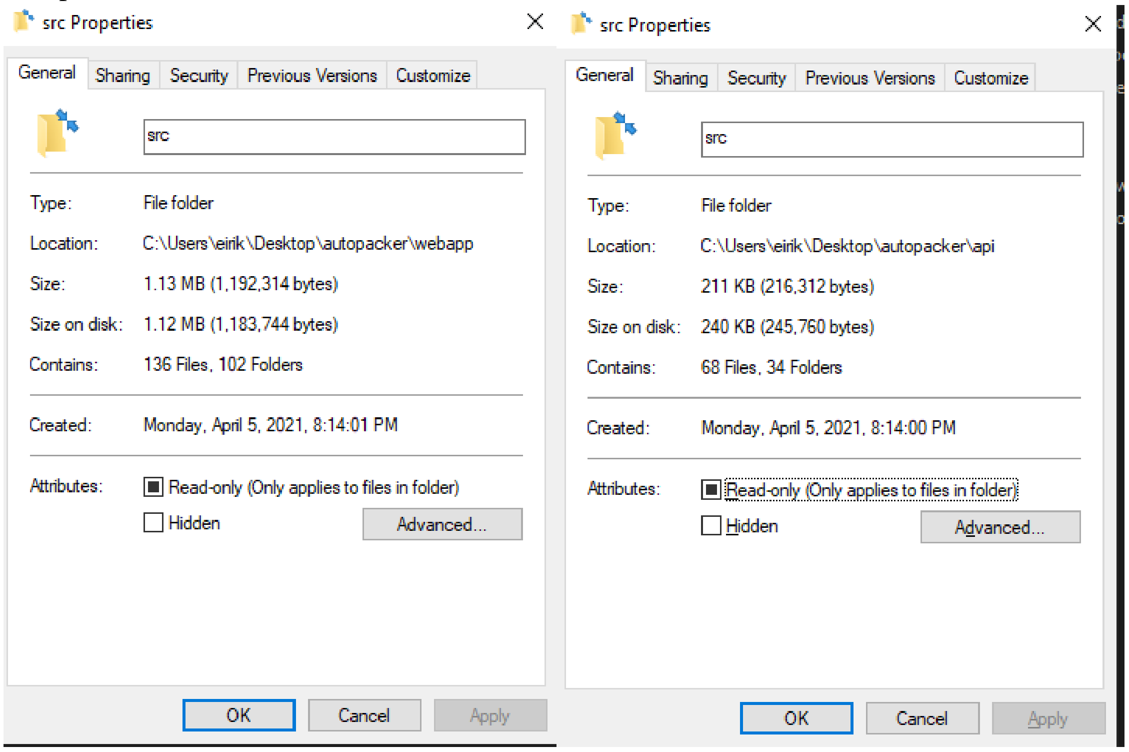
Starting off the back end looked way different and probably what went through the most changes during this project. It was a microservice format for the APIs, so there were 4 different ones instead of one.

This made a lot of issues in relation between organization and project elements, the roughest part with working on the back end would be when we decided to merge them, we were working on getting roles into organization where which was a big change in itself but straight after came the merge which made most of the API calls not work at all.
But that's how development are, you might not be the one destroying, but you are still the one that are ensured to make it keep working.

The result with roles in the organization also made it safer with organizations being able to know what different users should have access to or not.

5.4 Code Quality

The code quality has most definitely hit and miss some places, thankfully we had a CI/CD system, we use SonarCloud to track bugs and duplications. It made it easier to fix mistakes that I had done in the code, however some code could be even more abstract and reuse of components.



src
Type: File folder
Location: C:\Users\eirik\Desktop\autopacker\webapp
Size: 1.13 MB (1,192,314 bytes)
Size on disk: 1.12 MB (1,183,744 bytes)
Contains: 136 Files, 102 Folders
Created: Monday, April 5, 2021, 8:14:01 PM
Attributes: ■ Read-only (Only applies to files in folder)
☐ Hidden

src
Type: File folder
Location: C:\Users\eirik\Desktop\autopacker\api
Size: 211 KB (216,312 bytes)
Size on disk: 240 KB (245,760 bytes)
Contains: 68 Files, 34 Folders
Created: Monday, April 5, 2021, 8:14:00 PM
Attributes: ■ Read-only (Only applies to files in folder)
☐ Hidden

Figure 42: Webapp Source / API Source

Coming blind into an already big and growing project, with 136 files for the front end and 68 files for the back end, it is a lot to learn and get used to. While you learning what goes where there could be some better reuse of code.

5.5 Limitations

The organization have minimal control over what to do with projects after they are submitted, I wanted to implement more but did not have time to do that. My plan is to keep working on this project at least for the parts I feel I did not get to finish.

6. Conclusion

The purpose with this project was to give insight in some open-source project while also developing the organization part of the Auto Packer.

I feel I achieved a new version of Auto Packer that added value to the product, the application goes some new feature added to itself that was not there before. The organization system can be taken in use right out of the box with little explanation for the for the different parts of the site.

This was a steep learning curves in terms of how many technologies were in used there but I am happy I was given the chance on this project. Leaving the project fully working and open for the next person that want to venture into the project was always the goal.

Personally, I really liked the project and I'm open to further expand it down the line. As I mentioned there was more I wanted to expand upon but this is it for this time.

Bibliography

[1] "General Data Protection Regulation," [Online]. Available: https://gdpr-info.eu/.

[2] Datatilsynet. [Online]. Available: https://www.datatilsynet.no/en/.

[3] A. Birkeland, 08 10 2020. [Online]. Available: https://www.forskningsetikk.no/en/resources/the-research-ethics-library/legal-statutes-and-guidelines/the-personal-data-act/.

[4] J. Gisle, "SNL," 2018. [Online]. Available: https://snl.no/Lovdata.

[5] TechTarget. [Online]. Available: https://searchsqlserver.techtarget.com/definition/hashing.

[6] Commons, Creative, [Online]. Available: https://www.educative.io/edpresso/what-is-hashing.

[7] "Upload WIkimedia," [Online]. Available: https://upload.wikimedia.org/wikipedia/commons/thumb/d/da/Hash_function.svg/1200px-Hash_function.svg.png.

[8] "Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/Security_token.

[9] "JSON Web Tokens," [Online]. Available: https://jwt.io/introduction.

[10] K. Sandoval, "Nordic APIs," [Online]. Available: https://nordicapis.com/why-cant-i-just-send-jwts-without-oauth/.

[11] "Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/HTTPS.

[12] J. Melnick, "NetWrix," 15 05 2018. [Online]. Available: https://blog.netwrix.com/2018/05/15/top-10-most-common-types-of-cyber-attacks/.

[13] RedHat, "RedHat," [Online]. Available: https://www.redhat.com/en/topics/api/what-is-a-rest-api.

[14] Wikipedia, "State Transfer," [Online]. Available: https://en.wikipedia.org/wiki/Representational_state_transfer.

[15] JavaPoint, "JavaPoint," [Online]. Available: https://www.javatpoint.com/software-engineering-coupling-and-cohesion.

[16] E. Doherty, "Educative," [Online]. Available: https://www.educative.io/blog/object-oriented-programming.

[17] GitHub, [Online]. Available: https://git-scm.com/book/en/v2/Getting-Started-About-Version-Control.

[18] Wikipedia, "Version Control," [Online]. Available: https://en.wikipedia.org/wiki/Version_control.

[19] Miro Medium, "https://miro.medium.com/," [Online]. Available: https://miro.medium.com/max/2625/1*f-66KJ3QEhv9EUAHoweMVA.png.

[20] Codecademy, [Online]. Available: https://www.codecademy.com/articles/what-is-an-ide.

[21] E. Thue, Writer, *Syntax Highlight side to side.* [Performance].

[22] Wikipedia, "Java Program Language," [Online]. Available:

https://en.wikipedia.org/wiki/Java_(programming_language).

[23] M. Tyson, "Infoworld," [Online]. Available: https://www.infoworld.com/article/3272244/what-is-the-jvm-introducing-the-java-virtual-machine.html.

[24] Wikipedia, "SQL," [Online]. Available: https://en.wikipedia.org/wiki/SQL.

[25] Khanacademy, [Online]. Available: https://www.khanacademy.org/computing/computer-programming/sql#sql-basics.

[26] The Apache Software Foundation, [Online]. Available: https://maven.apache.org/.

[27] Yarn, [Online]. Available: https://yarnpkg.com/.

[28] LF Charities Inc, "CloudBees," [Online]. Available: https://www.cloudbees.com/continuous-delivery/continuous-integration.

[29] PagerDuty, [Online]. Available: https://www.pagerduty.com/wp-content/uploads/2020/01/continuous-integration-2.png.

[30] C. TOZZI, "DevOps.com," [Online]. Available: https://devops.com/continuous-integration-vs-continuous-delivery-theres-important-difference/.

[31] S. J. Vaughan-Nichols, "Zdnet," [Online]. Available: https://www.zdnet.com/article/what-is-docker-and-why-is-it-so-darn-popular/.

[32] Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Docker_(software).

[33] Nuage Networks, [Online]. Available: https://www.nuagenetworks.net/wp-content/uploads/2020/06/slide1.jpg.

[34] Docker, "Blind Mounts," [Online]. Available: https://docs.docker.com/storage/bind-mounts/.

[35] Docker, "Volume Mount," [Online]. Available: https://docs.docker.com/storage/volumes/.

[36] Docker, "Docker Registry," [Online]. Available: https://docs.docker.com/registry.

[37] CPrime, [Online]. Available: https://www.cprime.com/resources/what-is-agile-what-is-scrum.

[38] Wikipedia, "Agile Software Developement," [Online]. Available: https://en.wikipedia.org/wiki/Agile_software_development.

[39] A. Pawlicka, 23 Feb 2021. [Online]. Available: https://selleo.com/blog/agile-software-development-process-everything-you-need-to-know.

[40] Wikipedia, "Open Source License," [Online]. Available: https://en.wikipedia.org/wiki/Open-source_license.

[41] Wikipedia, "Open Source," [Online]. Available: https://en.wikipedia.org/wiki/Open_source.

[42] Wikipedia, "Open-Source Software," [Online]. Available: https://en.wikipedia.org/wiki/Open-source_software.

[43] RedHat, "What is Open Source," [Online]. Available: https://www.redhat.com/en/topics/open-source/what-is-open-source.

[44] Wikipedia, "GitHub," [Online]. Available: https://en.wikipedia.org/wiki/GitHub.

[45] MySQL , "MySQL Homepage," [Online]. Available: https://www.mysql.com/.

[46] H. D. P., "What Is MySQL," [Online]. Available: https://www.hostinger.com/tutorials/what-is-mysql.

[47] Wikipedia, "MySQL," [Online]. Available: https://en.wikipedia.org/wiki/MySQL.

[48] JetBrains, "Discover IntelliJ," [Online]. Available: https://www.jetbrains.com/help/idea/discover-intellij-idea.html.

[49] Wikipedia, "IoC," [Online]. Available: https://en.wikipedia.org/wiki/Inversion_of_control.

[50] Spring, "Tutorial," [Online]. Available: https://spring.io/guides/tutorials/rest/.

[51] Guru00, "Postman Tutorial," [Online]. Available: https://www.guru99.com/postman-tutorial.html.

[52] Postman, Inc., [Online]. Available: https://documenter.getpostman.com/view/631643/JsLs/?version=latest.

[53] Wikipedia, "Single Page Application," [Online]. Available: https://en.wikipedia.org/wiki/Single-page_application.

[54] Devopedia, "Single Page Application," [Online]. Available: https://devopedia.org/images/article/222/9278.1593947515.jpg.

[55] Facebook Inc., "Components and Props," [Online]. Available: https://reactjs.org/docs/components-and-props.html.

[56] Facebook Inc., "Hooks," [Online]. Available: https://reactjs.org/docs/hooks-intro.html.

[57] GitHub, "Mastering Issues," [Online]. Available: https://guides.github.com/features/issues/.