Bacheloroppgave

NTNU
Norges teknisk-naturvitenskapelige universitet
Fakultet for informasjonsteknologi og elektroteknikk
Institutt for datateknologi og informatikk

Christian Ryddheim Dahlin

Adblock for podcasts

Bacheloroppgave i Bachelor Ingeniørfag, Data
Veileder: Donn Morrison
Mai 2024

NTNU
Kunnskap for en bedre verden

Christian Ryddheim Dahlin

Adblock for podcasts

Bacheloroppgave i Bachelor Ingeniørfag, Data
Veileder: Donn Morrison
Mai 2024

Norges teknisk-naturvitenskapelige universitet
Fakultet for informasjonsteknologi og elektroteknikk
Institutt for datateknologi og informatikk

NTNU
Kunnskap for en bedre verden

Abstract

As people spend more time on digital services, advertisement in digital media becomes more lucrative for advertisers [4]. Advertisement can take away from the consumer experience by adding unwanted breaks and distractions, or by consisting of inappropriate content. This has lead to the development of adblockers.

The predominant form of adblocking is through adblockers for Web browsers [8]. Another type of adblocking include targeting sponsored segments in video via crowdsourcing [56], and using machine learning to detect advertisements in audio shows promising results [78] [38] [9]. However, Adblock Radio develop by Storelli [67] still stands out almost 10 years later as possibly the only complete solution to an audio adblocker by not only detecting, but also removing advertisement in live radio. Storelli accomplished this via a rather complex combination of machine learning, fingerprinting of audio segments and crowdsourcing [65] [66].

With advertisement technology advancing through the development of technologies such as programmatic advertisement [7] and dynamic advertisement insertion (DAI) [57] [70], adblockers have the possibility of playing a large part in the future of enjoyment of consuming digital media. This project aims to investigate how an adblocker solution can be seamlessly integrated with the podcast listening experience, and research the state of the art of audio-based adblocking. In the end, a proxy-based adblocker targeting DAI by comparing two audio files fetched with different IP-addresses was implemented and integrated with an existing open-source podcast application.

Sammendrag

Ettersom mennesker bruker mer tid på digitale tjenester blir reklame i digitale media mer ettertraktet for annonsører [4]. Annonser kan redusere brukeropplevelsen ved å være uønskede pauser eller distraksjoner, eller ved å bestå av upassende innhold. Dette har ført til utviklingen av adblockers.

Adblocking blir først og fremst brukt gjennom nettlesere [8]. Andre former for adblocking er gjennom å fjerne sponsor deler av video ved bruk av nettdugnad [56], og bruk av maskinlæring til å detektere annonser i lyd viser lovende resultater [78] [38] [9]. Allikevel er Adblock Radio utviklet av Storelli [67] fortsatt unik nesten 10 år senere ved å muligens være den eneste fulle implementasjonen av en lydbasert adblocker ved å både oppdage og fjerne reklame i direkte radio. Storelli oppnådde dette ved en relativt avansert kombinasjon av maskinlæring, generering av fingeravtrykk for ulike deler av lyd og nettdugnad [65] [66].

Med eksempler på utvikling i reklameteknologi som programmatisk annonsering [7] og dynamic advertisement insertion (DAI) [57] [70], har adblockere muligheten til å spille en stor rolle i fremtiden av brukerglede ved bruk av digitale tjenester. Dette prosjektet søker å undersøke hvordan en adblocker kan sømløst integreres med podcast-lytting, og undersøke det fremste innenfor lydbasert adblocking. I løpet av prosjektet ble en proxy-basert adblocker rettet mot DAI ved å sammenligne to filer hentet med ulike IP-adresser implementert og integrert med en eksisterende open-source podcast tjeneste.

Preface

This project was chosen because I believe adblockers play a large part in the enjoyment of consuming digital media, and will continue to do so in the future. I was interested in learning more about how adblocking is done, the challenges associated with it, as well as getting an overview of the state of the art.

Much work within audio-based adblocking and advertisement detection is done through machine learning. I was interested in finding out if a more naive, direct approach to the problem with less complexity could be implemented. I ended up looking at using a Tor to get audio-files of the same episode from different countries, thereby getting different dynamically inserted ads, and comparing these audio-files to strip the different parts. This was combined with a HTTP(S) proxy to manage requests between the app and podcast-servers.

I would like to thank my supervisor Donn Morrsion, without whose help, advise and kindness this project could not have been completed.

Christian Ryddheim Dahlin
30.05.24, Trondheim

Assignment Details

The original assignment details are available in Appendix A.

The chosen solution was without any form of user interface beyond output in the terminal, and because of shortage of time user surveys were not performed. Therefore, the sentence underscored in the assignment details can be ignored.

For further details, see the vision document in Appendix C and the requirements documentation in Appendix D.

Contents

Figures

Tables

1. Introduction

As people spend more time on digital services, advertisement in digital media becomes more lucrative for advertisers [4]. Advertisements can take away from the consumer experience by adding unwanted breaks and distractions, or by consisting of inappropriate content. This has lead to the development of adblockers.

The predominant form of adblocking is through adblockers for Web browsers, having millions of users [8]. These adblockers target advertisements on websites through filter lists. Another type of adblocking include targeting sponsored segments in video via crowdsourcing [56], and using machine learning to detect advertisements in audio shows promising results [78] [38] [9]. However, Adblock Radio develop by Storelli [67] still stands out as possibly the only complete solution to an audio adblocker by not only detecting, but also removing advertisement in live radio. Storelli accomplished this via a rather complex combination of machine learning, fingerprinting of audio segments and crowdsourcing [65] [66].

As spending on digital advertisement continues to increase [20] [4], advertisement technology has gotten more advanced with the development of technologies such as programmatic advertisement, i.e. buying and selling advertisements in real time [7], and dynamic advertisement insertion (DAI) making it possible to provide updated, relevant advertisements based on parameters such as IP geolocation [57] [70]. An estimate for the U.S. from 2023 shows podcasts being one of the fastest growing digital channels, and podcast advertisement revenue growing more than twice as fast as total internet advertisement revenue in 2022. In the same year, DAI was estimated to represent 92% of the generated advertisement revenue from podcasts [5].

Facing these advancements in advertisement technology, adblockers have the possibility of playing a large part in the future of enjoyment of consuming digital media. This project aims to investigate how an adblocker solution can be seamlessly integrated with the podcast listening experience, and research the state of the art of audio-based adblocking. In the end, a proxy-based adblocker targeting DAI by comparing two audio files fetched with different IP-addresses was implemented and integrated with the podcast application AntennaPod [10].

1.1 Research questions

- What is the state-of-the-art of audio-based adblocking?
- What adblocking architectures can be seamlessly integrated into the podcast listening experience?
- How can audio adblocking adapt to the advancements in advertisement technology?

1.2 Thesis structure

Chapter 1: Introduction
Introduces the background for the project and the research questions the thesis aims to answer.

Chapter 2: Theory and Related Work
Theoretical background for the project, including related work. Describes the technical elements required for understanding the project and the results from it.

Chapter 3: Method
Details the methods used in research, development and execution of the project.

Chapter 4: Results
Presents the results achieved from the project.

Chapter 5: Discussion
Discussion of the results and the process of the project.

Chapter 6: Conclusion and Further Work
Presents conclusions that can be drawn from the results and discussion, and what further work should focus on.

Societal Impact
Discusses the different societal impacts of the project, including the ethic, environmental and economic aspects of the project.

1.3 Acronyms and abbreviations

- **DAI:** Dynamic ad insertion
- **VPN:** Virtual private network
- **ISP:** Internet service provider
- **Tor:** The onion router
- **HTTP:** Hypertext Transfer Protocol
- **HTTPS:** Hypertext Transfer Protocol Secure
- **TLS:** Transport Layer Security
- **CA:** Certificate authority
- **MITM:** Man in the middle

2. Theory and Related Work

2.1 Adblocking

Adblocking, particularly audio-based adblocking, is a small field of study. The agents in this field are often small groups or individuals making open source and/or free adblockers [75] [65] [23] [56].

2.1.1 Machine learning

A related, more developed field of study is audio classification. The work in this field primarily consists of training a machine learning algorithm to classify audio based on features extracted from the audio stream. Commonly, these features are based on comparing energy over time [61] [28] [48], but can also be speech [9] [17] or ”fingerprinting” of audio segments [16] [65]. With several approaches achieving accuracy in the 90% range [77], audio classification by machine learning is viable for use in advertisement detection [78] [38] [9] [65].

2.1.2 Advertisement filter lists

Adblockers for Web browsers with millions of users [52] [6] [30] are primarily dependent on filter lists for their functionality [51] [74] [29]. Filter lists are lists with rules and criteria that determine the behaviour of the adblocker, i.e. what content and sites to block [14]. When a website is loaded the given adblocker scans for content stemming from entries in its filter lists and blocks this content from appearing.

2.1.3 Crowdsourcing

Crowdsourcing is the method of distributing a workload on a decentralised group of people [27]. The contributors can be volunteers or paid for their work, but are typically not directly affiliated with the body organising the crowdsourcing. Filter lists used by adblockers are maintained and updated through crowdsourcing [8] by users submitting content they think should be blocked on a forum [21], and ”authors” of the filter list reviewing the submission by either adding it to the list or discarding it.

Similarly, an adblocker for Web-browsers targeting advertisements in videos on YouTube developed by Ramachandran [56] is primarily dependent on crowdsourcing by users submitting timestamps for sponsored segments. When starting a video, Sponsorblock checks the database for submitted timestamps for that video and if they exist, use them to skip the sponsored segment when it is reached. To eliminate poor or malicious submissions users have the option to vote on submissions used in the video they are watching [54].

2.2 Dynamic ad insertion

Dynamic ad insertion (DAI) is a technology that allows for greater flexibility and adaptability in advertising in podcasts and other digital media. Standard, so called ”static” or ”baked-in” advertisements are embedded in the audio file and cannot be changed unless the audio file is replaced [70]. This is disadvantageous as the selected advertisements will continue to be played throughout the episodes lifetime. Static advertisement does not allow for customisation of which advertisement to play for which user, but provides the same advertisements for everyone, and provides no overview of how the different advertisements perform. To keep episodes of a show updated with relevant advertisements using static advertisements would be unreasonably time consuming.

Using DAI, podcast creators mark insertion points in episodes where advertisements can be placed and the company hosting the podcast, or an advertisement company working with the hosting company, use the insertion points to place advertisements. The advertisements are selected and personalised based on parameters, e.g. the time of day, the user agent and the IP-address of the user. DAI ensures that listeners always get updated, relevant advertisements even when listening to old episodes, and companies providing DAI can give an overview of how the inserted advertisements are performing, i.e. the number of times a specific ad was delivered [57].

An estimate shows the share of total podcast advertisement revenue generated from DAI in the U.S. increased from 48% in 2019 to 92% in 2022 [5].

2.3 Proxy

The Hypertext Transfer Protocol (HTTP) is fundamental to communication on the Web. By default, communication between a client and a server is direct, but HTTP allows for the use of different intermediaries to form a chain of communication between a client and a server [22]. One such intermediary is a proxy.

A proxy is an agent that sits between a client and a server, and can do anything from blindly forwarding to modifying and filtering messages. Proxies can be used to increase security by hiding or masking IP-addresses, increase performance via caching, regulate accessibility by placing or bypassing firewalls, or otherwise get a desired behaviour by modifying communication. Such modification could also be used maliciously, e.g. accessing and modifying communication without the client and/or server knowing. With plain HTTP communication, a proxy is able to view all data passing between the client and the server.

```
          request   >
     UA ========================================= O
                                        <   response
```

Figure 2.1: Default direct HTTP communication. UA is the client and O is the server [22].

```
       >              >              >               >
  UA =========== A =========== B =========== C =========== O
              <              <              <               <
```

Figure 2.2: HTTP communication via proxies. A, B and C are proxies [22].

2.3.1 HTTPS

Hypertext Transfer Protocol Secure (HTTPS), or HTTP over TLS, is HTTP with encryption and the use of digital certificates, making the communication secure. Whilst HTTP communication is cleartext, making it vulnerable to eavesdropping and tampering, HTTPS encrypts data using Transport Layer Security (TLS) to ensure three properties [59] [60]:

- Authentication: The server is always authenticated, the client is optionally authenticated

- Confidentiality: Data sent over the channel is only visible to the endpoints, i.e. the client and the server
- Integrity: Data sent over the channel cannot be modified without detection

TLS handshake

HTTPS communication is established by the client and server performing a *TLS handshake*. The client initiates the handshake by sending a *Client Hello* message to the server, containing among other things information about which TLS version it supports and a list of which cryptographic cipher suites it prefers. Using the information provided by the client, the server selects the parameters for the communication and sends this back to the client in a *Server Hello*. The Server Hello also contains the server's digital certificate. By verifying the server's digital certificate, the client will trust the server and continue establishing the TLS communication. The client uses the public key of the server provided in the digital certificate to encrypt a random message which it sends to the server, and the client and the server use this random message to compute a secret key to be used in subsequent communication. Finally, the client and server each send a *Finished* message encrypted with the secret key to its counterpart to indicate that its part of the handshake is complete [60] [2].

HTTPS communication between the client and server is now established.

Digital certificates and certificate authorities

A digital certificate is a certificate issued by a trusted third-party known as a certificate authority (CA) that binds the ownership of a web server to a set of cryptographic keys and authenticates the server. When a server sends its digital certificate to a client, the client verifies it by checking that it is issued and signed by a trusted CA, that it has not expired or been revoked, and that the certificate is indeed issued to the web server it is attempting to connect to, and not some other server [26] [3].

2.3.2 HTTP CONNECT

TLS ensures that the client is communicating with a authenticated server and that no intermediary can view or modify the data without detection. This means proxying HTTPS communication cannot be done as easily as with plain HTTP. The *HTTP CONNECT* method is intended for use with proxies to get around this difficulty [22]. The client sends a request to the proxy with only the host and port number of the desired server prefixed with the CONNECT keyword:

```
CONNECT server.example.com:443 HTTP 1/1

Host: server.example.com
```

If the proxy responds successfully, all parties in the chain of communication - the client, the server, and any given number of proxies - switches to *tunnel* communication. A tunnel is an end-to-end virtual connection, in which the client and the server can perform a TLS handshake and establish HTTPS communication. The data will pass through the proxies, but the proxies can neither read or modify it: Its confidentiality and integrity, as well as the authentication of the server, is ensured by TLS.

2.3.3 MITM proxy

A man in the middle (MITM) attack is an attack where an intermediary intercepts communication between two parties without said parties knowing [19]. This attack can be utilised to create a proxy which is able to alter HTTPS communication, unlike a proxy using the HTTP CONNECT method.

Such a proxy is called a MITM proxy, and works by splitting the original connection into two separate ones: One between the client and the proxy, and one between the proxy and the server. When the client sends a CONNECT-request to the proxy to initiate the creation of a tunnel, the proxy responds successfully without actually contacting the server. The client initiates the TLS handshake believing it is communicating with the server through a tunnel, when it is actually communicating with the proxy. The proxy extracts enough information from the TLS handshake with the client to be able to start a separate TLS handshake with the server, which believes it is communicating with the client based on the information passed by the proxy.

At this point, digital certificates (see 2.3.1) create a line of defence, as the client expects a valid server-certificate from the proxy. To combat this the proxy generates certificates issued with its own CA implementation and uses information provided in the actual server's certificate received through the TLS handshake with the server to fill out the generated digital certificate and send it to the client.

Consequently, the prerequisite for this entire operation is that the client trusts the CA implementation of the proxy. The proxy can use details from the server's certificate to fill out its own, but as it is not a trusted CA the client will not trust it. To get around this, the CA certificate of the proxy has to be installed on the client as a trusted CA by some previous interfering.

With the proxy's CA implementation trusted by the client, the proxy uses information from one side of the handshake to finish the opposite one and establish HTTPS communication both ways, with an important adjustment: The proxy can view and modify all data [42].

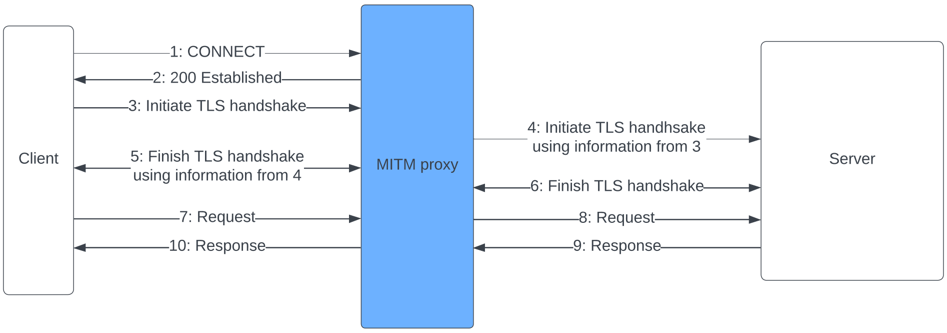

Client
1: CONNECT
2: 200 Established
3: Initiate TLS handshake
5: Finish TLS handshake using information from 4
7: Request
10: Response
MITM proxy
4: Initiate TLS handhsake using information from 3
6: Finish TLS handshake
8: Request
9: Response
Server

Figure 2.3: MITM proxy. Created from [42].

2.4 Tor network

The onion router (Tor) project is a project working for the right to privacy and freedom in technology [34]. The main part of the project is the Tor network, and a Web browser using the Tor network called the Tor Browser.

The Tor network is an overlay network consisting of thousands of servers hosted by volunteers, called *relays*, between which a Transmission Control Protocol (TCP) connection encrypted by TLS is established. The connection is called a *channel*, and through these channels messages can be transmitted. When connecting to the network, a client chooses a sequence of minimum three relays called a *path* and opens a channel to the first relay in it. Through this channel the client creates a cryptographic structure called a *circuit* in which each relay knows about its predecessor and successor, but no other relay in the circuit. The client negotiates a separate set of cryptographic keys with each relay, so that when the client exchanges a message with any relay on the circuit, each relay adds (or removes) one layer of encryption. When a circuit is established, communication is secure and anonymised [39].

To get a resource from the Web, the client tells the last relay of the circuit, called an *exit node*, to establish a TCP connection with a server. In doing so, the internet service provider (ISP) or anyone else watching the connection will not be able to track the internet activity. The website the exit node connects to, and all associated services and surveillance, will see the public IP-address of the exit node, and not of the client [33]. Because data will pass through several relays that might be located in different countries [69], and each channel implements encryption, using the Tor network is generally slower compared to other browsers and performance will vary [35] [68]. The Tor network also only consists of about 6000 relays with over 1 million daily users, which means performance will also depend on the strain on the network at a given time, e.g. how many users that are connected to it and what it is being used for, as large downloads and uploads will strain the network to a higher degree compared to loading a website.

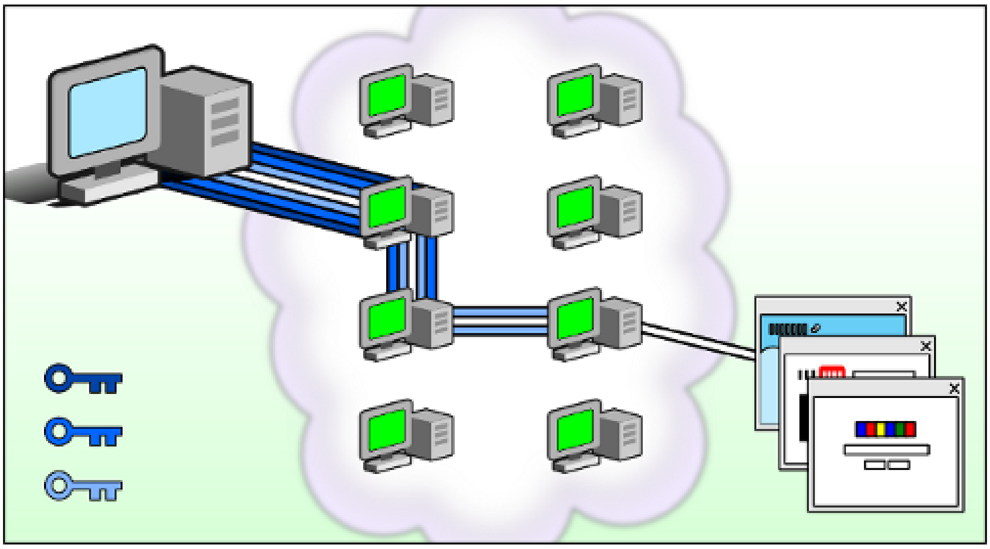


Figure 2.4: The Tor network [33].

2.5 Virtual private network

A virtual private network (VPN) creates a secure connection between a client and a network by establishing an encrypted connection to a remote server, and rerouting all data through it. The encryption protects against tracking of internet activity, and the IP-address of the client is masked by using the IP-address of the remote server to make connections, protecting against location tracking [44]. Because of the extra layer of encryption and that data is rerouted through a server which might be located far from the client and might handle a lot of data, using a VPN can decrease internet speed.

To use a VPN one has to choose a VPN provider which hosts the remote server the data is relayed through, which makes it possible for the VPN provider to gather data on the connections made to their servers. Additionally, many VPN providers require users to create a user profile on their services to access the servers, and many VPN services are blocked behind paywalls. A VPN connection resembles a connection made using the Tor network in that it reroutes traffic, provides encryption and masks the local IP-address of the client, but is significantly different in that it is generally proprietary and cannot grant anonymity as the VPN provider has access to the remote servers and (if required) ones user profile. Tor also provides security through encryption and privacy by hiding the local IP-address, but is additionally free, decentralised and anonymised.

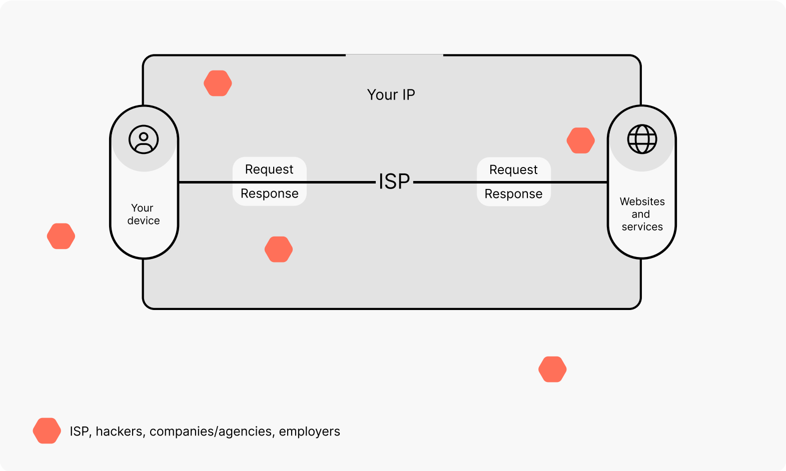


Figure 2.5: Internet communication without using a VPN [46].

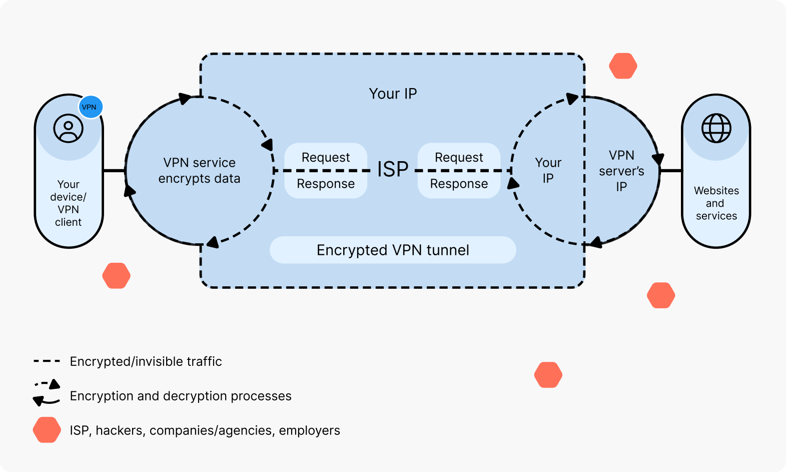

VPN
Your device/ VPN client
VPN service encrypts data
Your IP
Request
Response
ISP
Request
Response
Your IP
VPN server's IP
Websites and services
Encrypted VPN tunnel
Encrypted/invisible traffic
Encryption and decryption processes
ISP, hackers, companies/agencies, employers

Figure 2.6: Internet communication using a VPN. Created from [46].

3. Method

3.1 Choice of Development

3.1.1 Development plan

A Gantt chart was used for time management and visualisation of activities and milestones. The first iteration was made in the planning phase of the project and can be seen in Appendix B. The Gantt chart was continuously updated throughout the project, and the updated version can be seen in Appendix E.

3.1.2 Development method

The project was executed by one person, and no particular development method was fully implemented. Beginning with code development in April each week served as a sort of sprint, drawing inspiration from the development method Scrum [62]. At the end of each week, a review of the week was performed by writing a summary and evaluation of the work performed. This served as a Sprint Review and Sprint Retrospective in one, where I was able to review what went well and what could have been better, and plan for what to do in the following week in terms of both administration and development. After each review and retrospective, the Gantt chart was updated if needed. Although useful, it was less effective compared to my experience of doing it with a team, as there was no other input than my own. The weekly reviews can be seen in the time chart in Appendix E.

For code development, an issue board was used in addition to the Gantt chart to track code development tasks. The issue board can be seen in the projects repository, available in Appendix F.

3.2 Choice of Technology

3.2.1 Git

For storage of code and version control, Git [24] through GitHub [25] was used. This allowed for safe cloud-based storage and protection of a main branch, with development taking place on separate branches. One of the main advantages of GitHub is the practice of code reviews and pull requests. When development on a branch is completed a pull request is created, meaning the changes made on the branch have to be reviewed and approved before the branch is merged into the main branch, updating the actual codebase of the project. When done correctly, this practice ensures code quality and protects the development from errors.

As the project was executed by one person this practice was impossible to implement correctly. All code was reviewed by the same person that wrote it, taking away much of the intended effect of the practice. To mitigate this disadvantage it was attempted to have as much time as possible passing between a pull request and a review, to combat the ”blindness” occurring from looking at the same work for an extended period of time.

3.2.2 AntennaPod

The Assignment Details states that the selected approach was to have a focus on seamless integration with an existing open-source podcast application. This application was chosen to be AntennaPod as it is highly rated and widely used [71] [10], which makes it more rewarding to develop towards even though the developed product most likely will not be widely used. AntennaPod also supports setting a proxy within the application, which allows for easier and safer integration compared to an intercepting proxy unknown to the application.

3.2.3 Waydroid

AntennaPod is an application for Android devices with no web player available [11]. The application was accessed through Waydroid, an emulator for a Android device on GNU/Linux systems [76], as I did not have access to an Android device. Using Waydroid was further beneficial in that it meant having the whole system on one device.

3.2.4 mitmproxy

mitmproxy is set of tools that provides an intercepting proxy capable of accessing HTTPS communication by being a MITM proxy [43] [42]. In this project mitmproxy was used through its Python API and the command line version called *mitmdump*.

3.2.5 Tor

To fetch a second audio file with a different IP-address, a Tor SOCKS5 proxy is used. SOCKS5 is an internet protocol used by Tor to route data through the Tor network [37].

3.2.6 Python

The development was done in Python 3 because I have prior experience with it and because it has artificial intelligence libraries available [53] [72] [47]. Although artificial intelligence ended up not being used, it was considered as an approach in the beginning of the project, and when it became clear it would not constitute the main part of the approach, was considered being used in parts or as an addition to the advertisement removal algorithm.

3.3 Research method

With only one project member the time available for making changes during development is limited. In an attempt to combat this, a significant portion of the project time frame was devoted to research and design theorisation before starting development. The hope was that with thorough research and prioritising the iterative segment between literature review and design theorisation, one could make educated choices and avoid meeting dead ends in development, instead of relying on the development process being iterative. Drawing inspiration from Design Science Research [15], the research process in Figure 3.1 was implemented.

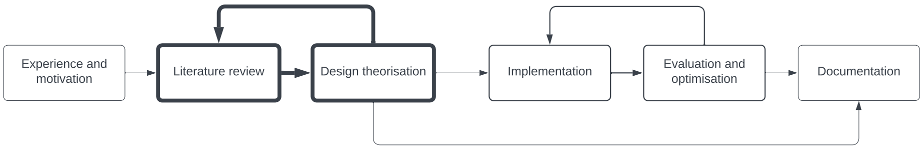


Figure 3.1: Research process flowchart. Created from [15] and [63].

3.3.1 Research process

Experience and motivation

Identification of the problem, i.e. the motivation for a solution. Through experience, discussion with the supervisor and initial research, the problem domain that formed the basis for the literature review and the research questions in Section 1.1 was established.

Literature review

Research of relevant work in the problem domain through academic works and other publications. Investigation of solutions that have been attempted to gain an understanding of successful and unsuccessful approaches.

See Section 2.1 for more details. The literature review also explored societal impacts of the problem domain and previous solutions, see Societal Impact for findings.

Design theorisation

Using information from the previous stages, the objectives of a solution was defined. Design theorisation began, and an iterative process between literature review and design theorisation was made: As a design was theorised, the domain of the literature review was narrowed down and more carefully researched, which further informed the design theorisation. See Section 3.4 for details.

Implementation

Design theorisation informed the vision document in Appendix C and the requirements documentation in Appendix D. From the theorised designs an approach was selected and developed. See Section 3.5 for details.

Evaluation and optimisation

At stages in implementation where a piece of functionality was developed to a point where it allowed for testing, evaluation and optimisation of said functionality was performed, and at the end of the stage usability tests were performed. See Section 3.6.

Documentation

The documentation stage began during design theorisation by creating the vision document in Appendix C and the requirements documentation in Appendix D, and was continuously worked on through the other appendices and finally this report.

3.4 Design theorisation

3.4.1 Removing advertisement

Machine learning

As described in Section 2.1, using machine learning for audio classification and advertisement detection shows promising results. Using it, either as a complete or partial solution to detect and remove advertisement, would be beneficial: There was previous work which might be incorporated into, or at least guide, my own solution. Training of the machine learning model would require a dataset, which would have to be created as no relevant dataset was found from the literature review. Furthermore, optimising a machine learning model requires considerable amounts of adjustment and testing, e.g. of features and the hyperparameters. Seeing as the aim of this project was not only advertisement removal but also integration with an existing podcast application, it was feared that with only one project member a machine learning approach might be too time consuming.

Moreover, as machine learning already was attempted and successful in detecting advertisements, it would be interesting to see if a more naive and lightweight approach could be applied.

Crowdsourcing

Section 2.1 shows crowdsourcing being used to target ads on websites and, more interestingly, target advertisements in video. Crowdsourcing on its base level is logically less complex compared to machine learning by gathering timestamps for advertisement and using the timestamps to skip during playing of the episode, but would in turn require a large scale database implementation. Ramachandran's Sponsorblock database contains almost 17 million submissions [55], and even though this project would receive few submissions, it would need to be able to extend to a large scale for crowdsourcing to be a viable solution.

A downside of crowdsourcing is that performance would be inconsistent. Popular podcasts would have many submission and skip all advertisement, but less popular ones might not have a single submission. Furthermore, if one listens to an episode a short time after its release it is likely one will have to listen to all advertisements regardless of the popularity of the podcast, as submissions have yet to be received.

Crowdsourcing would also have a negative impact on how seamless the integration with a podcast application would be. Sponsorblock is used for videos, a media where one normally is watching the content at all time. With podcasts, a common use case is having an episode playing in the background while performing other tasks. Making a submission would require a user to go to the podcast application, interfering with the listening experience and what the user might be doing at that time.

Podcasts using DAI would make crowdsourcing challenging. Ramachandran's Sponsorblock is targeted at sponsor segments in videos that are equal for all viewers, and crowdsourced advertisement filter lists used by Web adblockers can simply check all content up against its filter lists independently of the client. Contrarily, DAI results in podcast episodes having different advertisements with different lengths. As DAI can vary content using a broad array of parameters including what day the episode is played on and what time of day it is within that day [12], logic for fetching the right timestamps to skip for each user would need to be rather advanced.

Comparing two audio files

Given that a podcast uses DAI and that the DAI implementation uses IP-addresses as one of the parameters for selecting which advertisements to insert, all advertisements added by DAI could be removed by comparing two audio files of the same episode fetched with two different IP-addresses and removing the different parts, as different advertisements would have been added to the podcast. With a simple algorithm for comparing the two audio files this approach could be made lightweight and fast, but podcasts not using DAI would not have any advertisements removed.

3.4.2 Integration with the podcast application

Forking the application

By forking the podcast application the adblocking solution could be added as a native part of the application, making the integration seamless. However, it is disadvantageous in that is would broaden the scope of the project from an adblocker add-on to the entire application. In long terms, this approach would mean being responsible for keeping the application up to date, e.g. as new features or changes was added to the actual application.

Proxy

A proxy implementation could operate in the background, forwarding communication as if it was not there unless it receives something purposeful to process. It would allow for an invisible integration and compartmentalisation: The proxy, and by extension the adblocker, would be a separate piece of logic apart from the podcast application. Consequently, it would not have the advantage of being a native part of the application and would have to be ran separately.

3.4.3 Fetching a second audio file

To fetch a second audio file with different advertisements, an IP-address different from the address of the local machine would have to be used. The considered options for achieving this was using a VPN or Tor.

VPN

A VPN is generally faster than using the Tor network. Every user would need a VPN-instance separate of the adblocker, and because there are many different VPN services which a user might have, it would be challenging to make the adblocker compatible with the different ones. It could mean users would have to configure their respective VPN service with the adblocker in the source code, which would reduce the ease of the set-up and be difficult for users without experience in programming.

Tor

Tor is freely available, requires a small amount of configuration to open a proxy, and the user would not need to change anything in the source code. However, using Tor can result in bad performance, as detailed in Section 2.4. This is far from ideal when a relatively large audio file is being fetched. It is also worth noting that there are more important reasons why Tor should not be used in this context. See Section **ADD LATER**.

3.5 Implementation

Given the advantages of DAI and its use growth in the U.S. mentioned in Section 2.2, it seems possible that DAI will be the predominant form of advertisement in podcasts moving forward. Following this assumption, and the potential simplicity of the approach of comparing two audio files, it was chosen to be implemented for advertisement removal. The adblocker would be proxy-based to get the advantage of compartmentalisation and fetching an audio file with a second IP-address was done using Tor to prioritise ease of set-up. Figure 3.2 shows the implemented design.

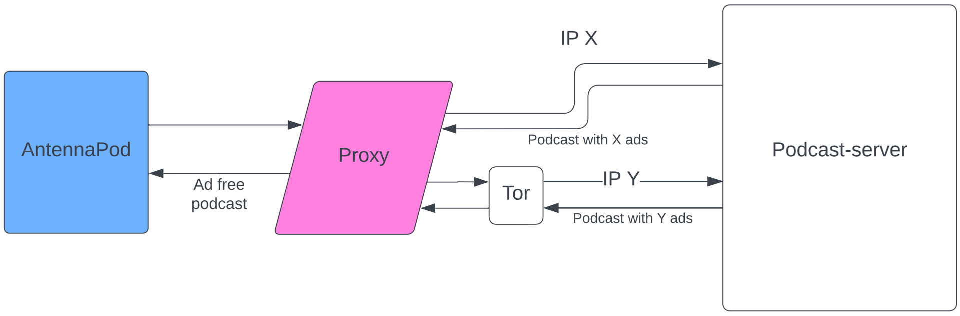


Figure 3.2: The architecture of the system.

3.5.1 Proxy

To ensure that the proxy would have optimised functionality for the task at hand, it was developed from the ground up. In its final iteration it was a streaming proxy using HTTP CONNECT for HTTPS communication, which I mistakenly believed the proxy would be able to intercept. To be able to modify HTTPS communication the proxy had to be extended to a MITM proxy, a time consuming and complex task. Because of time constraints, the developed proxy was discarded after discussion with the supervisor, and development of an extension to mitmproxy (see Section 3.2.4) began.

The proxy uses two dictionaries for data management and functionality. The *route* dictionary logs the complete path of a resource. The *origin URL*, i.e. the first URL being requested from the podcast application, is the key and all subsequent URLs leading to the episode are added as values belonging to that key.

```
Origin: http://open.live.bbc.co.uk/mediaselector/6/redir/(...)

Value: http://flex.acast.com/ak/mpg_mp3_vlow/modav/bUnknown-(...)

Value: http://stitcher2.acast.com/livestitches/75d9000b0517ca(...)
```

Figure 3.3: The route of a episode. The last (bottom) value is the URL that led to the episode.

The route dictionary serves two purposes, the first one being enabling the second request for the episode made by Tor to be made using the origin URL, increasing the probability of generating different advertisements by DAI. Podcasts vary in how many redirects through 302 responses happen, and if the second request is made with a later URL, the DAI might already have taken place and the same advertisements would be in both audio files.

In addition to this, the route dictionary enables a naive form of caching together with the *stripped* dictionary. The stripped dictionary holds processed episodes as values, i.e. episodes with advertisements removed, and origin URLs as keys. AntennaPod sends one request when downloading an episode, but multiple requests when streaming. The first request is handled, the advertisements are removed and the ad-free episode returned in the response, and the three other responses retrieve the episode from the cache (the stripped dictionary). The cache is further helpful in dealing with the unstable performance of Tor, as the advertisement removal in worst case scenarios makes playing the episode in AntennaPod time out. The proxy will continue the advertisement removal after the time out and once completed, the user can press play again and be served the episode without advertisement instantly from the cache.

```
[14:43:44.305] Response(200, audio/mpeg, 3.4m)
[14:43:44.305] Audio file recieved
[14:43:44.305] Starting ad stripping...
Progress: 216 / 216
[14:44:03.476] Ads stripped, sending response
192.168.240.112:56156: GET https://nyt.simplecastaudio.com/(...)
                    << 200 OK 3.4m
[14:44:03.948] Response(200, audio/mpeg, 3.4m)
[14:44:03.948] Audio file recieved
[14:44:03.948] Found stripped file, sending response
192.168.240.112:56146: GET https://nyt.simplecastaudio.com/(...)
                    << 200 OK 3.4m
```

Figure 3.4: Caching on a response. The first response is handled, advertisements are removed, and the response is sent. The second response fetches the processed episode from the cache, no processing needed.

The route and stripped dictionaries are also used in enabling skipping functionality. Although the entire episode is downloaded and served to AntennaPod whether streaming or downloading an episode, if a user skips backward or too far forward during streaming, AntennaPod issues a new request with a *Range* header. The Range header modifies the request to only request a certain portion or range of the resource [22]. If not for the cache, this would mean removing advertisements all over again for the same episode. Instead, the proxy returns the requested range of the episode without advertisements from the stripped dictionary.

The cache is checked on both requests and responses. When first starting an episode by streaming, all requests will get as far as requesting the episode from the podcast server before the episode without advertisements is added to the cache, i.e. all requests download the episode and only save time by not processing it. However, when skipping in the episode or resuming the episode at a later time, the cache will be checked on the request and no request to the podcast server is made.

```
[15:29:38.079] Found stripped file, sending response
[15:29:38.079] Found stripped file, sending response
192.168.240.112:53728: GET http://open.live.bbc.co.uk/medias(...)
                    << 200 OK 14.2m
192.168.240.112:46764: GET http://open.live.bbc.co.uk/medias(...)
                    << 200 OK 14.2m
[15:29:38.126] Found stripped file, sending response
192.168.240.112:46764: GET http://open.live.bbc.co.uk/medias(...)
                    << 200 OK 14.2m
```

Figure 3.5: Caching on a request. Here, AntennaPod issued three requests. No request to a podcast server is actually made as the episode is fetched from the cache.

As dictionaries are used and not persistent memory, the proxy limits entries in the route and stripped dictionary to five. When a sixth origin URL is requested the dictionaries are emptied, and the previously sixth URL is added as a first entry in the route entry. See Section 6.2.7 for discussion.

To optimise performance the proxy is to do as little work as possible. On receiving a request, it checks the cache and returns the episode, or a range of the episode if the request contains a *Range* header, if the origin URL of the request URL is in the stripped dictionary. If not, it forwards the request unmodified.

On responses, the proxy only interferes in three cases:

1. Response has status code 302

The HTTP status code 302 signifies that the requested resource (episode) is available at a different location, and provides that location [22]. Upon receiving a 302 response, the proxy updates the route of the resource in the route dictionary.

2. Response has status code 200 and audio content

Status code 200 indicates that the request has succeed [22], in this context meaning the episode has been found. The content type is checked via the *Content-Type* header, which indicates the media type of the resource in the response [22], and ensures that the proxy only operates on received episodes and not other 200 responses such as updating the podcast feed.

Given status code 200 and audio content, the proxy gets the origin URL belonging to the request from the route dictionary and checks if it is in the stripped dictionary. If it is, the processed episode is returned to the podcast application. If not, advertisement removal takes place (see Section 3.5.2). After the advertisements are removed, the processed episode is returned to AntennaPod and added to the stripped dictionary with its origin URL as key.

3. Response has status code 206 and audio content

Status code 206 indicates that the request has succeeded and a portion of the episode content is enclosed in the the response. Which portion is specified by the *Content-Range* header field in the response [22]. When the proxy receives a 206 response it means that associated request got past the cache check and was sent to a podcast server with a *Range* header. If the origin URL of the episode is in the stripped dictionary, the requested range is returned. If not, the requested range of the original file is served.

In practice it should not be possible for the origin URL not to be in the stripped dictionary if a 206 status code is returned unless the proxy is restarted during the streaming of an episode, as AntennaPod requests the entire episode leading to a 200 response from the podcast server and advertisement removal taking place, before issuing a request with a *Range* header.

3.5.2 Removing advertisement

To optimise performance, the algorithm initially was to stream both files concurrently and compare fixed size chunks. I was not able to implement this functionality, and as a fallback only the file fetched by Tor is streamed while the other one is fetched in its entirety. Coincidentally, this made it fitting for use after changing from using the self developed proxy to using mitmproxy, as mitmproxy heavily discourages modification of streams [41] [40].

Using the local IP-address, mitmproxy fetches the audio file in the 200 response that, given the origin URL is not in the stripped dictionary, initiates advertisement removal.

The algorithm

$session$: a Tor session enabling fetching via a Tor proxy

url: the origin URL

x: the stream of the episode fetched with Tor

$data$: the episode fetched with the local IP-address by mitmproxy

d: the processed episode without advertisement, initially empty

$prev\_idx$: the index of the previous $chunk$ in $data$, initialised to 0

$chunk$: the fixed-size chunk from Tor of size $delta$

idx: the index of $chunk$ in $data$, or -1 if non-existent

```python
 1  def strip_ads(data: bytes) -> bytes:
 2      with session.get(url, stream=True) as x:
 3          d = b""
 4          prev_idx = 0
 5          for chunk in x.iter_content(delta):
 6              idx = data.find(chunk)
 7              if idx != -1 and abs(idx-prev_idx) <= 500*delta:
 8                  d += chunk
 9                  data = data.replace(chunk, b"", 1)
10                  prev_idx = idx
11
12      return d
```

Using url the proxy opens a stream via the Tor network called x, processing fixed-sized chunks of size $delta$ at a time.

For every $chunk$, the algorithm finds the index idx of the chunk's occurrence in $data$.

If idx exists, and the absolute value of the difference between idx and $prev\_idx$ is below the threshold $500 \times delta$, the $chunk$ is added to the response. See Section 3.5.2 for details about the threshold.

The $chunk$, now confirmed to exist in $data$, is then removed from $data$ to shorten the search time for the next search. To minimise the time of this operation, the parameter 1 is added in the function call on line 9. This tells the function to only remove one occurrence of $chunk$ in $data$, making the function return after finding one occurrence instead of searching through all of $data$ to make sure all occurrences are removed. Although the same advertisement could be in multiple places in the same episode, meaning searching for all occurrences could be useful, this scenario is more effectively handled by the threshold.

Following the same reasoning of optimising the replace operation, the function is only called for a $chunk$ confirmed to exist in $data$. If it were called for all chunks, some would not be in $data$, meaning the function would waste time by searching through all of $data$ with no results. Lastly, $prev\_idx$ is updated to equal idx before processing of the next $chunk$ begins.

If idx does not exist, i.e. it is equal to -1, or the absolute value of the difference between idx and $prev\_idx$ is above the threshold, no operation takes place before processing of the next $chunk$ begins.

The threshold

Some advertisements are added to both episodes despite the podcast using geo-based DAI and the episode being fetched with two different IP-addresses. These advertisements are typically advertising a product related to the company publishing the podcast, e.g. another podcast hosted by the same company. This means that only checking if the received $chunk$ exists in $data$ is not enough.

The initial idea was to calculate the amount of bytes per second of audio, and then set a threshold to be a given amount of seconds. The amount of bytes per second of audio depends, among other things, on how the audio file is encoded and throughout the project all encountered episodes were provided as MP3 files. However, the literature review proved it difficult to find trustworthy standards for the MP3 file format.

Instead, the use of idx and $prev\_idx$ was implemented: By calculating the difference between the index of the previous $chunk$ in $data$ and the index of the current $chunk$, advertisements that were in both audio files but had different placement within the episode could be removed.

Assume the beginning of an advertisement that exists in both files is received through a $chunk$. Up until then the files have been identical, meaning $idx = prev\_idx = 0$ as chunks are deleted from $data$. The same advertisement is at the end of the locally fetched file, and makes $idx = 10000000$. This makes the threshold kick in, and the $chunk$ is not added to the response d and $prev\_idx$ is not updated. As all chunks with the advertisement finish, a chunk with normal content arrives, making $idx = 10000$. This is below the threshold, and adding to the response resumes, and $prev\_idx$ is updated.

3.6 Evaluation and optimisation

3.6.1 Self developed proxy

Buffering proxy for HTTP only

The first iteration was a threaded, buffering proxy for HTTP only receiving entire responses before sending it to a client. Each thread handled a a single request and response. This iteration had terrible performance, and the unacceptable lack in functionality in that it did not support HTTPS communication.

Buffering proxy with HTTP CONNECT

Support for HTTPS communication was implemented through HTTP CONNECT. Performance for HTTP communication remained unchanged, but HTTPS had good performance because of tunneling.

Streaming proxy with HTTP CONNECT

In its final iteration, the proxy was a threaded, streaming proxy for HTTP and supporting HTTPS communication through HTTP CONNECT. Performance was excellent for both HTTP and HTTPS communication, but the proxy was unreasonably computational expensive because of poor thread handling. When it was realised HTTP CONNECT would not allow for modification of HTTPS data, the proxy was discarded.

3.6.2 Extension to mitmproxy

With the self developed proxy discarded, time was limited to create a working implementation of an adblocker. The defining iterations of the mitmproxy add-on were:

MVP of adblocker

A functioning MVP for removing advertisements for both HTTP and HTTPS communication. Fetching of a second file through Tor was done with the last used URL of mitmproxy, and there was no functionality for skipping or resuming episodes without advertisement.

Implementation of caching on responses

The poor performance of Tor meant downloading or streaming an episode could time out. When a time out occurred, the proxy completed processing the episode and removed advertisements before discarding it, as the the client (AntennaPod) had disconnected. By attempting one more time, processing of the episode started all over.

To combat this the route and stripped dictionaries were implemented to allow for caching, but was only checked by responses. Skipping and resuming episodes without advertisement was not possible. With the route dictionary, requests through Tor began using the origin URL.

Implementation of the threshold

Logic for the threshold was implemented, making it a possibility that advertisements that are in both episodes are removed.

Updated Tor use

Realisation that Tor was not implemented correctly. mitmproxy provides an option for setting which IP-address mitmproxy uses for upstream connections, and this was set to be the IP-address of the exit node. However, the option was only usable with local IP-addresses and mitmproxy had been ignoring the option. To fix this, using the Python requests library [58] to fetch the second audio file through the Tor network was implemented. How different advertisements had been fetched up until this point is not understood.

Added skipping functionality and progress bar

Use of the route and stripped dictionaries were extended to allow for skipping and resuming of episodes without advertisements. Although a progress bar has no affect on performance, it improves the user experience by allowing the user to track the progress.

Implemented cache use on requests

The cache was checked on requests, improving performance and reducing traffic by reducing the amount of requests being sent to podcast servers.

3.6.3 Advertisement removal algorithm

After abandoning the idea of streaming both audio files at the same time, the advertisement removal algorithm was in its first iteration checking if the received $chunk$ from Tor was in the file fetched by mitmproxy. If it was it was added to the response, if not it was discarded.

Delta

The $delta$ value is used to determine the size of the $chunk$ being received from Tor and the size of the $chunk$ being searched for in $data$. One could use different values for receiving and searching by implementing logic where the program buffers data as it is received from Tor, and processes a different sized amount of data from this buffer.

However, by testing the advertisement removal algorithm without using Tor it was discovered that the values best suited for use with the algorithm also worked well with Tor. The algorithm was tested by providing $delta$ sized chunks in a list instead of through the Tor stream, and varying the values of $delta$. See Section 4.1.1 for results.

Consequently, the need for buffering logic was deemed unnecessary.

The threshold

Because of how the threshold is implemented, one has to make sure that no advertisement or continuous set of advertisements can last longer than the size of threshold. If that is to happen, updating the $prev\_idx$ will not take place from that point on and for the duration of the episode, meaning no more content will get added to the response.

Because trustworthy and consistent standards for the MP3 file format was not found during the literature review, the value for the threshold was tested and found manually by writing a threshold-sized chunk to file for different podcasts for different values of the threshold, and seeing how many seconds of audio it equalled.

3.6.4 Usability tests

As per the Assignment Details and the goals for the project outlined in Appendix B and Appendix C, usability tests of the end product was performed. The template for the test is available in Appendix G and the results from the tests are available in 4.1.4

Preparations

Initially, the test objectives were set:

- Uncover usability issues with adblock25
- Measure the ease/difficulty of set-up and installation
- Measure the ease/difficulty of use
- Measure the degree of integration with AntennaPod

Where *adblock25* is the developed adblock system.

All questions in the template were worded as neutral as possible in an attempt to not influence the participants opinion in any way.

As stated in the Pre-Project Plan the original aim was to have the participants have different backgrounds to get a realistic measurement of the ease of installation and set-up. This aim was changed when it was decided to develop towards AntennaPod on Waydroid, as Waydroid is exclusively available on GNU/Linux systems.

Consequently, the user profile was set to be: Users with some familiarity with GNU/Linux systems.

With tests aimed at doing research having many participants is advantageous, and often times required for the gathered data to have value or significance. However, with usability tests of a product this is not necessarily the case as increasing the amount of participants can have diminishing results, according to Nielsen and Landauer [45], who suggest rather doing multiple studies with as few as five participants. Although this claim has been challenged [64] [32], Nielsen and Landauer's assumption was followed in this project with five usability tests performed, but with only one study because of time constraints.

Method

The template guides the user through the entire test. For each test, an observer was present and took notes. The participants were told to attempt to solve everything themselves using the system documentation in Appendix F and/or the *README.md* in the projects repository available in the same appendix, and clearly indicate when they required help and intervention from the observer. This was done in an attempt to recreate a realistic installation and set-up scenario, and to be able to document when the participants required help.

4. Results

4.1 Engineering

Goals and requirements are available in the vision document in Appendix C and the requirements documentation in Appendix D.

4.1.1 Delta

Figure 4.1 and Table 4.1 show the advertisement removal algorithm presented in Section 3.5.2 tested with different values of $delta$. The algorithm is comparing 20 episodes of *BBC Word Service Global News Podcast*, 10 episodes of *Pod Save The World* and 10 episodes of *Pod Save the UK*. All three podcasts have multiple insertion points for DAI in episodes that are used in varying degrees for each episode. The values $delta = 2^{10}$ and $delta = 2^{11}$ had difficulties processing larger episodes and are consequently only used on the 20 episodes of *BBC Word Service Global News Podcast*.

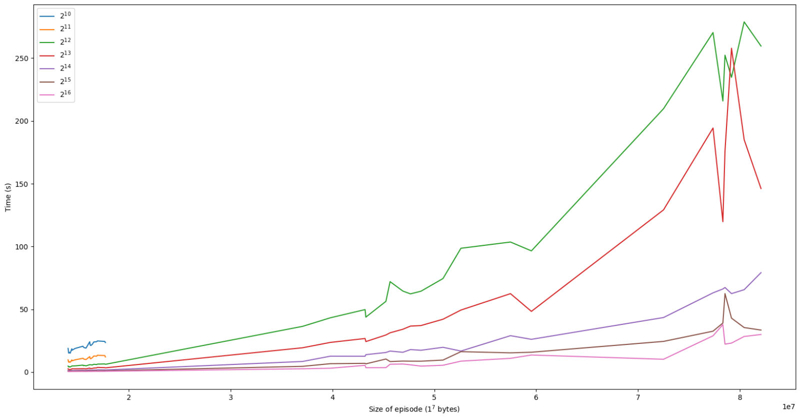


Figure 4.1: The advertisement removal algorithm for different values of $delta$.

File size	2^{10}	2^{11}	2^{12}	2^{13}	2^{14}	2^{15}	2^{16}
14039052	18.903	9.809	4.833	2.631	1.371	0.819	0.465
14099204	15.101	7.833	4.073	2.129	1.127	0.675	0.426
14132498	15.845	8.327	4.448	2.148	1.225	0.692	0.459
14245738	15.070	7.807	4.036	2.039	1.126	0.718	0.416
14405536	18.586	9.724	4.843	2.517	1.361	0.892	0.481
14422939	17.414	8.968	4.882	2.671	1.251	0.784	0.500
14726089	18.828	9.783	4.971	2.615	1.320	0.794	0.494
15493650	20.606	10.987	5.585	2.745	1.497	0.888	0.539
15538090	20.100	10.833	5.107	2.759	1.486	0.847	0.516
15601203	19.506	10.313	5.374	2.709	1.381	0.832	0.514
15801213	19.118	10.016	5.006	2.619	1.400	0.870	0.502
16163954	24.163	12.329	5.895	3.198	1.702	0.941	0.586
16208094	21.628	10.915	5.868	2.912	1.549	0.927	0.542
16268054	21.166	10.666	5.861	2.814	1.518	0.883	0.536
16417003	22.162	11.50	5.658	2.937	1.575	0.924	0.566
16561694	24.018	12.834	6.235	3.229	1.630	1.024	0.595
16839828	24.190	12.694	6.029	3.386	1.650	1.114	0.620
16915034	24.821	13.392	6.368	3.709	1.825	0.985	0.612
17638624	24.440	13.198	6.521	3.563	1.741	1.083	0.613
17707140	23.543	12.068	6.225	3.403	1.692	1.024	0.661
37059277	N/A	N/A	36.417	19.384	8.4827	4.589	2.656
39788624	N/A	N/A	43.261	23.655	12.674	6.714	3.078
43200350	N/A	N/A	49.780	26.777	12.639	6.982	5.415
43273716	N/A	N/A	43.792	24.312	13.886	6.575	3.506
45244590	N/A	N/A	56.337	29.5	15.621	10.429	3.563
45654608	N/A	N/A	72.039	31.299	16.755	8.410	6.272
46910038	N/A	N/A	64.526	34.128	15.782	8.803	6.452
47656215	N/A	N/A	62.303	36.641	17.937	8.726	5.808
48687405	N/A	N/A	64.413	37.075	17.399	8.700	4.7710
50849423	N/A	N/A	74.525	42.110	19.749	9.592	5.461
52610285	N/A	N/A	98.613	49.414	16.767	16.280	8.7401
57475338	N/A	N/A	103.58	62.475	29.021	15.342	10.971
59528627	N/A	N/A	96.544	48.359	26.116	15.819	13.618
72504098	N/A	N/A	209.76	129.20	43.472	24.465	10.238
77362399	N/A	N/A	270.40	194.36	63.108	32.583	28.912
78329411	N/A	N/A	215.90	119.79	66.200	39.077	37.838
78536722	N/A	N/A	252.36	175.95	67.364	62.530	22.276
79175980	N/A	N/A	234.72	257.91	62.508	42.982	23.116
80417737	N/A	N/A	278.89	185.09	65.615	35.475	28.371
82072509	N/A	N/A	259.64	146.17	79.215	33.474	29.971

Table 4.1: The advertisement removal algorithm for different values of $delta$ sorted by file size in bytes. Time is in seconds.

Figure 4.2 shows the adblocker being used with AntennaPod for the same episodes of *BBC Word Service Global News Podcast*, *Pod Save The World* and *Pod Save the UK* with values for $delta = 2^{13}$ and $delta = 2^{14}$.

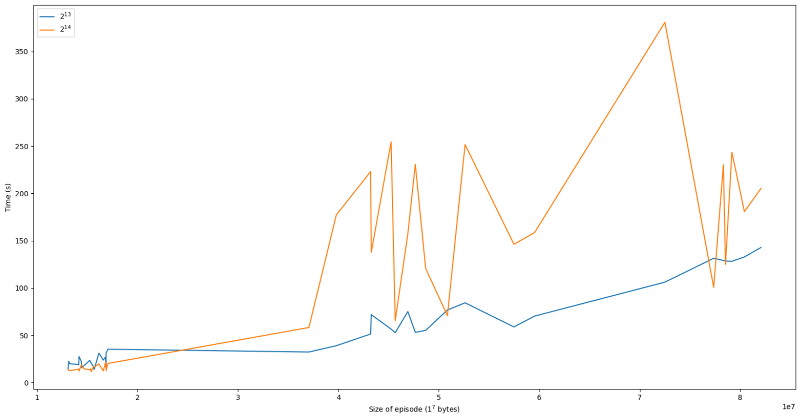


Figure 4.2: The advertisement removal used with Tor on AntennaPod for two different values of $delta$.

4.1.2 Functional demands

Removal of advertisement

The adblocker is guaranteed to remove all different parts of the two fetched episodes. If a podcast uses DAI with IP geolocation as a parameter for selecting advertisements, this results in up to all advertisement being removed. Occasionally, a podcast insert advertisements, e.g. advertisements for podcasts made by the same company, in both episodes. See Section 3.5.2 for details.

A podcast not using DAI, or a podcast using DAI but not with IP geolocation as a parameter for selecting advertisements, will not have any advertisements removed, except by chance, as sometimes simply fetching the advertisement through Tor will result in fewer advertisements being added to the episode.

Resuming episodes without advertisements

Resuming episodes without advertisements is implemented.

Undo processing of episode

Undoing the processing of an episode and getting the original file is **not** implemented.

Skipping in episodes without advertisements

Skipping forwards and backwards in episodes without advertisements is implemented.

Streaming and downloading

Both streaming and downloading of episodes without advertisement is implemented. By downloading AntennaPod will store the episode without advertisements in persistent memory.

4.1.3 Non-functional demands

The solution should focus on seamless integration with the podcast application and that set-up and maintenance for the end user is minimal

After set-up, the adblocker requires no maintenance from the user and AntennaPod can be used as if nothing has changed. The only change to the program flow of AntennaPod is increased load times, and AntennaPod can be used the same way as before the adblocker was added.

The integration with AntennaPod and the ease of installation and set-up were subjects of the usability tests, see Section 4.1.4 for results.

User tests and UI/UX guidelines should inform design choices

As explained in the Assignment Details, the chosen solution ended up being without any form of user interface beyond output in the terminal. User surveys were not performed due to shortage of time. In agreement with the supervisor, this demand was discarded.

Perform usability tests of the end product

Usability tests of the end product were performed. See Section 3.6.4 for method and Section 4.1.4 for results.

4.1.4 Usability tests

See Appendix G for the usability test template and Appendix H for a full overview of the responses. There were five participants of the usability tests.

Installation and set-up

Participants differed in 60% finding the installation and set-up *Easy* and 40% finding it *Difficult*.

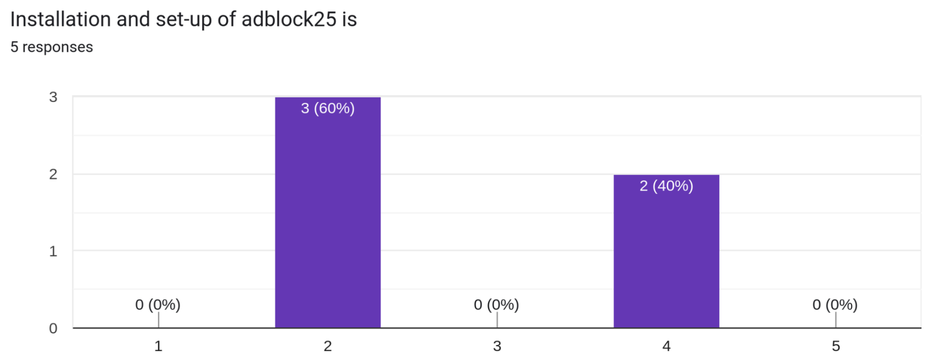


Figure 4.3: Results from installation and set-up. Here, 1 is *Very easy* and 5 is *Very difficult*.

60% found the installation of the CA certificate of mitmproxy on the Waydroid device challenging, and 40% found opening a Tor SOCKS proxy challenging. 20% found use of the terminal challenging, and 20% had no difficulty with installation and set-up.

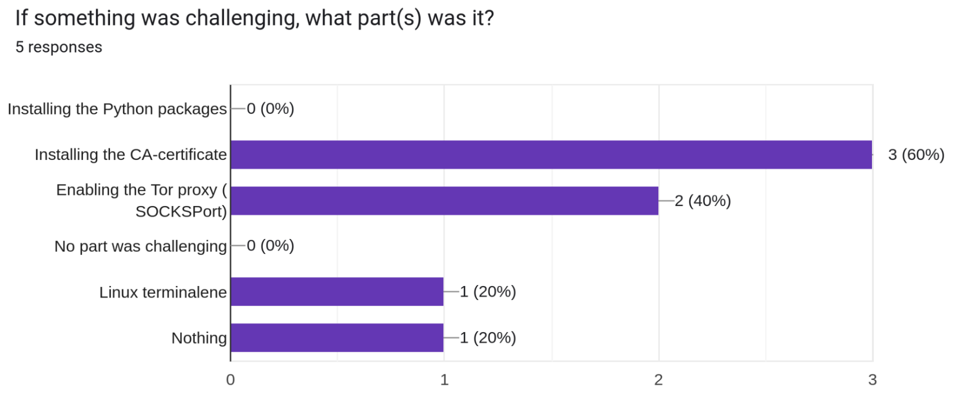


Figure 4.4: Results from what part(s) of installation and set-up were challenging.

Improvement suggestions from the participants were:

- Make all paths relative or absolute

- Make it more clear that the proxy can be shut down after generating the CA-certificate
- Make it more clear that the port of the adblock proxy, and not the Tor SOCKS proxy, is the one to be inputted in AntennaPod
- Make it clearer that the listed Tor options are in fact options, and not steps

Additionally, the following observations were made:

- The postboxes in the README.md had $\$$ in front to indicate that the commands were to be used in a terminal. However, this made the postboxes require modification by removing the $\$$ before use.
- Two participants mistakenly used the example value of the hash of the CA certificate before replacing it with the actual value.
- The README.md and system documentation did not specify to press *Test* and *OK* when setting the proxy in AntennaPod, which made some participants exit the setting of the proxy prematurely.
- One participant inputted the IP-address of the Waydroid device ending with 255 instead of the actual address

Running the adblocker

60% of participants found running the adblocker *Easy*, and 40% found it *Very easy*.

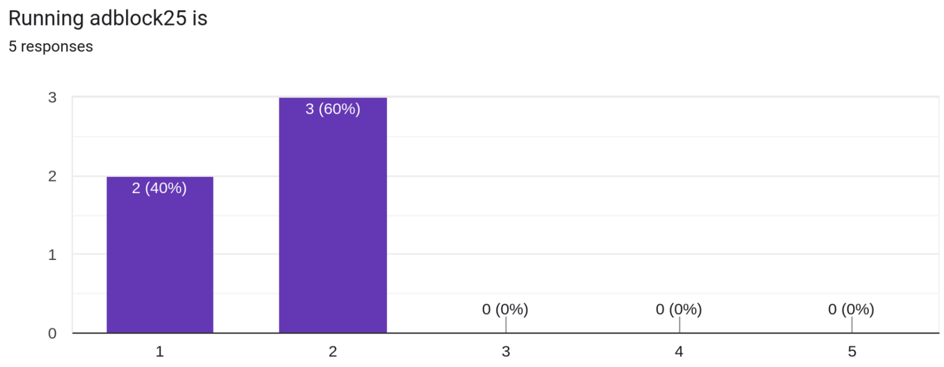


Figure 4.5: Results from running the adblocker. Here, 1 is *Very easy* and 5 is *Very difficult*.

Additionally, the following observations were made:

- 40% were confused by the *200 OK* response being displayed in the terminal before advertisement removal started, as they believed it indicated advertisement removal being complete.

All 100% of participants found the usage of the port option *Very easy*.

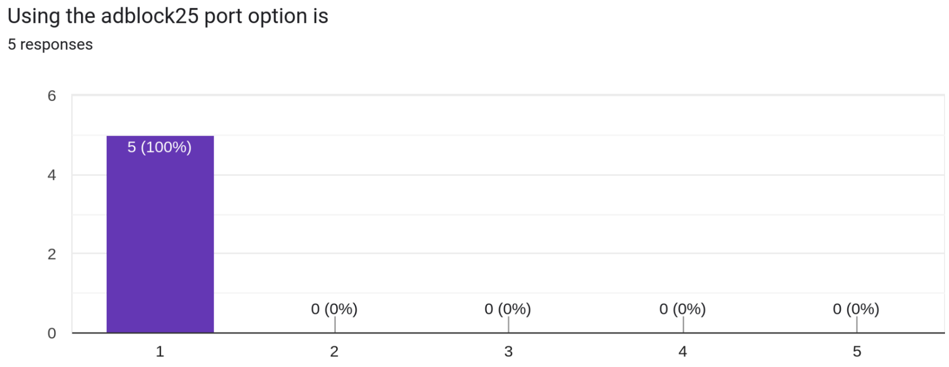


Figure 4.6: Results from using the port option. Here, 1 is *Very easy* and 5 is *Very difficult*.

All 100% of participants found the usage of the add-on option *Very easy*.

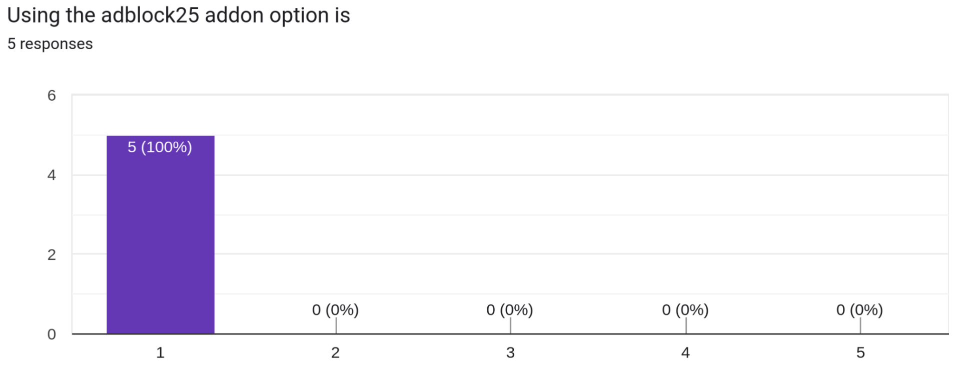


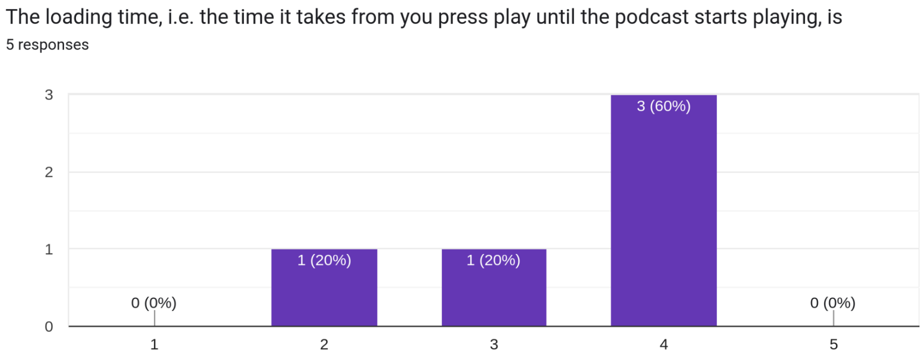
Figure 4.7: Results from using the add-on option. Here, 1 is *Very easy* and 5 is *Very difficult*.

Improvement suggestions from the participants were:

- Make the add-on option use a name for features instead of actual file names

User experience

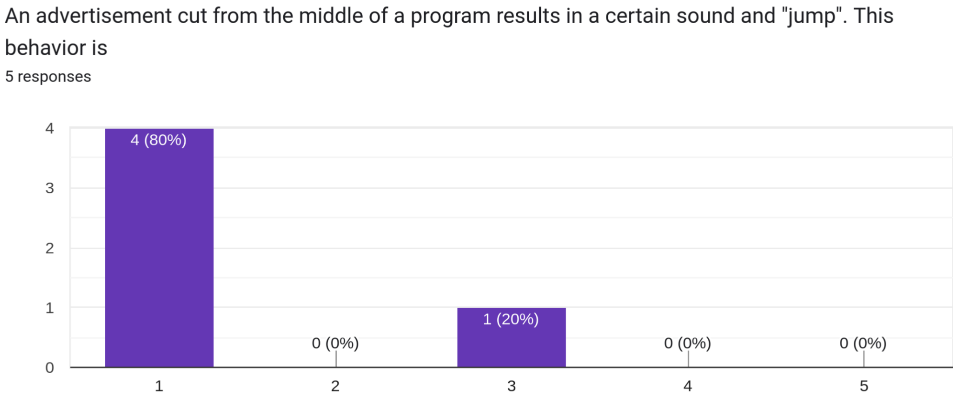
60% of participants found the loading time between pressing play and receiving the processed episode *Unacceptable*. 20% found it *Acceptable*, and 20% found it in between *Acceptable* and *Unacceptable*.


The loading time, i.e. the time it takes from you press play until the podcast starts playing, is

5 responses

Figure 4.8: Results from the loading time. Here, 1 is *Very acceptable* and 5 is *Very unacceptable*.

80% of participants found the ”cut” sound resulting from removing advertisements *Very acceptable*, and 20% found it in between *Acceptable* and *Unacceptable*.


An advertisement cut from the middle of a program results in a certain sound and "jump". This behavior is

5 responses

Figure 4.9: Results from the ”cut” sound. Here, 1 is *Very acceptable* and 5 is *Very unacceptable*.

40% of participants found the overall integration with AntennaPod *Very good*, another 40% found it *Good*, and the last 20% found it *Bad*.

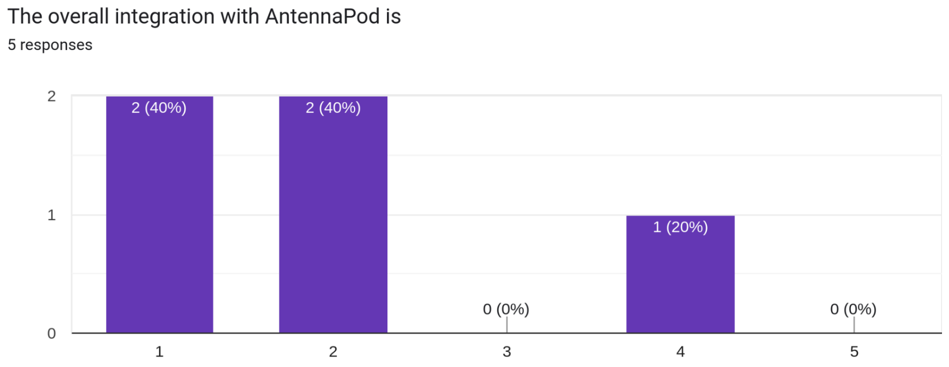


Figure 4.10: Results from integration with AntennaPod. Here, 1 is *Very good* and 5 is *Very bad*.

All 100% of participants felt that the loading time of the adblocker should be improved.

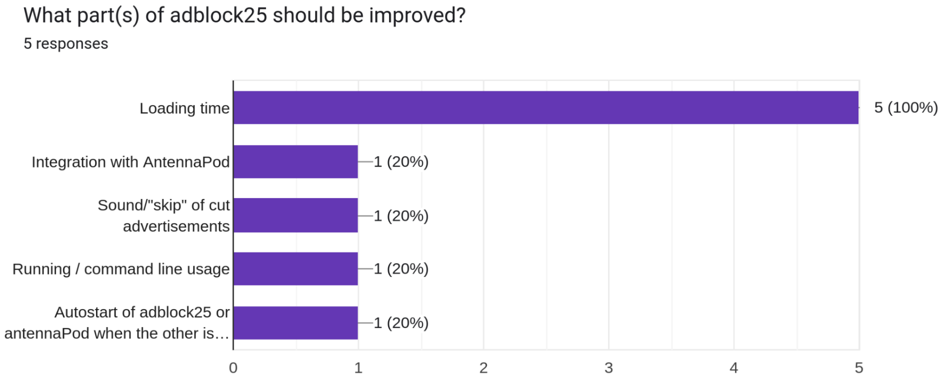


Figure 4.11: Results from what part(s) of the adblocker that should be improved.

Additional improvement suggestions from the participants were:

- Interactive set-up that guides and/or does part of the installation and set-up
- Add a way to see if the adblocker is running in AntennaPod
- Auto start of AntennaPod or the adblocker when the other part is started
- Auto start the adblocker on system start, i.e. when turning the computer on
- Pre-loading the next episode of a podcast
- Functionality to pre-process selected episodes

4.2 Administrative results

4.2.1 Time management

The Gantt chart was continuously used and updated throughout the project, with the initial version being made for the Pre-Project Plan. The updated version, available in Appendix E, gives an updated overview of the different activities and milestones and their respective hour count.

4.2.2 Development method

As described in Section 3.1.2, the Gantt chart was used for a review and retrospective of the week at the end of each week starting in April with code development, drawing inspiration from the development method Scrum. These are available in the time chart in Appendix B.

For code development an issue board was used to track tasks, and standard development practice of separate branches for each issue was implemented. The issue board and a complete overview of the branch structure is in the project repository available in Appendix F.

5. Discussion

5.1 Engineering

5.1.1 Sources of error

Testing of delta

Several values of $delta$ demonstrate great variation in performance around the 8×1^7 bytes mark in Figure 4.1 and Table 4.1. The tests were ran several times with similar results, but some test runs also had sudden increases in performance, i.e. one or several of the larger episodes requiring only about half the time of the data presented. All tests were ran with the same parameters and on identical files every time.

The cause of this is unclear. One theory is that the testing scenario was not ideal: Each episodes was loaded into a list consisting of $delta$-sized lists. The streaming from Tor was then replicated by processing one chunk (list) at a time from this list of lists. The deviance in the tests runs could stem from weird behaviour in Python caused by loading data structures of such large sizes into non-persistent memory.

The values presented are the ones closer to the average, as the increases in performance for larger episodes happened rarely.

Testing of the adblocker

As described in Section 2.4, performance of the Tor network is unstable. The values presented in Figure 4.2 should therefore serve first and foremost as data demonstrating that Tor makes performance of the adblocker inconsistent. The data should not be interpreted as representing average values for episodes in the size range of the test set of episodes. Neither can this data be used to say anything exact of which value of $delta$ that is ideal, as it cannot be known how large part of the times is caused by Tor.

5.1.2 Delta

Both 2^{13} and 2^{14} worked well in production, and setting $delta$ to be 2^{13} was implemented. Although 2^{14} increasingly outperformed 2^{13} as the size of the episode grew, it also resulted in slightly more content being removed from the episode. As per the goals for the product set in the Vision Document, the solution was to remove advertisement without further altering the listening experience. By setting $delta$ to be 2^{13} the user experience was prioritised.

This value had the additional advantage of working well with the Tor stream. As the performance of the Tor network is unstable, waiting for larger amounts of data can take a lot of time. By setting setting $delta$ to be 2^{13} both the advertisement removal algorithm and the stream could use the same value, removing the need for the additional buffer logic discussed in Section 3.6.3.

Although nothing exact can be said, during the project unstable performance with Tor using $delta = 2^{14}$ was experienced and this behaviour might be represented in Figure 4.2.

5.1.3 The threshold

The threshold used throughout the project was $100 \times delta = 100 \times 2^{13} = 819200$. As stated in 3.6.3, if this value is too low it will break functionality and return only parts of requested episodes. Towards the end of the project this value was more extensively tested, and in the highest quality episodes found throughout the project, this equalled about 40 seconds of sound. This was found to be too uncertain, and the value was tripled to be $500 \times delta = 500 \times 2^{13} = 4096000$, i.e. about 200 seconds.

Although the logic behind the threshold can work well, the value set for it should be backed by more testing and data. More work is required and should be done to find a safe value.

5.1.4 Functional demands

Removal of advertisement

The adblocker must not be thought of as as an all-purpose adblocker, but an adblocker for use with podcasts using geo-based DAI. For such podcasts, it is guaranteed to remove all advertisements except the special case described in Section 3.5.2.

As of delivery of the project, there is no easy way to toggle the adblocker off without having to unset the proxy in AntennaPod. The only way to achieve this is by shutting down the Tor instance, which results in no processing being done to episodes and the original file being served. If a user listens to a selection of different podcasts and these vary in degrees of use of geo-based DAI, this means either having to unset the proxy in AntennaPod, shutting down the Tor instance, or sitting through processing time with no effect for the podcasts not using geo-based DAI. Consequently, the adblocker is rather inflexible.

Performance

Table 4.1 shows the algorithm using between 2 and 3 seconds with $delta = 2^{13}$ on the top 20 episodes. i.e. the episodes of *BBC Word Service Global News Podcast*, which average in length of about $20-35$ minutes. The bottom 20 episodes have lengths between 40 minutes and 1.5 hour, with processing times between 20 seconds and about 4 minutes. Even using the outlier at 257.91 seconds and setting the episode length to be somewhere between $1 - 1.5$ hour, this is between 5% and 7% of episode time spent processing. For the *BBC* episodes, if we set an over-estimate for average processing time at 4 seconds and every episode having length of 30 minutes, this equals $\approx 0.2\%$ processing for episode time.

From these calculations, the algorithm's performance can be said to be decent. However, the adblocker as a whole performs much worse due to the use of the Tor network, making performance very fluctuating. Although performed with few participants, the result from the usability tests deeming the performance as *Unacceptable* (see Section 4.1.4) can be assumed to apply to a larger user base. In its current state, the adblocker has unacceptable performance issues in the inconsistency of the processing due to Tor.

Resuming episodes without advertisements

Resuming an episode without advertisements works well. Performance is optimised if the episode is still in the the cache, but if not it will be processed and starts playing where previously paused.

The only way for resuming an episode without advertisements not to work is by pausing an episode that is being streamed, restarting the adblocker, and then resuming the episode. This will result in the original file being served.

Undo processing of episode

This functionality was unfortunately not implemented.

The main reason behind this functionality being included in the Vision Document was in the case that the adblocker makes a mistake. With the delivered version of the adblocker, a mistake has not been encountered. If the adblocker for some reason (i.e. the Tor instance not being active) does not manage to process the episode, the original file is served. The value for $delta$ is also selected specifically to minimise the amount of content being cut away, although it still happens that seconds or fraction of seconds are cut away. Furthermore, the threshold value has been set excessively high compared to the experience throughout the project in an attempt to make sure the mistake specified in Section 3.6.3 does not occur.

This functionality had the lowest priority and was the only one not implemented, but other steps have been made to mitigate the lack of this functionality.

Skipping in episodes without advertisements

Skipping in episodes without advertisement works well. When skipping, the episode being played is almost guaranteed to be in the cache, providing good performance.

The only way for skipping in an episode without advertisements not to work is by restarting the adblocker while streaming an episode, and then resuming the episode. Or, if during streaming of an episode enough episodes are downloaded to empty the cache and route dictionary. These weaknesses can be attributed to the weakness of the cache implementation, see Section 6.2.7 for discussion.

Streaming and downloading

Both streaming and downloading of episodes without advertisement works. Downloading is superior, as restarting the proxy will have no effect on downloaded content. When downloading an episode, AntennaPod stores the processed episode in its own persistent memory, meaning it will be available at later times without the need for processing, and instantaneous performance in skipping and resuming.

The optimal way to use the adblocker is therefore by downloading episodes and if scarce on memory, deleting them when finished.

5.1.5 Non-functional demands

The solution should focus on seamless integration with the podcast application and that set-up and maintenance for the end user is minimal

Making the integration seamless has been a focus throughout the entire project. Notwithstanding the exceptions in functionality caused by restarting the proxy during streaming of an episode, the program flow of AntennaPod is unchanged except for loading times, and no maintenance is required after installation and set-up. Error handling is also prioritised, as the worst error that could happen is the original file being served. The results from the usability tests in Section 4.1.4 show that most participants find the integration as good.

Several participants found the set-up it difficult. Particularly the installation of the CA certificate was challenging. After the usability tests the set-up instructions in the README.md and System Documentation was improved with the findings from the tests, but the set-up is still rather challenging. This could have been improved by making a set-up script.

User tests and UI/UX guidelines should inform design choices

Discarded, see Section 4.1.3 and the Assignment Details for details.

5.1.6 Usability tests

The usability tests provided useful feedback that was used to improve the set-up documentation in the README.md and the System Documentation. The method implemented worked well by both receiving feedback in the template from the participants and making notes from observing.

Regarding the discussion in Section 3.6.4 the number of participants (5) proved useful for the testing of functionality and finding missing points of the set-up and installation guides. However, after improving the documents, a second iteration should have been performed to measure if the changes had any effect. Furthermore, with the participant count so low, not much significance can be attributed to the gathered data on more subjective questions, i.e. how acceptable the loading time is or how good the integration with AntennaPod is.

5.2 Administrative results

5.2.1 Time management

Time management in the beginning of the project was not well performed. As described in Section 3.3 the intention was to do thorough research to avoid meeting dead-ends in development, but this did not give the results that was hoped for. Too much time was spent on research without development, as initial development should have helped shape the research. Additionally, too much time passed before selecting an approach, and even with all the research a dead end was met in use of the HTTP CONNECT method by mistakenly believing it would be able to intercept HTTPS communication. This mistake resulted in the developed proxy and two weeks worth of time being discarded. However, the knowledge gained in developing the first proxy from the ground up was crucial in the development of the final product, as time was short.

Additionally, as a consequence of personal circumstances progress in February and March was below the necessary level for keeping up with the outlined plan and deadline, and a deadline extension was requested and granted in the beginning of April. At the same time an approach was selected and development began, and from this point on time management was done well. Having each week serve as a informal sort of sprint where some tasks were given priority and having the goal of completing them during the week helped make the sessions effective, and the weekly reviews and retrospective were also useful.

5.2.2 Development method

Usage of the issue board and standard branch practice made development effective. As mentioned in Section 3.1.2 preserving code quality was challenging, especially as the later parts of development was affected by being short on time. When development of a functionality or feature finished it was important to get it reviewed and merged as soon as possible, but reviewing the code without letting time pass would barely serve any purpose as it was reviewed by its author. Apart from this, development worked well.

6. Conclusion and Future Work

6.1 Research questions

6.1.1 What is the state-of-the-art of audio adblocking?

Almost ten years later, Storelli's Adblock Radio utilising a combination of machine learning, fingerprinting of audio and crowdsourcing still stands out as possibly the only complete solution to an audio adblocker by not only detecting, but also removing advertisement in live radio [66]. Since then, advancements in machine learning have been made, e.g. through the introduction of OpenAI's Whisper model [47], an automatic speech recognition system with multilingual capabilities. In March of 2024, Álvarez et. al [9] used Whisper to create a machine learning model capable of detecting advertisements with an F1 macro score close to 90%. Other machine learning approaches focusing on energy over time features such as Mel-frequency cepstral coefficients (MFCC), but also incorporating use of features such as fingerprinting and and speech [78] [38], have comparatively good results. However, these last two models have difficulties with more challenging scenarios, e.g. like promotions for CDs and host-read advertisements, which Álvarez' model claim to be able to handle.

However, the crux of all machine learning approaches is the complexity compared to limitations. Zamanirad and Douterloigne's model [78] from 2022 does not manage to filter out host-read advertisements and advises fine tuning the model every couple of months for long term performance, and adding a layer of semantic logic to the model further increasing its complexity. The model of Álvarez et al. does not need neither a database for updates nor energy over time features, but stresses the importance and difficulty in training these models with the right (hyper)parameters and that post-processing techniques should be implemented to improve performance, adding a layer of complexity. Additionally, the processing is time-consuming, with the model of Álvarez et al. spending between 4 to 17 minutes to process 1 hour of audio.

Compared to these heavy implementations the approach presented in this thesis is lightweight and naive, but although having several limitations, works well for what it is designed for: DAI based on geolocation. These heavy machine learning algorithms are overkill and not needed for every scenario, meanwhile naive, direct approaches like the one implemented in this project cannot handle every scenario. To maximise effectiveness and resource use, the state of the art of adblocking would be implementing a flexible adblocker framework with several approaches available for use individually and combined, enabling the use of the right tool for the job every time.

6.1.2 What adblocking architectures can be seamlessly integrated into the podcast listening experience?

The proxy-based approach in this project could have any set of backend, e.g. machine learning or crowdsourcing, instead of the implemented geo-based logic and possibly achieve similar levels, or better, of integration. Of the approaches discussed in this thesis, the use of crowdsourcing seem to be the least ideal for seamlessly integrating with the podcast listening experience, as podcasts differ from other media forms like video by not necessarily being consumed while having the application open or ready. Additionally, as Section 3.4.1 details, the advancements in advertisement technology like DAI makes the prospect of crowdsourcing even more difficult.

Machine learning is the approach with the most theory and previous work behind it, and have shown promising results for several years. Having an effective machine learning model in the backed of the proxy would have worked well, with possible increases to load time. The challenges of using machine learning is all the work predating the finished model, i.e. how logically complex it is and the amount of testing and adjustment, in addition to some models requiring regular refreshment in training, or increased complexity in forms of added layers of processing, for optimal performance.

As discussed in Section 6.1.1, the most seamless and effective integration seems to be to have multiple tools depending on what advertisement the podcast uses. For international podcasts using geo-based DAI the approach from this project can be used, and if some podcasts do not use DAI or use DAI at sufficiently consistent points and lengths, a crowdsourcing approach can be applied. If none of these are effective, the heavy machine learning models can be used to remove all advertisements, even the host-read ones.

6.1.3 How can audio adblocking adapt to the advancements in advertisement technology?

With the rise of programmatic advertising and DAI, the challenges for audio adblockers increase. Big podcasts with a known international audience like *BBC Word Service Global News Podcast* seem likely to use geo based DAI as this is the optimal way to give audiences relevant advertisements. Other podcasts will stick to using DAI without using IP-addresses as a parameter, adding the need for additional logic. One grim but enabling possibility is that as podcast advertisement revenue continues to grow a selected few advertisement companies create a worldwide oligopoly for podcast advertisement, and that these services because of their reach uses geo-based DAI no matter what podcast one listens to. Even machine learning approaches are not safe, with threats such as adversarial machine learning [18] existing and having been used on adblockers previously [73].

As revenue in podcast advertising increases, advancements in advertisements technology to provide advertisements that are as relevant as possible, as fast as possible will continue concurrently. Consequently, the only way for adblockers to adapt to these advancements is by having a flexible framework able to utilise different and combined architectures.

6.2 Further work

6.2.1 Tor network

The highest priority for further work on this project is to move the implementation away from using the Tor network. Tor is an important technology in a increasingly digital and surveillanced world, and as the network have few relays compared to the amount of users causing poor and inconsistent performance [35] [68], the network should be reserved for more important use [36] like people bypassing censorship. Additionally, having the adblocker be widely used via Tor is impossible because of the performance issues with the network.

The use case in this project, i.e. solely getting another IP-address, is the ideal task for a VPN. The reason a VPN was not used in this project was because of time constraints and to make the installation and set-up of the adblocker as easy and minimal as possible. Using a VPN would be beneficial in several ways, as it would decrease strain on the Tor network and increase the performance of the adblocker.

6.2.2 Implement full adblocker

A natural next step would be to implement a full adblocker and not just one for for geo-based DAI by extending and integrating it with other architectures, as discussed in Section 6.1.

6.2.3 Threading

As of delivery, the adblocker processes every episode that is streamed or downloaded. There is no way to cancel the processing except by shutting the adblocker down, and all other requests and responses received by the proxy are interrupted and have to wait until processing of the episode is finished. This should be improved by making the proxy threaded, with each thread handling a single request or connection.

6.2.4 Selecting shows

Following the same reasoning, logic for selecting which shows should be processed should be implemented. Having to wait for processing of episodes one knows does not use geo-based DAI is poor functionality.

6.2.5 Pre-processing of episodes

Functionality for pre-processing episodes should be implemented. This is in one way implemented by downloading an episode, resulting in AntennaPod storing the processed episode in persistent memory. It should be improved by allowing for users to choose, or automatically detect by seeing which podcasts are often played, shows that the adblocker automatically processes the next episode of automatically.

6.2.6 Add silence to cuts

As of delivery the proxy can make a sharp sound resulting from the removal of advertisement. To improve the user experience, some fractions of a second of silence should be added after removing the advertisement or before continuing with content.

6.2.7 The route and stripped dictionary

The use of the route and the stripped dictionary is not ideal. For one, the implemented logic of simply emptying the dictionaries on every sixth entry is not an actual solution, but implemented because of time constraints. Additionally, storing data structures as large as podcast episodes in non-persistent memory like dictionaries can cause difficulties and unwanted behaviour, e.g. by eating away at RAM usage. One approach would be to store the dictionaries and processed episodes in persistent memory, and then let users define for how long they want to keep them, or use a form of timeout based on last access time.

Societal Impact

Adblockers have the possibility of doing good by removing inappropriate content and reducing energy usage [50]. On a more subjective note, they can increase well-being by allowing users to avoid advertisement they find distracting or bothersome.

However, adblockers also have possibility of doing harm by disrupting the monetary system that is providing free availability to services through advertising. Podcasts are one such service and media. If one imagines use of the adblocker developed in this project or another product like it becoming wide spread, chances are advertisement companies and podcast creators will start fighting back, e.g. like Web browsers have attempted [13] [1] [31]. One possible outcome could be that more podcasts get locked behind paywalls, which is neither desirable nor fair towards users who do not use adblockers. Many podcasts already have the option of supporting them in other ways than advertisements, e.g. through Patreon [49], which in return grants benefits like private URLs leading to advertisement free episodes. This is a more sustainable option than adblockers, as it gives podcast creators the safety of income while at the same time giving users who want to avoid advertisement a meaningful way to do so.

Lastly, the ethics of the developed adblocker is worsened by unnecessary and egotistical use of the Tor network, which should be reserved for more important use.

Bibliography

[1] British Broadcasting Corporation (BBC). *Facebook's hidden battle against ad-blockers*. 2018. URL: https://www.bbc.com/news/technology-46508234 (visited on 30/05/2024).

[2] International Business Machines Corporation (IBN). *An overview of the SSL/TLS handshake*. 2024. URL: https://www.ibm.com/docs/en/ibm-mq/9.3?topic=tls-overview-ssltls-handshake (visited on 07/05/2024).

[3] International Business Machines Corporation (IBN). *Digital certificates*. 2024. URL: https://www.ibm.com/docs/en/integration-bus/10.1?topic=overview-digital-certificates (visited on 05/07/2024).

[4] PricewaterhouseCoopers LLP (PwC). *Internet Advertising Revenue Report*. 2023. URL: https://www.iab.com/wp-content/uploads/2023/04/IAB_PwC_Internet_Advertising_Revenue_Report_2022.pdf (visited on 25/05/2024).

[5] PricewaterhouseCoopers LLP (PwC). *U.S. Podcast Advertising Revenue Study 2023*. 2023. URL: https://www.iab.com/wp-content/uploads/2023/10/IAB_US_Podcast_Advertising_Revenue_Study_2023_Part_2.pdf (visited on 23/05/2024).

[6] AdAvoid. *AdBlocker Ultimate*. No date. URL: https://addons.mozilla.org/en-US/firefox/addon/adblocker-ultimate/ (visited on 25/05/2024).

[7] Amazon Ads. *Programmatic advertising*. 2024. URL: https://advertising.amazon.com/blog/programmatic-advertising (visited on 28/05/2024).

[8] Mshabab Alrizah, Sencun Zhu, Xinyu Xing and Gang Wang. 'Errors, Misunderstandings, and Attacks: Analyzing the Crowdsourcing Process of Ad-blocking Systems'. In: *Proceedings of the Internet Measurement Conference*. IMC '19. Amsterdam, Netherlands: Association for Computing Machinery, 2019, pp. 230–244. ISBN: 9781450369480. DOI: 10.1145/3355369.3355588. URL: https://doi.org/10.1145/3355369.3355588.

[9] Jorge Álvarez, Juan Carlos Armenteros, Camilo Torrón, Miguel Ortega-Martín, Alfonso Ardoiz, Óscar García, Ignacio Arranz, Íñigo Galdeano, Ignacio Garrido, Adrián Alonso, Fernando Bayón and Oleg Vorontsov. 'RADIA – Radio Advertisement Detection with Intelligent Analytics'. In: 2024. DOI: https://doi.org/10.48550/arXiv.2403.03538.

[10] AntennaPod. *About*. 2024. URL: https://antennapod.org/about/ (visited on 20/05/2024).

[11] AntennaPod. *Download*. 2024. URL: `https://antennapod.org/download/` (visited on 20/05/2024).

[12] Bryan Barletta. *How Dynamic Ad Insertion Actually Works*. 2021. URL: `https://soundsprofitable.com/article/how-dynamic-ad-insertion-actually-works/` (visited on 25/05/2024).

[13] Karl Bode. *Google Struggles to Justify Why It's Restricting Ad Blockers in Chrome*. 2019. URL: `https://www.vice.com/en/article/evy53j/google-struggles-to-justify-making-chrome-ad-blockers-worse` (visited on 30/05/2024).

[14] Brave. *What is a filter list?* 2023. URL: `https://brave.com/glossary/filter-list/` (visited on 25/05/2024).

[15] Jan vom Brocke, Alan Hevner and Alexander Maedche. 'Introduction to Design Science Research'. In: *Design Science Research. Cases*. Cham: Springer International Publishing, 2020, pp. 1–13. ISBN: 978-3-030-46781-4. DOI: `10.1007/978-3-030-46781-4_1`. URL: `https://doi.org/10.1007/978-3-030-46781-4_1`.

[16] Patrick Cardinal, Vishwa Gupta and Gilles Boulianne. *Content-based advertisement detection*. Ed. by Takao Kobayashi, Keikichi Hirose and Satoshi Nakamura. 2010. URL: `https://www.researchgate.net/publication/221489859_Content-based_advertisement_detection` (visited on 09/05/2024).

[17] Aotian Chen and Tianao Chen. 'Advertisement monitoring system based on C++'. In: *2021 IEEE International Conference on Advances in Electrical Engineering and Computer Applications (AEECA)*. 2021, pp. 742–747. DOI: `10.1109/AEECA52519.2021.9574388`.

[18] Matt Churilla, Nathan M. VanHoudnos and Robert W. Beveridge. *The Challenge of Adversarial Machine Learning*. 2023. URL: `https://insights.sei.cmu.edu/blog/the-challenge-of-adversarial-machine-learning/` (visited on 30/05/2024).

[19] European Union Agency for Cybersecurity (ENISA). *Man-in-the-Middle*. 2024. URL: `https://www.enisa.europa.eu/topics/incident-response/glossary/man-in-the-middle` (visited on 08/05/2024).

[20] Statista Reasearch Departement. *Advertising - Worldwide*. 2024. URL: `https://www.statista.com/outlook/amo/advertising/worldwide` (visited on 28/05/2024).

[21] EasyList. *Easy List Forum*. No date. URL: `https://forums.lanik.us/viewforum.php?f=62-report-unblocked-content` (visited on 28/05/2024).

[22] Roy T. Fielding, Mark Nottingham and Julian Reschke. *RFC 9110 - HTTP Semantics*. 2022. URL: `https://datatracker.ietf.org/doc/html/rfc9110` (visited on 06/05/2024).

[23] Uros Gazvoda. *uBlock Origin - Free, open-source ad content blocker*. No date. URL: `https://ublockorigin.com/` (visited on 09/05/2024).

[24] Git. *git*. No date. URL: https://git-scm.com/ (visited on 20/05/2024).

[25] GitHub. *About GitHub and Git*. 2024. URL: https://docs.github.com/en/get-started/start-your-journey/about-github-and-git (visited on 20/05/2024).

[26] GlobalSign. *Certificate Authorities & Trust Hierarchies*. 2024. URL: https://www.globalsign.com/en/ssl-information-center/what-are-certification-authorities-trust-hierarchies (visited on 07/05/2024).

[27] Samuel Greengard. *Crowdsourcing*. 2024. URL: https://www.britannica.com/money/crowdsourcing (visited on 10/05/2024).

[28] Shawn Hershey, Sourish Chaudhuri, Daniel P. W. Ellis, Jort F. Gemmeke, Aren Jansen, R. Channing Moore, Manoj Plakal, Devin Platt, Rif A. Saurous, Bryan Seybold, Malcolm Slaney, Ron J. Weiss and Kevin Wilson. *CNN Architectures for Large-Scale Audio Classification*. 2017. URL: https://doi.org/10.48550/arXiv.1609.09430 (visited on 09/05/2024).

[29] Raymond Hill. *uBlock Origin (uBO)*. 2023. URL: https://github.com/gorhill/uBlock/blob/b1530e26591cadea712e5dd1378a2377d9c1c9de/README.md (visited on 25/05/2024).

[30] Raymond Hill. *uBlock Origin*. No date. URL: https://addons.mozilla.org/en-US/firefox/addon/ublock-origin/ (visited on 25/05/2024).

[31] Andrew Hutchinson. *YouTube Steps up Its Fight Against Ad Blockers With Load Delays*. 2023. URL: https://www.socialmediatoday.com/news/youtube-steps-up-fight-against-ad-blockers-new-load-delays/700512/ (visited on 30/05/2024).

[32] Wonil Hwang and Gavriel Salvendy. 'Number of people required for usability evaluation: the 10±2 rule'. In: *Commun. ACM* 53.5 (May 2010), pp. 130–133. ISSN: 0001-0782. DOI: 10.1145/1735223.1735255. URL: https://doi.org/10.1145/1735223.1735255.

[33] The Tor Project Inc. *About Tor browser*. No date. URL: https://tb-manual.torproject.org/about/ (visited on 08/05/2024).

[34] The Tor Project Inc. *History*. No date. URL: https://www.torproject.org/about/history/ (visited on 08/05/2024).

[35] The Tor Project Inc. *How can I make Tor run faster? Is Tor Browser slower than other browsers?* No date. URL: https://support.torproject.org/tbb/tbb-22/ (visited on 22/05/2024).

[36] The Tor Project Inc. *Inception*. No date. URL: https://2019.www.torproject.org/about/torusers.html.en (visited on 30/05/2024).

[37] The Tor Project Inc. *SOCKS5*. No date. URL: https://support.torproject.org/glossary/socks5/ (visited on 24/05/2024).

[38] Shashidhar G. Koolagudi, Shriyak Sridhar, Narendran Elango, Karthik Kumar and Fathima Afroz. 'Advertisement detection in commercial radio channels'. In: *2015 IEEE 10th International Conference on In-*

dustrial and Information Systems (ICIIS). 2015, pp. 272–277. DOI: 10.1109/ICIINFS.2015.7399023.

[39] Nick Mathewson. *A short introduction to Tor*. 2023. URL: https://gitlab.torproject.org/tpo/core/torspec/-/blob/38fa996c1cd8691cbd57b8fab543e74b33501fe3 spec/intro/index.md (visited on 08/05/2024).

[40] mitmproxy. *Addon examples*. No date. URL: https://docs.mitmproxy.org/stable/addons-examples/#http-stream-modify (visited on 26/05/2024).

[41] mitmproxy. *Features*. No date. URL: https://docs.mitmproxy.org/stable/overview-features/#streaming (visited on 26/05/2024).

[42] mitmproxy. *How mitmproxy works*. No date. URL: https://docs.mitmproxy.org/stable/concepts-howmitmproxyworks/ (visited on 08/05/2024).

[43] mitmproxy. *Introduction*. No date. URL: https://docs.mitmproxy.org/stable/ (visited on 24/05/2024).

[44] Mozilla. *What is a VPN?* No date. URL: https://www.mozilla.org/en-US/products/vpn/resource-center/what-is-a-vpn/ (visited on 24/05/2024).

[45] Jakob Nielsen and Thomas K. Landauer. 'A mathematical model of the finding of usability problems'. In: *Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems*. CHI '93. Amsterdam, The Netherlands: Association for Computing Machinery, 1993, pp. 206–213. ISBN: 0897915755. DOI: 10.1145/169059.169166. URL: https://doi.org/10.1145/169059.169166.

[46] NordVPN. *What is a VPN?* 2024. URL: https://nordvpn.com/what-is-a-vpn/ (visited on 24/05/2024).

[47] OpenAI. *Introducing Whisper*. 2022. URL: https://openai.com/index/whisper/ (visited on 20/05/2024).

[48] Kamalesh Palanisamy, Dipika Singhania and Angela Yao. 'Rethinking CNN Models for Audio Classification'. In: *CoRR* abs/2007.11154 (2020). URL: https://arxiv.org/abs/2007.11154 (visited on 09/05/2024).

[49] Patreon. *Patreon for Podcasters*. No date. URL: https://www.patreon.com/creators/podcasts (visited on 30/05/2024).

[50] Joshua M. Pearce. 'Energy Conservation with Open Source Ad Blockers'. In: *Technologies* 8.2 (2020). ISSN: 2227-7080. URL: https://www.mdpi.com/2227-7080/8/2/18.

[51] Adblock Plus. *About Adblock Plus*. 2023. URL: https://adblockplus.org/en/about (visited on 25/05/2024).

[52] Adblock Plus. *Adblock Plus*. No date. URL: https://addons.mozilla.org/en-US/firefox/addon/adblock-plus/ (visited on 25/05/2024).

[53] PyTorch. *PyTorch Foundation*. No date. URL: https://pytorch.org/foundation (visited on 20/05/2024).

[54] Ajay Ramachandran. *How it works*. No date. URL: https://sponsor.ajay.app/about/ (visited on 28/05/2024).

[55] Ajay Ramachandran. *Overall Stats*. No date. URL: `https://sponsor.ajay.app/stats/` (visited on 21/05/2024).

[56] Ajay Ramachandran. *SponsorBlock*. No date. URL: `https://sponsor.ajay.app/` (visited on 25/05/2024).

[57] RedCircle. *What is Dynamic Insertion?* No date. URL: `https://support.redcircle.com/what-is-dynamic-insertion` (visited on 06/05/2024).

[58] Kenneth Reitz. *requests 2.32.2*. 2024. URL: `https://pypi.org/project/requests/` (visited on 26/05/2024).

[59] Eric Rescorla. *RFC 2818 - HTTP over TLS*. 2000. URL: `https://datatracker.ietf.org/doc/html/rfc2818` (visited on 07/05/2024).

[60] Eric Rescorla. *RFC 8446 - The Transport Layer Security (TLS) Protocol Version 1.3*. 2018. URL: `https://datatracker.ietf.org/doc/html/rfc8446` (visited on 06/05/2024).

[61] Feng Rong. 'Audio Classification Method Based on Machine Learning'. In: *2016 International Conference on Intelligent Transportation, Big Data & Smart City (ICITBS)*. 2016, pp. 81–84. DOI: `10.1109/ICITBS.2016.98`.

[62] Ken Schwaber and Jeff Sutherland. *The 2020 Scrum Guide*. 2020. URL: `https://scrumguides.org/scrum-guide.html` (visited on 10/05/2024).

[63] Nicolai Thorer Sivesind and Andreas Bentzen Winje. *Turning Poachers into Gamekeepers: Detecting Machine-Generated Text in Academia Using Large Language Models*. 2023. URL: `https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/3078096?locale-attribute=en` (visited on 21/05/2024).

[64] Jared Spool and Will Schroeder. *Testing web sites: five users is nowhere near enough*. 2001. DOI: `http://dx.doi.org/10.1145/634067.634236`. URL: `https://www.researchgate.net/publication/200553186_Testing_web_sites_five_users_is_nowhere_near_enough` (visited on 28/05/2024).

[65] Alexandre Storelli. *Designing an audio adblock*. 2018. URL: `https://www.adblockradio.com/blog/2018/11/15/designing-audio-ad-block-radio-podcast/` (visited on 09/05/2024).

[66] Alexandre Storelli. *Adblock Radio is a perceptual ad blocker*. 2019. URL: `https://www.adblockradio.com/blog/2019/10/25/adblock-radio-is-a-perceptual-ad-blocker/` (visited on 28/05/2024).

[67] Alexandre Storelli. *Let's improve our radio & podcasts experience*. 2020. URL: `https://www.adblockradio.com/en/` (visited on 28/05/2024).

[68] Tails. *Why is Tor slow?* No date. URL: `https://tails.net/doc/anonymous_internet/tor/slow/index.en.html` (visited on 22/05/2024).

[69] Hacker Target. *Tor Exit Nodes Located and Mapped*. Np date. URL: `https://hackertarget.com/tor-exit-node-visualization/` (visited on 22/05/2024).

[70] Acast Team. *The Benefits of Dynamic Ad Insertion in Podcast Advertising*. 2023. URL: https://advertise.acast.com/news-and-insights/the-benefits-of-dynamic-ad-insertion-in-podcast-advertising (visited on 06/05/2024).

[71] AntennaPod Open Source Team. *AntennaPod*. 2024. URL: https://play.google.com/store/apps/details?id=de.danoeh.antennapod&hl=en&gl=US (visited on 20/05/2024).

[72] TensorFlow. *An end-to-end platform for machine learning*. No date. URL: https://www.tensorflow.org/ (visited on 20/05/2024).

[73] Florian Tramèr, Pascal Dupré, Gili Rusak, Giancarlo Pellegrino and Dan Boneh. 'Ad-versarial: Defeating Perceptual Ad-Blocking'. In: *CoRR* abs/1811.03194 (2018). arXiv: 1811.03194. URL: http://arxiv.org/abs/1811.03194.

[74] AdBlocker Ultimate. *Why AdBlocker Ultimate?* No date. URL: https://support.adblockultimate.net/en/articles/3309097-why-adblocker-ultimate (visited on 25/05/2024).

[75] Adblocker Ultimate. *Adblocker Ultimate for Browsers*. No date. URL: https://adblockultimate.net/browsers (visited on 25/05/2024).

[76] Waydroid. *Waydroid*. 2022. URL: https://waydro.id/ (visited on 20/05/2024).

[77] Khalid Zaman, Melike Sah, Cem Direkoglu and Masashi Unoki. 'A Survey of Audio Classification Using Deep Learning'. In: *IEEE Access* 11 (2023). DOI: 10.1109/ACCESS.2023.3318015.

[78] Shayan Zamanirad and Koen Douterloigne. 'Say No2Ads: Automatic Advertisement and Music Filtering from Broadcast News Content'. In: *Service-Oriented Computing – ICSOC 2021 Workshops*. Ed. by Hakim Hacid, Monther Aldwairi, Mohamed Reda Bouadjenek, Marinella Petrocchi, Noura Faci Fatma Outay, Amin Beheshti, Lauritz Thamsen and Hai Dong. Cham: Springer International Publishing, 2022, pp. 18–31. ISBN: 978-3-031-14135-5. DOI: https://doi.org/10.1007/978-3-031-14135-5_2.

A. Assignment Details

The original assignment details in form of a PDF.

Arbeidstittel:	Adblock for podcasts

Problemstilling
How can an adblock solution for podcasts be seamlessly integrated into the podcast listening experience?

Beskrivelse av oppgaveforslag:
Develop a flexible adblocking framework for podcasts with a focus on seamless integration into an existing open soruce podcast application (e.g., Antennapod) or as an HTTP intercepting proxy. Make use of user surveys and UX design guidelines to inform design choices. Several implementation options are possible, from machine learning aided ad detection, to stripping of ads from audio files dynamically inserted via IP geolocation, to crowd-sourced ad annotation. The delivered thesis should offer a survey of state-of-the-art adblocking and audio-based adblocking in particular, a user-informed design and development life cycle, and usability tests of an end product.

Oppgaven passer for (kryss av de(t) som passer og skriv evt. en kommentar til oss):	- Bacheloroppgave
Hvilket studieprogram og emne passer oppgaven til? (spesifiseres ved bacheloroppgaver)	IDATT2900 – Bacheloroppgave Dataingeniør
Oppgaven passer best for, antall studenter:	- 1 - 2

Opplysninger om oppgavestiller

Er du fra en bedrift/virksomhet eller er du student med en egendefinert/selvlaget oppgave?	- Bedrift/virksomhet
Navn på bedrift/organisasjon/student:	AIT/IDI
Addresse	NTNU
Postnummer	7000
Poststed	Trondheim
Navn på kontaktperson/veileder:	Donn Morrison
Telefon:	00000000
Epost:	donn.morrison@ntnu.no

Forventinger
Oppdragsgiverne som får en gruppe til å velge sitt forslag, må være innstilt på å bidra med minimum 25 timeverk til møter og avklaringer rundt oppgaven med studentene i perioden januar-mai. NTNU sin standardavtale må benyttes. Lenke til informasjon om standardavtalen: https://i.ntnu.no/wiki/-/wiki/Norsk/Standardavtale+mellom+bedrift+og+student

Frist for å levere oppgaveforslag er 5. november 2023.

B. Pre-Project Plan

The pre-project plan in form of a PDF.

025

Adblock for podcasts
Forprosjektplan

Versjon 1.3

Revisjonshistorie

Dato	Versjon	Beskrivelse
20/01/2024	0.1	Formatering, stikkord på punkter 1 - 5
22/01/2024	0.2	Arbeid med punkter 1 - 2 og 4 - 6
25/01/2024	0.3	Arbeid med punkt 3 og 6
26/01/2024	1.0	Arbeid med punkt 3 og 6, ferdigstilling av første versjon
15/03/2024	1.1	Endret link til Gantt-diagram til full URL
13/05/2024	1.2	Omformulering og rettskriving
25/05/2024	1.3	Rettet distinksjon mellom brukerundersøkelse og brukertest

Innholdsfortegnelse

1. Mål og rammer

1.1 Orientering

Oppgaven ble forespurt fordi jeg er interessert i adblocking og mener det spiller en økende rolle for trivsel på internett og ved bruk av digitale tjenester, ettersom mengden reklame på internett virker til å bli stadig større. Dette prosjektet vil nok ikke resultere i noe nytt på adblock-fagfeltet, men vil gi meg økt forståelse av hvordan adblocking gjøres og hvilke utfordringer det byr på, samt hvordan tjenestene adblocking gjøres mot er bygd opp. Dette utgjør ønsket for å gjennomføre prosjektet.

Oppgaven er en standard oppgave som ble lagt ut gjennom oppgaveoversikten på Blackboard og er gitt av NTNU ved Donn Morrison.

1.2 Problemstilling, prosjektbeskrivelse og resultatmål

1.2.1 Problemstilling

Hvordan kan en en adblock-løsning sømløst integreres med lytting til podcast?

1.2.2 Prosjektbeskrivelse

Utvikle en fungerende adblock-tjeneste, med fokus på sømløs integrasjon, for en open-source podcast-tjeneste. Brukerundersøkelser og UX-retningslinjer skal være styrende i designvalg av løsningen. Sluttrapporten for prosjektet skal inkludere en oversikt over det nåværende teknologiske nivået og fagfeltet innenfor adblocking, med fokus på lydbasert adblocking.

1.2.3 Resultatmål

Utvikle minimum én fungerende løsning av en adblocker for en open-source podcast tjeneste som reduserer reklametid per podcast-episode med 70%.

Dette er hovedresultatmålet. Andre resultatmål:

- Utvikle en poster for prosjektet
- Utvikle en sluttraport med alle tilhørende vedlegg
- Løsningen etterstreber sømløs integrasjon som krever minimal oppmerksomhet fra bruker
- Løsningen er utformet i henhold til UX-retningslinjer
- Løsningen er utformet i henhold til tilbakemeldinger fra brukerundersøkelser
- Gjennomføre minimum fem brukertester av sluttproduktet
- Brukerne som deltar på brukerundersøkelsene og brukertester har varierende/spredt fagbakgrunn

Tidsramme: Fire og en halv måned.

1.3 Effektmål

Effektmålet for dette prosjektet er å tilegne seg økt kompetanse innenfor hvordan adblocking gjøres, utfordringene det byr på og tjenestene det gjøres mot, samt gjennomføring av større prosjekt.

Prosjektet gjennomføres ikke på vegne av en arbeidsgiver, så det er ingen effektmål relatert til vinning for en bedrift eller liknende. Det er ikke planlagt at produktet skal lanseres på noen måte annet enn å være et offentlig repository. Det kan allikevel være nyttig å se på hva effektmålene ville vært hvis prosjektet var gitt fra en arbeidsgiver.

I så fall kunne effektmålene vært å utvikle en tjeneste brukere kan kjøpe som reduserer tiden man må høre på reklame ved lytting til podcasts, og gjennom dette øke bruk og lytteglede ved podcast-tjenesten. Herunder et mål om fortjeneste for arbeidsgiver. Et annet mål kunne være energi- og strømsparing, med mål om at løsningen reduserer forbruk ved å ikke hente reklame.

1.4 Rammer

Prosjektet krever ikke utstyr eller materialer utover egen laptop. Prosjektet rettes mot en open-source podcast-tjeneste, som gjør at tilgang på tjenesten og dens kildekode er gitt. Tidsrammen for prosjektet er anslått til fem hundre timer over fire og en halv måned.

2. Organisering

Prosjektet gjennomføres av student ved NTNU Institutt for datateknologi og informatikk (IDI) Christian Ryddheim Dahlin, med førstelektor ved NTNU IDI Donn Morrison som veileder.

Utover dette er Grethe Sandstrak emneansvarlig og kan kontaktes ved generelle spørsmål om emnet.

3. Gjennomføring

3.1. Hovedaktiviteter

Prosjektet er initsielt delt inn i følgende hovedaktiviteter. For datoer, se Gantt-diagram i avsnitt 6.1.

3.1.1 Forberedende arbeid

Forelesninger om vitenskapelig metode og prosjektet, sette seg inn i maler, sette opp en tidsplan og lage forprosjektplan. Arbeid for å forberede prosjektet og for å sikre at man begynner på prosjektet på en konstruktiv og planlagt måte.

3.1.2 Undersøkelser

Lese seg opp på fagfeltet og se på eksisterende løsninger. Bruke informasjonen man samler her til å bestemme seg for hvilken løsning man vil forsøke å utvikle, samt mot hvilken tjeneste.

3.1.3 Utvikling

Utvikling av produktet. Starter med grunnfunksjonalitet, for så å fokusere på UX og UI. Iterativ prosess der man må gå tilbake og endre/forbedre deler, samt oppdatere etter man har funnet ny informasjon f.eks. gjennom brukertester. Forutsetning for å begynne på denne aktiviteten er at man har funnet løsning å utforske, samt tjeneste å rette løsningen mot.

3.1.4 Lage og presentere poster

Lage poster i henhold til mal og presentere denne for andre prosjektgrupper og veiledere. Forutsetter det samme som 3.1.3.

3.1.5 Brukertester

Gjennomføre brukertester og bruke informasjon fra disse til å endre produktet. Forutsetter at man har utviklet en løsning eller MVP som brukerne kan teste.

3.1.6 Rapport

Skrive hovedrapporten og lage alle tilhørende vedlegg. Forutsetter ingenting for å begynne på rapporten, men for å fullføre den må alle andre deler ved prosjektet bortsett fra videopresentasjonen være ferdigstilt.

3.1.7 Lage videopresentasjon

Lage en video av en presentasjon av prosjektet. Forutsetter at alle andre deler ved prosjektet er ferdigstilt.

3.2. Milepæler

Datoer for hovedaktivitetene og andre kritiske datoer kan ses i Gantt-diagrammet i avsnitt 6.1.

4. Oppfølging og kvalitetssikring

4.1 Kvalitetssikring

Punktene prøver å ta høyde for utfordringene ved kvalitetssikring som medfølger av at prosjektet utføres individuelt.

- Gå gjennom produkter en lengre tid etter de er skrevet
- Sende inn arbeid til veileder for tilbakemelding når det er naturlig/mulig
- Lese andre, tidligere oppgaver for å kunne sammenligne
- Teste det som er utviklet jevnlig og omstendig, både ved egen utførte brukertester og ved testing av kode per kodestandarder
- Sette opp pull-request struktur for kodingen der godkjenning/underkjenning gjøres ved senere økter

4.2 Rapportering

Rapportering gjøres til veileder Donn Morrison. Han har uttrykt at rapportering ikke er nødvendig ved faste tidsintervaller, men vil heller gjøres samtidig som prosjektmøter eller ved forespørsel. Unntaket er hvis veileder får inntrykk av at ting ikke går eller gjøres som det skal, da vil veileder ta kontakt og muligens endre kravene til rapportering.

5. Risikovurdering

Risikoene er delt inn i tre kategorier for sannsynlighet: lav, middels og høy.

Hendelse	Sannsynlighet	Konsekvens	Tiltak
Sykdom	Høy	Tap av tid	Jobbe inn tapte timer på tidspunkter utenfor timeplanen Forespørre utsettelse på innlevering av prosjekt
Tilgang/endring av podcast-tjenesten	Lav	Påbegynt / utviklet løsning blir irrelevant	Se på mulighet for å få tjeneste/versjon tilbake Endre tjeneste løsningen rettes mot
Tap av data	Lav	Tilbakesetting / tap av arbeid	Lagre i skytjeneste (Google Drive) Ta jevnlig backup til ekstern disk
Feiltolkninger / misoppfatninger	Middels	Feil/unødvendig arbeid utføres, tap av tid	Gode, konkrete spørsmål til veileder Forsikre om felles forståelse av avtaler/planer

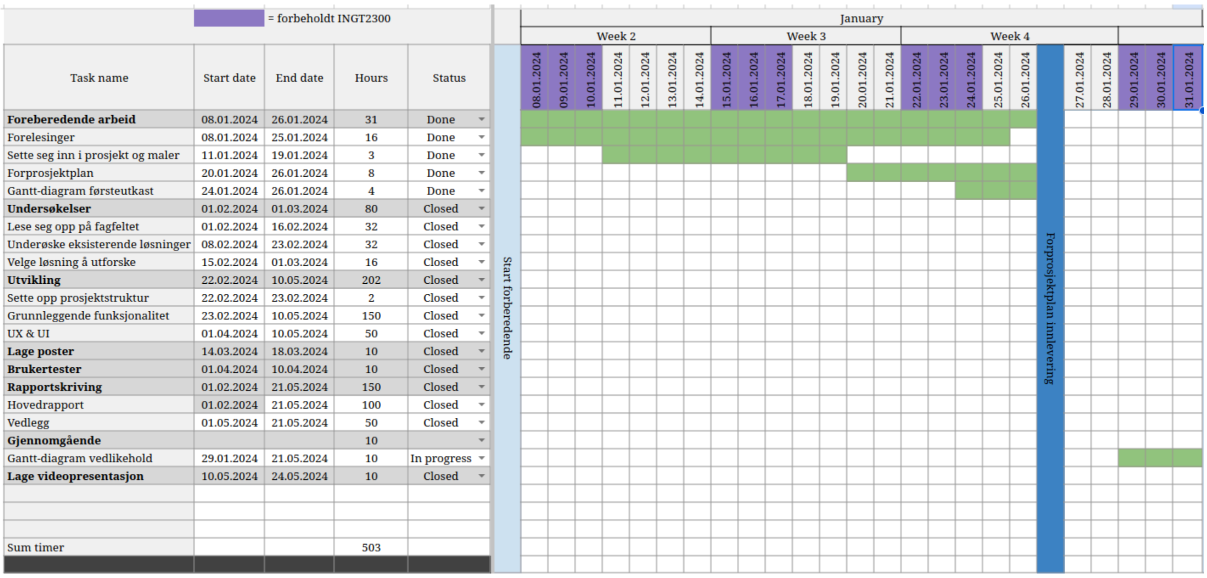
6. Vedlegg

6.1 Tidsplan

Tidsplanen er utformet gjennom et Gantt-diagram. Gantt-diagrammet vist her er et førsteutkast, og vil bli fyllt ut mer i detalj og revidert fortløpende gjennom prosjektet. Diagrammet et stort og dermed utydelig i dette dokumentet, men kan ses i detalj i Google Sheets gjennom denne linken: https://docs.google.com/spreadsheets/d/1HEQ-6-v6O43wHH_0Y3YWxjjGDEpwo8METbpw_MMbL00/edit?usp=sharing.

For å gjøre det mulig å legge ved i dette dokumentet er diagrammet delt opp måned for måned.

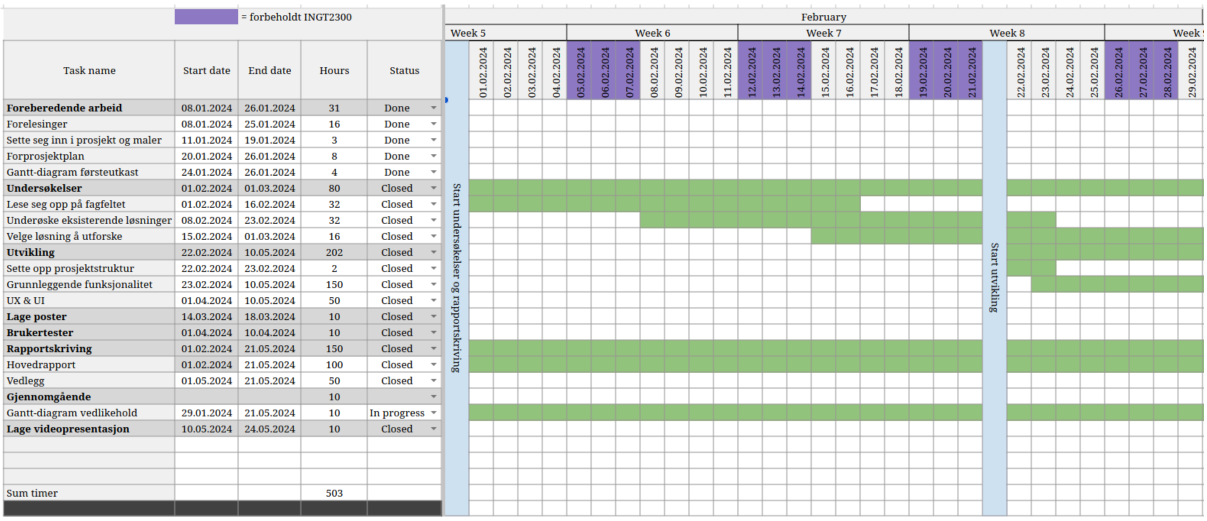
Januar



🟪 = forbeholdt INGT2300

Task name	Start date	End date	Hours	Status
Foreberedende arbeid	08.01.2024	26.01.2024	31	Done
Forelesinger	08.01.2024	25.01.2024	16	Done
Sette seg inn i prosjekt og maler	11.01.2024	19.01.2024	3	Done
Forprosjektplan	20.01.2024	26.01.2024	8	Done
Gantt-diagram førsteutkast	24.01.2024	26.01.2024	4	Done
Undersøkelser	01.02.2024	01.03.2024	80	Closed
Lese seg opp på fagfeltet	01.02.2024	16.02.2024	32	Closed
Underøske eksisterende løsninger	08.02.2024	23.02.2024	32	Closed
Velge løsning å utforske	15.02.2024	01.03.2024	16	Closed
Utvikling	22.02.2024	10.05.2024	202	Closed
Sette opp prosjektstruktur	22.02.2024	23.02.2024	2	Closed
Grunnleggende funksjonalitet	23.02.2024	10.05.2024	150	Closed
UX & UI	01.04.2024	10.05.2024	50	Closed
Lage poster	14.03.2024	18.03.2024	10	Closed
Brukertester	01.04.2024	10.04.2024	10	Closed
Rapportskriving	01.02.2024	21.05.2024	150	Closed
Hovedrapport	01.02.2024	21.05.2024	100	Closed
Vedlegg	01.05.2024	21.05.2024	50	Closed
Gjennomgående			10	
Gantt-diagram vedlikehold	29.01.2024	21.05.2024	10	In progress
Lage videopresentasjon	10.05.2024	24.05.2024	10	Closed
Sum timer			503	

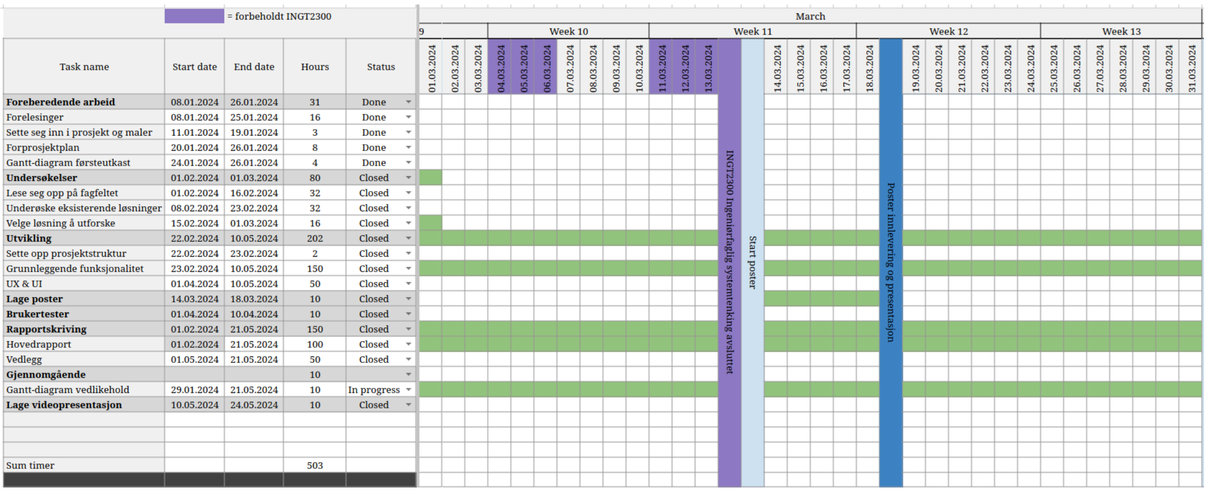
Februar



🟪 = forbeholdt INGT2300

Task name	Start date	End date	Hours	Status
Foreberedende arbeid	08.01.2024	26.01.2024	31	Done
Forelesinger	08.01.2024	25.01.2024	16	Done
Sette seg inn i prosjekt og maler	11.01.2024	19.01.2024	3	Done
Forprosjektplan	20.01.2024	26.01.2024	8	Done
Gantt-diagram førsteutkast	24.01.2024	26.01.2024	4	Done
Undersøkelser	01.02.2024	01.03.2024	80	Closed
Lese seg opp på fagfeltet	01.02.2024	16.02.2024	32	Closed
Underøske eksisterende løsninger	08.02.2024	23.02.2024	32	Closed
Velge løsning å utforske	15.02.2024	01.03.2024	16	Closed
Utvikling	22.02.2024	10.05.2024	202	Closed
Sette opp prosjektstruktur	22.02.2024	23.02.2024	2	Closed
Grunnleggende funksjonalitet	23.02.2024	10.05.2024	150	Closed
UX & UI	01.04.2024	10.05.2024	50	Closed
Lage poster	14.03.2024	18.03.2024	10	Closed
Brukertester	01.04.2024	10.04.2024	10	Closed
Rapportskriving	01.02.2024	21.05.2024	150	Closed
Hovedrapport	01.02.2024	21.05.2024	100	Closed
Vedlegg	01.05.2024	21.05.2024	50	Closed
Gjennomgående			10	
Gantt-diagram vedlikehold	29.01.2024	21.05.2024	10	In progress
Lage videopresentasjon	10.05.2024	24.05.2024	10	Closed
Sum timer			503	

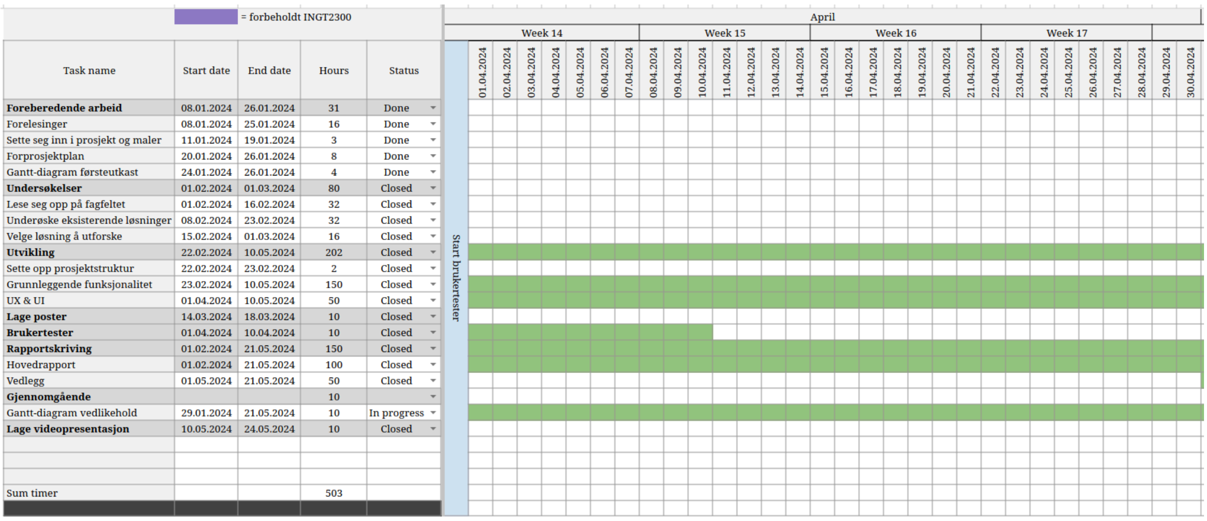
Mars



🟪 = forbeholdt INGT2300

Task name	Start date	End date	Hours	Status
Foreberedende arbeid	08.01.2024	26.01.2024	31	Done
Forelesinger	08.01.2024	25.01.2024	16	Done
Sette seg inn i prosjekt og maler	11.01.2024	19.01.2024	3	Done
Forprosjektplan	20.01.2024	26.01.2024	8	Done
Gantt-diagram førsteutkast	24.01.2024	26.01.2024	4	Done
Undersøkelser	01.02.2024	01.03.2024	80	Closed
Lese seg opp på fagfeltet	01.02.2024	16.02.2024	32	Closed
Underøske eksisterende løsninger	08.02.2024	23.02.2024	32	Closed
Velge løsning å utforske	15.02.2024	01.03.2024	16	Closed
Utvikling	22.02.2024	10.05.2024	202	Closed
Sette opp prosjektstruktur	22.02.2024	23.02.2024	2	Closed
Grunnleggende funksjonalitet	23.02.2024	10.05.2024	150	Closed
UX & UI	01.04.2024	10.05.2024	50	Closed
Lage poster	14.03.2024	18.03.2024	10	Closed
Brukertester	01.04.2024	10.04.2024	10	Closed
Rapportskriving	01.02.2024	21.05.2024	150	Closed
Hovedrapport	01.02.2024	21.05.2024	100	Closed
Vedlegg	01.05.2024	21.05.2024	50	Closed
Gjennomgående			10	
Gantt-diagram vedlikehold	29.01.2024	21.05.2024	10	In progress
Lage videopresentasjon	10.05.2024	24.05.2024	10	Closed
Sum timer			503	

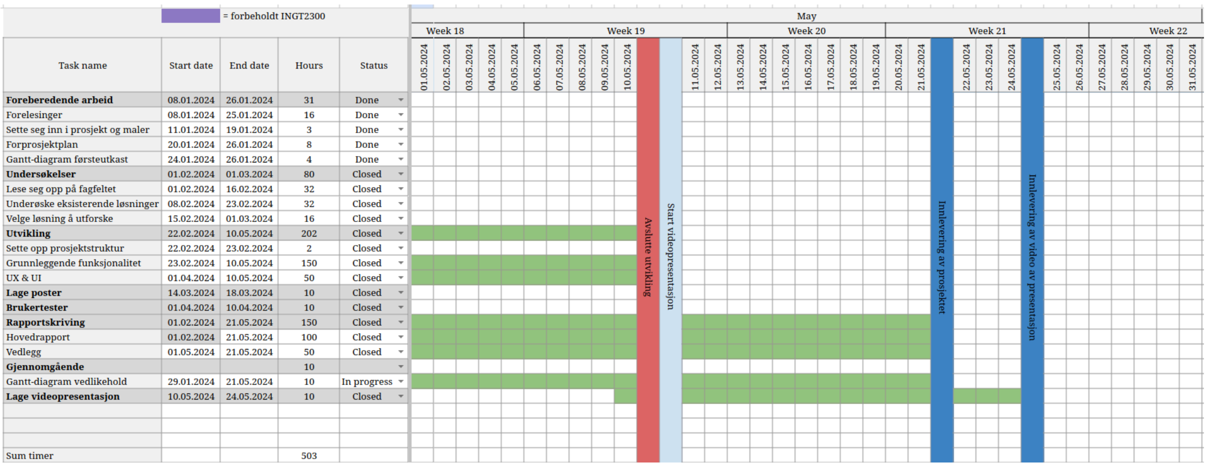
April



🟪 = forbeholdt INGT2300

Task name	Start date	End date	Hours	Status
Foreberedende arbeid	08.01.2024	26.01.2024	31	Done
Forelesinger	08.01.2024	25.01.2024	16	Done
Sette seg inn i prosjekt og maler	11.01.2024	19.01.2024	3	Done
Forprosjektplan	20.01.2024	26.01.2024	8	Done
Gantt-diagram førsteutkast	24.01.2024	26.01.2024	4	Done
Undersøkelser	01.02.2024	01.03.2024	80	Closed
Lese seg opp på fagfeltet	01.02.2024	16.02.2024	32	Closed
Underøske eksisterende løsninger	08.02.2024	23.02.2024	32	Closed
Velge løsning å utforske	15.02.2024	01.03.2024	16	Closed
Utvikling	22.02.2024	10.05.2024	202	Closed
Sette opp prosjektstruktur	22.02.2024	23.02.2024	2	Closed
Grunnleggende funksjonalitet	23.02.2024	10.05.2024	150	Closed
UX & UI	01.04.2024	10.05.2024	50	Closed
Lage poster	14.03.2024	18.03.2024	10	Closed
Brukertester	01.04.2024	10.04.2024	10	Closed
Rapportskriving	01.02.2024	21.05.2024	150	Closed
Hovedrapport	01.02.2024	21.05.2024	100	Closed
Vedlegg	01.05.2024	21.05.2024	50	Closed
Gjennomgående			10	
Gantt-diagram vedlikehold	29.01.2024	21.05.2024	10	In progress
Lage videopresentasjon	10.05.2024	24.05.2024	10	Closed
Sum timer			503	

Mai



▨ = forbeholdt INGT2300

Task name	Start date	End date	Hours	Status
Foreberedende arbeid	08.01.2024	26.01.2024	31	Done
Forelesinger	08.01.2024	25.01.2024	16	Done
Sette seg inn i prosjekt og maler	11.01.2024	19.01.2024	3	Done
Forprosjektplan	20.01.2024	26.01.2024	8	Done
Gantt-diagram førsteutkast	24.01.2024	26.01.2024	4	Done
Undersøkelser	01.02.2024	01.03.2024	80	Closed
Lese seg opp på fagfeltet	01.02.2024	16.02.2024	32	Closed
Underøske eksisterende løsninger	08.02.2024	23.02.2024	32	Closed
Velge løsning å utforske	15.02.2024	01.03.2024	16	Closed
Utvikling	22.02.2024	10.05.2024	202	Closed
Sette opp prosjektstruktur	22.02.2024	23.02.2024	2	Closed
Grunnleggende funksjonalitet	23.02.2024	10.05.2024	150	Closed
UX & UI	01.04.2024	10.05.2024	50	Closed
Lage poster	14.03.2024	18.03.2024	10	Closed
Brukertester	01.04.2024	10.04.2024	10	Closed
Rapportskriving	01.02.2024	21.05.2024	150	Closed
Hovedrapport	01.02.2024	21.05.2024	100	Closed
Vedlegg	01.05.2024	21.05.2024	50	Closed
Gjennomgående			10	
Gantt-diagram vedlikehold	29.01.2024	21.05.2024	10	In progress
Lage videopresentasjon	10.05.2024	24.05.2024	10	Closed
Sum timer			503	

6.2 Adresseliste

Navn	Rolle	Org.	Telefon	E-post	Adresse
Christian Ryddheim Dahlin	Student, utfører av prosjektet	NTNU	97129860	00chrisryda@gmail.com	Rosenborg gate 14B, 7043 Trondheim
Donn Morrison	Veileder	NTNU	N/A	donn.morrison@ntnu.no	IT-bygget, sydfløy, 242, Gløshaugen
Grethe Sandstrak	Emneansvarlig	NTNU	73559561	grethe.sandstrak@ntnu.no	IT-bygget, sydfløy, 108, Gløshaugen

C. Vision Document

The vision document in form of a PDF.

025

Adblock for podcasts Visjonsdokument

Versjon 1.0

Revisjonshistorie

Dato	Versjon	Beskrivelse
15/03/2024	0.1	Oppsett, tilpasning av mal
16/03/2024	0.2	Arbeid med punkter 1 - 4
04/04/2024	0.3	Arbeid med alle punkter
05/04/2024	0.7	Arbeid med alle punkter
06/04/2024	0.8	Arbeid med alle punkter
08/04/2024	0.9	Små endringer, førsteutkast
13/05/2024	1.0	Oppdatert med Tor-forutsetning

Innholdsfortegnelse

1. Innledning

Dette dokumentet gir en overordnet beskrivelse av produktet som utvikles gjennom prosjektoppgave nr. 025 i emnet IDATT2900 Bacheloroppgave. Produktet er en adblocker for open-source podcast-tjenesten AntennaPod (AntennaPod, 2024).

2. Sammendrag problem og produkt

2.1 Problemsammendrag

Problem med	reklame i podcasts
berører	alle aldersgrupper
som resultatet av dette	er brukere nødt til å høre på reklame de vil slippe, eller manuelt spole over reklame
en vellykket løsning vil	fjerne reklame i podcast-episoder uten videre å endre lytteopplevelsen

2.2 Produktsammendrag

For	privatpersoner i alle aldersgrupper
som	ønsker å høre på podcasts uten reklame
adblock25	er en adblocker
som	fjerner reklame fra podcast-episoder
I motsetning til	AntennaPod, som ikke har noen form for adblocking-funksjonalitet
Er det nye produktet	spesielt tilpasset tjenesten og fjerner reklame fra podcast-episoder

3. Overordnet beskrivelse av interessenter og brukere

3.1 Oppsummering interessenter

Navn	Utdypende beskrivelse	Rolle under utviklingen
Veileder	Representerer kunde og sensor	Veileder av produktet, bistår med innspill og er sentral ved prioritering av funksjonalitet
Ekstern sensor	Ekstern sensor med hovedansvar for sensur, får ikke samme kjennskap til produktet som veileder og kan dermed ta en mer objektiv vurdering av produktet	Ingen rolle under utvikling
Utvikler	Utvikler av produktet	Organisere og gjennomføre utviklingen av produktet
Bruker	Potensiell målgruppe	Bistår ved å gjennomføre brukertester og gi tilbakemeldinger

3.2 Oppsummering brukere

Navn	Utdypende beskrivelse	Rolle under utviklingen	Representert av
Veileder		Bistå med tilbakemeldinger og prioritering av funksjonalitet	Seg selv
Bruker	Potensiell målgruppe, privatpersoner som hører på podcasts og vil slippe reklame	Bistår ved å gjennomføre brukertester og gi tilbakemeldinger	Seg selv og ekstern sensor

3.3 Brukermiljøet

Produktet utvikles mot tjenesten AntennaPod kjørende i Waydroid, en emulator som tillater kjøring av en android-enhet på et GNU/Linux-system (Waydroid, 2022).

3.4 Sammendrag av brukernes behov

Behov	Prioritet	Påvirker	Dagens løsning	Foreslått løsning
Fjerne reklame	Høy	Lytting	Ingen, manuelt hoppe over	Automatisk fjerning av reklame fra episoder
Sømløs integrasjon	Høy	UX		Løsning må kreve minst mulig oppsett og vedlikehold av brukere
Lagring av status	Middels	Lytting		Episoder kan gjenopptas der man sist sluttet å høre på dem (uten reklame)
Tilbakestille endring	Lav	Lytting		Hvis en endring i episoden som følge av et forsøk på å fjerne reklame ikke er bra, kan brukeren tilbakestille endringen

3.5 Alternativer til produktet

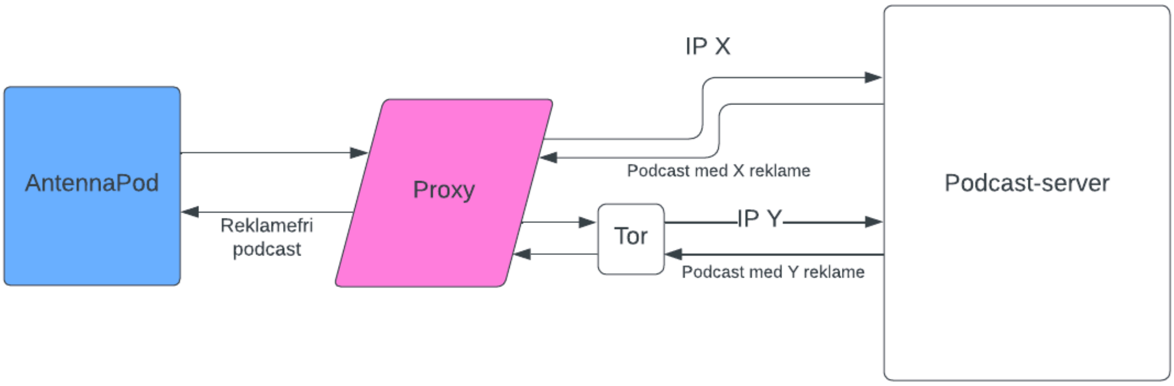
Adblockere med lignende funksjonalitet:

- Adblock Radio (Storelli, 2018)
  - Lydbasert adblocker for radio
- Sponsorblock (Ramachandran, Plsek, u.å.)
  - Adblocker for Youtube som baserer seg på crowdsourcing
- uBlock Origin (Gazvoda, u.å.)
  - Adblocker- og innholdsfiltreringsutvidelse for nettlesere
- AdBlocker Ultimate (Adblocker Ultimate, u.å.)
  - Adblocker-utvidelse for nettlesere

4. Produktoversikt

4.1 Produktets rolle i brukermiljøet

Produktet er en HTTP(S) proxy som sitter mellom AntennaPod og servere. Den mottar forespørsler fra AntennaPod, og henter en episode to ganger: En gang med maskinens lokale IP-adresse og en gang via Tor-nettverket. Ved å sammenligne de to hentede lydfilene levere den en reklamefri lydfil til AntennaPod.



Figur 1: Produktets rolle i brukermiljøet

4.2 Forutsetninger og avhengigheter

Produktet må kjøres lokalt på egen maskin, og opp mot tjenesten AntennaPod brukt gjennom Waydroid på et GNU/Linux-system. I tilleg kreves det tilgang på en Tor-instans (Tor Project, u.å.), enten via Tor-browser eller en daemon.

Produktet er utviklet i Python 3.11, og er ikke garantert å fungere med andre versjoner.

4.3 Produktets funksjonelle egenskaper

Beskrivelse
Funksjonalitet som fjerner reklame fra episoden
Funksjonalitet som gjør at man kan gjenoppta episoder uten reklame
Funksjonalitet som lar en tilbakestille endring

4.4 Ikke-funksjonelle egenskaper og andre krav

Beskrivelse
Løsningen skal etterstrebe sømløs integrasjon med AntennaPod og kreve minst mulig oppsett og vedlikehold av bruker
Brukertester og UI/UX-retningslinjer skal være styrende i designvalg av løsningen
Brukertester av sluttproduktet skal gjennomføres og dokumenteres i rapporten
Rapporten skal gi en oversikt over moderne adblocking med fokus på lydbasert adblocking
Rapporten skal beskrive ulike mulige arkitekturer for løsningen, og hvorfor den valgte løsningen ble valgt

Referanser

AntennaPod (2024) *About*. Tilgjengelig fra https://antennapod.org/about/ (Hentet 04. april 2024)

Waydroid (2022) *Waydroid*. Tilgjengelig fra https://waydro.id/ (Hentet 04. april 2024)

Storelli, A. (2018) *Designing an audio adblock*. Tilgjengelig fra https://www.adblockradio.com/blog/2018/11/15/designing-audio-ad-block-radio-podcast/ (Hentet 04. april 2024)

Ramachandran, A., Plsek, J. (u.å.) *SponsorBlock*. Tilgjengelig fra https://sponsor.ajay.app/ (Hentet 04. april 2024)

Gazvoda, U. (u.å.) *uBlock Origin - Free, open-source ad content blocker*. Tilgjengelig fra https://ublockorigin.com/ (Hentet 04. april 2024)

Adblocker Ultimate (u.å.) *Adblocker ultimate*. Tilgjengelig fra https://adblockultimate.net/ (Hentet 04. april 2024)

Tor Project (u.å.) *Browse Privately. Explore Freely.* Tilgjengelig fra https://www.torproject.org/ (Hentet 13. mai 2024)

D. Requirements Documentation

The requirements documentation in form of a PDF.

025

Adblock for podcasts
Kravdokumentasjon

Versjon 1.0

Revisjonshistorie

Dato	Versjon	Beskrivelse
15/03/2024	0.1	Oppsett, tilpasning av mal
05/04/2024	0.2	Arbeid med alle punkter
06/04/2024	0.3	Arbeid med alle punkter
08/04/2024	0.9	Små endringer, førsteutkast
13/05/2024	1.0	Formulering og rettskriving

Innholdsfortegnelse

1. Introduksjon

Dette dokumentet gir en overordnet beskrivelse av de funksjonelle kravene til produktet som utvikles gjennom prosjektoppgave nr. 025 i emnet IDATT2900 Bacheloroppgave. Produktet er en adblocker for open-source podcast-tjenesten AntennaPod (AntennaPod, 2024).

2. User stories

Som bruker
Ønsker jeg å få fjernet reklame fra podcast-episoder
Slik at jeg slipper å høre på dem

Som bruker
Ønsker jeg å kunne gjenoppta episoder uten reklame
Slik at det ikke er nødvendig å starte episoder fra starten av

Som bruker
Ønsker jeg å kunne tilbakestille endring av episoder
Slik at den vanlige episoden er tilgjengelig hvis produktet gjør en feil

2.1 Scenarioer

Som bruker
Ønsker jeg å få fjernet reklame fra podcast-episoder
Slik at jeg slipper å høre på dem

Scenario: starte podcast-episode
Gitt at produktet kjører
Når jeg begynner å spille en episode
Så fjerner produktet reklamen automatisk fra episoden

Som bruker
Ønsker jeg å kunne gjenoppta episoder uten reklame fra tidligere
Slik at det ikke er nødvendig å starte episoder fra starten av

Scenario: gjenoppta podcast-episode uten reklame
Gitt at produktet kjører
Når jeg gjenopptar en episode
Så fjerner produktet reklamen automatisk fra episoden

Som bruker
Ønsker jeg å kunne tilbakestille/fjerne endring av episoder
Slik at den vanlige episoden er lett tilgjengelig hvis produktet gjør en feil

Scenario: tilbakestille endring av episoder
Gitt at produktet kjører
  Og jeg hører på en episode som produktet har fjernet reklame fra
Når jeg velger å tilbakestille endringen
Så får jeg det originale lydsporet til episoden

Referanser

AntennaPod (2024) *About*. Tilgjengelig fra https://antennapod.org/about/ (Hentet 05. april 2024)

E. Project Manual

The project manual in form of a PDF.

025

Adblock for podcasts
Prosjekthåndbok

Versjon 1.0

Revisjonshistorie

Dato	Versjon	Beskrivelse
19/03/2024	0.1	Formatering, oppsett av punkter 1- 3, la inn møter #00 og #01
03/04/2024	0.2	La inn møte #02
15/04/2024	0.3	La inn møte #03
27/05/2024	0.4	La inn alle timelister bortsett fra uke 22 og oppsummering
29/05/2024	1.0	La inn alt manglende

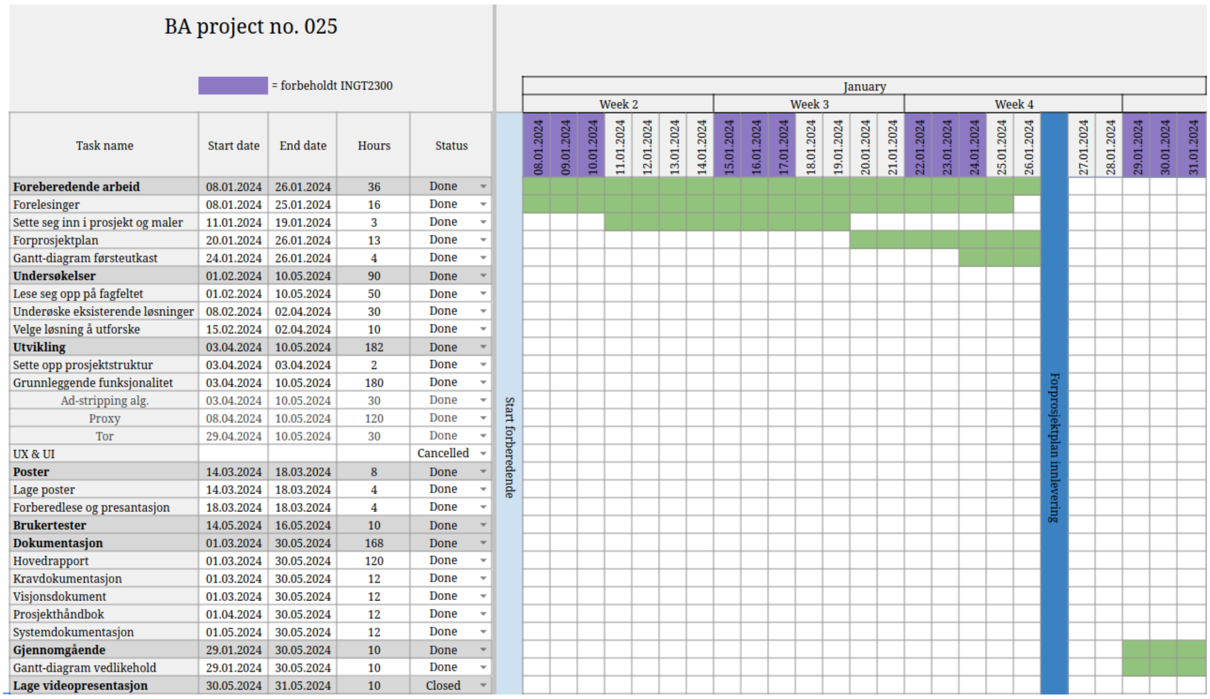
Innholdsfortegnelse

1. Framdriftsplan - Gantt-diagram

Gantt diagrammet er tilgjengelig via denne URL:

https://docs.google.com/spreadsheets/d/1HEQ-6-v6O43wHH_0Y3YWxjjGDEpwo8METbpw_MMbL00/edit#gid=0

Januar

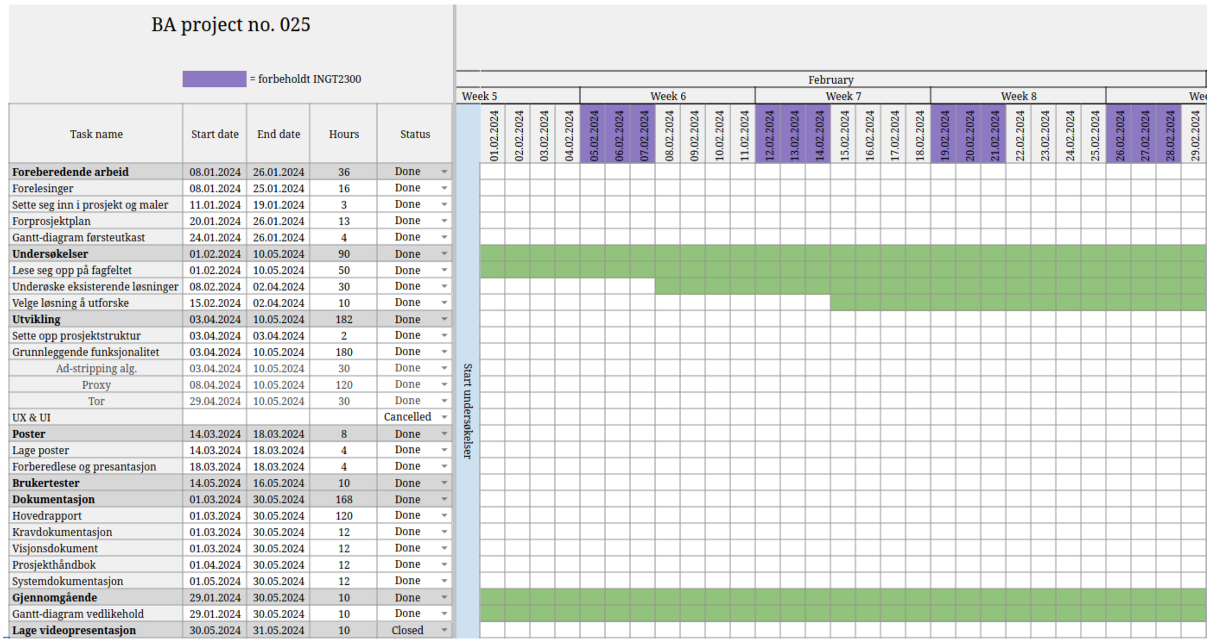


BA project no. 025

▨ = forbeholdt INGT2300

Task name	Start date	End date	Hours	Status
Foreberedende arbeid	08.01.2024	26.01.2024	36	Done
Forelesinger	08.01.2024	25.01.2024	16	Done
Sette seg inn i prosjekt og maler	11.01.2024	19.01.2024	3	Done
Forprosjektplan	20.01.2024	26.01.2024	13	Done
Gantt-diagram førsteutkast	24.01.2024	26.01.2024	4	Done
Undersøkelser	01.02.2024	10.05.2024	90	Done
Lese seg opp på fagfeltet	01.02.2024	10.05.2024	50	Done
Underøske eksisterende løsninger	08.02.2024	02.04.2024	30	Done
Velge løsning å utforske	15.02.2024	02.04.2024	10	Done
Utvikling	03.04.2024	10.05.2024	182	Done
Sette opp prosjektstruktur	03.04.2024	03.04.2024	2	Done
Grunnleggende funksjonalitet	03.04.2024	10.05.2024	180	Done
Ad-stripping alg.	03.04.2024	10.05.2024	30	Done
Proxy	08.04.2024	10.05.2024	120	Done
Tor	29.04.2024	10.05.2024	30	Done
UX & UI				Cancelled
Poster	14.03.2024	18.03.2024	8	Done
Lage poster	14.03.2024	18.03.2024	4	Done
Forberedlese og presantasjon	18.03.2024	18.03.2024	4	Done
Brukertester	14.05.2024	16.05.2024	10	Done
Dokumentasjon	01.03.2024	30.05.2024	168	Done
Hovedrapport	01.03.2024	30.05.2024	120	Done
Kravdokumentasjon	01.03.2024	30.05.2024	12	Done
Visjonsdokument	01.03.2024	30.05.2024	12	Done
Prosjekthåndbok	01.04.2024	30.05.2024	12	Done
Systemdokumentasjon	01.05.2024	30.05.2024	12	Done
Gjennomgående	29.01.2024	30.05.2024	10	Done
Gantt-diagram vedlikehold	29.01.2024	30.05.2024	10	Done
Lage videopresentasjon	30.05.2024	31.05.2024	10	Closed

Februar

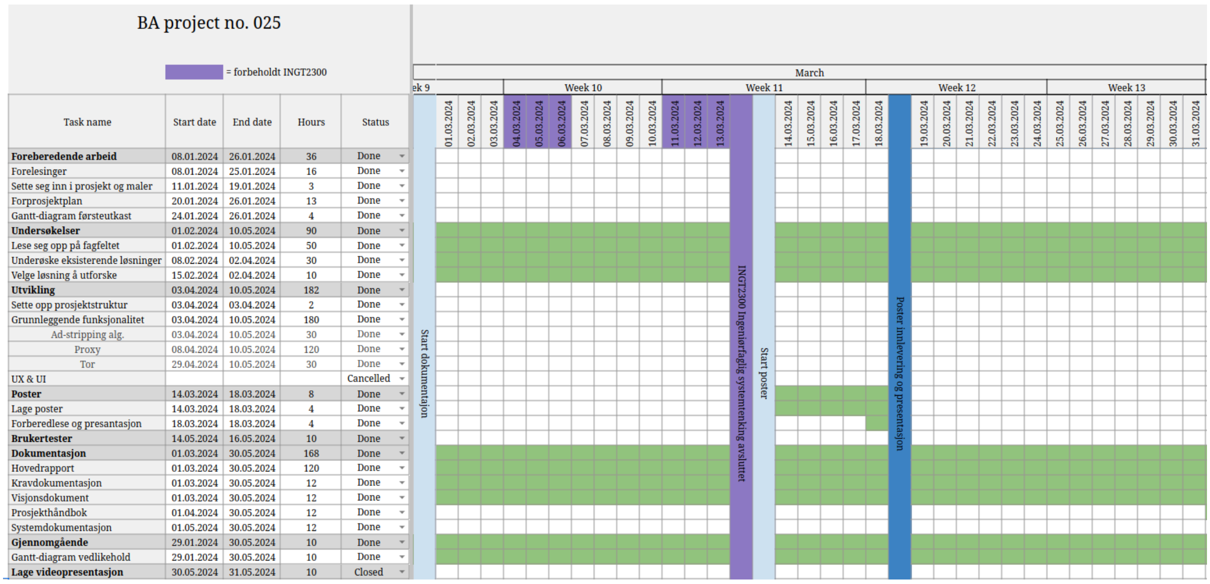


BA project no. 025

▨ = forbeholdt INGT2300

Task name	Start date	End date	Hours	Status
Foreberedende arbeid	08.01.2024	26.01.2024	36	Done
Forelesinger	08.01.2024	25.01.2024	16	Done
Sette seg inn i prosjekt og maler	11.01.2024	19.01.2024	3	Done
Forprosjektplan	20.01.2024	26.01.2024	13	Done
Gantt-diagram førsteutkast	24.01.2024	26.01.2024	4	Done
Undersøkelser	01.02.2024	10.05.2024	90	Done
Lese seg opp på fagfeltet	01.02.2024	10.05.2024	50	Done
Underøske eksisterende løsninger	08.02.2024	02.04.2024	30	Done
Velge løsning å utforske	15.02.2024	02.04.2024	10	Done
Utvikling	03.04.2024	10.05.2024	182	Done
Sette opp prosjektstruktur	03.04.2024	03.04.2024	2	Done
Grunnleggende funksjonalitet	03.04.2024	10.05.2024	180	Done
Ad-stripping alg.	03.04.2024	10.05.2024	30	Done
Proxy	08.04.2024	10.05.2024	120	Done
Tor	29.04.2024	10.05.2024	30	Done
UX & UI				Cancelled
Poster	14.03.2024	18.03.2024	8	Done
Lage poster	14.03.2024	18.03.2024	4	Done
Forberedlese og presantasjon	18.03.2024	18.03.2024	4	Done
Brukertester	14.05.2024	16.05.2024	10	Done
Dokumentasjon	01.03.2024	30.05.2024	168	Done
Hovedrapport	01.03.2024	30.05.2024	120	Done
Kravdokumentasjon	01.03.2024	30.05.2024	12	Done
Visjonsdokument	01.03.2024	30.05.2024	12	Done
Prosjekthåndbok	01.04.2024	30.05.2024	12	Done
Systemdokumentasjon	01.05.2024	30.05.2024	12	Done
Gjennomgående	29.01.2024	30.05.2024	10	Done
Gantt-diagram vedlikehold	29.01.2024	30.05.2024	10	Done
Lage videopresentasjon	30.05.2024	31.05.2024	10	Closed

Mars

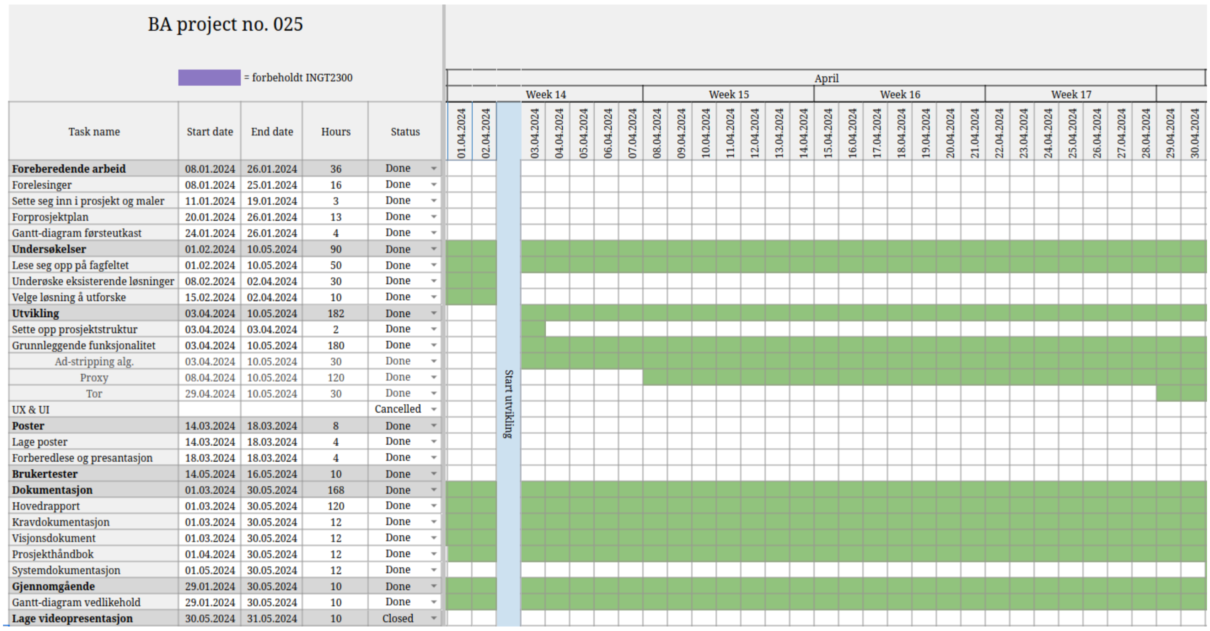


BA project no. 025

🟪 = forbeholdt INGT2300

Task name	Start date	End date	Hours	Status
Foreberedende arbeid	08.01.2024	26.01.2024	36	Done
Forelesinger	08.01.2024	25.01.2024	16	Done
Sette seg inn i prosjekt og maler	11.01.2024	19.01.2024	3	Done
Forprosjektplan	20.01.2024	26.01.2024	13	Done
Gantt-diagram førsteutkast	24.01.2024	26.01.2024	4	Done
Undersøkelser	01.02.2024	10.05.2024	90	Done
Lese seg opp på fagfeltet	01.02.2024	10.05.2024	50	Done
Underøske eksisterende løsninger	08.02.2024	02.04.2024	30	Done
Velge løsning å utforske	15.02.2024	02.04.2024	10	Done
Utvikling	03.04.2024	10.05.2024	182	Done
Sette opp prosjektstruktur	03.04.2024	03.04.2024	2	Done
Grunnleggende funksjonalitet	03.04.2024	10.05.2024	180	Done
Ad-stripping alg.	03.04.2024	10.05.2024	30	Done
Proxy	08.04.2024	10.05.2024	120	Done
Tor	29.04.2024	10.05.2024	30	Done
UX & UI				Cancelled
Poster	14.03.2024	18.03.2024	8	Done
Lage poster	14.03.2024	18.03.2024	4	Done
Forberedlese og presantasjon	18.03.2024	18.03.2024	4	Done
Brukertester	14.05.2024	16.05.2024	10	Done
Dokumentasjon	01.03.2024	30.05.2024	168	Done
Hovedrapport	01.03.2024	30.05.2024	120	Done
Kravdokumentasjon	01.03.2024	30.05.2024	12	Done
Visjonsdokument	01.03.2024	30.05.2024	12	Done
Prosjekthåndbok	01.04.2024	30.05.2024	12	Done
Systemdokumentasjon	01.05.2024	30.05.2024	12	Done
Gjennomgående	29.01.2024	30.05.2024	10	Done
Gantt-diagram vedlikehold	29.01.2024	30.05.2024	10	Done
Lage videopresentasjon	30.05.2024	31.05.2024	10	Closed

April

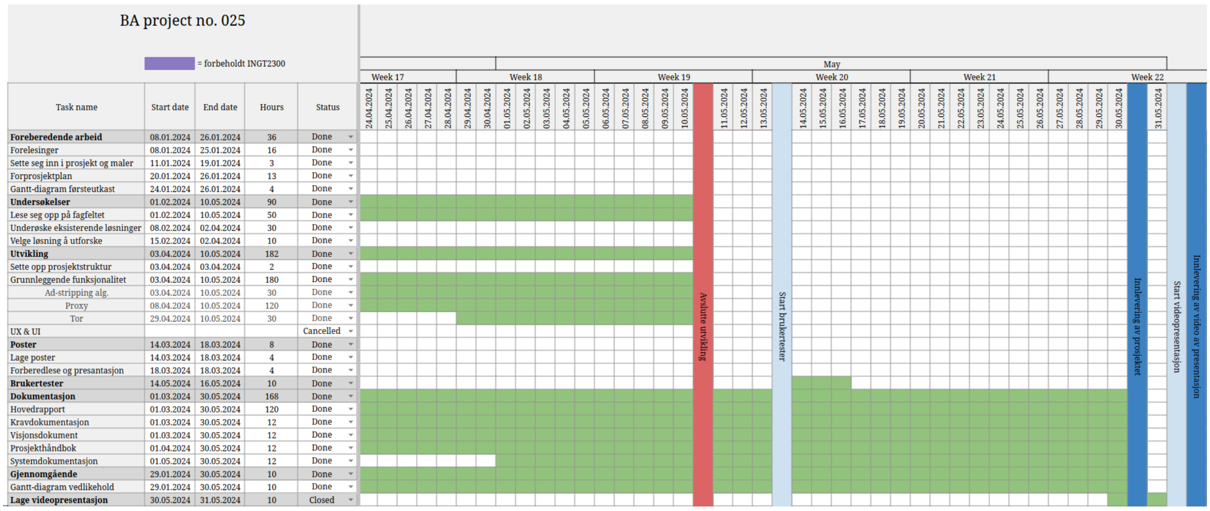


BA project no. 025

🟪 = forbeholdt INGT2300

Task name	Start date	End date	Hours	Status
Foreberedende arbeid	08.01.2024	26.01.2024	36	Done
Forelesinger	08.01.2024	25.01.2024	16	Done
Sette seg inn i prosjekt og maler	11.01.2024	19.01.2024	3	Done
Forprosjektplan	20.01.2024	26.01.2024	13	Done
Gantt-diagram førsteutkast	24.01.2024	26.01.2024	4	Done
Undersøkelser	01.02.2024	10.05.2024	90	Done
Lese seg opp på fagfeltet	01.02.2024	10.05.2024	50	Done
Underøske eksisterende løsninger	08.02.2024	02.04.2024	30	Done
Velge løsning å utforske	15.02.2024	02.04.2024	10	Done
Utvikling	03.04.2024	10.05.2024	182	Done
Sette opp prosjektstruktur	03.04.2024	03.04.2024	2	Done
Grunnleggende funksjonalitet	03.04.2024	10.05.2024	180	Done
Ad-stripping alg.	03.04.2024	10.05.2024	30	Done
Proxy	08.04.2024	10.05.2024	120	Done
Tor	29.04.2024	10.05.2024	30	Done
UX & UI				Cancelled
Poster	14.03.2024	18.03.2024	8	Done
Lage poster	14.03.2024	18.03.2024	4	Done
Forberedlese og presantasjon	18.03.2024	18.03.2024	4	Done
Brukertester	14.05.2024	16.05.2024	10	Done
Dokumentasjon	01.03.2024	30.05.2024	168	Done
Hovedrapport	01.03.2024	30.05.2024	120	Done
Kravdokumentasjon	01.03.2024	30.05.2024	12	Done
Visjonsdokument	01.03.2024	30.05.2024	12	Done
Prosjekthåndbok	01.04.2024	30.05.2024	12	Done
Systemdokumentasjon	01.05.2024	30.05.2024	12	Done
Gjennomgående	29.01.2024	30.05.2024	10	Done
Gantt-diagram vedlikehold	29.01.2024	30.05.2024	10	Done
Lage videopresentasjon	30.05.2024	31.05.2024	10	Closed

Mai



BA project no. 025

🟪 = forbeholdt INGT2300

Task name	Start date	End date	Hours	Status
Foreberedende arbeid	08.01.2024	26.01.2024	36	Done
Forelesinger	08.01.2024	25.01.2024	16	Done
Sette seg inn i prosjekt og maler	11.01.2024	19.01.2024	3	Done
Forprosjektplan	20.01.2024	26.01.2024	13	Done
Gantt-diagram førsteutkast	24.01.2024	26.01.2024	4	Done
Undersøkelser	01.02.2024	10.05.2024	90	Done
Lese seg opp på fagfeltet	01.02.2024	10.05.2024	50	Done
Underøske eksisterende løsninger	08.02.2024	02.04.2024	30	Done
Velge løsning å utforske	15.02.2024	02.04.2024	10	Done
Utvikling	03.04.2024	10.05.2024	182	Done
Sette opp prosjektstruktur	03.04.2024	03.04.2024	2	Done
Grunnleggende funksjonalitet	03.04.2024	10.05.2024	180	Done
Ad-stripping alg.	03.04.2024	10.05.2024	30	Done
Proxy	08.04.2024	10.05.2024	120	Done
Tor	29.04.2024	10.05.2024	30	Done
UX & UI				Cancelled
Poster	14.03.2024	18.03.2024	8	Done
Lage poster	14.03.2024	18.03.2024	4	Done
Forberedlese og presantasjon	18.03.2024	18.03.2024	4	Done
Brukertester	14.05.2024	16.05.2024	10	Done
Dokumentasjon	01.03.2024	30.05.2024	168	Done
Hovedrapport	01.03.2024	30.05.2024	120	Done
Kravdokumentasjon	01.03.2024	30.05.2024	12	Done
Visjonsdokument	01.03.2024	30.05.2024	12	Done
Prosjekthåndbok	01.04.2024	30.05.2024	12	Done
Systemdokumentasjon	01.05.2024	30.05.2024	12	Done
Gjennomgående	29.01.2024	30.05.2024	10	Done
Gantt-diagram vedlikehold	29.01.2024	30.05.2024	10	Done
Lage videopresentasjon	30.05.2024	31.05.2024	10	Closed

2. Møteinnkallinger med referat

#00 - Oppstartsmøte 17.01.24

Møteinnkalling

Møtet ble avtalt på dagen så møteinnkalling ble ikke sendt ut.

Referat

#00 - Referat fra oppstartsmøte bacheloroppgave 025

Dato og tid: 17.01.24 kl 08

Sted: Digitalt via Jitsi meet

Til stede: Christian, Donn (veileder)

Frafall: Ingen

Møteleder: Christian

Sak 01/2024: Kommentarer til møteinnkalling

N/A - Møtet ble avtalt på dagen så møteinnkalling ble ikke sendt ut

Sak 02/2024: Kommunikasjon og samarbeid

- Hvilke(n) plattform skal kommunikasjon foregå gjennom?
  - E-post
- Hvilke(n) plattform skal dokumenter deles gjennom?
  - Google drive, e-post

Sak 03/2024: Konkretisering av oppgaven

- Bred og generell konkretisering av oppgaven
  - Open-ended, fleksibel på hvilken løsning jeg utforsker
  - Løsning som fungere bra for en spesifikk podcast-tjeneste er bra, evt. proxy er like bra
- Ad block
  - Maskinlæring, prosessere lyd (klassifisering, window-by-window)
  - Geolokasjon IP
    - Sammenligne lyd fra ulike land for å finne reklame
  - Sponsor-block, brukes bla. på YT
  - Dele opp lyd ut ifra ulike talere

  - Se på hva som er gjort tidligere, hva som er mulig/sannsynlig å få til

Sak 04/2024: Rammer for oppgaven

- Bruk av prosjekthåndboka
  - Vi avtaler dette senere
- Timelister - hvor jevnlig, hvilken form
  - Jeg velger format selv
  - Donn trenger ikke se timelister med mindre problemer oppstår

- Møteplan
  - On demand
- Krav til prosess
  - Opp til meg
- Krav til dokumentasjon
  - Bruke malene som utgangspunkt
  - User stories
- Språk i oppgaven
  - Hovedrapporten på English, resten velger jeg selv
- Ambisjonsnivå
  - All in

Sak 05/2024: Eventuelt

Er noen ML-prosjekter fra i høst som kan være relevante og en tidligere BA-oppgave, Donn sender dem.

17.01.2024, Christian

Revidert 01.02.2024

#01 - Prosjektmøte 23.02.24

Møteinnkalling

Innkalling til møte: Bacheloroppgave 025

Tidspunkt/sted: 23.02.24 kl 14:00, 242, sydfløy, IT-bygget

Følgende personer innkalles:

Christian

Donn (veileder)

Agenda:

Sak 01/2024: Møtereferat fra forrige møte gjennomgås

Sak 02/2024: Kommentarer til møteinnkalling

Sak 03/2024: Språk

Sak 04/2024: Hvordan bør jeg ligge an?

Sak 05/2024: Generell hjelp

Sak 06/2024: Eventuelt

Trondheim 22.02.24

Referat

#01 - Referat fra prosjektmøte bacheloroppgave 025

Dato og tid: 23.02.24 kl 14:00

Sted: IT bygget, sydfløy 242

Til stede: Christian, Donn (veileder)

Frafall: Ingen

Møteleder: Christian

Sak 01/2024: Møtereferat fra forrige møte gjennomgås

Ser bra ut.

Sak 02/2024: Kommentarer til møteinnkalling

Ingen kommentarer.

Sak 03/2024: Språk i oppgaven

Vi ble enige sist om at selve rapporten skal være på engelsk, resten som jeg ønsker. Vil bare bekrefte at dette betyr at rapporten kan være engelsk og alle vedlegg norske.

- Donn bekrefter at dette er greit

Sak 04/2024: Hvordan bør jeg ligge an?

Hva bør være ferdig, hva bør være underveis / planlagt per nå og i nær fremtid?

- Dokumentasjon bør være på vei, spesielt visjonsdokument og kravdokumentasjon

Sak 05/2024: Generell hjelp

Det jeg finner baserer seg gjerne på å laste ned lydfiler og behandle de lokalt. Kan du peke meg i noen retninger mot “sømløs integrasjon” og “real-time” behandling? HTTP proxy? Pakkefangst? Dataset er tidkrevende å skaffe, så jeg prøver å finne en løsning som ikke krever ML.
Forslag:

- Forskjellige reklamer basert på IP fra land, kan sammenligne to lydfiler samtidig og fjerne delene som er ulike, buffer på f.eks. 100 kB

- HTTP proxy
- Add-on til tjenesten som forteller proxyen å oppdatere podcasten jevnlig eller ved forandringer
- Fork a podcast with Tor (removes ads)
- Bruke ML delvis, men ikke som selve løsningen
- Sponsor block, som på YT
- Python whisper lib, speech-to-text bibliotek
- Crowd sourcing?

Rapporten: Viktig å få med diskusjon rundt etikk og mulighet (feasibility) av løsningen jeg velger, samt kostnader av løsningen, environmental and computational costs, f.eks. at web-basert ad-blocking sparer energi da ressurser ikke må hentes, lengre batteritid osv., trade-offs mm.

Sak 06/2024: Eventuelt

Ingenting.

23.02.2024, Christian

Revidert 29.02.2024

#02 - Prosjektmøte 03.04.24

Møteinnkalling

Innkalling til møte: Bacheloroppgave 025

Tidspunkt/sted: 03.04.24 kl 14:00, 242, sydfløy, IT-bygget

Følgende personer innkalles:

Christian

Donn (veileder)

Agenda:

Sak 01/2024: Møtereferat fra forrige møte gjennomgås

Sak 02/2024: Kommentarer til møteinnkalling

Sak 03/2024: Veiledning / hjelp

Sak 04/2024: Eventuelt

Trondheim 03.04.24

Referat

#02 - Referat fra prosjektmøte bacheloroppgave 025

Dato og tid: 03.04.2024 kl 11:00

Sted: IT bygget, sydfløy 242

Til stede: Christian, Donn (veileder)

Frafall: Ingen

Møteleder: Christian

Sak 01/2024: Møtereferat fra forrige møte gjennomgås

Ser bra ut.

Sak 02/2024: Kommentarer til møteinnkalling

Ingen kommentarer.

Sak 03/2024: Veiledning / hjelp

Ting har vært vanskeligere enn jeg hadde håpet, og jeg sliter med å arbeide effektivt

- Bør be om utsettelse, kontakte Grethe med Donn på kopi
- Påvirker utsettelse mastersøknad?

- Prøve å komme i arbeidsmodus, komme i gang med utvikling raskt

Må være sikker på hvordan ting gjøres med cookies og HTTP. Må kunne dokumentere alt som skrives i thesis. Ekstern sensor vil først og fremst se på thesis og vedlegg, ikke source-code.

Alternativer

- Add on til tjeneste
- Forke hele tjenesten legge til adblocker
- Pga. tid er kanskje proof of concept best
- Backend HTTP-proxy / interceptor / reverse proxy
    - Dette gir lite UX, kanskje en liten web-side med innstillinger

- NginX har reverse proxy, web-server med domenenavn kan kanskje skaffes gratis
- Sikte på prototype som kun kjører lokalt på egen maskin
- Apache har også reverse proxy
- Integrere reverse proxy med vpn / form for tunnel, lese begge filer samtidig og fjerne de ulike delene (som vil være plasserte reklamer)

Bør starte med å skrive program som sammenligner binære filer og fjerner forskjellene, for så å integrere dette med reverse proxy

Må fikse VPN-løsning selv, ha en VPN instans kjørende og integrere den
Kan også bruke Tor til samme formål, rescripte rc-filen
VPN er kanskje enklere enn Tor, men legger muligens til mer kompleksitet for bruker av sluttproduktet

Ha fokus på at den endelig løsningen er slik at den krever minst mulig set up av brukeren.

Sammenligning av bytestreams:
Kan bruke XOR, og shifte reader frem til likheter igjen
Comp diff

Dokumenter:
Skal være enkelt for en bruker å bruke, fokus på at set up er enkelt, fleksibilitet når det gjelder valg av land som brukes til IP

Sak 04/2024: Eventuelt

Ingenting.

03.04.2024, Christian

#03 - Prosjektmøte 15.04.24

Møteinnkalling

Innkalling til møte: Bacheloroppgave 025

Tidspunkt/sted: 15.04.24 kl 11:00, 242, sydfløy, IT-bygget

Følgende personer innkalles:

Christian

Donn (veileder)

Agenda:

Sak 01/2024: Møtereferat fra forrige møte gjennomgås

Sak 02/2024: Kommentarer til møteinnkalling

Sak 03/2024: Veiledning / hjelp

Sak 04/2024: Eventuelt

Trondheim 15.04.24

Referat

#03 - Referat fra prosjektmøte bacheloroppgave 025

Dato og tid: 15.04.2024 kl 11:00

Sted: IT bygget, sydfløy 242

Til stede: Christian, Donn (veileder)

Frafall: Ingen

Møteleder: Christian

Sak 01/2024: Møtereferat fra forrige møte gjennomgås

Ser bra ut.

Sak 02/2024: Kommentarer til møteinnkalling

Ingen kommentarer.

Sak 03/2024: Veiledning / hjelp

Proxy

- Utvikle selv i Python
- Man kan konfigurere en proxy inn på AntennaPod

VPN/Tor

- Vil muligens kreve SSL sertifikat, kan være self-signed.
- Bruke tor binary, en deamon som kan bruke SOCKS proxy hvor all kommunikasjon går gjennom Tor. Kan videreutvikle til å kunne konfigurere exit-nodes som vanligvis kjøres på VPS-er

Beste starten er å sette opp en proxy på lokale maskin som logger hver individuell request

- Kan filtrere basert på forespørsler, vil hente ut mp3 forespørsler
- Eks: En Squid autentisert proxy kjørende i Docker

Kan se an angående brukerundersøkelser senere

- Hvis det kun blir en backendserver blir det ikke noe UI å snakke om uansett
- Brukertester av sluttprodukt er kanskje viktigere
    - Lurt å få tak i personer allerede nå

Thesis

- Undersøke og sammenligne mange forskjellige løsninger, skrive om fordeler og ulemper ved hver enkelt
- Andre tilnærminger:
    - Kunne f.eks. laget en proxy som oppdager RSS feeds som ofte oppdateres (som vil utgjøre de podcastene man hører på jevnlig), og pre-fetche nye episoder og gjøre prosesseringen i forkant slik at en reklamefri episode er klar
    - Crowdsourcing som sjekker hvilke segmenter som er mest hoppet over
    - Forskjellige ML løsninger
- Thesis må utforske de ulike løsningenes fordeler og ulemper, f.eks. mer eller mindre UX/UI, computational expensive, oppsett, hastighet, kompleksitet

Sak 04/2024: Eventuelt

Ingenting.

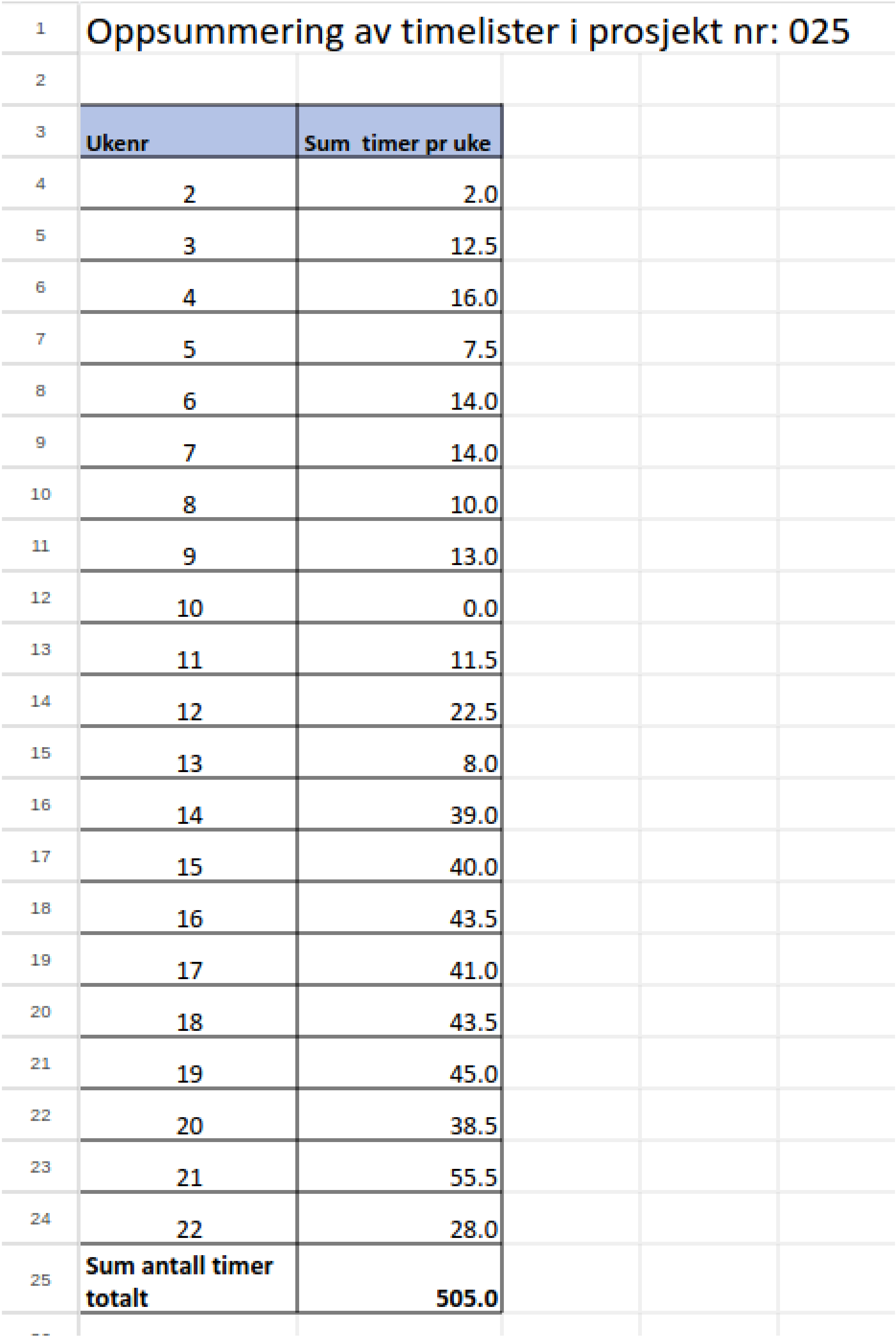
15.04.2024, Christian

3. Timeliste med statusrapporter

Timelistene er tilgjengelig via denne URL:

https://docs.google.com/spreadsheets/d/1efHS7is6W5V45BvaZmNKJofgGPn1Sc8XzRCpZ0SJXVI/edit#gid=1234621017

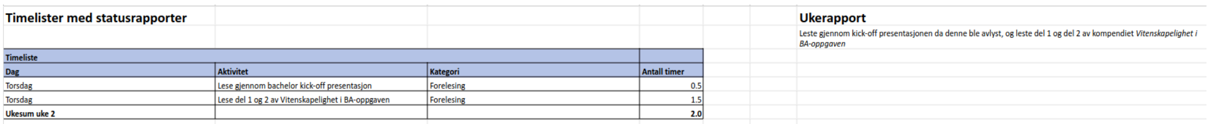
Oppsummering



Oppsummering av timelister i prosjekt nr: 025

Ukenr	Sum timer pr uke
2	2.0
3	12.5
4	16.0
5	7.5
6	14.0
7	14.0
8	10.0
9	13.0
10	0.0
11	11.5
12	22.5
13	8.0
14	39.0
15	40.0
16	43.5
17	41.0
18	43.5
19	45.0
20	38.5
21	55.5
22	28.0
Sum antall timer totalt	**505.0**

Uke 2



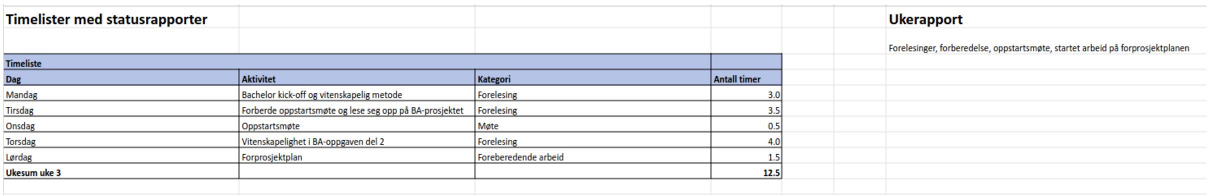
Timelister med statusrapporter

Timeliste			
Dag	**Aktivitet**	**Kategori**	**Antall timer**
Torsdag	Lese gjennom bachelor kick-off presentasjon	Forelesing	0.5
Torsdag	Lese del 1 og 2 av Vitenskapelighet i BA-oppgaven	Forelesing	1.5
Ukesum uke 2			**2.0**

Ukerapport

Leste gjennom kick-off presentasjonen da denne ble avlyst, og leste del 1 og del 2 av kompendiet *Vitenskapelighet i BA-oppgaven*

Uke 3



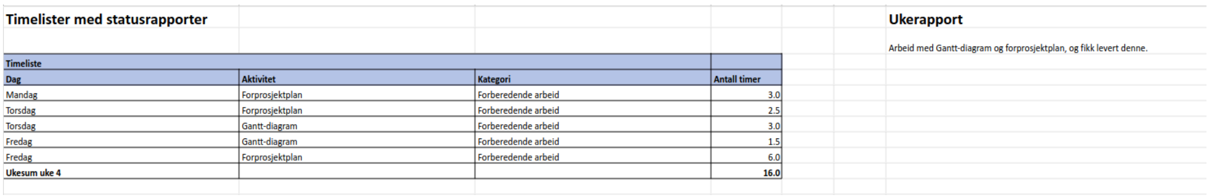
Timelister med statusrapporter

Timeliste			
Dag	**Aktivitet**	**Kategori**	**Antall timer**
Mandag	Bachelor kick-off og vitenskapelig metode	Forelesing	3.0
Tirsdag	Forberde oppstartsmøte og lese seg opp på BA-prosjektet	Forelesing	3.5
Onsdag	Oppstartsmøte	Møte	0.5
Torsdag	Vitenskapelighet i BA-oppgaven del 2	Forelesing	4.0
Lørdag	Forprosjektplan	Forberedende arbeid	1.5
Ukesum uke 3			**12.5**

Ukerapport

Forelesinger, forberedelse, oppstartsmøte, startet arbeid på forprosjektplanen

Uke 4



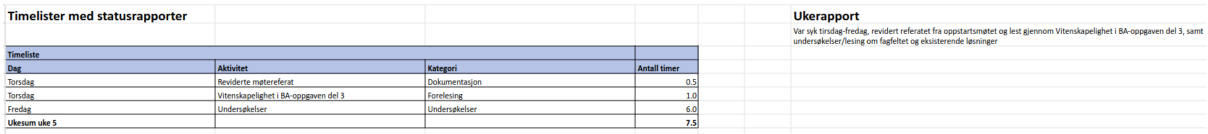
Timelister med statusrapporter

Timeliste			
Dag	**Aktivitet**	**Kategori**	**Antall timer**
Mandag	Forprosjektplan	Forberedende arbeid	3.0
Torsdag	Forprosjektplan	Forberedende arbeid	2.5
Torsdag	Gantt-diagram	Forberedende arbeid	3.0
Fredag	Gantt-diagram	Forberedende arbeid	1.5
Fredag	Forprosjektplan	Forberedende arbeid	6.0
Ukesum uke 4			**16.0**

Ukerapport

Arbeid med Gantt-diagram og forprosjektplan, og fikk levert denne.

Uke 5



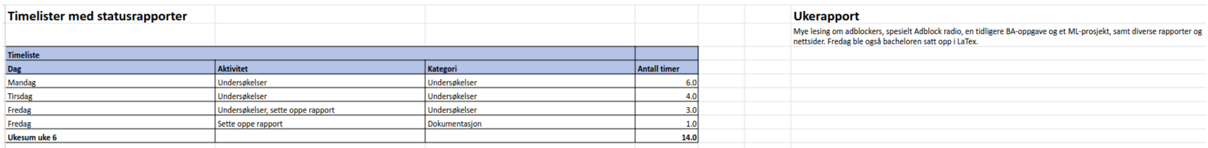
Timelister med statusrapporter

Timeliste			
Dag	**Aktivitet**	**Kategori**	**Antall timer**
Torsdag	Reviderte møtereferat	Dokumentasjon	0.5
Torsdag	Vitenskapelighet i BA-oppgaven del 3	Forelesing	1.0
Fredag	Undersøkelser	Undersøkelser	6.0
Ukesum uke 5			**7.5**

Ukerapport

Var syk tirsdag-fredag, revidert referatet fra oppstartsmøtet og lest gjennom Vitenskapelighet i BA-oppgaven del 3, samt undersøkelser/lesing om fagfeltet og eksisterende løsninger

Uke 6



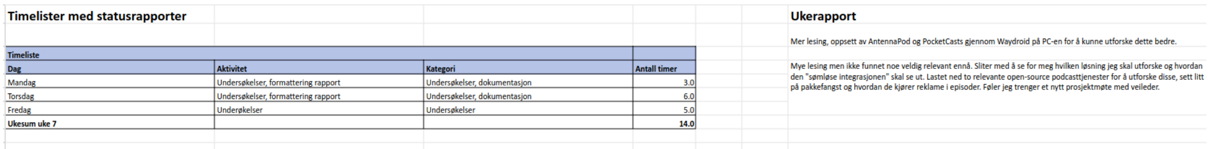
Timelister med statusrapporter

Timeliste			
Dag	**Aktivitet**	**Kategori**	**Antall timer**
Mandag	Undersøkelser	Undersøkelser	6.0
Tirsdag	Undersøkelser	Undersøkelser	4.0
Fredag	Undersøkelser, sette oppe rapport	Undersøkelser	3.0
Fredag	Sette oppe rapport	Dokumentasjon	1.0
Ukesum uke 6			**14.0**

Ukerapport

Mye lesing om adblockers, spesielt Adblock radio, en tidligere BA-oppgave og et ML-prosjekt, samt diverse rapporter og nettsider. Fredag ble også bacheloren satt opp i LaTex.

Uke 7



Timelister med statusrapporter

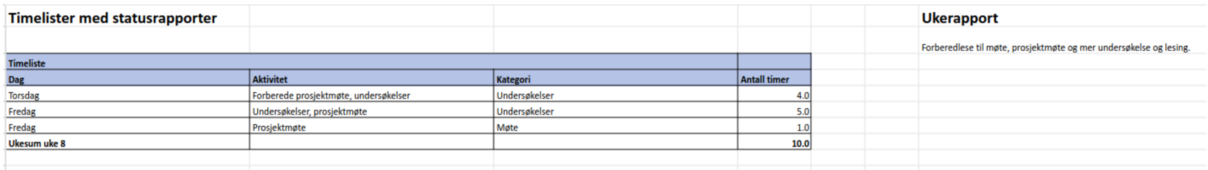
Timeliste			
Dag	**Aktivitet**	**Kategori**	**Antall timer**
Mandag	Undersøkelser, formattering rapport	Undersøkelser, dokumentasjon	3.0
Torsdag	Undersøkelser, formattering rapport	Undersøkelser, dokumentasjon	6.0
Fredag	Underøkelser	Undersøkelser	5.0
Ukesum uke 7			**14.0**

Ukerapport

Mer lesing, oppsett av AntennaPod og PocketCasts gjennom Waydroid på PC-en for å kunne utforske dette bedre.

Mye lesing men ikke funnet noe veldig relevant ennå. Sliter med å se for meg hvilken løsning jeg skal utforske og hvordan den "sømløse integrasjonen" skal se ut. Lastet ned to relevante open-source podcasttjenester for å utforske disse, sett litt på pakkefangst og hvordan de kjører reklame i episoder. Føler jeg trenger et nytt prosjektmøte med veileder.

Uke 8



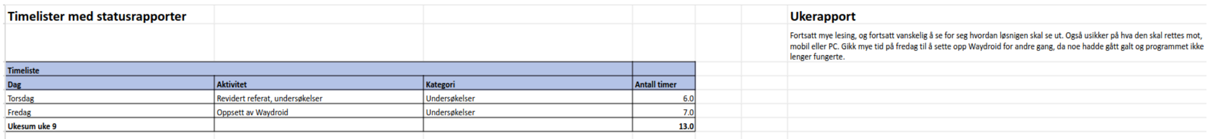
Timelister med statusrapporter

Timeliste			
Dag	**Aktivitet**	**Kategori**	**Antall timer**
Torsdag	Forberede prosjektmøte, undersøkelser	Undersøkelser	4.0
Fredag	Undersøkelser, prosjektmøte	Undersøkelser	5.0
Fredag	Prosjektmøte	Møte	1.0
Ukesum uke 8			**10.0**

Ukerapport

Forberedlese til møte, prosjektmøte og mer undersøkelse og lesing.

Uke 9



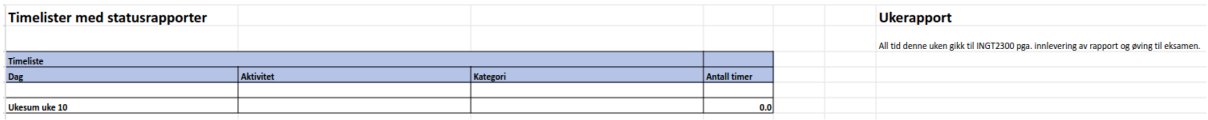
Timelister med statusrapporter

Timeliste			
Dag	**Aktivitet**	**Kategori**	**Antall timer**
Torsdag	Revidert referat, undersøkelser	Undersøkelser	6.0
Fredag	Oppsett av Waydroid	Undersøkelser	7.0
Ukesum uke 9			**13.0**

Ukerapport

Fortsatt mye lesing, og fortsatt vanskelig å se for seg hvordan løsnigen skal se ut. Også usikker på hva den skal rettes mot, mobil eller PC. Gikk mye tid på fredag til å sette opp Waydroid for andre gang, da noe hadde gått galt og programmet ikke lenger fungerte.

Uke 10



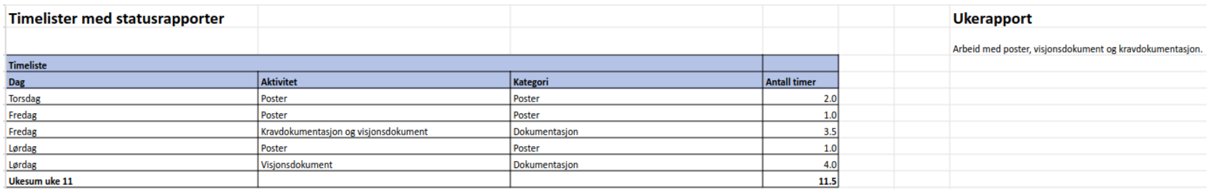
Timelister med statusrapporter

Timeliste			
Dag	**Aktivitet**	**Kategori**	**Antall timer**
Ukesum uke 10			**0.0**

Ukerapport

All tid denne uken gikk til INGT2300 pga. innlevering av rapport og øving til eksamen.

Uke 11



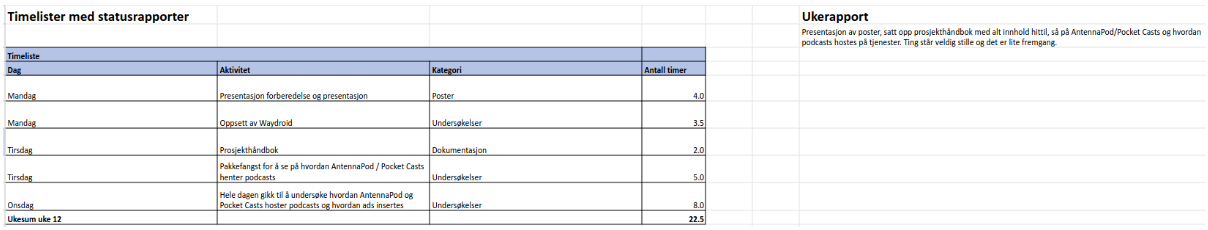
Timelister med statusrapporter

Timeliste			
Dag	**Aktivitet**	**Kategori**	**Antall timer**
Torsdag	Poster	Poster	2.0
Fredag	Poster	Poster	1.0
Fredag	Kravdokumentasjon og visjonsdokument	Dokumentasjon	3.5
Lørdag	Poster	Poster	1.0
Lørdag	Visjonsdokument	Dokumentasjon	4.0
Ukesum uke 11			**11.5**

Ukerapport

Arbeid med poster, visjonsdokument og kravdokumentasjon.

Uke 12



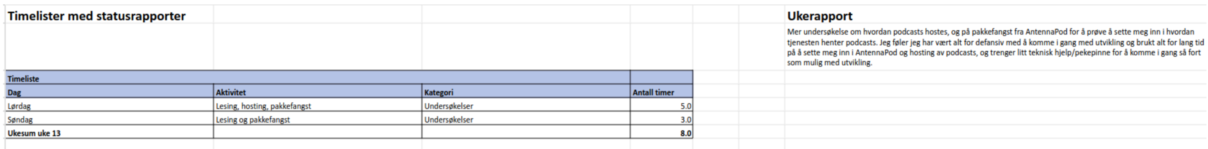
Timelister med statusrapporter

Timeliste			
Dag	**Aktivitet**	**Kategori**	**Antall timer**
Mandag	Presentasjon forberedelse og presentasjon	Poster	4.0
Mandag	Oppsett av Waydroid	Undersøkelser	3.5
Tirsdag	Prosjekthåndbok	Dokumentasjon	2.0
Tirsdag	Pakkefangst for å se på hvordan AntennaPod / Pocket Casts henter podcasts	Undersøkelser	5.0
Onsdag	Hele dagen gikk til å undersøke hvordan AntennaPod og Pocket Casts hoster podcasts og hvordan ads insertes	Undersøkelser	8.0
Ukesum uke 12			**22.5**

Ukerapport

Presentasjon av poster, satt opp prosjekthåndbok med alt innhold hittil, så på AntennaPod/Pocket Casts og hvordan podcasts hostes på tjenester. Ting står veldig stille og det er lite fremgang.

Uke 13



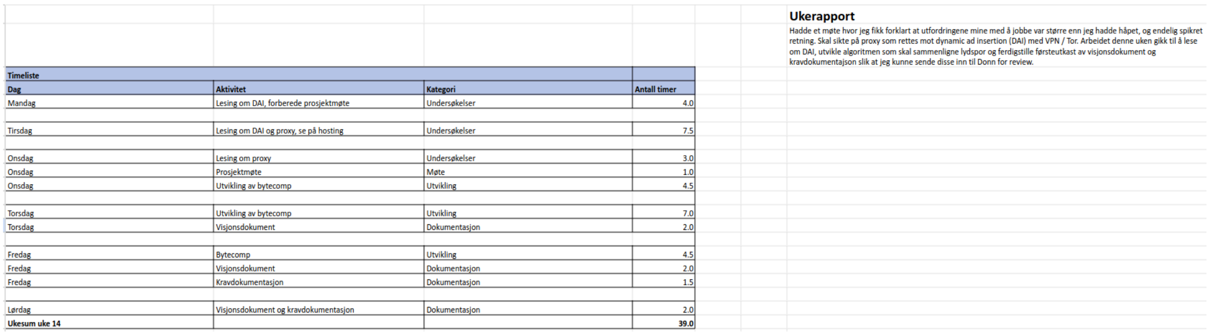
Timelister med statusrapporter

Timeliste			
Dag	**Aktivitet**	**Kategori**	**Antall timer**
Lørdag	Lesing, hosting, pakkefangst	Undersøkelser	5.0
Søndag	Lesing og pakkefangst	Undersøkelser	3.0
Ukesum uke 13			**8.0**

Ukerapport

Mer undersøkelse om hvordan podcasts hostes, og på pakkefangst fra AntennaPod for å prøve å sette meg inn i hvordan tjenesten henter podcasts. Jeg føler jeg har vært alt for defansiv med å komme i gang med utvikling og brukt alt for lang tid på å sette meg inn i AntennaPod og hosting av podcasts, og trenger litt teknisk hjelp/pekepinne for å komme i gang så fort som mulig med utvikling.

Uke 14

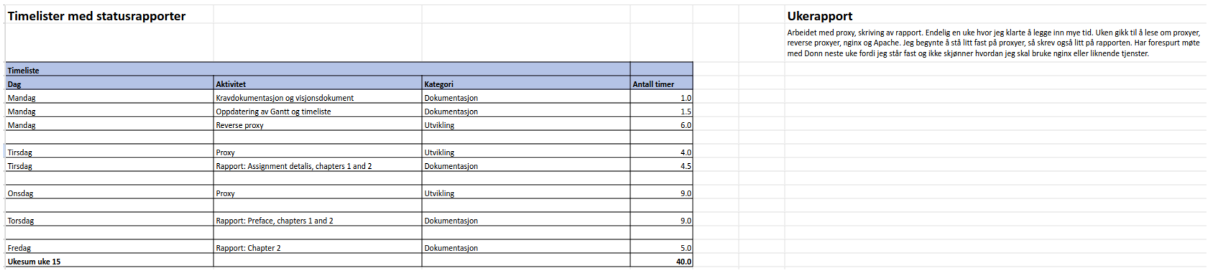


Timeliste			
Dag	**Aktivitet**	**Kategori**	**Antall timer**
Mandag	Lesing om DAI, forberede prosjektmøte	Undersøkelser	4.0
Tirsdag	Lesing om DAI og proxy, se på hosting	Undersøkelser	7.5
Onsdag	Lesing om proxy	Undersøkelser	3.0
Onsdag	Prosjektmøte	Møte	1.0
Onsdag	Utvikling av bytecomp	Utvikling	4.5
Torsdag	Utvikling av bytecomp	Utvikling	7.0
Torsdag	Visjonsdokument	Dokumentasjon	2.0
Fredag	Bytecomp	Utvikling	4.5
Fredag	Visjonsdokument	Dokumentasjon	2.0
Fredag	Kravdokumentasjon	Dokumentasjon	1.5
Lørdag	Visjonsdokument og kravdokumentasjon	Dokumentasjon	2.0
Ukesum uke 14			**39.0**

Ukerapport

Hadde et møte hvor jeg fikk forklart at utfordringene mine med å jobbe var større enn jeg hadde håpet, og endelig spikret retning. Skal sikte på proxy som rettes mot dynamic ad insertion (DAI) med VPN / Tor. Arbeidet denne uken gikk til å lese om DAI, utvikle algoritmen som skal sammenligne lydspor og ferdigstille førsteutkast av visjonsdokument og kravdokumentasjon slik at jeg kunne sende disse inn til Donn for review.

Uke 15



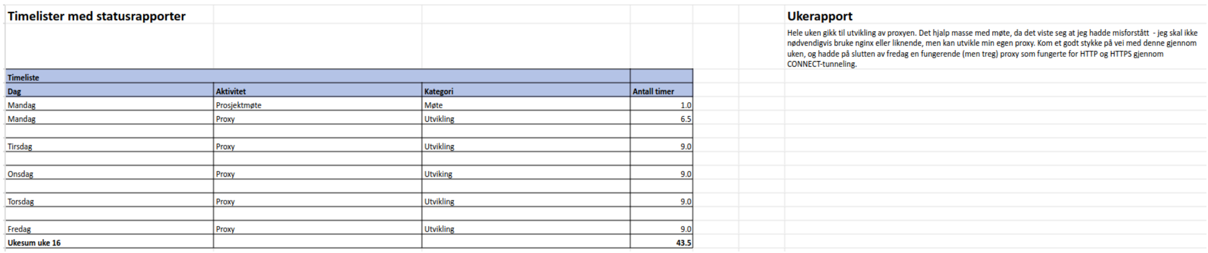
Timelister med statusrapporter

Timeliste			
Dag	**Aktivitet**	**Kategori**	**Antall timer**
Mandag	Kravdokumentasjon og visjonsdokument	Dokumentasjon	1.0
Mandag	Oppdatering av Gantt og timeliste	Dokumentasjon	1.5
Mandag	Reverse proxy	Utvikling	6.0
Tirsdag	Proxy	Utvikling	4.0
Tirsdag	Rapport: Assignment detalis, chapters 1 and 2	Dokumentasjon	4.5
Onsdag	Proxy	Utvikling	9.0
Torsdag	Rapport: Preface, chapters 1 and 2	Dokumentasjon	9.0
Fredag	Rapport: Chapter 2	Dokumentasjon	5.0
Ukesum uke 15			**40.0**

Ukerapport

Arbeidet med proxy, skriving av rapport. Endelig en uke hvor jeg klarte å legge inn mye tid. Uken gikk til å lese om proxyer, reverse proxyer, nginx og Apache. Jeg begynte å stå litt fast på proxyer, så skrev også litt på rapporten. Har forespurt møte med Donn neste uke fordi jeg står fast og ikke skjønner hvordan jeg skal bruke nginx eller liknende tjenster.

Uke 16



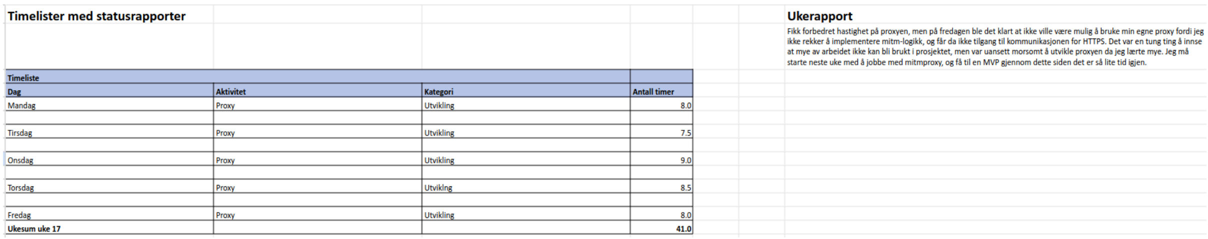
Timelister med statusrapporter

Timeliste			
Dag	**Aktivitet**	**Kategori**	**Antall timer**
Mandag	Prosjektmøte	Møte	1.0
Mandag	Proxy	Utvikling	6.5
Tirsdag	Proxy	Utvikling	9.0
Onsdag	Proxy	Utviking	9.0
Torsdag	Proxy	Utvikling	9.0
Fredag	Proxy	Utvikling	9.0
Ukesum uke 16			**43.5**

Ukerapport

Hele uken gikk til utvikling av proxyen. Det hjalp masse med møte, da det viste seg at jeg hadde misforstått - jeg skal ikke nødvendigvis bruke nginx eller liknende, men kan utvikle min egen proxy. Kom et godt stykke på vei med denne gjennom uken, og hadde på slutten av fredag en fungerende (men treg) proxy som fungerte for HTTP og HTTPS gjennom CONNECT-tunneling.

Uke 17



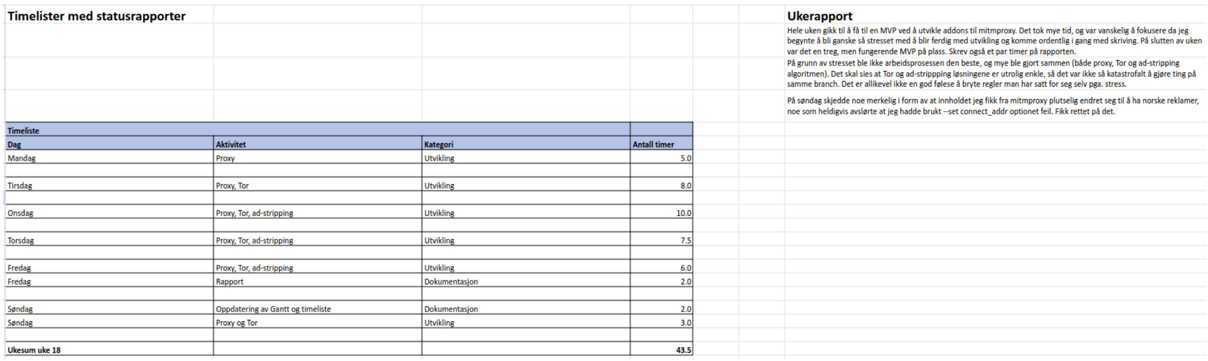
Timelister med statusrapporter

Timeliste			
Dag	**Aktivitet**	**Kategori**	**Antall timer**
Mandag	Proxy	Utvikling	8.0
Tirsdag	Proxy	Utvikling	7.5
Onsdag	Proxy	Utvikling	9.0
Torsdag	Proxy	Utviklng	8.5
Fredag	Proxy	Utvikling	8.0
Ukesum uke 17			**41.0**

Ukerapport

Fikk forbedret hastighet på proxyen, men på fredagen ble det klart at ikke ville være mulig å bruke min egne proxy fordi jeg ikke rekker å implementere mitm-logikk, og får da ikke tilgang til kommunikasjonen for HTTPS. Det var en tung ting å innse at mye av arbeidet ikke kan bli brukt i prosjektet, men var uansett morsomt å utvikle proxyen da jeg lærte mye. Jeg må starte neste uke med å jobbe med mitmproxy, og få til en MVP gjennom dette siden det er så lite tid igjen.

Uke 18



Timelister med statusrapporter

Timeliste			
Dag	**Aktivitet**	**Kategori**	**Antall timer**
Mandag	Proxy	Utvikling	5.0
Tirsdag	Proxy, Tor	Utvikling	8.0
Onsdag	Proxy, Tor, ad-stripping	Utvikling	10.0
Torsdag	Proxy, Tor, ad-stripping	Utvikling	7.5
Fredag	Proxy, Tor, ad-stripping	Utvikling	6.0
Fredag	Rapport	Dokumentasjon	2.0
Søndag	Oppdatering av Gantt og timeliste	Dokumentasjon	2.0
Søndag	Proxy og Tor	Utvikling	3.0
Ukesum uke 18			**43.5**

Ukerapport

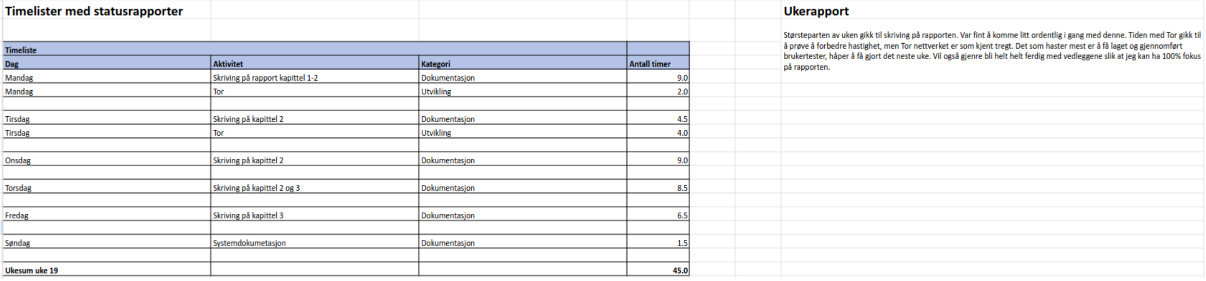
Hele uken gikk til å få til en MVP ved å utvikle addons til mitmproxy. Det tok mye tid, og var vanskelig å fokusere da jeg begynte å bli ganske så stresset med å blir ferdig med utvikling og komme ordentlig i gang med skriving. På slutten av uken var det en treg, men fungerende MVP på plass. Skrev også et par timer på rapporten.

På grunn av stresset ble ikke arbeidsprosessen den beste, og mye ble gjort sammen (både proxy, Tor og ad-stripping algoritmen). Det skal sies at Tor og ad-strippping løsningene er utrolig enkle, så det var ikke så katastrofalt å gjøre ting på samme branch. Det er allikevel ikke en god følese å bryte regler man har satt for seg selv pga. stress.

På søndag skjedde noe merkelig i form av at innholdet jeg fikk fra mitmproxy plutselig endret seg til å ha norske reklamer, noe som heldigvis avslørte at jeg hadde brukt --set connect_addr optionet feil. Fikk rettet på det.

025

Uke 19



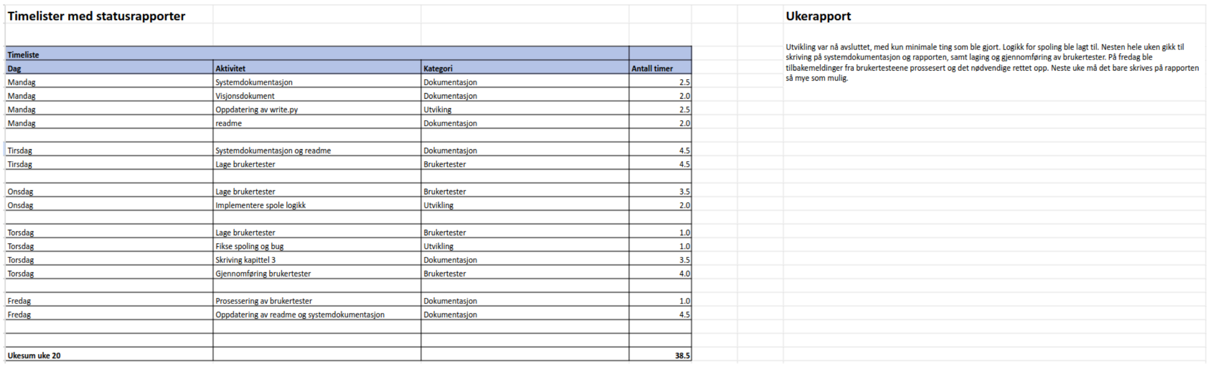
Timelister med statusrapporter

Timeliste			
Dag	**Aktivitet**	**Kategori**	**Antall timer**
Mandag	Skriving på rapport kapittel 1-2	Dokumentasjon	9.0
Mandag	Tor	Utvikling	2.0
Tirsdag	Skriving på kapittel 2	Dokumentasjon	4.5
Tirsdag	Tor	Utvikling	4.0
Onsdag	Skriving på kapittel 2	Dokumentasjon	9.0
Torsdag	Skriving på kapittel 2 og 3	Dokumentasjon	8.5
Fredag	Skriving på kapittel 3	Dokumentasjon	6.5
Søndag	Systemdokumetasjon	Dokumentasjon	1.5
Ukesum uke 19			**45.0**

Ukerapport

Størsteparten av uken gikk til skriving på rapporten. Var fint å komme litt ordentlig i gang med denne. Tiden med Tor gikk til å prøve å forbedre hastighet, men Tor nettverket er som kjent tregt. Det som haster mest er å få laget og gjennomført brukertester, håper å få gjort det neste uke. Vil også gjenre bli helt helt ferdig med vedleggene slik at jeg kan ha 100% fokus på rapporten.

Uke 20



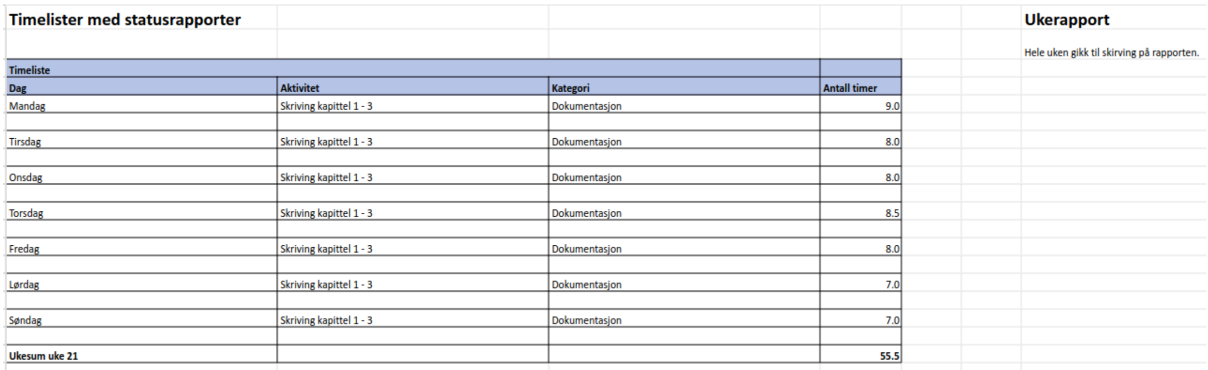
Timelister med statusrapporter

Timeliste			
Dag	**Aktivitet**	**Kategori**	**Antall timer**
Mandag	Systemdokumentasjon	Dokumentasjon	2.5
Mandag	Visjonsdokument	Dokumentasjon	2.0
Mandag	Oppdatering av write.py	Utviking	2.5
Mandag	readme	Dokumentasjon	2.0
Tirsdag	Systemdokumentasjon og readme	Dokumentasjon	4.5
Tirsdag	Lage brukertester	Brukertester	4.5
Onsdag	Lage brukertester	Brukertester	3.5
Onsdag	Implementere spole logikk	Utvikling	2.0
Torsdag	Lage brukertester	Brukertester	1.0
Torsdag	Fikse spoling og bug	Utvikling	1.0
Torsdag	Skriving kapittel 3	Dokumentasjon	3.5
Torsdag	Gjennomføring brukertester	Brukertester	4.0
Fredag	Prosessering av brukertester	Dokumentasjon	1.0
Fredag	Oppdatering av readme og systemdokumentasjon	Dokumentasjon	4.5
Ukesum uke 20			**38.5**

Ukerapport

Utvikling var nå avsluttet, med kun minimale ting som ble gjort. Logikk for spoling ble lagt til. Nesten hele uken gikk til skriving på systemdokumentasjon og rapporten, samt laging og gjennomføring av brukertester. På fredag ble tilbakemeldinger fra brukertesteene prossesert og det nødvendige rettet opp. Neste uke må det bare skrives på rapporten så mye som mulig.

Uke 21



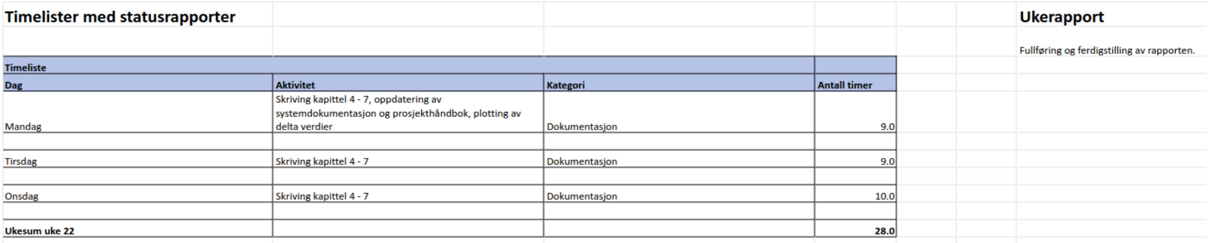
Timelister med statusrapporter

Timeliste			
Dag	**Aktivitet**	**Kategori**	**Antall timer**
Mandag	Skriving kapittel 1 - 3	Dokumentasjon	9.0
Tirsdag	Skriving kapittel 1 - 3	Dokumentasjon	8.0
Onsdag	Skriving kapittel 1 - 3	Dokumentasjon	8.0
Torsdag	Skriving kapittel 1 - 3	Dokumentasjon	8.5
Fredag	Skriving kapittel 1 - 3	Dokumentasjon	8.0
Lørdag	Skriving kapittel 1 - 3	Dokumentasjon	7.0
Søndag	Skriving kapittel 1 - 3	Dokumentasjon	7.0
Ukesum uke 21			**55.5**

Ukerapport

Hele uken gikk til skirving på rapporten.

Uke 22



Timelister med statusrapporter

Timeliste			
Dag	**Aktivitet**	**Kategori**	**Antall timer**
Mandag	Skriving kapittel 4 - 7, oppdatering av systemdokumentasjon og prosjekthåndbok, plotting av delta verdier	Dokumentasjon	9.0
Tirsdag	Skriving kapittel 4 - 7	Dokumentasjon	9.0
Onsdag	Skriving kapittel 4 - 7	Dokumentasjon	10.0
Ukesum uke 22			**28.0**

Ukerapport

Fullføring og ferdigstilling av rapporten.

F. System Documentation

The system documentation in form of a PDF.

025

Adblock for podcasts
Systemdokumentasjon

Versjon 1.0

Revisjonshistorie

Dato	**Versjon**	**Beskrivelse**
12/05/2024	0.1	Oppsett, tilpasning av mal, arbeid med punkt 1
13/05/2024	0.2	Arbeid med alle punkter
14/05/2024	0.3	Arbeid med punkt 5
17/05/2024	0.4	Oppdatering av punkt 5 etter brukertester
25/05/2024	1.0	Oppdatert Tor med ExcludeExitNodes

Innholdsfortegnelse

1. Introduksjon

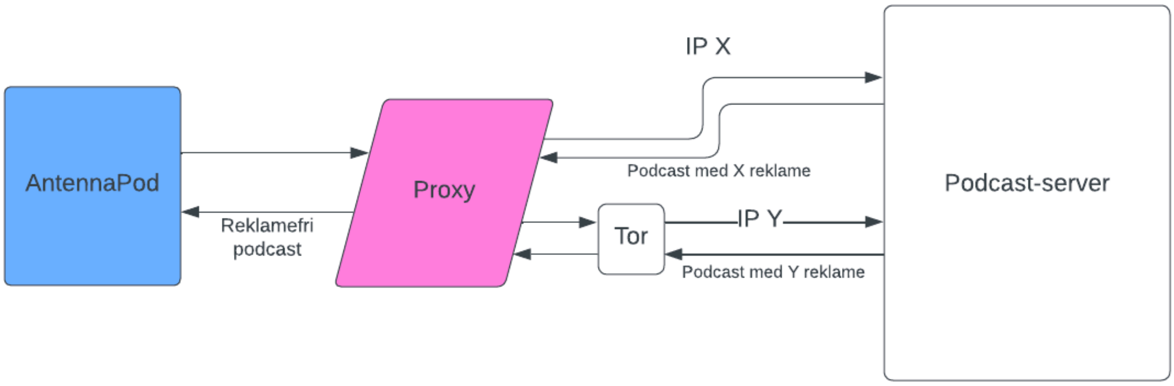
Dokumentet beskriver systemet som er utviklet gjennom prosjektoppgave nr. 025 i emnet IDATT2900 Bacheloroppgave. Hensikten med dokumentet er å beskrive de tekniske aspektene ved systemet, og beskrive hvordan systemet skal settes opp og brukes.

2. Arkitektur

Systemet er en utvidelse til mitmproxy (mitmproxy, u.å.) som fungerer som en adblocker for bruk med AntennaPod (AntennaPod, 2024) gjennom Waydroid (Waydroid, 2022). Systemet henter en lydfil med maskinens lokale IP-adresse og samme lydfil gjennom Tor (The Tor Project Inc., u.å.), og fjerner de ulike delene av lydfilene.

Systemet består av tre deler:

1. Tilpasset proxy-funksjonalitet ved behandling av data som blir sendt fra servere til proxyen
2. En algoritme som fjerner forskjellene mellom to lydfiler.
3. En Tor instans for å hente en lydfil gjennom Tor-nettverket.



Figur 1: Arkitekturen til systemet

Systemet kalles *adblock25*.

3. Prosjektstruktur

mitm.py: Starter adblock25 ved starte starter mitmproxy-instansen og laste inn utvidelsen.

addon.py: Den utviklede utvidelsen til mitmproxy. Lastes inn i mitmproxy-instansen som startes ved å kjøre mitm.py.

write.py: Samme funksjonalitet som addon.py, i tillegg til at lydfiler skrives til fil. Se beskrivelse i 5.3.2.

requirements.txt: Python-avhengighetene for systemet.

proxy.py: Utdatert egenutviklet proxy som opprinnelig var tenkt til å brukes i systemet.

4. Sikkerhet

All data fra AntennaPod og servere vil gå gjennom proxyen og være leselig for proxyen. Systemet er utelukkende ment for bruk på lokal maskin, og all kommunikasjon mellom AntennaPod og proxyen, og proxyen og servere, er kryptert (gitt at den opprinnelige forespørselen fra AntennaPod er kryptert). Det er kun mens dataen behandles av proxyen at den er lesbar. Under kjøring lagrer systemet de fem siste forespørslene fra AntennaPod med evt. tilhørende episode uten reklame, men dette slettes når kjøringen avsluttes. Systemet tar ikke i mot input.

Proxyen utgjør en redusering i sikkerhet ved å legge til et punkt i kommunikasjonsflyten hvor data mellom AntennaPod og servere er leselig.

Systemet krever manuell registrering av et digitalt sertifikat på den emulerte Android-instansen. Å manuelt installere digitale sertifikater medfører risiko: Ved å operere i sikkerhetssystemet til en enhet kan skadeomfanget bli stort ved feil.

Registrering av det digitale sertifikatet utgjør en sikkerhetsrisiko.

5. Installasjon og kjøring

5.1 Installasjon

Installer adblock25 ved å klone repoet eller laste ned `.zip` fra:

https://github.com/chrisryda/adblock25

5.2 Avhengigheter

Avhengighetene til systemet er:

- Python 3.11 eller nyere
- Pakkene i **`requirements.txt`** filen
- mitmproxy CA-sertifikatet installert på Waydroid-enheten
- Tor SOCKS proxy

5.2.1 Python avhengigheter

mitmproxy: Pythonpakken til mitmproxy.

requests: HTTP(S) bibliotek. Brukes for å hente en lydfil gjennom Tor.

logging: Et bibliotek for å logge til terminalen.

PySocks: Bibliotek for bruk av SOCKS-proxyer. Brukes for å hente en lydfil gjennom Tor.

Python avhengighetene kan installeres ved å kjøre følgende kommando i rotmappen til prosjektet:

```
$ pip install -r requirements.txt
```

Hvis det ikke fungerer å installere PySocks-pakken, kjør følgende kommando i rotmappen til prosjektet:

```
$ pip install -U ‘requests[socks]’
```

5.2.2 Installering av mitmproxy CA-sertifikatet på Waydroid-enheten

Generer CA-sertifikatet ved å kjøre adblock25. Se delkapittel 5.3. Programmet trenger kun kjøre i kort tid, og når beskjeden:

```
HTTP(S) proxy listening at *:<port>
```

Er synlig i terminalen kan programmet avsluttes igjen.

Sertifikatet, med navn `mitmproxy-ca-cert.pem`, ligger nå i mappen `~/.mitmproxy`.
For å installere det på Waydroid-enheten, gjør følgende:

Lag `/system/etc/security/cacerts/` mappen i Wadroid sitt filsystem:

```
$ sudo mkdir -p
/var/lib/waydroid/overlay/system/etc/security/cacerts/
```

Finn hashet til sertifikatet:

```
$ openssl x509 -subject_hash_old -in ~/.mitmproxy/mitmproxy-ca-cert.pem | head -1
```

Dette vil gi output som ligner på: `12example34`

Kopier sertifikatet inn i den lagde mappen, og endre navnet til hashet med `.0` på slutten:

```
$ sudo cp ~/.mitmproxy/mitmproxy-ca-cert.pem
/var/lib/waydroid/overlay/system/etc/security/cacerts/12example34.0
```

Gi sertifikatet nødvendige tillatelser:

```
$ sudo chmod 644 /var/lib/waydroid/overlay/system/etc/security/cacerts/12example34.0
```

Waydroid må muligens restartes for at endringene skal inntre.

5.2.3 Tor proxy

For å åpne en Tor SOCKS proxy, må `torrc` filen redigeres og Tor instansen restartes. Hvor denne filen er plassert avhenger av hvordan Tor er installert på ditt lokale system. 5.2.3.1 Bruk av Tor pakken (AUR) og 5.2.3.2 Bruk av Tor-browser viser to forskjellige alternativer.

Følgende to linjer skal legges til i `torrc` filen:

```
$ SOCKSPort 0.0.0.0:9050
$ ExcludeExitNodes {<cc>}
```

Ved bruk av annen port enn `9050`, må `addon.py` og `write.py` oppdateres med det valgte portnummeret.

`cc` er en 2-bokstavers `ISO3166 Alpha-2` landskode. Se liste på:

https://www.iso.org/obp/ui/#search/code/

For å finne landskoder. Bytt ut `cc` med landskoden til din lokale IP, slik at Tor er garantert ikke å hente lydfiler med samme IP-adresse som din lokale.

5.2.3.1 Alternativ 1: Bruk av Tor pakken (AUR)

Ved bruk av `tor` pakken fra Arch user repository (AUR) ligger `torrc` filen normalt i `/etc/tor`.
Legg til følgende to linjer i `torrc` filen og restart Tor-instansen:

```
$ SOCKSPort 0.0.0.0:9050
$ ExcludeExitNodes {<cc>}
```

Ved bruk av annen port enn `9050`, må `addon.py` og `write.py` oppdateres med det valgte portnummeret.

`cc` er en 2-bokstavers `ISO3166 Alpha-2` landskode. Se liste på:

https://www.iso.org/obp/ui/#search/code/

For å finne landskoder. Bytt ut `cc` med landskoden til din lokale IP, slik at Tor er garantert ikke å hente lydfiler med samme IP-adresse som din lokale.

For å restarte instansen, kjør følgende kommandoer:

```
$ sudo systemctl stop tor

$ sudo systemctl start tor
```

5.2.3.2 Alternativ 2: Bruk av Tor-browser

Ved bruk av Tor-browser ligger `torrc` filen typisk i `Browser/TorBrowser/Data/Tor` i Tor-browser mappen. Legg til følgende linje i `torrc` filen og restart Tor-instansen:

```
$ SOCKSPort 0.0.0.0:9050
$ ExcludeExitNodes {<cc>}
```

Ved bruk av annen port enn `9050`, må `addon.py` og `write.py` oppdateres med det valgte portnummeret.

`cc` er en 2-bokstavers `ISO3166 Alpha-2` landskode. Se liste på:

https://www.iso.org/obp/ui/#search/code/

For å finne landskoder. Bytt ut `cc` med landskoden til din lokale IP, slik at Tor er garantert ikke å hente lydfiler med samme IP-adresse som din lokale.

Restart instansen ved å lukke og åpne browseren.

5.3 Kjøring av adblock25

Kjør systemet ved å kjøre følgende kommando:

```
$ python mitm.py [options]
```

Tilgjengelige options:

Option	Beskrivelse
-h	Vis hjelp og avslutt.
-p `tall`	Setter hvilken port adblock25 bindes til. Default er `8080`.
-a `fil.py`	Setter hvilken utvidelse som lastes inn. Default er `addon.py`.

5.3.1 Tilgjengelige utvidelser

`addon.py:` Fjerner reklame

`write.py:` Fjerner reklame og skriver behandlet lyd til filer.

5.3.2 write.py

Ved å bruke `write.py` får en oversikt over hvilke lydfiler som behandles, hva som fjernes og hva som sendes som svar til AntennaPod. Filene lagres i mappen /`tmp`.

`mitm.mp3:` Filen hentet av mitmproxy med maskinens lokale IP-adresse.

`tor.mp3:` Filen hentet gjennom Tor-nettverket.

`removed.mp3:` Delene av `tor.mp3` som ikke ble med i svaret til AntennaPod

`response.mp3:` Filen som sendes til AntennaPod.

5.4 Sett proxy på AntennaPod

Mens adblock25 kjører, inne på AntennaPod, gå til:

```
Settings → Downloads → Proxy
```

Velg HTTP som type. Fyll inn IP-adressen til Waydroid-enheten og porten adblock25 lytter til (default `8080`).

IP-adressen kan ses ved å kjøre kommandoen under. Adressen er listet under `waydorid0`. Denne er for eksempel `192.168.240.1`.

```
$ ip address show
```

Forsikre deg om at adblock25 kjører. Etter å ha fylt inn IP-adressen til Waydroid-enheten og porten til adblock25, trykk `Test → OK`.

6. Dokumentasjon av kildekode

Kildekoden er tilgjengelig fra:

https://github.com/chrisryda/adblock25

Referanser

AntennaPod (2024) *About*. Tilgjengelig fra https://antennapod.org/about/ (Hentet 14. mai 2024)

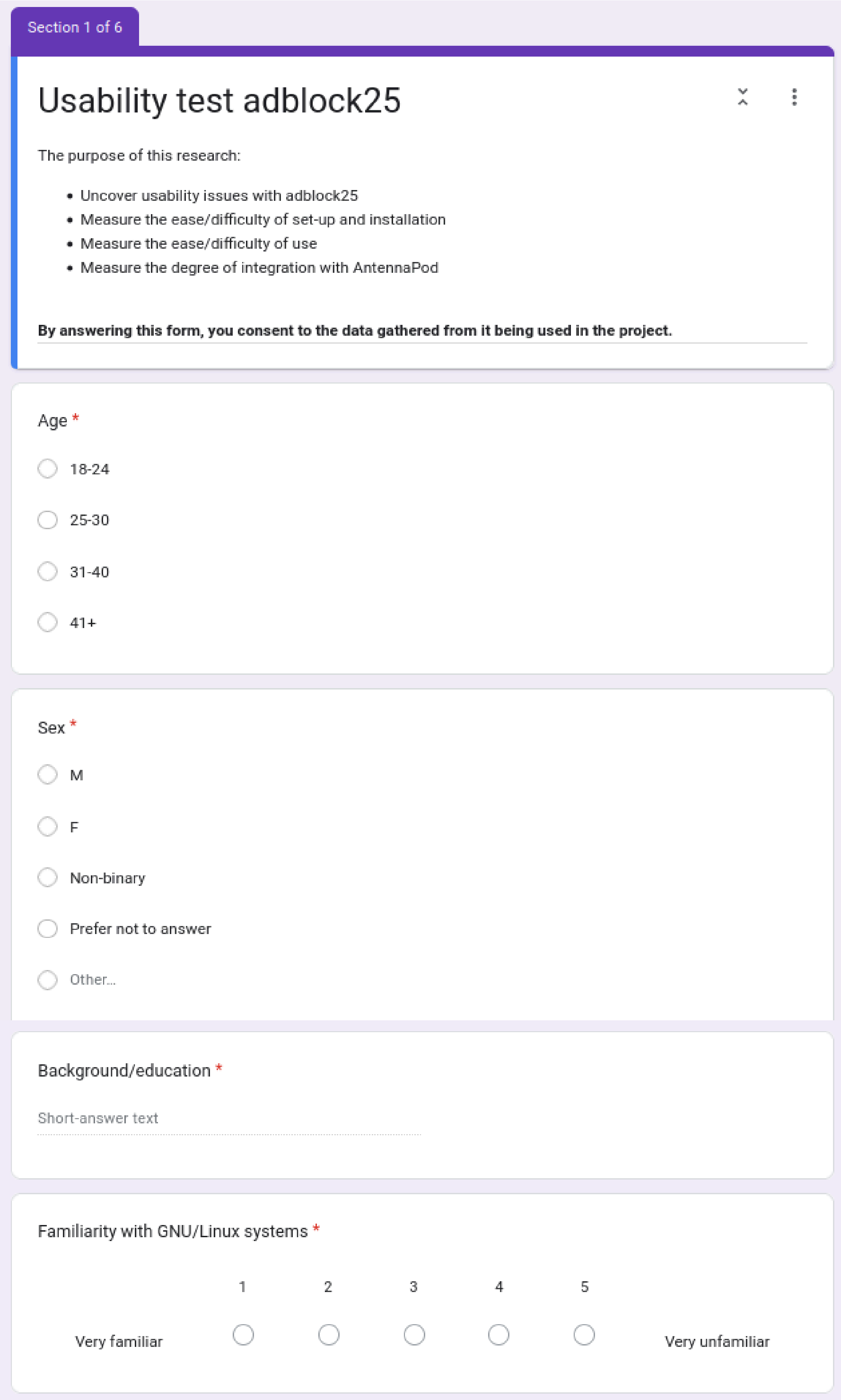
Waydroid (2022) *Waydroid*. Tilgjengelig fra https://waydro.id/ (Hentet 14. mai 2024)

mitmproxy (u.å.) *How mitmproxy works*. Tilgjengelig fra https://docs.mitmproxy.org/stable/concepts-howmitmproxyworks/ (Hentet 14. mai 2024)

The Tor Project Inc. (u.å.). *Browse Privately. Explore Freely*. Tilgjengelig fra https://www.torproject.org/ (Hentet 25. mai 2024)

G. Usability Test Template

The usability test template in the form of a PDF.



Section 1 of 6

Usability test adblock25

The purpose of this research:

- Uncover usability issues with adblock25
- Measure the ease/difficulty of set-up and installation
- Measure the ease/difficulty of use
- Measure the degree of integration with AntennaPod

By answering this form, you consent to the data gathered from it being used in the project.

Age *

○ 18-24

○ 25-30

○ 31-40

○ 41+

Sex *

○ M

○ F

○ Non-binary

○ Prefer not to answer

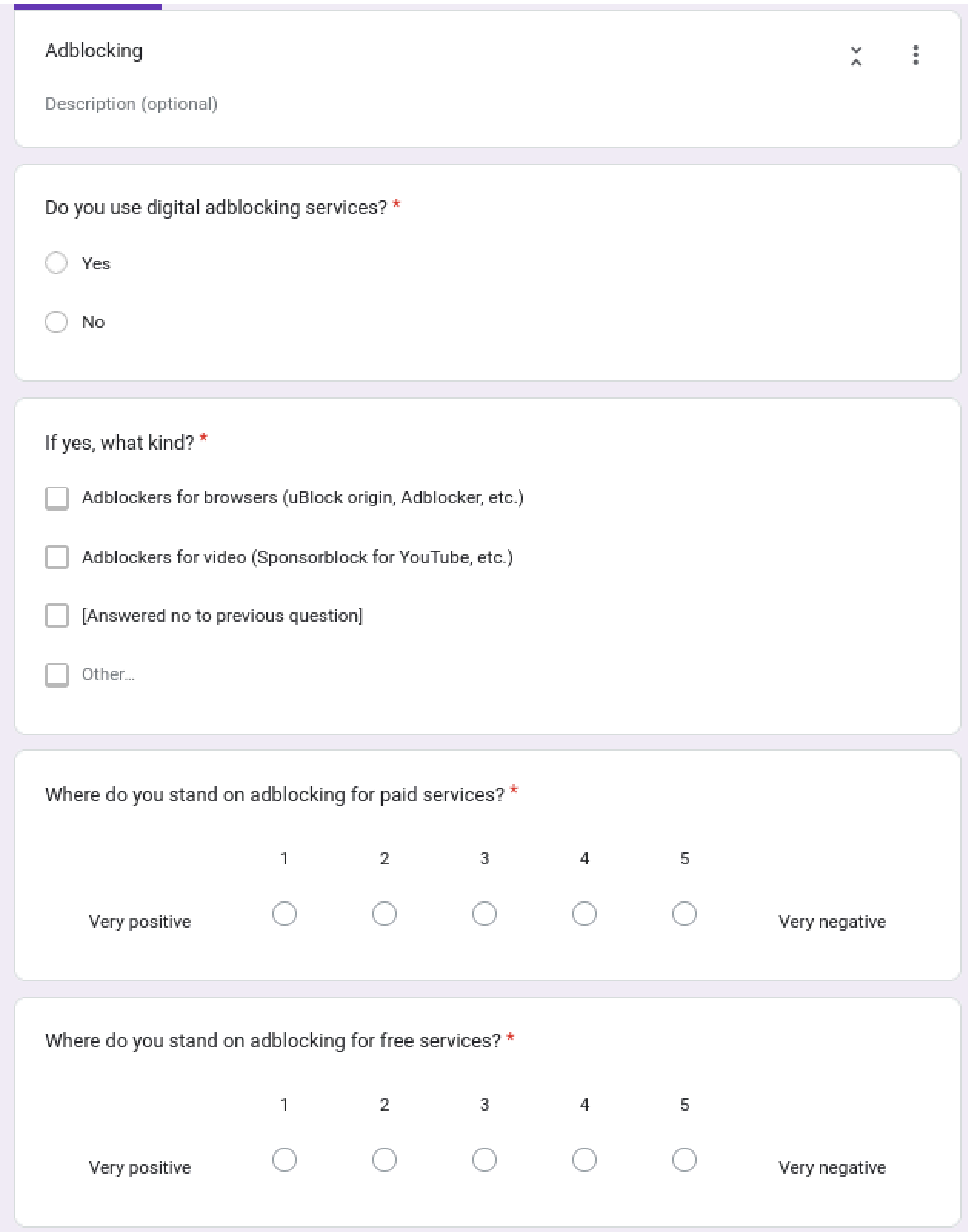
○ Other...

Background/education *

Short-answer text

Familiarity with GNU/Linux systems *

	1	2	3	4	5	
Very familiar	○	○	○	○	○	Very unfamiliar



Adblocking

Description (optional)

Do you use digital adblocking services? *

○ Yes

○ No

If yes, what kind? *

□ Adblockers for browsers (uBlock origin, Adblocker, etc.)

□ Adblockers for video (Sponsorblock for YouTube, etc.)

□ [Answered no to previous question]

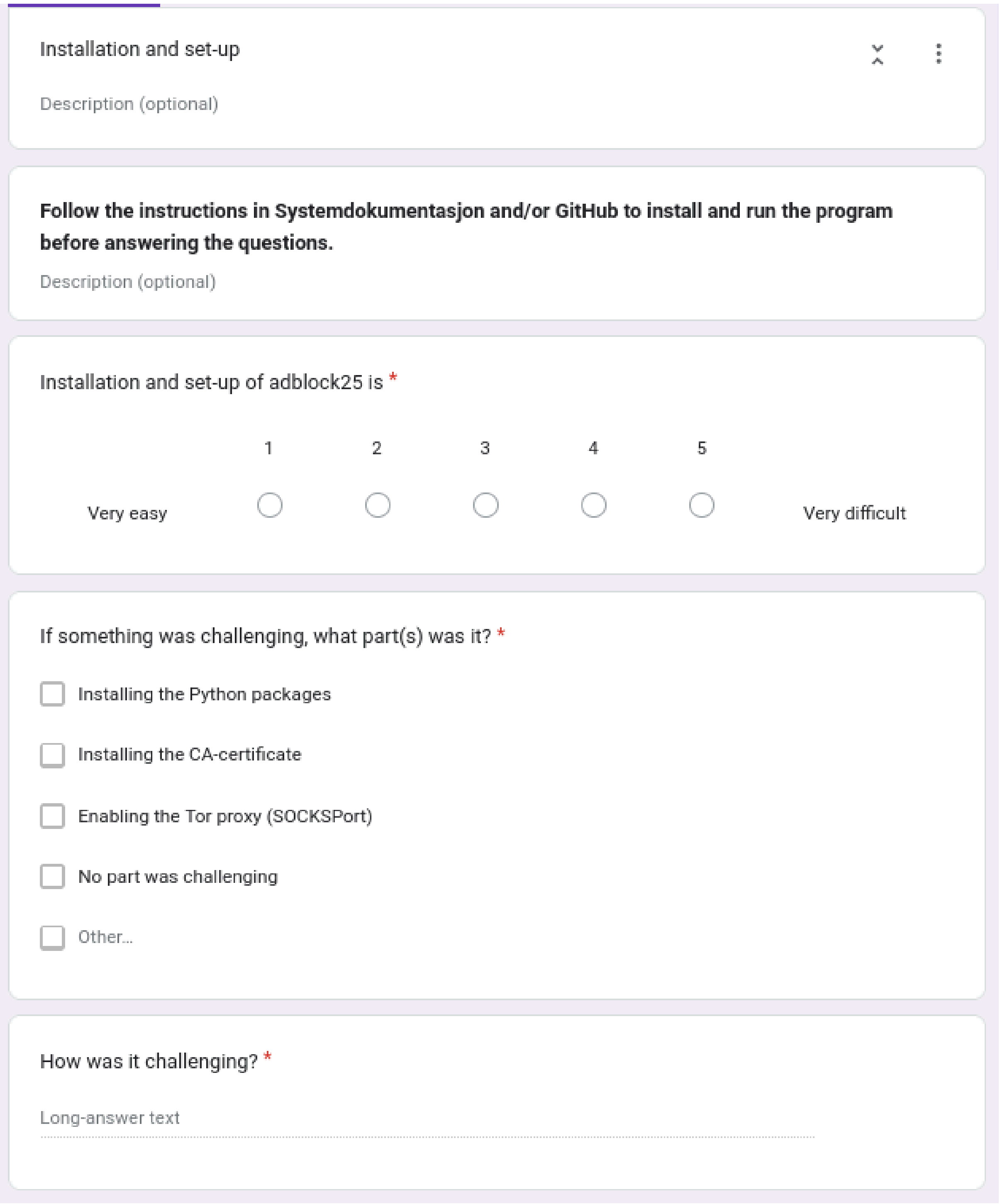
□ Other...

Where do you stand on adblocking for paid services? *

	1	2	3	4	5	
Very positive	○	○	○	○	○	Very negative

Where do you stand on adblocking for free services? *

	1	2	3	4	5	
Very positive	○	○	○	○	○	Very negative



Installation and set-up

Description (optional)

Follow the instructions in Systemdokumentasjon and/or GitHub to install and run the program before answering the questions.

Description (optional)

Installation and set-up of adblock25 is *

	1	2	3	4	5	
Very easy	○	○	○	○	○	Very difficult

If something was challenging, what part(s) was it? *

□ Installing the Python packages

□ Installing the CA-certificate

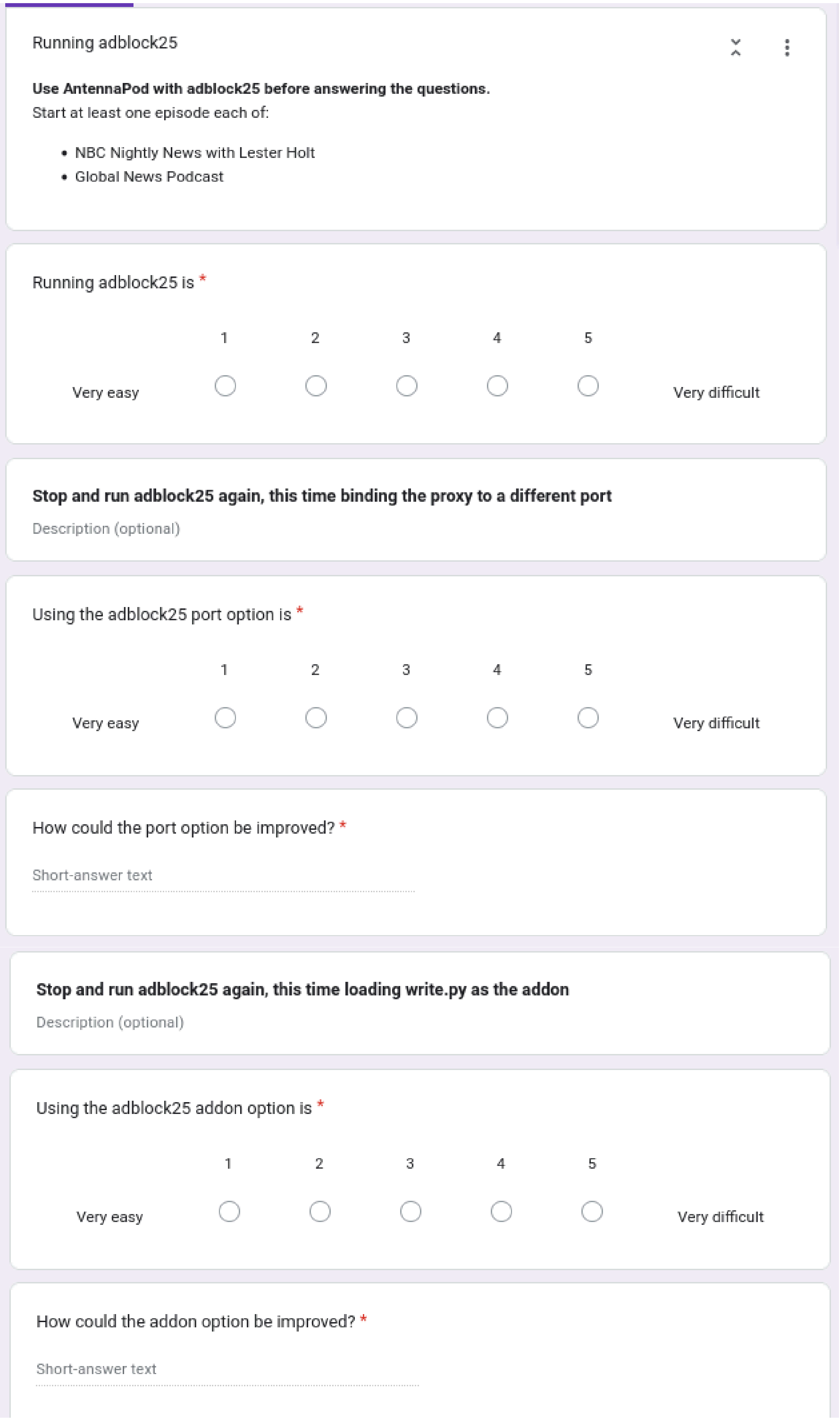
□ Enabling the Tor proxy (SOCKSPort)

□ No part was challenging

□ Other...

How was it challenging? *

Long-answer text



Running adblock25

Use AntennaPod with adblock25 before answering the questions.
Start at least one episode each of:

- NBC Nightly News with Lester Holt
- Global News Podcast

Running adblock25 is *

	1	2	3	4	5	
Very easy	○	○	○	○	○	Very difficult

Stop and run adblock25 again, this time binding the proxy to a different port

Description (optional)

Using the adblock25 port option is *

	1	2	3	4	5	
Very easy	○	○	○	○	○	Very difficult

How could the port option be improved? *

Short-answer text

Stop and run adblock25 again, this time loading write.py as the addon

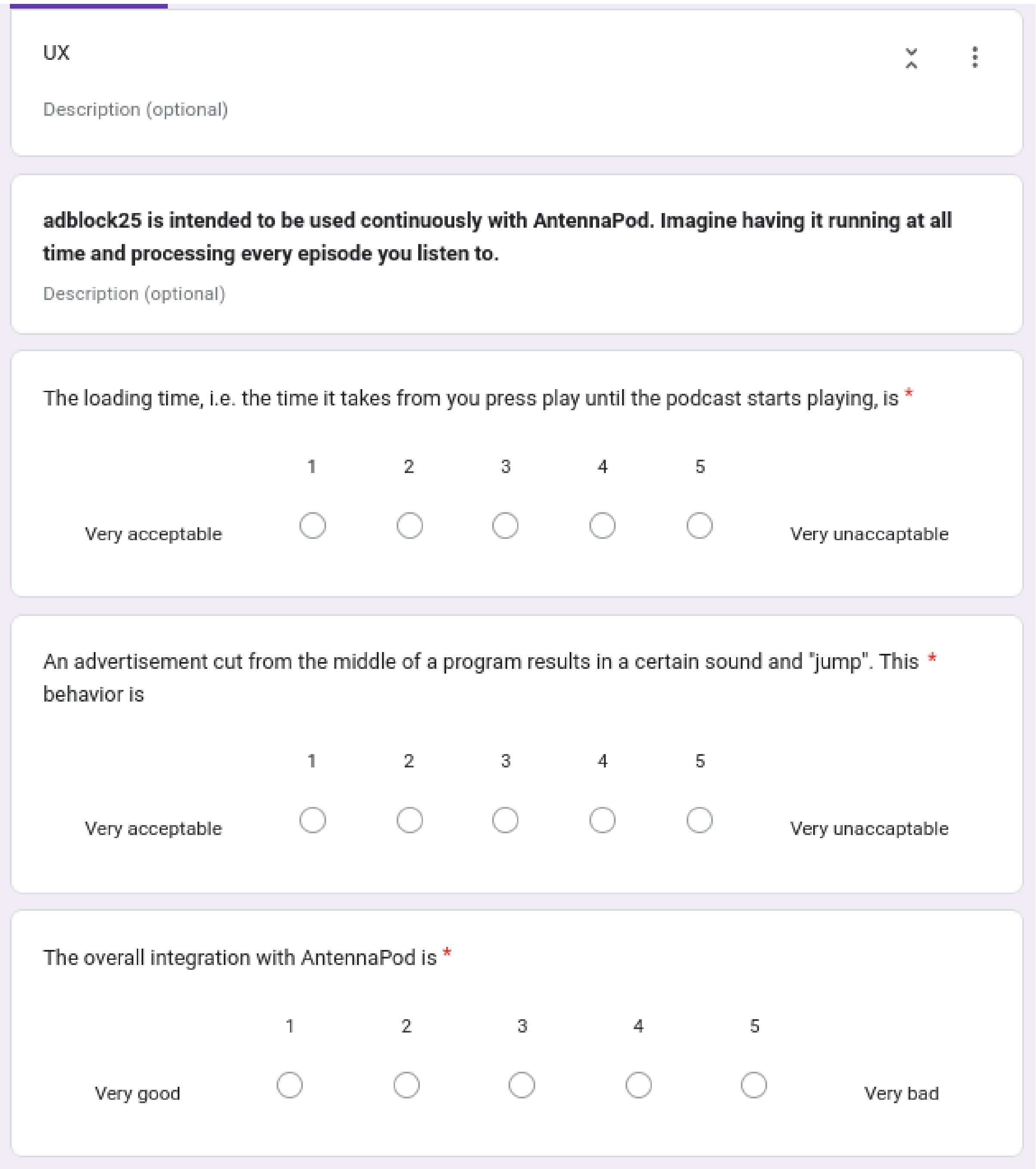
Description (optional)

Using the adblock25 addon option is *

	1	2	3	4	5	
Very easy	○	○	○	○	○	Very difficult

How could the addon option be improved? *

Short-answer text



UX

Description (optional)

adblock25 is intended to be used continuously with AntennaPod. Imagine having it running at all time and processing every episode you listen to.

Description (optional)

The loading time, i.e. the time it takes from you press play until the podcast starts playing, is *

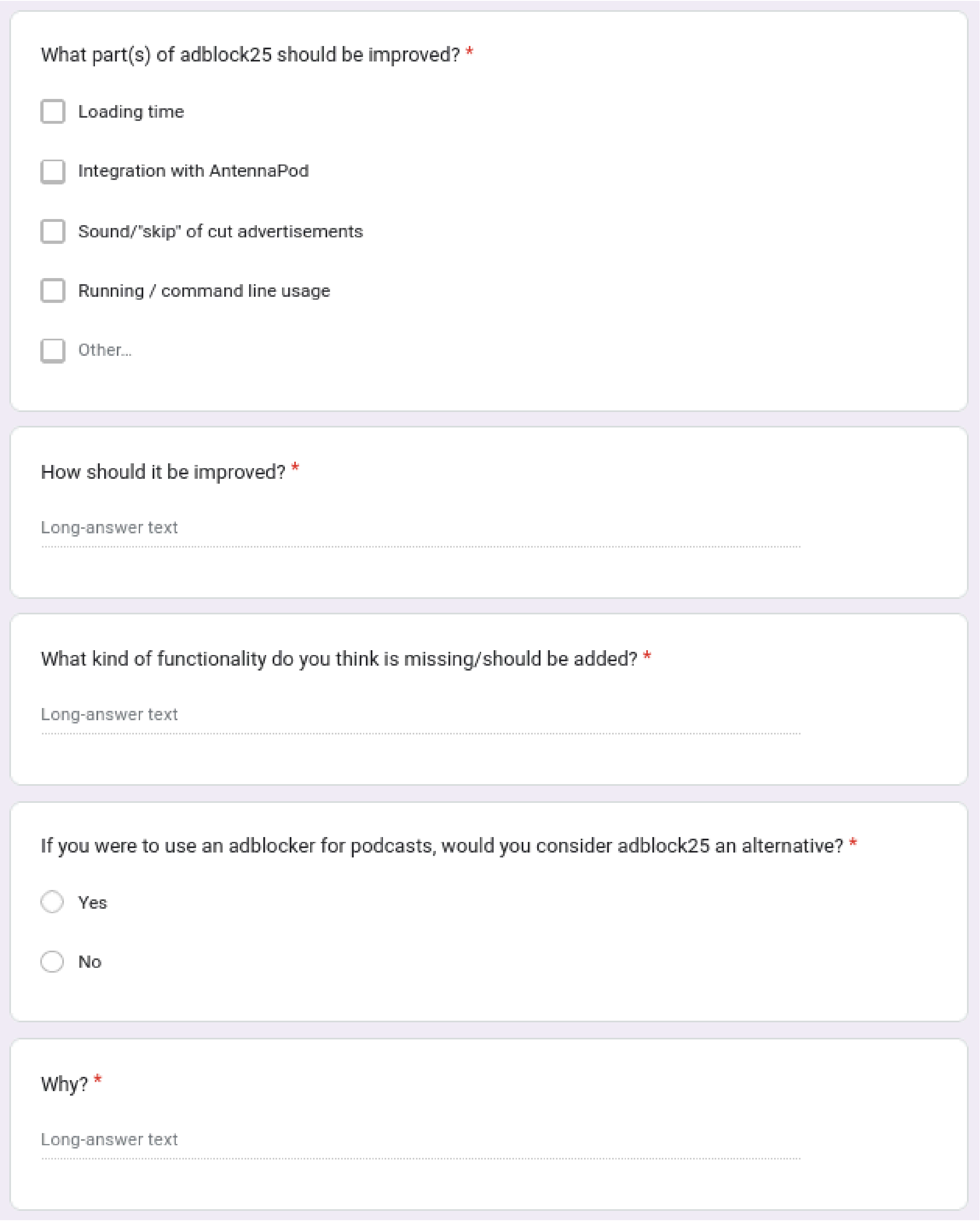
	1	2	3	4	5	
Very acceptable	○	○	○	○	○	Very unaccaptable

An advertisement cut from the middle of a program results in a certain sound and "jump". This * behavior is

	1	2	3	4	5	
Very acceptable	○	○	○	○	○	Very unaccaptable

The overall integration with AntennaPod is *

	1	2	3	4	5	
Very good	○	○	○	○	○	Very bad



What part(s) of adblock25 should be improved? *

□ Loading time

□ Integration with AntennaPod

□ Sound/"skip" of cut advertisements

□ Running / command line usage

□ Other...

How should it be improved? *

Long-answer text

What kind of functionality do you think is missing/should be added? *

Long-answer text

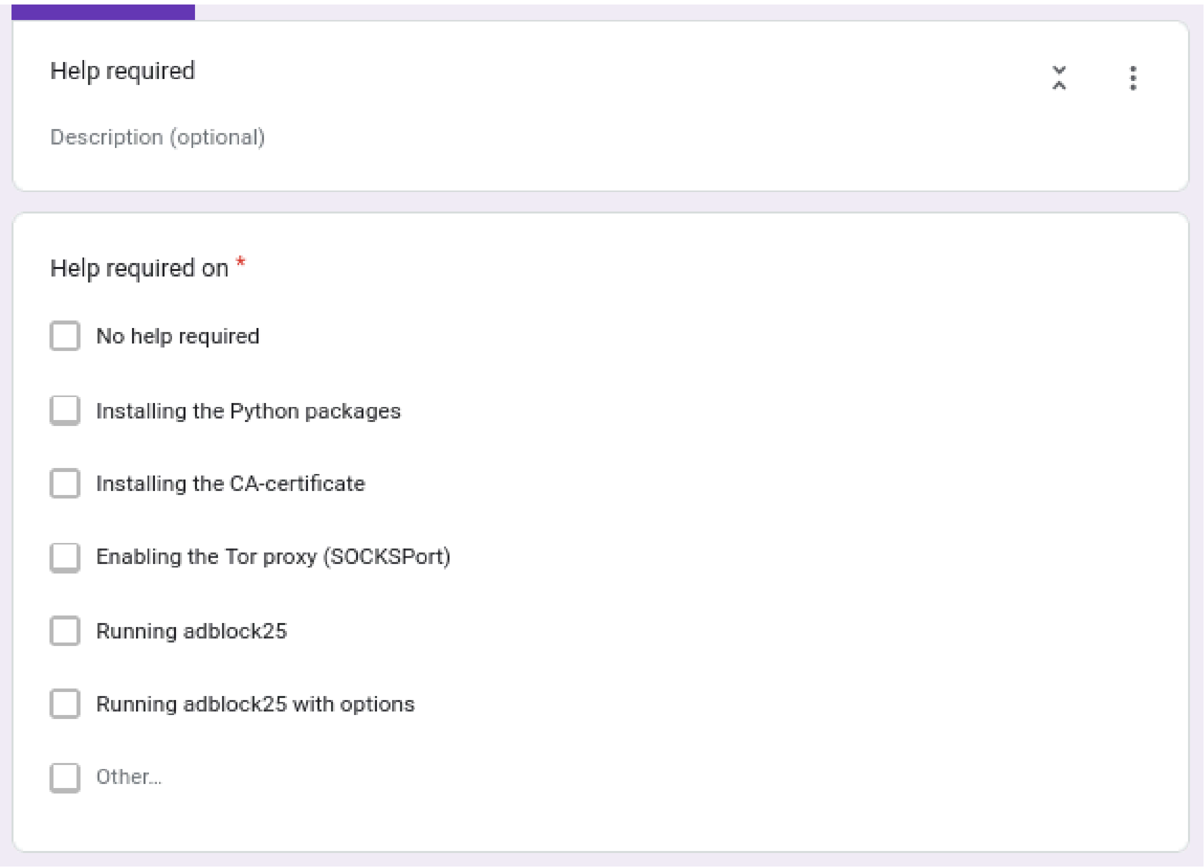
If you were to use an adblocker for podcasts, would you consider adblock25 an alternative? *

○ Yes

○ No

Why? *

Long-answer text



Help required

Description (optional)

Help required on *

□ No help required

□ Installing the Python packages

□ Installing the CA-certificate

□ Enabling the Tor proxy (SOCKSPort)

□ Running adblock25

□ Running adblock25 with options

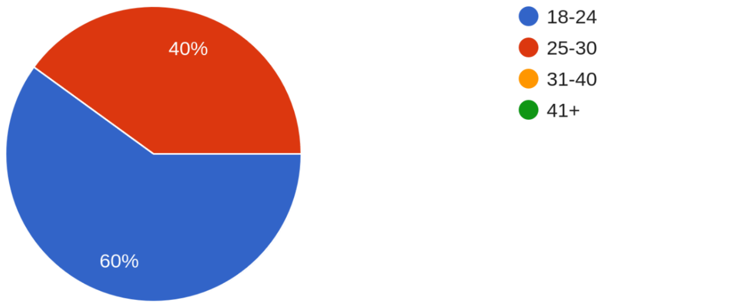
□ Other...

H. Usability Test Responses

The usability test responses in the form of a PDF.

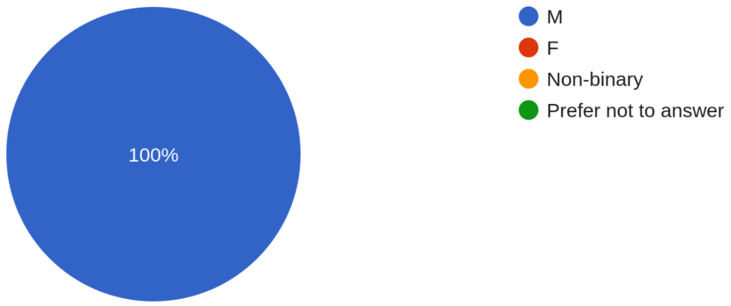
Age

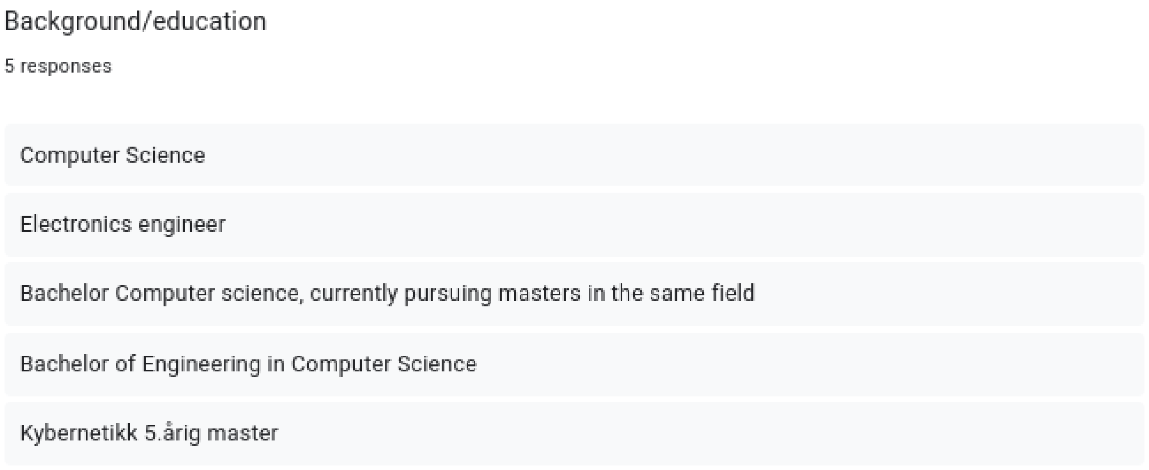
5 responses



Sex

5 responses





Background/education

5 responses

Computer Science

Electronics engineer

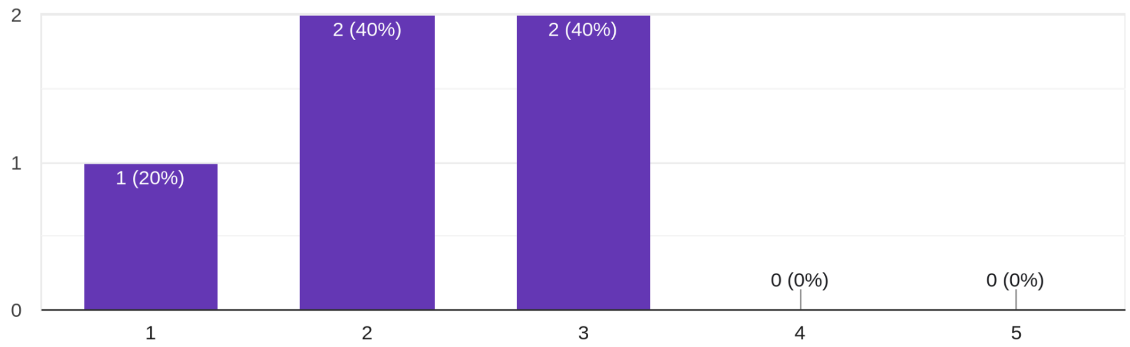
Bachelor Computer science, currently pursuing masters in the same field

Bachelor of Engineering in Computer Science

Kybernetikk 5.årig master

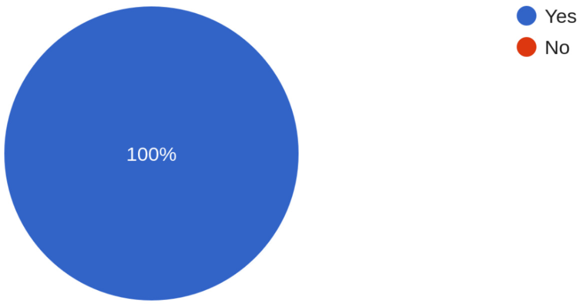
Familiarity with GNU/Linux systems

5 responses



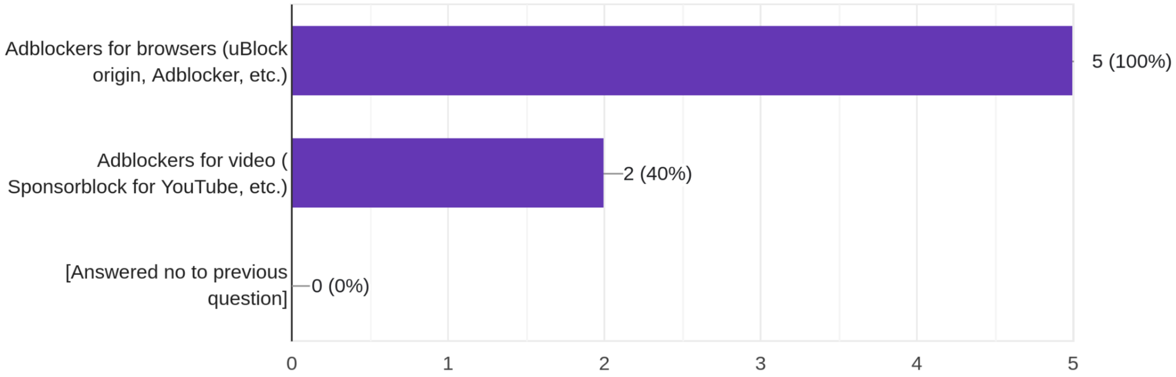
Do you use digital adblocking services?

5 responses



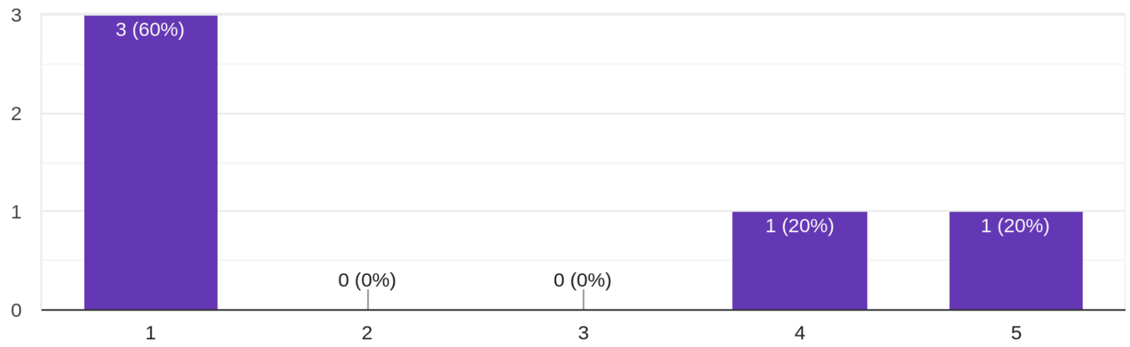
If yes, what kind?

5 responses



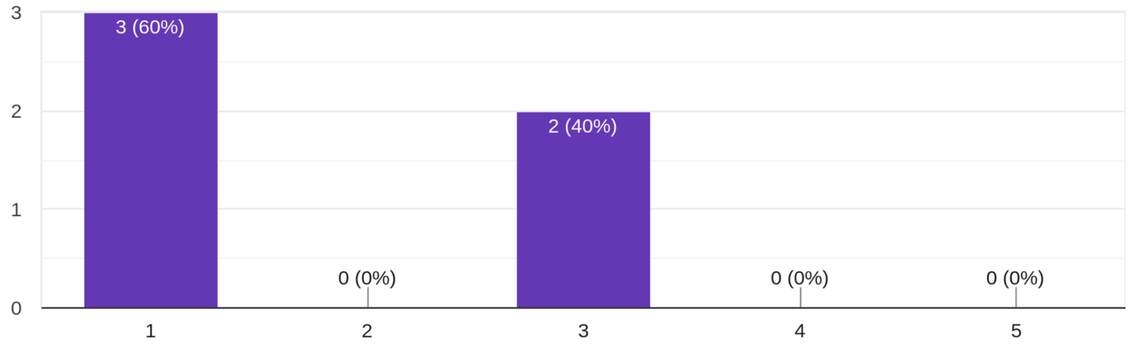
Where do you stand on adblocking for paid services?

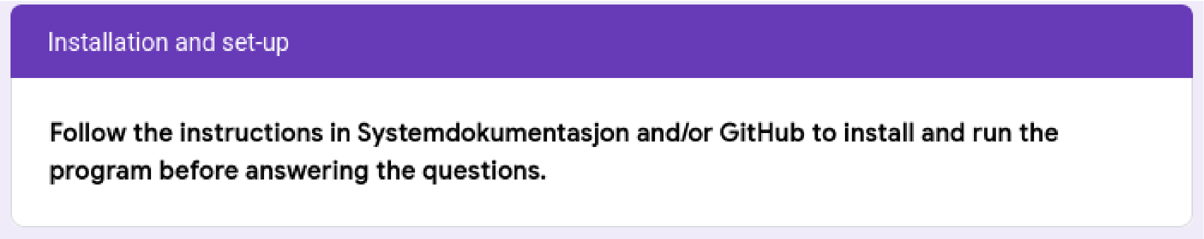
5 responses



Where do you stand on adblocking for free services?

5 responses



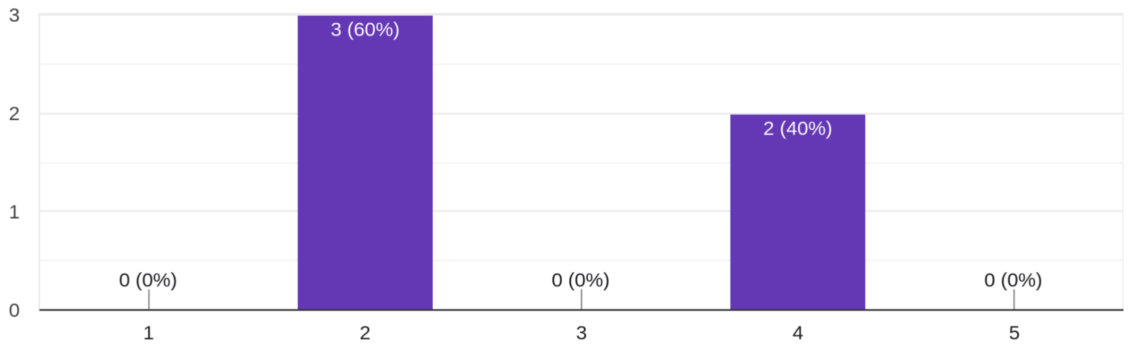


Installation and set-up

Follow the instructions in Systemdokumentasjon and/or GitHub to install and run the program before answering the questions.

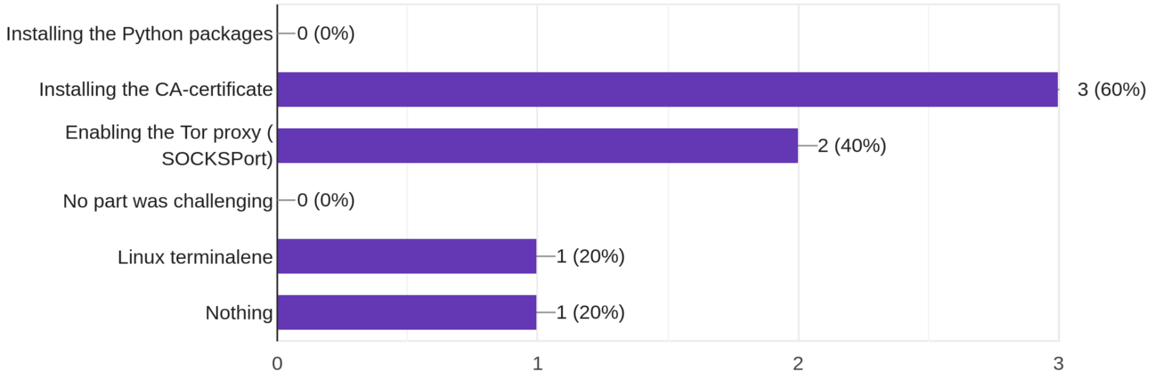
Installation and set-up of adblock25 is

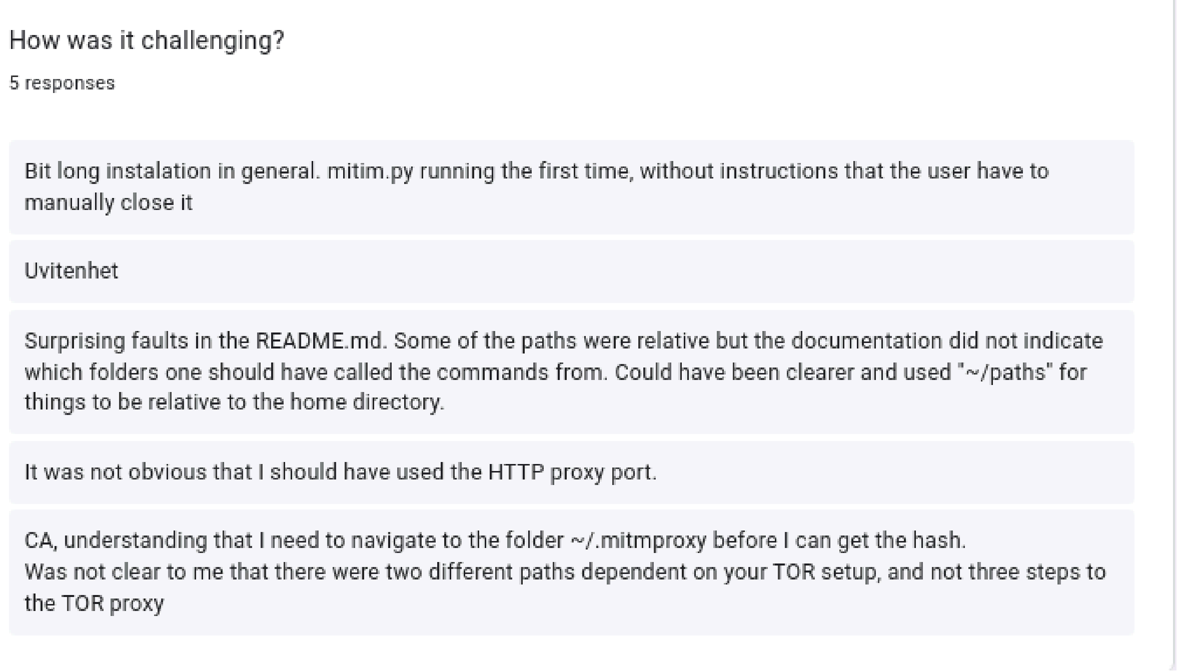
5 responses



If something was challenging, what part(s) was it?

5 responses





How was it challenging?

5 responses

Bit long instalation in general. mitim.py running the first time, without instructions that the user have to manually close it

Uvitenhet

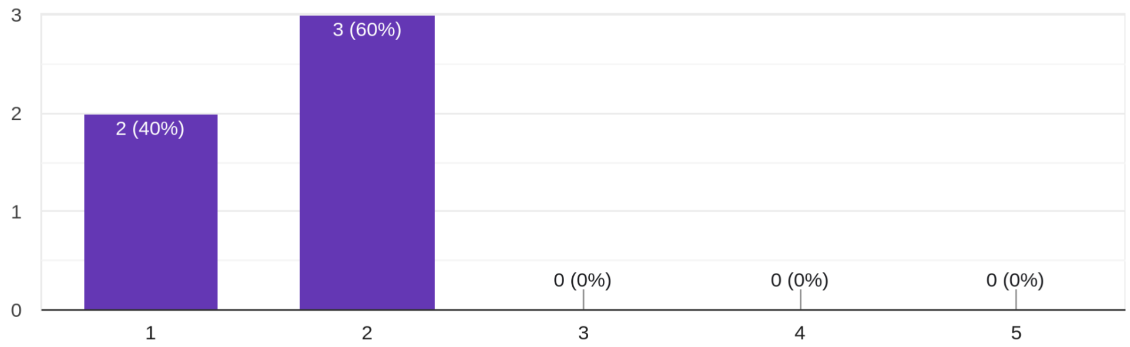
Surprising faults in the README.md. Some of the paths were relative but the documentation did not indicate which folders one should have called the commands from. Could have been clearer and used "~/paths" for things to be relative to the home directory.

It was not obvious that I should have used the HTTP proxy port.

CA, understanding that I need to navigate to the folder ~/.mitmproxy before I can get the hash.
Was not clear to me that there were two different paths dependent on your TOR setup, and not three steps to the TOR proxy

Running adblock25 is

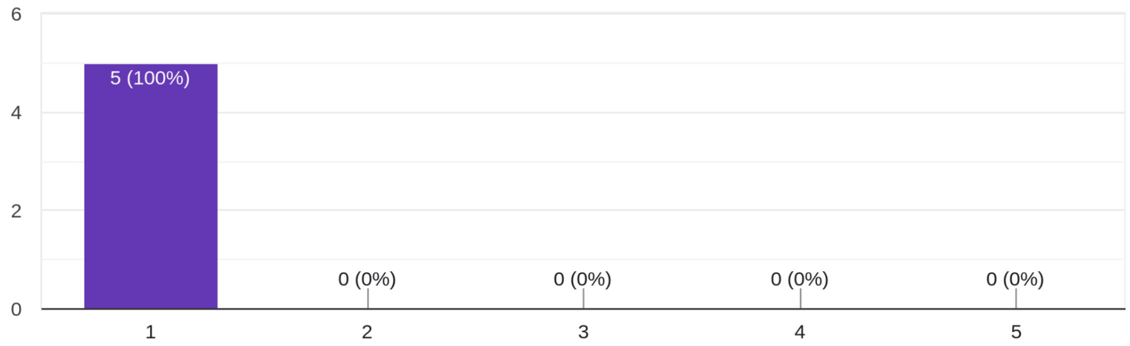
5 responses



Stop and run adblock25 again, this time binding the proxy to a different port

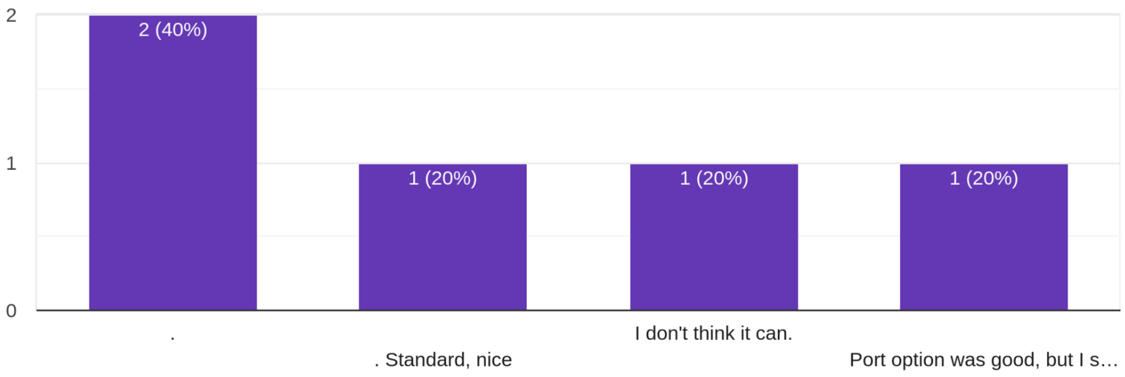
Using the adblock25 port option is

5 responses



How could the port option be improved?

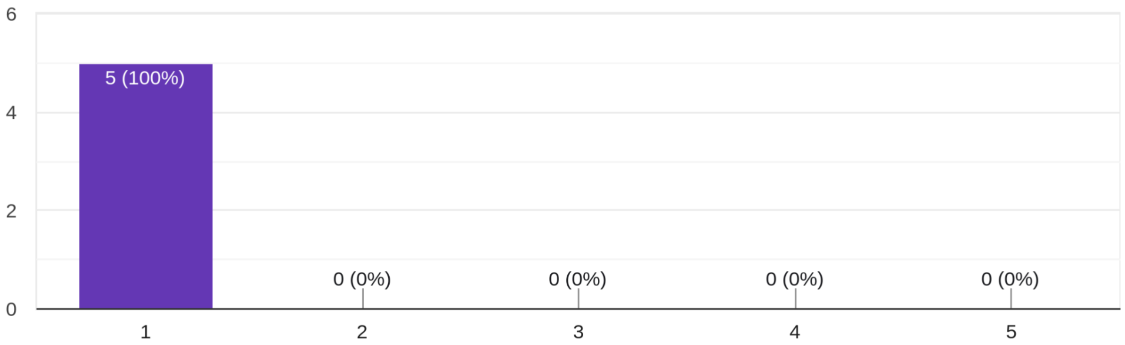
5 responses



Stop and run adblock25 again, this time loading write.py as the addon

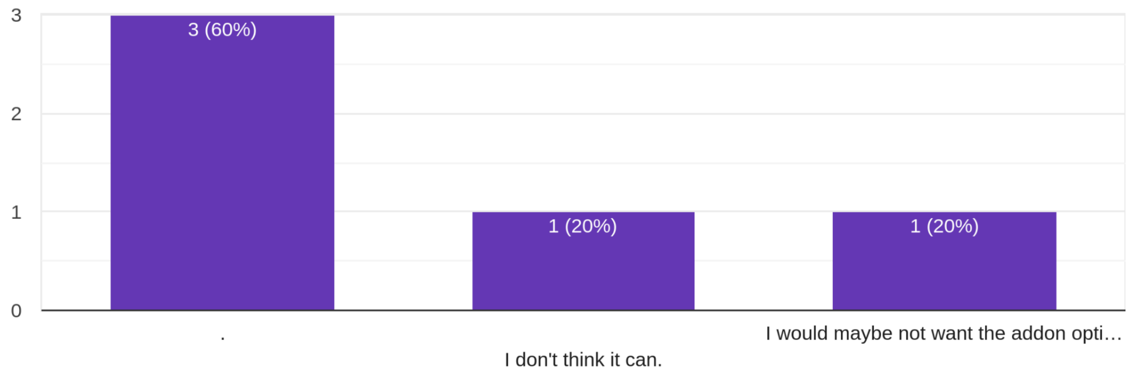
Using the adblock25 addon option is

5 responses



How could the addon option be improved?

5 responses

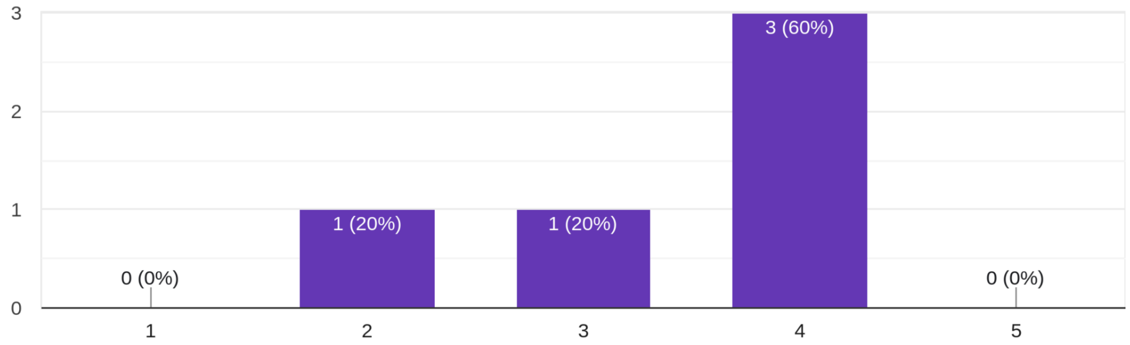


UX

adblock25 is intended to be used continuously with AntennaPod. Imagine having it running at all time and processing every episode you listen to.

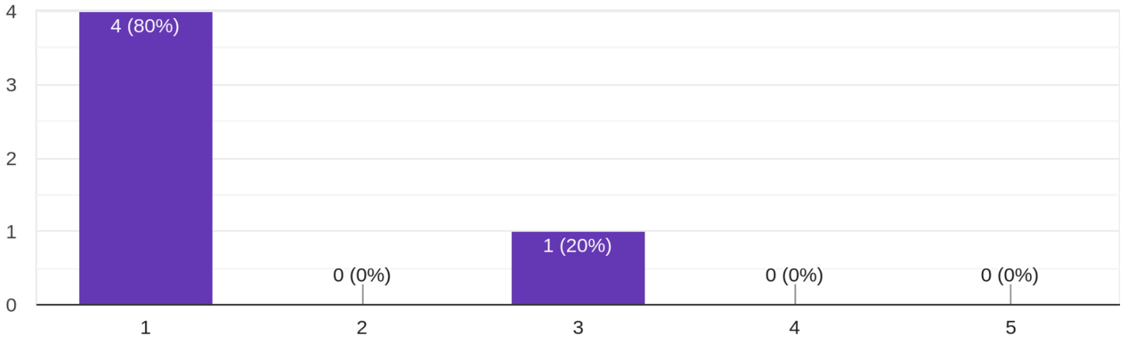
The loading time, i.e. the time it takes from you press play until the podcast starts playing, is

5 responses



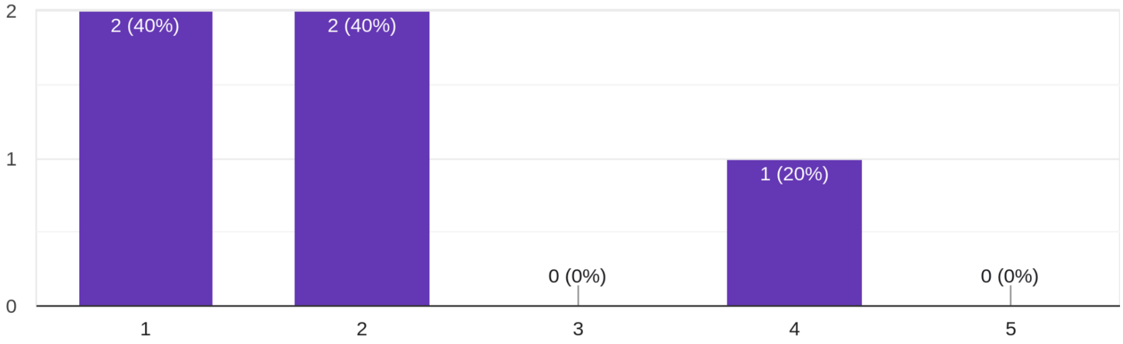
An advertisement cut from the middle of a program results in a certain sound and "jump". This behavior is

5 responses



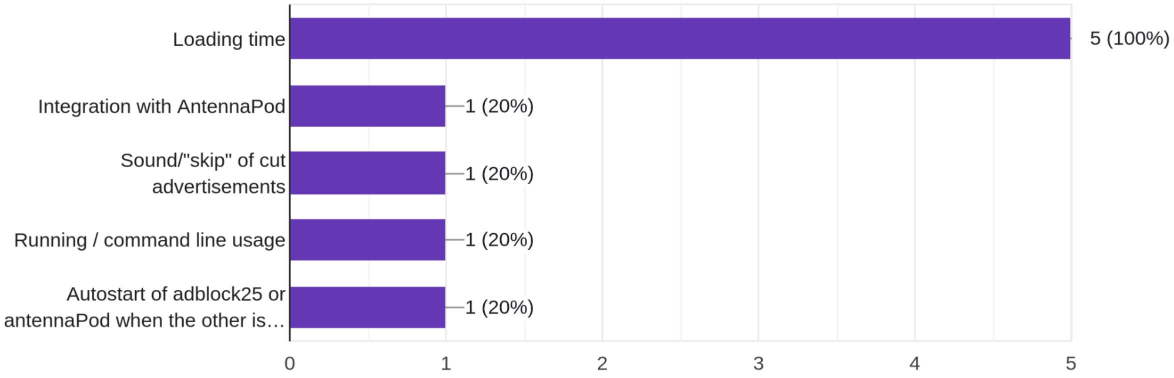
The overall integration with AntennaPod is

5 responses



What part(s) of adblock25 should be improved?

5 responses



How should it be improved?

5 responses

Loading times

Shorter loading time, perhaps precedural loading so that it plays when it has gone over a certain point if that is possible?

Mostly the loading time was unexpectedly long. The installation documentation can be improved. Make sure all paths are home relative or absolute.

Maybe it's possible to stream it while it's being processed?

I think the app works well. It would be nice if there was some built in function in AntennaPod (something visible in the antennapod settings) that shows that a proxy is active.
Reducing loading time is always a good thing and will make people happy. I do think that the current loading time seams good and reasonable

What kind of functionality do you think is missing/should be added?

5 responses

Pre loading next episode if loading times cannot be reduced

Nothing special it seems to run "as advertised"

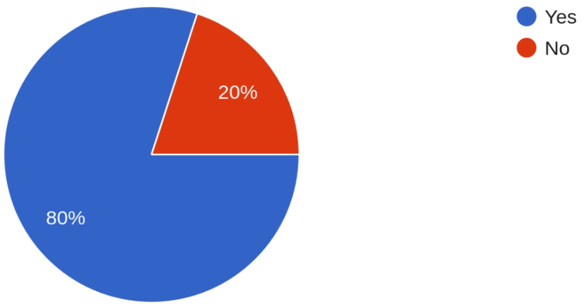
For linux systems it would be nice to have it as a service that can start on boot/session login.

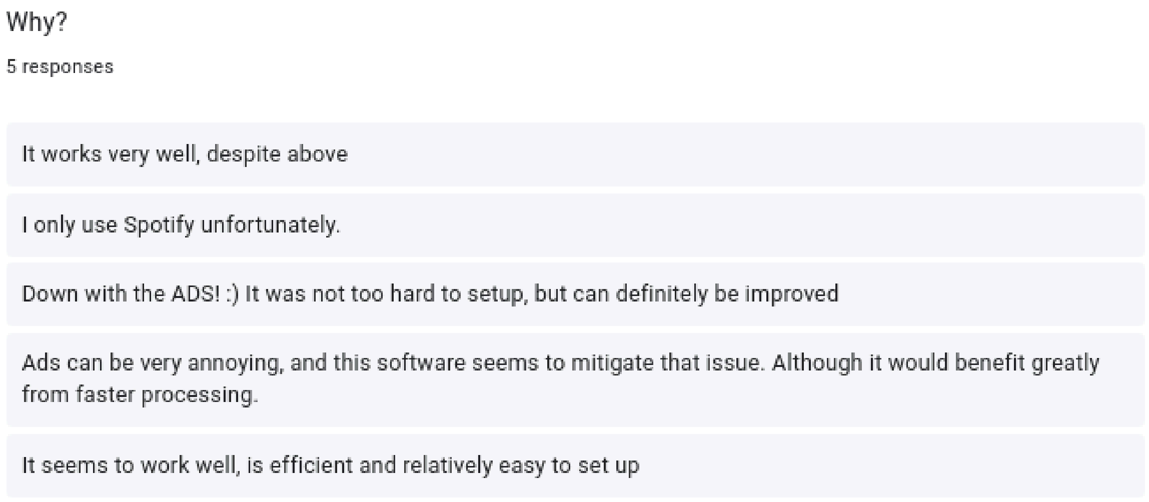
Could consider an interactive setup that guides you or does part of the process for you.

A possibility to pre-process a bunch of podcast episodes (remove the commercial and store them, e.g. in a downloaded library) so that you dont need the loading time when listening

If you were to use an adblocker for podcasts, would you consider adblock25 an alternative?

5 responses





Why?

5 responses

It works very well, despite above

I only use Spotify unfortunately.

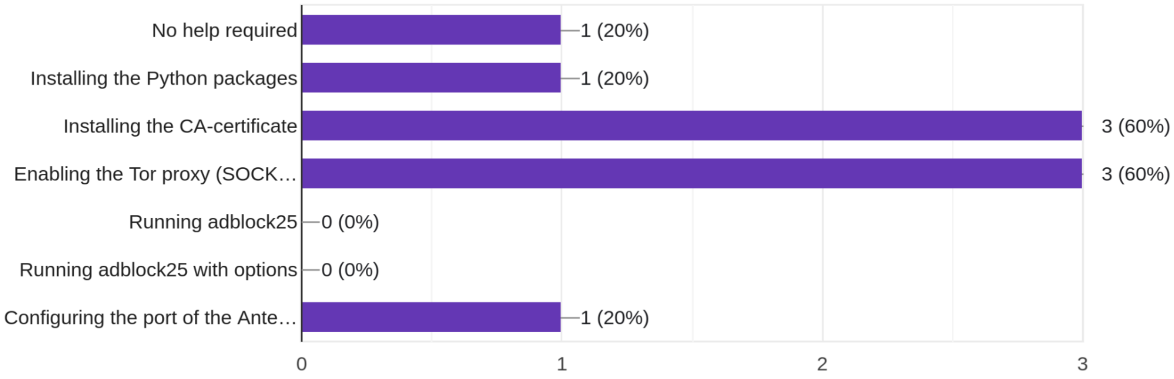
Down with the ADS! :) It was not too hard to setup, but can definitely be improved

Ads can be very annoying, and this software seems to mitigate that issue. Although it would benefit greatly from faster processing.

It seems to work well, is efficient and relatively easy to set up

Help required on

5 responses



I. AI Declaration

The standard NTNU declaration regarding use of AI-tools in form of a PDF.

NTNU

Declaration of AI aids and -tools

Have any AI-based aids or tools been used in the creation of this report?

◉ No

○ Yes

If *yes*: please specify the aid/tool and area of use below.

Text

□ **Spell checking.** Are parts of the text checked by:
Grammarly, Ginger, Grammarbot, LanguageTool, ProWritingAid, Sapling, Trinka.ai or similar tools?

□ **Text-generation.** Are parts of the text generated by:
ChatGPT, GrammarlyGO, Copy.AI, WordAi, WriteSonic, Jasper, Simplified, Rytr or similar tools?

□ **Writing assistance.** Are one or more of the report's ideas or approach suggested by:
ChatGPT, Google Bard, Bing chat, YouChat or similar tools?

If *yes*, use of text aids/tools apply to this report - please specify usage here:

Codes and algorithms

□ **Programming assistance.** Are parts of the codes/algorithms that i) appear directly in the report or ii) have been used to produce results such as figures, tables or numerical values been generated by: *GitHub Copilot, CodeGPT, Google Codey/Studio Bot, Replit Ghostwriter, Amazon CodeWhisperer, GPT Engineer, ChatGPT, Google Bard* eller lignende verktøy?

If *yes*, use of programming assistance aid/tools apply to this report - please specify usage here:

Images and figures

□ **Image generation.** Are one or more of the reports images/figures generated by:
Midjourney, Jasper, WriteSonic, Stability AI, Dall-E or similar tools?

If *yes*, use of image generator aids/tools apply to this report – please specify usage here:

Other AI aids or tools. Have you used other types of AI aids or -tools in the creation of this report?
If *yes*, please specify usage here:

☑ I am familiar with NTNU's regulations on artificial intelligence. *I declare that any use of AI aids or tools are explicitly stated i) directly in the report or ii) in this declaration form.*

Christian Ryddheim Dahlin
Trondheim 30.05.24

Signature/Date/Place

NTNU

Kunnskap for en bedre verden