User interface for AI-Rehabilitation

Research and Development Project 2

January 4, 2021

Student id	Name
201508251	Frederik Andersen
201506770	Theis Egsgaard Hansen

Counselor: Christian Fischer Pedersen

Contents

1 Termlist

Table 1: Terms used throughout the report

Term	Explanation
AI	Artificial Intelligence or machine learning.
Assistive aid	An item to help a citizen in their everyday life e.g. wheelchair or walker.
Case worker	A person employed by a municipality to prescribe training courses or assistive aids to a citizen.
SSN	Social Security Number or CPR in Danish.
GOP	A citizen's rehabilitation plan made by a hospital.
R&D1	A previous R&D project made by the group [1], which this report is a continuation of.
Meeting 1	01-10-2020: A Focus group interview and Wizard of Oz test conducted during R&D1.
Meeting 2	28-10-2020: A semi-structured interview with the addition of a Wizard of Oz test and Think-aloud technique.
Pilot group	A group of potential end users, consisting of five case workers, and was used to evaluate the progression of the project.
UI	User Interface.

2 Introduction

This project is a continuation of the groups previous R&D1 work [1], and is part of a project between Aarhus University and Aalborg Kommune called AIR [2]. The goal for this project is to create a web application for case workers, using an AI made by Aarhus University. The web application is meant to visualize the result from the AI to serve as a decision support tool to assist case workers in deciding whether or not a citizen should receive training. The relevance for this project stems from the case workers not being fully aware if a citizen is able to complete a training course. If a citizen is not able to complete the prescribed training, it results in a waste of resources.

The web application is supposed to reach a low fidelity prototype level with a focus on the UI. While results from this project might not be useful in the real world, they will in all likelihood form the basis of future work and decisions. Because the foundation for possible future projects are formed here, there will in this project also be an emphasis on the decision making behind the web application, such as which programming language to use.

2.1 AIR

AIR or AI-Rehabilitering is a project between 7 municipalities in Denmark, using AI as a support tool for case workers [2]. The AI can create an indication on whether or not a citizen is able to complete a training course, by calculating an estimated probability, based on various factors. The reason behind the AI is twofold with the first being quality of life. If the citizen is able to train, it becomes possible to prolong the citizen's life and the need of fewer assistive tools in the future. The second reason is that the municipalities is able to save both money and personal time on citizen's who might not complete the training course, and thereby reap the benefits [1, p. 3]. A rich picture of the AIR project is illustrated on figure 1.

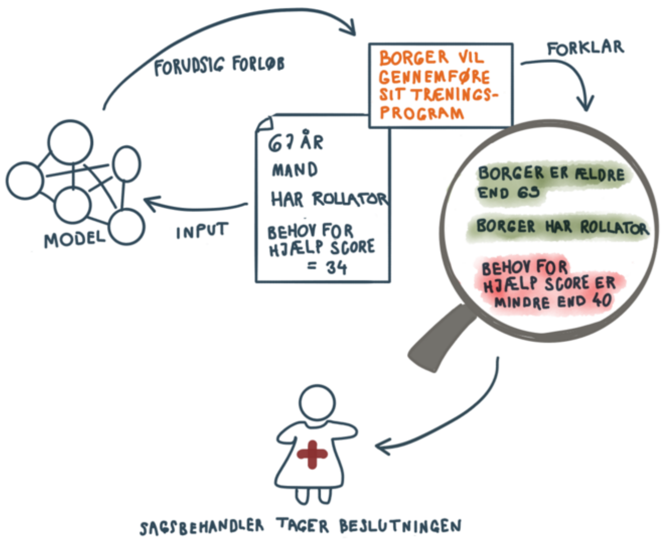

FORUDSIG FORLØB
BORGER VIL GENNEMFØRE SIT TRÆNINGS-PROGRAM
FORKLAR
67 ÅR
MAND
HAR ROLLATOR
BEHOV FOR HJÆLP SCORE = 34
MODEL
INPUT
BORGER ER ÆLDRE END 65
BORGER HAR ROLLATOR
BEHOV FOR HJÆLP SCORE ER MINDRE END 40
SAGSBEHANDLER TAGER BESLUTNINGEN

Figure 1: Rich picture of AIR [2]

2.2 Problem formulation

The focus of the R&D project is to design, implement, test, and document a web application for the research project AI Rehabilitation (AIR). The web application should reach a proof of concept prototype" level of maturity. The functionality of the web application should support the rehabilitative decisions done by the case worker. The exact functionalities, interaction design and user experience should be established through participatory design efforts with a pilot group of potential end-users. This R&D project also includes the formation of a pilot group and running a participatory design process with the group where the web application prototype is incrementally refined. The pilot group should be established in cooperation with the R&D project's external partners. Finally, the web application prototype must be proven functional via concrete experiments including the pilot group.

3 Methods

This section describes the different methods used throughout this project. Each method are presented and its usages described. This project follows the same user driven development principles as described in R&D1 [1, pp. 4–5].

3.1 Semi-structured interview

A semi-structured interview is a combination of a structured and an unstructured interview. A structured interview typically has closed and short questions, and can be used if the possible answers are known or if the participants are in a hurry [3, p. 269]. An unstructured interview is generally more loose and more similar to a conversation, but the interviewer still needs to have an agenda of questions regarding

certain topics [3, pp. 268–269].
A semi-structured interview includes the open ended answers from the unstructured interview, and a plan with short and structured topics from the structured interview. The interviewer uses prepared questions as a script to make sure that all of the relevant aspects are covered [3, p. 269]. The questions should not be stated in such a way, that the interviewee is encouraged to answer untruthfully to not offend the interviewer [3, pp. 270–271].

A different problem to avoid is the lack of time for each interview. If the questions for the interviewee and by extension the answers are rushed, it could lead to answers not been thought out thoroughly, thus making the answer less useful [3, p. 270].

This interview format was used as part of the finalization of the design phase during meeting 2, based on a wireframe design from R&D1 [1, p. 11]. Using this interview form was ideal for getting feedback from the interviewees regarding the design in various different aspects without anticipating a certain type of answer. Each interview was conducted for 45 minutes with each participant from the pilot group. The allotted time proved enough to get the answers needed from all the prepared questions.

3.2 Wireframes

Wireframes is an easy and cheap way to capture and test the system flow including some of the core functionalities of a product [4]. This is usually done in the early design stage by making an interactable UI blueprint of a system [5]. Wireframes was used to create a visual and interactable design representation of the intended system. The wireframes in this report is a continuation of the design based on meeting 1 [1, p. 11]. Before attending meeting 2, this design was refined, and can be seen in section 6.1. Based on the feedback gathered from meeting 2, a final wireframe design was made, which is presented and discussed in chapter 7.

3.3 Wizard of Oz Test

A Wizard of Oz test is a method of using an interactable prototype, and making changes according to the user both in terms of design and system flow [3, p. 428]. The idea is that the changes happens behind the curtain, without disturbing the user experience, and thus the name of the method. Part of meeting 2 was to use the Wizard of Oz test to change the behavior and visuals of the refined wireframe design. This was done according to the user by following the guidelines of the semi-structured interview form.

3.4 Think-Aloud Technique

The think-aloud technique is used to get a sense as to how the user thinks. In order to get the technique working, the user has to say whatever he or she thinks [3, p. 296]. It is important to remember that there are no wrong answers, as every comment and thought is useful. This technique was used in combination with the Wizard of Oz test to get a further understanding in the change of system flow and behavior.

3.5 MoSCoW

The MoSCoW technique was developed in 1994 by Dai Clegg [6], and creates a prioritized list of requirements for a product, using natural language. A benefit of using a natural language is the ability to be applied with different groups of people, with varying levels of product knowledge. It is therefore possible to collaborate on a list with a group of stakeholders, without compromising their ability to generate requirements. When creating the MoSCoW list, it is important to know that the requirements written down is only for a specific product release, and not for the rest of the product lifetime.

The **M**o**SC**o**W** has four levels of prioritization:

- Must have
- Should have
- Could have
- Will not have

These four categories each symbolize the importance of a feature in a system. The first three levels symbolize requirements supposed to be in the product, while the last level symbolize requirements that will not. Knowing the meaning behind the words "must", "should", "could" and "will not" is key to understand how the MoSCoW works. The first three prioritization levels are based on the words "must have", "should have" and "could have", which specifies a level of certainty of the requirement being included in the product.

Starting with "must have". This specifies that the requirement is "almost guaranteed" to be present. Almost guaranteed simply means that it is possible for development to take longer than anticipated. This may result in the feature requirement not being ready when the product is supposed to be released.
Looking at the next MoSCoW level "should have", a requirement using this phrase is a more "nice to have" feature. This does not mean the feature will not be present in the product, but rather it is "expected" to be present rather than "guaranteed". The "could have" phrase indicates that the requirement is likely not to be present, but it might be if enough resources to complete the task is available.
Finally is the "will not have" category which may seem unimportant, as it could be argued that if a feature is not listed in the first three categories, it is not included in the product. While this is partly true, having the "will not have" category prevents the feature from being included in the other categories during development, and therefore prevents scope creep [6].
Knowing the meaning behind the four categories can help determine the importance of a feature or requirement. While this will help determine which features

are a ”must have” and which are ”should have”, the MoSCoW technique will not help in determining a features relative importance within the same category [6]. This means that if two features belongs to the same category, it is not possible to determine which feature should be implemented first.

In this project the MoSCoW technique has been used to create a requirements list using input from the pilot group, which have little to no knowledge of the technical details of the system. This means that it is possible to prioritize requirements based on a conversation with the users, and having the requirements on a more conceptual level, rather than being overly specific. While a requirements list on a more conceptual level is a great addition to the project, this also presents a challenge. The challenge stems from using requirements to create a system that adheres from a vague requirement description made by the end users. But as for this project, it still gives a useful direction as to where the system is expected to go.

4 Requirements

The following requirements was made using the MoSCoW technique, and based on feedback from meeting 1 and 2.

4.1 Functional Requirements

This section contains functional requirements. The list is divided into four sections each corresponding to the MoSCoW prioritization list.

Table 2: Functional Requirements

Req. Id	**Description**
FR1	The system must be a web application.
FR2	The system must have a score showing the result from the AI.
FR3	The score must be shown as a whole number.
FR4	The score must be explained by text.
FR5	The system must have a graphic representation supporting the score from FR2.
FR6	If FR8 is completed, the score must be supported by arguments for conducting training.
FR7	If FR8 is completed, the score must be supported by arguments against conducting training.
FR8	The user must enter the citizen’s SSN to get a score and arguments.
FR9	If FR8 is completed, the system should be able to explain both the positive and negative influences of the arguments.
FR10	The SSN from FR8 should be displayed along with the citizen’s age and name.

Continued on next page

Table 2 – continued from previous page

Req. Id	Description
FR11	If FR8 is completed, the system should be able to display a list with the citizen's assistive tools if available.
FR12	If FR8 is completed, the system should display a citizen's GOP if available.
FR13	If FR8 is completed, the system should display the citizen's diagnosis if available.
FR14	If FR8 is completed, the system should display the citizen's previous or currently active training plans if available.
FR15	The tools from FR11 could be visually represented.
FR16	The number of home help hours could be visually represented.
FR17	The score could be supported by a recommended training plan.
FR18	If FR8 is completed, the system could display a future forecast about the citizen's need for assistive tools.
FR19	The system will not be able to display the citizen's motivation.

4.2 Non-functional Requirements

This section contains the non-function requirements.

Table 3: Non-functional Requirements

Req. Id	Description
NFR1	If FR8 is completed, the system should be able to show a result in under 3 seconds.
NFR2	The system should be operated by using as few mouse clicks as possible.
NFR3	The system should be operated by scrolling as little as possible.
NFR4	The system should be compatible with all modern browsers [7].

5 Analysis

This section contains an analysis of the various programming languages for which to build a web application, and thus acts as a decision foundation.

5.1 Web Application framework

As for framework regarding the web application, three options were considered: Angular [8], React [9] and Vue [10]. Since the group had no previous experience with either of these an analysis was made. The goal of the analysis was to shed some light on which framework would be best suited for the project. These frameworks were chosen after cross referencing four independent websites for the three ”best front end websites in 2020” [11] [12] [13] [14]. This resulted in Angular, React and Vue consistently trading places in top 3. To support these findings, a request for help was sent via email to a teacher [15] at Aarhus School of Engineering [16]. He stated that both React and Vue would be good candidates.

Table 4 gives a short overview of some of the basic information regarding each of the contestants.

Table 4: Basic information about the three web application frameworks

	Angular	React	Vue
Developer	Google	Facebook	Ex-Google employee and Vue community
Initial release	2010	2013	2014
Current version[1]	10.2.0[17]	17.0.1[18]	2.6.12[19]
Type	Framework	Library	Framework
Language	HTML TypeScript	JSX	HTML JavaScript TypeScript support JSX support
Approx. size (KB)[20]	500	100	80

[1] As of 11-11-2020

From this point and forward Angular, React and Vue will all be referred to as a frameworks, unless specified otherwise, even though React is not a framework, but rather a library.

Looking at table 4 one of the big differences is the *Type* and *Language*. Starting with the type section, React is the only of the three contenders that is considered a library. This is a design choice by the developers. React, by it self only offers very limited functionality compared to Angular and Vue. This is because Facebook has created the core library, and the community is responsible for coming up with ideas for extensions and improvements. An example of this is the state management, which is not part of the core React library, but rather a third party inclusion which must be present in the project. Having the need to be reliant on third party libraries for important core functionality, which may or may not be up to date can cause huge problems, since the library might not support the latest version of the core React library. This ”problem” of being dependent on third party libraries does not have the same magnitude when using Angular or Vue, since these two are full fledged frameworks.

Shifting the focus to the language section, both Angular and React are limited in what languages they support. Angular uses a mix of HTML and TypeScript, which enables static type checking and therefore makes it possible to find potential bugs early in the development [21]. While TypeScript can be a blessing with the static checks, some developers might see it as a curse, since it is not dynamic like JavaScript.

React on the other hand uses its own language JSX, which is a mix of JavaScript and HTML. Because JSX is a mixed language, it enables the developer to write HTML elements in JavaScript [22], which can make it easier to see a code coherence. While React does not force developers to use JSX, it is highly recommended [22].
Vue uses HTML like Angular, and JavaScript. The JavaScript allows the developers to implement things more dynamically, because the language itself is dynamic. While the default language for Vue is JavaScript, it is also possible to use TypeScript to achieve some of the features like static type checking. To make the framework cater to an even wider audience, Vue also offers JSX support, which could cause some React developers to change their framework of choice.

When comparing Angular and Vue, both tend to share some of the same core functionalities and principles, since they both use HTML and support TypeScript. Angular provides more functionality than Vue because Angular has had longer time to develop. This increased functionality comes at the cost of a steeper learning curve. Vue on the other hand has a much flatter learning curve. This flatter learning curve partly comes from the fact that Vue is less opinionated than Angular [23], which can be a benefit or a disadvantage depending on the developer.

Angular is more scalable as the complexity of an application goes up compared to Vue [24]. While this might prove useful in the long run, it may slow down product development in the early stages.

A clear difference between Angular and Vue is the size as seen on table 4. The size difference means that Angular supports more functionality compared to Vue, but it also means that Angular utilizes more memory.

Another metric to determine which framework to use is popularity. As this can be a difficult metric to measure, a solution has been found using GitHub [25]. Since all three frameworks are available on GitHub for download, GitHub's star system has been used for the popularity metric. A star on GitHub can mean different things depending on the person using it, and can even be used by bots. This is by no means the single best way to determine popularity, but it can still be used as a guideline.

Figure 2 shows the GitHub stars over time for the three frameworks [1]. Looking closer it is possible to see four entries rather than three. The reason for this is the vuejs/vue-next is Vue 3, while vuejs/vue is Vue 2.6.12. Vue 3 has been included in the graph as it is the new rewritten version of Vue, using TypeScript instead of JavaScript. Another thing to notice is the time of Angular's first appearance. According to table 4, Angular were released in 2010, but the graph states that it was released during 2015. The explanation is simple, as Angular or AngularJs, was released in 2010 but then was completely reworked and later released as Angular. The GitHub repository in figure 2 does therefore not include the time before the rework of Angular.

[1]The graph was created the 11-11-2020

Looking at the graph it is clear that even though Vue was the last of the three frameworks to be released it has the greatest popularity among the GitHub users. This indicates that there is a big community supporting Vue. Shifting the focus to Angular there is a clear lack of stars compared to React and Vue. A reason for this could simply be because of how opinionated Angular is.

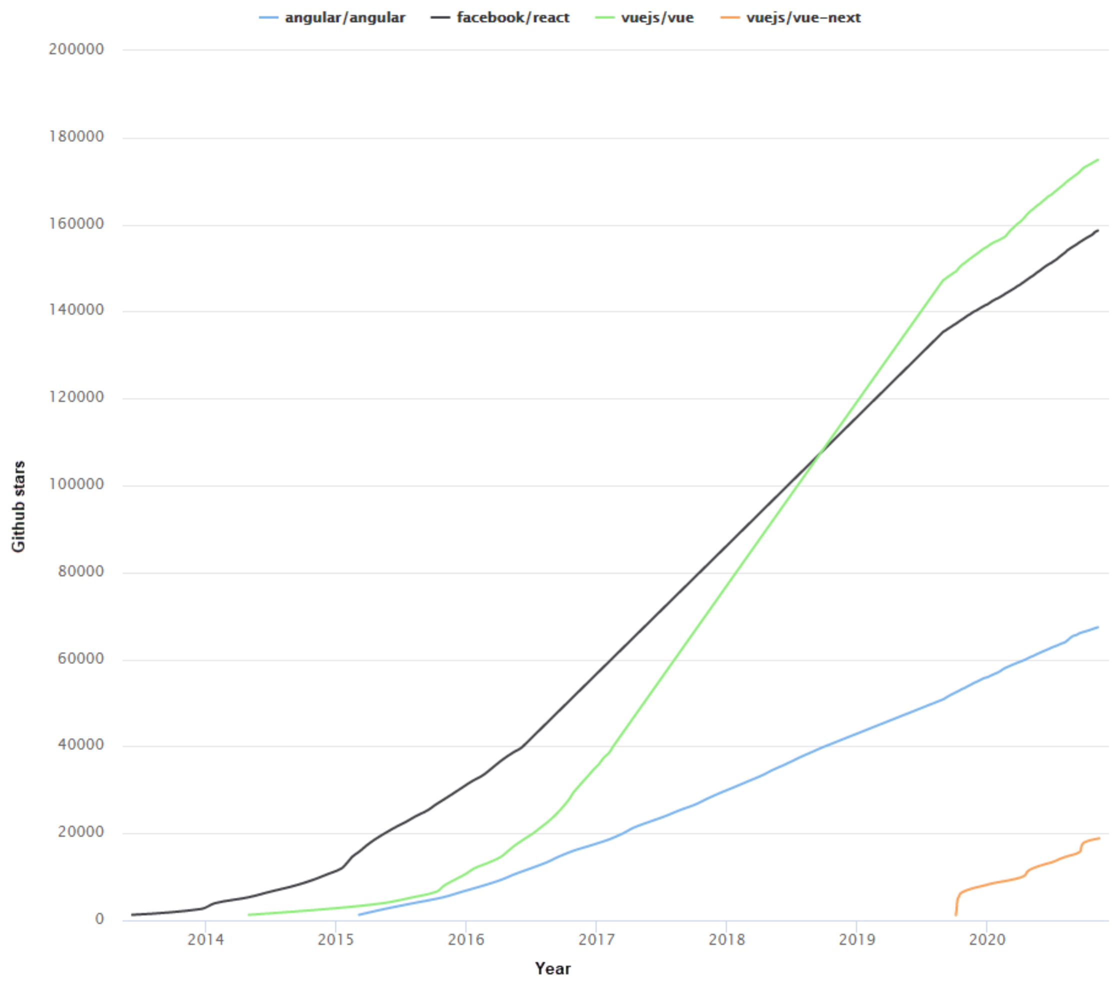


Figure 2: Graph showing GitHub stars for the different contestants

Conclusion
For the project the group has chosen to work with Vue. This decision comes from the fact that Vue is more lightweight than Angular and has a fairly large community. This choice is also backed up by the recommendation received from Poul Ejnar at Aarhus University.

5.2 Library

This section presents some of the considerations which impacted the choice of a UI library used alongside with Vue.

5.2.1 UI library

As the framework of choice landed on Vue, and specifically version 2.6.12 [2] the next step was to figure out how to create conventional UI elements [26, p. 29]. A solution could be to spend time and create all the elements from the ground up including layout, customizability, functionality, and styling. Since the group members previous experience with HTML, CSS and JavaScript is limited, this solution was not feasible within the scope of the project.
Another solution was to use a library with premade conventional UI elements, which would only need to be placed in the application and styled. As using libraries is a common practice it was decided to go with this solution.

Choosing a library can be a somewhat daunting task as this can have a huge impact on the ability to achieve the desired look and functionality. The group however did not have the same issues, as the functionality of the web application was fairly limited. This means that even if a library change proved useful in certain cases, it might not be all elements that would have to be changed, which saves time.

Because of the above reasons, and the fact that there are many different libraries providing similar functionality [27], the UI library of choice for this project became Vuetify [28].

6 Design

In this section a wireframe design is presented and described. The refined design was made based on the results from R&D1 and thus made prior meeting 2. The design therefore only encapsulates some of the requirements described in section 4. To see the final wireframe design, go to chapter 7.

6.1 Wireframe

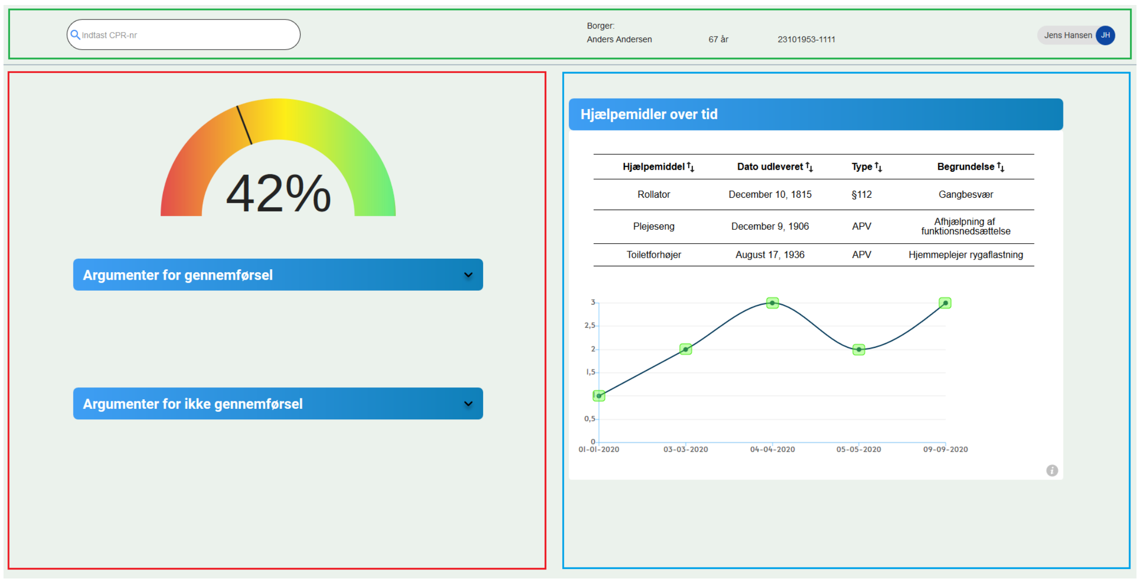
It is possible to divide the wireframe into three sections. The sections are a top section marked with a green rectangle, a left section marked with blue rectangle and a right section marked with a red rectangle.
In the green rectangle from the left, the user can enter a citizen's SSN to get a result from the AI. A citizen's name, age and SSN is shown to inform the user that the correct citizen is being processed.
Lastly a user profile badge is shown on the right, where it should be possible to change some user settings. A small almost invisible gray line is made to distinctly separate the top area from the rest [26, p. 36].

The number and the radial bar in the red rectangle symbolizes the result from the AI. Each blue accordion [26, pp. 29–30] under the result is describing the arguments for and against a citizen for completing training. Each accordion has been marked

[2]The latest version which is not Vue 3.x.x at the time of writing.

by a small arrow on the right side. This is to make it obvious for the user that the accordion is clickable [26, p. 37].

The blue rectangle shows a drop down window containing a graph with the citizen's usage of assistive aids over time. If additional information were to be added, this could easily be done with the addition of tabs [26, pp. 80–81].



Hjælpemiddel ⇅	Dato udleveret ⇅	Type ⇅	Begrundelse ⇅
Rollator	December 10, 1815	§112	Gangbesvær
Plejeseng	December 9, 1906	APV	Afhjælpning af funktionsnedsættelse
Toiletforhøjer	August 17, 1936	APV	Hjemmeplejer rygaflastning

Figure 3: Wireframe design prior to meeting 2

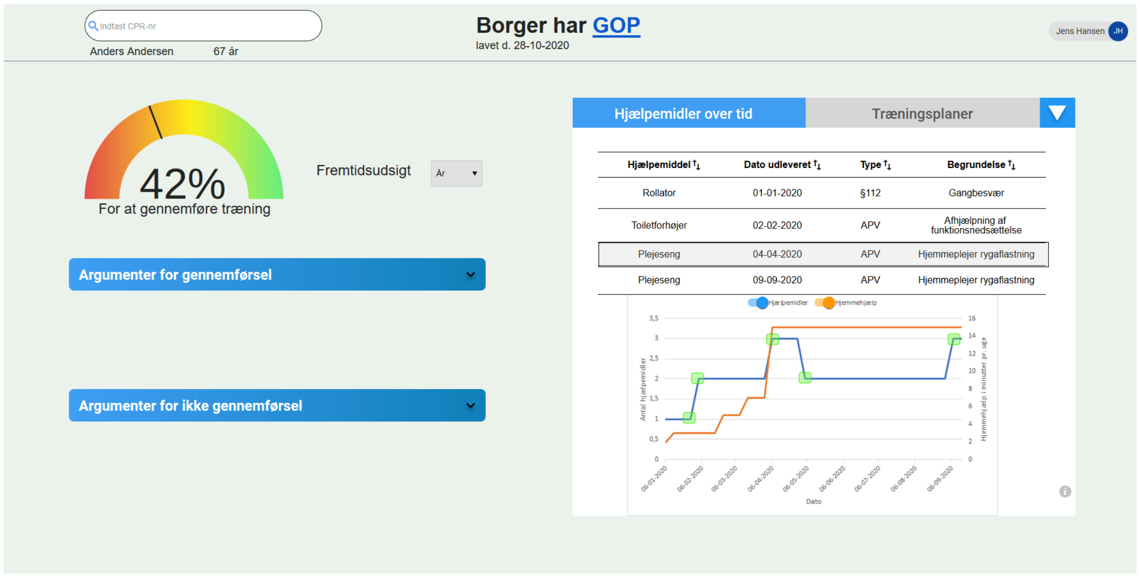
7 Results

In this section the results from meeting 2 is presented. The results are two fold as the results are both in the form of a final wireframe design based on the feedback from meeting 2, and also in the form of an initial Vue web application.

7.1 Wireframe

Figure 4 shows the final wireframe design based on the feedback from meeting 2. While the elements are mostly the same as shown in section 6.1, there are some small, but important changes. Starting at the top portion of the wireframe, the citizen's information has been moved below the field for entering the SSN. This enables the user to only look down the left side of the screen for all the necessary information, while the right side is more focused on extra information [26, p. 36]. A new element was also added at the top which was a GOP. A citizen with a GOP could make the result from the AI unimportant. The GOP was for that reason made bold with a large text size to indicate its importance and blue to indicate clickability [26, pp. 34, 37].

Shifting the focus to the left side of figure 4 the score and the radial bar has been slightly moved to the left, to make room for a dropdown menu. The dropdown



Hjælpemiddel ↑↓	Dato udleveret ↑↓	Type ↑↓	Begrundelse ↑↓
Rollator	01-01-2020	§112	Gangbesvær
Toiletforhøjer	02-02-2020	APV	Afhjælpning af funktionsnedsættelse
Plejeseng	04-04-2020	APV	Hjemmeplejer rygaflastning
Plejeseng	09-09-2020	APV	Hjemmeplejer rygaflastning

Figure 4: Final wireframe design

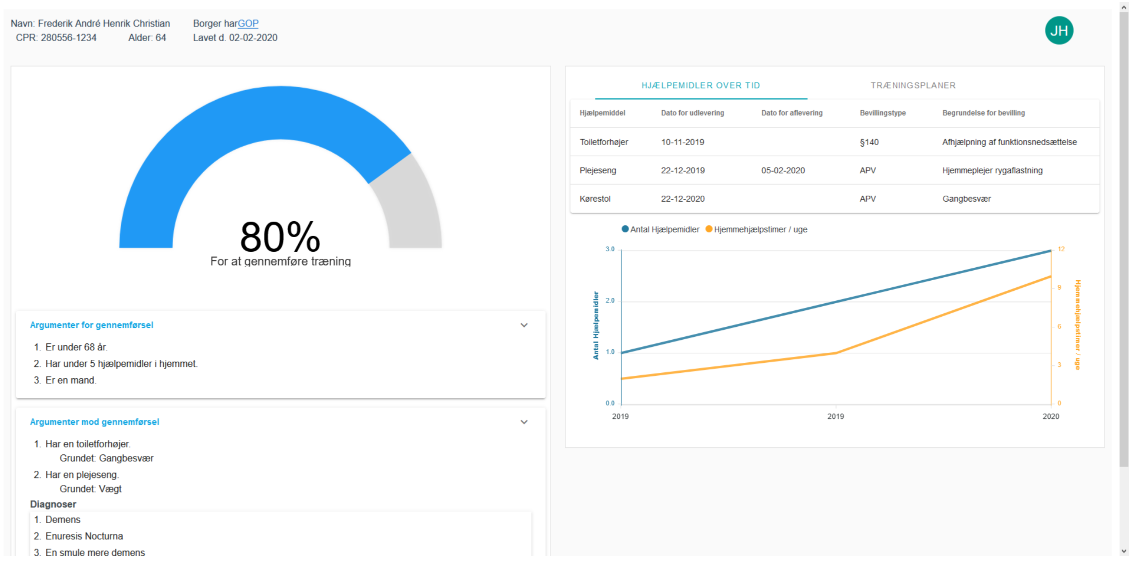
menu enables the user to get a future prediction of the citizen's ability to complete a training course. The rest of the left side remains the same compared to section 6.1. Looking at the right side there is one very large change, namely that the section has been placed inside tabs [26, pp. 80–81]. The use of tabs makes it possible to have additional information on the same page without having the need to scroll and cluttering the screen. Additionally it also makes it easier to extend the application in the future, so that more information can be added without having to redesign the entire web application. To avoid the wrong tab being clicked, the active tab has been made in a different shade and color than the rest [26, p. 81]. Figure 4 shows two tabs with the active tab representing the same data as shown in figure 3. The second tab adds new information to the web application, which is training plans. Being able to see the citizen's current and previous training plans can help the case workers decide whether or not the citizen needs additional training.

Looking further down the right side, the graph is also changed, as there are two graphs instead of one compared to section 6.1. On figure 3, only a single graph was present which is still the case on figure 4. The second graph however displays the number of hours the citizen has received in-home care in average per week of a given year. Using these two graphs in combination could possibly show a correlation in the number of assistive aids and the amount of time the citizen needs in-home care. Since this kind of information is not relevant for all users, it was made possible to hide either one of the graphs. both or collapse the tabs all together.

7.2 Vue application

This section present the developed web application made in Vue. While the looks of the application on figure 5 and figure 6 is somewhat close to figure 4, there have been made new changes in the design. For this project there have been a clear focus on the look and feel of the application, which in turn gives it a lack of functionality.

With that said the basis functionality of switching between the tabs on the right side is working, and the same for opening and closing the argument accordions on the left side. The values shown in figure 5 and figure 6 are hardcoded.



Navn: Frederik André Henrik Christian
CPR: 280556-1234 Alder: 64
Borger harGOP
Lavet d. 02-02-2020

JH

80%
For at gennemføre træning

Argumenter for gennemførsel

1. Er under 68 år.
2. Har under 5 hjælpemidler i hjemmet.
3. Er en mand.

Argumenter mod gennemførsel

1. Har en toiletforhøjer.
   Grundet: Gangbesvær
2. Har en plejeseng.
   Grundet: Vægt

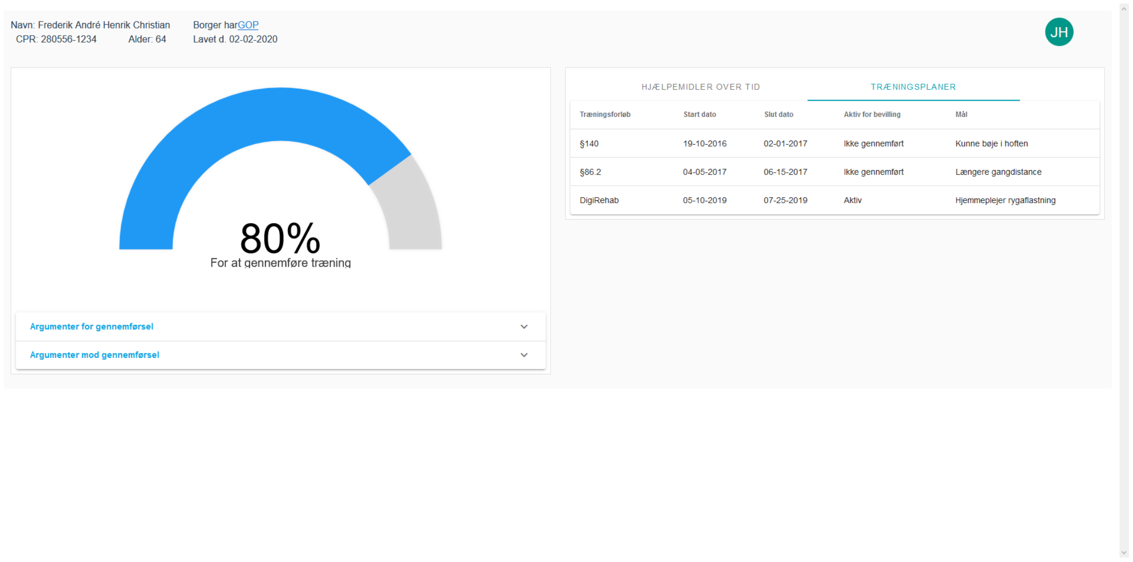
Diagnoser

1. Demens
2. Enuresis Nocturna
3. En smule mere demens

HJÆLPEMIDLER OVER TID | TRÆNINGSPLANER

Hjælpemiddel	Dato for udlevering	Dato for aflevering	Bevillingstype	Begrundelse for bevilling
Toiletforhøjer	10-11-2019		§140	Afhjælpning af funktionsnedsættelse
Plejeseng	22-12-2019	05-02-2020	APV	Hjemmeplejer rygaflastning
Kørestol	22-12-2020		APV	Gangbesvær

Figure 5: Image of the web application with the "Hjælpemidler over tid" tab active, both graphs showing and both argument tabs folded out.



Navn: Frederik André Henrik Christian
CPR: 280556-1234 Alder: 64
Borger harGOP
Lavet d. 02-02-2020

JH

80%
For at gennemføre træning

Argumenter for gennemførsel

Argumenter mod gennemførsel

HJÆLPEMIDLER OVER TID | TRÆNINGSPLANER

Træningsforløb	Start dato	Slut dato	Aktiv for bevilling	Mål
§140	19-10-2016	02-01-2017	Ikke gennemført	Kunne bøje i hoften
§86.2	04-05-2017	06-15-2017	Ikke gennemført	Længere gangdistance
DigiRehab	05-10-2019	07-25-2019	Aktiv	Hjemmeplejer rygaflastning

Figure 6: Image of web application with the "Træningsplaner" tab active

Even though the focus of the web application has been UI elements, the focus has not been on fine tuning all the elements to fit any screen size. The focus has rather been on making sure that each element is able to perform the task necessary and ensure that elements are placed correctly relative to each other.

The lack of fine tuning is clearly visible in the images as there are two elements that needs a closer look. The first example is a minor flaw where the SSN in the top left does not match the start of the sentences of the name above it. The second flaw is also minor as the text below the score is partly cut off in the bottom, thereby not showing the lower part of some letters. This flaw is not very visible and can be difficult to spot, but if the screen is small enough, this particular flaw starts to increase and cut off more of the letters.

Keeping in tone with the wireframes where it should be possible to hide one or both of the graphs, this functionality is also present in the web application. Hiding both graphs removes some of the visual clutter, but the space taken up by the graph is still being used, and the horizontal guidelines does not disappear.

One thing has been removed from the final wireframe design on figure 4. The requirement regarding the future forecast of a citizen's need of assistive tool was removed, due to the uncertainty of achieving such feature.

8 Conclusion

In this project the results from R&D1 have been used to create a wireframe which was discussed, tested and verified by the end users during meeting 2. This wireframe contained the ideas from the R&D1 and the general layout while trying to improve it and make it look like an actual web application. After the wireframe was verified by the end users an actual web application was created using wireframe design as a basis, and incorporating the ideas from meeting 2. The application was created using Vue after a careful analysis of different programming languages.

The product of this R&D project is a prototype application for the AIR project. While the prototype does not have any real functionality, the UI can be categorized as mostly finished with only minor bugs to fix.
Having this prototype is the first step in being able to see an actual product for the AIR project.

As for the group members, this project has been full of opportunities to learn a new language and framework, but also get a better handle on how actual development is done. Besides learning opportunities regarding the design phase and the coding phase, there have also been new challenges regarding how to choose a programming language to work with. While it is possible to create a web application in wide variety of programming languages, having to do an actual analysis which could impact future decisions were a very good learning experience.

9 Future work

Continuing work with the application involves a lot to do. One of the obvious things is to fix the flaws mentioned in section 7.2. While these flaws might be minor they are still important as they give a professional look to the application.

After the UI of the application has been completed it should be verified by the end user to get their opinions and to ensure that the prototype has the right direction.

Even though the prototype barely resemblance the initial drawing and the wireframes, it is important to take the time to get the prototype verified. Should the elements in the prototype need a change, it can be caught early and some of the work on the backend regarding that specific part may not have been created yet, and therefore does not have to redone.

Because the verification might not happen the day the application is finished, it is possible to start work on the backend of the web application. This can be a difficult task to undertake as some of the components might not be present after the prototype is verified.
As an alternative to creating all the backend functionality it is possible to start researching on an entirely different aspect, as there is also code quality, privacy and security to consider. Theses three topics can be very difficult to implement. Looking at code quality there are numerous different ways to measure this, as there are both static and dynamic measurements, and covering the entire application might not be feasible.
The privacy aspect of an application is important to the user, making the application more trustworthy. Having bad user privacy can cause the user to stop using the application altogether.
Lastly is the biggest of the three aspects, security. Because this application is using and showing sensitive information, it is of the utmost importance that the security of the application is extremely high. If there are any security holes, it may be possible for an attacker to gain access to some of the sensitive information, and use it in a malicious way.

With that said the most important thing is still to create a prototype or product that the user is actually happy and comfortable to use, and thus having the desired functionality.

10 References

[1] Frederik Andersen and Theis Egsgaard Hansen. "User interface for AI-Rehabilitation". Research and Development Project. Aarhus University, Oct. 20, 2020. 13 pp.

[2] Aarhus Universitet. *AIR*. URL: `https://projekter.au.dk/air/` (visited on 10/09/2020).

[3] Yvonne Rogers, Helen Sharp, and Jennifer Preece. *Interaction Design: beyond human-computer interaction*. 5th. John Wiley & Sons, Inc, 2019. 657 pp. ISBN: 978-1-119-54725-9.

[4] *How to Create a Wireframe - A Beginner's Guide to Wireframing*. en-US. URL: `https://www.invisionapp.com/inside-design/how-to-wireframe/` (visited on 12/31/2020).

[5] *What Exactly Is Wireframing? A Comprehensive Guide*. en. URL: `https://careerfoundry.com/en/blog/ux-design/what-is-a-wireframe-guide/` (visited on 12/19/2020).

[6] ProductPlan. *MoSCoW Prioritization*. URL: `https://www.productplan.com/glossary/moscow-prioritization/` (visited on 11/10/2020).

[7] *The Best Web Browsers for 2020*. Digital Trends. Section: Computing. Dec. 10, 2020. URL: `https://www.digitaltrends.com/computing/best-browser-chrome-vs-firefox-vs-safari-vs-edge/` (visited on 12/29/2020).

[8] Google. *Angular*. URL: `https://angular.io/` (visited on 12/29/2020).

[9] Facebook. *React – A JavaScript library for building user interfaces*. URL: `https://reactjs.org/` (visited on 12/29/2020).

[10] *Vue.js*. URL: `https://vuejs.org/` (visited on 12/29/2020).

[11] Hiren Dhaduk. *Best Frontend Frameworks of 2020 for Web Development*. Feb. 10, 2020. URL: `https://www.simform.com/best-frontend-frameworks/` (visited on 11/10/2020).

[12] Duomly. *The best front-end framework to learn in 2019*. June 19, 2020. URL: `https://dev.to/duomly/the-best-front-end-framework-to-learn-in-2019-dn7` (visited on 11/10/2020).

[13] Existek. *Top Front-End Frameworks in 2020 — Existek Blog*. Jan. 14, 2020. URL: `https://existek.com/blog/top-front-end-frameworks-2020/` (visited on 11/10/2020).

[14] Aman Goel. *10 Best Web Development Frameworks*. Hackr.io. Dec. 15, 2020. URL: `https://hackr.io/blog/web-development-frameworks` (visited on 12/20/2020).

[15] Aarhus University. *Poul Ejnar Rovsing - Forskning - Aarhus Universitet*. URL: `https://pure.au.dk/portal/da/persons/poul-ejnar-rovsing(252c9404-1c55-4a92-bd0c-97a201cd414a).html` (visited on 12/20/2020).

[16] Poul Ejnar Rovsing. *Web app framework hjælp*. E-mail. Oct. 6, 2020.

[17] *angular/cli*. npm. URL: `https://www.npmjs.com/package/@angular/cli` (visited on 11/11/2020).

[18] *react*. npm. URL: `https://www.npmjs.com/package/react` (visited on 11/11/2020).

[19] *vue*. npm. URL: `https://www.npmjs.com/package/vue` (visited on 11/11/2020).

[20] Shaumik Daityari. *Angular vs React vs Vue: Which Framework to Choose in 2020*. CodeinWP. Jan. 10, 2019. URL: `https://www.codeinwp.com/blog/angular-vs-vue-vs-react/` (visited on 11/10/2020).

[21] Cordenne Brewster. *Angular vs Vue: Which One to Choose in 2020*. URL: `https://trio.dev/blog/vue-vs-angular` (visited on 11/11/2020).

[22] W3Schools. *React JSX*. URL: `https://www.w3schools.com/react/react_jsx.asp` (visited on 12/20/2020).

[23] Aphinya Dechalert. *Why are developers still using Angular?* Medium. May 23, 2019. URL: `https://medium.com/madhash/why-are-developers-still-using-angular-b9ef29d1f97f` (visited on 11/10/2020).

[24] Zsolt Nagy. *Comparing Angular vs Vue — ButterCMS*. Mar. 17, 2019. URL: `https://buttercms.com/blog/comparing-angular-vs-vue` (visited on 11/11/2020).

[25] GitHub. *GitHub*. GitHub. URL: `https://github.com` (visited on 12/21/2020).

[26] Steve Krug. *Don't Make Me Think, Revisited: A COMMON SENSE APPROACH TO WEB USABILITY*. 3th. New Riders, 2013. 216 pp. ISBN: 978-0-321-96551-6.

[27] *14 Best Vue UI Component Libraries 2020*. en-US. July 2020. URL: `https://athemes.com/collections/vue-ui-component-libraries/` (visited on 12/31/2020).

[28] *Vuetify — A Material Design Framework for Vue.js*. Vuetify. URL: `https://vuetifyjs.com/en/` (visited on 12/20/2020).