Front Page

IS-304: 2020

Title:

Course Code	IS-304
Course Name	Bachelor thesis in information systems
Course Coordinator:	Hallgeir Nilsen
Guidance Counselor:	Ilias Pappas
Client:	Gridd AI Robotics

Students:

Last Name	**First Name**
Otero Barbosa	Xohan
Marvik	Vegard
van der Meij	Yaguel
Trydal	Vegard
Wöllert	Andreas

We confirm that we are not citing, or in another way use others work without it being stated, and that all references are present in the reference list.	**YES**	~~NO~~
Can the thesis be used for educational purposes?	**YES**	~~NO~~
We confirm that all group members have contributed to this delivery.	**YES**	~~NO~~

Summary

This report contains the thoughts and experiences of the group Ingenium's Bachelor project. We present the process that we followed and how we planned to develop our app with our clients Gridd.ai. The first thing that was decided was the method we would use to work efficiently and structured, which was a fairly easy choice since we had good experience with Scrum and working in sprints from previous projects. Something we had good success with. The next part of the project was planning with our client on what they wanted in their app and how it should look. This meant we had to do research and interview our clients and form user stories, requirement lists and form prototypes to present to the clients for approval, which took some time since the client had problems with sickness, but eventually got approved through Slack. Slack is one of the tools we used to communicate with our clients, the reason we used this tool was that the client already had a workspace in Slack and therefore made a channel for us to communicate. Another tool we used was Trello for structuring our work and planning sprints, although we later moved our project to Azure DevOps since some important features on Trello were locked behind a paywall, while through UiA we had access to the premium features of Azure DevOps. Other tools that were used in the project were Discord and Messenger for a more personal communication with our group members, while Zoom was also used with our guidance counselor for meetings. Another important part early on in the project was forming a MoSCoW, which is something that we made with our clients through several meetings to make a priority list of which functions should be added and how we should plan our work. Something that proved very useful as we managed to complete the must-haves and most of the should-haves. While the project was in a good flow with only some slight hiccups something that really changed the way we worked was SARS-CoV-2. This virus went from being a minor risk, where we thought maybe one guy would have to work at home to everyone having to distance themselves and our only communication being online. We can say this had a minor impact on how we worked and how motivated some members of the group were, when we no longer had that physical contact and discussion. While we handled it fairly well with at least 3 meetings a week to plan our sprints and discuss further work. This also meant we had to communicate with our clients and guidance counselor through online services. While it had a minor impact on work effort, it had a major impact on the possibility of testing our product in a relevant environment. In the end, we believe we did a good job based on what was asked and what resources we had. While we know that now, with the experience we have gained, if we were to do it again some things would be done differently.

Table of Contents

Table of Figures

1. Introduction

For our final assignment on our Bachelor's course in information technologies and information systems we were assigned with the task of acquiring a client, make them a product in the form of an IT application then compile a report on this project. We acquired a newly established business as our client for this project, Gridd.AI. Gridd is a newly established business that specializes in what they call Automated Process Management, a process involving the usage of Artificial Intelligence to understand and structuralize the operations within an organization. Gridd brought forward the idea of developing an application that would integrate with their already existing process mapping tool to help new or existing employees with instructions to any process that encompassed their department. We used a hybrid between Scrum and Extreme programming to make sure our project was managed properly and focused on delivering the right quality for the customer, and by using the MoSCoW framework we were able to prioritize our tasks. While managing our project with an agile approach we gained a lot of knowledge on developing a mobile application. We have also learned how to use different technologies both for front-end and back-end development, and different tools for project management.
We chose Gridd because of their enthusiasm as a newly established business, the possibility to experience exciting technologies such as IBM's Watson, and the challenge of working on a new field in our case mobile development.
The result is an application capable of showing new employees how to perform certain tasks as part of their training.

2. Central decisions

In this chapter we describe the important decisions we have made throughout our project, as well as why we made these decisions and their effect.

2.1 Methodology

We have chosen to use an Agile approach so that we could minimize risk by having short iterations and focus on having direct communication with the client. This allowed us to adapt swiftly to the unforeseeable and often changing requirements that system development projects run into (Cervone, 2011). We chose to use a hybrid of Scrum and Extreme Programming (XP) so that we could adapt the methodologies to benefit from the best of both.

2.1.1 Scrum

We chose Scrum since we had used it in previous semesters and it provides many benefits when having to manage system development projects. The parts of Scrum we used to manage our project are making a list of requirements together with the product owner, having timeboxed two week long sprints, and dividing requirements into smaller tasks and then between the different sprints while making a sprint backlog. Also having weekly standup meetings where everybody discussed the work they had done, the work that we planned to do, as well as the difficulties we faced during our tasks. The reason we did not have daily stand up meetings was since we were able to sit next to each other at the office of Gridd. This gave us the possibility to consult and control each other to make sure everybody was making progress and to find out if somebody was struggling. During the quarantine of SARS-CoV-2 we had to change our ways of working together. Instead everybody had to work from home and we had to communicate with the use of digital tools. We also decided not to have a Scrum Master since we are used to working as a self-organizing team and this is also one of the reasons why we chose to combine Scrum with XP.

2.1.2 Extreme Programming (XP)

We chose to use parts of XP to provide and integrate engineering practices that are essential when developing a product that focuses on the customer and quality (Mushtaq & Qureshi, 2012). *"Extreme programming improves a software project in five essential ways; communication, simplicity, feedback, respect, and courage."* (Wells, 2013) Following these principles helped us to collaborate and stay confident even with all the changes that came while working with a visionary client.

We used different practices of XP like pair-programming, whenever one of the group members needed advice or had problems we worked together by either asking for advice or notify others of problems that arized. When somebody was stuck two of the group members sat together where one of them was writing code and the other was watching to get or give advice. This helped to prevent bugs as well as enabling them to discuss and brainstorm while figuring out how to solve the issue. This helped us to improve the quality of our code and solve issues more promptly. It also created an informative workspace where everyone could discuss important information that was related to the project. We also used Trello and later switched to Azure DevOps as digital task boards so that everybody had a complete overview of the project. We wanted to use a wall or physical task board, but sadly this was not available at the space we were provided by Gridd.

2.1.3 Test-driven development

Working with React Native, a framework that was new to us, made it hard to use test-driven development and therefore we chose not to use it. Though, working with React Native gave us the possibility to test our code while we wrote it, since the framework allows the app to refresh and compile instantly. It also provided feedback whenever errors were caught, which helped reduce the number of bugs. We also made use of version control and sandboxing to prevent that code was accidentally overwritten (Stellman & Greene, 2016). At the end of every sprint we merged the new code to our master repository and let all of the group members install the source code that contained the new functionality. By using continuous integration we tested the newly implemented functions between our group members. Also, when possible, we let our client test the app to get feedback. We would have prefered to be able to test with real users, but the client told us that this would not be possible before the end of the project. Instead he would take full responsibility for giving feedback .

2.1.4 Timeboxing

We decided to have iterations/sprints of two weeks since we had experienced from previous projects that this was an appropriate time limit to finish a task. Having short iterations helped us to reflect on our progress and make changes when necessary. We reviewed the features we managed to implement with our team and when possible asked the client for feedback. Whenever we or the client were not satisfied with the result we discussed the matter and made sure we came to an agreement before we proceeded to plan our next tasks.

2.1.5 Flat management hierarchy

We chose to use a flat management structure by dividing the responsibility between all of our group members instead of pointing one person as a Scrum Master. This gave all the group members the opportunity to contribute in taking responsibility as well as enabling every member to come with different perspectives. We have used this structure before and it helps our group to make better decisions. In this democratic way we keep information open without filtering for all the group members to learn from different situations. It also ensures that every member can influence the direction of the project and keeps everyone up to date.

2.2 Quality

Defining Quality in any project is important. To ensure that we have good quality it is important that the criteria is “measurable” and we can test it. This is a central part of the SMART-method, used to define criteria in project management(“What is SMART in Project Management?”, 2020). To achieve this we started off making our MoSCoW. With a MoSCoW we fulfill those criteria set in SMART, as all functions are well defined, measurable and testable. Every item in our MoSCoW has been approved in collaboration with the client and can be seen in the appendix (Attachment 1).

Quality Assurance

Items in our MoSCoW were also divided into more specific tasks when they were not descriptive enough. An example of this is our "Login" item, that was divided into 7 smaller tasks. As an example, one of these was *passing the user ID from the login to identify against the database*. The expected quality for this item would for example be reached once the user ID was attributed to the user automatically after performing a login.With this method we also knew when the expected quality of the product had been reached, as every function had a clear and concise purpose the quality was met when the function performed the task that was defined. The MoSCoW also had the added benefit of making it clear what to test. Must haves had a clear priority over Could Haves to be tested and our tests were performed with this in mind, by focusing on functions with high priority.

2.3 Project management

Managing our project has been a daunting task. So, in order for us to organize, plan, and maintain control of our project we have used certain techniques and tools that we have learned to utilize throughout our study. To get started we arranged weekly meetings to create good communication flow. We wanted to avoid doing extra work or duplications, but we made some mistakes when we forgot to use the same library versions when coding. To best focus our efforts we assigned tasks and duties to all team members, dividing into front end, back end and also SysOps to keep a good communication flow with the clients. Our deadline was set to the end of May. Throughout this project we faced several new challenges and we gained important experience by working on ways to solve them.

2.3.1 Planning

Before we started the project we had introductory meetings with our client to go through the scope of our project. We got an explanation of how their existing system worked and what they wanted us to develop. This gave us an idea of what we needed and during the pre-sprint we made a MoSCoW together with our client so that we could come up with a list of requirements to complete our project. This helped us to prioritize the different work items that our product backlog should include. After having made the MoSCoW we tried to split up the different work items into tasks. We then tried to make estimates as to how long each task would take. Even though we were uncertain of how long the different tasks would take to complete it helped us to try and figure out the size of each task. By knowing the size we were more capable of dividing the tasks into the different sprints.

In the beginning it was hard to estimate how long we would use to complete the different tasks, but after a couple of sprints we started to get more used to planning ahead and our estimates became more accurate the more experience we gained. We had meetings every week on Monday to discuss our progress, which tasks we planned to work on, and whether there were any new tasks that came up, or changes that needed to be made. These meetings helped us to keep control on whether we were on track, as well as making sure nobody was struggling for too long with the task at hand. If there were any challenges that needed to be tackled we asked one of our group members for help. This was done either directly or during the weekly meetings. If we were not able to solve the issue ourselves we asked if the client could help.

2.3.2 Analysis

After having our product backlog prioritized we arranged a semi-structured interview with the client to be able to further analyze the design and functions they wanted us to implement (Attachment 2). With the acquired information we made a navigation map and PACT framework (People, Activities, Contexts, Technologies) which can be found in the appendix (Attachment 3 & 4). After several revisions the client agreed with the design and we were able to start implementing the front-end design with React Native.

2.3.3 Design

We then used the information gained by the interview to make sketches on how we interpreted their ideas for the design. After going through the sketches with the clients we revised the design according to the feedback we got and made a UXPin prototype. We tried to keep a continuous dialog with the client to make sure they were satisfied with the prototype before we started on implementing our code. After several revisions the client agreed with the design and we were able to start implementing the front-end design with React Native.

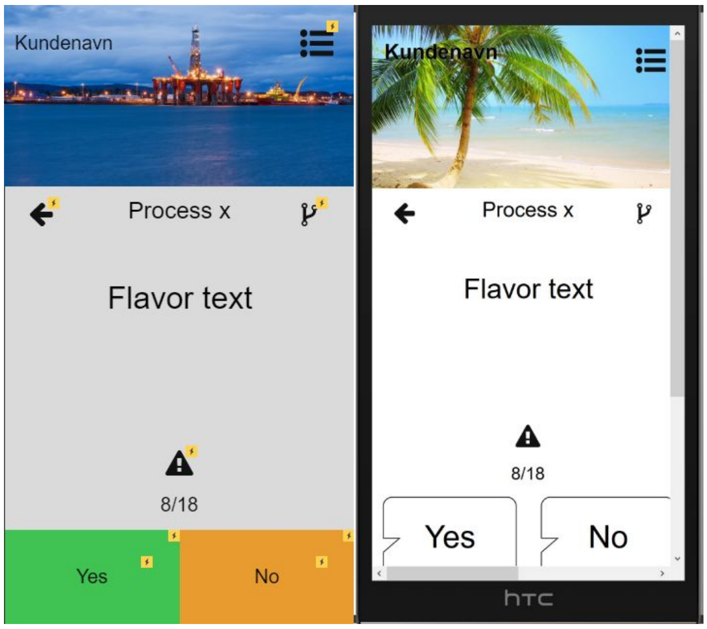


Figure 1: Different UXPin versions of Process Step Page

Displayed above are the model versions from our UXPin developed in the design process. On the left side is the initial version we made digitally for our process page (version 1), and on the right side is the final digital model (version 2). It includes a header with the organization name, buttons for maneuvering through the functionality as well as a text section (with the sample text: "Flavor Text") for where the description of each process step will be displayed. The key difference we made between version 1 and version 2 is a change in colors for the backdrop, as well as on the buttons on the bottom stating: "Yes" and "No".

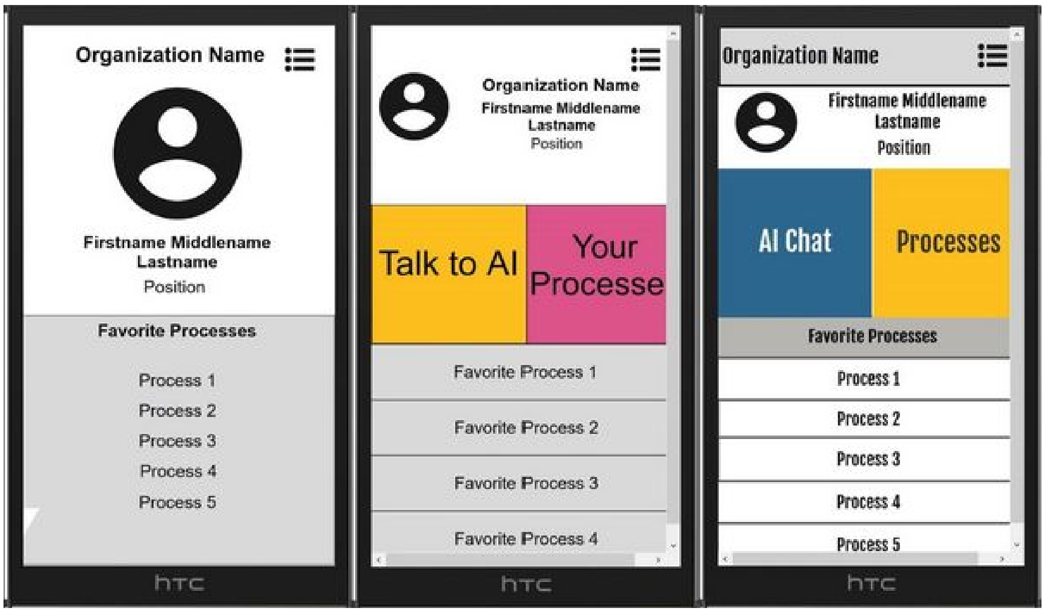


Figure 2: UXPin versions of Home Page

Pictured above is version 1, 2 and 3 of our UXPin models for the app home screen. Starting with the leftmost version (version 1) it is fairly bland color-wise and it lacks lines segregating the different sections of the screen. In our second version we added some more colors and lines separating buttons, as well as adding 2 buttons ("Talk to AI" and "Your Processes"). This was a request made by the CTO(Chief Technology Officer) wanting the two potential main functions of the application being readily available from the home screen. The rightmost model is our final version for the home screen. On the final model we cleaned it up and added some indicators, improving user experience.

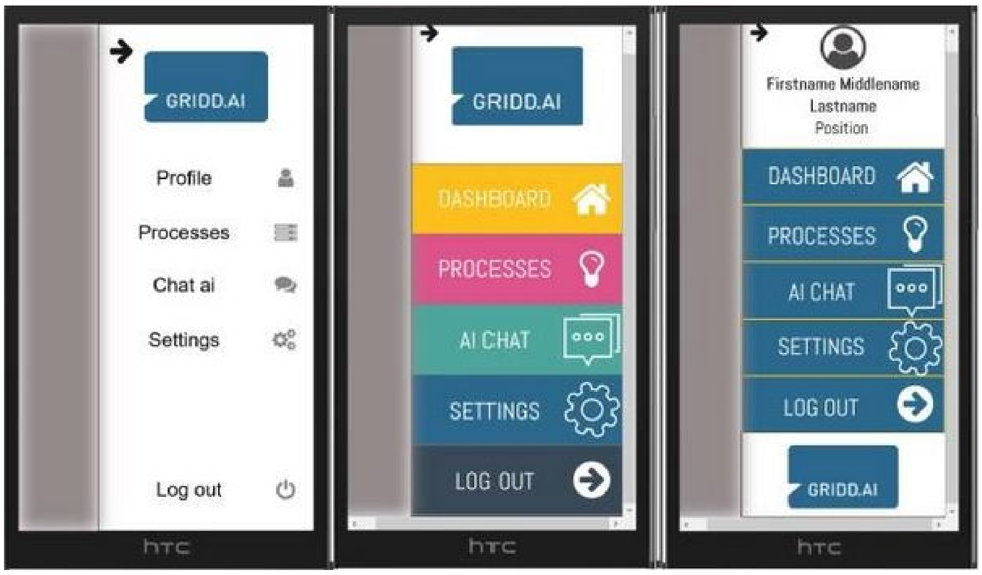


Figure 3: Different UXPin versions of Sidebar Menu

The final design evolution we will cover is the evolution of our sidebar menu design. Version 1 is simple and sterile with no colors apart from the logo at the top. When presented with our initial version the CTO requested more "Kahoot-like" buttons. This change was made on the version 2 where we in true Kahoot fashion blew up the font and icon sizes, as well as adding colors from the company color-palette as the background colors. This change did not appease the CTO who again requested that the colors would be more monotone. For our final version at the right we did just that, making all the buttons blue with a yellow border to separate them. User information was added as well at the top.

All of the changes done to our UXPin models were all done in tandem with our clients, creating a design that they felt comfortable with us turning into an actual product.

2.3.4 Implementation

For the implementation to go as smooth and fast as possible we divided the responsibilities between front and back end developers and one person responsible for various tasks and general communication with the clients. While the front end developers worked on analyzing the design of the app. The back end developers focused on learning how to use GraphQL and IBM's App ID, since these were new technologies we were not familiar with. We used approximately two weeks to learn as much as possible and despite our learning efforts we still had to go through trial and error for us to be able to implement these technologies. Having used extra time to learn helped, but two weeks of learning is not enough to get a full grasp of how a technology works.

Even if we decided to divide the workload and responsibilities we were always working close together and we always helped each other out if any of us got stuck. Communication is essential in a system development project and we have learned that this is very true. We tried to have as much contact with the client as possible which was quite hard, because the IT responsible was often not available. Even so, we had to stay in contact and we switched from direct contact to using Slack. This made sure we could continue with our development.

After a function was completed the person responsible for our GitHub repository merged the code with our master branch and every person of the team pulled the code to their own repositories. By doing so everybody could go through the code and test if everything worked as expected.

2.3.5 Risk analysis

Risk can never be eliminated, but it can be managed. For us to be able to identify and manage risks we performed a qualitative risk analysis (Kerzner, 2013). By considering the probability and impact we got a matrix score so that we could visualize the level of risk involved. Based on the matrix score we could make decisions on how we would monitor and control the situation.

At the beginning of our Risk Analysis planning we primarily thought about scenarios we have encountered in the past or scenarios other groups had spoken about through our time at the university. We divided the analisis into several categories, mainly Technology, Organization or User Input/Commitment. We then accounted for several scenarios we were aware of through a formal risk review, where we added the scenarios we knew about from either colleagues, classmates, or our own experience from earlier work.

One of the risks we identified was the remote repository GitHub becoming inaccessible. We assumed that the probability was low, but half-way through the project it became unavailable. It was only inaccessible for one day, but having planned how to react and control the risk prepared us for a worst-case scenario. Another risk we anticipated was the outbreak of SARS-CoV-2 although we only imagined it would affect individuals of our group. When we first identified this our knowledge of the virus was limited, and we did not know how contagious it was. Even so, our response consisted of "transitioning to remote work" which was still eligible in our case when everybody was quarantined. The client was very helpful to make sure we could continue our work from home. Further results of our risk analysis can be found in the appendix (Attachment 5).

2.6 System

The existing system we were introduced to upon the start of a project was a service written in GraphQL and TypeScript, delivered through an IBM server with various IBM services, including the IBM Watson AI. Although we were free to use whatever language we prefered, the system we created still had to adhere to the existing system for us to be able to connect them together. The Application Programming Interface (API) was written in GraphQL so for us to query the database we would have to use the same query language, and we needed authentication through an IBM service so a solution had to be developed for both iOS and Android.

2.7 IBM's App ID

App ID is a Software Development Kit (SDK) provided by IBM that helped us to add authentication, authorization, and accounting to our app (IBM, 2020). Our client wanted us to implement this in our app to make sure users had to sign in before they were given access to the information in the app. The GraphQL API made by Gridd was also protected with the use of App ID and therefore needed authorization with the use of an access token. By using the access token provided by App ID, we were able to query the API and retrieve the data that were stored in the database. Implementing this was a daunting task since IBM only had SDK's and documentation for the Native programming languages which were Java for Android and Swift for iOS. We spent quite some time on research because we wanted to make sure there was no documentation or examples on how to implement it with React Native. Eventually we agreed that we had to implement it with the native SDK's, because we wanted to make sure the app was properly secured with the latest features. Having to implement it for both platforms was laborious, but helped gain us a lot of experience on how to work with both platforms and integrating it with React Native.

2.8 React Native

For our mobile application we decided to go with React Native, a framework developed by Facebook. React Native is based on the React Framework. React is a framework that was already familiar to some of the members for web development, in turn making the transition to mobile application development with React Native easier.
The *native* part of our chosen framework name comes from it being a native app development framework, meaning that it is downloaded from a phone app store directly and does not require external tools to launch, and has the ability to store data and information into the phone memory or into a dedicated cloud service (MLSDev, 2020).

There are multiple boons in developing native applications, summarized in the graphic below. Unlike a hybrid application a native application builds itself for whatever operating system the device it is downloaded for, allowing the application to use the device's processing power to the fullest, as well as take advantage of modules most people would need to get externally on other devices such as GPS, camera and a touch-screen.

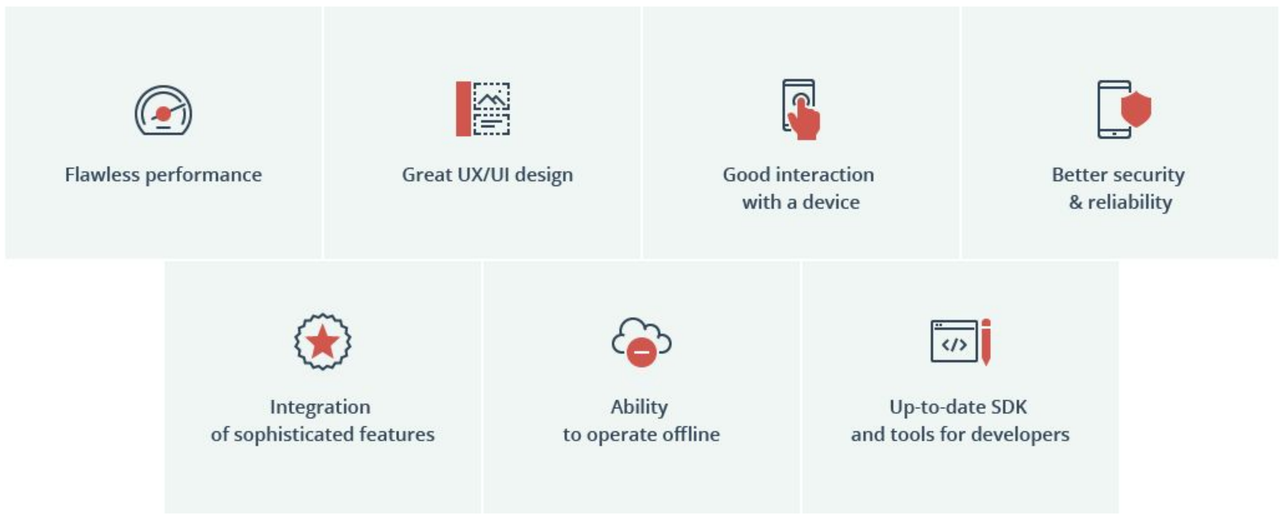

Flawless performance
Great UX/UI design
Good interaction with a device
Better security & reliability
Integration of sophisticated features
Ability to operate offline
Up-to-date SDK and tools for developers

Figure 4: Benefits of native apps (MLSDev, "Pros of Native App Development", 2018)

2.9 Standards

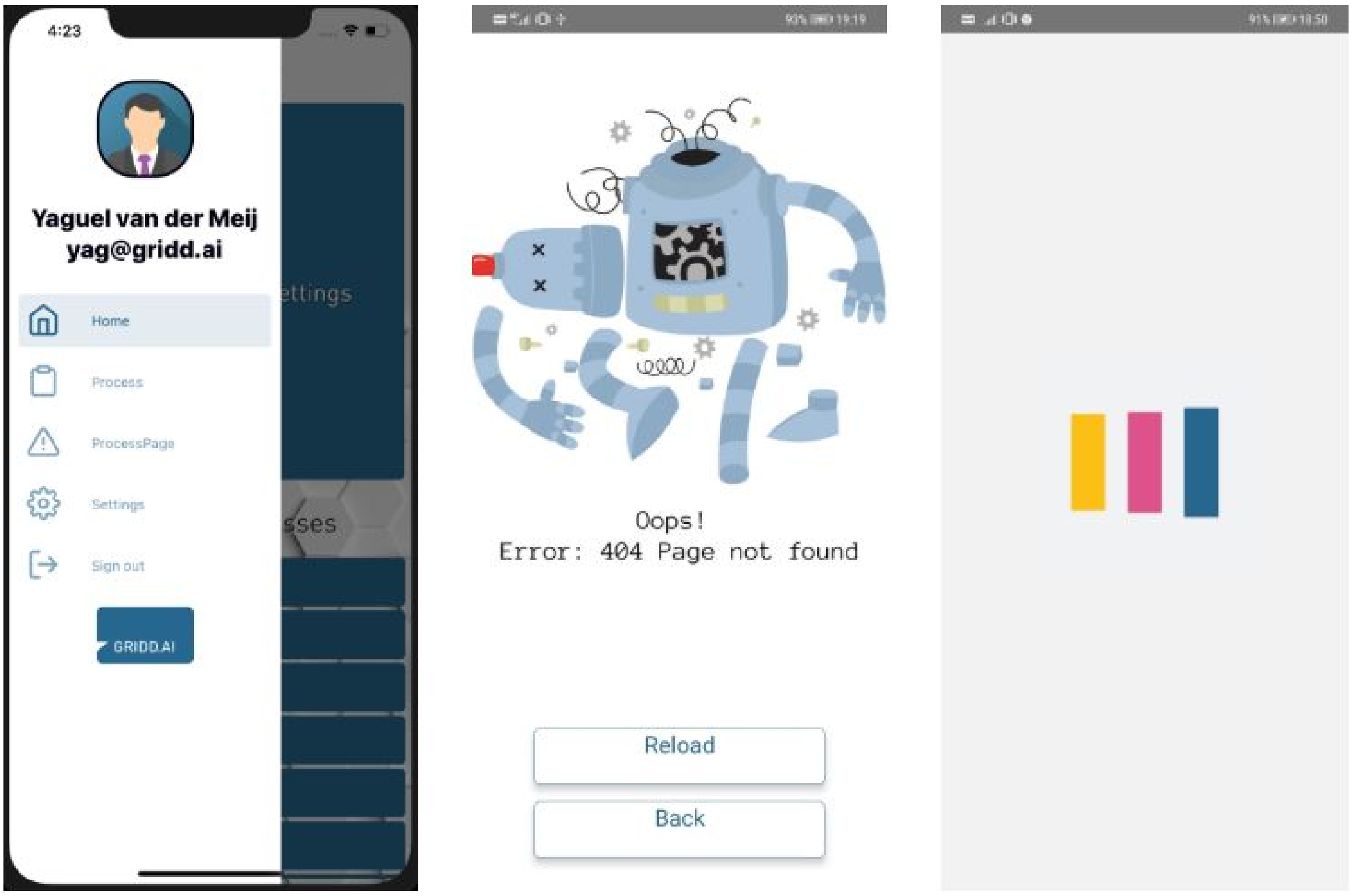
When starting a new project it is important to set standards for how the development will take place. Early on we started setting standards for how we would document or design our application. When we found our standards lacking we made sure we updated them to make sure our product had high quality.

2.9.1 Documentation standards.

For documentation we decided to use Trello and Azure DevOps to save and visualise all tasks for each sprint. Trello and Azure DevOps are tools designed to keep consistent structure to development projects. We also used Excel to register how many hours we used a day to see how much time we used. This was also requested by our customers.

2.9.2 Design standards

We based our design standards on the 10 heuristics by Jakob Nielsen and Rolf Molich. The heuristics were reworked in 1994 by Nielsen (Nielsen, 1994, 152-158) and has since been broadly used in designing interfaces. A few examples are familiarity and visualization of system status. Our menu system is based on familiar icons and functions found in several apps and has been widely used for years now. Loading states and error states are visually shown to give the user adequate information without overloading them.



Oops!
Error: 404 Page not found

Figure 5: Implemented Sidebar Menu, Error Stage & Loading Stage

2.9.3 Coding standards

The main reason to use a coding standard or guide is to make sure that the structure of the code is similar throughout. Making sure everyone follows the same standards will make it easier to navigate, understand, rework, add and remove code. Having the same consistency throughout the code does not only help the reader, but also makes it look professional.

For coding standards, we decided to use what we have learned throughout the bachelor. For variables and method names, we try to use logical names, English as the language and camelCase for the naming conventions. CamelCase starts with lowercase and every new word after starts with uppercase. For classes and components, we use PascalCase, every word starts with uppercase in PascalCase. This sets the standard for how you should write code in our project, but we took it a bit further. We all decided on using visual studio code as our coding software. The reason for this is that there are many great tools available for this software, one such tool being beautify. This will help you put everything together neatly, remove all extra spaces you might have, shorten unnecessary long lines of code by putting it in a new line. Everyone used this in our project, so the code looks similar throughout. Because everyone was relatively new to React Native and their libraries, comments were added to describe what the code does.

When reviewing or helping others with their code there were little formalities. We often sent the code to other group members or we used pair programming. Our only requirement was to follow the standards we had set, and to take breaks, as to not get hard stuck on issues. Everything new added to the code, should be tested and preferably usable. We used extreme programming to some degree in our project to allow for changes. We wanted to keep it simple, flowing and to always have a testable model.

A well-organized project has a good folder structure. React native already comes with a good folder structure, so our only standards here were to follow the setup React Native made for us. If you add assets like a font or a picture, place them in the respective assets folder.

When you use extreme programming in your project you need version control. There are several reasons for using a version control, but no matter what its main function is to prevent or minimize the loss of data. We used GitHub(Github, 2020) for our project. We had a master and 4 branches. 1 person had control of the master and had the responsibility of merging a branch to master. Everyone had their own branch that pulled the master at the beginning of each sprint. When you had done something, tested it, and wanted to deploy it to the master, the person responsible for the master branch would always check your branch and test it before deploying it or merging it to the master. This way the app on master was always working and continuously updated as new features were being made.

For security reasons we do not store any personal information on the phone or in the code. For login we used IBM's App ID to make sure we abide by the law. Since IBM is always up to date with regulations and the latest security features we could be sure that the safety of our users and their data would be high.

2.10 Requirements from clients

The clients of our project did not impose any requirements for what platform and tools we had to use in our system development. In chapter 2.7 we have written about our choice of platform for this project, being React Native. This was chosen over the before the project started, after all of the group members researched and tried out the different options (e.g. Flutter, Ionic). What was required of the application was the ability for it to communicate with the existing systems, meaning it had to be able to fetch information from the database and communicate with their servers.

Features of the app

There were many features that our clients wanted us to include. The first two essential features of the application are as follows:

- *The ability to display processes*

First and most importantly they want us to deliver an application that can display the processes in an orderly and easily understood way for the users. This means that all of the processes created by their *Automated Process Management* is accessible on the mobile application, and users are able to go through these processes stepwise from start to end.

- *Authenticate through the existing system of the business*

Although processes can be displayed by building an API for bridging the gap between the application and the information that is to be displayed, establishing an authentication between the application and their existing systems improves both flexibility and practicality for everyone involved on the server side.

If time allowed there were also other features the clients would like for us to include in the application. These are:

- *The ability to report bugs and deviations within both the app and processes.*

If anything is wrong with either the application or the processes created by their *Automated Process Management,* they would like a way for users to report them back to the system directly from the mobile application.

- *Artificial Intelligence Chat*

The AI used in their systems is made by IBM. If possible, they would like for us to be able to include this inside the application, allowing users to talk to and possibly sharpen the system by reporting their own routines into the system.

- *Notifications displayed when changes occur*

If changes are made to either the application or individual processes our clients would like for there to be sent out notifications to the users highlighting what has been changed, or what changes have been made to the processes and routines.

- *Gamification*

To keep users active and make the mobile application engaging our clients suggested including gamification. An example of gamification is rewarding users for their engagements either with achievements or coupons.

- *Machine Learning query analysis*

A lot of data is generated and queried by the system. By using machine learning it is possible to generate information about the application use and processes. Our clients would like to see if there are any processes or steps that are causing problems, or take more time than they should.

- *Geolocation*

A major boon with mobile applications is the accessibility of a GPS. By using geolocation one could include features like the ability to distribute information or privileges by looking at the user location

3. Running the project

The project was run primarily using the Scrum-methodology, which involves arranging sprints to strategically handle the programming backlog. Scrum is a method we have some previous knowledge in from working with it in previous school projects, this helped us greatly when planning how to run our project and helped us keep a consistent schedule. While scrum was the main way we ran our project, we also adapted certain aspects of the agile method Extreme-programming. The sprints we used were two weeks, so we planned our tasks for two weeks and adapted.f tasks were not completed in the time set, we discussed why and adapted our time schedule with more correct observations. What we used to track these sprints were “Azure DevOps” a project management tool from windows and self made burndown charts. What we experienced when using a familiar method in Scrum, was that work effectiveness increased and we had clear guidelines to follow when tasks were completed or ended up being complicated. This helped us get a better idea of the scope of the project and how we should properly manage our time. While we primarily used Azure DevOps for our sprint organizing it was not what we started out with, we started with “Trello”, this was since it was a known tool amongst several of us and it was free to use. But that changed after we learned from some other groups that you could use Azure DevOps for free through our Student account and as we did not use the premium version of “Trello” we found that Azure DevOps offered more tools to organize so we chose to make a swap, although since this was about a month into the project we felt we should have used more time to adapt to this new tool to get a better understanding of it.

We show our sprint process in (Attachment 6) on how we planned and prioritized different tasks. It also shows a detailed view of our different sprints and what tasks they held. The arrows show the priority.

4. The product

The goal was to implement a mobile app that could show processes generated by Gridd's AI chatbot in a way that they could be used to train employees of different companies. Our end product contains most of the requirements and functionalities we and the client had planned to implement, even if there are some more features we would have liked to implement. Our app includes the functionality of signing in with IBM's App ID, which in itself, enables the user/employee to view the processes that are assigned to him/her from the database with the use of a GraphQL API. The app also provides the possibility to go through the processes step by step, which is necessary to complete the training. The user is also able to report nonconformities found within a certain step of the process, which is then sent back to the database with the use of a GraphQL mutation. We have also implemented the possibility to report bugs, crash-reports or send feedback on improvements. When the user triggers the bug report function they are able to add screenshots, screen recordings, and voice notes to explain the problem they encountered. All of which are made possible on both platforms, Android and iOS.

We have included a file of the video where we go through our app on an actual phone and show the final result of our product. We have also uploaded our video to YouTube and the link can be found in the appendix (Attachment 7).

The link for our uploaded video to YouTube is:
https://youtu.be/LNmFQWWwVt0

5. Reflection

Through the project we have worked on tasks amounting to several different parts of the project with varying degrees of success. After finishing the project we reflected on how the decisions we made ended up impacting our project and how we could have done it better.

5.1 Involvement of the client in the project

This is our first “real world” project and set the stage for what we expect the professional world to be. We were a little uneasy at the beginning whether or not to accept such a daunting task as to develop an app since none of us had previous experience with this type of development, but Gridd reassured us that it would not be an issue and that help would be arranged. Our client is the local company Gridd.AI, spearheaded by Lage Gundersen as CEO and Johan Wedel as CTO. Wedel, being the CTO and head of the technology department, functioned as project owner through this project. Not only would the IT department help us from Gridd, but it was said that they would hire a senior developer to help look over the development for a short period of time. We felt that this would be a great opportunity and went all in.

The client was very involved in the beginning making sure we felt comfortable in their offices, providing us with tools like laptops to work on so that their intellectual property, or IP, would never leave their control. The project owner expressed that it would be beneficial for us to sit in their offices rather than working from home, so that he would have the ability to quickly check up on our progress, and to be able to easily review any changes in the system and design choices. In the beginning of the project in January we agreed with the project owner to have a meeting every Monday. These worked fine the first two weeks but as time went on for unforseen circumstances the project owner became less involved in these. Because of this, feedback towards what we were working on, as well as addition in the requirements or any change at all, kept coming in late. This then had the effect of making the information exchanged in these meetings denser. We made several attempts to contact the project owner to get feedback on time with varying degrees of luck. We also contacted the CEO, Gundersen, to see if he could provide any feedback when Wedel was missing, and while he did provide us with great feedback he also expressed that his lack of technical knowledge was limited and for some we just had to wait for Wedel to become available. We made our best efforts to implement the changes and work on the feedback we got, but we found the experience very discouraging. Sadly, we never got the help we were promised. Early on we were told we could get help from senior developers at a later date, the longer we got into the project the more curious we got as to when that help would arrive. When the pandemic hit halfway through the project we had other things to focus on and in the end we never asked about the seniors and just assumed that it was either forgotten, not possible because of the pandemic, or a financial issue. This was sorely missed, as proper guidance is essential for good development of not only the students, but also the product. Nonetheless, we would soldier on, finding solutions to issues ourselves and keep motivating each other. We learned a lot about handling different kinds of situations and clients. We kept the meetings alive as they were crucial for the development of our project, and we found the meetings we had to be extremely important and very helpful in guiding the development in the right direction.

Another important aspect of involvement of the clients was the steering committee meetings where we sent an agenda to our clients and our guidance counselor a few days in advance. In these meetings we planned to discuss our status of the project, the challenges we were dealing with and general questions from clients and guidance counselors. We planned our meetings a couple of weeks upfront so that we could make sure everyone was available. However, the first two meetings were not fully realized since the client involved in the technical aspects did not manage to attend these meetings. This resulted in the discussion becoming less technical which resulted in less dialog between our guidance counselor and clients which is something we greatly valued. This also meant that the client had to be briefed at another time taking more time from our schedule, time that could be used on development.

5.2 Reflecting on project management

We figured out early that we wanted to use online digital tools to help us manage the project. One of the tools we decided to use was Trello. It is a great tool that helped us a lot, but it did not do everything. We thought that a tool that could document everything we did in the development would be better. Azure DevOps is a tool with these properties, but if you have already started using one tool, familiarized yourself with it and started development, you might want to reconsider swapping tools mid project. The moment we found out about Azure DevOps we changed, in hindsight we should have taken more time to decide and investigate how much time is required to master this tool. Azure DevOps is far more advanced than Trello. We spent a lot of time trying to figure out the ins and outs of this new tool. In the end we figured out that unless you start using Azure DevOps from the beginning of your project it is nothing more than a backlog and glorified Trello board. The automatic burndown charts and other good statistics will not show up properly because you cannot add previously finished work to old sprints. We lost some time doing this but learned a far more valuable lesson that we can take with us. Do proper research before acting and do not rush everything. Sometimes taking a step back might be better.

What we could have done differently next time is to have a shared list of what libraries we are using and what version of that library we are using. Same goes for addons and software. Reasons for this is that it can cause problems if you are not using the exact same version or software. We learned this the hard way when the back end developers wanted to test some features out with the front-end and wanted to make sure it was compatible with the libraries. The same libraries were used, but the version was off by 1. Meaning that some code was written in version 4 and other parts of the code were written in version 5. For some libraries this might be fine, however for the library we were using the version difference of the libraries was almost night and day. The newer version had just come out and had reworked almost everything. This was not noticed straight away, so a lot of code had to be rewritten to make everything work together. It was a good experience, but something we would like to avoid in the future.

Management of time

Time management was something of great importance to us, because as we have learned during our time at UiA if you do not manage it properly it is something that will severely reduce your effort when it comes to a project. So therefore, we planned ahead and had several ideas in mind of how to manage time when it came to the project. The first thing we concluded was that we would treat it like a day job minus the hours that were required at the University, this is something our clients also helped with as they let us work at their location five minutes away from the University each day which proved a great benefit. The way this time was tracked was with two things, the first was an Excel sheet shared with us by our clients where we would report our day to day hours. The other way we managed our time was as previously mentioned through sprints that were tracked in Azure DevOps, where we wrote down assumed completion time of tasks based on previous experience. The sprints were two weeks long with an assumed amount of working hours. Despite hiccups and SARS-CoV-2 doing their best to slow us down, we have achieved a lot during our time at Gridd. We have learned a lot and experienced different situations. Unique for all Bachelor students this year is the worldwide pandemic raging on. Resulting in a more difficult work environment for everyone and forcing us to adapt to meet new challenges. We have little experience doing everything online, so the beginning was a bit slow with problems such as connectivity issues alongside bad equipment and noise interruptions made the beginning rough. To make things better, we met three times a week in group chats to plan work, see if anyone was stuck and decide where to focus our effort. Making small changes like keeping a schedule, and "forcing" people to meet online boosted morale and progress. One of the challenges with working from home was the prioritization of work. When you are at home there are constant temptations and no one monitoring your workday, forcing someone to attend a meeting and showing results can help mitigate some of those temptations. Other challenges can be that you do not live alone. Family, pets and other interruptions can ruin your workflow at any given time. Mastering these interruptions can be challenging. We tried to hold us to the same standards, but it was a lot harder to maintain. When looking at the finished project we agree that we managed our time in an acceptable manner when working at the location, but might have managed it better during quarantine by reinforcing the tool use and having more structured sprints.

5.3 Quality

Our MoSCoW proved to be very valuable when assuring the quality of the application. It clearly defined what the customer wanted, and how they expected it to function. With the use

of a checklist that we had previously defined, we could test these functions against the checklist and check them off as they passed these tests.
These tests were all performed by hand, by going through the application ourselves and with the client whenever he was available. Some of these could have been automated by using the Azure DevOps Test Plans, and running automated tests each time we pushed our code to our repository. The reason we didn't do this was because we started using Azure DevOps further along the project, and were not familiar with all functionalities, as we have previously mentioned.

5.3.1 Code Testing Improvements and Future Testing

Our code has not been tested for scenarios we either deemed less likely or did not account for from our user stories. There are tests we would have liked to perform, such as unexpected inputs, unexpected outputs from the database, or various UI tests. We did not have to consider performance tests, as the application itself does not have to work through any data, only inject it or output it.
In the future we would like to hand the application to users, and ideally see how they use the application. This is often very useful because not everyone interacts with their phone the same way, giving us very valuable data on where eventual bugs might be found. Ahead we will discuss why we were not able to perform such tests even though the application was ready for them.

5.3.2 User tests

Our tests focus on identifying usability problems. We performed a participant-based evaluation by writing a set of tasks we wanted our subject to go through, then observing and taking notes as they stepped through the application (Benyon, 2014) .
We also had some smaller scale tests with the client where we walked them through the application and they gave us the feedback verbally, these tests were early and the feedback focused on small concerns such as text size, font, and color scheme.
Early in the project we made plans for testing the application with actual end users. With our client we planned on visiting work sites from their clients and handing them a version of the application they could test live while we observed and took notes. We wanted to note several things, *how often did the users find themselves reaching for the app?, when they reached for it did they find it useful?* among other things. Before we could perform these the global pandemic of SARS-CoV-2 set sanctions on how the public could meet, making us unable to test this way.
In this situation we discussed with our client other possibilities for testing, concluding that we could test the application with people near us, roommates, friends or even ourselves, as long as no sensitive information was shown. We adapted the application to fit these demands to fulfill the testing.
The results of these tests showed us that the application falls short on some parts, where it is not as intuitive as we planned to use. Our process page had some issues on the "split" button,

that takes a users answer if a process requires it. These buttons stayed on regardless of the process requiring them or not and the users found this confusing.
We found ourselves limited by who we were able to test. We were only able to perform the tests with a very limited number of people, with only 3 people outside the ones directly involved with the project having the chance to take a thorough look. We would have ideally given the application to Gridd's clients to take a look, something we communicated early on with our client to be able to do, unfortunately as time went on this was scrapped because Gridd had to shift away the priority from this because of the global pandemic.

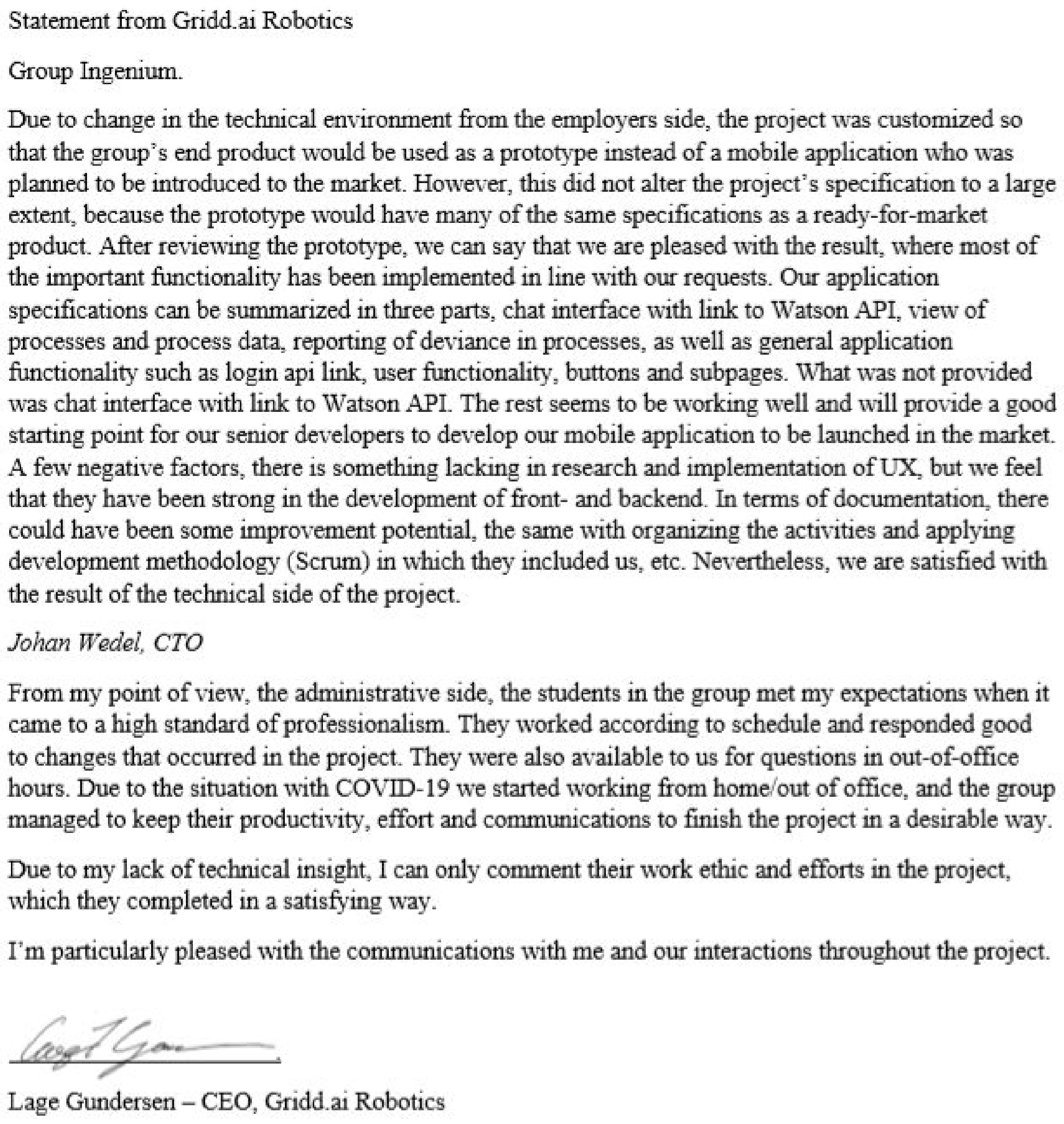
5.3.3 How challenges have been dealt with

Challenges were something we were expecting, as we had little knowledge about mobile development and we had to integrate ourselves with different tools, languages and systems to get a proper understanding of how we should develop the app. While some challenges were expected we faced a major unexpected one in the SARS-CoV-2 virus, which was something that changed the entire process of how we worked. When talking about how we dealt with challenges, they primarily were handled with discussion and consultation, both with clients and our guidance counselor. If we faced a challenge, we discussed what could be done to solve it as fast as possible, and if we did not have an immediate solution, discuss it with clients and our guidance counselor and maybe change the way we look at the challenge.
When discussing how we dealt with challenges we need to mention SARS-CoV-2 which is probably the biggest challenge we have yet faced in a project. The way we dealt with it was to have regular online meetings to discuss important topics and plan our sprints and also regularly update our clients and have bi weekly meetings with our guidance counselor. While we dealt with it the best way we could it clearly affected the way we worked, since we no longer could peer program and some of us lost some motivation when working from home.
Another challenge was communication with the CTO, while the first weeks of our project went fairly smoothly in that department, after a while the client that was responsible for the technical side of the business were regularly sick and/or unavailable. This clearly affected our work as we had less direct feedback from the client and often had to wait days for essential information. Even if the person responsible for giving feedback was hard to reach we tried to keep contact by trying different options such as Slack or by phone. Since we often lacked the feedback we needed we made sure we communicated between our group members and the CEO to be able to continue on implementing the requirements. What we learned from this was that we had to be more proactive and rather than wait for answers we would try to find alternative ways to communicate and get our thoughts and ideas through.

5.4 Working at Gridd

One of the first lessons we learned at Gridd was that things will take time. It was often not enough to ask for something once, it had to be requested several times. Things like these are not unique for Gridd and it did not affect us in the beginning, since we would work with what we had, but later this would limit us in our progress. Since we did not know if it was an access or permission issue right away, we could sit on one issue for a long period of time trying to figure out why something was not working even when it was done right. This put more pressure on the group as we thought the problems lied with us and not because we did not have the correct access. Gridd was only trying to protect its intellectual property by limiting our access to their system, but because of these restrictions and the client not granting the proper access rights and privileges it harmed the progress of the project. We think that to avoid these issues in the future they should set up their Active Directory system in a proper manner with premade groups that have the right accesses and permissions required for the type of work they intend. If we as the hired part can not get the access we need to do proper work the overall quality will fall.

6. Statement from the Client



Statement from Gridd.ai Robotics

Group Ingenium.

Due to change in the technical environment from the employers side, the project was customized so that the group's end product would be used as a prototype instead of a mobile application who was planned to be introduced to the market. However, this did not alter the project's specification to a large extent, because the prototype would have many of the same specifications as a ready-for-market product. After reviewing the prototype, we can say that we are pleased with the result, where most of the important functionality has been implemented in line with our requests. Our application specifications can be summarized in three parts, chat interface with link to Watson API, view of processes and process data, reporting of deviance in processes, as well as general application functionality such as login api link, user functionality, buttons and subpages. What was not provided was chat interface with link to Watson API. The rest seems to be working well and will provide a good starting point for our senior developers to develop our mobile application to be launched in the market. A few negative factors, there is something lacking in research and implementation of UX, but we feel that they have been strong in the development of front- and backend. In terms of documentation, there could have been some improvement potential, the same with organizing the activities and applying development methodology (Scrum) in which they included us, etc. Nevertheless, we are satisfied with the result of the technical side of the project.

Johan Wedel, CTO

From my point of view, the administrative side, the students in the group met my expectations when it came to a high standard of professionalism. They worked according to schedule and responded good to changes that occurred in the project. They were also available to us for questions in out-of-office hours. Due to the situation with COVID-19 we started working from home/out of office, and the group managed to keep their productivity, effort and communications to finish the project in a desirable way.

Due to my lack of technical insight, I can only comment their work ethic and efforts in the project, which they completed in a satisfying way.

I'm particularly pleased with the communications with me and our interactions throughout the project.

[illegible]

Lage Gundersen – CEO, Gridd.ai Robotics

7. Self evaluation

Having to manage our project has been a very rich learning experience. We have gained a lot of new knowledge during the semester and we have managed to make a valuable product for the client. Even if we had zero experience in developing mobile applications we managed to implement all of the most important features. Combining the skills we gained in previous semesters, together with what we have learned in this course, helped us to make sure we were able to deliver a product that not only works, but also has the main components. Being pushed to manage our own project helped us in getting to know what is important to be able to succeed. It showed us the rights and wrongs by trying to do our best with the knowledge at hand. We have gained knowledge not only of what methods to be used, but also things that should be avoided. We have learned how important it is to set goals and to set deadlines to tasks and milestones to make sure the workflow stays optimized.
After having gained this experience and knowledge we also realize that there are certain things we would have done differently, like being more structured from the beginning, or having used a tool such as Azure DevOps from the start. However, the beneficial part of going through the process of making mistakes is that we will be able to use what we have learned for future projects.

Andreas Wöllert
My contribution to this project has been front end based and documentation. We had to learn react native front end. We also had to design how the app should look and function on paper and via UXpin. These concepts had to be approved before we could start working on the front-end coding. Since we had never worked with react native before we faced a few challenges. One such challenge was because a recent update had been released for a library we used, but we used the older version, but it was not good enough documented and someone used the newest version and it was not compatible at all resulting in us having to upgrade the entire app with the newer library. As one should know in development, it is not always beneficial to update to the newest version when the changes are quite radical. This was solved, but caused some extra work. Moving on with the issue of SARS-CoV-2 it has been very difficult to find motivation. We work best as a group when we can meet up and plan our progress. Suddenly moving everything online and digital has been a difficult transition for my motivation, but we try to keep up the same amount of optimism and work effort as to not make it affect our result.

Yaguel van der Meij
The initial plan was for me to be responsible for the back end development of the application. Even so, in the end I have been involved with the full stack of Gridd's system and our application. This includes adding a table to their PostgreSQL database, implementing the changes to their GraphQL API, and then implementing this in the front and back end of our

application. As well as adding all the sign in functionality (AAA) of IBM's App ID to the back-end of our app using Java for Android and Swift for iOS. I have helped our group members with implementing some of the functionality like the Apollo client and getting hold of the GraphQL queries. As well as, passing data through different components with React Native, which was needed in the front-end. I was also responsible for keeping our GitHub repository up-to-date by merging all the completed features and testing whether everything worked as was required. Furthermore, working on the planning, analysis and design by working on the risk analysis, interview, user stories, requirements list, keep track of the product backlog/sprint backlog, etc.
This semester has been a rich experience and it was very nice to get more familiar with how an agile project is run and managed. I have gained a lot of knowledge by learning how to deal with the different challenges we have stumbled upon and the new technologies we used during this project. Of course, there are things that I wish we had done differently, but having gained more experience and knowledge gives me the possibility to use this for future projects.

Vegard Trydal
For this project I have contributed on the front end and on the documentation of the project. Much like the rest of the group, creating mobile apps was an unexplored part of system development, meaning that we both had to learn how to develop and style programs for a mobile phone and we had to learn the react native framework. The design choices were mostly developed by me and Andreas in tandem with our clients, starting off with lo-fi models on both paper and on UXPin. Upon getting our initial models approved by our clients we started on developing the various screens and states for our application. These screens and states were under constant revisions due to feedback from our clients as well as limitations of the platform we were developing on and the framework we were using. It has been a great learning experience being that our first "real-life" project happened to pose as many challenges as this, giving us an insight on what could have been done differently on both ends, as well as giving us an understanding of what hurdles can be thrown our ways post graduation.

Xohan Otero Barbosa
I was involved in the back end development of the project. This included working with the GraphQL API the client had for their previous system. I was tasked with adapting our application so that we could use this API in our application by implementing the Apollo Framework into our existing React Native project. While I did implement Apollo to our project, I found the documentation of the framework to be very confusing, making it very hard to learn. In the end though we only had to set it up once and we documented how, so I have the knowledge should I ever need it again. Along the way I learned how GraphQL works, how the schemas are defined and how the data is fetched. I also learned how to set up a GraphQL API but I did not end up using that knowledge in this project.
I also had the responsibility for the function to step through the processes in the process page. This was the biggest challenge I faced during the project, but also where I learned the most. I

took into use all my previous GraphQL, Javascript and React Native knowledge. Lastly I also had the opportunity to experience first hand the importance of communication.
Communication through the project was a challenge, with situations that could have been avoided if we communicated clearer from the start.
There are things I believe could have been done better, but overall I come out of this experience excited to work on future projects with a lot of newly acquired knowledge.

Vegard Marvik
In this project my contribution was fairly divided between many different tasks while primarily the focus to have a "sysops" role which led me to have more direct contact with our clients and in general getting more information back and forth between the clients and us. I also contributed with some front-end programming in React Native which was a fun challenge, but got a bit limited later on since we needed macs/android phones to program in the respective programs Android Studio/Xcode which is something i did not have. Another main contribution was documentation which included user stories, requirement lists and doing the Risk Assessment with Yaguel. When reflecting on my contribution i wish i could have helped more on the programming part, but i feel like what i had as a task is equally as important to provide the group with a good line between us and the clients. I also know that SARS-CoV-2 was not really beneficial to my work efforts as I prefer working in a physical location with the entire group and having access to our clients each day. But in general it was a very learningful experience that I will bring with me.

8. References

Benyon, D. (2014). *Designing interactive systems: a comprehensive guide to Hci, Ux and interaction design*. (3rd ed.). pp. 220-223. Harlow: Pearson Education.

Cervone, H.F. (2011), "Understanding agile project management methods using Scrum", *OCLC Systems & Services: International digital library perspectives*, *27*(1), pp. 18-22. https://doi.org/10.1108/10650751111106528

IBM. (n.d.). What is App ID?. Retrieved April 21, 2020 from https://www.ibm.com/cloud/app-id

Kerzner, H. (2013). *Project Management: A System Approach to Planning, Scheduling and Controlling* (11th ed.). New Jersey: Wiley Publishing, Hoboken.

MLSDev, A. L. (2018). *Summarization of Native App Development pros*. Retrieved from https://d32myzxfxyl12w.cloudfront.net/assets/images/article_images/29842da186f5782dcee48e3bd8e3ed5fc2cde303.png?1541076686

Mushtaq, Z., & Qureshi, M. R. J. (2012). Novel Hybrid Model: Integrating Scrum and XP. *International Journal of Information Technology and Computer Science*, *4*(6), pp. 39-44. doi: 10.5815/ijitcs.2012.06.06

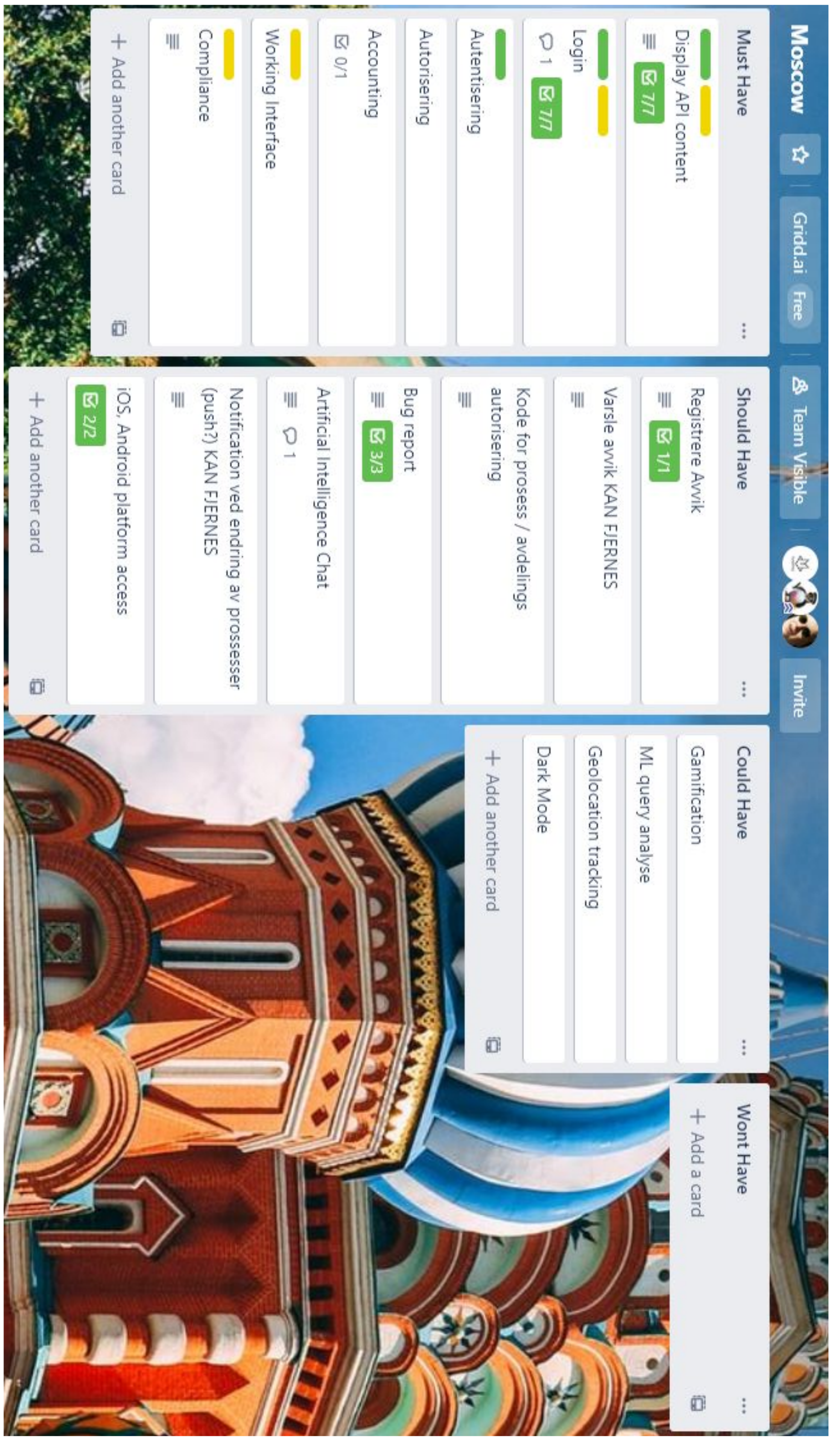
Native App Development vs. Hybrid and Web App Building. (2020). Retrieved from https://mlsdev.com/blog/native-app-development-vs-web-and-hybrid-app-development

Nielsen, J. (1994). *Enhancing the explanatory power of usability heuristics*. Proc. ACM CHI'94 Conf. (Boston, MA, April 24-28), pp. 152-158. https://dl.acm.org/doi/10.1145/191666.191729

Stellman, A., & Greene, J. (2016). *Learning Agile*. Sebastopol, CA: O'Reilly.

9. Appendix

Attachment 1 - MoSCoW


Moscow

Gridd.ai Free

Team Visible

Invite

Must Have

- Display API content — 7/7
- Login — 1 — 7/7
- Autentisering
- Autorisering
- Accounting — 0/1
- Working Interface
- Compliance
- + Add another card

Should Have

- Registrere Avvik — 1/1
- Varsle avvik KAN FJERNES
- Kode for prosess / avdelings autorisering
- Bug report — 3/3
- Artificial Intelligence Chat — 1
- Notification ved endring av prosesser (push?) KAN FJERNES
- iOS, Android platform access — 2/2
- + Add another card

Could Have

- Gamification
- ML query analyse
- Geolocation tracking
- Dark Mode
- + Add another card

Wont Have

- + Add a card

Figure 6: MoSCoW

Attachment 2 - Interview with the clients

Interview Questions

1. <u>Hvor ser du for deg at en slik app vil brukes ?</u>

J:Typisk bruker ansatt i en stor virksomhet, statlig evt privat. Ansatte vil bruke appen når nyansatte trenger opplæring. Eller vanlig ansatte som ser prosessene de driver med for kvalitetssikring. Opplæring, oppdateringer til prosesser, snakke med AI.

L: Kan brukes hvor som helst.

2. <u>Hvordan ser du for deg at en app skal brukes?</u>

J: Rapportere avvik etc se arbeidsoppgaver. Bruker trenger info eller appen trenger bruker til å snakke med den.

L:Erstatte desktop? Ikke helt nødvendig. Brukes i tillegg til desktop. Tilleggsverktøy til å brukes i andre scenarioer. Komplimentere desktop funksjonere. Bedre å trykke seg videre på prosessene. Greit med modellene men kan også ha tekst som kan manøvrere frem og tilbake. Kan ha et "hjul" for manøvrering.

3. <u>Hva føler du en slik app bør ha?</u>

J: Personvern. Brukerkonto knyttet til App AI (bilde, navn, passord) Enten blir de invitert av firma for brukernavn/mail, eller så legger firmaer til nye brukere, evt alle ansatte får sine brukere integrert i adminpanelet. Optimalt er å sende ut mail til en liste(CSV) som ber folk om å registrere seg på App AI. Førstegangsregistrering-varsel via SMS.

L :Chat, en funksjon der du kan se selve prosessen, og en der man kan trykke på hva man gjør.Tracke tid, for eksempel hvor lenge en ansatt bruker på en prosess. Avviksrapportering.

4. <u>Hvem kommer til å bruke denne appen?(admins/brukere)</u>

J: Alle aldre (gamle travere) og nye ansatte. Lagt opp til å være enkelt for alle, nyregistrerte skal optimalt snakke med AI ved innlogging for å digitalisere og normalisere prosessene.

L: Trenger ikke å ha ulik Admin/bruker tilgjengelighet.

5. Hvordan ser du for deg at appen skal se ut?

J: Blå grønn følg primærfargene til gridd. Detaljer kan gjøres med kontrastfarger (gul). Moderne og simpelt (apple design (os6), moderne windows, clean, lavmælt og intuitivt(kahoot) med business preg). Elementer av self-gratification via "gamification"/achievements/skryt/belønning ved utførelse av ting som: send avvik, fullfør prosess, snakket med AI. LinkedIn er et eksempel (profil utfyllelse og kontaktskaping). Månedens beste på prosesser?

L: Lik der vi holder på, synes den ser ganske fin ut kan fortsette ut i fra der.

6. Skal det være ulike tilganger til ulike brukere?

J: Nesting av prosessene, prosessene blir "eid" av ulike bransjer. Hvis du er lagt inn i en prosess ser du bare de.

L: Nevnt ovenfor

7. Hvis ja, hvilke tilganger ser du for deg ulike brukere skal ha?

J:----

L: Nevnt ovenfor.

8. Skal appen tilpasses til ulike brukere? evt store knapper/stor text

J: Veldig enkel, skal allerede være innafor for å bruke for de fleste. Folk med problemer kan gå på desktop. Tydelige farger. Tydelig tekst pen design. Ikke ekstreme kontraster med nok til at folk kan se. Dark Mode hvis tid.

L: Nei? Skal være tilpasset nok fra før. Og fin for vanlige brukere.

9. Hvordan ser dere foran dere at brukere skal logge seg inn? (Flere eksempler)

J:

L: App id? Synes den er en grei løsning. Mulig med integrering av å logge på med biometrisk data(Fingeravtrykk/ansiktsgjenkjenning)

10. <u>Hvordan skal prosessene fremstilles? (bilde/tekst)?</u>

J: Oversikt på prosessen. Mulighet for å collapse ja og nei. Ha alle “text” deler i en xml fil. Blokk format, scrolle gjennom men kan expanded horisontalt. Utfordring i splitten? Trykker du på en av boksene får du en fullverdig view av prosessen. Ikon med tekst som er beskrivelsen av det som skjer.

L: Nevnt ovenfor.

11. <u>Hvordan ser du foran deg at app ikonet skal se ut?</u>

J:Hvis det er mulig(funker ihvertfall på android) med transparent så vil han ha hvit med blå GriddGo med gjennomsiktig kutt.

L:Kan ha hele navnet men er ellers opp til dere.

12. <u>Ikoner for lett manøvrering?</u>

J:

L: Viktig med kryss evt tilbake knapp for å kunne komme seg ut.
L: Guide? hadde vært kult.

Mål: <u>Viktigste er at appen er enkel å bruke</u>**.**

Attachment 3 - PACT

PACT Framework

<u>Preface</u>

In this project we were tasked with the development of an application that companies can use as a tool to help them educate new employees. The app should make the learning experience faster, and lower the resources needed to teach a new employee. Our thought is that there are many companies that want such an app, but developing an app is not always cheap or easy. That's why developing one app that can work for many companies could be the solution. Therefore, the goal of our project is to create this app for our clients and according to their needs. At first, we try to apply the PACT framework, attempt to describe our thoughts in regard to the various elements included in our work, and investigate user needs in relation to our project. Then we decide on the method of obtaining further information for what functions our app should include.

Our Goal: To create an app that satisfies the needs of our clients.

The PACT Framework

In using the PACT (People, Activities, Contexts, Technologies) framework, we try to understand who would use our app, how they would use it, and what needs they would have regarding it (Benyon, 2014). We also want to try to determine the scope of what the app needs to be in terms of design and technology, to create a useful framework that we can use to design our app.

People:

The key topics in the People part of PACT are physical differences, psychological differences, social differences, ergonomics, and mental models. We have incorporated mental models under the other categories and discuss some of the related models.

Physical differences

The app is going to be used in a variety of different situations such as: training new employees, training employees for new technologies or changes in their work environment/processes, and when an employee gets a different position. This means that our app will most likely be used by people aged between 18 and 65. We therefore need to take into consideration some physical handicaps and others such as color blindness, deafness, etc. What we will focus on will be on the average person, but it should be open for everyone according to their own abilities. However, there may be people with reservations training with people with certain handicaps, but we want to include this in our interviews to check if this is a big or small problem.

Psychological Differences

The app is meant to be used in several different industries, many of which have no requirement of being proficient with modern mobile technologies. Therefore we find it appropriate that the design be simple and intuitive, with all fields clearly marked with what they do and should be understood from a quick view at the text/icon.

Even though the app will initially be rolled out in a norwegian market, because the variance of industries it will be used in we assume some people of an international background will use the app and therefore the interface will be written in English. We will not translate it initially, although it could be done later if needed.

Since the app is to be used among professionals/people in the field, all information entered into the app is provided to us by their employer, meaning we do not have to actively think about the personal data users might enter since they do not give it to us directly.

Ergonomics

The app is meant to be used on a mobile, which is ergonomic because of its size. Though, since our app will be used on the go, it would be most convenient with one-handed use. This is important since the more convenient the app is, the more it will increase the productivity of the employees that will use the app.

Social Differences

Initially our goal is to make the design so comprehensible that there shouldn't be a big difference between new users and experienced users regarding their ability to use our app. Certain types of people may be deterred from using complex apps. Therefore, making it quick and easy to use is key, in our case.

Activities:

"Activities" relates to the various activities we need to consider, whether they are quick and simple or lengthy and complex. The main characteristics of activities are temporal, cooperation, complexity, safety-critical, and the nature of the content.

Our goal is to make a focused and easy to use app, which should not be very complex, yet have the features necessary. How often people use it may vary, but new employees will likely use it a lot the first few weeks to help them adapt to a new workplace. More seasoned employees might only use it once in a while to refresh their memory or when there are major changes at the workplace.

The app should be usable in all activity situations for example while sitting in a cafeteria or on the bus, or when you are doing the assigned task, by looking at the app first, then performing the task.

This app is meant to be used as a tool by new and existing employers and employees. We assume that this app will be used frequently by all parts of the company, especially for new employees and when minor or major changes occur at the workplace.

Regarding eventual interruptions people may experience, Writing/talking with the chat bot. As for now we do not think continuous saving of information is necessary.

The response time of the app should preferably be fast for most functions such as pressing buttons, and should be faster than 100 milliseconds. It is best for the app to be reasonably fast and able to find the results within a second. This is to prevent any frustration or confusion users may experience if the app is reacting too slow.

The app is only meant to be used personally, meaning that it will not be usable as a cooperation tool where different people can edit or fill in information simultaneously. Since it is a personal app it will also not be used by two people on one screen at the same time.

Because we expect some users to use the app very frequently, there should not be any complex details. It is preferable that the app is easy to learn, without difficult functions for relatively simple activities.

Spelling correction is an element we do not need to consider, as smartphones and most modern devices have this feature built in, therefore our app can rely on the device to handle this task.

Most of the safety related problems would be related to giving away private information, and corporate secrets. The security issues will most likely be related to hacking. We expect responsible use from our users. (Proper training should be given.).

The activity of using the app should be simple, you should be able to log in then see a page which directs you to the processes you want to find, this should only require a few steps and should not be difficult to execute as the app should be usable by a rage of different people with different experience levels. The expected peak of the app is unknown as of now, although we can expect a few thousand. This might change in the future, hopefully the users will be recurrent as it is a product that the corporations will want the employees to use and report to.

The frequency at which the app should be used is to be defined by the corporations using it, we assume it should be used daily as the processes might change and the employees should keep themselves updated and have the opportunity to report faults or errors of the process at any time.

Contexts:

The context relates to the context surrounding the activities, such as where and why the application is used, and in what environment. The main topics of contexts are physical environment, social context, and organizational context.

Physical context:

On the phone in a work environment, this might include outdoors working on construction sites, inside working on accounting or revisions, being a volunteer working at a festival or working in the warehouse of a chainstore.

Social context:

The social context of the app is to be used in a work environment. You will often be alone when using it as it is to help and guide you in your work processes, and potentially for you to report changes or faults in the process.

Organizational context:

We assume that the app will most likely be used by corporations that have a large amount of employees. As the app aims to make processes more effective in that it should not need for employees to ask their supervisors what to do as often as they do now.

Technologies:

Technologies relate to what type of hardware and software the users will utilize. We need to consider the various possibilities for input, output, communication, and content.

For input, we aim at using text using a touchscreen with a keyboard function on handheld devices. The sources of output will mainly be the device screen, notification sounds, and maybe haptics. As handheld devices rely on Wifi and phone networks, these will be the methods of communication. The contents consist of information that is uploaded and stored in the system, and should be retrieved when required in the form of processes shown as text.

<u>Methods of Understanding</u>

Our initial thought is to start things off with a semi-structured interview because it provides more detailed information, it is open to new questions if we get unexpected answers, and we can get some personal opinions as well.

We want to be able to emphasize on how the interviewee feels about our app. The goal of this is to get information we might not have already, and give us specific ideas for the further development.

Benyon, D. (2014). PACT: A framework for designing interactive systems. In *Designing interactive systems: A comprehensive guide to HCI, UX and interaction design* (pp. 25-36). Harlow (England): Pearson.

Attachment 5 - Navigation map

Navigation map proposal. Ver1.0 (23.01.2020)

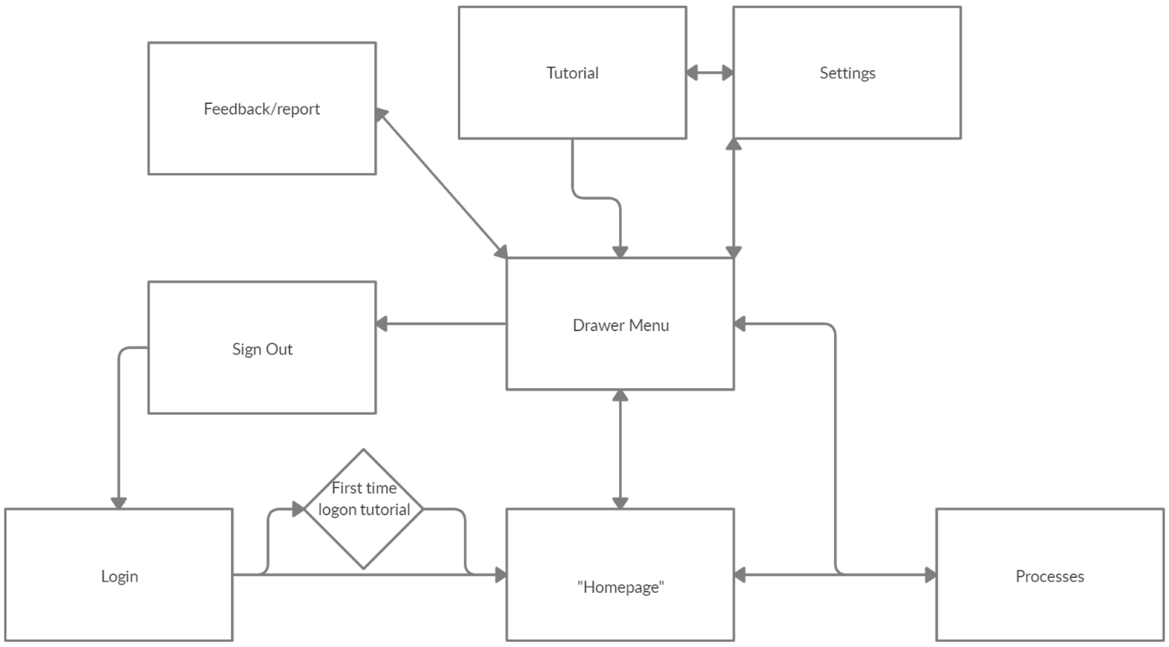


Figure 7: Navigation map proposal. Ver1.0 (23.01.2020)

Attachment 6 - Risk Analysis

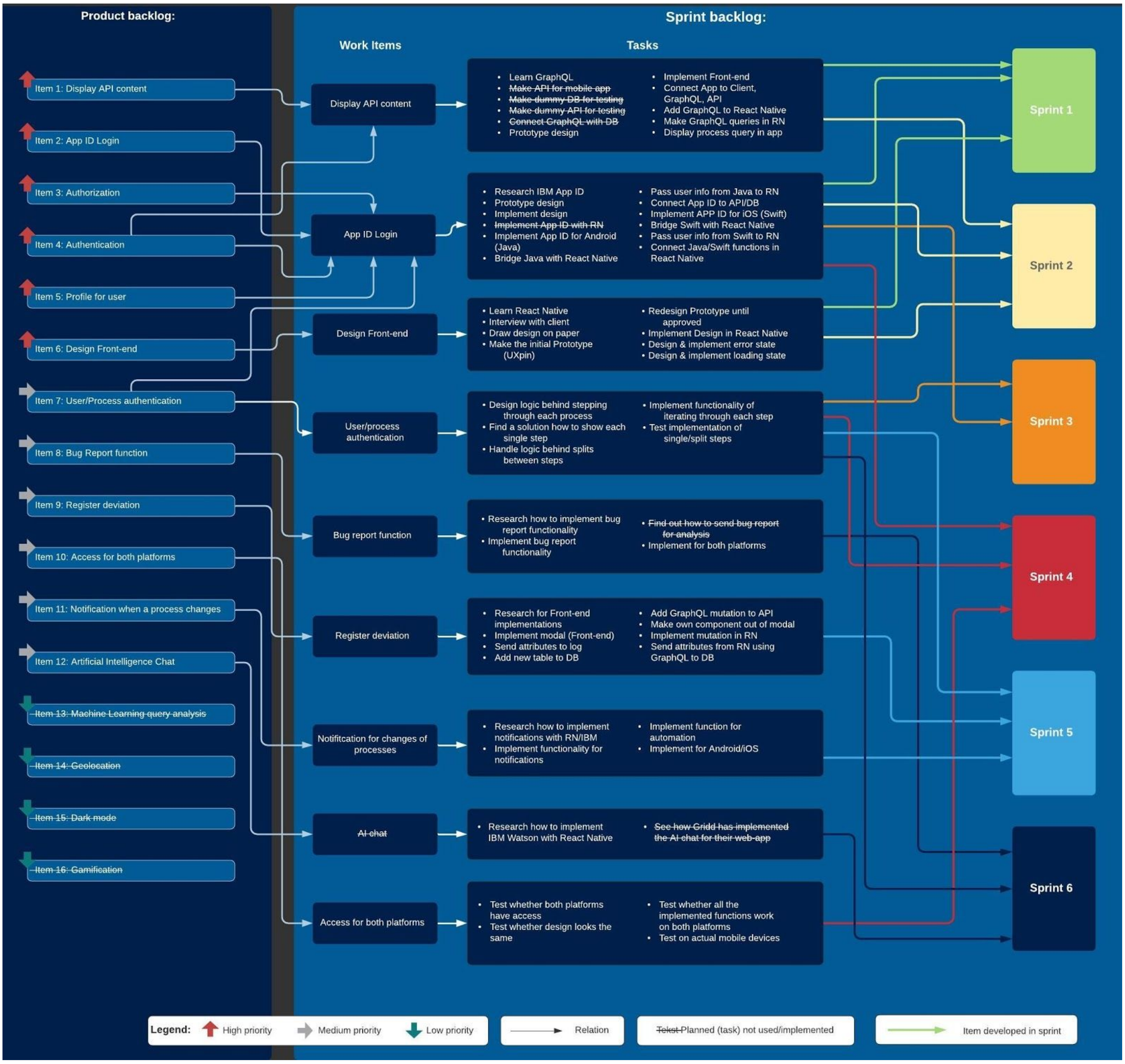
Link to our Risk analysis (please read the instructions and readme included in the file):
https://drive.google.com/file/d/1Ar0fMKG5IF5zwmlkJwXSJR1iswiRzx_B/view?usp=sharin
g

Or have a look in the added Excel file.

Attachment 7 - Chart of Product Backlog & Sprint Backlog

The image below gives an idea of what the chart looks like.

Please have a look at the attached pdf file to be able to see all the details.



Product backlog:
Item 1: Display API content
Item 2: App ID Login
Item 3: Authorization
Item 4: Authentication
Item 5: Profile for user
Item 6: Design Front-end
Item 7: User/Process authentication
Item 8: Bug Report function
Item 9: Register deviation
Item 10: Access for both platforms
Item 11: Notification when a process changes
Item 12: Artificial Intelligence Chat
~~Item 13: Machine Learning query analysis~~
~~Item 14: Geolocation~~
~~Item 15: Dark mode~~
~~Item 16: Gamification~~
Sprint backlog:
Work Items
Tasks
Display API content
App ID Login
Design Front-end
User/process authentication
Bug report function
Register deviation
Notifitcation for changes of processes
~~AI chat~~
Access for both platforms
Sprint 1
Sprint 2
Sprint 3
Sprint 4
Sprint 5
Sprint 6
Legend:
High priority
Medium priority
Low priority
Relation
~~Tekst~~ Planned (task) not used/implemented
Item developed in sprint

- Learn GraphQL
- ~~Make API for mobile app~~
- ~~Make dummy DB for testing~~
- ~~Make dummy API for testing~~
- ~~Connect GraphQL with DB~~
- Prototype design
- Implement Front-end
- Connect App ID to Client, GraphQL, API
- Add GraphQL to React Native
- Make GraphQL queries in RN
- Display process query in app

- Research IBM App ID
- Prototype design
- Implement design
- ~~Implement App ID with RN~~
- Implement App ID for Android (Java)
- Bridge Java with React Native
- Pass user info from Java to RN
- Connect App ID to API/DB
- Implement APP ID for iOS (Swift)
- Bridge Swift with React Native
- Pass user info from Swift to RN
- Connect Java/Swift functions in React Native

- Learn React Native
- Interview with client
- Draw design on paper
- Make the initial Prototype (UXpin)
- Redesign Prototype until approved
- Implement Design in React Native
- Design & implement error state
- Design & implement loading state

- Design logic behind stepping through each process
- Find a solution how to show each single step
- Handle logic behind splits between steps
- Implement functionality of iterating through each step
- Test implementation of single/split steps

- Research how to implement bug report functionality
- Implement bug report functionality
- ~~Find out how to send bug report for analysis~~
- Implement for both platforms

- Research for Front-end implementations
- Implement modal (Front-end)
- Send attributes to log
- Add new table to DB
- Add GraphQL mutation to API
- Make own component out of modal
- Implement mutation in RN
- Send attributes from RN using GraphQL to DB

- Research how to implement notifications with RN/IBM
- Implement functionality for notifications
- Implement function for automation
- Implement for Android/iOS

- Research how to implement IBM Watson with React Native
- ~~See how Gridd has implemented the AI chat for their web-app~~

- Test whether both platforms have access
- Test whether design looks the same
- Test whether all the implemented functions work on both platforms
- Test on actual mobile devices

Figure 8: Sprint Backlog Chart

Attachment 8 - Group Contract

Group Contract

Group: *Ingenium*

Andreas Wöllert, Andrew17@uia.no

Yaguel van der Meij, Yaguev17@uia.no

Xohan Otero Barbosa, Xohano17@uia.no

Vegard Trydal, Vegart13@student.uia.no

Vegard Marvik, Vegarm17@student.uia.no

Vi er en demokratisk gruppe hvor alle har ansvar for delta og engasjere hverandre.

Mål

A som standpunkt i Bacheloroppgaven.

Rolleoppgaver:

Ingen får endre på noe uten godkjennelse fra gruppa.

Møtetider:

Oppmøte er obligatorisk etter avtale.

Alle skal møte opp på avtalte tider med mindre det er gode grunner til at de ikke kan. Oppmøtetider kan endres til å passe timeplanen.

Hvis et medlem kommer for sent til avtalt møtetid er de nødt til å si ifra uansett. Det er relativt greit å komme opp til 10 minutter for sent, så lenge man sier ifra. Likevel skal alle prøve å møte opp til tide så godt de kan.

Arbeidsmengde:

Alle har ansvar for å jobbe individuelt minimum 10 timer per uke. Først og fremst er det krav på at alle deltar med sin del av hver oppgave.

Konsekvenser:

Hvis man møter sent tre ganger, møter over 10 minutter for sent, hvis en ikke gjøre sin del, eller ikke møter opp, kan gruppen avgjøre om det skal få konsekvenser, innenfor rimelige rammer. Hvis noen ikke gjør sin del vil de ikke bli med på innleveringen for den oppgaven de ikke har deltatt på.

Gruppen kan bestemme om et gruppemedlem skal hives ut hvis de ikke deltar i gruppearbeidet. Gruppen må bli enig om valget og advarsel må gis på forkant.

Aksepterte avvik:

Hvis man har en god grunn for å avvike fra kontrakten så kan konsekvenser eventuelt frafalle. Dette gjelder for hendelser utenfor ens egen kontroll.

Man kan i så fall kompensere med ekstra arbeid for å gjøre opp for der man har sviktet.

Annet:

Det er viktig at man sier ifra om det er noe man er uenig med. Ikke hold det inne. Si ifra.

Hvis det er problemer sosialt i gruppen, skal vi sammen prøve å løse det og holde god stemning innad i gruppen. Om det skulle bli veldig vanskelig, så tar vi det opp med lærer for hjelp til å løse konflikten.

Attachment 9 - Steering Committee Meeting Example

Steering committee meeting

Date 06.05.2020 **Time** 13:15 - 14:20 **Location** Zoom

Attendees

Yaguel van der Meij	Xohan Otero Barbosa	Vegard Marvik	Vegard Trydal
Andreas Stubberud Wöllert	Illias Pappas	Johan Wedel	Lage Isachsen Gundersen

Link to zoom meeting:
https://zoom.us/j/5654343354331

Plan for the day:

13:00 - 13:05	Welcome/Introduction
13:05- 13:25	Status
13:25 - 13:45	Challenges of the project/Covid-19
13:45 - 14:00	Conclusion of the project
14:00 - 14:15	Questions/Discussion
14:15 - 14:20	Ending the meeting

NB: Scheduled times might be subject to change according to the flow of the meeting.

Sted: Kristiansand Dato: 13.01.2020