Eco-driving assistant application

By combining interaction design with cloud computing and machine learning

Amanda Eliasson and Robin Timan

DEPARTMENT OF DESIGN SCIENCES
FACULTY OF ENGINEERING LTH | LUND UNIVERSITY
2018

MASTER THESIS

Eco-driving assistant application

By combining interaction design with cloud computing and machine learning

Amanda Eliasson

Robin Timan

LUND UNIVERSITY

June 7, 2018

Eco-driving assistant application

Published by
Department of Design Sciences
Faculty of Engineering LTH, Lund University
P.O. Box 118, SE-221 00 Lund, Sweden

Subject: Interaction Design (MAMM01)
Supervisors: Kirsten Rassmus-Gröhn, department of Design Science, Lund University & Jacek Malec, department of Computer Science, Lund University
Examiner: Joakim Eriksson, department of Design Science, Lund University

Abstract

The global environmental impact made by cars is substantial and will always be in focus when discussing climate change and our carbon footprint. Eco-driving has long been a given set of general rules for drivers to follow such as smooth acceleration.

We wanted to explore the use of machine learning to identify and learn unknown driving patterns that may affect the fuel consumption. We also wanted to explore how communication between the driver and the application should be designed so that it doesn't disturb the driver while driving and how guidelines should be illustrated to motivate the driver in driving eco-friendlier. We did it by exploring the use of gamification as a motivational tool.

We investigated different machine learning techniques and explored each method's limitations and possibilities. We also tested different design alternatives for the application to see which design is both suitable for driving and motivates the driver to reach their eco-goal.

We found that there are many ways to apply machine learning for eco-driving purposes and each method has its own set of pros and cons. In this report we provide no single right answer to how to apply gamification and machine learning in a driving environment but rather a proof of concept to follow for further development.

Data-driven development has many applications in real-world problems and eco-driving is one of them. We learned that personalizing feedback and displaying it with a gamified design encouraged drivers to be motivated into reaching an eco-friendlier driving style.

Keywords: Machine learning, Gamification, Android, eco-driving

Abstract

Bilars globala miljöpåverkan är substantiell och kommer alltid vara i fokus när man diskuterar klimatförändringar och vårt koldioxidavtryck. Eko-körning har länge bara varit ett givet antal av riktlinjer åt förare, om till exempel mjuk acceleration.

Vi ville utforska användningen av maskininlärning för att identifiera och lära okända körmönster som kan påverka bränsleåtgången. Vi ville också utforska hur kommunikationen mellan föraren och applikationen bäst designas så att den inte stör föraren under körning och hur riktlinjer ska illustreras för att motivera föraren att köra mer eko-vänligt. Vi gjorde detta genom att utforska spelifiering som motivationsredskap.

Vi undersökte olika maskininlärningstekniker och utforskade varje metods begränsningar och möjligheter. Vi testade även olika designalternativ för applikationen för att se vilken design som är både lämplig under körning och motiverar föraren att nå sitt eko-mål.

Vi kom fram till att det finns många sätt att applicera maskininlärning för eko-körningssyften och att varje metod har sina för- och nackdelar. I den här rapporten kommer vi inte bistå med ett enda rätt svar på hur man applicerar spelifiering och maskininlärning i en bilmiljö utan snarare ett bevis på ett koncept att följa för vidareutveckling.

Data-driven utveckling har många appliceringsområden i verkliga problem och eko-körning är en av dem. vi lärde oss att genom att personifiera återkoppling och visa upp det med en spelifierad design kan förare motiveras till att uppnå en mer eko-vänlig körstil.

Nyckelord: Maskininlärning, Gamification (Spelifiering), Android, Eko-körning

Acknowledgements

First, we would like to thank Kirsten Rassmus-Gröhn who has been helping and guided us during the project. We will also thank Jacek Malec for advice about machine learning.

We will also thank Martin Gunnarsson at Jayway for his welcoming us to their company and for supporting us during the project.

We will also thank each other for a nice and good cooperation.

Lund, May 2018

Amanda Eliasson & Robin Timan

Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

In this chapter we introduce the project by describing the background, aim, and related work. We will also specify how we divided the work between the two authors.

1.1 Background

The project started with us finding the project idea on Jayways website. Jayway is an IT consulting company with 51-200 employees. The company was founded 2000 and the headquarter is located in Malmö, Sweden.

After meeting the studio lead at Jayway in Malmö we came up with new ideas and saw an opportunity to combine interaction design and android development with cloud computing and machine learning.

The automotive industry is adopting to recent technological advances where smart cars are becoming more frequent. We wanted to explore the possibility of using machine learning and gamification to create an insight among drivers about their own driving behaviour and use that insight to motivate eco-driving behaviours. It stands clear that drivers driving style have great impact on fuel consumption. How much a driver can expect to save on adopting eco-driving is highly individual, but studies have shown that the fuel consumption is on average reduced by 10% [1][2]. More detailed explanations about gamification and eco-driving can be found in chapter 2

1.2 Aim of thesis

This thesis is focusing on two main goals:

- How to best design an application with gamification to motivate drivers to drive more eco-friendly.
- Explore how to use machine learning to analyse driving patterns and gather insight about drivers' behaviour.

We believe that most people want to decrease their carbon footprint, but it is easy to forget or belittle the problem since its instant effect on ourselves is basically non-existent. We want to present the effect in an obvious way to make the driver more aware and more willing to work towards a minimal fuel consumption and show that even a small change can have a great impact.

1.3 Question formulation

Interaction design:

- Explore how to design an application that uses gamification to encourage drivers to use more environmentally friendly driving behaviours.
- Explore how to design positive and negative feedback about the driving behaviours both given during a drive and after a drive
- Explore how to design the communication between the driver and the application, to be comprehensive, non-disturbing, and encouraging.

Analyse:

- Explore how to use a car simulator to generate data.
- Explore machine learning techniques to find a correlation between driving style and fuel consumption.
- Explore cloud computing and its possible uses in an in-car application.

1.4 Contributions

This thesis is the work of two students, Amanda Eliasson and Robin Timan. Amanda studies Information- and communication engineering and Robin studies computer engineering, both at Lund University in Lund, Sweden.
Amanda has designed and implemented the front-end of the application and performed and evaluated the user tests. Robin has designed and implemented the back-end, including analytics and cloud support.

1.4.1 Work methodology

Our work has been carried out independently in parallel. We had a joint schedule for our parallel tasks to allow each other to keep up to date with the other person's progress.

1.5 Related work

Our work is inspired by [3] where they, like us, combined machine learning and gamification for eco-driving purposes. We used this article as a starting point for our own prototype and as a proof of concept. Their work is focusing on optimizing drivers braking style to conserve energy for electrical vehicles as opposed to our work where we are looking to optimize the fuel consumption.

The data used for prototyping our predictive model is the same data as used in [4]. Although they used the data for profiling drivers for an anti-theft purpose.

During the project we found a company called Autoliv, which developed an application called Driving Avatar [5] which is a study of driving behaviours and user experience to save lives in traffic. Driving Avatar gives the driver a score connected to both their behaviour in traffic and their profile. With the application the driver can look closer and learn from their behaviours. It is also possible to track the progress. Like we do, Autoliv uses machine learning and the driver will be provided with actionable insights and suggestions on how to improve driving behaviours.

CHAPTER 2

Theoretical background

In this chapter we describe all theoretical information that has been used in the project. Starting by defining how to achieve a good user-friendly design. Then the design process and the design aspects are described. Finally, we explain the concepts gamification, eco-driving and machine learning.

2.1 Usability goals

To be able to design a good and usable product it is important to have the users and their needs in focus. The key is to design the product or system with a user centered design. It is also important to think of what usability and usability experience is and how it can be achieved in this context.

2.1.1 User centered design

User Centered Design (UCD) can be described as an early focus on the user and their needs, having empirical measurements and an iterative design [6]. It is also stated that UCD includes an iterative design is about continuously updating the system and having in mind that a product is never perfect from the beginning. The product should go through many steps or iterations in the design. The designer has to identify users' needs, generate ideas, designs, prototypes, tests and get feedback from users and make changes. Then start all over again until the designer is satisfied and have reached the best possible product. In [7], Gibbons divided iterative design into three steps; understand, explore and materialize.

Understand

This step is divided into empathize and define. During empathize the developer conduct research about what users do, say, think, and feel. Next the developer defines, by combining all research and observe where users are having problems.

Explore

Explore is divided into ideate and prototype. Ideate is about generating a range of crazy and creative ideas. This is a brainstorming phase were all team members sketch many different ideas and share them with each other.

Prototype is when the ideas become real and tactile. Here the team can see the prototypes potential and get fast feedback about the prototype.

Materialize

Materialize is divided into test and implement. During testing, the team returns to users for feedback. This is done by putting the prototype in front of real customers and verify if it meets the goal obtained in the understand phase.

Next implement, by putting the vision into effect. In [7], Gibbons claims that this is the most important part of design thinking. An illustration of Gibbons iterative design can be seen in figure 2.1.

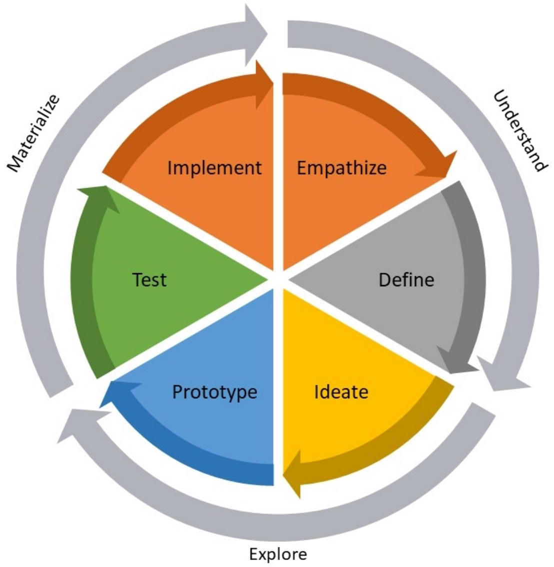


Figure 2.1: Iterative design

The International Organization for Standardization (ISO) have also defined User Centered Design (UCD), or Human Centered Design as they call it. In ISO 9241-210 they define it as "An approach to interactive systems development that aims to make systems usable and useful by focusing on the users, their needs and requirements, and by applying human factors/ergonomics, and usability knowledge and techniques. This approach enhances effectiveness and efficiency, improves human well-being, user satisfaction, accessibility and sustainability; and counteracts possible adverse effects of use on human health, safety and performance"[8].

Norman [9], has defined User Centered Design (UCD) as “A philosophy based on the needs and interests of the users, with an emphasis on making products usable and understandable”. He describes that it is important that the actions that are possible for the system should be easy to determine. Things should be visible, and it should be easy to evaluate the current state of the system. The design of the system should also follow the natural mapping between interactions and the required action, between actions and the resulting effect, between the information that is visible and the interpretation of the system state. In other words, the user should know what to do and be able to tell what is going on. If the user think “how am I going to remember that?” the design has failed. In [9], Norman furthermore presents seven principles for UCD and how to transform difficult tasks into simple ones.

Use both knowledge in the world and knowledge in the head

The first is to use both knowledge in the world and knowledge in the head. This means that the designer should design with respect to what the user of the product knows. By doing that the user is more likely to feel more comfortable, which hopefully helps the user to learn things easily. When talking about the use of knowledge of the world it is important for the designer to develop a conceptual model, with respect to three different aspects of mental models: The design model, the user’s model and the system image [9, p. 189-190]. Figure 2.2 shows an illustration of Normans conceptual model.

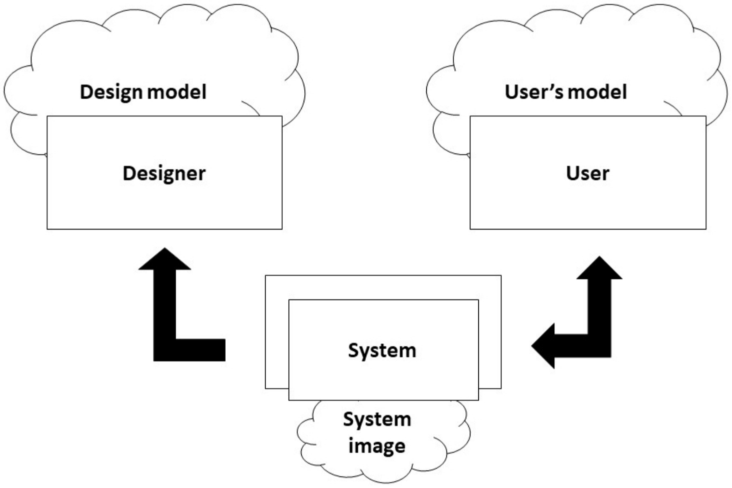


Figure 2.2: Conceptual model with respect to design model, user's model and system image

Simplify the structure of tasks

The second principle for UCD is simplify, it's what it sounds like, to design simple. For this it is important to pay attention to the user's psychology and limitations of short-term and long-term memory.

Make things visible

The third principle is to make the system visible. Users should see what possible actions there are to do with the system and therefore also know what actions should be done. Before the user interacts with the system, the users should also know the effects of their actions.

Get the mapping right

The fourth principle is to exploit the natural mappings. The user should be able to determine the relationship between possible actions and their outcomes. An example of natural mapping can be seen in figure 2.3.

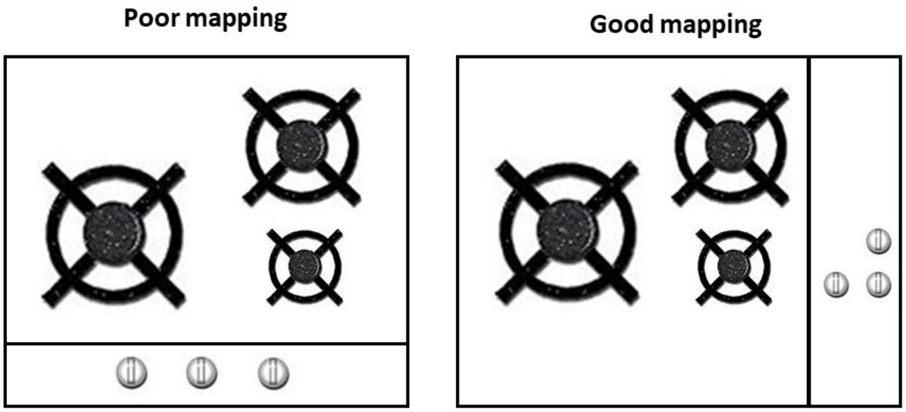


Figure 2.3: Good and poor mapping

For the poor mapping the user has problems to determine which control is connected to which burner. This because the kitchen stove is designed with different arrangements of burners and controls. To achieve good mapping the user should have no problem to understand which control is connected to which burner. To achieve that, the burners and the control should have the same arrangements.

Exploit the power of constraints

The fifth principle is to utilize natural and artificial constraints, meaning to give the feeling that there is only one possible thing to do and that it is the right thing to do.

Design for errors

The sixth principle is to design with knowledge of both system and user errors. By planning for possible errors, it is no surprise when the errors become reality, which makes the consequences hopefully smaller. The designer should design to minimize causes for errors, make it possible to reverse actions, make it easy to discover and correct errors that do occur and change the attitude toward errors [9, p. 131-140].

Standardize

The last principle is to standardize, when all else fails. This means if the system can't be designed without having difficulties then the best thing is to standardize things that are possible to standardize.

2.1.2 Interaction design

Interaction design is about creating interactive products that are usable and the products or systems should be easy to learn, effective to use, and provide a good user experience [10]. In other words, interaction design is about understanding the user's needs with respect to designing an interactive system to support them. To reach perfect interaction design there are two top-level concerns, the first is the usability goals and the second is the usability experience goals [10, p. 13-19].

Usability

Usability is mostly associated with the functionality of the product [11]. Rogers, Sharp and Preece describes that usability is about making the product effective to use (effectiveness), efficient to use (efficiency), safe to use (safety), have good utility (utility), easy to learn (learnability), and easy to remember how to use (memorability). In [11], They also say that industry design is important, meaning that the product attractiveness is important for good usability. When talking about usability it is important to have in mind that a product can be usable in different ways depending on the view of the product. In [12] Soegaard presents three different views:

1. When the view of the product is in mind and the usability has to do with the ergonomic design.
2. When the user of the product is the point of view and the user experience is directly connected to the products usability.
3. When the view has to do with the user's performance.

Soeqaard also describe usability as "A design is not usable or unusable per se; its features, together with the user, what the user wants to do with it, and the user's environment in performing tasks, determine its level of usability."

The International Organization for Standardization (ISO) have described usability in ISO 9241 were they describe it as "Extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use" [13]. From all the definitions above, it is important to agree that usability not only handles the development of satisfactory products It is equally important that the product is effective and efficient.

User experience (UX)

User experience is mainly about the users feeling when they're using the product. It includes the functional scope, product brand, psychological expectation, and actual emotional feeling [11]. In addition, if the designer wants good user experience the product should be satisfying, enjoyable, entertaining, helpful and more. Good usability is important for making a good user experience, because if the usability is good the product or the system is easy to understand and therefore the user feel comfortable to use the product. Feeling happy and enjoy the product improve the user experience. Although, it is important to be aware of that usability is not the entire experience [12]. It is also important to have in mind that experience is individual and subjective. The experience is very much connected to the user's past experiences, personal preferences, mood, and a myriad of other things.

Analyse Usability and User experience

There are many ways of analysing usability and the User experience (UX) of a product or system. One way is to use word clouds, which visualizes the most frequently-used words in a collection of text [14]. This could be done by using a word list with both negative and positive words [15]. The task is divided into two steps:

1. Mark all words that describes the experience of the product.
2. Circle five of the selected words that are most descriptive for the product.

Words with circle gives two points and words with only a marker gives one point. By calculating the points, a word cloud with the strongest words can be made. The designer/developer can then see if the product matches the UX that they want.

The NASA Task Load Index (NASA-TLX) is another method to analyse usability. The questionnaire has been described as "a widely-used, subjective, multidimensional assessment tool that rates perceived workload to assess a task, system, or team's effectiveness or other aspects of performance" [16]. The questionnaire is divided into two parts, where the first part contains six statements with a Likert scale. A Likert scale is a type of rating scale that is commonly used in surveys. The scale is linked to an assertion where the participant in the survey can rate how much the statement is correct. The second part of TLX intends to create an individual weighing of these sub-scales by letting the subjects compare them pairwise based on their perceived importance. The six sub-scales are mental demand, physical demand, temporal demand, performance, effort and frustration.

System Usability Scale (SUS) is a method to get an overall impression of usability and experience. It is designed with 10 questions with a Likert scale of 5. When calculating the SUS score subtract every odd item with one from the user response. For even-numbered items subtract the user responses from 5. This maps all values to the range 0 to 4 (with four being the most positive response) and then the total is multiplied by 2.5. This gives a number between 0 to 100. The average score should then be above 68 to be considered good. A value between 70-80 is to aim for and if the score is 90 or more it is outstanding [17].

2.2 Design, prototyping and construction

There are two types of design, the first is conceptual design, meaning to capture what the product or system will do and how it will behave. Conceptual design is included in the first step of the design process. The second type of design is physical design and involves, for example, what icons to use or how to structure the menu of an interface [10]. When starting a project there is often a lot of ideas. To be able to try out ideas, prototypes can be made. In [18] Babich, explains that the primary goal of a prototype is to be able to test the designs before creating real products. Babich claims that by prototyping it is possible to determine if the design concept works as intended. It is also possible to determine if people can use the product, this by finding usability issues before the launch. Normally prototypes don't look like the final product, they can have different fidelity. Babich explains that fidelity vary in visual design, content, and interactivity. Low-fidelity (lo-fi) and high-fidelity (hi-fi) are two examples of prototypes.

2.2.1 Low-fidelity prototyping

A low-fidelity prototype (lo-fi prototype) is one of the first prototypes in the design process. Lo-fi prototypes help the designers to get fast feedback about their ideas and possibility to quickly modify the prototype. Lo-fi prototypes are not designed to be kept and integrated into the final product. They are instead made for exploration [10]. In [18] Babich, state that the most important part of lo-fi is to check and test the functionality. The visual appearance of the product is less important for lo-fi prototypes. The advantages with lo-fi prototypes is that it is cheap and easy because it is normally made of paper. Lo-fi are also collaborative, many people can be involved in the prototyping and design process, because it doesn't require special design experience. The disadvantage with lo-fi prototypes is that it is impossible to show complex animations or transitions. Another disadvantage with lo-fi prototypes is that it could be hard to test the prototype on a real user. This because lo-fi testing requires a lot of imagination to support intended functions.

2.2.2 High-fidelity prototyping

A high-fidelity prototype (hi-fi prototype) is where the ideas have been transformed to a functional product that is possible to test. The prototype is functionally like the final product. In [18] Babich, explains that hi-fi prototypes are created when having a solid understanding of what to develop. Advantages with hi-fi prototypes is that usability tests could give meaningful results because the prototype often look like the final product and therefore users' behaviours could be quite similar. Another advantage is that it is possible to test animations, transactions and special interactions. Hi-fi prototypes also give an opportunity to demonstrate to potential investors or clients the idea of how the product is meant to be working. The disadvantages are of course that it is more expensive to develop compared to a lo-fi prototype and often takes more time.

2.3 Design aspects

Donald Norman is a professor of cognitive science. He has for many years been researching on what is catching the user's attention and how a product should be designed to be as usable and pleasurable as possible [19]. In [9] Norman present six design aspects: affordance, mapping, feedback, visibility, constraints and consistency.

2.3.1 Affordance

Affordance refer to the actual properties of the thing and determines how the thing could be used [9]. Affordance should answer the question "how do I use it?". For example, if you see a pair of glasses you know that you should put the spectacles on the ears to make the glasses sit perfect and in front of the eyes.

2.3.2 Mapping

Mapping is a technical term describing the relationship between two things [9]. To see if the mapping is working well the designer should be able answer the question "Where am I and where can I go?" meaning that the user should know what to do with one thing and know the consequences after interacting with it.

2.3.3 Feedback

Feedback refer to when the system gives the user information about what action has been made and what result this action gives [9]. This principle answers the question "What is it doing now?"

2.3.4 Visibility

Visibility is about making things visible that are important for the systems functionality [9]. Visibility should answer the question "Can I see it?". In a car the controls are positioned in a way, so they are easily found and used. Bad visibility is when we can't see how the system work.

2.3.5 Constraints

Constraint is about limiting the kind of interaction that can take place [9]. The meaning of that is that it reduces the errors and can also help the designer to get the users attention in certain ways. Constraints should answer the question “Why can’t I do that?”.

2.3.6 Consistency

Consistency is about designing similar systems with similar operations and elements. This makes the system usable and learnable, and it helps the user to use previous knowledge about similar systems. [9]. Consistency could be divided into four types: aesthetic, functional, internal, and external. Consistency answers the question: “Have I seen this before?”.

2.4 Gamification

Since 2000 the term gamification has been used [20]. In [20], Sailer, Hense, Mayr and Mandl state that the goal of gamification relies on four things; game, elements, design, and non-game contexts. By combining these four the definition of gamification can be defined as “the use of design elements characteristic for games in non-game contexts”. The motivation for gamification is like Ken Washington describes it: “Applying the fun, engaging and rewarding aspects of games to journey planning can allow people to improve their commutes, track their success and become aware of how their behaviour impacts the transport infrastructure as a whole” [21].

Nissan, Honda and Ford are three examples of car manufacturers where they have experimented with leveraging games as a mechanics to promote fuel efficiency [21]. Ford gives the driver visual feedback about the driving in the form of a plant where the amount of leaves of the plant are directly related to the driver’s behaviour. Instead of a plant Honda uses a tree and the reason to the implementation is like Honda VP Dan Bonawitz describes it "to help drivers improve their efficient driving skills by making the hybrid experience more fun and rewarding"[22]. Nissan has taken the concept one step further by launching a car in Japan that instead of giving the driver visual feedback about the fuel consumption, the car service let the driver know how the driving is compared to other people with the same car. The reason to this is to let social networking make eco-driving more popular.

2.5 Eco-driving

Eco-driving, or economical driving, is a driving style which has been shown to save up to 10% in fuel [2]. This means that a driver who have adopted eco-driving will have less impact on the global environment than a regular driver while also be able to reduce his/her fuel costs.

EcoDriver is a Canadian project with support from Natural Resources Canada devoted to promoting sustainable use of energy. In [23], EcoDriver define a core set of rules which are applicable to most drivers:

- Drive smoothly - Try to anticipate situations to avoid unnecessary braking and acceleration.
- Step off the accelerator - When slowing down or driving downhill, take your foot of the accelerator as this will activate the fuel cut-off switch.
- Shift up early - Always try to have as high gear as possible.
- Avoid excessive speeds - For every 10 km/h you go over 100, fuel efficiency drops by 10%.

2.6 Machine learning

Machine learning is the act of getting computers to learn without being explicitly told what to do. This requires data, and lots of it. A machine learning model is based on statistics and will use this knowledge to find patterns in the data. There are three main types of machine learning types:

- Classification - The goal is to predict discrete values, or classes. For example, this is used in spam filters where the goal is to predict an e-mail as spam or not (class 1 or class 2).
- Regression - The goal is to predict continuous values. For example, future stock prices.
- Reinforcement learning - Based on a reward-system where the algorithm rewards the agent for behaving "correctly". For example, the classical snake game where the agent would be rewarded for eating.

2.6.1 Types

In this section the types of machine learning will be explained more thoroughly.

Classification

Consider figure 2.4 as data fed to a machine learning model. The task is to create a model that can correctly classify a fruit as either an apple or a banana. To us humans the distinction between an apple and a banana is clear and we do it without even thinking about it, but we didn't always know the difference, we have been taught this growing up and we've been told which is what. A machine learning model is taught in the same way.

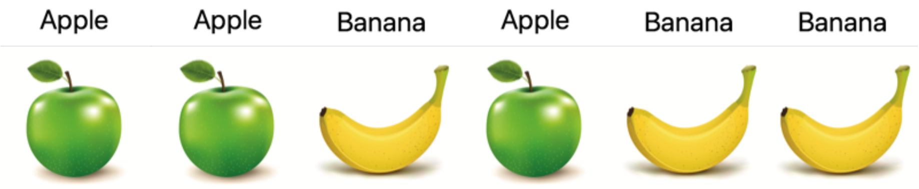


Figure 2.4: Simple ML example

Let's consider those patterns we're hoping to find. We can tell instantly that there are indeed some obvious differences between an apple and a banana. For example, a banana is "banana-shaped" and yellow while an apple is green and round. These observation is what becomes the features for our model. In table 2.1 the observations have been transformed into a language a computer can understand; true or false.

Table 2.1: The fruit observations from 2.4 displayed as features.

	Apple	**Apple**	**Banana**	**Apple**	**Banana**	**Banana**
Banana-shaped	False	False	True	False	True	True
Round	True	True	False	True	False	False
Green	True	True	False	True	False	False
Yellow	False	False	True	False	True	True

What we now have is a model which knows that an apple is round and green, and a banana is "banana-shaped" and yellow. So, when fed an unknown fruit the model will:

1. Find the features of the fruit.
2. If the features are round and green:
    - Classify the fruit as an apple
3. If the features are banana-shaped and yellow:
    - Classify the fruit as a banana.

Regression

Regression is another form of machine learning technique where, unlike in classification, the goal is to predict an unknown value based on previously observed values. This is most easily understood by considering a basic 2-D plot of x and y values. In figure 2.5 blue points is known data and the red point is an unknown point whose value we're looking to predict. The idea is still the same when dealing with high-dimensional data.

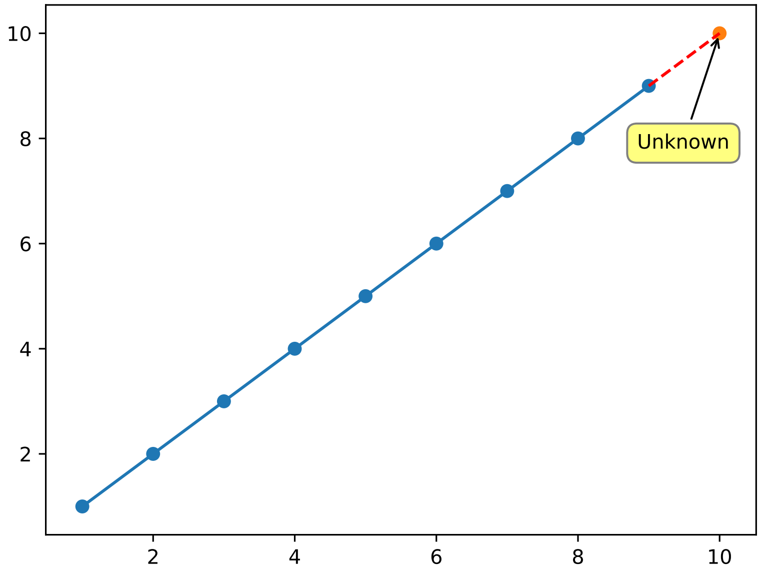


Figure 2.5: A simple regression example

Reinforcement learning

Unlike the others where the goal was to predict a certain value, we're looking to predict a certain action. It's based on rewards (or punishments) and the algorithm is training an agent which is looking to maximize the reward by exploring what action in each state that leads to the highest reward.

Reinforcement learning was briefly considered but it required the use of a simulator where the agent could be trained. It wasn't possible to do by just using the data we had at hand. In [24], Korolev gives more information about the snake example.

2.6.2 Algorithms

The algorithm decides how to achieve the goal. Some algorithms are only for one type (classification or regression) while others can be used for both.

Clustering

Refers to the algorithms where data is clustered into groups based on its location in the N-dimensional space, where N is the number of features describing the data. It's useful when the number of classes the data is divided into is unknown. In theory, these classes would be distinctly separated in these different groups. Reality however, is seldom that simple.

Clustering isn't used in our project, but we experimented with it a bit during our initial exploring phase. Although it isn't used, we believe that it could be used as driver classification and to detect anomalies in the training data [25].

LSTM

LSTM or Long Short-Term Memory is a type of recurrent neural network that remembers previous states and applies that knowledge when predicting the next state. This is particularly useful when we suspect that the previous events will affect our next event. This mimics the process we humans go through when we apply our previous experience to the present task. This is an important feature since we don't start our thinking from scratch every second.

A regular neural network has a major shortcoming when it comes to remembering previous states. The solution for this is LSTM which is especially good at connecting previous information to the current task [26].

2.7 Data management

2.7.1 Scaling

A useful technique when it comes to high value data. Its purpose is to scale down all values to a given interval (usually $[0, 1]$) while keeping the relationships between the data intact [27].

2.7.2 Normal distribution

One of the most common distributions and possibly the most important one. It describes data that is evenly distributed around a central point with a given standard deviation from that central point [28].

CHAPTER 3

Technical background

In this chapter we are going to describe the technical tools we have used during the project.

3.1 Software

In this section the software used in this project are going to be presented.

3.1.1 Development environment

In the project Android Studio was used to develop the application. Android Studio was released on May 2013 and is the official integrated development environment for Google's Android operating system. The developer of Android Studio is Google and JetBrains is built on IntelliJ IDEA. Android Studio is available for Windows, MacOSX, and Linux [29].

The python development used JetBrains IDE PyCharm which is also available for Windows, MacOSX, and Linux.

3.1.2 Github

Github is a web-based version control service that is using git. Git is a version control system that tracking changes in files that is used by multiple people [30]. We used it for merging our separate tasks and to keep track of changes.

3.1.3 Amazon Web Service

Amazon Web Service, or AWS, is a cloud storage solution created and hosted by Amazon. They supply an intuitive and easy-to-work-with data storage solution for hosting and processing data. We are using AWS to store the collected data and to perform post-drive processing.

3.2 Hardware

In this section the hardware used in this project are going to be presented.

3.2.1 Google Pixel C

The Android application was developed for Google Pixel C (figure 3.1), which is an Android tablet created and designed by Google. This is a replacement for the in-car infotainment system where the application is intended to be used.

Figure 3.1: Google Pixel C

3.2.2 Computers

The development was mainly performed on two MacBook Pro's 15". More information about the computers used in this project can be found at [31].

3.2.3 Car simulator

For testing purposes, we used a car simulator. The car simulator is a Logitech G920 Driving Force (figure 3.2). It is fully equipped with a steering wheel, gas pedal, brake pedal, and gear stick.

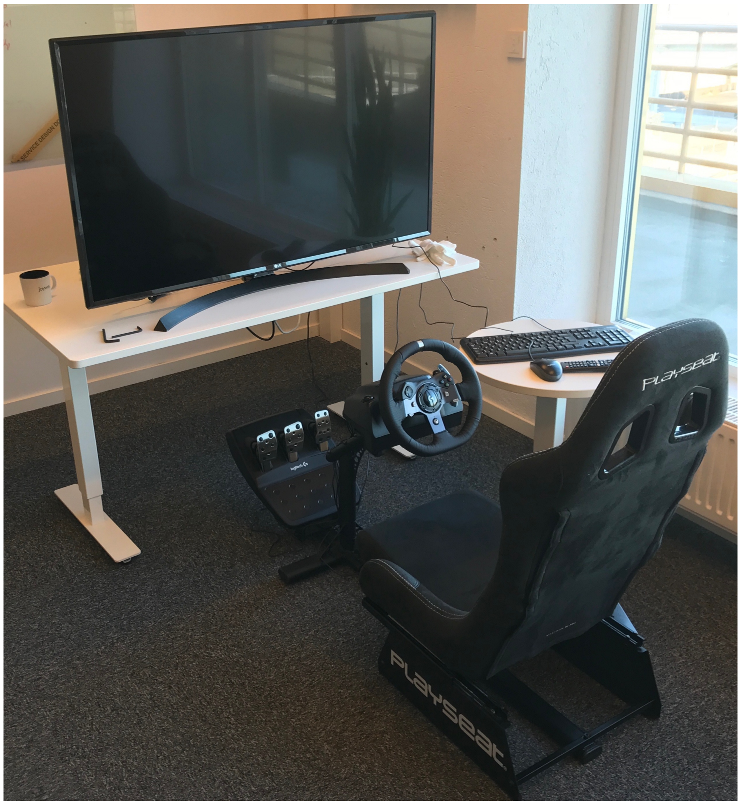


Figure 3.2: Car simulator used in this project

3.3 Data

The data used in this project comes from an on-board diagnostics tool (OBD). It refers to a vehicles self-diagnostics data which comes from different sub-systems of the car. Since 1996 [32], it is incorporated by law in each vehicle's on-board computer and is used to monitor every part of the vehicle. It is easily accessed using an OBD-reader which can be bought in most appliances stores.

CHAPTER 4

Front-end

In this chapter we are going to describe the method for the front-end. The chapter describes the whole process, from brainstorming to the final product.

4.1 Iterative design

The front-end of the application has gone through many steps. The design and the implementations have been developed iteratively by repeating the steps: analyse user's needs, design, prototype or implement, test and evaluate. Figure 4.1 illustrates the iterative workflow of the applications front-end.

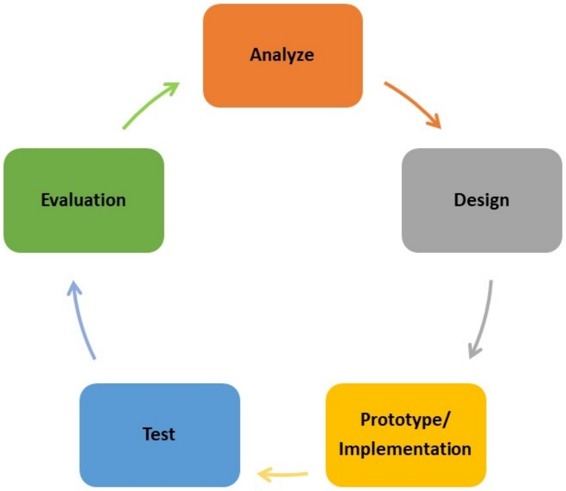


Figure 4.1: The iterative workflow

4.2 Brainstorming

We started with a brainstorming phase, where we drew sketches on a whiteboard and discussed different design alternatives for both the layout showing when driving and the layout showing after a drive. The result from this phase is described more in depth in chapter 6. Figure 4.2 and figure 4.3 illustrates the most important steps during the brainstorming activities. We also discussed how to implement machine learning in our project during our brainstorming activities.

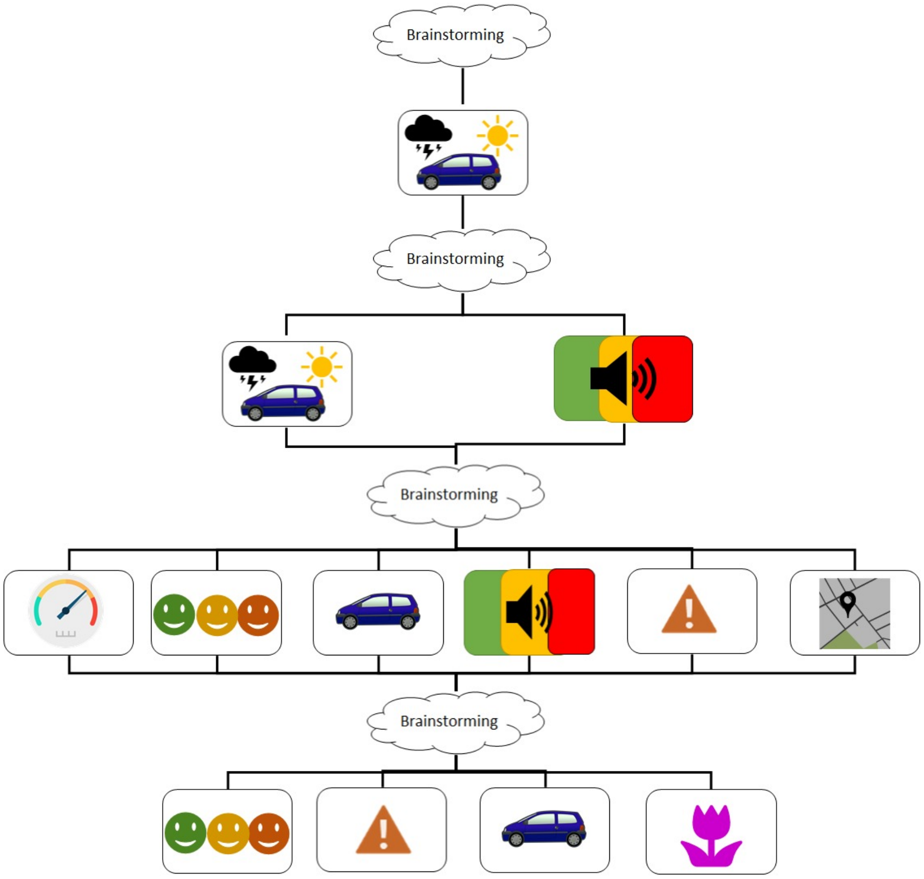


Figure 4.2: Brainstorming sessions for Drive

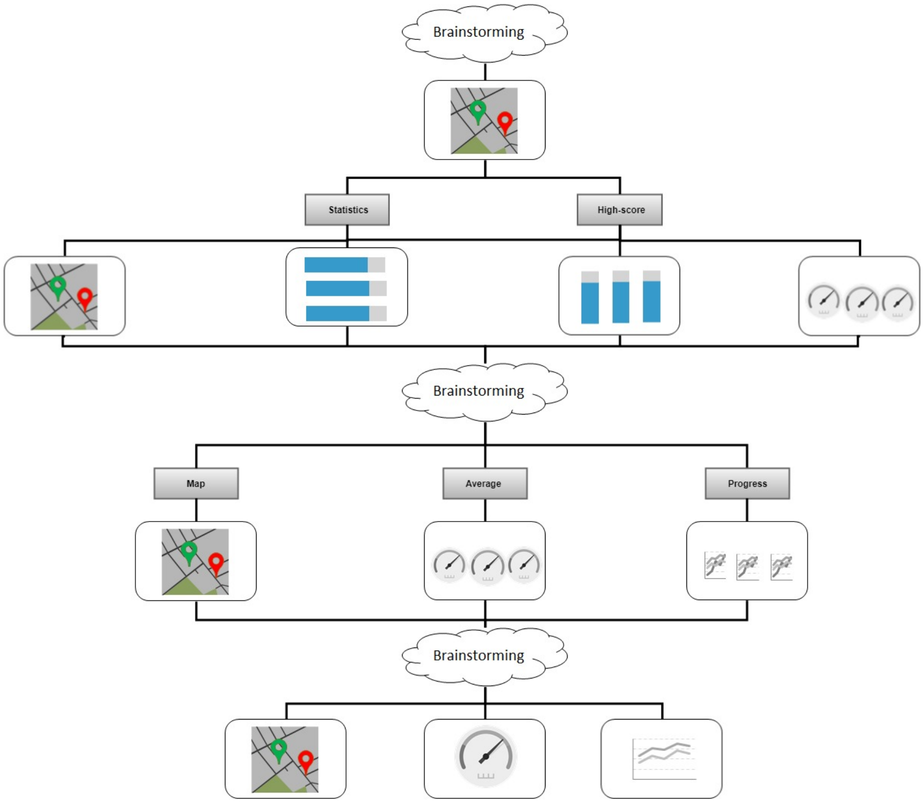


Figure 4.3: Brainstorming sessions for Score

4.3 Low-fidelity prototyping

Lo-fi prototypes are used because they are simple to develop, easy to change and not much effort is wasted if they need to be discarded. Lo-fi prototypes are also efficient and makes it possible to get quick feedback [33]. Lo-fi is limited in usability testing [10]. Because they look less finished, users sometimes criticize more during lo-fi testing. For more information about lo-fi prototype please see chapter 2.

4.3.1 Drive layout

We held in total 4 brainstorming sessions for this phase that generated 9 different design ideas. All these were sketched on papers and scrutinized by us and the supervisor at Jayway. We also asked others not related to the project for feedback. Finally, 4 designs were chosen for the hi-fi-prototype. The steps made in the brainstorming sessions can be seen in figure

4.3.2 Score layout

For the Score layout we held in total 3 brainstorming sessions. In total this generated 8 design ideas. The lo-fi were made of paper and were tested by the project group and the supervisor at Jayway. Finally, 3 designs were chosen for the hi-fi prototypes. The steps made in the brainstorming sessions can be seen in figure 4.3.

4.4 High-fidelity prototyping

In this project the hi-fi prototypes were implemented in Android Studio. That means that all hi-fi prototypes were coded. This because then they could easily be tested and improved. The most important implementation steps can be seen in figure 4.4. The numbers in the activities denotes the chronological order in which they were carried out. The colour groups activities related to each other together. Between every step there were of course also smaller implementations. For more technical information go to chapter 3.

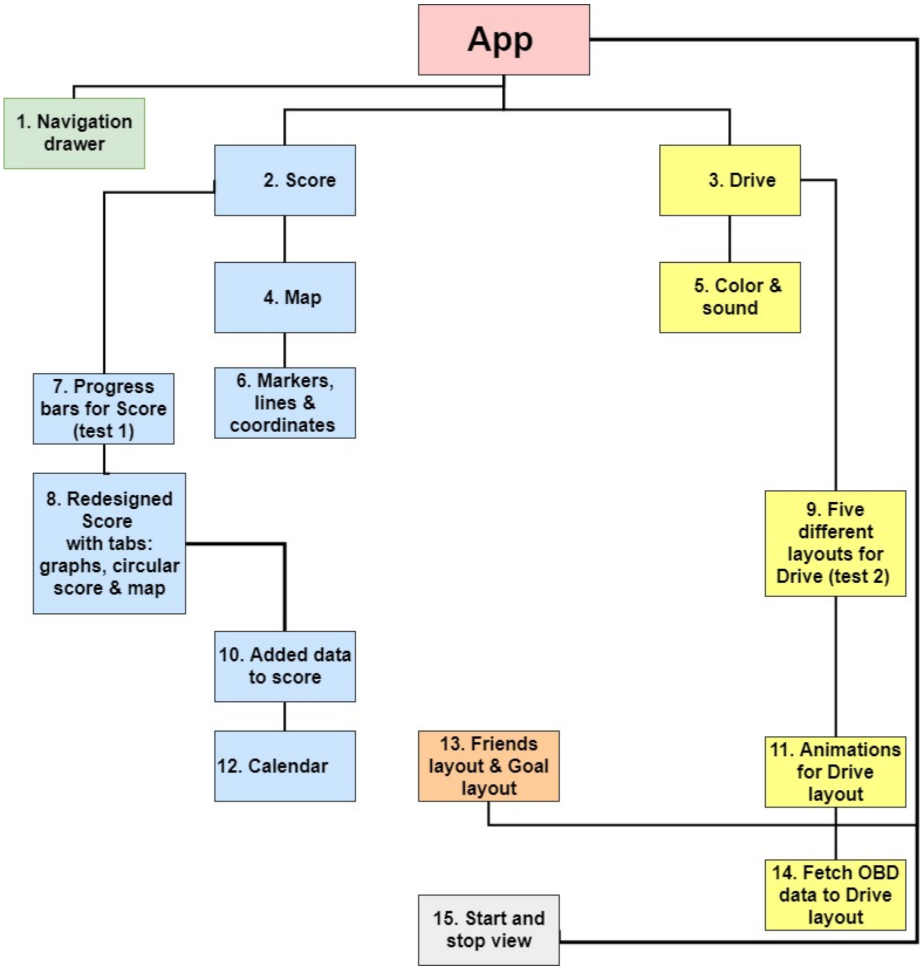

App
1. Navigation drawer
2. Score
3. Drive
4. Map
5. Color & sound
6. Markers, lines & coordinates
7. Progress bars for Score (test 1)
8. Redesigned Score with tabs: graphs, circular score & map
9. Five different layouts for Drive (test 2)
10. Added data to score
11. Animations for Drive layout
12. Calendar
13. Friends layout & Goal layout
14. Fetch OBD data to Drive layout
15. Start and stop view

Figure 4.4: High-fidelity implementation. Numbers denotes their relative chronological order and colours how they are related.

4.4.1 Test 1: High-fidelity test, Score-layout

During the first hi-fi test the high-score layout was tested. This was a small test where only 4 persons participated. The reason to the small number of participants was that we wanted to get fast feedback about how the high-score should be illustrated. Three different views were implemented. One horizontal graph, one vertical graph and one circular graph.

4.4.2 Test 2: Second high-fidelity test, Drive layout

The second high-fidelity test started with a short presentation of the project and the test. After that the participant read and signed an informed consent which is a document that explains the purpose of research, how the study is going to be conducted and if there are any possible risks and what benefits the participant gets by participating. Finally, it tells the participant that it's anonymous and all data is only available for the research team and is going to be stored under an anonymous code. To make the test as realistic as possible the test was performed in a car simulator with the tablet placed in front and to the right of the participant, where the In-Vehicle Infotainment (IVI) normally is placed. Before the test started the participant took part in an interview about the experience of a car simulator and after the test the participant was interviewed a second time where they gave us feedback about the application. The pre-test questions and post-test questions can be seen below.

Pre-test questions:

1. Have you ever used a car simulator? If Yes, what is your overall experience?

Post-test questions:

1. What was your general feeling after the test?
2. Which view of the five you have seen was most appealing? Why?
3. What would you like to change/improve with your favourite view?
4. Was there any view that raised too much attention? What was disturbing? What would you like to change with that view?
5. What was the positive and negative feedback in each view?
6. How did the negative/positive feedback affect you?
7. Did you miss any feedback? For example, voice?
8. Did you think that any object was unnecessarily large / small?
9. What is your experience of the colours for the background and the pictures?
10. What was your overall experience of the animations?

For this test we had 6 participants, all with past experiences of driving in a car simulator. The first participant was a pilot test where we after the test changed the time interval for the input data. We realized that the input data changed too fast to be realistic.

4.5 Final product

From the results and feedback from test 1 and 2 a final product was implemented. It was developed for the final test, where Drive and Score were tested together. We enhanced the animations, made images and numbers larger and designed them with clear colours. We merged the front-end with the back-end using a single analyser class which was called from the front-end and then delivered the result back for display to the user. The goal with the final product was to equate the product as a game. More detailed information about the final product design and functionality can be found in chapter 7.

4.5.1 Test 3: Final test

This final test was designed to test almost the entire application, with focus being on the Drive and Score layouts. The test was held at the Jayway office in Malmö, Sweden where the car simulator is located. During the test the participant was driving in a car simulator.

The test plan with a more detailed explanation of the test can be found in appendix B.

Research questions

Six research questions were presented. Drive layout:

- How well do users understand the feedback from the smiley?
- How well do users understand the score to the right of the smiley?
- Do the users get unnecessarily disturbed?
- How do the users feel when getting the feedback?

Score layout:

- How well do users understand the Map, Progress and Score layout?
- How easily and successfully do users find the tools or options they want?

Participant characteristics

We also needed to differentiate between users to discuss the results regarding a user's experience in the discussion chapter. These questions were answered in the pre-test questionnaire, which can be seen in figure B.1.

Test method

Each test took approximately 30 minutes to complete. During the test the test moderator gave the participants tasks and the observer took notes. In the test plan (appendix B) a more detailed description of the test method and the test tasks can be found.

After the test, participants answered three different questionnaires (NASA TLX rating worksheet, SUS and Word-cloud) because we wanted to get a large result set and we did not know which questionnaire would give the most credible results. As Jeff Sauro put it, "there isn't a usability thermometer to tell you how usable your software or website is." Both NASA TLX and SUS are satisfaction metrics, so they don't count completion rates, errors, task time, and other important rates. But they could give us a good indicator of the applications satisfaction. See NASA TLX rating worksheet, Word-choices for Word-cloud and SUS in Appendix B.

CHAPTER 5

Back-end

This chapter will explain how the back-end was developed. Here we will present a short and concise overview of the workflow as an introduction for the reader. We will go through and explain each part later in the chapter.

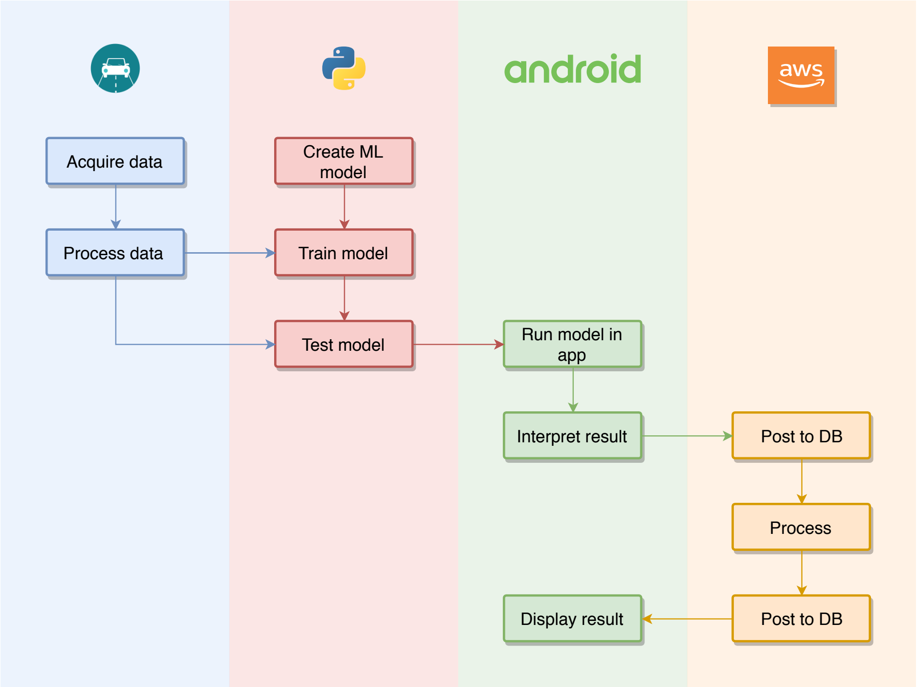


Figure 5.1: Back-end workflow. The columns represent what tool the work was carried out with.

5.1 Data

Initially our plan was to collect our own data using an OBD-tool [32] which is real-time data from a real vehicle, alternatively generate data using a simulator. This is crucial because machine learning algorithms require data, and lots of it, it therefore became apparent quickly that collecting our own would take too long time.

Generating data using a simulator would also have been a time-consuming task. Although driving in a simulator would have been a lot more convenient than an actual car, it still would have required manual driving.

Existing datasets available online were quite sparse but we found a few and eventually decided to use the one that was used in [34]. The data has 53 features all collected from the same vehicle. The first 10 samples from the first 5 features can be seen in table 5.1.

Table 5.1: Table showing the first 10 rows of the dataset

Fuel_consumption	Accelerator_Pedal_value	Throttle_position_signal	Short_Term_Fuel_Trim_Bank1	Intake_air_pressure
268.8	0.0	5.2	0.0	33
243.2	0.0	6.1	0.0	40
217.6	0.0	5.2	0.0	41
204.8	0.0	4.7	0.0	38
217.6	0.0	5.7	0.0	40
243.2	0.0	5.7	0.0	41
217.6	0.0	5.7	0.0	42
294.4	0.0	6.6	0.0	52
332.8	0.0	7.1	0.0	60
358.4	0.0	7.1	0.8	65

5.1.1 Data filtering

As mentioned there is a total of 53 features in the dataset but all of those might not be needed. To filter out unnecessary data we used three criteria that can be seen in table 5.2. If a feature failed any of these it was dropped from the dataset.

5.1.2 Scaling the data

Scikit-learn provides a min-max scaler as part of their library [36]. We used this to scale each value down to the range 0 to 1 while also preserving the information carried by each feature. The reason for this is that machine learning algorithms might not converge if the numbers are too large.

Table 5.2: Table showing criteria for filtering data

Question	**Answer**
Is the feature directly or indirectly affecting the fuel consumption?	We are using the Kendall correlation [35] to determine the correlation between two features.
Is the data of the feature valid?	Plotting the data and checking for unexpected values such as all zeros or null values.
Is the feature providing new information?	If the correlation between the features are higher than a set threshold value we consider them to be too similar.

5.2 The model

One of the main goals of this master thesis was to explore how to apply machine learning in an eco-driving application.

5.2.1 Designing the model

So first off, we needed decide in what way we could apply machine learning in our application. We considered the following:

1. Driver classification - Classify drivers into categories based on how environmental they are.
2. Action prediction - Predict the next action to take as to minimize the fuel consumption (shift gear, brake, etc.)
3. Difference classification - Predict an expected fuel consumption and use that as a reference point.

We evaluated each method one by one and then picked the one that suited our needs best. The evaluation showed that option 3 suited us best. More on this in chapter 7.

5.2.2 Building the model

For real-time evaluation we needed a virtual model that mimics the features of a car and predicts the fuel consumption according to a given state. We then compare the expected fuel consumption with the actual one to find a reference point to which we would be able to classify the driver while he/she is driving. This results in a regression problem where the vehicle data will be used to predict future fuel consumptions.

The model was implemented in Python as a LSTM-network using the Keras library [37]. In chapter 7, this will be covered more thoroughly.

5.3 Run real-time predictions in app

The model was trained and tested on a desktop and not directly on the tablet. This is common practice due to computing limitations on the average tablet. However, running the computations in real-time had to be done in-app so we needed to transfer the model and the following is how it was done [38]:

1. Save the model as a protobuf file to local storage.
2. Put the file in the android assets directory.
3. Load the model into the app using TensorFlowInferenceInterface.

TensorFlowInferenceInterface is very straight-forward to use. The code below is the predict function of our app where the TensorFlowInferenceInterface-instance is denoted as `tf`.

```java
private float predictFuelConsumption(float[] data) {
    float[] output = new float[OUTPUT_SIZE];
    // Feed the data to the model.
    tf.feed(INPUT_NODE, data, INPUT_SHAPE);
    // Process the data.
    tf.run(OUTPUT_NODES);
    // Fetch the result.
    tf.fetch(OUTPUT_NODE, output);
    return output[0];
}
```

5.3.1 Interpreting the result

What we have achieved up to this point is to train a LSTM model, transfer it to the application, and performing online predictions using it in the application. However, to make it more informative for the user we need a way of transforming the difference in expected and actual fuel consumption to a measure that could be easily understood while driving.

To determine which method to use we identified the problems we encountered and proposed solutions to the identified problem. The solution that solved all problems was chosen as classification method.

Table 5.3: Table describing problems and proposed solutions

Problem	**Solution**
In theory there is no limit for the difference. In practice we know that there is only a finite amount of fuel that can be spent but since there is not a set limit for this we should consider the difference to be in the range $[-\infty, \infty]$	Define a set of fixed intervals, one for each class, and then classify the driving according to which interval it falls within.
How should those intervals be defined?	It stands to reason that some differences are more common than others and we need to take this into account when determining the intervals. This is now starting to look like a distribution where differences close to 0 should be a lot more common than differences further out.
What distribution has these properties?	Normal distribution

5.4 Storing and post-drive analysis

We are using AWS DynamoDB for storage and AWS Lambda for post-drive analysis. The lambda function is simply a piece of code that runs when a new item is uploaded to the database and performs actions on the uploaded data.

We have in total two DynamoDB tables; one where the driving data will be uploaded to after a drive has finished, and one that holds the results from the post-drive-analysis.

The Lambda function is developed in Python due to the language's simplicity which allowed for a faster development cycle. In it we parse and process the resulting data from the first data table and then uploads it to the other table.

CHAPTER 6

Design iterations

The application was designed and developed iteratively. In this chapter a description of the most important low-fidelity and high-fidelity iterations is presented. Test results from test 1 and 2 along with a more detailed explanation of the application design is also going to be given.

6.1 Low-fidelity prototyping

The first idea was to design the Drive layout using metaphors such as different weather to illustrate good/poor driving. For this idea, sunny and blue sky was good driving behaviours and rain and thunder was poor driving. In the layout there was also an animated car. We drew an overview of the connection between the front-end and the back-end part of the project. Images from the first brainstorming session can be seen in figure A.1 in appendix.

Score was from the beginning designed with a map where users could see their driving and get information about it by clicking on the map. After re-designing the Score layout, we divided the layout into two layouts called Statistics and High-score. The statistics layout is shown directly after a finished drive. This layout was the map layout where the markers where placed where the user have been driving and were designed with different colours (green, yellow and red) depending on the driving behaviour.

The high-score was meant to show the percentage score of how good or poor the driving was compared to the expected value. The layout was designed with three different parameters that were related to the driving. Those three were: legal driving, acceleration, and braking. We designed three different designs for this view. The first was a circular progress bar, second a horizontal progress bar and the third a vertical progress bar.

During the low-fidelity prototyping we drew many different prototypes. In figure 4.2 and 4.3 the brainstorming sessions for Drive and Score can be seen. The result of the first low-fidelity prototyping can be seen in figure 6.1.

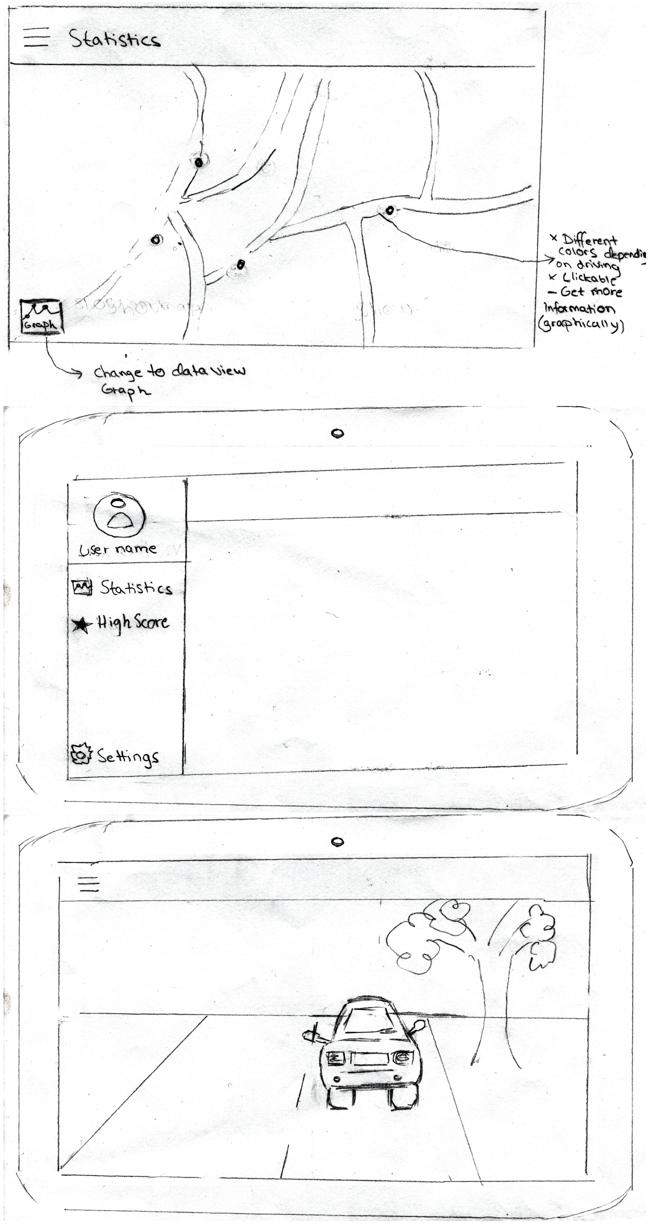


Figure 6.1: Low-fidelity prototypes

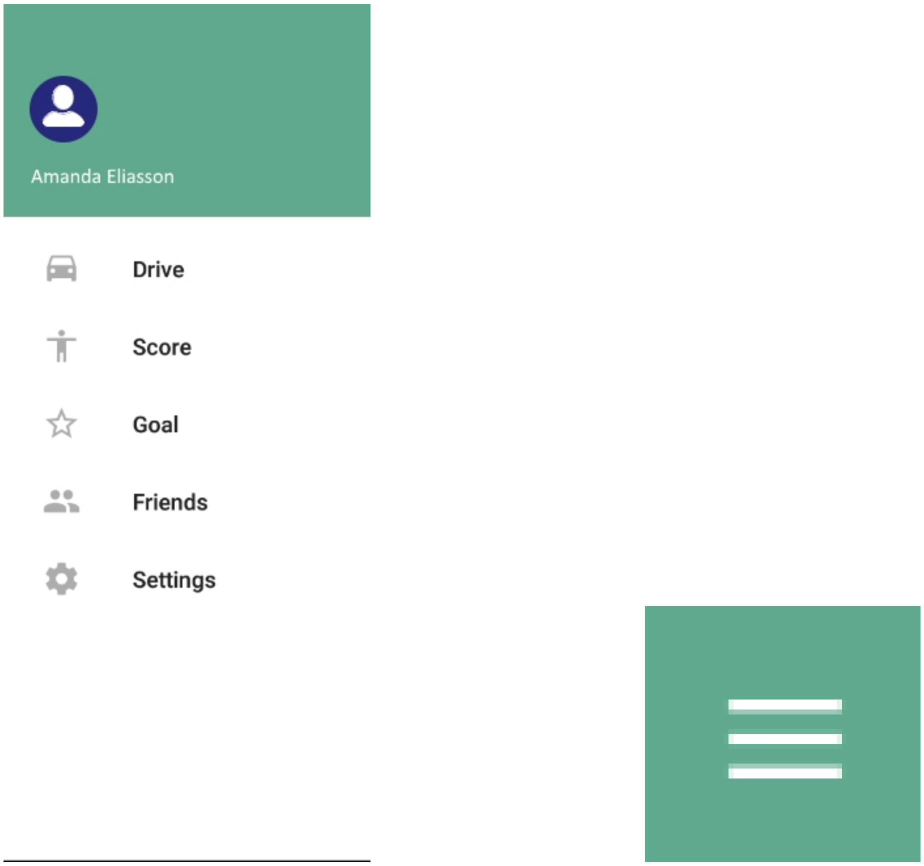
6.2 Application design

We decided to have the application in landscape mode, that way it will more resemble the infotainment-system of the car where the application in the future is meant to be placed. When designing the application, we had the user's needs in mind. The need was to design the application in a non-distracting manner and therefore focus was kept on making it as minimal as possible. One way of achieving a clean design is to use a navigation drawer. A navigation drawer is a panel that is placed on the left side of the application screen and displays the different navigation options [39]. When an option is selected the drawer disappears and hides the other options. The navigation options can be found by clicking on a marker (figure 6.2b) placed on the left side of the screen or by swiping from left to right. The navigation drawer can be seen in figure 6.2a.

When designing the application, we had Norman's design aspects in mind throughout the process. The application was designed to be simple and the interaction should be obvious, in other words the application should have good affordance. Visibility was also an important aspect because the driver's eyes should just be on the application a short time or preferably the application should be possible to understand by periphery viewing. Therefore, it was important to determine which functions or objects that should be visible and make these objects visible by periphery viewing without causing a disturbance.

6.3 Navigation drawer

The first step in the hi-fi prototype was to develop a navigation drawer. We chose to implement all the options as fragments. A fragment is a kind of modular section of an activity, which has its own life-cycle. It receives its own input events and it is possible to add or remove input while the activity is running. A fragment can be described as a sort of sub-activity that is possible to reuse in different activities [40]. The hi-fi result of the navigation drawer is shown in figure 6.2.



(a) Drawer

(b) Marker

Figure 6.2: Navigation drawer and navigation marker

6.4 Drive layout

In this section the most important iterations during the implementation of Drive are going to be explained. The result from test 2, where five different layouts were tested, is also going to be presented.

6.4.1 Changing background

The Drive layout was redesigned to a layout with a changing background colour depending on the driver's behaviour. This made it possible for the user to understand the feedback in the periphery. Red background meant poor driving and green meant good driving. It was designed with sound to let the user solely focus on the road.

6.4.2 Six different layouts

Five new layouts were designed during our next brainstorming session:

1. A smiley as a way of illustrating good or poor driving.
2. A circular scale instead of smiley that changes from green (good driving) to red (poor driving).
3. A warning triangle that changes its transparency depending on the driving.
4. An animated car.
5. A map.

The map was never implemented because we found it to be out of scope for this project. The sixth layout was still the coloured layout but without sound feedback, to let us focus on the visual design.

6.4.3 Five different layouts

The design was redesigned after one more brainstorming activity to five different views: one warning triangle layout, one smiley layout, one animated car layout, one static car layout and one growing flower layout. The reason to the amount of views was that we had a lot of ideas about the look and we wanted individuals not involved in the project to say which view they preferred. To make the change between different images within the various views as smooth as possible, the transitions were animated. To get exactly the images we wanted, we drew the images ourselves in Photoshop. We also took away the sound/voice because we wanted the images to be the focus of the test. These five designs were tested during test 2.

In parallel to the implementation of these designs; the implementation of the back-end that was going to support them started being developed. Although there are 5 layouts they all could use the same back-end so this allowed the back-end to be developed in parallel without any risk of the work becoming obsolete after the final design was appointed.

Warning triangle

The triangle view is an image of a triangle that changes its transparency depending on the driving classification. This view was continuously tested to get the best changes of transparency. The transparency goes from zero (very good driving) to one (very poor driving). The low-fidelity prototype and the high-fidelity prototype of the triangle can be seen in figure 6.3 and 6.4.

This layout is different from the others in the sense that it doesn't require classes. It only requires a transparency value between 0 and 1 which is exactly what the back-end returns.

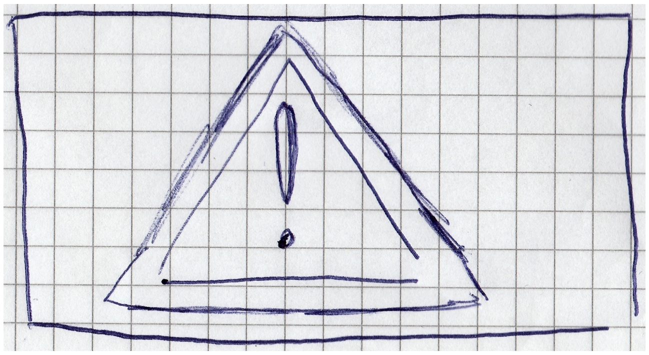


Figure 6.3: Low-fidelity prototypes for warning triangle

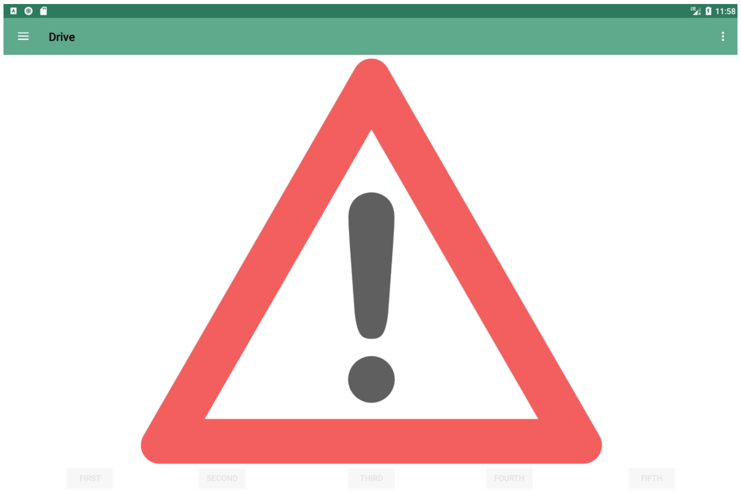


Figure 6.4: High-fidelity prototypes for warning triangle

Smiley

This view contains four different smileys ranging from very happy to very angry and is color-coded accordingly as: green, yellow, orange and red. The green and happy smiley corresponds to very good driving and the red and very angry to very poor driving. These smileys are changing using a fade out/fade in animation to make the transition smooth. During the project this views images has been changed and tested to make the smileys as clear as possible. The low-fidelity prototype and the high-fidelity prototype can be seen in figure 6.5 and 6.6 respectively.

This is a typical classification layout. While the use of the warning triangle is straight-forward and works well with the back-end out of the box, this layout raised some questions as to how the classes should be determined. We came to the conclusion that this is a sensitivity issue of how harsh we want the classifier to be and it is something that would have been evaluated in a long-term test of its own. A test of that magnitude would require real users in real cars and is out of scope for this project.

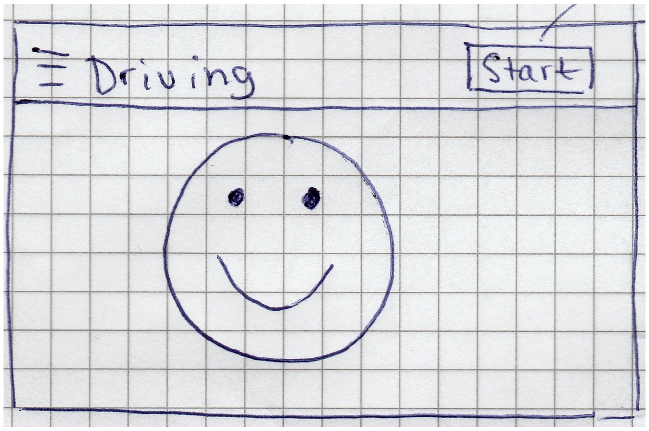


Figure 6.5: Low-fidelity prototype for smiley

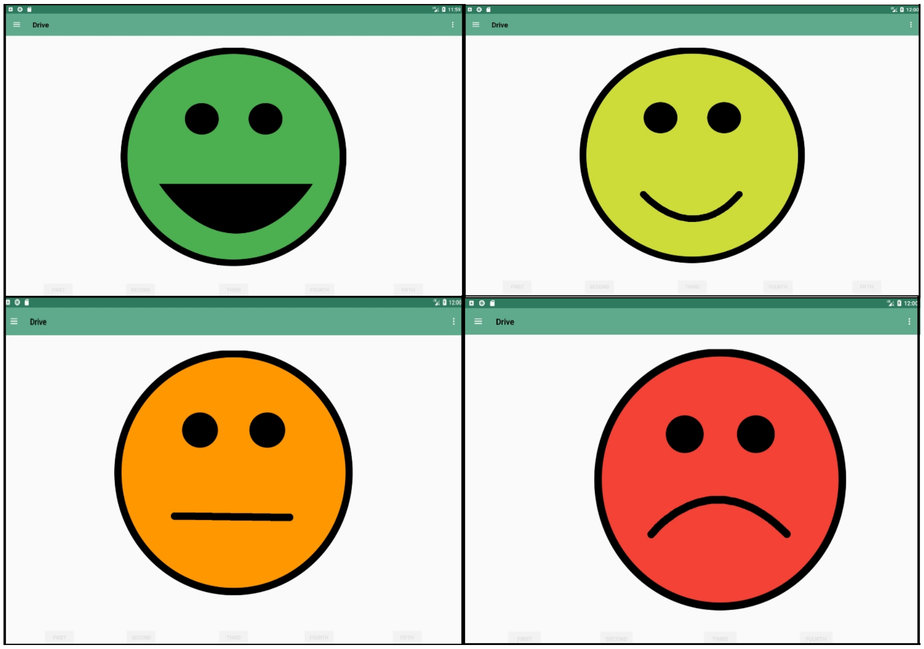


Figure 6.6: High-fidelity prototype for smiley

Animated car and stationary car

The animated car and the stationary car view is the same view with the difference that the animated car is a moving and the stationary car is not. When the driver is driving poor exhaust is coming gradually from the exhaust pipe and when the driving is good the exhaust gradually disappears. The car is implemented and designed in PowerPoint and has during the project been changing colour and layout. The high-fidelity prototype can be seen in figure 6.7 and 6.8 respectively.

This could both be implemented using classification and regression. In the case of classification, the number of exhaust clouds would be the number of classes; and for regression it could have been translated to a transparency value much like the warning triangle layout.

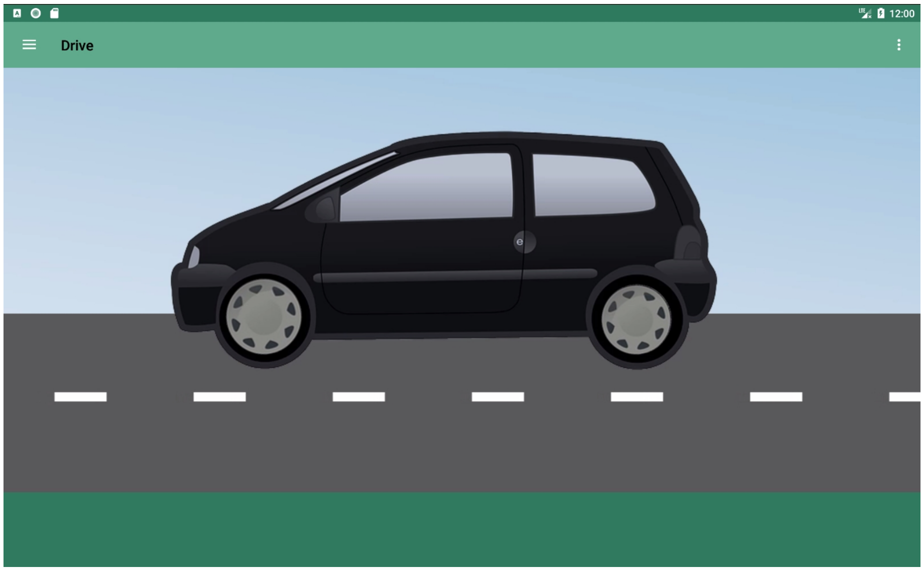


Figure 6.7: High-fidelity prototype of the car

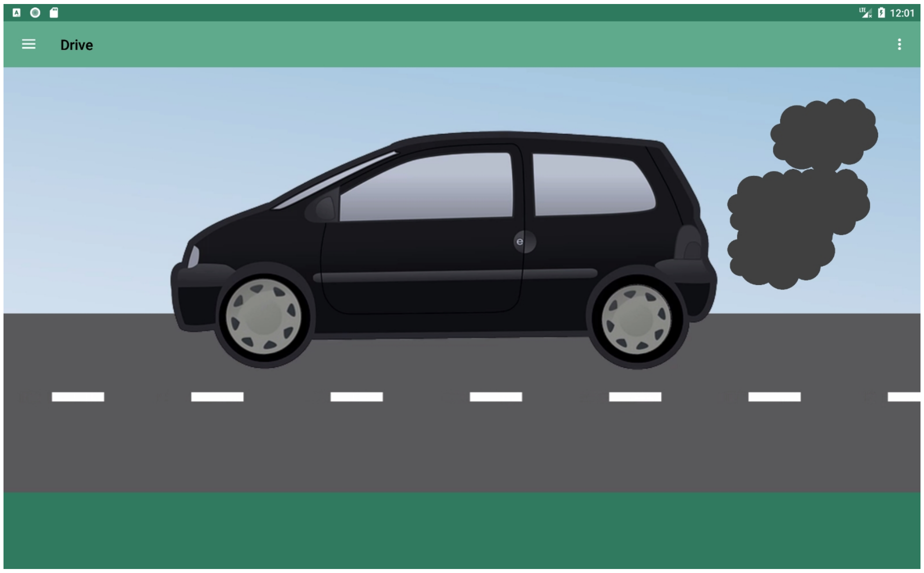


Figure 6.8: High-fidelity prototype of the car with cloud animation

Growing flower

The growing flower view consists of three flowers that starts to grow when the driving is good and wilt if the driving is poor. The high-fidelity prototype can be seen in figure 6.9 and 6.10.

It connects to the back-end as a classification problem where the number of growing steps would correspond to the number of classes.

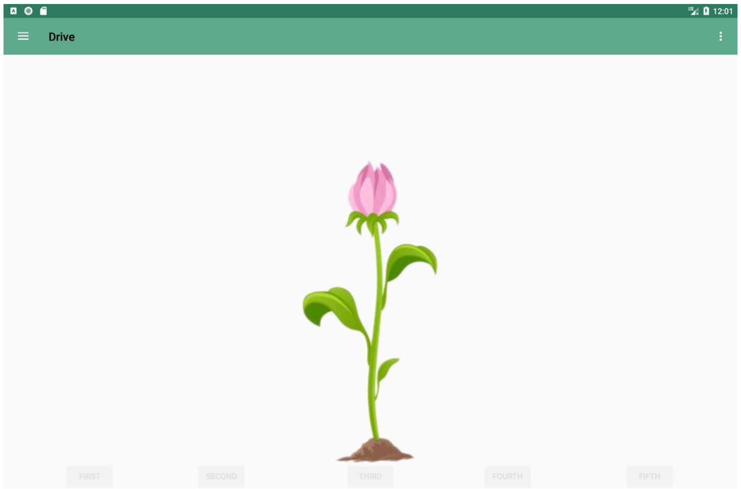


Figure 6.9: High-fidelity of one animated flower

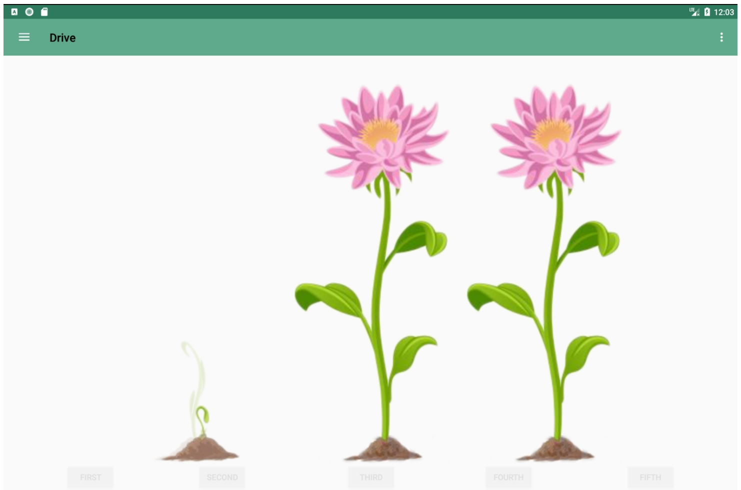


Figure 6.10: High-fidelity of three animated flowers

6.4.4 Test Drive layout

Below is the total and average result from test 2. For this test we used dummy data due to the fact that the back-end was being implemented in parallel and yet wasn't available at this point. We believe that the affect of dummy data on the test was minimal since the test focused on evaluating the visual feedback only.

Questions:	**Answer:**
What was your general feeling after the test?	The feeling was good and exciting.
What view of the five you have seen was the most appealing? Why?	Triangle and Smiley, because they are easy to understand and give quick feedback. The smiley gives both negative and positive feedback which is good. Triangle took less focus compared to the other views.
What would you like to change/improve with your favourite view?	The triangle could be misinterpreted as an indicator of damage to the car. Improvement with the smiley could be to add an eco-driving score, where the driver could see how good/poor the driving is.
Did any view take too much attention? What was disturbing?	The car views and the flower took too much attention. The flower view would have been better if there only was only one flower because it was hard to understand how many flowers it could be. It took a lot of attention to count them. In the car views the car was in focus instead of the exhaust. So, an improvement is to put the exhaust clouds more in focus, by making the car smaller and the exhaust bigger and with different colours.
What was the positive and negative feedback in each view?	All the participants could see the negative and the positive feedback in the views, but for some it was hard to understand on a scale how good or poor the driving was in the car view, triangle, and the flower.
How did the negative/-positive feedback affect you?	The negative feedback made them react and the positive feedback made them happy. The general reflection on the feedback was that it was important to have both negative and positive feedback.
Did you miss any feedback? For example, voice?	Some of the participant said that they maybe wanted audio feedback, but they also said that it could be annoying.

Did you think that any object was unnecessarily large/small?	It was good that the objects were big, the bigger the better.
What was your experience of the colours for the background and the pictures?	For the car view the exhausts was hard to see, the colour would have been better if they weren't designed in the same colour as the car. It would have been nice if it was possible to change the background colour to black or if there was a functionality that changes to night mode when driving in the dark. The colour of the smiley was a bit too strong.
What was your overall experience of the animations?	The animations were good, but some of the participants had problem seeing that the animated car was animated.

Table 6.1: The summarized result from the interviews during test 2

The overall comments about the drive layout was that the drive layout should be in full screen and have an eco-driving score.

6.4.5 Improved Drive layout

From the test most of the participants agreed that the smiley was the best view when driving. The improvement that we made after the test was putting the application in full-screen. To let the driver get a better sense of their progress a real time score was added to the view. This score is intended give the application a more game-like feeling and is simply the real-time score returned by the back-end during driving.

6.5 Score layout

Score has gone through many iterations throughout the project. In this section the most important implementations are going to be given. The result and improvements from test 1 are also presented. This layout where the post-drive analysis is presented. Upon starting to work we decided what type of scores we wanted to present and in what scale they should be. The design was iteratively developed alongside the back-end for the post-drive analysis and we were able to do so because we made sure to keep in mind what was expected from the other. At this point we hadn't decided upon what the scores should represent, just the scale and type, therefore the names of the scores presented in this section are just dummy names and aren't represented in the final product. More information about the scores are available in chapter 7.

6.5.1 Map layout

The first step with the Score layout was to implement a map layout. Next, clickable markers were added. These markers were designed with different colours depending on the driving behaviour. By clicking on a marker user got information about the event it represented. This layout is implemented visually only and we're lacking support for it in the back-end.

The map was redesigned with lines between the markers to make the driving route clearer. The lines were also coloured. To let the driver, get feedback on their progress the colour of the lines and markers were the average colour. In other words, if the driver has driven on the same road more than once; the colour of the line is the average colour. Example the first time driving on the road the driving was good (green) the second time it was poor (red) then the resulting colour of that road is yellow. The map layout can be seen in figure 6.11.

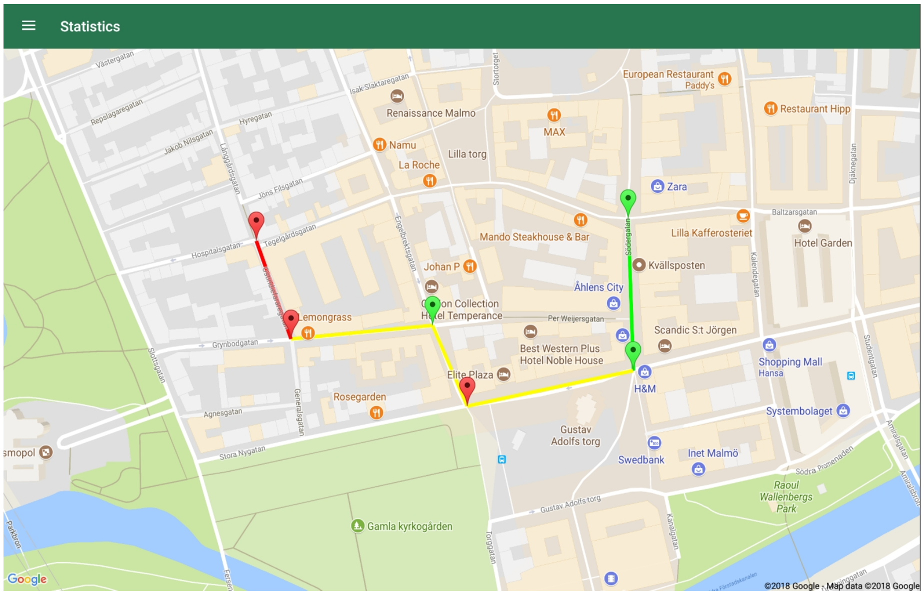


Figure 6.11: Map layout with coloured markers

6.5.2 Test of Score layout

The second iteration with the Score layout was to develop and test it. The Score layout was designed in three different ways: a circular view designed by ourselves, the second and the third were two different progress bars, one horizontal and one vertical. The low-fidelity prototype and the high-fidelity prototype can be seen in figure 6.12, 6.13, 6.14 and 6.15 respectively.

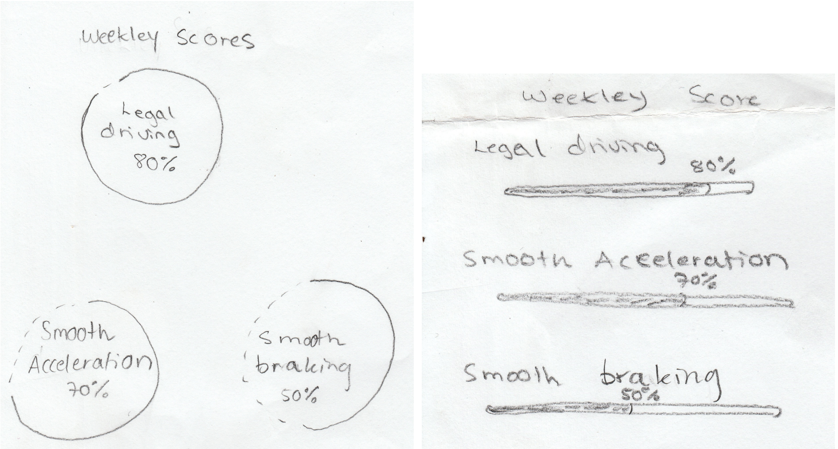


Figure 6.12: Low-fidelity prototype of score

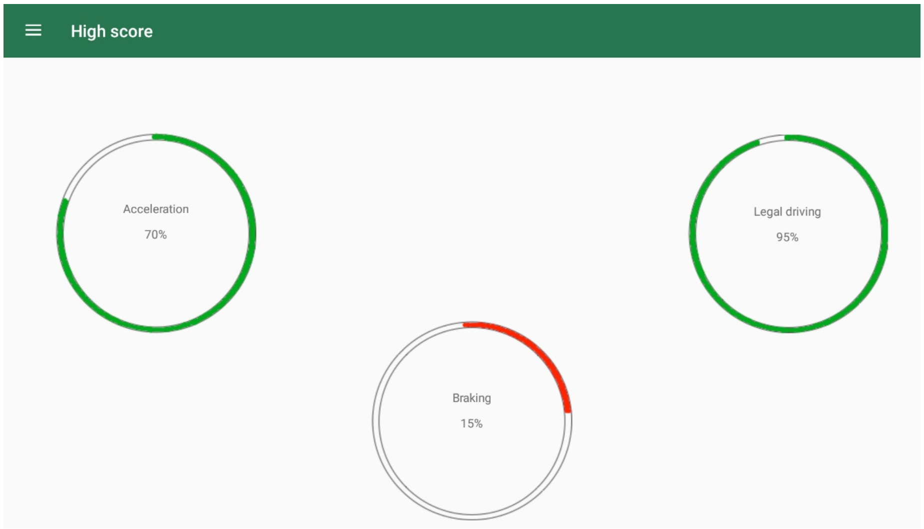
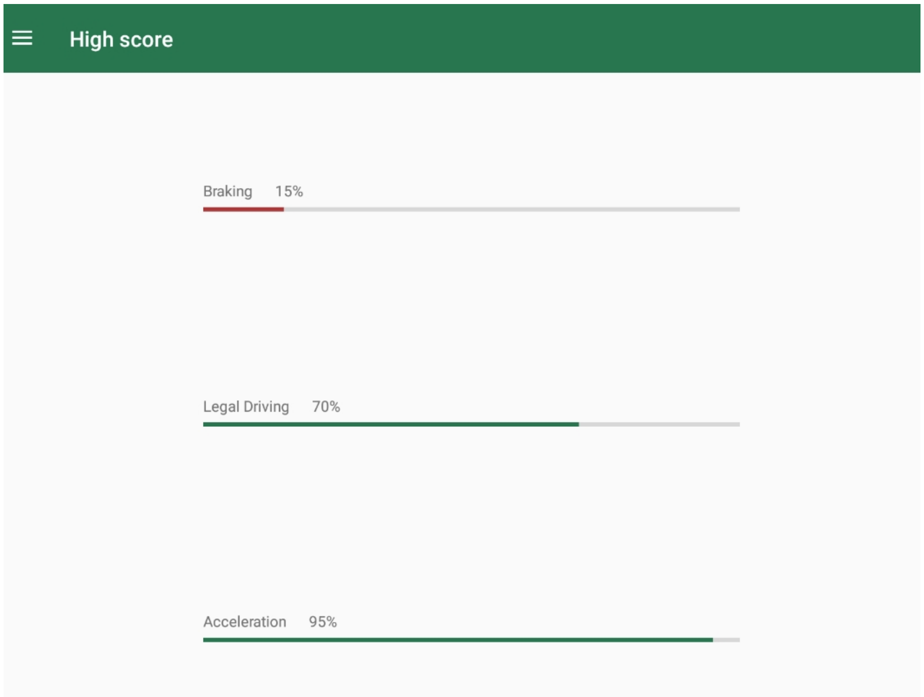


Figure 6.13: Circular score view for test 1



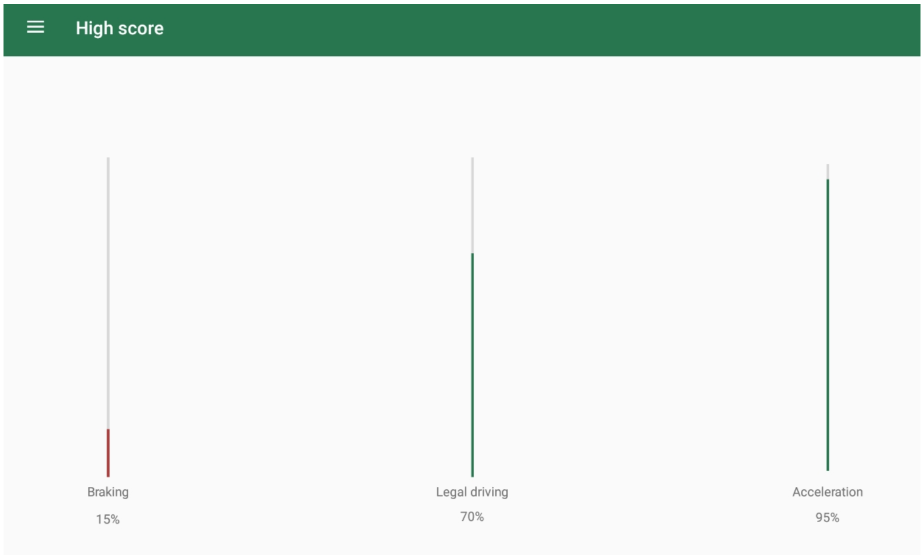
High score

Braking 15%

Legal Driving 70%

Acceleration 95%

Figure 6.14: Horizontal score view for test 1



High score

Braking
15%

Legal driving
70%

Acceleration
95%

Figure 6.15: Vertical score view for test 1

The first test was intended to illustrate the driving score with three circular graphs. One for acceleration, one for braking and one for legal driving. The participants commented that the design was motivating and friendly. They also said that the design resembled a personal training application. This was a good reflection because eco-driving can in many ways be compared with a training application but instead of e.g. running, the user is driving. After the test the circular graphs were upgraded to a design that look like a speedometer, which fit our application very well (see figure 6.16). The speedometers were found in [41], developed by the user anastr.

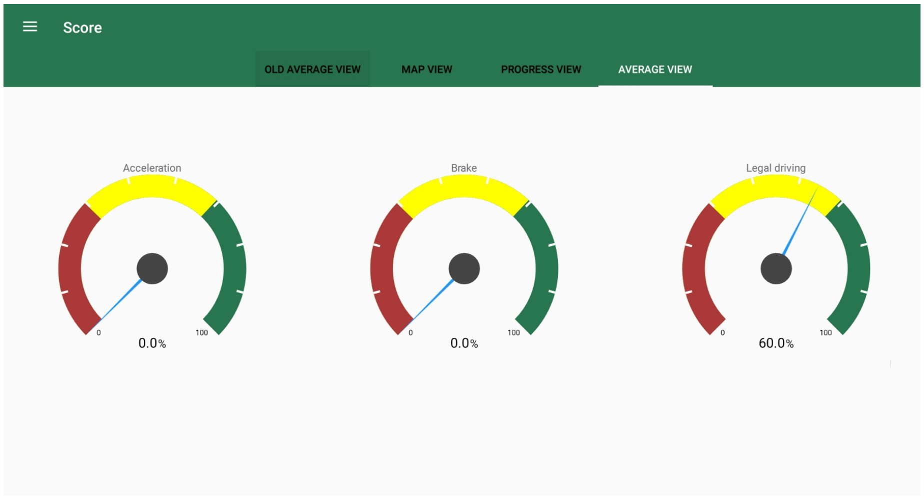


Figure 6.16: Circular average score shown by speedometers

6.5.3 Improved Score layout

During the implementation of the circular graphs we also upgraded the score view with tabs and added a view called progress where the drivers can see their progress. Besides upgrading the circular graphs, we also added a calendar in the header were the user can choose which dates he or she are interested to get information in between. We also made the layout scrollable using the ScrollView from the Android library [42]. The improvements of the score layout can be seen in figure 6.17, 6.18 and 6.19.

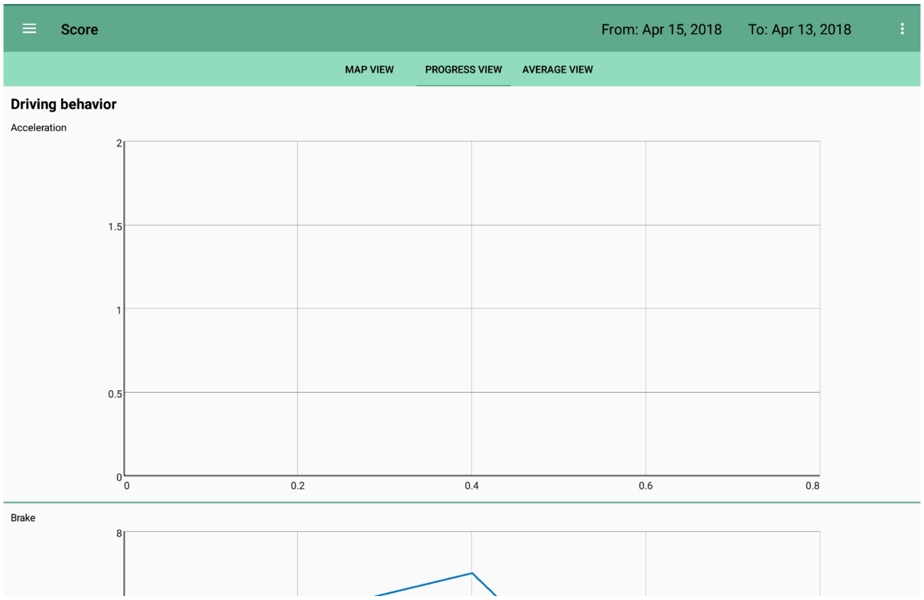


Figure 6.17: Progress view for Score layout

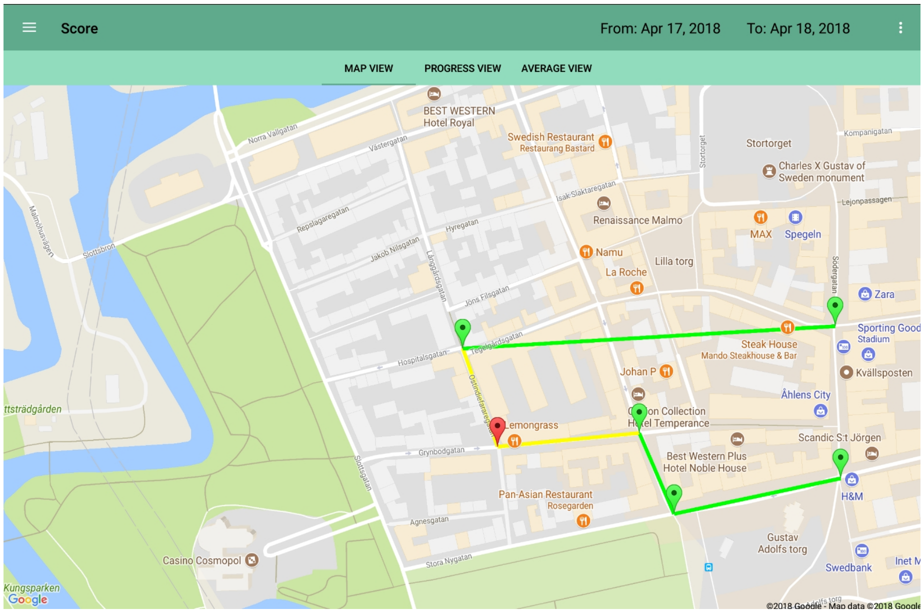


Figure 6.18: Map view for Score layout

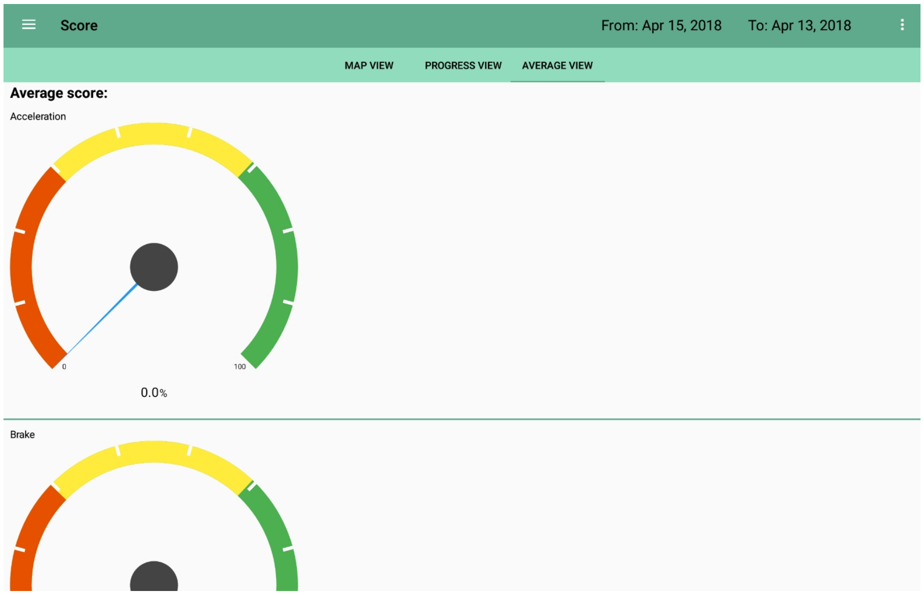


Figure 6.19: Average score view for Score layout

6.6 Application design

The application design has in many ways changed during the project when we discovered more about what was possible and what wasn't. We decided to design the application in a green colour scheme and by the help from the Material Design colour [43] tool we chose one primary colour and then the program helped us to choose other good matching colours.

CHAPTER 7

Result

In this chapter we will present the final product, the final test, the machine learning model and the system design. Sub-results like brainstorming, low-fidelity prototypes, high-fidelity prototypes, and other tests than the final one can be explored in chapter 6.

7.1 Final product

The final product is a prototype, that doesn't support functions for real-time data processing. The data used for testing is however real data from the same dataset that was used to train the machine learning model. The application is fed one instance from the dataset once a second to simulate a real drive.

7.1.1 Drive

The final drive layout was designed with a smiley that is both changing its look and colour in real-time as feedback. Beside the smiley there is a number reporting how good or poor the current fuel consumption is related to the machine learning model. This number is also changing in real time and is what decides the colour and expression of the smiley. Figure 7.1, 7.2, 7.3 and 7.4 shows some examples of the drive layout at different number values.

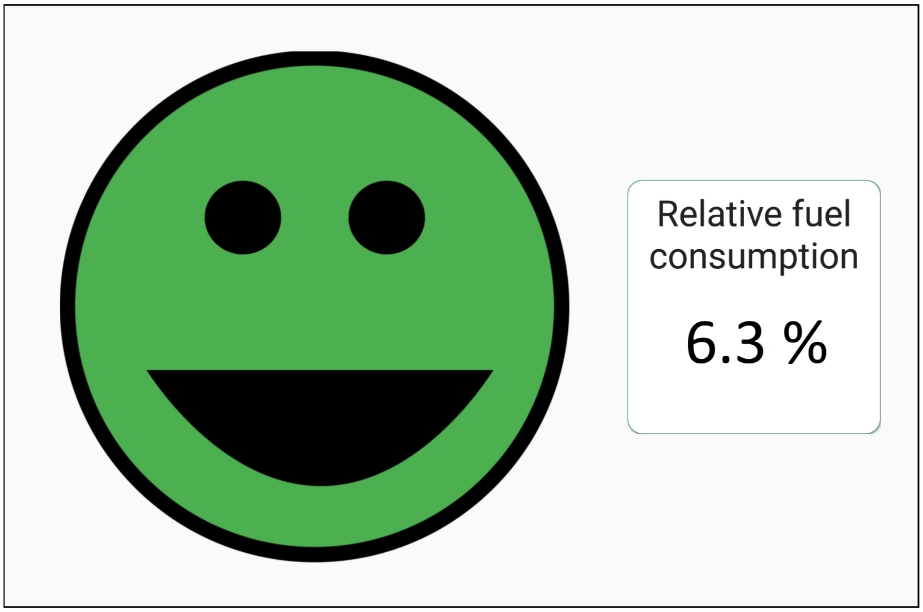


Figure 7.1: Final high-fidelity result for drive, green smiley

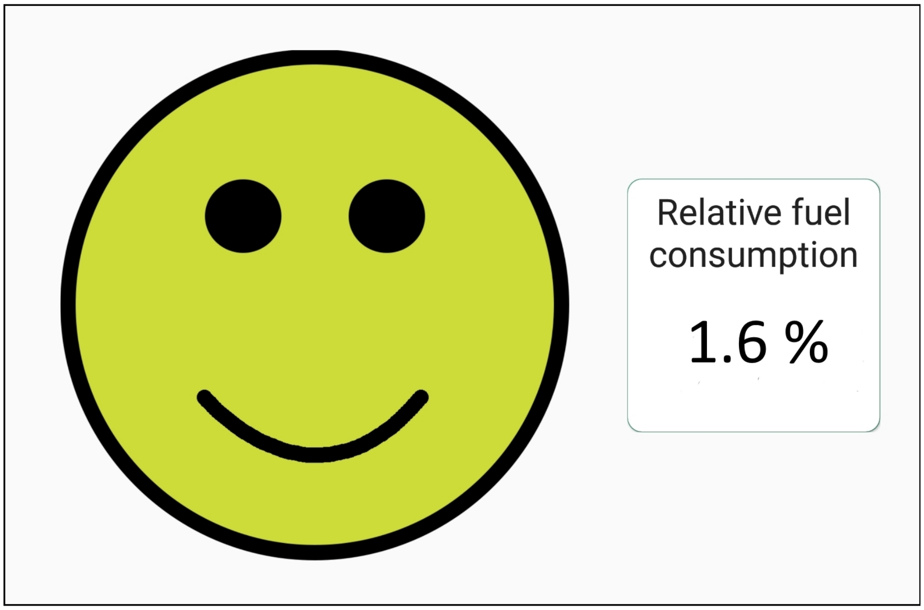


Figure 7.2: Final high-fidelity result for drive, yellow smiley

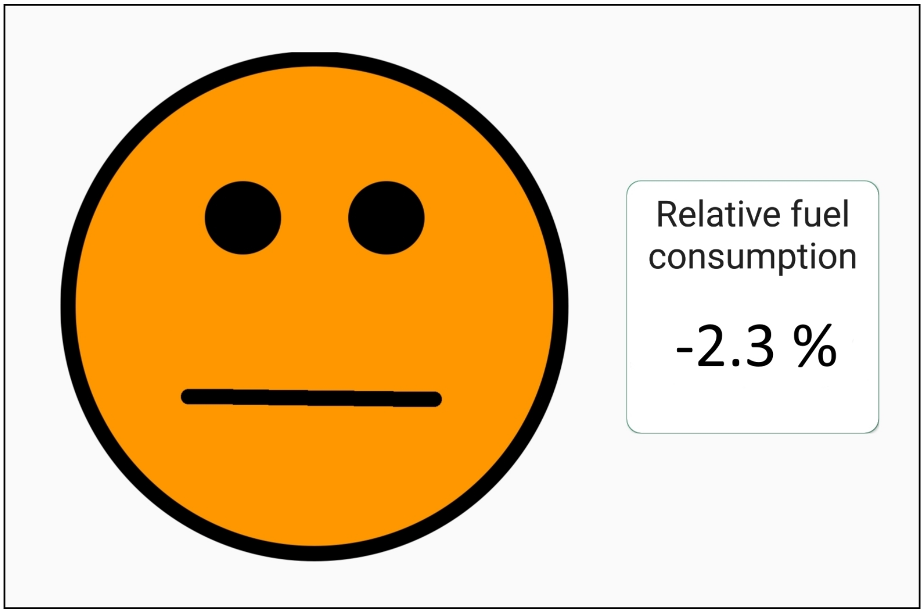


Figure 7.3: Final high-fidelity result for drive, orange smiley

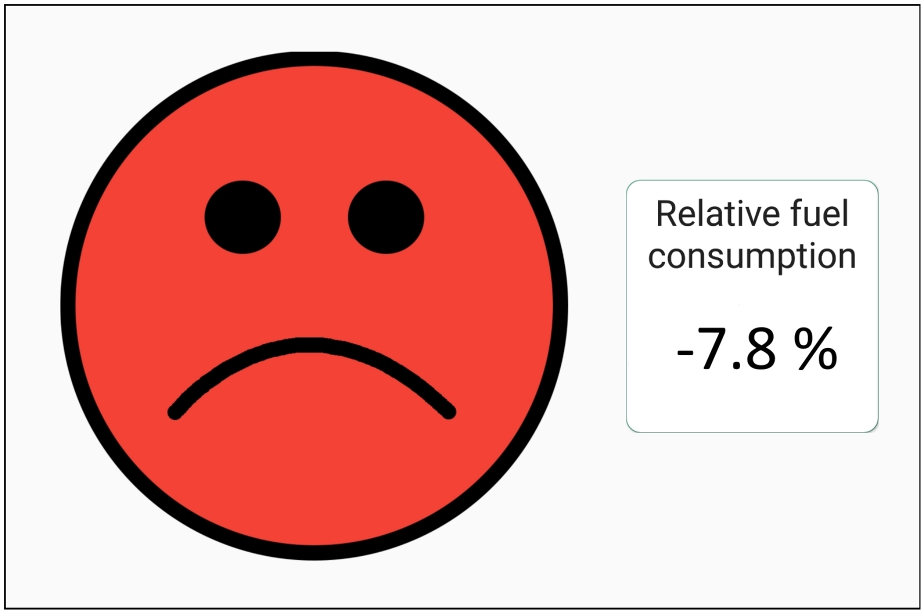


Figure 7.4: Final high-fidelity result for drive, red smiley

7.1.2 Score

The score view is divided into three tabs and a header with a calendar to select dates which is connected to all the tabs.

The first tab is a map using Android's Google maps. On the map, the user can see both where he or she has been driving and get information about the driving behaviour during the selected dates. The information is designed with both coloured lines and markers where the driver has been driving. If the driver has been driving on the same road more than once the colour is the average from all drives. By clicking on the markers, the driver gets information about how the acceleration, braking and gear was. We lack functionality for this, so the events are unfortunately made up.

The second tab is Progress, where the driver can see their progress for the selected dates. This view also holds the improvement score which is a measure of how well a driver is performing compared to the machine learning model. We defined it as:

Definition: Given the standard deviation σ and mean μ of driving scores, we can define an improvement area Im_{area} as $\mu + \sigma$ and a total deviation area D_{area} as $2 * \sigma$ then the improvement score Im_{score} becomes $\frac{Im_{area}}{D_{area}}$

The result is a score in the range $[0, 1]$ where a mid-range value of 0.5 is considered as no improvement while 0 and 1 is the worst and the best score respectively. The definition is more easily visualized in figure 7.5 where the improvement area is the red area and the total area is the red and the blue area.

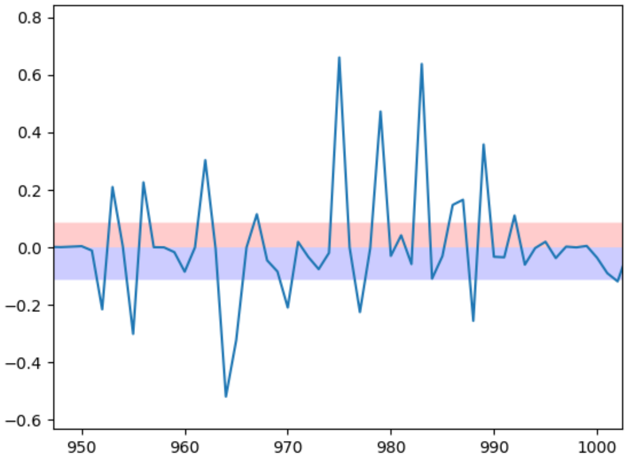


Figure 7.5: Definition of improvement score

The third tab is the Score tab, where the driver can see their eco-score, driving distance, and driving duration. The eco-score is a measurement of how smooth a drive is. That means if a driver has few extreme values we consider the drive to be good. It is defined as:

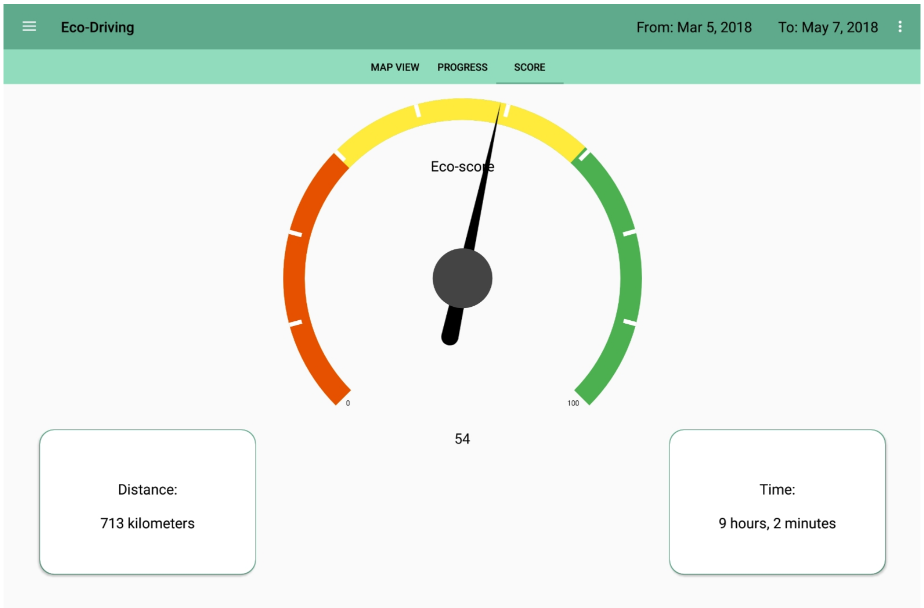
Definition: The count of instances where the driving score is placed within the total deviation area D_{area} divided by the count that is located outside D_{area}.

The implementation in code of the improvement and eco-score as it's done in the AWS Lambda function can be seen below:

```python
def eco_score(scores, avg, dev):
    """
    :param scores: Each score from the real-time classification.
    :param avg: The average of the scores.
    :param dev: The standard deviation of the scores.
    :return: The eco-score
    """
    num_outside = 0
    for score in scores:
        if score > avg + dev or score < avg - dev:
            num_outside += 1
    return 1 - num_outside / len(scores)


def improvement_score(dev, avg):
    """
    Ranges between 0 and 1 and a value over 0.5 means improvement.
    :param avg: The average of the scores.
    :param dev: The standard deviation of the scores.
    :return: The improvement score
    """
    deviation_area = dev * 2
    improvement_area = avg + dev
    return improvement_area / deviation_area
```

The different tabs and the selection dates in the calendar can be seen in figure 7.6, 7.7, 7.8 and 7.9.



From: Mar 5, 2018 To: May 7, 2018

54

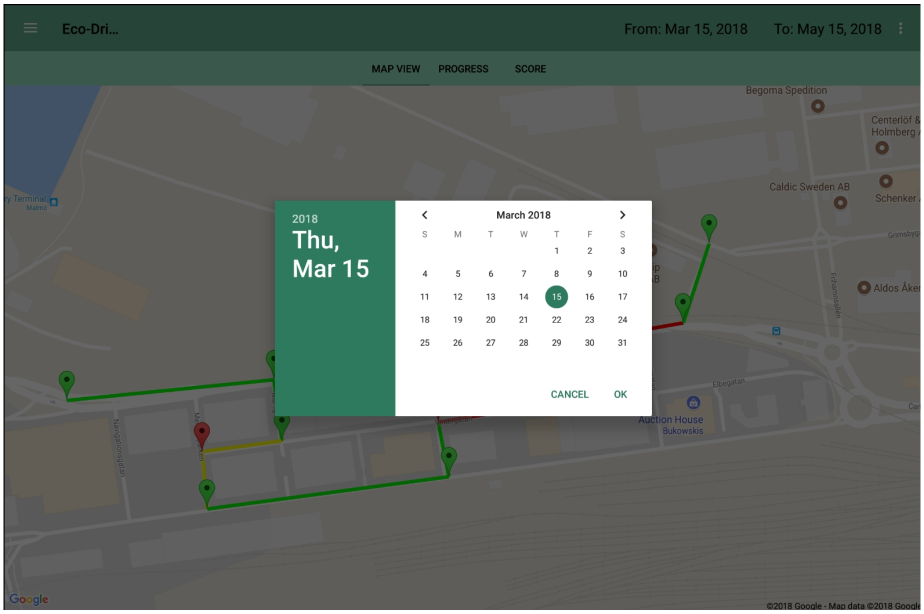
Distance:

713 kilometers

Time:

9 hours, 2 minutes

Figure 7.6: Final high-fidelity result, score tab



From: Mar 15, 2018 To: May 15, 2018

2018

Thu, Mar 15

March 2018

S	M	T	W	T	F	S
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

Figure 7.7: Final high-fidelity result, selecting dates in calendar

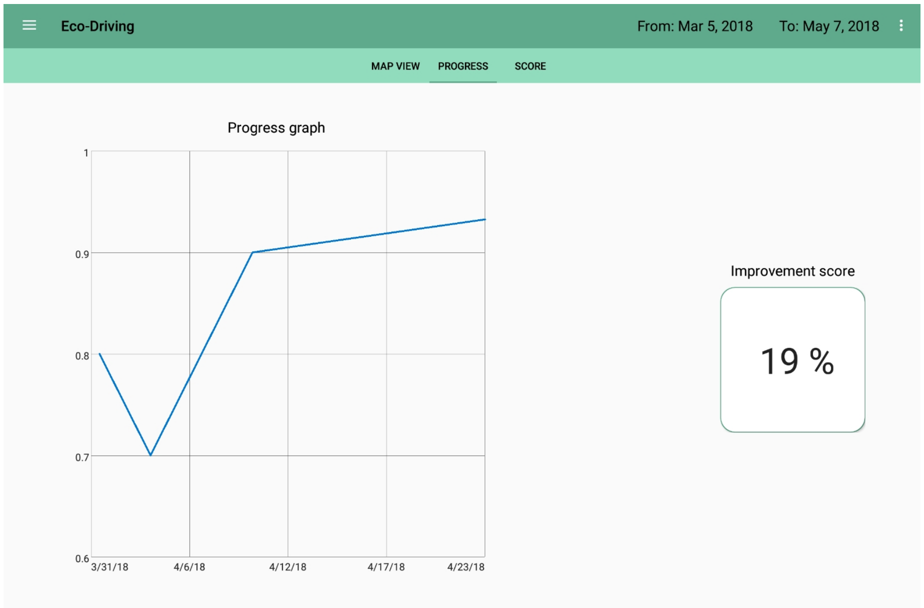


Figure 7.8: Final high-fidelity result, progress tab

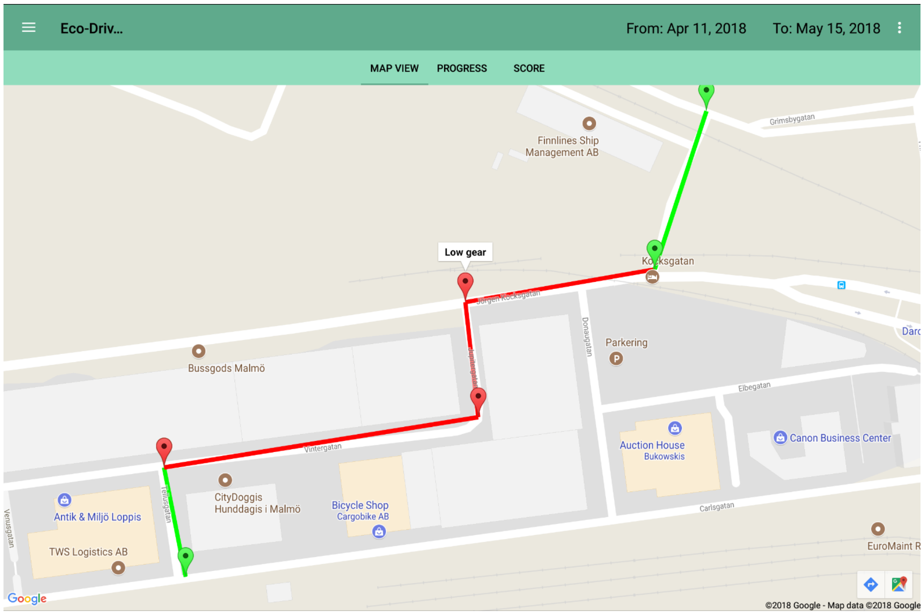


Figure 7.9: Final high-fidelity result, map tab

7.1.3 Other implementations

To make the application look even more like a game, a start screen and a stop screen were implemented. The start screen was designed and implemented with two buttons that were linked to the Drive layout and Score layout. By clicking on one of the buttons the user is taken to the respective layout. The stop layout is viewed directly after a drive and shows the eco-score from the drive. This view also has a button that takes the driver to the Score view. The high-fidelity prototypes can be seen in figure 7.10 and 7.11.

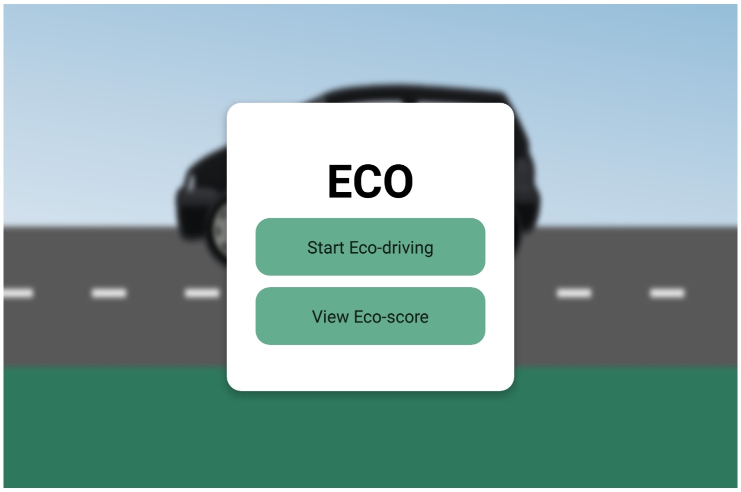


Figure 7.10: High-fidelity prototype of start layout

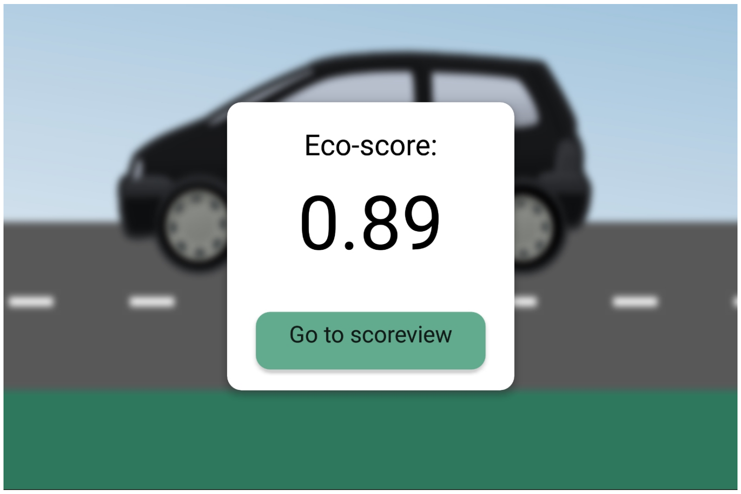


Figure 7.11: High-fidelity prototype of stop layout

Apart from the main options in the navigation drawer the final product also contains a Goal, Friends, and Settings layout.

7.2 Final test

The results from the test were compiled and can be seen below. No personal data is shown, only the result from the test.

7.2.1 Pre-test questionnaire

Before starting the test, the participant answered a questionnaire. The result can be seen in table 7.1.

Table 7.1: Pre-test questionnaire result

Pre-test questionnaire	
Characteristics	**Number of participants**
Pilot	2
Regular	19
Total number:	21
Age	
21–30	13
31–40	3
41–50	3
51–60	1
61-70	1
Gender	
Female	6
Male	15
Any experience of eco-driving assistant?	
Yes	5
No	16
Driving experience	
-Driving mostly the same routes (to and from work, to and from grocery store, to and from the gym, etc.)	13
-Own a car	10
-Driving with respect to the environment (eco-driving)	9
-No special experience	4
Driving frequency	
-Infrequently: 1 drive per month or less	8

-Moderately often: 6 - 12 drives per month	5
-Very often: 5 or more drives per week	8
Driving simulator experience	
Yes	10
No	11
Android experience	
Likert:	
Agree 1 2 3 4 5 Disagree	Agree 9 6 3 2 1 Disagree

7.2.2 Interview

A summarized result from the interview can be seen in table 7.2.

Table 7.2: Interview result

Interview	
Questions	**Answer**
How was the general experience after the test	Most of the participants agreed that the application was fun, interesting, and usable. They liked the colour-coded feedback of the smiley and thought it was understandable and motivating. Some of the participants had issues with connecting the app to the car game and said that they had probably reacted differently if they were using the application in a real car.
Were you stressed and/or distracted by the application while driving?	Almost every participant agreed that they weren't stressed during the drive. Some said that they were a bit stressed in the beginning, but after driving a minute the stress was gone. Most of the participants said that they weren't distracted either. Some said that they were a bit distracted when the smiley was changing too fast, but others said that the smiley was changing smoothly. Some of the participants also said that they probably would have been more distracted in a real car and if it was the first time using the app, but after getting used to the application there should be no problems.

How was the navigation of the application	Every participant thought the navigation was good, easy and intuitive. Some said that the navigation followed Android standards very well and that they therefore could take advantage of previous knowledge. Some of the participants had trouble finding the calendar in the header, but after finding it once it was never an issue again.
Was something unclear?	Some said that they understood everything and that nothing was unclear. Many had problem to understand if the eco-score showed after a drive was good or poor. It was also unclear that the eco-score in the score layout was between 0-100 and the eco-score showed after a drive was from 0-1. The relative fuel consumption was also hard to understand and the markers on the map was a bit confusing for some participants
What did you think about the smiley? Did you understand the feedback? What do you think about the colours	Almost every participant agreed on that the colours were good because it was possible to see in the periphery. They said that it gave them quick feedback and was easy to understand. Some were a little doubtful about the smiley itself, and said that it could be done differently, for example with a thumb. One also said that the yellow smiley could be something good if you are too used with the smileys for chat. For one participant the car game took too much attention, so he didn't pay attention to the smiley at all.
Were you motivated by seeing your fuel consumption score while driving?	Most of the participant didn't pay attention to the relative fuel consumption, they thought the smiley was enough feedback. Some said that the value was hard to understand, because they didn't have anything to compare with. Others agreed on that the value could have been usable and motivating in a real car and after using the application regularly.
Did you understand the feedback after driving?	Most of the participants understood the eco-score shown after a drive, but they were a bit confused about the scale of it because it was from 0-1 and the eco-score in score view was from 0-100. Some said that it was hard to know if the eco-score was good or poor.

How was it to get feedback on a map?	Most of the participant agreed that the map was good, usable, and motivating. They said that it was cool to see the drive on a map and nice to be able to zoom. Some had issues connecting the dates in the calendar to the map and didn't understand that the colour represented the average value. Some also said that the information when clicking on the markers were confusing and wanted more information about them.
What was your perception of the progress graph and improvement score? Did you understand its meaning?	Most of the participants thought the progress graph was usable, nice, and good. Some had problems to understand the x-axis and y-axis and where the data came from. The improvement score was hard for some to understand. One said it could be misleading if you are driving poor one day because it should be okay to have one poor day. While other said that after seeing the improvement score changing they understood it and thought it was helpful because it gave them a good overview.
How was the design of the app?	The overall reflection on the application design was that it was good, usable, and following the standard look-and-feel of an Android application. Some said that it was sometimes hard to know where to click, for example when trying to find the calendar. One said that the application was maybe to clean and therefore get a feeling that the application isn't finished. One said that it in some ways felt like two different apps, because of the full-screen for Drive and tabs in Score.
Did you miss any feedback?	Most of the participant said that they want more feedback about the driving behaviour, so they know how they should change their behaviour to improve their score. Some said that they wanted more details in the graph and more details about the markers in the map. One said that it would have been cool and fun to see how much fuel or money it saved because of the eco-driving. Another said that they wanted to have information about the speed and other driving data in the score view.

Do you want audio feedback?	Most of the participant said no to audio feedback because it would have been too disturbing. There is already enough noise in the car. Other said yes, but most of them wanted to be able to turn it off. Some said that audio feedback is especially good during the night because then you can turn off the screen and just listening to the voice. There were also slightly different opinions about if the sound should be implemented as a voice or tone. One also said that the audio feedback could maybe be implemented as a vibration.

Improvements mentioned by the participants during the interview

- Eco-score:
  - The eco-score shown after a drive should be viewed together with a previous eco-score so it's clear if it's good or poor.
  - Eco-score shown after a drive should be designed as the eco-score in the score view
  - Eco-score should be shown as how much money or fuel the eco-driving led to.
- Progress:
  - Progress should be based on the average instead of start to end.
  - Zoom and point function in the graph to get information on a specific day.
  - Should be shown automatically after a drive
  - Have two lines in the progress showing this month/weeks and the previous month/week
- Map
  - There should be more information in the map about the markers, it's not enough with the acceleration, brake and low gear text when clicking on the markers.
  - It should be possible to choose a radius with markers instead of a specific point in the map
  - Should have been designed as a heat-map instead
  - More advice on the map, which will lead to a greener map

  - Directions in the map
  - Information about how many times the driver has driven there.
  - Map as a navigator during the drive and just have the smiley in the corner of the map
  - Speed information by the markers

- Other
  - The selections in the calendar should be month or week, instead of dates.
  - It should be clearer that the calendar is clickable
  - Only a colour screen instead of a colour smiley when driving
  - A button in every view or just in score view that is directly linked to the drive, so it's possible to start driving fast and easy
  - Have an option where it's possible to get more information on how to improve the score and get traffic information.
  - Have an option where it's possible to get more information about other drivers
  - Information if driving the same route as someone else, more gamification
  - Select both dates in the same calendar, don't want to open two
  - Be able to talk to the app during the drive and ask questions to the app
  - Be able to see driving history during the drive.
  - Information tab about the application and eco-driving
  - Drive should start when starting to drive
  - In Score view there should be one more tab directly linked to drive.
  - The smiley should be replaced with an eco-score like eco-score in score view.

7.2.3 SUS

The average result from the SUS questionnaire can be seen in table 7.3. The calculation can be seen in appendix C.1.

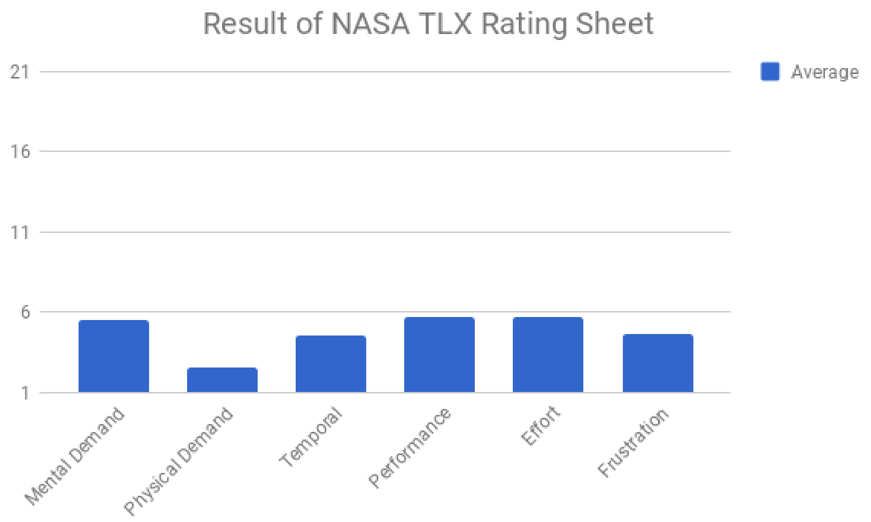
Table 7.3: Average SUS result. Raw score is calculated as the summarizing of all odd numbered questions (items 1,3,5,7 and 9) minus 1 and 5 minus all even numbered questions (items 2,4,6,8 and 10). Final score is calculated as Raw score multiply 2.5.

SUS Raw Score: 35	**SUS Final Score:** 87

7.2.4 NASA TLX rating worksheet

Due to the lack of time the participants only answered the first part of the NASA TLX, the rating worksheet. The total and average result can be seen in figure 7.12. The calculation can be seen in appendix C.2.

Figure 7.12: Average result from NASA TLX rating sheet



7.2.5 Word-cloud

The result from the word-cloud can be seen in figure 7.13. The word list used in this project can be found in appendix B.

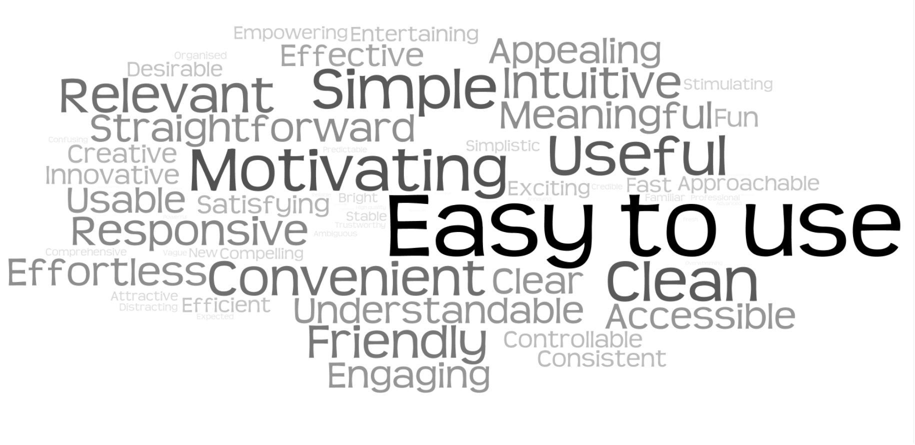


Figure 7.13: The resulting word-cloud from the final test

7.3 Observation

The result from the observation of the test can be seen in table 7.4. Participants were asked to think aloud, so some of the notes are what the participants said during the test.

Table 7.4: Average result from the observations.

Task	Notes
1	No participant had any issues.
2	All participants understood the meaning of the colour-coded smileys. Some found it distracting but most participants said they wanted to make the smiley happy.
2	Most participants said that since they don't know the scale of the score they can't know if it is good or poor. Some guessed the right scale.
4	No participant had any issues.
5	A few participants had trouble finding where to change dates. Most found it without any issues and those who didn't, found it fast after looking around a bit.
6	All participants understood the colour-coding and that the lines represented where they had driven. A few noted that they would have wanted to know in which direction they had driven.

7	All participants intuitively clicked the markers to get more information about them. There were various interpretations of the markers. Some thought the marker noted an event that took place at that point and some thought they were related to the lines.
8	No participant had any issues changing the dates now and everyone understood the score view.
9	All participants interpreted the progress graph correctly. Most participants interpreted the improvement score as the improvement over the entire date range, but some interpreted it as the improvement since the latest drive.
10	No participant had any issues.
11	Most participants used the navigation drawer to start driving again. Some used the back button at first but used the navigation drawer when asked if they could find any other way.

7.4 Machine learning implementation

During the initial design phase, we landed in three different ways of applying machine learning and those were driving style classification, action prediction, and difference classification. We eventually decided to go with the last option, difference classification, which resembles the way machine learning was applied in our main related work [3].

Definition: Given current state S where $S = [x_1, x_2, ..., x_n]$ create a model M that takes S as input and predicts the next state S' so as to $M(S) = S'$.

7.4.1 Model design

It's a simple model with only three components. The input layer is exactly what it sounds like, it's a layer that receives the input data. The LSTM layer is the LSTM cell explained under Theoretical Background [26]. Finally, the Dense layer is simply explained as layer with one node that is connected to all output nodes from the LSTM layer [44]. The reader should take special note of the input shape (1, 5, 27) which is because we have 1 instance of a 5 seconds time frame described by 27 features.

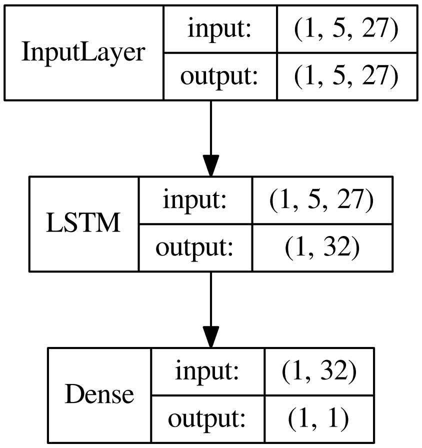


Figure 7.14: Model overview

7.4.2 Calculation of drive score

The problem-solution method 5.3 used for solving this proved itself to be very effective. It was clear early on that distribution had to be used and we just had to decide which one.

Below is the resulting code implementing the classification functionality. The steps in words are:

1. Define a new normal distribution with the expected value (`refValue`) as mean.
2. Calculate the probability density value for the expected and the actual value.
3. Find the ratio between the two density values and the result will be a normal distributed value between 0 and 1.
4. Invert the score and multiply with -1 if the actual fuel consumption is more than the expected. Now we have a value between -1 and 1 instead depending on if the score is considered good or poor.

```
private double densityRatio(float refValue, float actValue) {
    NormalDistribution nd = new NormalDistribution(refValue,
        STANDARD_FUEL_DEVIATION);
    double upper = nd.density(refValue);
    double lower = nd.density(actValue);
    return lower / upper;
}

public double classify(float[] data, float actualFuelConsumption) {
    float expFuel = predictFuelConsumption(data);
    double score = 1 - densityRatio(expFuel, actualFuelConsumption);
    if (actualFuelConsumption > expFuel) {
        score *= -1;
    }
    appendScore(score);
    return score;
}
```

And the result is a range that is easily divided into intervals where each interval represents a class (or smiley).

7.4.3 Final evaluation

In figure 7.15 the normalizing function and the predictions are displayed. The top graph shows the expected vs the actual fuel consumption and you can see that they follow each other well. The bottom graph shows the normalizing function where the difference between expected and actual fuel consumption has been transformed into the drive score. The most interesting part to notice here is that the peaks and slopes in the normalizing function graph corresponds to the differences in the top graph.

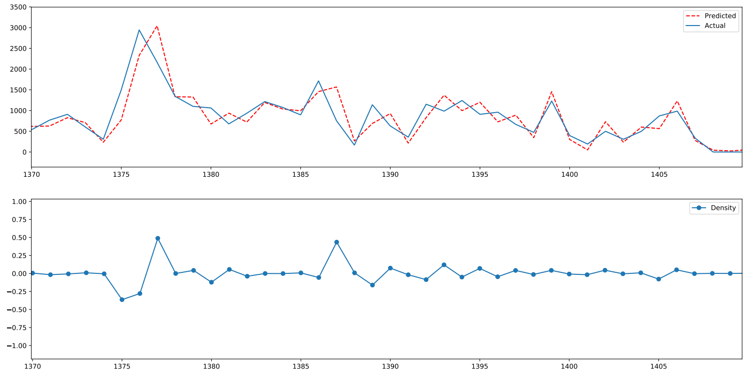


Figure 7.15: Classification and prediction chart

7.5 System design

Our system is straight-forward and can be explained in a few simple steps:

1. The app collects OBD data and stores the real-time predictions.
2. The data is posted to a DynamoDB table.
3. A lambda function is triggered and performs the post-drive calculations.
4. The new calculated data is posted to another DynamoDB table.
5. The app fetches the new data and displays it to the user.

This system design is illustrated in the figure 7.16.

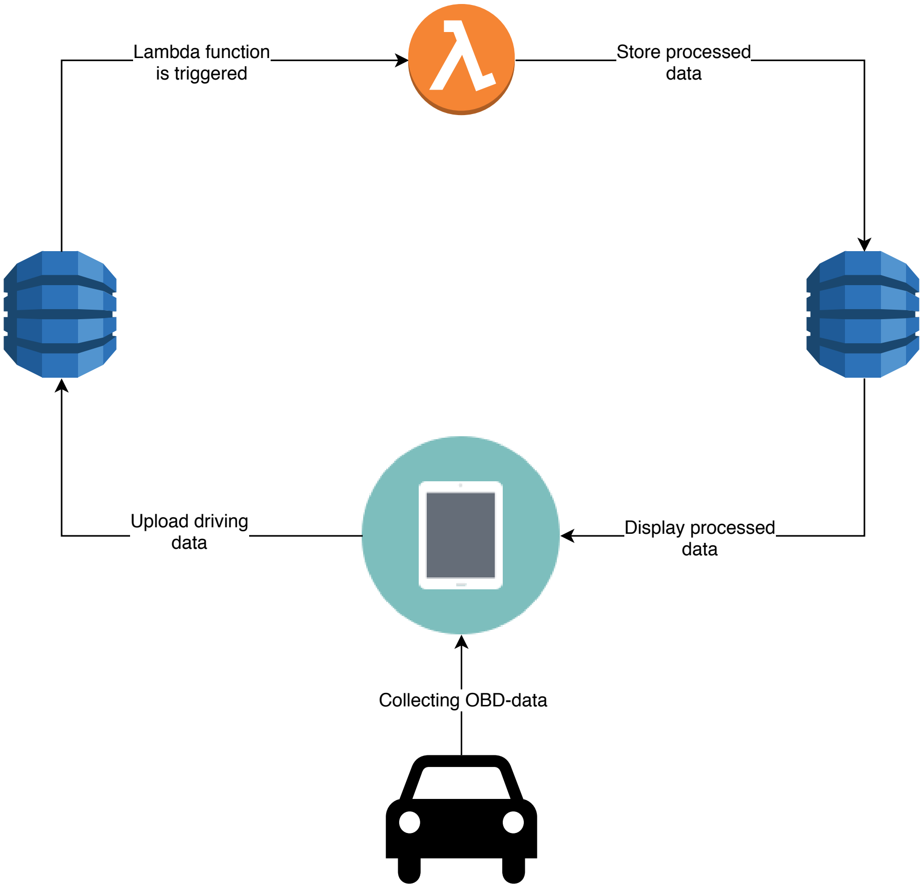


Figure 7.16: System design

CHAPTER 8

Discussion

In this chapter we are going to discuss the result of tests by comparing the result with earlier information from articles and previous hypothetical thoughts. We will also discuss methodology and tool selection. Finally, we will address various points that we could have done better during the project. There will also be a discussion about future implementations.

8.1 The application's emergence

Our goal of the application was to make the application useful and with good user experience. To achieve this, we have throughout the project put the user and their need in focus. The application grew through many iterations, when we discussed each week what could be done differently and improved it. We were also assisted with advisory from third parties. During the project, in addition to small and short tests, three major and more thorough tests were performed. These tests had three focuses:

1. Develop a design for Score and evaluate why this design best suited Score.
2. Test different image proposals with the goal of designing a Drive design and evaluating why this design fits best and why the others don't.
3. Test with the goal of testing both Drive and Score in relation and interaction between each other.

8.2 The final test

The final test was the most thorough one and during this test the participants were driving in a car simulator. The first two participants drove in a rally game, which gave a little misleading result, we then tested to let the participants drive a car game more like regular city driving. This suited us a lot better and allowed the participants to drive with more realistic traffic. For the final test it should be taken into consideration that:

- We have not tested the application in a real car, we have only been using a car simulator which, of course, may have affected our results.
- 13 out of 21 were in the age of 21-30, the rest was between 31-70.
- Most participants were men (25% women).
- 9 out of 21 said that they already employ eco-driving.
- Nearly half of the participants had previously tested a car simulator.
- The majority had Android user experience.

In the description of the final test, we described that we would have objective/quantitative data by counting successful tasks, errors, interventions and calculating time per task. After a few tests, we realized that it was not worth it to measure time, as all test data followed the schedule. We also realized that it was just as good to take notes of the participants errors, instead of counting them. During the test, the participants was told to think out loud which also was noted.

The overall reflection of the application design was that it was good, usable, and followed standard Android design. That it followed standard Android design meant that with Android experience they could take advantage of their previous knowledge and easily understand how the application behaved. Which follows Norman's principles for user centered design. A detail that did not follow Norman's design aspect "visibility" good enough was the calendar function, which was placed in the header of the Score layout. Some participants found it difficult to find it at the beginning. The reason to that was probably that the function didn't show that it was clickable. It's also not common to have a calendar in the headline. They also pointed out that after finding it once, it was easy to know where it was. Another comment about the application was that it was clean and that it was almost too clean, which gave the feeling the application was not finished. The reason we designed it so clean was because we just did not want anything unnecessary and wanted all the functionality to be visible and obvious. There was also a person who pointed out that Drive and Score felt like two different apps. This is because they only tested two parts of the application and the application is unfortunately not a fully functional application but only a prototype, but with the correct OBD-data (3.3).

Jeff Sauro, a statistical analyst says that a good product gets an SUS value between 70-80 and outstanding products 90 or more [17]. After calculation, our average SUS score was 87 (table 7.3). Which means it's almost outstandingly good from usability point of view. Here, however, it should be added that SUS may not be particularly suitable for a prototype, but rather should be used on a fully functional product. We included it in this project to get a quick and straightforward response on how useful the application was. In the project, we also used the NASA TLX, but we only took the rating sheet, because we did not feel we had time to perform more tests. The NASA TLX also provides a quick answer as to the usefulness of a product. As with SUS, the average performance of the NASA TLX rating sheet was also very good. Which may have been affected of the same reason as with SUS. Another reason could also be that we did not perform the second part of the test, which weighs different measurements based on which one considers the most important.

Looking at our word-cloud (figure 7.13) we found that the participant's user experience was also positive. The application's most suitable descriptions were "Easy to use", "Clean", "Useful", "Motivating", "Simple". It also felt like these words were consistent with what the participants said in the interview, which indicates that they not only chose nice words to be nice to us. However, we could see that one of the pilot tests was not as positive to the application. The reason could be related to that person was driving a rally game and couldn't connect the car game to the feedback from the application. When driving a rally game, it's easy to get distracted.

8.3 Gamification in a car

Designing applications for cars is different from designing common applications. When designing for cars the designer both want to improve the driving experience and at the same time make the application as non-distracting as possible. Android Auto developed by Google describes that it's important that the UI is simple, so that the driver is watching the road and have the hands on the wheel [45].

In [22], the author describe how eco-driving in relation to gamification today occurs in cars. The author lists a lot of quotes from different car manufacturers who explain how they think about subject. Earlier in the report, we noted that Dan Bonawitz of Honda believes that by incorporating the feeling of fun; the driver improves his driving. Washington [22] have the same opinion and expresses "Applying the fun, engaging and rewarding aspects of games to travel planning can allow people to improve their commutes, track their success and become aware of how their behaviour impacts the transport infrastructure as a whole." According to him, an application should be both fun, engaging, and rewarding. When the application fulfils these three words, the user will use it and want to improve. In the interview during the final test, some said that they thought the application was fun, but that was nothing we found in the word-cloud. However, the word "engaging" was prominent.

From the final test, we also saw that the participants were motivated to drive better by seeing the smiley changing its colour. That some participants said that the colours triggered them to drive better and they wanted to turn the smiley green. During test 1, almost every participant agreed that the circular graphs illustrated the score best. Some pointed out that the circular graphs reminded of graphs usually found in training applications. They pointed out that it was easy to understand and that it motivated them.

In [22], Nass ends with a quote: "If you make it too much like a game, you'll have people concentrating on the game and not on the road, if you're driving and thinking only about the environment, you're going to smack into a tree." This quote can feel obvious in many ways, but it is easy to forget it.

8.4 Feedback while driving

In [22], Jeff Greenberg a technical leader at Ford, explains why they use virtual leaves to symbolize good eco-driving. He says that the metaphor is a way to both get an emotional response and in the same time trigger those who are interested in the data. Cliff Nass, whose work with Stanford University CarLab focuses on the psychology of making cars safer and more enjoyable says a very interesting thing in the article and that is that "People will enjoy adding leaves to their trees, and they're going to be ticked off if these leaves disappear"[22]. After reading the article, it was easy to get the feeling that Nass believed that the positive feedback is important. The negative feedback is just disturbing and makes the driver angry.

Nass quote felt fair and exciting, which was one of the reasons to why we in this project designed a drive view with animated flowers that mostly focused on the positive feedback. During test 2, as mentioned earlier in the report, we tested five different design suggestions to Drive (Triangle, Smiley, Driving Car, Static Car and Flower). Prior to the test, we thought that the driving car would be the one that won, as it had followed us throughout the design process and was really the design that resembled the design we had drawn up during our initial brainstorming. We therefore spent the most time on it to make the animation as good as possible. We also spent a lot of time designing the flowers so that it was growing in a nice and credible manner. The result of the test was not as expected. As a result, the layouts we had spent the last time on, namely the triangle and smiley were the ones that worked best. The reason for this was that they were easy to understand and gave quick feedback. Unlike, for example, the flower, which required far too much attention. For the car layouts, it was not as expected either. The car required too much focus, which meant that you did not notice the exhaust in the image. There was also some that did not see any difference between the animated and the non-animated car. Which indicates that the animation of the car was unnecessary.

Back to Nass's quote about the positive and negative feedback. Based on test 2, the general opinion was that the participants wanted both positive and negative feedback. There were even some who only preferred negative feedback because it made them react. It should again be mentioned in this discussion that we have not tested this application in a real car. Some of the participants pointed out that they probably would have reacted differently if they had tested it in a real car.

The final design of Drive became the smiley as it both gave the user positive and negative feedback, because of its simple and clean design. It was, in practice, impossible to misinterpret the smiley due to how well known both a smiley and its colour-coding is. A smiley is also a good symbol if the driver is colour blind because then the driver can at least understand the facial expression. However, not everyone agreed that the smiley was what they would prefer in their car. One suggested that they wanted a thumb in different colours and someone said they would rather have had an eco-score like the one in Score. How well these suggestions would serve as good feedback in a car, is beyond this report.

During test two, there were some who suggested that they would like a value next to the smiley that showed how well they were driving. Therefore, we added a value called relative fuel consumption. This is a value for the fuel consumption in relation to the machine learning model. Although we did as participants from test 2 said, it turned out that this value for most participants in the final test did not matter. Most were more interested in the smile and its colours. This can partly be because the smiley took quite a lot of space on the application screen in relation to the value. It may also be because those who participated did not understand what the value meant or knew if it was good or poor. During the interview, however, it was found that the value had been motivating in the long run of the application. Some said that because this was their first run, it was not possible to compare the value with any previous value.

In the beginning, we considered to add audio feedback in addition to the visual feedback. The project's first prototype was implemented with sound, but this was removed after a few iterations when we wanted to focus on the visual. We still planned to revisit the area when we were finished with the visual, but we needed to test it first because we suspected that it might be distracting. In the interview from test 2, we asked the participants if they would be interested in audio feedback the answer was somewhat doubtful. They said that they might have liked it but that it could also be very annoying. Therefore, we did not include it in the next design proposal. However, during the final test, we were still interested in knowing if those who participated would have wanted audio feedback and again time there were doubts. Some said no, saying that it would be too distracting and claimed they would have shut down the application. Some also replied that there is already a lot of noise in a car so if there were any sounds from the application, it would be noisy and disturbing. Those who said yes to audio feedback agreed that there should be a possibility to turn it off. One mentioned that it was especially good at night when it might be disturbing to see a big bright smiley while it is dark outside. There were also some mixed opinions about how the sound would have been implemented. Some of the participants wanted a voice, others preferred a sound or vibration. We didn't investigate the subject any further, but we do see it as a future complement to the visual feedback with the option to turn it on and off.

8.5 Feedback after driving

For our application, it was possible to get feedback about the driving in three different ways, on a map, through an eco-score and by seeing the progress on a graph. Because we wanted our application to have a more game-like design, the user of the application received an eco-score from the current drive immediately after it was complete. Many participants had difficulties in understanding this score. The need for an easy-interpreted scale was prominent because a value on its own doesn't say much.

Most participants in the final test agreed that it was good to get feedback about their driving on a map. They said, among other things, that it was good, motivating and useful. They also said it was cool and rewarding to see where they had driven and to have the opportunity to zoom in on the map. However, there were problems with understanding that the lines between the markers were coloured with the colour representing an average value. The reason why it might not be so clear was that some believed that the data on the map was where they have been driving with the car simulator. This lead to that it was not realistic for them that they could have been driving there twice since they didn't drive on the same place twice while using the car simulator. Another problem with the markers was that there were too few data points between the dates they chose, which meant that there were no major conclusions. There was also some commenting that they wanted to know which direction they had driven. Had they seen the direction, the meaning of the lines would probably have been clearer. There was also someone who suggested it would have been good to have a counter of some kind that showed how many times they had been driving on a road. Most of participants understood that the markers were clickable, but they wanted to know more about the reason to the text saying what was good or poor. Some also said that it was not obvious that the markers were clickable, and it should have been better if the lines were clickable and giving the information instead. To make the lines clickable instead of the markers might have been a more logical design and something we should take with us for a future implementation. Apart from that the participants wanted to get more information about what was good and bad, they also said that it was nice to see exactly which road they had been driving badly, so that they in an easy way can improve and turn that road green.

The majority of those who participated in the final test liked to see their score in a progress graph, they said it was useful and nice. There were some who had problems with the x-axis and the y-axis which is understandable because they are missing scales. Even the improvement score was difficult for some to understand. However, it was some that said it was nice to get a figure on their development and said it was motivating and helpful. There were some who pointed out that the improvement score should be calculated in a different way, so it is okay to fail one day without effecting the score.

8.6 Non-distracting communication

Almost every participant agreed that they were not stressed during the final test. There was someone who said that he/she was a bit stressed at first but after driving for a while, it went well. This stress may not be the application's fault, it could have been that the participant was driving in a car simulator for the first time. There were not many that said that they felt distracted by the application, which is good. It was someone who pointed out that the smiley's animations sometimes changed a bit too quick, which gave the feeling that the application was flashing. However, there were others who said the opposite and said that the smiley was good and was changing colour calmly and smoothly. How the animations should best be implemented so that the communication will be as non-distracting as possible is, probably individual. For example, our oldest participant in the test did not notice the smiley at all during the test, which may have been an exception, as this person had not before been driving a car simulator. It can also be related to the person's age and vision. This indicate that communication between the user and the application may be designed differently depending on who is driving.

8.7 Machine learning

Here we will discuss the implications of using machine learning in an in-car application and what issues and successes we faced during development.

8.7.1 Data filtering and processing

The filtering is highly affecting our result because we are actively excluding information based on metrics we've set up so it's reasonable to ask ourselves how well defined those metrics are. In our pre-processing step we are currently filtering out data based on a correlation threshold and basic data visualization properties. We consider these methods to be well-functioning considering that we get a well-functioning result from the model. However, we didn't dig very deep in this area so it's reasonable to believe that an even better result can be achieved.

Data filtering and smoothing is often critical when dealing with machine learning and is an area of research by itself. We could have spent all our time on this subject, so we reasoned that a decent result is good enough for us to move on with using the data in our model.

Another point to make is that we're lacking a noise filter completely. We experimented a bit with outlier detection during the initial phase before we decided what kind of model to use but no further exploration was done as to how it would affect our result. It would have been interesting to test some form of smoothing to get rid of extreme values such as emergency braking and similar events. This kind of events are impossible to predict given just the vehicle data and therefore such events shouldn't be allowed to affect our model.

Before we started the project, we didn't research how much driving data was publicly available. We knew that there was on-board diagnostics (OBD) and data could be recorded directly from a real car, but we never reflected upon how hard it would be to collect enough data by ourselves. The truth is OBD datasets (table 5.1) are quite rare and we believe that this is because this kind of data is valuable for car companies and they are hesitant to share it with the public.

8.7.2 Choosing the technique

We initially had several ideas of how to use machine learning in our application.

Driver Classification

In theory, clustering would have enabled us to distinctly separate poor driving from good driving in a way that would have allowed us to directly classify a current state as either good or poor. We experimented with this method and it showed promising results where the model successfully distinguished typical poor driving behaviour such as hard acceleration. We found this to be a promising method and we believe that it could be used as a complement to our final method to further improve its results.

Action prediction

Ideally, we would have been able to use a simulator that could explore and learn what actions to take to minimize the fuel consumption. In theory, this was a good plan, in practice however it required tools that just wasn't available for open source projects. Therefore, this approach was scrapped at an early stage.

Difference classification

This is the option that we implemented. An interesting take on the issue at hand. The main idea behind it is that the LSTM-model will improve long with the driver. The application is implemented so that the driver is pushed to be better than the model and, in theory, this will lead to a driver being pushed towards a minimal fuel consumption.

We consider this to be a solid approach, but it leaves some questions unanswered. What happens when the driver can't get any better? That means that the model also can't get any better, but the driver will still be pushed into becoming more fuel efficient. The result would be that a perfect driver would be classified as bad and one can easily argue that this would cause irritation and a sense of being unappreciated. For now, we lack a solution to this, because how do you know that someone is a perfect driver? We believe that this is impossible to answer with just one user, but it could possibly be solved in a future where many drivers are using the app. Then we could compare them to each other and see where the limit for perfect driving is in practice.

8.8 Future work

As mentioned before, the application is a prototype and not a fully functional application. Here we will list ideas for improvements for a continued future work.

8.8.1 Car simulator

Our main obstacle was to test our application in a stable and reliable environment. Always testing it in a real car is obviously unfeasible in the long run and only running it on data recorded in beforehand doesn't allow you to get a sense of how well it works in real-time. This creates the need of a realistic simulator connected to the application, so the actions taken in the simulator is reflected in real-time in the application.

Although there are a few open-source simulators out there none were ready for us to unpack and go. They all required changes that were out of scope for this project. One can also argue as to how reliable the result would be if it was based on simulator data. The fuel consumption and other related vehicle data would then be known mathematical expressions which would be realistic enough for the simulator, but we argue that it would have simplified our model too such an extent that the result wouldn't be trustworthy.

8.8.2 Application design and functionality

The main improvement is to make the application fully functional. Another improvement is to have the application implemented in the infotainment system of the car instead of having it running on a separate tablet. It should also be possible to have the application on different devices. For example, the Score layout is probably something the driver is more interested in having a mobile phone and by that see the score when not driving. By having the application in the infotainment system, the application should of course be able to draw conclusions about driving in real time. The driver should also be able to compete against other drivers and be able to be the most environmentally friendly driver in the city.

Other improvements mentioned by the participants during the final test can be found in chapter 7 under section 7.2.2: "Improvements mentioned by the participants during the interview".

8.8.3 The model

For the model we are currently using a 5 second sample-time. This number was loosely based upon the reaction time for an average driver which is 2.3 seconds [46]. In [46], the police radar information centre tells us that this number varies a lot depending on the person and situation. It seems reasonable to question the appropriateness of a fixed sample-time for such a variety of situations that may occur during driving.

The standard deviation used for normalizing the difference between the expected and the actual fuel consumption can be seen as a sensitivity setting. It is set to a fixed value that comes from the standard deviation in fuel consumption for the entire training set. We're again faced with the fact that there is such variety in driving and the sensitivity should be based on the situation rather than a fixed value. Again, this requires research that is out of scope for this project.

When should the model be re-trained? This is a good question because progress is also something that varies greatly from person to person. However, unlike the previous, we believe that a fixed interval would be sufficient in this case. Obviously, this interval would have to be picked with some care but since this isn't directly affecting the driving the negative effects of a fixed re-training rate would be negligible.

How should the model deal with multiple drivers using the same car? For simplicity we ignored this issue. In [3] they showed that it is possible to detect different drivers using the same dataset that we are using. We can therefore conclude that the same kind of feature would greatly enhance our product in making it more personal even when a car is shared between multiple drivers.

CHAPTER 9

Conclusion

In this chapter we will share our conclusions and the insights we gained during the project.

9.1 Reflection on goals

Goal: Explore how to design an application that uses gamification to encourage drivers to use more environmentally friendly driving behaviours.

> This goal was achieved. Gamification in a car should be designed to be entertaining and in a way, that helps the driver to improve its driving. It is also important to not make the game too much like a game, because then the driver will focus more on the game than on the road. Which could lead to horrible accidents.

Goal: Explore how to design positive and negative feedback about the driving behaviours both given during a drive and after a drive.

> This goal was achieved. During driving it is preferred that the feedback can be seen and understood in the periphery. One way of achieving this is to use a clear colour scheme, like traffic-light colours (green, yellow and red). From our tests it was clear that the feedback should be both positive and negative and that the visual feedback was more important than audio feedback. If using number feedback during driving it is important that there is a clear and useful unit beside the number. There should be no doubt to what it means, and it should be easy to compare the number with numbers from previous drives.
>
> It is good to design the feedback after completing a drive in a way so that it resembles other improvement applications, for example a training application. This allows the driver to take advantage of past experiences

and therefore, understand the feedback better. It is good to let the driver get feedback on a map, because it helps the driver to see exactly where to change their behaviour. It is also good to get feedback graphically to see its progress during selected dates. For all feedback it is important that all data have units and are consistent.

Goal: Explore how to design the communication between the driver and the application, to be comprehensive, non-disturbing, and encouraging.

This goal was achieved. The communication should be designed so that it is possible to communicate without eye-contact. There must not be something flashing or too colourful. The communication should consist of information that is necessary and not too complex. Sound should be a part of the communication, but it should be possible to turn it off.

Goal: Explore how to use a car simulator to generate data.

This goal was achieved but not implemented. A simulator would have been optimal for our project. We reasoned that we should have been able to use one of the various open-source simulators that are available. What we failed to realize was that most of these simulators are for developing autonomous cars and none provided much vehicle data. We could have adopted our model to the available data, but we reasoned that doing so would make our result unreliable for a real-world application.

Goal: Explore cloud computing and its possible uses in an in-car application.

This goal was achieved. We are using Amazon Web Service to store data and to perform post-drive processing. We have also explored other options such as re-training the model in the cloud although this was never implemented but it is fully possible to do.

Goal: Use machine learning techniques to find a correlation between driving style and fuel consumption.

This goal was achieved. We achieved this since the driving style is reflected in the vehicle data that we are using to predict the expected fuel consumption. The correlation has therefore been found since it's been shown that we are able to predict the fuel consumption with a reasonable accuracy.

9.2 Concluding thoughts

We found that there are more improvements that need to be done before the application is a finished product ready for production. For this project the application ended up as a proof of concept and a set of guidelines to use for further exploration.

References

[1] H. Johansson, J. Färnlund, and C. Engström, "Impact of ecodriving on emissions and fuel consumption," p. 33, 1999. "https://trafikverket.ineko.se/Files/sv-SE/11395/RelatedFiles/1999_165E_impact_of_ecodriving_on_emissions_and_fuel_consumption.pdf".

[2] J. N. Barkenbus, "Eco-driving: An overlooked climate change initiative," *Energy Policy*, vol. 38, no. 2, pp. 762 – 769, 2010.

[3] P. Di Lena, S. Mirri, C. Prandi, P. Salomoni, and G. Delnevo, "In-vehicle human machine interface: An approach to enhance eco-driving behaviors," in *Proceedings of the 2017 ACM Workshop on Interacting with Smart Objects*, SmartObject '17, (New York, NY, USA), pp. 7–12, ACM, 2017.

[4] B. I. Kwak, J. Woo, and H. K. Kim, "Know your master: Driver profiling-based anti-theft method," *CoRR*, vol. abs/1704.05223, 2017.

[5] "Join our driving avatar program." https://www.drivingavatar.com/. (accessed: 2018-05-01).

[6] J. D. Gould and C. Lewis, "Designing for usability, key principles and what designers think," *Research Contributions*, no. 3, pp. 300–311, 1985.

[7] S. Gibbons, "Design thinking 101." https://www.nngroup.com/articles/design-thinking/, 2016. (accessed: 2018-04-15).

[8] "Ergonomics of human–system interaction — Part 210: Human-centred design for interactive systems ," standard, International Organization for Standardization, Geneva, CH, Mar. 2010.

[9] D. A.Norman, *The design of everyday things.* 2002.

[10] J. P. Yvonne Rogers, Helen Sharp, *Interaction Design, Beyond human-computer interaction 3th Edition.* Human-computer interaction, Wiley, 2013.

[11] "The ultimate guide difference between usability and user experience." https://www.mockplus.com/blog/post/difference-between-usability-and-user-experience. (accessed: 2018-02-16).

[12] M. Soegaard, “Usability: A part of the user experience.” https://www.interaction-design.org/literature/article/usability-a-part-of-the-user-experience. (accessed: 2018-02-17).

[13] “Guidance on usability.” https://www.userfocus.co.uk/resources/iso9241/part11.html. (accessed: 2018-03-01).

[14] J. Nielsen, “Tag cloud examples.” https://www.nngroup.com/articles/tag-cloud-examples/, 2009. (accessed: 2018-05-20).

[15] D. Travis, “Measuring satisfaction: Beyond the usability questionnaire.” https://www.userfocus.co.uk/articles/satisfaction.html. (accessed: 2018-04-01).

[16] A. Birkett, “8 ways to measure satisfaction (and improve ux).” https://conversionxl.com/blog/8-ways-to-measure-ux-satisfaction/. (accessed: 2018-04-01).

[17] J. Sauro, “Measuring usability with the system usability scale (sus).” https://www.userfocus.co.uk/articles/measuring-usability-with-the-SUS.html. (accessed: 2018-04-10).

[18] N. Babich, “Prototyping 101: The difference between low-fidelity and high-fidelity prototypes and when to use each.” https://theblog.adobe.com/prototyping-difference-low-fidelity-high-fidelity-prototypes-use/, 2017.

[19] “Donald Norman’s Three Levels of Design.” https://www.interaction-design.org/literature/article/donald-norman-s-three-levels-of-design. (accessed: 2018-03-10).

[20] M. Sailer, J. U. Hense, S. K. Mayr, and H. Mandl, “How gamification motivates: An experimental study of the effects of specific game design elements on psychological need satisfaction,” *Computers in Human Behavior*, vol. 69, pp. 371 – 380, 2017.

[21] R. Zepeda, “Why gamification matters to the auto industry.” https://www.linkedin.com/pulse/why-gamification-matters-auto-industry-robert-zepeda.

[22] “Next-gen dashboards teach leadfoots how to hypermile.” https://www.wired.com/2008/12/when-ford-and-h/. (accessed: 2018-02-20).

[23] “Tips for fuel-efficient driving.” http://www.ecodriver.org/pages/Fuel-EfficientDriving.php. (accessed: 2018-03-05).

[24] S. Korolev, “Neural network to play a snake game.” https://towardsdatascience.com/today-im-going-to-talk-about-a-small-practical-example-of-using-neural-networks-training-one-to-6b2cbd6efdb3.

[25] I. Hussain, “Clustering in machine learning.” http://www.cad.zju.edu.cn/home/zhx/csmath/lib/exe/fetch.php?media =2011:presentation_ml_by_ibrar.pdf.

[26] C. Olah, “Understanding lstm networks.” http://colah.github.io/posts/2015-08-Understanding-LSTMs/.

[27] J. Brownlee, “How to scale data for long short-term memory networks in python.” https://machinelearningmastery.com/how-to-scale-data-for-long-short-term-memory-networks-in-python/.

[28] “Normal distribution.” http://www.encyclopediaofmath.org/index.php? title=Normal_distribution&oldid=33876. (accessed: 2018-02-01).

[29] “Meet android studio.” https://developer.android.com/studio/intro/index.html. (accessed: 2018-02-02).

[30] “Github is how people build software.” https://en.wikipedia.org/wiki/GitHub. (accessed: 2018-02-02).

[31] “Macbook pro specifications.” https://www.apple.com/se/macbook-pro/specs/. (accessed: 2018-04-20).

[32] Californa Air Resources Board, *On-Board Diagnostic II (OBD II) Systems - Fact Sheet / FAQs*, Oct. 2015.

[33] “Low-fi prototyping: What, why and how?.” https://mobgen.com/low-fi-prototyping/, 2016. (accessed: 2018-05-03).

[34] B. I. Kwak, J. Woo, and H. K. Kim, “Know your master: Driver profiling-based anti-theft method.” https://arxiv.org/pdf/1704.05223.pdf, 2016.

[35] H. Abdi, “The kendall rank correlation coefficient.” https://www.utdallas.edu/ herve/Abdi-KendallCorrelation2007-pretty.pdf.

[36] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, “Scikit-learn: Machine learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[37] “Keras: The python deep learning library.” https://keras.io, 2015.

[38] Y. Tsafir, “Deploying a tensorflow model to android.” https://medium.com/joytunes/deploying-a-tensorflow-model-to-android-69d04d1b0cba.

[39] “Creating a navigation drawer.” https://developer.android.com/training/implementing-navigation/nav-drawer.html. (accessed: 2018-02-16).

[40] “Fragments.” https://developer.android.com/guide/components/fragments.html. (accessed: 2018-01-27).

[41] anastr, “Speedview.” https://github.com/anastr/SpeedView. (accessed: 2018-03-01).

[42] “Scrollview.” https://developer.android.com/reference/android/widget/ScrollView. (accessed: 2018-02-04).

[43] “Color tool.” https://material.io/design/color/color-usage.html#hierarchy. (accessed: 2018-02-20).

[44] Keras, “Keras, dense layer.” https://keras.io/layers/core/#dense.

[45] “Android auto.” https://designguidelines.withgoogle.com/android-auto/android-auto/introduction.html. (accessed: 2018-04-10).

[46] “Braking factors.” https://copradar.com/redlight/factors/index.html. (accessed: 2018-03-01).

APPENDIX A

Brainstorming

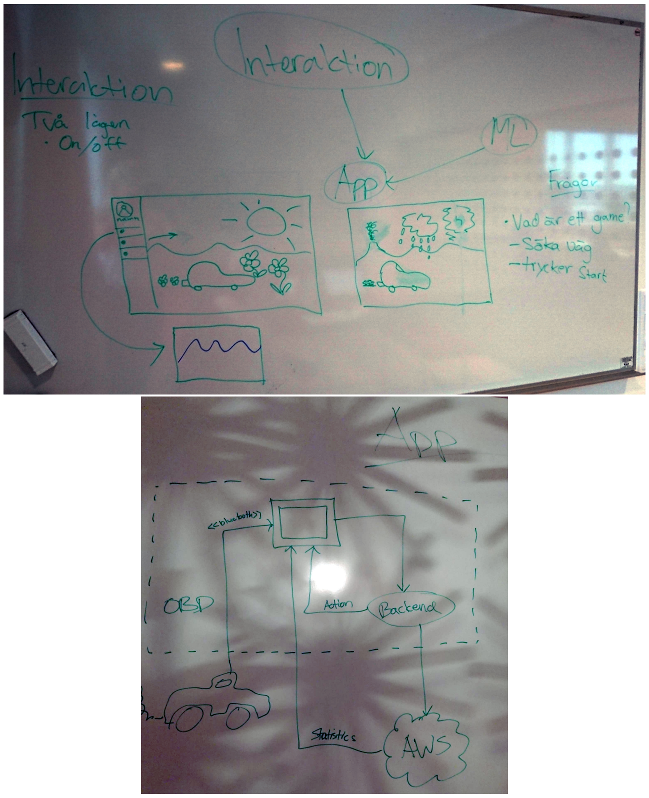


Figure A.1: Brainstorming activity's

APPENDIX B

Test material

Final test plan

Purpose, goals, and objectives of the test:

This is a final test of an eco-driving application developed as a part of a master thesis on Lund University.

This test is going to test the whole application with the focus on the layout shown when driving and the layout after driving. When testing Drive the purpose is to see that the driver understands the feedback from the application and doesn't get unnecessarily disturbed. When testing Score the purpose is to test if the driving result and the navigation opportunities of the application are understandable.

Research questions

Drive layout:

- How well do users understand the feedback from the smiley?
- How well do users understand the score to the right of the smiley?
- Do the users get unnecessarily disturbed?
- How does the users feel when getting the feedback?

Score layout:

- How well do users understand the Map, progress and score layout?
- How easily and successfully do users find the tools or options they want?

Participant characteristics

Characteristics	**Number of participants**
Participant characteristics: Pilot Regular Backup **Total number:**	

Age 21–30 31–40 41–50 51–60	
Gender Female Male	
Driving experience: Driving mostly the same routes (to and from work, to and from grocery store, to and from the gym, etc) Own a car Driving with respect to the environment (eco-driving)	
Driving frequency: Infrequently: 1 drive per month or less Moderately often: 6 - 12 drives per month Very often: 5 or more drives per week	
Driving simulator experience: Have used a driving simulator	

Method (test design)

1. **Introduction (2 minutes)**
   - Welcome the participant
   - Short presentation about the test and the project.

- Discuss:
- Moderators role.
- Observers role - Recording?
- Thinking aloud
- Informed consent
- The participant read and sign an informed consent

2. Pre-questions (2 minutes)

3. Tasks (14 minutes):

  a. Test moderator give the participant tasks during test.
  b. Observers observe the participant and writing down the result and thoughts.

4. Post-questions and debriefing (12 minutes)

  c. Post-questions
  d. Interview
  e. Follow up on problems

5. Test moderator thanks the participant for taking part of the test.

Total time: 30 minutes

Task list

When starting the test, the application should be started and the participant should sit in the car simulator. The car game linked to the car simulator should be started and the eco-driving application should be placed to the right side of the car simulator.

1. Start Eco-driving

During driving:

2. Drive (2 minutes)

After Driving:

After driving a new view with the eco-score and a button that leads to the score view is showing.

3. Reflect on “Eco-score”
4. Go to Score view
5. Show the driving from 1 April to 18 April.
6. Describe the meaning of the lines between markers on the map.
7. Get information about the marker (clue: click on the markers)
8. Go to Score and see the Eco-score. Change dates from 1 March to 2 June.
9. Go to Progress and explain the progress graph and the improvement score
10. Change date from 4 April to 9 May.
11. Go to Drive again (clue: navigation drawer).

Task number:	Description	Required to perform	Success criteria	Maximum time
1	Start Eco-driving	Click on the button named Start Eco-driving	Click on Start Eco-driving button	2 seconds
2	Drive	Drive (car game) and get feedback from the application. React when getting the feedback.	Drive without getting disturbed of the application. Understand the feedback and reflect on it during driving.	3 minutes
3	Reflect on “Eco-score”	Thinking aloud and describe its meaning	Describe correctly	1 min
4	Show score view	Click on the Score button		5 second
5	Show the driving from 1 April to 18 April	Choose from 1 April to 18 April in the calendar placed in the header.	Choose from 1 April to 18 April in the calendar placed in the header.	1 min
6	Describe the meaning of the lines between markers.	Look at the map and describe	The meaning of the colour should be a part of the description.	1 min
7	Get information about the marker.	Click on the markers	Click on the markers	1 min

8	Go to Score and see the Eco-score. Change dates from 1 March to 2 June.	Change view and choose another time slot	Click on Score and reflect on the result. Change dates in the calendar view placed in the header.	1 min
9	Go to Progress and explain the progress graph and the improvement score.	Change view and explain the graph and the improvement score	Explain correctly	2 min
10	Change date from 4 April to 9 May			1 min
11	Go to Drive again	Use the navigation drawer	Click on the symbols in the left corner or swipe from left to right to get the navigation drawer and then choose drive.	2 min

Test environment and equipment

During the test the participant are going to drive in a car simulator. The test is therefore going to be held in the Jayway office in Malmö were the car simulator is placed.

Equipment:
- Pixel C were the application is implemented
- Car simulator rig
- Computer running simulator game.

Environment:
- Quiet and closed room.

Test moderator role

The test moderator's role is to welcome and support the participant during the test. The moderator is also responsible to answer some questions during the test. During the test the moderator give the participant tasks and after the test it is the moderator that is interviewing the participant.

Data to be collected and evaluation measures

Objective/Quantitative data:

Calculate:

- Number of successful tasks
- Approximate time per task
- Number of errors
- Number of Intervention

Objective/Qualitative data:

Describe the observations:

- Amount of errors. What was going wrong?
- What interventions was given? Did the intervention help?

Subjective/Quantitative data:

After test the participant answered questions in an interview. Below is a list of questions that was asked after the test.

- How was the general experience after the test?
- Were you stressed and/or distracted by the application while driving?
- How was the navigation of the application?
- Was anything unclear?
- What did you think about the smiley? Did you understand the feedback? What did you think about the colours?
- Were you motivated by seeing your fuel consumption score while driving?
- Did you understand the feedback after driving?
- How was it to get feedback on a map?
- What was your perception of progress and improvement. Did you understand its meaning?
- How was the design of the app?
- Did you miss any feedback?
- Do you want audio feedback?
- Other improvements?

Subjective/Qualitative data:

After test the participant answered three different questionnaires (NASA TLX rating worksheet, SUS and Word-cloud), this because we wanted to get a large result set and we did not know which questionnaire would give the most credible results. As Jeff Sauro put it, “there isn’t a usability thermometer to tell you how usable your software or website is.” Both NASA TLX and SUS are satisfaction metrics, so they don’t count completion rates, errors, task time, and other important rates. But they could give us a good indicator of the applications satisfaction.

Pre-questionnaire	
Characteristics	**Number of participants**
Participant characteristics: Pilot Regular **Total number:**	
Age 21–30 31–40 41–50 51–60 61-70	
Gender Female Male	
Any experience of Eco-driving assistant? Yes No	
Driving experience - Driving mostly the same routes (to and from work, to and from grocery store, to and from the gym, etc) - Own a car - Driving with respect to the environment (eco-driving)	
Driving frequency -Infrequently: 1 drive per month or less -Moderately often: 6 - 12 drives per month -Very often: 5 or more drives per week	
Driving simulator experience Yes No	
Android experience Likert: Agree 1 2 3 4 5 Disagree	

Table B.1: Pre-questionnaire for final test

Wordchoice

Step 1: Read over the following list of words. Considering the product you have just used, tick those words that best describe your experience with it. You can choose as many words as you wish.

❑ Vague

❑ Accessible

❑ Advanced

❑ Ambiguous

❑ Annoying

❑ Appealing

❑ Approachable

❑ Attractive

❑ Awkward

❑ Boring

❑ Bright

❑ Business-like

❑ Busy

❑ Clean

❑ Clear

❑ Cluttered

❑ Compelling

❑ Complex

❑ Comprehensive

❑ Confusing

❑ Consistent

❑ Contradictory

❑ Controllable

❑ Convenient

❑ Counter-intuitive

❑ Creative

❑ Credible

❑ Cutting edge

❑ Dated

❑ Desirable

❑ Difficult

❑ Distracting

❑ Dull

❑ Easy to use

❑ Effective

❑ Efficient

❑ Effortless

❑ Empowering

❑ Energetic

❑ Engaging

❑ Entertaining

❑ Exciting

❑ Expected

❑ Familiar

❑ Fast

❑ Faulty

❑ Flexible

❑ Fresh

❑ Friendly

❑ Frustrating

❑ Fun

❑ Hard to Use

❑ High quality

❑ Illogical

❑ Impressive

❑ Inadequate

❑ Incomprehensible

❑ Inconsistent

❑ Ineffective

❑ Innovative

❑ Insecure

❑ Intimidating

❑ Intuitive

❑ Irrelevant

❑ Meaningful

❑ Misleading

❑ Motivating

❑ New

❑ Non-standard

❑ Obscure

❑ Old

❑ Ordinary

❑ Organised

❑ Overwhelming

❑ Patronising

❑ Poor quality

❑ Powerful

❑ Predictable

❑ Professional

❑ Relevant

❑ Reliable

❑ Responsive

❑ Rigid

❑ Satisfying

❑ Secure

❑ Simple

❑ Simplistic

❑ Slow

❑ Sophisticated

❑ Stable

❑ Stimulating

❑ Straightforward

❑ Stressful

❑ System-oriented

❑ Time-consuming

❑ Time-saving

❑ Too technical

❑ Trustworthy

❑ Unattractive

❑ Unconventional

❑ Understandable

❑ Unpredictable

❑ Unrefined

❑ Usable

❑ Useful

Step 2: Now look at the words you have ticked. Circle five of these words that you think are most descriptive of the product.

INFORMED CONSENT

Please read this consent carefully before you decide to participate in this study.

Background and purpose. This is a final test of an Eco driving application which is a part of a Master Thesis at Faculty of engineering, Lund University. During the test the participant will test the main parts of the application for the purpose of investigating the feedback from the application. The test will also evaluate the navigation and the general experience of the application.

How will the study be conducted? Both before test, the participant will answer a questionnaire about the participants previous experiences. Followed by the questionnaire the test begins and during the test the participant will be driving a car in a car simulator. After the test the participant will answer another questionnaire about the test. The duration of the study in whole will be approximately 30 minutes.

What are the possible risks and benefits? The experiment is completely safe and does not cause any risk of injury. The participant will get a chance to test a car simulator as well as take part the development of a Eco driving assistant.

Data management and privacy. Your answers and results will be stored under an anonymous code that will only be available to the research team. The results of the study are presented in groups, no individual patterns will be described or identifiable. No information associated with personal information will be saved.

The head of the current study at Lund University is Amanda Eliasson, 0703-415262, amanda.eliasson@gmail.com

Voluntarism. Your participation in the project is entirely voluntary. You can skip the tasks you do not want to complete in the study. You can refuse your participation or at any time cancel your participation without giving any reasons and without this having any negative consequences for you.

Consent (Keep this copy)

I have read the form and have had the opportunity to ask questions and receive them answered as well as get information about the content and purpose of the study. I agree to the participate in the study.

Signature Date

________________________________ ________________________________

Name

____

Consent (Submit to the test supervisor)

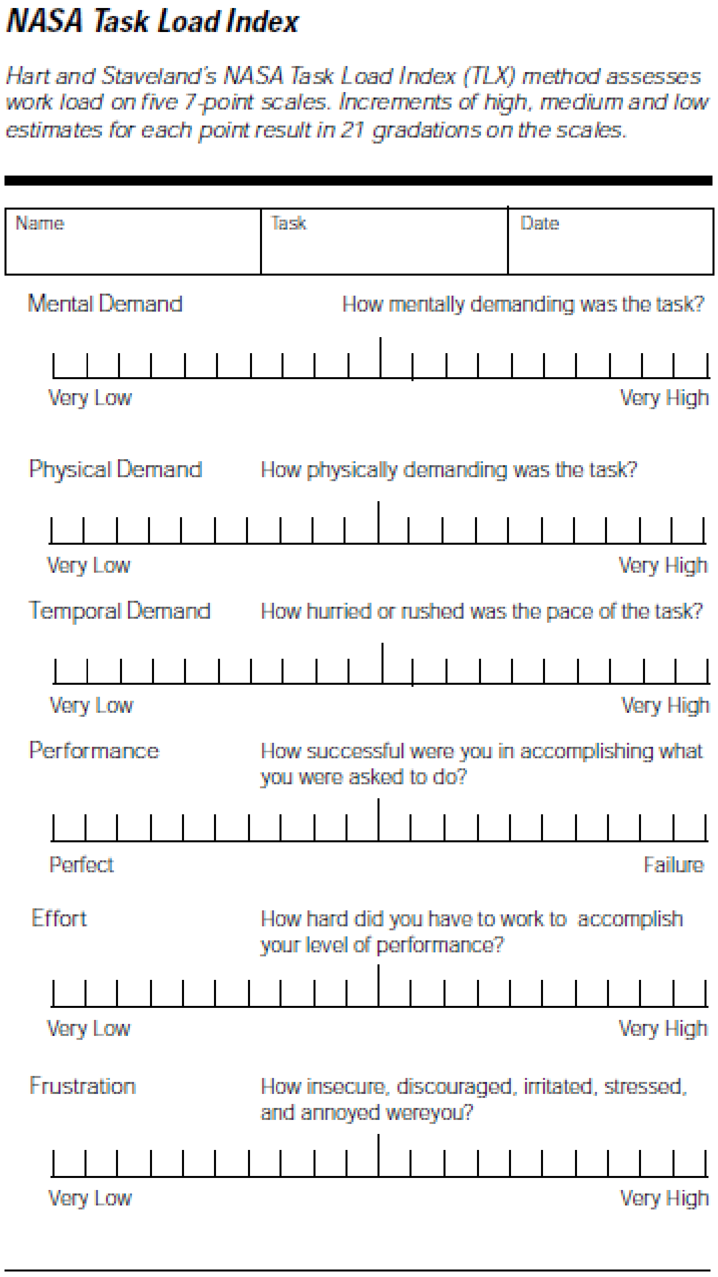
I have read the form and have had the opportunity to ask questions and receive them answered as well as get information about the content and purpose of the study. I agree to the participate in the study.

Signature　　　　　　　　　　　　　　Date

_______________________________　　_______________________________

Name

____



NASA Task Load Index

Hart and Staveland's NASA Task Load Index (TLX) method assesses work load on five 7-point scales. Increments of high, medium and low estimates for each point result in 21 gradations on the scales.

Name	Task	Date

Mental Demand — How mentally demanding was the task?

Very Low — Very High

Physical Demand — How physically demanding was the task?

Very Low — Very High

Temporal Demand — How hurried or rushed was the pace of the task?

Very Low — Very High

Performance — How successful were you in accomplishing what you were asked to do?

Perfect — Failure

Effort — How hard did you have to work to accomplish your level of performance?

Very Low — Very High

Frustration — How insecure, discouraged, irritated, stressed, and annoyed wereyou?

Very Low — Very High

Figure B.1: NASA TLX Rating sheet

APPENDIX C

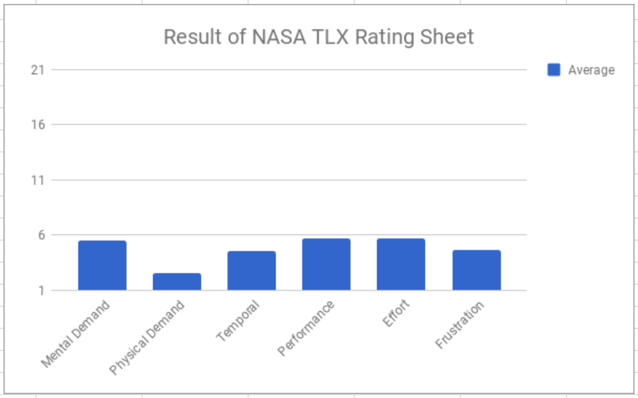
Test calculation

C.1 SUS calculation

		1.	2.	3.	4.	5.	6.	7.	8.	9.	10.	SUS Raw Score	SUS Final Score	Statements
	P	2	1	3	2	1	2	3	1	2	1	24	60	I think that I would like to use this system frequently.
	P	4	2	5	1	4	1	5	1	5	1	37	92,5	I found the system unnecessarily complex.
														I thought the system was easy to use.
	R	4	1	5	1	4	1	5	1	4	1	37	92,5	I think that I would need the support of a technical person to be able to use this system.
	R	5	1	4	1	4	1	4	1	4	1	36	90	I found the various functions in this system were well integrated.
	R	1	1	5	1	4	2	5	2	5	1	33	82,5	I thought there was too much inconsistency in this system.
	R	3	1	5	1	4	1	5	1	4	2	35	87,5	I would imagine that most people would learn to use this system very quickly.
	R	4	1	5	1	5	1	4	3	4	1	35	87,5	I found the system very cumbersome to use.
	R	4	1	5	1	4	1	5	1	4	1	37	92,5	I felt very confident using the system.
	R	4	2	3	1	4	2	4	2	4	2	30	75	I needed to learn a lot of things before I could get going with this system.
	R	4	1	5	1	4	1	5	1	5	1	38	95	
	R	3	2	4	1	4	4	3	2	3	3	25	62,5	P = Pilot test (not inclouded)
	R	4	1	4	1	2	1	5	1	5	1	35	87,5	R = Regular test
	R	4	1	5	1	4	1	4	1	5	1	37	92,5	
	R	5	2	4	1	4	1	4	2	4	1	34	85	
	R	5	1	5	1	4	1	4	1	5	1	38	95	
	R	4	2	4	1	3	3	5	1	5	2	32	80	
	R	4	1	4	1	4	2	5	1	4	1	35	87,5	
	R	5	1	5	1	4	1	5	1	4	1	38	95	
	R	4	1	4	1	2	2	4	2	4	1	31	77,5	
	R	4	1	5	1	4	1	5	1	4	1	37	92,5	
	R	4	2	5	1	4	1	5	1	4	1	36	90	
Average												**35**	**87**	

C.2 TLX calculation

	Mental Demand	Physical Demand	Temporal Demand	Preformance	Effort	Frustration
	7	7	5	5	11	2
	3	4	3	6	3	3
	9	2	2	6	6	2
	4	2	6	3	4	9
	7	2	4	7	3	4
	4	2	3	4	4	4
	7	1	2	9	4	3
	5	2	8	2	8	4
	7	3	11	6	9	5
	4	2	3	3	2	4
	6	3	5	7	7	5
	6	2	4	2	2	7
	5	1	2	9	12	5
	4	2	5	7	8	10
	8	2	8	7	4	2
	5	5	5	4	5	4
	2	2	2	6	4	3
	8	2	2	8	9	9
	3	3	7	7	3	3
Total	**104**	**49**	**87**	**108**	**108**	**88**
Medel	5	3	5	6	6	5



Mental Demand	Physical Demand	Temporal Demand	Preformance	Effort	Frustration
5,473684211	2,578947368	4,578947368	5,684210526	5,6842	4,631578947

Type	Average
Mental Demand	5,473684211
Physical Deman	2,578947368
Temporal Demai	4,578947368
Performance	5,684210526
Effort	5,684210526
Frustration	4,631578947