TALLINN UNIVERSITY OF TECHNOLOGY
School of Information Technologies

Lizaveta Lasitskaya – 153236 IAPM

The efficient way of building an MVP with user story-mapping technique. Case study of Impuls sports club mobile application

Master's thesis

Supervisor: Jekaterina Tšukrejeva
MSc

Tallinn 2022

TALLINNA TEHNIKAÜLIKOOL

Infotehnoloogia teaduskond

Lizaveta Lasitskaya – 153236 IAPM

Efektiivne viis MVP loomiseks kasutaja lugude kaardistamise tehnikat kasutades spordiklubi "Impuls" mobiilirakenduse uuringu näitel

Magistritöö

Juhendaja: Jekaterina Tšukrejeva
MSc

Tallinn 2022

Author’s declaration of originality

I hereby certify that I am the sole author of this thesis. All the used materials, references to the literature and the work of others have been referred to. This thesis has not been presented for examination anywhere else.

Author: Lizaveta Lasitskaya

Date: 10.05.2022

Abstract

Nowadays, a very large number of startups and projects fail due to an incorrectly chosen approach to implementing their ideas. Choosing the right method to achieve an appropriate result plays an important role. The purpose of this research is to find the right approach, and how to achieve a good MVP (minimal viable product) with minimal cost, best profit, and a high customer satisfaction level. The work in this research paper is based on the case study method where the following methods were analyzed: requirements - and improved version of it - PRDs (product requirement document) and user stories together with user story mapping technique. The advantages and disadvantages of each were identified, the results were compared based on the initial setup goals and an assessment of each approach was made. The conclusions were drawn precisely on the implementation of an MVP for the Impuls sports club mobile application. Also, based on the obtained results, ideas were proposed on how each method can be adjusted and improved. The study did not include actual software development. For the study, a clickable user-friendly prototype was created, which was tested on 15 different users. As a result, all planned work with high and medium priority was completed within 4 iterations. Moreover, each iteration's backlog was adjusted to include tasks for improvements based on the collected feedback from user testing. Work was prioritized accordingly.

Thesis is written in English and contains 67 pages, 4 chapters, 29 figures and 8 tables.

Annotatsioon

Väga suur arv startup-e ebaõnnestub oma idee elluviimisel vale lähenemise tõttu. Eduka tulemuse saavutamiseks mängib õige meetodi valik olulist rolli. Antud uuringu eesmärk oli leida parim lähenemisviis kasumliku MVP arendamisel minimaalste kuludega ja kõrge kliendirahuolu tasemel. See töö põhineb juhtumiuuringu meetodil, kus analüüsiti järgmisi meetodeid: nõuded - ja selle täiustatud versioon - PRD-d (tootenõuete dokument) ja kasutajalood koos kasutajalugude kaardistamise tehnikaga. Tuvastati iga meetodi puudused ja eelised. Tulemusi võrreldi esialgsete eesmärkide alusel ja iga lähenemisviis sai oma hinnangu. Kõik järeldused tuginevad spordiklubi “Impuls" mobiilrakenduse MVP-l. Tulemuste põhjal pakuti väljavälja ideid, kuidas iga meetodit saaks kohandada ja täiustada. Uuring ei sisalda tegelikku tarkvaraarendust. Uuringu jaoks loodi interaktiivset ja kasutajasõbraliku prototüüpi, mida katsetas 15 erinevat kasutajat. Kogu plaanitud töö valmis kõrge ja keskmise prioriteediga 4 iteratsioonis. Lisaks kohandati iga iteratsiooni backlog-i, et see sisaldaks parandusi, mida kasutajad katsetamisel välja tõid. Töö täidetati vastavalt prioriteedile.

Lõputöö on kirjutatud inglise keeles ning sisaldab teksti 67 leheküljel, 4 peatükki, 29 joonist, 8 tabelit.

List of abbreviations and terms

SD	Software development
MVP	Minimum viable product
PRD	Product Requirement Document
PM	Product Manager
UI	User Interface
UX	User Experience

Table of contents

List of figures

List of tables

1. Introduction

Nowadays there is a huge amount of bankrupt companies, failed projects, and even word catastrophes due to poor product management approaches.
CB Insight's did a study related to this topic. They analyzed 101 failed startups and found that 9 out of 10 startups fail within 1-3 years of launch [1]. A similar analysis was made by Bloomberg and their results showed that 8 out of 10 startups fail within 1.5 years [1]. Studies of two different companies show similar results. CB Insight identified 20 main factors that led to the failure, figure 1 provides the chart that illustrates main reasons of failures:

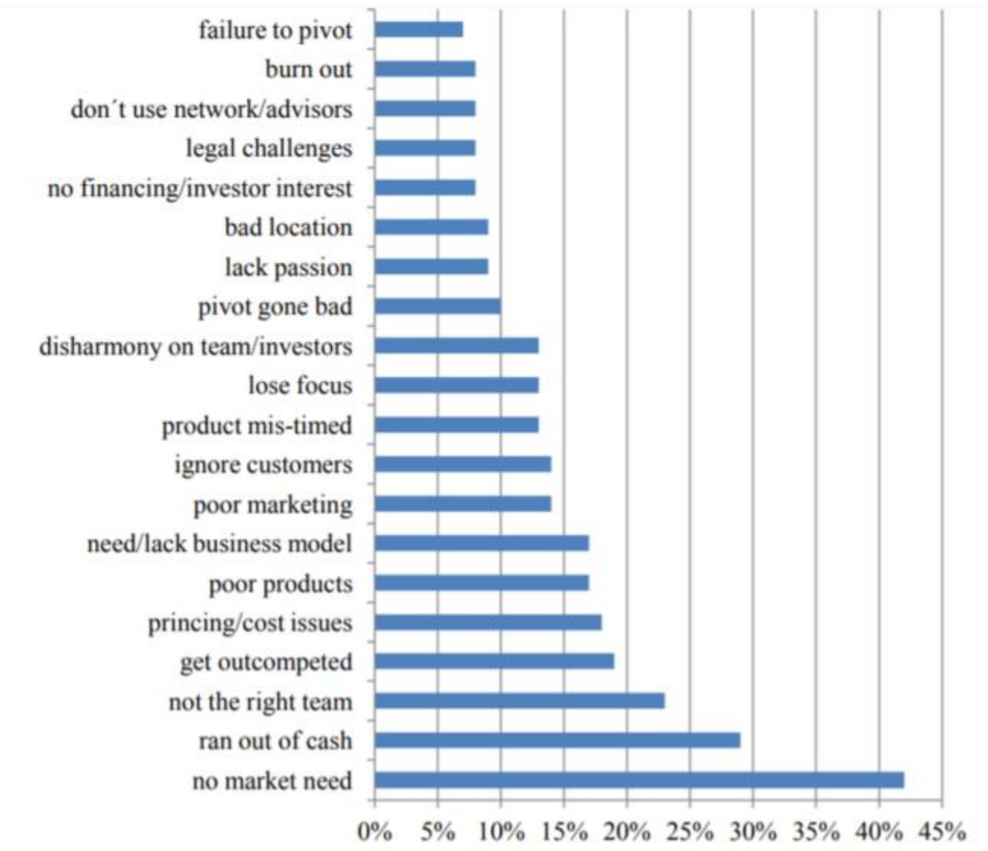


Figure 1: Top 20 reasons for startups failure [1]

Analyzing the graph, it is clearly seen that the most common reason for the failure of the project is “No market need” which is more than 40%, all other factors are in range of 10 - 30%. All these reasons appear due to poor product manager strategy, wrongly chosen methodology, which can negatively affect the project, ignoring the needs of product users, ignoring market analysis, and also the wrong prioritization and lack of a common understanding among the team and people involved in the project.
As an example, can be considered such projects as Google Glasses or Apple Newtone. These projects failed because market analyses were not done, customer needs where ignored and wrong implementation strategy was selected.
It is very important to choose the correct way to build a product, to avoid such situations as extra costs for a company, poor product quality which can lead to irreversible consequences, very low profit, and unhappy customers.
Shared understanding is the key to success. Shared understanding is when all sides (end-users, partners, product managers, sales team, engineering, and all other involved parties) totally understand the broader picture and have the same interests and goals. Shared understanding keeps it focused on users and their experience, and the result is better communication, planning, and a better product in the end.
How to build this shared understanding between all parties? Why does the list of predefined requirements not work and why user story-mapping is helpful in the product management stage? What if it is impossible to avoid pre-defined requirements? How this technique can be improved?

1.1. Research Problem and Objectives

The main goal of this research is to find the right approach to create an MVP (minimal viable product) with the lowest costs, high customer satisfaction level, and appropriate quality.
To achieve this goal, two methods were selected and compared in this research:
1) building a product based on predefined requirements
2) building a product following the user story mapping technique, and continuous user interactions.
Advantages and disadvantages will be identified, the strengths and weaknesses will be carefully studied, which will allow to conclude which of these two methods is best

suited to a specific product. It is very important to understand the set of specific characteristics that are suitable for the creation of projects of different directions in order to implement the project with maximum value and lower costs as possible.
The main focus of this study is to analyze and understand two product implementation strategies - user story-mapping and as alternative solution - a predefined requirements approach.
The main problem is to understand how these approaches work, how to use it by making the product development process more efficient and customers - happier. All practices will be applied on a real case - the mobile application for the Estonian sport club: Impuls. At the end of each iteration, the achieved result will be analyzed and appropriate decisions on how to improve the next cycle will be made.

The research will cover the following questions:

- What are user story and user story mapping?
- What privileges do user stories have over predefined requirements?
- What problems can appear using the requirements approach and are there any strategies to improve it? - analyzes of PRD approach
- How to build a user story map and correctly apply it?
- How to deliver a successful MVP using user story maps and PRDs?
- What are the main challenges of user story mapping and PRD approach?

1.2. Research Scope and Limitations

The research focuses on the planning and grooming phases of the product - gathering ideas and requirements, scoping, planning, and visualisation of the features, planning the development process. The study does not consider real software development.
As a result of the research will be the user-friendly, clickable prototype which will help to make an examination of the end result and make a conclusion based on it.

1.3. Research Methodology

Work in this research paper is based on the case study research method. The method includes the following stages: formulating the research problem and objectives, background and general information overview, analyses, applying selected methods for building the case, evaluation, and estimation of achieved results.

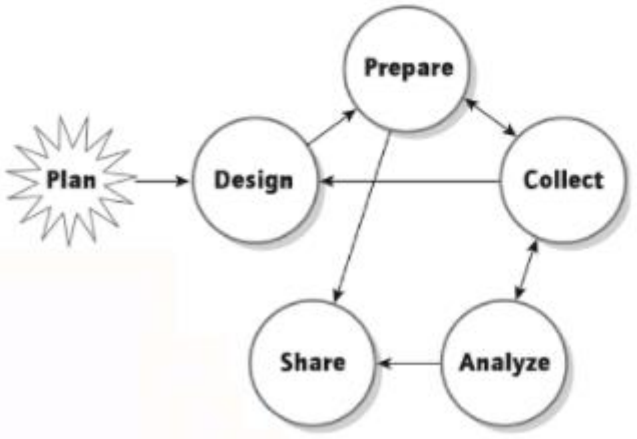


Figure 2: Case Study Process [2]

The figure 2 presented above represents 6 stages of the case study research method [2]. The First phase is Plan. The main goals of the Plan stage are formulating the main research problem and questions, identifying the limitations of the scope, selecting the case, and choosing the methods that will be analyzed and applied to the selected scenario.

The design stage is used for identifying the work process and planning the workflow.

In the prepare step the target audience was found, surveys were prepared, theoretical and background analyzes of selected methods were done.

The collect step includes receiving feedback from completed work; this feedback comes from practical exercises such as communication with users, testing, personal observations, and analyses based on the literature and other sources. To collect the necessary information all methods were applied to the real case - the implementation of a mobile application prototype.

All collected information is analyzed, outcomes and conclusions are made, based on the collected feedback - learnings were made and necessary steps were repeated in the loop. After that, the final result outcome and conclusion are shared.

1.4. Thesis Structure

This research work is divided into 5 main parts.

- **The introduction** chapter gives a brief overview of the problem and also contains the summary of the chosen research methods.
- **Background and main concepts** chapter provides theoretical information and analyzes on chosen techniques and approaches.
- **The case study** part concentrates on the implementation part, gathering data for research by applying techniques and methodologies which were introduced in the previous chapter.
- **Results and analysis** contain work on the analysis of results and their assessment. The final assessment of the result is made, conclusions are drawn.
- **The summary** part is a short overview of the achieved goals and results.

2. Background and Main concepts

There are different methodologies in the software development world that describe the way how the product can be built. The software development methodology is an instrumental kit that provides a set of necessary tools, guides, and different approaches that help engineering and product teams to build software in the most efficient way. These methods are addressed to improve all parts of the SD process: project management, software design, architecture, process improvement, implementation (software engineering). The main idea of all these models is to achieve the most successful and profitable result. Those methods have their own ideas and best practices on how to gain those goals. Also, each method has its own purpose and should be selected based on the product type, needs, and team. Some types of products require a certain structure and realization plan - in that case, pre-defined requirements cannot be avoided. Other projects can be more flexible in their implementation and then more flexible and Agile approaches can be applied. Based on that, it was decided to analyze two totally different approaches for planning and building the minimum viable product (MVP) - requirements and the "Agile" method - user stories and mappings. Both methods are highly used by different corporate and start-up companies.
Despite the chosen method, the final goal always should be the same - valuable product and customer satisfaction with the lowest implementation cost. How those two methods can be applied and what are the disadvantages and advantages of both? To understand that, below are presented theoretical analyzes of both methods.

2.1. Requirements and PRDs

As it was mentioned above, there are such types of products/projects where it is impossible to avoid written requirements, everything should be documented and approved in advanced. There are projects where the methodology cannot be flexible and it is necessary to prescribe strict requirements, which cannot be changed during the creation. However, there is a risk that different team members and other involved parties may read/understand them differently. As it happened, for example, with “NASA's Mars Climate Orbiter” project, which crashed because the requirements were prescribed by American engineers and had in mind one unit of measurement (used the metric system of millimetres and meters in its calculations), but English engineers, who followed these requirements, used other units of measurement (English system of inches, feet, and pounds) [3]. One of the reasons for the “failure” may be the absence of shared understanding. It is very important that the whole team understands requirements and context in the same way, otherwise, there is a huge risk that the result will be far from expectations.

Building the project by specified requirements does not necessarily mean following the waterfall model and being against the Agile set of principles. Some companies still call themselves an Agile team, follow all good practices of Scrum - such as working in interactions, constant user feedback, retrospective practices, constant improvements, but still use a requirements model to build the product.

Example below provides the main steps of the "requirement approach" that were taken from the example of how "ACME" company accomplishes it. (the company name was modified in this paper - real life example). Below are described the main principles/ideas that follows company "ACME":

- Development/Engineering team works in Agile environment
- The team follows the Scrum framework (2 weeks sprint, scrum meetings).
- The team has a product backlog and owns a sprint backlog.
- The product manager owns ideas on how to build/improve a feature. Market research was made and customers' needs were analysed.
- The product manager writes tickets. The ticket description contains a set of requirements for how the feature should be implemented and how it should behave.
- The product manager identifies how the sprint backlog should be looked like and what functionality should be delivered during the sprint.
- At the planning meeting, the team goes through the prepared backlog, reads requirements, discusses what should be done, and gives estimates (2 hours meeting).
- The product manager reviews estimates and modifies the sprint backlog based on the estimations to match the team's velocity.
- Sprint starts.

Even if the team was following Scrum practices and called themselves as an Agile company, based on the research and surveys that were made, it was identified that a lot of engineering and product teams struggle to deliver features on time, release things with low quality, and face customer complaints. It can be caused by multiple issues:

- The product team does perform a full market analysis, review and analyse customers' needs. However, requirements that make sense in the beginning, might become fully outdated after some time. A lot of features that were built are not in use by customers.
- Communication between the product and engineering team is broken. No shared understanding of what and why it should be built. Engineering team delivers and releases features in the production environment, however, after some time PM notices that it does not do what it is supposed to.
- No shared understanding between developers. Noticeable, when front end developers work together with back-end developers and it is becoming very hard to agree on contracts, combine two parts together.
- Sprints and backlogs are not well planned, the engineering team faces time constraints, pressure, deadlines. No opportunity to test code/solution properly or analyse all possible risks in advance.
- Team is not engaged. They don't see the full picture of what should be built. They don't understand WHY.
- Poor system quality. When developers blindly follow written requirements, there is no opportunity to think how it can be built in a better way. They face technical constraints in the middle of development work, and it is quite hard to take a step back and reconsider the solution.

To conclude, it is very important to build a shared understanding between all parties. Requirements tell us HOW the feature should be built, but in very rare situations explains WHY. However, even "requirement" approach can be used to achieve the goal. PRD - Product Requirement Document can be a powerful addition to this method, an approach that not only illustrates WHAT but also provides more context around the specific requirement. Below are described main points and ideas of this method.

A product requirements document (PRD) is a document that contains a set of requirements for a specific product. However, the approach itself and how it is written differs from the strict requirement list. In addition to general requirements, it also provides the main concepts of why and what should be built. It should give a full picture to engineers and the capability to analyze it and come up with technical solutions for a given problem.

What Should a Product Requirements Document Contain?

- Objectives and Goals

PRD should explain exactly WHY the feature should be built (purpose of the feature), WHAT should be achieved (overview of the feature).

- Feature and it's behaviour

Requirements, use cases. Without any technical details. Engineering team should be able to come up with technical/architecture decisions by themselves.

- UX and UI flows

Design team works together with the product team and provides prototypes for use cases that are described above. That will help engineering teams to visualise the feature.

- Assumptions, Constraints and Dependencies

Product team should specify any assumptions, constraints, and possible dependencies. Also, can specify what is out of scope to deliver the desired MVP for this feature.

As part of this research, multiple product requirements documents were analyzed. As the result, all best practices were combined together, and these main parts of the PRD were identified:

Goals and objectives - General idea and explanation of the feature. Why should it be implemented? What problems will it solve? This section will provide context to the engineering team. Developers will be aware why they are building it and what benefit it will bring to the customers.
Target users - Establishing and description of target users. It is important to understand who the target users will be. Understanding who is going to use the feature will help to come up with a proper solution with specifics that are important for a user or can affect the system usage.
Timeline/release plan - Timeline/plan of releasing the feature, when it should be delivered, how many phases of delivery. It will help the engineering team to plan it properly according to business needs and agreements.
Success Criteria - Definition of Done, what does it mean that the functionality was

successfully delivered? Sometimes there is a misunderstanding between the product and engineering team in terms of what means "done". This section will provide exact product expectations of the successfully delivered feature.

Impacted Systems - What part of the system/product will be affected by this change? Provides confidentiality/context where exactly this feature should be built, what parts of the system will be affected.

In scope - What functionality is in scope? This section provides a full picture and shared understanding of what should be implemented/covered.

Out of scope - What is not going to be built (out of scope?) Identifies what exactly is out of scope. So developers can be sure that they don't need to think/plan this functionality during the implementation work.

Possible Future Expansion - Future plans and initial ideas how this functionality can be expanded. It will help the engineering team to find a proper technical solution that can be easily reused or expanded in the future. As a result, it will be possible to reduce technical constraints for future development work in the early stages.

Requirements - Main use cases and explanation of how the system should behave. It can be built using different approaches, user stories, requirements, and can contain UX/UI prototypes. Engineering team will have a picture of what should be built and how it should behave. They will still have room for thinking of possible technical solutions.

Open questions - Any open questions that the product team might have and require some additional investigations. It minimises back and forth questions between product and developers. "Investigation" work can be done, additional brainstorming sessions can be arranged.

Dependencies - Any known dependencies between system/product parts. It will provide an additional context to the engineering teams. It saves time, flags possible complexity.

Potential risks - Any known potential risks. The engineering team might put some effort into refactoring software parts that are risky in order to decrease the risk.

2.2. User Stories and Story Mapping

A lot of different Agile companies go with user stories and mapping techniques in their initial stages of building MVPs. That helps them to understand their users and make the correct prioritization and decisions in the planning stages.
As an example, at the initial stage of creating the project, the User story mapping method was used by the successful well-known company Airbnb. They used mapping to visualize their MVP release and plan the first stages of implementation. They used a user story map (a combination of collected user stories based on customer research and market needs). At the top, there were user stories prioritized from left to right [4]. Such personas as “Guest” and “Host” were identified, the main actions that users can perform on their site were highlighted: “Find accommodation”, “book accommodation”, “create your own place”, and “learn about Airbnb” [4]. On the map, they created a three-level vertical organization that flows from more general and high-level ideas to more detailed requirements, tasks. When a full map of user stories was built, it was possible to analyze the priorities, risks, and dependencies that were clearly visible, which allowed determining the minimum package of user stories that needed to be developed and released in their MVP. This helped them to be flexible to market needs, they were able to achieve the minimum valuable product that customers loved, and as a result, they were able to build the most demanded product in this focus area [4].
Below are described the main concepts of this methodology in more detail from a theoretical point of view.

A User Story is a description of the system feature in natural, human-readable language. "Good story conversations are about who and why, not just what." [5] It is important to understand that user stories are not documentation or a list of predefined requirements. It is a combination of words, pictures. It explains why this feature should be built, not just describes its behaviour. Below is presented the suggested template for a user story in the book:
"As a [type of user]
I want to [do something]
So that I can [get some benefit]" - [5]

It can be written on the card, sticky note, or other suitable places for that. It is suggested that each story will also have a title. If the story describes a big solution, it should be sliced into smaller pieces that allow us to evaluate and see progress sooner.
To make the process of reviewing stories more efficient, Ron Jeffries came up with a 3 Cs strategy in his book "Extreme Programming Installed"[6].
3 Cs stands for:
Card. Write the user story. The template above can help to structure it.
Conversation. Tell the story to others. Build a shared understanding.
Confirmation. Confirm that everyone is on the same page. Agreement on acceptance criteria. Definition of done.

User Story Mapping is a technique of visualising the main idea/general picture of the complete system. It includes a systematisation of all the general points/actions - stories, that take place in the system, in chronological order. Story mapping helps to find the rabbit holes and helps to identify the risks before the actual implementation.
Here is an example of how story map can look like:

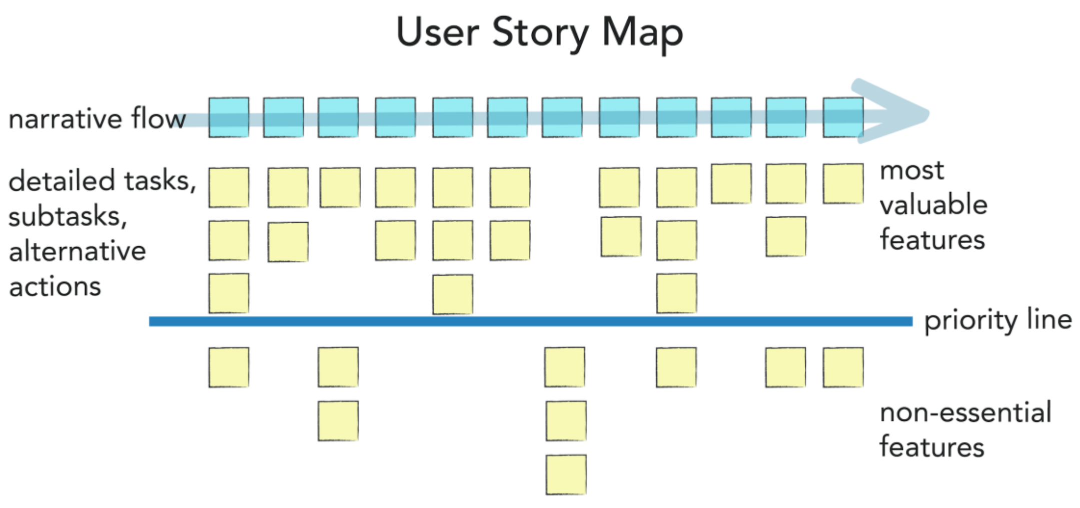


Figure 3: User Story Map [7]

The figure 3 illustrates the user story map structure. First level contains user stories in narrative flow. Each user story is divided into detailed tasks or subtasks. User story map combines all possible scenarios, dependencies and actions. Based on that it is much easier to identify what functionality is much more critical and required for building the

MVP. Priority line divides the map into two parts: most valuable features and non-essential features.

What does a user story map include?

Users or personas - Map describes a story that relates to a user or persona type that will help to solve/achieve a specific goal. Map should include a brief overview of personas type. Who is that person? What are the goals of this user? Main characteristics that describe the user type.
Activities - Activities organise tasks that should be achieved/performed by the same persona type.
Backbone - High level goals and tasks. The set of those stories build the structure of the map. All tasks across backbone should be a similar goal level. Usually they describe high level flow, and do not contain exact requirements that describe user behaviour. Tasks are organised in **the Narrative Flow** - the left to the right axis in a story map is built in the order the user tells the story.
Sub-tasks/details/exception - High level tasks are divided into sub-tasks, details and exceptions. It can contain more details about what system dependencies are involved, or UI details.
Release slice - the line that puts a limit on functionality that is achievable within iteration/ MVP scope.

2.3. MVP and User Story mapping technique

MVP is a minimum viable product. "The minimum viable product is the smallest product release that successfully achieves its desired outcomes" [5]. It is very important to learn what your MVP should look like. User stories and story mapping will help to understand it. There is always more to build than any team or company has time or resources to build. To achieve the goal, it is important to understand how to minimize output and maximize outcome and impact. In other words, it is impossible to build everything. New ideas will come all the time, but the team will not have enough time to build them. In this case, decisions are coming to the game. Things should be prioritized.

To make the correct decision, user stories or product requirement documents, or any other source of truth should have answers to those questions:

- Why should it be built? What is the idea behind it?
- Who are the customers and end-users?
- What are the benefits for the company and for the end-users who will use it?
- What problems does it solve for the company and for the customers?

Answers to those questions will help to prioritize outcomes and choose the most important stories.

What role do story maps play in building a successful MVP?
Story maps are a good visualisation of the whole product plan - a great tool to share the product view with the team, discuss details. Shared understanding grows.
Story maps show dependencies. It helps to make the correct scoping for the MVP.
Story maps technique helps to catch a lot of mistakes and thereby saves a lot of time for product managers and for the development team. It also saves a lot of money for the company.

Here are the main steps for creating a user story map that are presented by Jeff Patton in his book "User Story Mapping: Discover the Whole Story, Build the Right Product"[5].

- *Frame the problem.* The main idea, why should it be built?
- *Map the big picture.* Focusing on breadth, not depth. Try to catch all possible ways. Only then go and explore it in depth.
- *Explore.* Explore the users, different types of users. Explore different flows of the story.
- *Slice out a release strategy.* Prioritise things. Slice out less important stories.
- *Slice out a learning strategy.* Slice map for small MVP learning experiments. Try it out on the users. Learn what is really valuable for your customers.
- *Slice out a development strategy.* Plan what will be developed in the first place. It makes sense to focus on things that will help to identify the rabbit holes and risks sooner.

Storytelling is the main instrument to build user story maps. User stories maps should be always reviewed, discussed, and updated according to the latest learning outcomes.

"How to choose the first bundle of features that is both high value and immediately useful?" [8] - Jeff Patton wrote an article that gives an overview of how to achieve this goal.

The main idea is to always go with a user-centric approach. Always communicate with potential users, test your ideas, express features from a user's perspective.

Below are described steps that help to make a prioritisation and identify the most valuable features.

- First step is to collect data and information and create user stories. Analyse your users, receive feedback, and understand the user needs. After that proceed with collecting features and creating user stories. User story should describe how the feature will be used rather than how it will look or which implementation details it will include.
- Value estimation: Estimate how frequently the feature will be used. Identify the value (on the high level) that it will bring to the feature (low/medium/high).
- Build the Criticality vs Usage Sequence model. Draw a model with x and y axis. x: usage sequence and y: criticality as it is illustrated on the Figure 4 below.

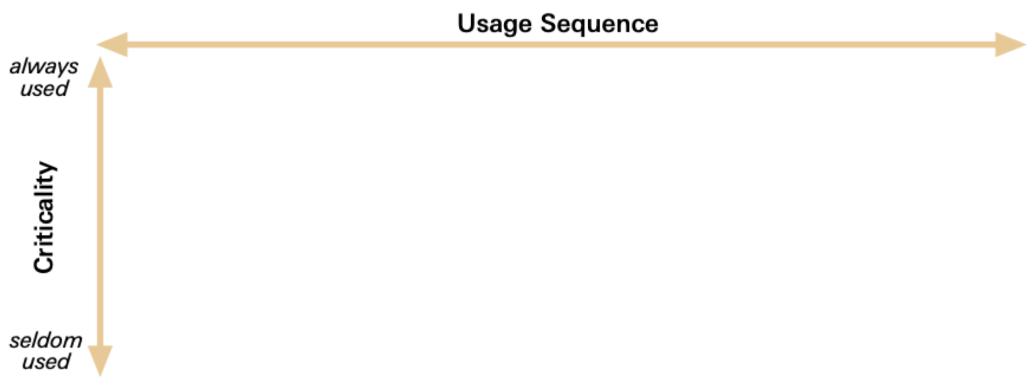


Figure 4: Build the Criticality vs Usage Sequence model [9]

Order cards (with user stories) by usage sequence first. Features can be overlapped if they don't have any dependencies with each other in the action flow. Figure 5 illustrates how user cards can be placed in usage sequence vs criticality model based on the usage sequence (step 1).

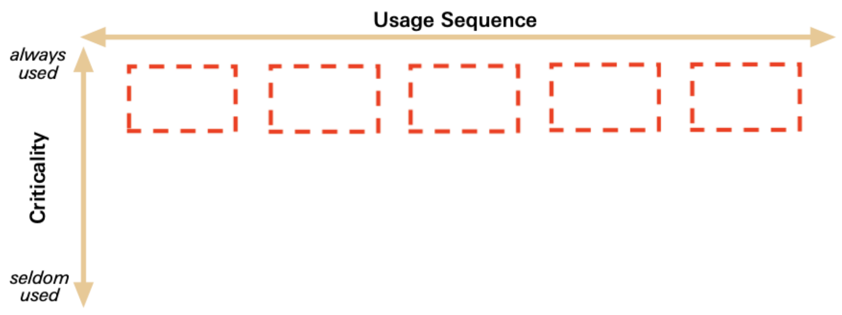


Figure 5: Usage sequence vs criticality model, step 1

Then using the analysis of frequency and value that were made before, adjust the model (y axis), as it is shown on Figure 6:

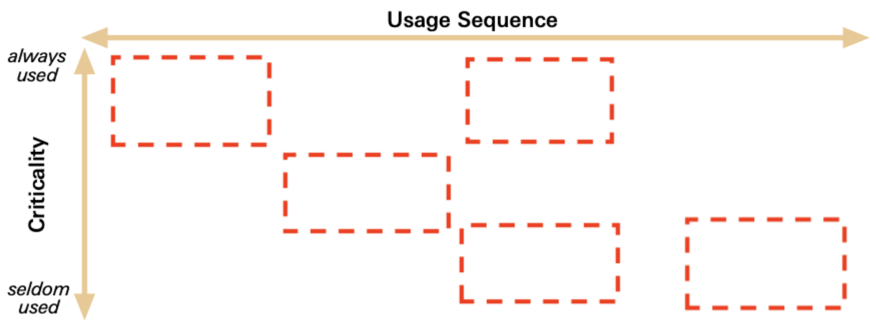


Figure 6: Usage sequence vs criticality model, step 2

- Break down the flow into logical pieces. Sometimes features have dependencies that are difficult to break. However, this diagram and exercise visuals the priorities. If there will be a logical break - slice the features into releases/interactions.
- After the scoping process, start to sketch and prototype. It is important to work in iterations and constantly receive feedback from your users. That is how knowledge can be received. That will be an indicator that shows if a solution is valuable and usable.
- Gather feedback, update user stories and maps, continue to build to learn more about user needs.

2.4. How to measure the success?

Above, two methods were analyzed in details, described how both of two methods play a role in creating of successful MVP. But what is the product's success and how to measure it? Here are provided some key factors which can help to give an estimate of product success.

Scope and Schedule

Have the goals been achieved within the set time frame? Was the planned work completed on time? To achieve these goals, it is very important to plan work correctly across all teams. Identify the most valuable features, correctly estimate implementation time, and descope non-priority features.

Customer satisfaction

How does the product meet the customer's expectations? Does it satisfy the end-user needs? To achieve that, customers and their needs should be always analyzed, feedback gathered, and the product development strategy should be agile for changes.

Team satisfaction

The team does not feel burned out. The team was not doing the work under pressure. No deadline-driven development. Working hours do not go beyond.
All these goals are aligned with well-planned work. To achieve the correct level of load balance the huge role plays the methodology of building the product. It is important that the team can focus on the clearly defined tasks and has a shared understanding of why and what should be built. The team should be engaged and have a correct focus.

Company profit

The whole product implementation process should bring a profit margin to the company. A lot of factors affect the company's profit and numbers. All resources should be optimally used, expenses minimised, and the main focus should be on building the right things with minimal input.

Quality

User experience quality. Is it easy and intuitive to use? Does it look good enough?
Functional quality. Does software work how it was agreed? Does it have a lot of bugs?

Maintainability. How easily can the software be maintained? How complex is it to extend it, add a new feature?
To meet good quality standards, the engineering, design, and product team should clearly understand what they are doing and why. Teams should have enough context to think about better implementation, usability, and possible future extension.

3. The Case Study

An active and healthy lifestyle is very common and popular nowadays. Now every second person is involved in some kind of sports activity, it can be football, yoga or just going to the gym. Many employers compensate for this service and that is why there is a huge demand for it. If we consider the city of Tallinn, then only here you can count dozens of different sports institutions, and this leads to the conclusion that there are too many competitors in this area. Today, standing out from the competition is not an easy task. To attract customers, you need to offer something that interests them. Mobile application is one of these tools that play a huge role in attracting more customers and receiving their satisfaction.

The mobile application can bring the following benefits to the company:

- Competitiveness
- Marketing
- Collection of statistics about more interesting services of the club
- Advertising - Since people visit app stores a lot (App Store, Play Market) - this is an advertisement for the club, also mobile applications can include free in-app advertising.

Impuls sport club does not have a dedicated mobile application, while their competitors like Gym club or MyFitness have it. Also, analyzing the gathered feedback, it was noticed that Impuls customers would like to use the mobile app as the website is not mobile responsive and time-consuming to use. It was decided to take this project as the case for studying the problem - how to build an effective MPV applying two selected implementation strategies.

In the chapters below two analyzed methods of building MVP, user story maps, and PRD-based implementation, will be applied to this case, and the results will be analyzed and compared.

3.1. Building MVP with PRD approach

This chapter describes steps that were done for identifying the MVP functionality for mobile application using PRD approach. First of all, it is important to understand all involved parties/ user types. Based on the surveys and researches it was identified three types of potential users: trainers, club administrators and club members.

Functionality was divided for four main parts: 1) authorization and initial setup, including role scoping, permissioning, login/logout functionality. 2) group classes/training management (opportunity to book, cancel, manage the class time) 3) capacity overview 4) account/profile information management.
Below all four parts are described and vizualized using use case diagrams.

- **Authorization** (login/logout functionality), role scoping, main screen (all roles)

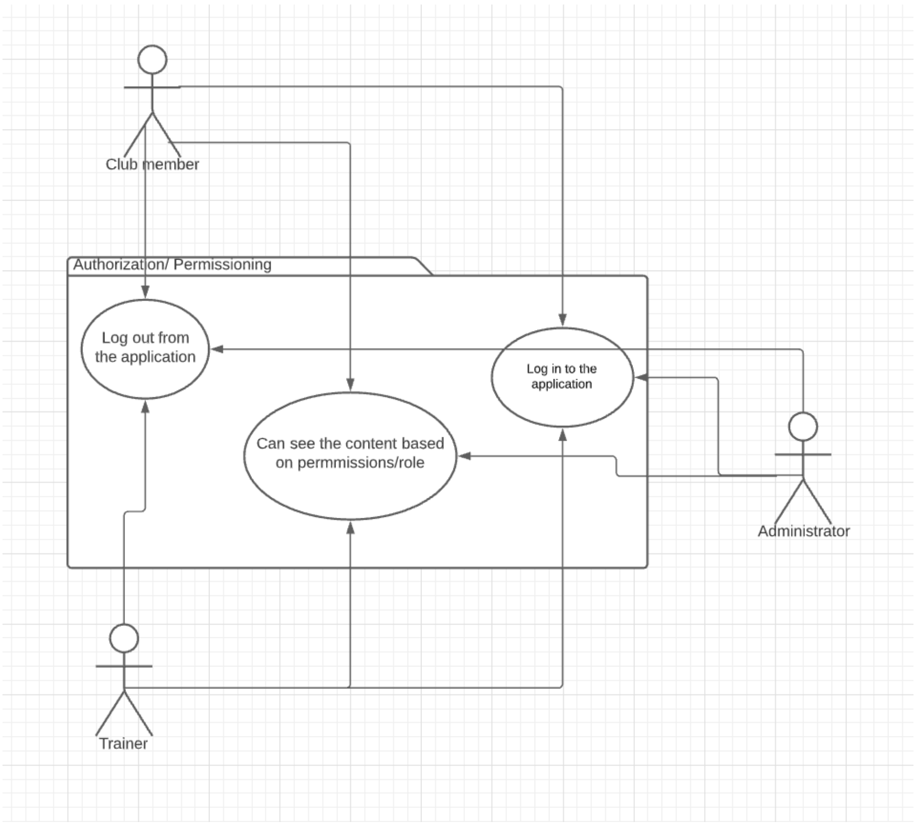
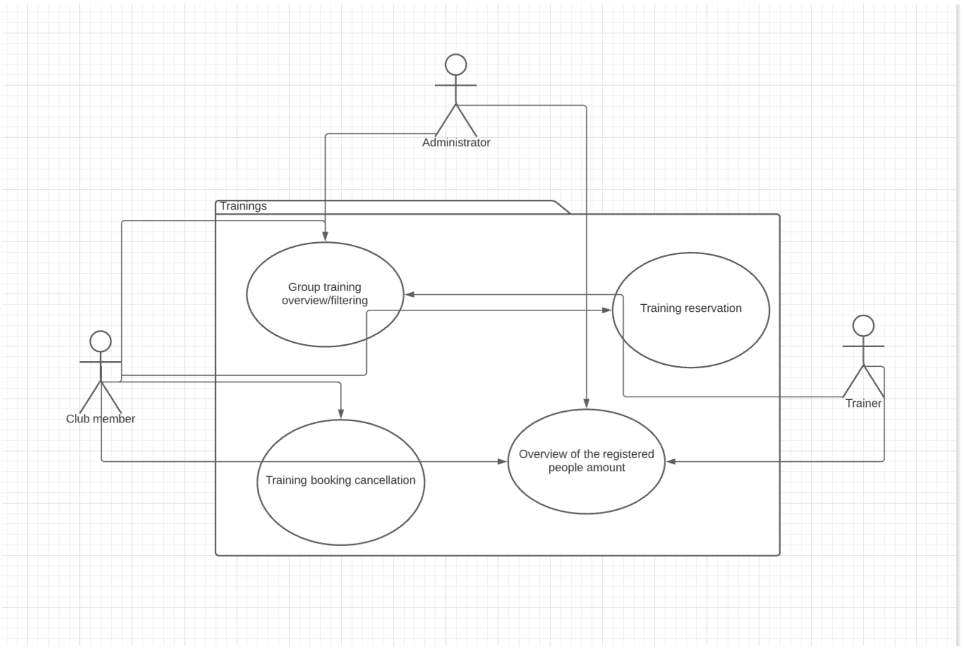

Club member
Authorization/ Permissioning
Log out from the application
Log in to the application
Can see the content based on permmissions/role
Administrator
Trainer

Figure 7: Authorization use case diagram

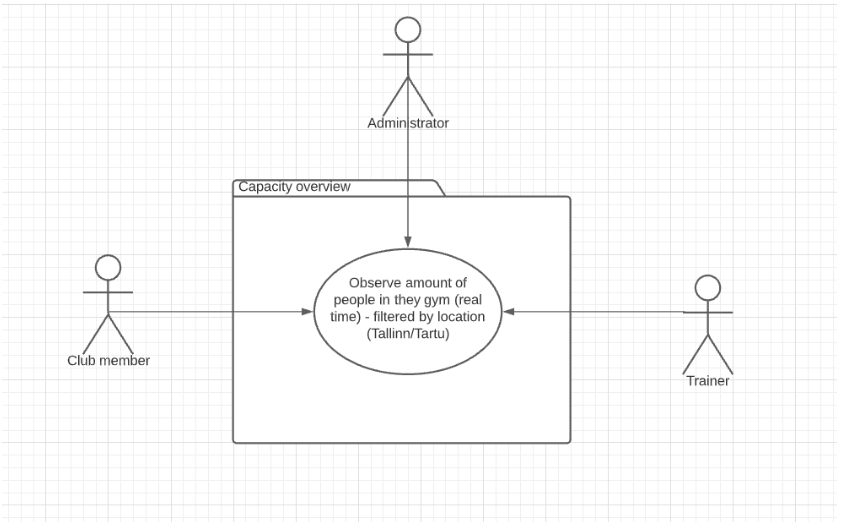
Club members, trainers and administrators can log in to the application. They should be able to see the content based on their permission and roles. For all roles there should be an option to log out from the application.

- **Group training management**

Club members, administrators and trainers should have an opportunity to search group training and filter them based on location, time and type. All roles should be able to observe the number of the registered people for a class. Club members should have an opportunity to register themselves for a call and cancel the reservation.


Administrator
Trainings
Group training overview/filtering
Training reservation
Club member
Trainer
Training booking cancellation
Overview of the registered people amount
Figure 8: Group training management use case diagram

- **Capacity overview**


Administrator
Capacity overview
Observe amount of people in they gym (real time) - filtered by location (Tallinn/Tartu)
Club member
Trainer
Figure 9: Capacity overview use case diagram

All roles can observe the amount of the people in the sport club in real time filtered by club location.

- **Profile information management**

All three roles should have an opportunity to review and edit their personal profile information. Club member roles should be able to upload COVID certificates into the system and also review their membership - invoice statement.

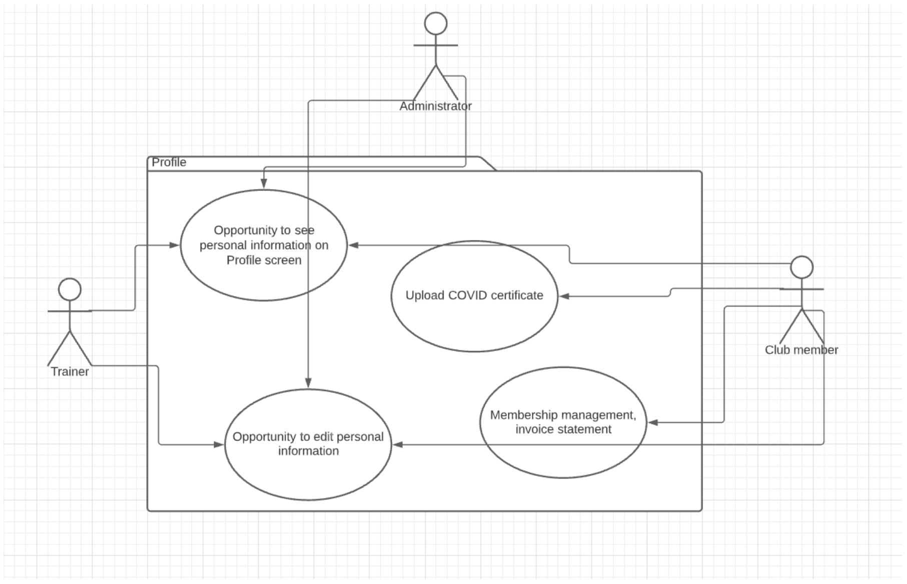

Administrator
Profile
Opportunity to see personal information on Profile screen
Upload COVID certificate
Trainer
Club member
Membership management, invoice statement
Opportunity to edit personal information

Figure 10: Profile information management use case diagram

As requirements and PRD approach still can be performed by Agile teams, using Scrum practices, it was decided to imitate the sprint scheduled work - implementation was divided into 4 iterations. 1 iteration is equal to 1 week.

1st iteration

Authorization (login/logout functionality), role scoping, main screen (all roles).

Training filtering and overview (club member).

2nd iteration

Group training reservation/bookings overview/cancellation (club member).

Bookings/training overview (trainer).

3rd iteration

Capacity overview (all roles).

Profile - general information overview/editing (trainer/club member).

4th iteration

Profile - Covid certificate upload (club member).

Profile - Membership overview and invoice state (club member).

Below are described feature requirements that were identified for the Impuls sports club mobile application using the PRD template that was provided above. As a result, it will be possible to compare this approach and story mapping technique and see which one provides more context and helps to build a proper MVP.

Table 1: PRD: Authorization, initial log in to application, main screen

Title	**Authorization, initial log in to application, main screen**
Goals and objectives	Allow users to login into the application to see the content related to their role. Without a need to verify the user role each time that they try to access the mobile app.
Target users	Club member, Trainer, Club Administrator
Timeline/ release plan	1st iteration
Success Criteria	Users can proceed through the authorization process. User stays logged into the app. Users can log out from the system. Application is able to correctly identify the user's role and permissions based on that.
Impacted Systems	Login, security and permission APIs.
In scope	Login flow, logged out functionality (assuming that user has account and registered in the system)
Out of scope	Registration process, changing role type during mobile app usage (if user has multiple roles), social logins
Possible Future Expansion	Registration through mobile application
Requirements	● Initial screen: form with two input fields: login name and password. Link "forgot password". ● System should be able to identify the current role of the user and display the content related to this role (considering all permissions, etc) ● After successful login, the main screen should have a club logo. Above the logo, a person's name is displayed. ● Log out button on the right corner.

Open questions	-
Dependencies	-
Potential risks	-

Table 2: PRD: List of group trainings, filtering by criteria

Title	**List of group trainings, filtering by criteria**
Goals and objectives	The ability to observe/review and plan a training in advance. Opportunity to observe what day and what time there is a workout of interest.
Target users	Club member, Administrator
Timeline/ release plan	1st iteration
Success Criteria	Opportunity to check group trainings (list of trainings), filtering by club location (Tartu/Tallinn), training and date
Impacted Systems	training API
In scope	List of all group trainings with time and date, filtering
Out of scope	Adding new training into the system (by administrator), booking (will be implemented and described in another PRD), cancel booking (will be implemented and described in another PRD).
Possible Future Expansion	Add more filters: trainers, training type, complexity, etc.
Requirements	Feature only available for club members. ● Navigation from the main screen to the screen with the opportunity to select the gym (Tallinn/Tartu).

	<ul><li>Users can also filter training by title and day.</li><li>After applying the filters, users can see a one day overview of all training sessions (by default), or a specific date/time + training.</li><li>Training information contains: club (location), title, day, time.</li></ul>
Open questions	How to arrange filtering? (User experience) What exactly can be filtered?
Dependencies	Authorization, role identification - should be implemented first.
Potential risks	-

Table 3: PRD: Group training reservation/bookings overview/cancellation

Title	**Group training reservation/bookings overview/cancellation**
Goals and objectives	A limited number of people are allowed in the gym / pool / group training. The ability to plan your training plan in advance.
Target users	Club member
Timeline/ release plan	2nd iteration
Success Criteria	User can register to the group training
Impacted Systems	training API, booking API
In scope	the ability to register for the training, booking cancellation.
Out of scope	Training description, trainer information, personal trainings
Possible Future Expansion	Personal training reservation, payment for the personal training
Requirements	<ul><li>After filtering/ fetching the training list (previous PRD), users</li></ul>

	can select the training from the list and make the reservation (Register button). ● Users should be able to see how many people are already registered and if there are any free spaces. ● If there are no free spaces, warning message is displayed: "There are no empty spaces, please try again later" ● After successful registration, all bookings are displayed on the main screen (with minimal information: training name, time). ● The first 5 are displayed and then the button “see all” appears. When a user clicks on a training, information appears with a confirmation and a cancel button. User can press the Cancel button, a confirmation of the cancellation appears and the training disappears from the list of registered workouts.
Open questions	-
Dependencies	Group training overview feature should be implemented first.
Potential risks	-

Table 4: PRD: Bookings/trainings overview for trainer

Title	**Bookings/trainings overview for trainer**
Goals and objectives	Full picture of upcoming/scheduled trainings. Understanding how many people were registered (so trainer can plan it properly, prepare the plan, equipment)
Target users	Trainer
Timeline/ release plan	2nd iteration
Success Criteria	List of upcoming trainings allocated for this specific trainer. Number of registered people.

Impacted Systems	training API, booking API
In scope	List of trainings (with minimal information: date/time/name). Number of registered people.
Out of scope	Training schedule management. (No opportunity to add/remove trainings, changing details).
Possible Future Expansion	Add opportunity for trainers change time/date of the training
Requirements	The first 5 trainings are displayed on the main page and then the button “see all” appears. "See all" screen - trainings are divided by dates. Training contains the following information: name, date, time.
Open questions	-
Dependencies	-
Potential risks	-

Table 5: PRD: Capacity overview

Title	**Capacity overview**
Goals and objectives	Covid times brings this topic to a high level. Allowance to have only 50% of club capacity, safety measures, etc. People want to know if there are a lot of people in the club at the moment and if it is safe to go there at this time.
Target users	Club member, Trainer
Timeline/	3rd iteration

release plan	
Success Criteria	Application shows how many people are in the club in real time (percentage).
Impacted Systems	user API
In scope	total occupancy
Out of scope	Detailed view on how many people are in the pool, main training hall, personal trainings.
Possible Future Expansion	Occupancy statistics by time and day
Requirements	● The main page displays the name of the club (city) and below it displays a number in the centre with occupancy in %. ● Data is updated every 5 minutes.
Open questions	-
Dependencies	Entry/exit gates should inform the system that a person enters or exits the club.
Potential risks	-

Table 6: PRD: Profile overview

Title	**Profile overview**
Goals and objectives	General information overview/editing.
Target users	Club member, Trainer, Club Administrator
Timeline/ release plan	3rd iteration

Success Criteria	Opportunity to change personal information as first and last names, date of birth, change password, contact information as e-mail and phone number.
Impacted Systems	user API
In scope	View/edit general information: name, date of birth, contact information and opportunity to change a password.
Out of scope	Invoice state, covid certificates upload (will be done in another iteration)
Possible Future Expansion	History of trainings, goals and progress.
Requirements	● Main screen should have a button that will navigate to the profile page. ● Read only view of all personal details: username, name, date of birth, e-mail, phone ● Opportunity to edit name, date of birth, e-mail, phone. Username field is not editable. ● Button to change password. ● When you change your email or password, a confirmation letter is sent to change your email (to a new address) or password.
Open questions	Each club member has their own Customer Coordinator. Do we need to display his/her name and contact details there? Or shall we move it to another screen?
Dependencies	-
Potential risks	-

Table 7: PRD: Invoice statement management

Title	**Invoice statement management**
Goals and objectives	All invoices are coming via email, sometimes it can be easily forgotten. Application can tell the user the current state of the invoice and notify if it should be paid and the due date was passed.
Target users	Club member
Timeline/ release plan	4th iteration
Success Criteria	Profile tab shows the current payment state of the invoice.
Impacted Systems	invoice API
In scope	Current payment status (Paid/unpaid).
Out of scope	Notifications, payment via mobile application, history of previous invoices.
Possible Future Expansion	Notifications for users that invoice date is in the past and invoice should be paid as soon as possible. Integration with the bank and payment via mobile application.
Requirements	Profile page contains information about invoice status. ● Invoice status: PAID/UNPAID. ● Unpaid status displayed in red colour ● Paid status displayed in green colour.
Open questions	-
Dependencies	Profile page
Potential risks	-

Table 8: PRD: Covid certificate upload

Title	**Covid certificate upload**
Goals and objectives	Right now, there are restrictions related to the latest Covid situation that says people can only visit public places with valid Covid certificates. To simplify the checking process, the covid certificate can be uploaded to the application. Club administrators will see it immediately when a person enters the gates.
Target users	Club member
Timeline/ release plan	4th iteration
Success Criteria	Opportunity to upload covid certificate Background validation if certificate is valid
Impacted Systems	document API
In scope	Document upload
Out of scope	Certificate validation and integration with digilugu.
Possible Future Expansion	-
Requirements	● On profile page there is a button to add a Covid certificate ● When you click on the "Add a Covid certificate" button, a user can select a file from the device and upload it. ● After saving the file, the file goes to the database. When a club member enters the club, the system sees that the person has a certificate.
Open questions	-
Dependencies	Profile page

Potential risks	-

These PRDs illustrate the way of providing requirements to the engineering team.
This is a complete version of all features that are considered in the scope of MPV.
The next chapter will focus on identifying the minimum set of functionalities that will be valuable for the end-users using the user story mapping method. How much common functionality will these two methods bring as a result? A comparison of these two approaches will help to identify missing points and gaps in PRD way of building MVPs.

3.2. Building MVP with user story mapping approach

As the second approach for creating MVP was selected story mapping technique.
Below are described steps that were part of the process.

Work plan

- Identify the timeframe and specify the borders

MPV (prototype version) will be built during 1-month implementation, using a 1-week sprint (4 iterations in total).

- Identify Personas - a potential type of users

The scope will be limited by three types of personas: Club administrator, Trainer, and Club Member.

- Prepare conversations (surveys, dialogs)

Initial surveys with questions for each type of personas.

- Gather data from users

Find people that correspond to selected personas types, ask them to fill in surveys, combine and analyse received answers

- Write user stories

Write user stories using the approach described above (WHO, WHAT, WHY)

- Build a story maps - Plan, scope, split (product backlog, sprint backlog)

Build a story map using user stories, identify the priorities, remove things that are not going to be part of the upcoming MVP. Plan the sprint backlog.

- Build a prototype

Build a prototype based on the user stories and user story map for sprint 1.

- User testing, gathering feedback

Allow users to go through the prototype (with predefined tasks/scenarios in advance), ask questions if they found this functionality easy to use, useful, and helpful.

- Learn and analyse

Analyse data received from the testing process. See what can be improved, descoped, prioritised.

- Improve and repeat

Make conclusions, based on the received results build new story maps, and perform the same loop again

- Present the final version. Final testing and assessment.

Review the original spec and what was built in the end. Measure the success.

Personas

Club administrator.
The administrator is responsible for organising the work of the club. She needs to constantly interact with managers, coaches, and clients. The administrator must have complete information about the functioning of the club, know all programs, schedules, changes in them, customers, employees, equipment, club services, and prices, resolve conflict situations, etc.
Trainer.
This is the person who works in the club, leads the training, monitors the correctness of physical exercises. The coach can work with a group or individually. She/he may specialise in exercise equipment or in areas such as aerobics, aqua aerobics, Pilates, fitness yoga, and others. The trainer's responsibilities also include body analysis and personal nutritional advice.
Club member.
This is a person who visits the club, uses the services provided by the club. A club member can just go to the gym, take a personal training session with a trainer in the gym, attend group training and the pool.

Initial Survey and data

A total of 15 people were interviewed, of which 10 are club members, 3 trainers and 2 administrators.

First question was asked to understand how often people use the gym. This question helps to identify if there will be any usage/profit from the mobile application. Almost half of the club members answered that they visit the club 2-3 times per week. 20% of club members say that they use the gym less than 2 times per week and others answers were divided between visiting the gym more than 3 times per week and not visiting the gym within the last month.

Almost everyone that are club members answered that they find mobile applications useful and helpful. Based on the given answers, the main reasons to have mobile applications are that it is much faster and easier to use rather than do the same actions using the web-application.

However, it was also mentioned that it might be only useful for club members, not for workers such as administrators or trainers. As they use totally different software to manage all activities there.

As must have features were selected: timetable, registration for training, list and description of trainers together with workout description.

As an additional features users also mentioned the following:

- Gym occupancy in the real time (mentioned by 4 people)
- Number of reservation available for a group training/ remaining number of slots (mentioned by 3 people)
- Personal information modification (mentioned by 3 people)
- Covid certificate upload (mentioned by 2 people)
- Description of the workout/training (mentioned by 2 people)
- Upcoming trainings list/reminders with amount of registered people (mentioned by 2 people with trainer/coach role)
- Chat with/contact information of Customer Coordinator (mentioned by 2 people)
- Description of equipment. At the moment, in the gym, there is no instruction on how to use the equipment (mentioned by 1 person)
- Personal training registration (mentioned by 1 person)
- Burned calories statistics, synchronisation process between watches, etc - (mentioned by 1 person)

Based on the survey, the most useless and distracting features of the mobile applications are: Advertising, Prices and News.

To draw the conclusion based on the received results, the following features can be considered for building MVP:

- Trainings timetable with training overview (club member)
- Group training registration/cancellation with available slots (club member)
- Gym occupancy in the real time (club member)
- Personal information modification (club member)

As an additional features (have lower priority based on the survey):

- Covid certificate upload (club member)
- Upcoming trainings reminders with amount of registered people (trainer)

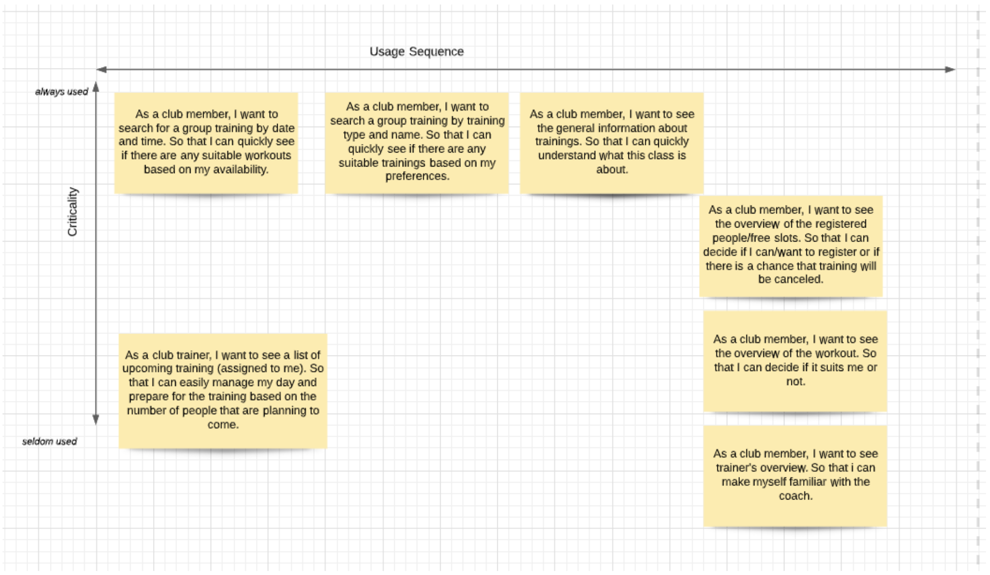
Compared with the initial plan based on a PRDs described in the previous chapter, invoice statement overview does not play an important role for the potential users. Also, people want to know more about group classes, this point is missing in initial PRDs. Also, the survey showed that administrators are not interested in mobile applications. Trainers think it might be a good addition to have such, but not a priority for them.

Usage sequence vs criticality model

Before building a user story map, we need to understand the main functionality that we want to consider and briefly identify the priority of the features. To build this understanding, we will use the **Usage sequence vs criticality model.** This model was described in the "Background and Main concepts" chapter.

Timetable with training overview (club member, trainer).

From the figure provided below, it can be observed that the main focus should be on club member roles - filtering by dates/time and opportunity to search by training type. As a quick win - just the general information about training can be displayed, and then adjusted to display the amount of registered people, detailed description about workout itself and trainer information. Overview of upcoming classes assigned to a trainer is not a priority and can be considered as a phase 2.



As a club member, I want to search for a group training by date and time. So that I can quickly see if there are any suitable workouts based on my availability.

As a club member, I want to search a group training by training type and name. So that I can quickly see if there are any suitable trainings based on my preferences.

As a club member, I want to see the general information about trainings. So that I can quickly understand what this class is about.

As a club member, I want to see the overview of the registered people/free slots. So that I can decide if I can/want to register or if there is a chance that training will be canceled.

As a club trainer, I want to see a list of upcoming training (assigned to me). So that I can easily manage my day and prepare for the training based on the number of people that are planning to come.

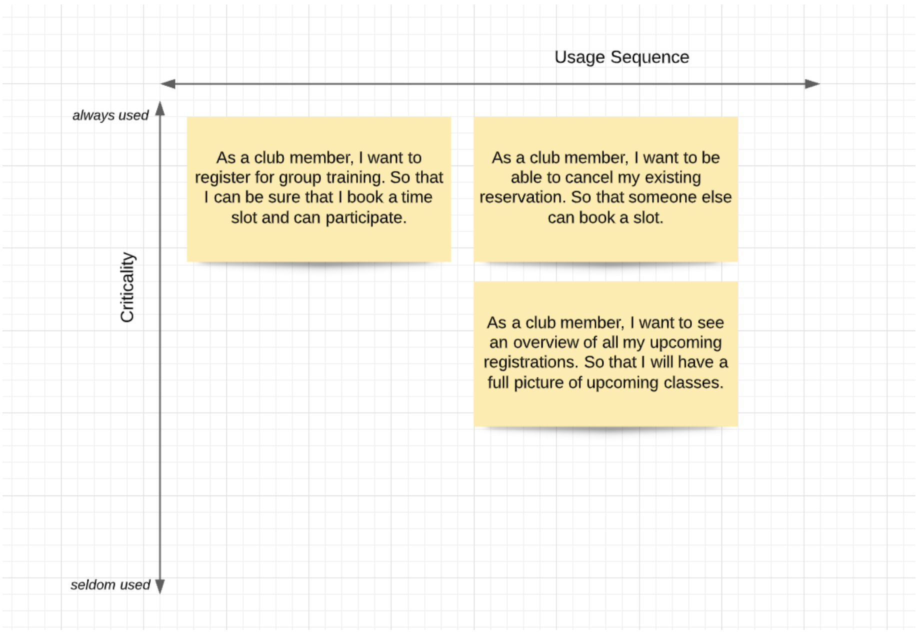
As a club member, I want to see the overview of the workout. So that I can decide if it suits me or not.

As a club member, I want to see trainer's overview. So that i can make myself familiar with the coach.

Figure 11: Usage sequence vs criticality model: Timetable with training overview

Training registration and cancellation.

Pre-condition for this functionality is Timetable with a training overview feature for club member roles.



As a club member, I want to register for group training. So that I can be sure that I book a time slot and can participate.

As a club member, I want to be able to cancel my existing reservation. So that someone else can book a slot.

As a club member, I want to see an overview of all my upcoming registrations. So that I will have a full picture of upcoming classes.

Figure 12: Usage sequence vs criticality model: training registration

Figure above illustrates that quite all functionality is important there, however, if some de-scoping will be required - overview of the upcoming classes can be implemented outside of MVP.

Gym occupancy.

Figure below shows that this feature does not have any dependencies and has a big value for potential users.

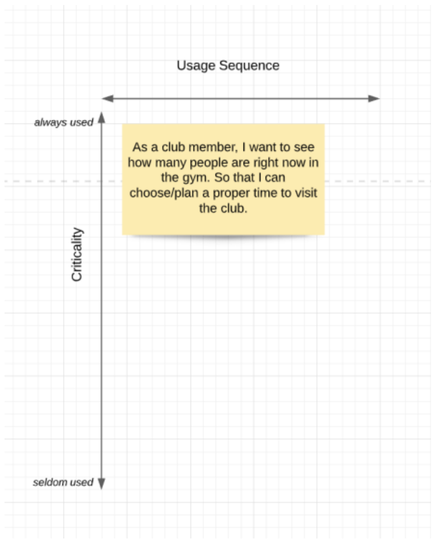

Usage Sequence
always used
Criticality
seldom used
As a club member, I want to see how many people are right now in the gym. So that I can choose/plan a proper time to visit the club.

Figure 13: Usage sequence vs criticality model: Gym occupancy

Profile information management.

The figure below shows that the most valuable functionality would be a modification of personal information. Covid certification upload and contact with customer coordinator can be considered in the phase 2 of implementation or some functionality can be descoped.

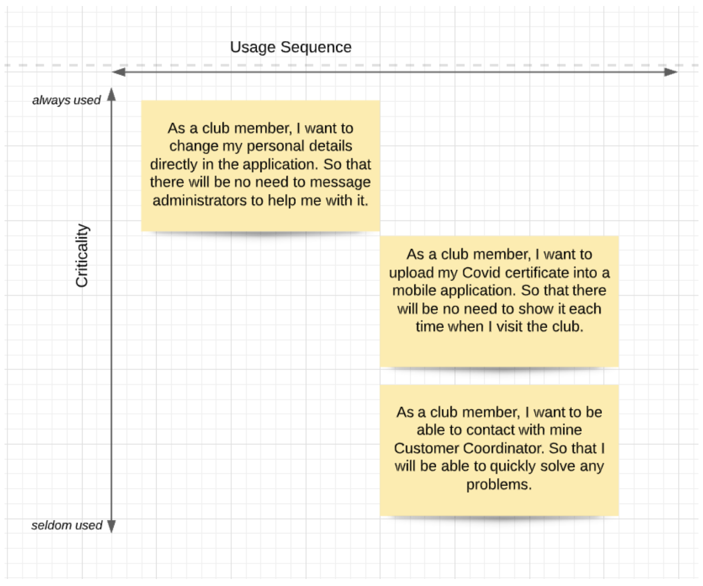

Usage Sequence
always used
Criticality
seldom used
As a club member, I want to change my personal details directly in the application. So that there will be no need to message administrators to help me with it.
As a club member, I want to upload my Covid certificate into a mobile application. So that there will be no need to show it each time when I visit the club.
As a club member, I want to be able to contact with mine Customer Coordinator. So that I will be able to quickly solve any problems.

Figure 14: Usage sequence vs criticality model: Profile information management

User story maps

Below are presented user story maps with detailed requirements. Blue line shows the main focus - scope that will go into release. Decision was made based on the priority estimation that was made in the previous chapter.

Club member - training filtering and overview.

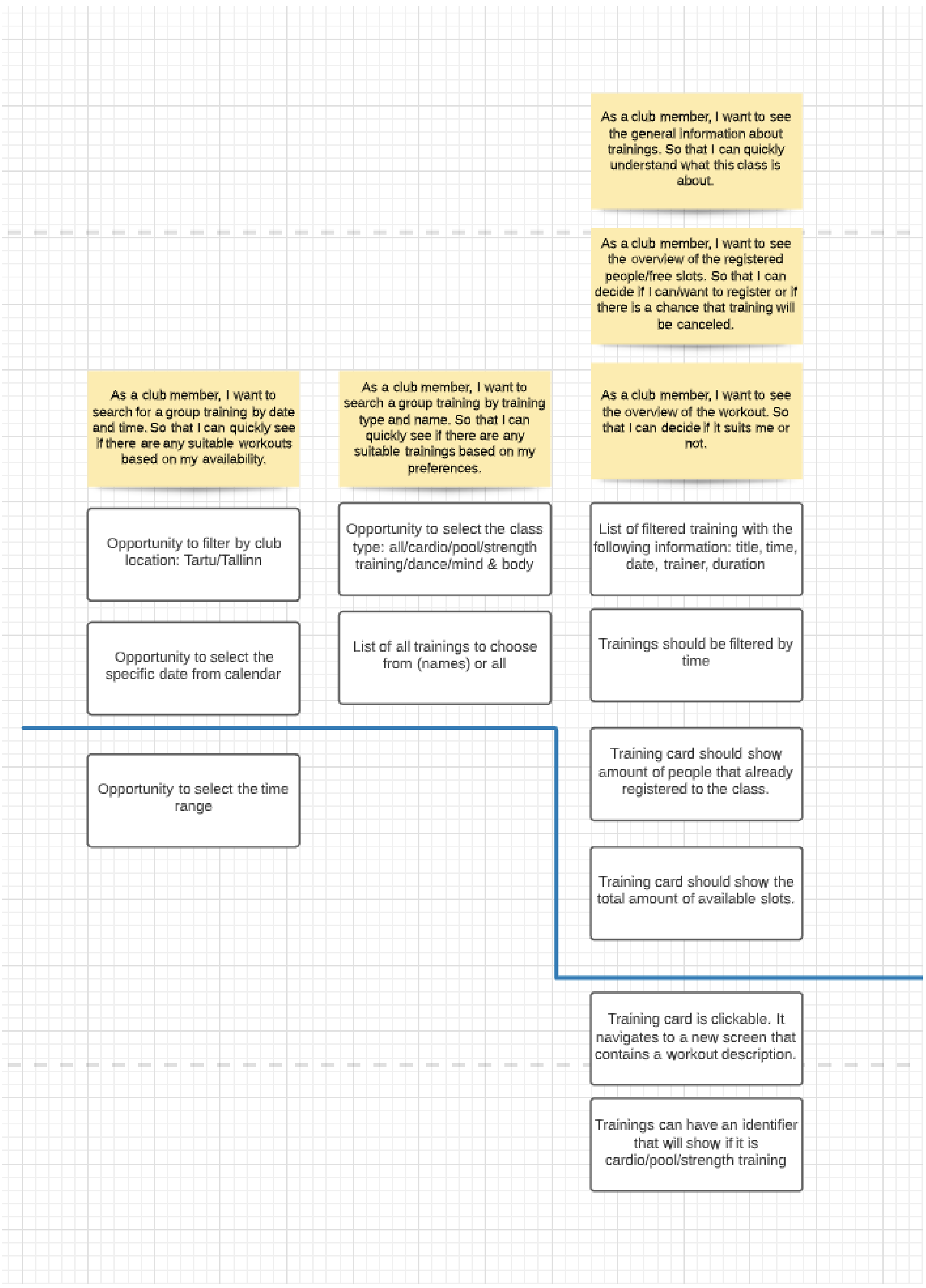

As a club member, I want to see the general information about trainings. So that I can quickly understand what this class is about.
As a club member, I want to see the overview of the registered people/free slots. So that I can decide if I can/want to register or if there is a chance that training will be canceled.
As a club member, I want to search for a group training by date and time. So that I can quickly see if there are any suitable workouts based on my availability.
As a club member, I want to search a group training by training type and name. So that I can quickly see if there are any suitable trainings based on my preferences.
As a club member, I want to see the overview of the workout. So that I can decide if it suits me or not.
Opportunity to filter by club location: Tartu/Tallinn
Opportunity to select the class type: all/cardio/pool/strength training/dance/mind & body
List of filtered training with the following information: title, time, date, trainer, duration
Opportunity to select the specific date from calendar
List of all trainings to choose from (names) or all
Trainings should be filtered by time
Opportunity to select the time range
Training card should show amount of people that already registered to the class.
Training card should show the total amount of available slots.
Training card is clickable. It navigates to a new screen that contains a workout description.
Trainings can have an identifier that will show if it is cardio/pool/strength training

Figure 15: User story map: Club member - training filtering and overview

After building a user story map for training filtering and overview functionality, low priority features are the following: opportunity to filter classes by time range, workout detailed description, identification that illustrates the training type. Main focus areas are: filtering by location, date, class type, scrolling list of available classes with available slots for registration.

Training reservation and cancellation.

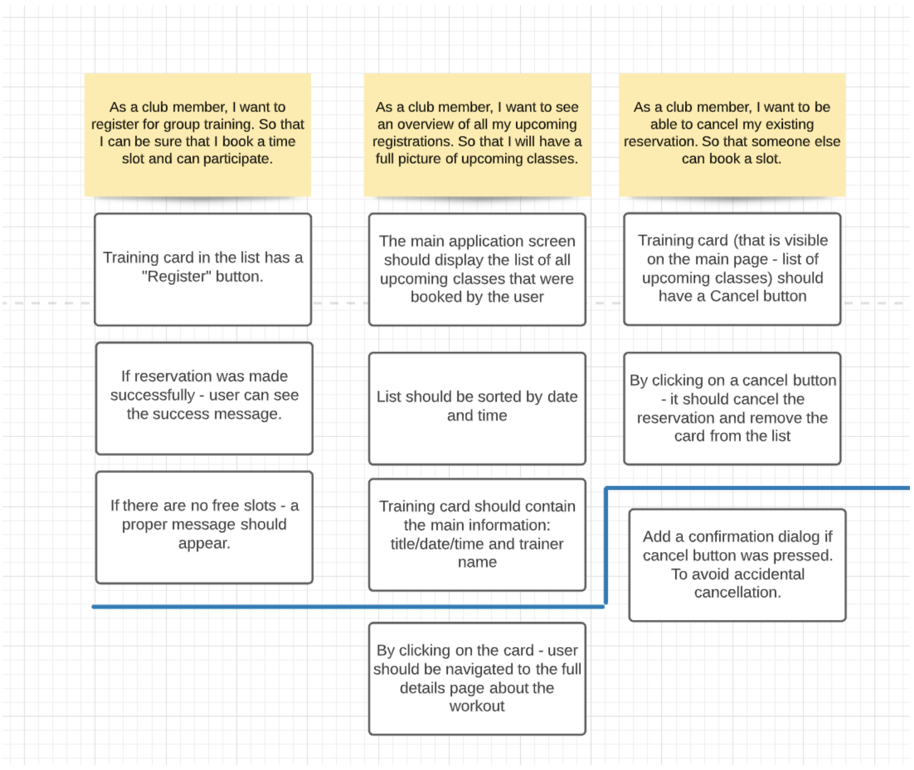

As a club member, I want to register for group training. So that I can be sure that I book a time slot and can participate.
As a club member, I want to see an overview of all my upcoming registrations. So that I will have a full picture of upcoming classes.
As a club member, I want to be able to cancel my existing reservation. So that someone else can book a slot.
Training card in the list has a "Register" button.
The main application screen should display the list of all upcoming classes that were booked by the user
Training card (that is visible on the main page - list of upcoming classes) should have a Cancel button
If reservation was made successfully - user can see the success message.
List should be sorted by date and time
By clicking on a cancel button - it should cancel the reservation and remove the card from the list
If there are no free slots - a proper message should appear.
Training card should contain the main information: title/date/time and trainer name
Add a confirmation dialog if cancel button was pressed. To avoid accidental cancellation.
By clicking on the card - user should be navigated to the full details page about the workout

Figure 16: User story map: Club member - Training reservation and cancellation

Story map of training reservation and cancellation functionality shows that non-focus areas are: detailed description of the workout, confirmation dialog after cancel button was pressed. Ordered by date and time overview of upcoming bookings is important. Ability to book and cancel the class is the main scope of the feature.

Gym occupancy.

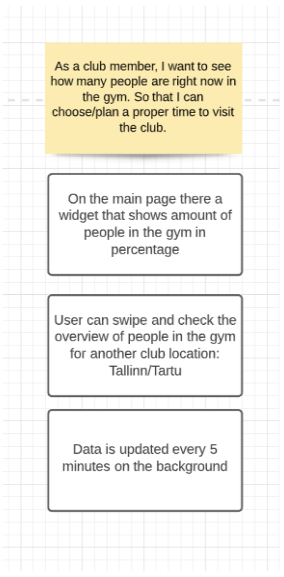

As a club member, I want to see how many people are right now in the gym. So that I can choose/plan a proper time to visit the club.
On the main page there a widget that shows amount of people in the gym in percentage
User can swipe and check the overview of people in the gym for another club location: Tallinn/Tartu
Data is updated every 5 minutes on the background

Figure 17: User story map: Club member - Gym occupancy

The gym occupancy story map is simple, and all functionality is in the scope of the project. Also, it is quite important to constantly update the data without disturbing the user experience.

Profile information management.

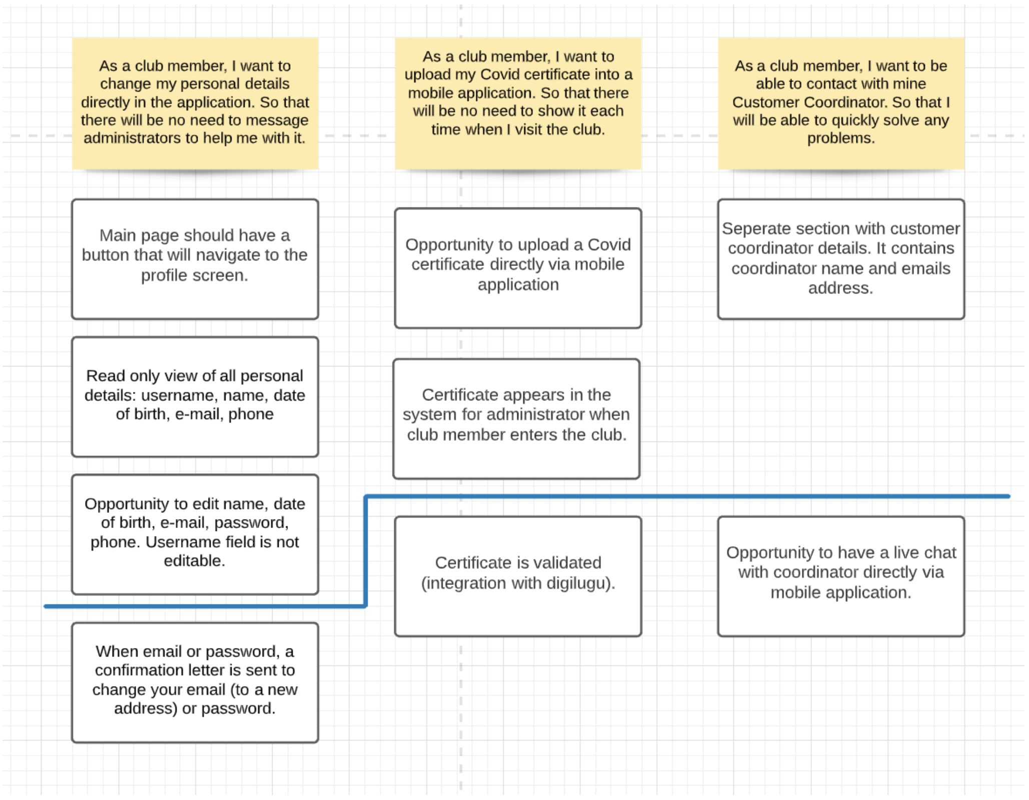

As a club member, I want to change my personal details directly in the application. So that there will be no need to message administrators to help me with it.
As a club member, I want to upload my Covid certificate into a mobile application. So that there will be no need to show it each time when I visit the club.
As a club member, I want to be able to contact with mine Customer Coordinator. So that I will be able to quickly solve any problems.
Main page should have a button that will navigate to the profile screen.
Opportunity to upload a Covid certificate directly via mobile application
Seperate section with customer coordinator details. It contains coordinator name and emails address.
Read only view of all personal details: username, name, date of birth, e-mail, phone
Certificate appears in the system for administrator when club member enters the club.
Opportunity to edit name, date of birth, e-mail, password, phone. Username field is not editable.
Certificate is validated (integration with digilugu).
Opportunity to have a live chat with coordinator directly via mobile application.
When email or password, a confirmation letter is sent to change your email (to a new address) or password.

Figure 18: User story map: Profile information management

Profile information management feature user story map shows that low priority features are COVID certificate validation, live chat with club coordinator and email confirmations in case of personal information changes. Main focus areas are opportunity to see profile data and modify it, certificate upload and contact information of customer coordinator.

Planning

In the previous chapter were generated usage sequence vs criticality models for each feature separately. Also, user story maps were presented for the individual feature. Now it is possible to combine all collected information together and build the overall sequence and criticality diagram that combines all features together. Referring to which it will be possible to identify the backlog and make a prioritisation.
High priority tasks were divided between first and second iteration. Medium level of priority tasks were assigned to the 3rd and 4th iteration.

The figure 19 illustrates three levels of priority according to usage and value that feature will bring to the end users. The main functionality that has a high priority is the basic training filtering, general training overview and opportunity to book and cancel the class.
The medium level of priority or nice to have features are: sport club occupancy, profile information management and additional information about training and trainer.
Low priority additions are: Covid certificate upload, customer coordinator contact information, and overview of upcoming classes for the trainer persona type.

Based on these findings, the initial backlog was organised using high and medium priorities stories and their sub-tasks that were described in story user maps above.
The low priority features were out of scope for the beginning.
High priority tasks were divided between first and second iteration. Medium level of priority tasks were assigned to the 3rd and 4th iteration.

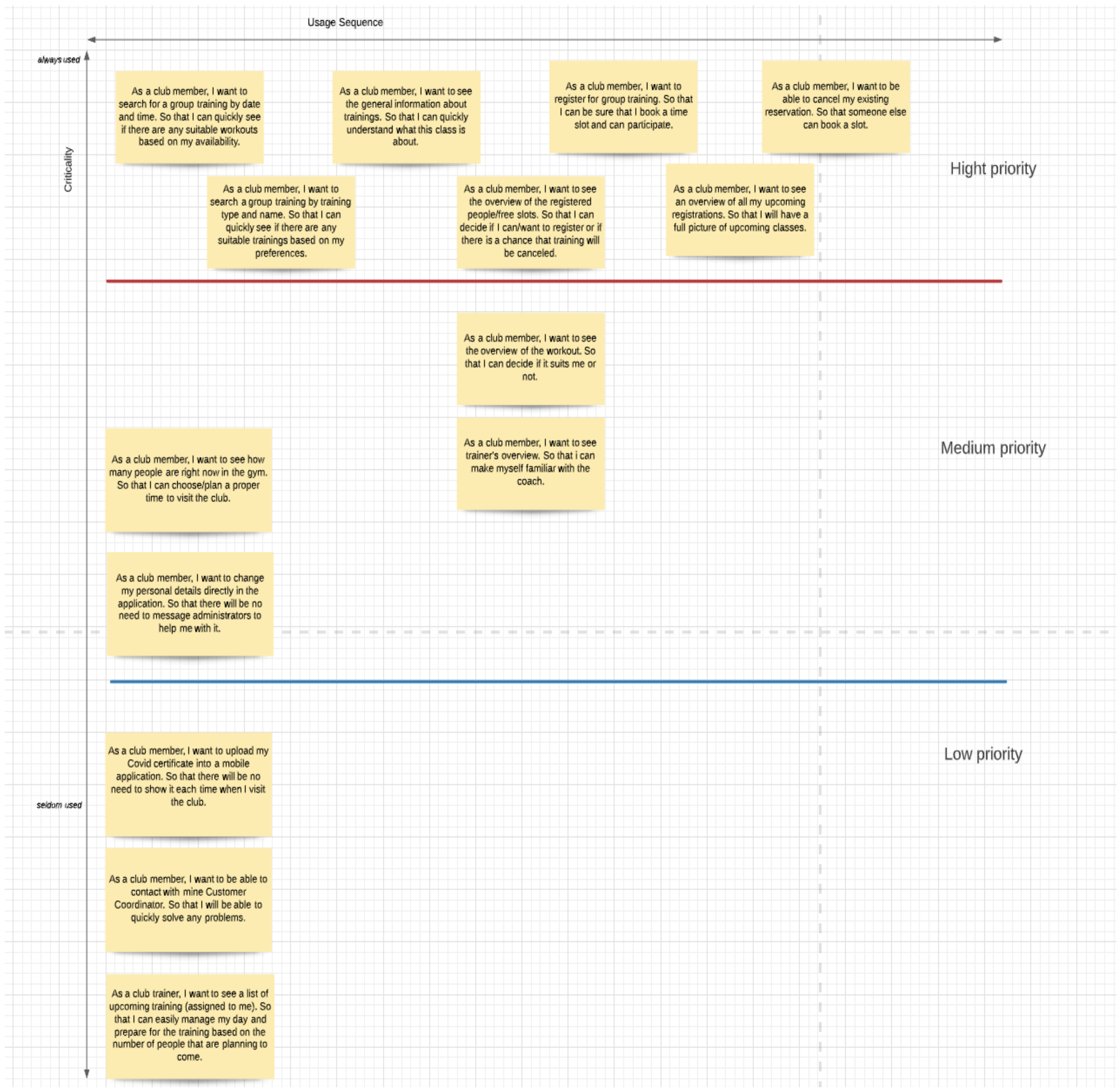

Usage Sequence
always used
Criticality
seldom used
As a club member, I want to search for a group training by date and time. So that I can quickly see if there are any suitable workouts based on my availability.
As a club member, I want to see the general information about trainings. So that I can quickly understand what this class is about.
As a club member, I want to register for group training. So that I can be sure that I book a time slot and can participate.
As a club member, I want to be able to cancel my existing reservation. So that someone else can book a slot.
As a club member, I want to search a group training by training type and name. So that I can quickly see if there are any suitable trainings based on my preferences.
As a club member, I want to see the overview of the registered people/free slots. So that I can decide if I can/want to register or if there is a chance that training will be canceled.
As a club member, I want to see an overview of all my upcoming registrations. So that I will have a full picture of upcoming classes.
Hight priority
As a club member, I want to see the overview of the workout. So that I can decide if it suits me or not.
As a club member, I want to see trainer's overview. So that i can make myself familiar with the coach.
As a club member, I want to see how many people are right now in the gym. So that I can choose/plan a proper time to visit the club.
Medium priority
As a club member, I want to change my personal details directly in the application. So that there will be no need to message administrators to help me with it.
As a club member, I want to upload my Covid certificate into a mobile application. So that there will be no need to show it each time when I visit the club.
Low priority
As a club member, I want to be able to contact with mine Customer Coordinator. So that I will be able to quickly solve any problems.
As a club trainer, I want to see a list of upcoming training (assigned to me). So that I can easily manage my day and prepare for the training based on the number of people that are planning to come.

Figure 19: User story map: Feature prioritisation based on usage and value

Implementation

Prototype was implemented using Proro.io tool [10].

First iteration outcomes

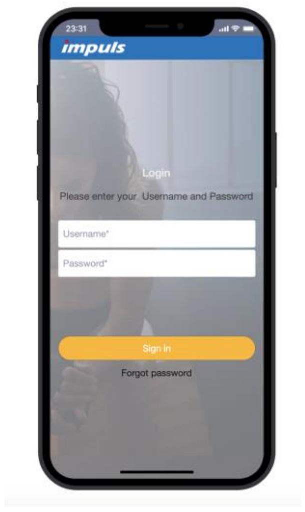


Figure 20: Prototype feature: Initial login screen.

Based on the user testing, the following feedback was received: some people tried to find a registration button. However, registration is out of scope. To improve the experience, it was decided to add a warning message on the bottom: "Not a member yet? Contact us: info@impuls.ee"

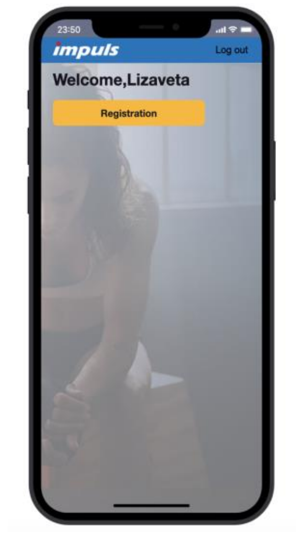
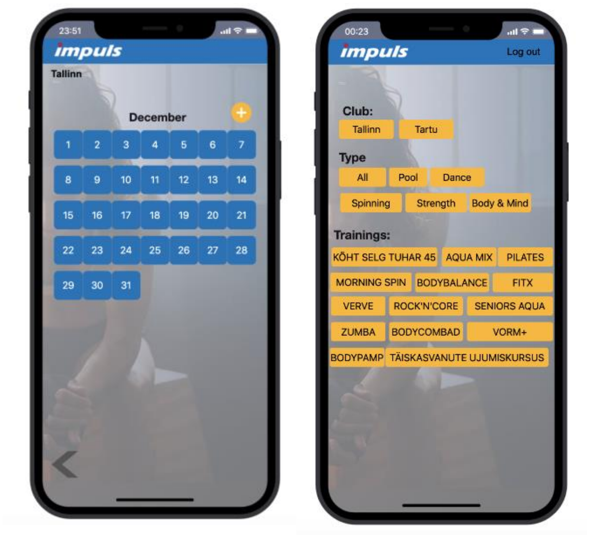


Figure 21: Prototype feature: Main page without any bookings/registrations.

Users were confused why it is blank and does not contain any information. To improve experience in case of no active reservation, it was decided to add a text: "You don't have any active bookings yet.".
"Registration" button is not self-explanatory. Should be renamed to "Register to a group training".



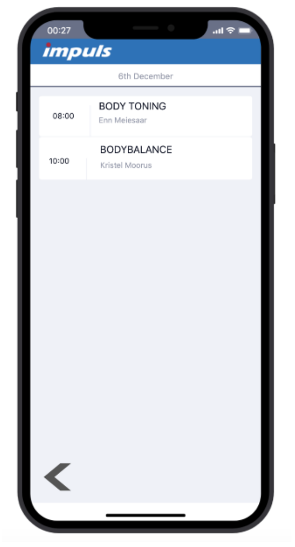
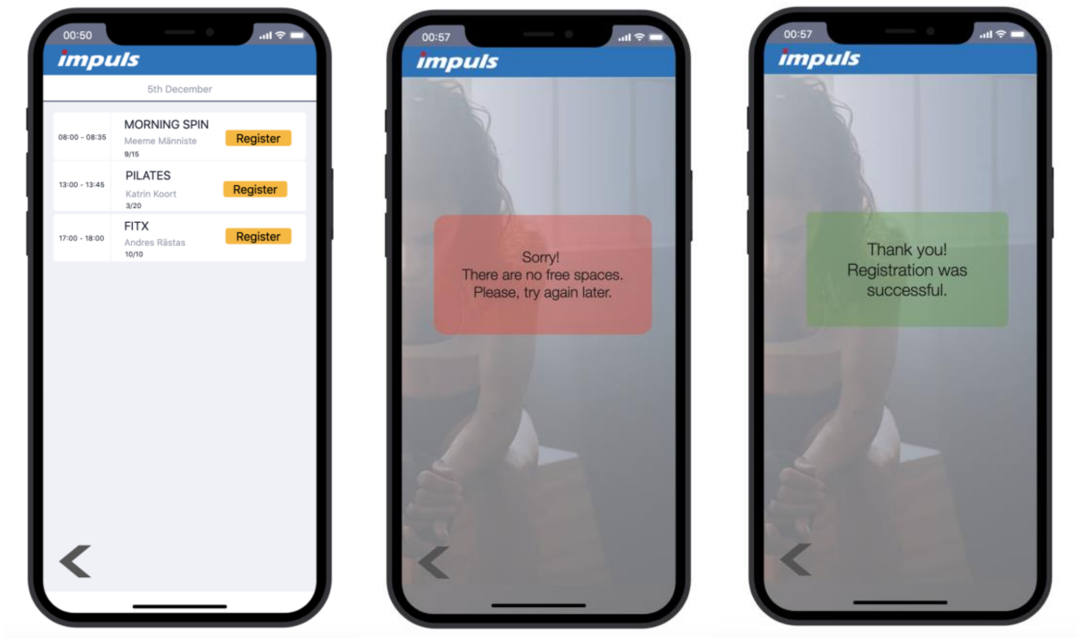


Figure 22: Prototype feature: Training filtering, Figure 23: Prototype feature: List of filtered trainings

Based on the feedback, the opportunity to filter by time range is quite critical and people find that it is missing. Club selection can be removed from the separate filtering window and moved to the initial filtering screen as a drop-down field as people usually visit only one club and no need to select the location each time. Also, the calendar should highlight the current date.

List of filtered training figure 23 can contain not just a start time but also the end time (or duration).

Second iteration outcomes



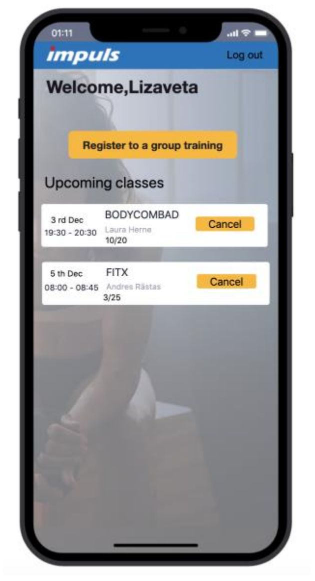
Sorry!
There are no free spaces.
Please, try again later.

Thank you!
Registration was successful.

Figure 24: Prototype feature: Training registration

The second iteration was focused on improving functionality based on the received feedback from user testing after the first iteration. Also, the registration functionality was implemented.

Users were happy to see notifications if booking was successful or not. However, on the list of trainings, the majority of users found numbers that indicate booked slots confusing, for example: "3/20". They do not understand what it means. To make it more clear, it can be changed to "3/20 booked".



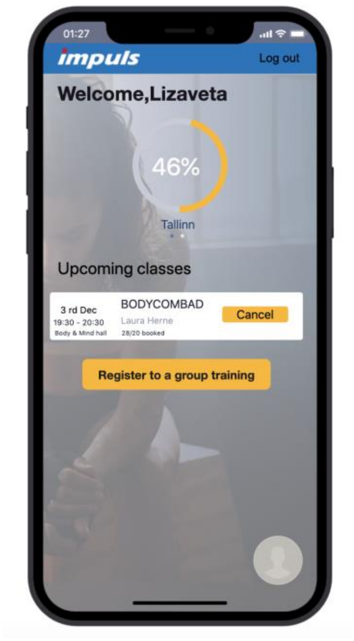
3 rd Dec 19:30 - 20:30	BODYCOMBAD Laura Herne 10/20	Cancel
5 th Dec 08:00 - 08:45	FITX Andres Rästas 3/25	Cancel

Figure 25: Prototype feature: List of personal reservations

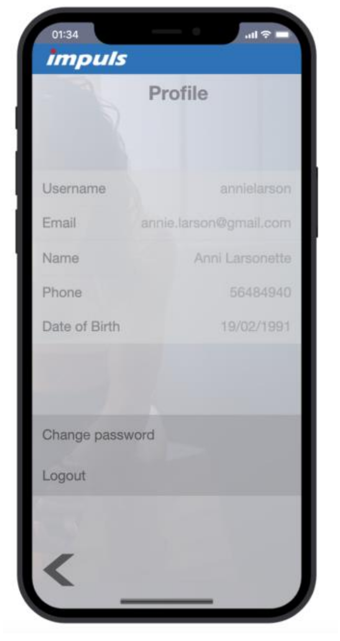
For personal reservations, users also want to see what location the training will take place (Tallinn or Tartu club). Also, the gym itself has multiple halls. Users want to see the hall name on the overview card.

During the testing, users accidentally pressed on the "Cancel" button. Confirmation dialog should appear before performing the action - "Are you sure you want to cancel this reservation?".

Third iteration outcomes



3 rd Dec 19:30 - 20:30 Body & Mind hall	BODYCOMBAD Laura Herne 28/20 booked	Cancel



Username	annielarson
Email	annie.larson@gmail.com
Name	Anni Larsonette
Phone	56484940
Date of Birth	19/02/1991

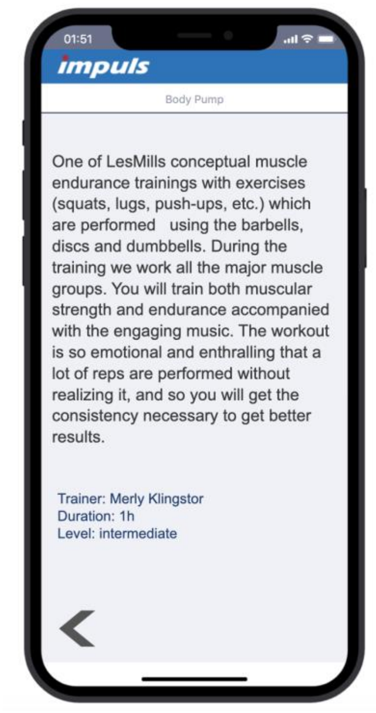
Figure 26: Prototype feature: Club occupancy

Figure 27: Prototype feature: Profile management

The positive feedback was received related to the profile management screen.

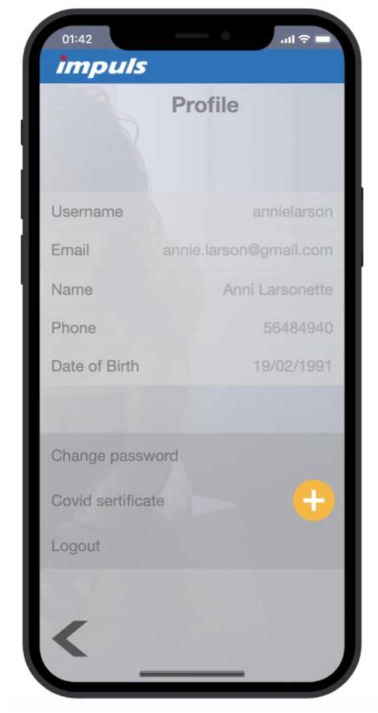
For club occupancy - not obvious from the first glance what this number means. Should have a title or a helper text.

Fourth iteration outcomes


01:51
impuls
Body Pump

One of LesMills conceptual muscle endurance trainings with exercises (squats, lugs, push-ups, etc.) which are performed using the barbells, discs and dumbbells. During the training we work all the major muscle groups. You will train both muscular strength and endurance accompanied with the engaging music. The workout is so emotional and enthralling that a lot of reps are performed without realizing it, and so you will get the consistency necessary to get better results.

Trainer: Merly Klingstor
Duration: 1h
Level: intermediate


01:42
impuls
Profile

Username	annielarson
Email	annie.larson@gmail.com
Name	Anni Larsonette
Phone	56484940
Date of Birth	19/02/1991

Change password
Covid sertificate
Logout

Figure 28: Prototype feature: Training overview   Figure 29: Prototype feature: Covid certificate upload

In the fourth iteration were implemented workout detailed overview page, fixed previous issues that were noticed during the testing phase. In the end, there was an opportunity to add one feature from the low priority list - covid certificate upload. Workout overview page can contain more pictures to make it more interesting and attractive to read. Otherwise, the feedback did not include any major issues.

Conclusion

As a result, all planned work with high and medium priority was completed within 4 iterations. Moreover, each iteration's backlog was adjusted to include tasks for improvements based on the collected feedback from user testing. Work was prioritized accordingly.

4. Results and Analysis

This chapter presents results and analysis of the outputs of chosen methods. Also, provided suggestions on how those solutions can be improved.

4.1. PRD approach

The main problems that were noticed during the preparation of PRDs:
Difficulty to identify potential risks and affected systems. The full picture of the system is missing. Dependencies are not visible as each feature is considered separately.
Priorities. Work in sprints/iterations is divided by features. However, sometimes it makes sense to develop small pieces of each feature instead of completely releasing only one functionality.
Wrong estimations might be done due to absence of the whole picture.
Also, during the implementation it was noticed that not enough attention was paid to non-functional requirements. The product requirement template can be adjust to include one more additional section - non-functional requirements. In this case, it will force to think about performance, maintainability, usability, security and other areas.
PRDs do not solve the problem of lack of shared understanding between teams. PRDs provide more information and more context around the feature, however, it still does not guarantee the shared understanding of what and how should be built.
If engineering and product teams want to follow this approach, they still need to add more practices around this process. For example, the improved versions of this process might look like:

- Product team does market and customer analysis.
- Product team prepares PRDs without technical details.
- Product team works together with the design (UI/UX team) and thinks about initial designs and workflows.
- Product team updates PRDs to include designs and provides more information about workflows.
- Product team shares PRD with the engineering team.
- Engineering team gets familiar with PRD and prepares high level questions to the design/product team.

- Product team arranges workshop sessions for the engineering team. Design team can also be involved. All parties gather together and discuss the feature, think about possible risks, dependencies, and build a shared understanding.
- After a workshop with the product team, the engineering team brainstorms together on how to divide requirements to technical tasks, think about dependencies and potential risks and after that plan the work into the sprints.
- All teams (product, design and engineering) are aligned with each other. After each sprint, the engineering team does a demo of the functionality and design and product team can review and adjust functionality.

4.2. User story mapping approach

User stories and user story mapping approaches are more focused on the end-users and their needs. Potential users of the application were interviewed, and stories were written based on this. User surveys helped to identify the main customer's needs and helped to illustrate the main scope. User stories helped to actually understand the problem that the mobile application is supposed to solve. The story mapping technique showed a great result on the prioritization part. It was clearly visible what items should be taken into development as the must-have functionality and what things can be descoped. As a result - the scope was clear, and the work was divided into logical parts. All planned work was fitted into iterations well and delivered on time. After each iteration was made a user testing round, great and useful feedback was gathered, and the prototype was improved accordingly. In the end, it was received a clickable prototype, usability was one of the main focuses - the mobile application is easy to navigate and intuitive in usage.
This method has a lot of advantages, however, it also has some disadvantages. Some of them can be avoided or be minor, but some of them are quite risky to have for specific types of products.
During the work it was noticed that having only user stories is not enough. Story just describes the customer's needs, desires, but does not cover a lot of details. In other words, user stories are incomplete requirements. To achieve better results - they should be followed up with some additional method/technique. In that case - user story maps are a great tool to cover more technical aspects. Maps cover technical tasks, show

dependencies and help to reduce risks. They also can be used as a documentation and reference tool in the future, while just having user stories will not have a lot of value. The second important moment that requires attention is the level of the details/ functionality range that should cover/tell the user story. It should not be very detailed and look like a precise requirement, and it also should not be too broad. In both cases it will be difficult to come up with the correct map based on the given story. It was one of the challenges during this research paper and applying this method in practice. Some of the user stories became very detailed and it was difficult to come with the user story map for it - because the story itself was looking as the requirement. Some other stories were too broad and required some specification, otherwise, user story maps can go in the totally wrong direction and do not cover the important functionality. As a result, It can become more challenging to estimate the work correctly and identify the priorities.

Using this approach, it is possible to lose focus and not cover the non-functional requirements, such as performance, usability, maintainability, and others. If the product requirement document can be adjusted to have a separate section for non-functional requirements, then it is a high risk to miss these aspects using the mapping technique. Some feedback is possible to receive during user testing and feedback sessions after each iteration, but it will only cover usability and some parts of the performance. Other points can be easily skipped or missed and as a result, the engineering team can produce more technical debts and face re-work needs in the future.

Another problem that was discovered during this implementation is that not always relying on user needs and desires is the correct way of building the product. Additional market and situation analyzes are necessary and cannot be replaced with user surveys and interviews. For example, covid certificate upload was the relevant functionality to build at the time of the implementation process. However, the situation in the world tends to change rapidly, and all these factors should be taken into account and analyzed as risks. Right now Estonian government has cancelled all restrictions, it is possible to attend public events and places without covid certification. It means that this functionality (certificate upload) will become redundant. This case should have been caught in advance and analyzed in the first place, and not included blindly in the first phase of MVP.

In terms of the context and shared understanding - user story maps still do not cover this full manner. To achieve a better result and build a shared understanding between all involved teams, all engineers should be involved in user story map constructions. For example, all user stories can be collected by product team members (it can be the product manager, project owner, or product analyst - depending on organization structure). This responsible person can also perform analyzes and build usage sequence vs criticality models. And as the next step, all involved engineering teams, design team members, and product owners should build user stories maps together. Each story should be discussed and everyone in the room should understand the actual user need. As the next step - engineers start to build the story maps, identify the main requirements and tasks, potential risks, and dependencies. When the full map is ready - engineers should present it to the product owner and then she will be able to make prioritization and draw the swimming line.

4.3. Comparison PRD vs User story mapping technique

Considering criteria's for "successful" MVP that were described in the background and main concept chapter, we can compare two approaches and draw the following conclusions:

- Scope and Schedule

If prioritization is done correctly, both approaches can work to achieve the desired result. It depends on the level of details that were covered using the prioritization process. From practical experience that was received during this research, it can be concluded that story mapping is a more powerful tool to show the full picture of desired functionality which allows the product team makes a proper decision on how functionality can be scoped or descoped. As it was mentioned in previous chapters, different development methods can suit different applications/ products, based on the final goal, market, and problem. In the case study that was selected for this research - user stories and mapping techniques showed better results in comparison with the PRD approach. User stories helped to identify the actual problems and customer needs. Some areas were missed during the initial PRD setup or vice versa - some functionality was described in PRDs that was not actually the main area of interest for potential users as shown in user surveys and interviews.

For example, in PRD approach was identified three personas: club member, trainer and club administrator. During user surveys, was found out that club administrators do not require mobile application to achieve their daily work goals. They use totally different system to monitor and manage club activities. There was no need to even think what functionality should support mobile application for administrator role in phase one. PRD approach did not show this, and extra resources might be wasted for building not required/valuable functionality.

Also, invoice management and payment functionality were mentioned in initial plan using product requirement documents. However, during analysis of user surveys' results and building user story maps - it was clear that it is not the priority for MPV first phase implementation.

- Customer satisfaction

The level of customer satisfaction would be the highest using user stories and a constant cycle of prototype testing. It is user stories that are aimed at identifying the direct needs of users and problems (and, accordingly, solutions) that would be the main focus in planning and implementation phases.

- Team satisfaction

Work can be planned correctly using both approaches. In the case of this research, both methods showed that it is possible to achieve results in the initial planned timeline. Also, both methods can be applied in the wrong way and cause confusion between understanding the actual picture. Approaches how to adjust both methods were suggested to achieve better understanding and engagement from all involved teams. To build a common understanding, in both cases it is necessary to arrange workshops, open brainstorming sessions and discuss all points involving all teams that will be the part of this project.

- Company profit

If provided product requirement documents were taken as the basis for building this specific software, then the result would be a product that does not cover all the needs of our customers and contains a lot of extra functionality that does not solve the actual problem. A lot of resources would be spent on the functionality that would not be used by users. This time that was spent on development could have been spent on building more important functionality. Any resource is the company's money. Accordingly, the company profit would be minimal and there is even a risk of money loss.

- Quality

Having only a working prototype and a theoretical result from PRD approach, it is quite difficult to make a final quality assessment. It can be concluded that such an aspect as usability - would definitely be higher quality using the method of user stories and constant feedback received from end users as part of testing after each iteration of development. Many nuances were noticed that were not obvious at the initial stages of development but were revised and finalized in subsequent iterations. However, as it was mentioned in the previous chapters, user stories and mapping techniques do not cover a lot of technical aspects and non-technical requirements that play a huge role in quality assurance. If PRD can be adjusted to include the non-functional requirements section, the results on quality assurance can be much higher using this approach. Alternatively, each technical task, that was identified during the building of a user story map, should be separately analyzed by engineers for acceptable technical risks and debts.

To draw the conclusion, it could be said that both methods have advantages and disadvantages and have a place to be. However, it is important to understand what product you are building, what teams are involved and what are the specifics of those teams, what goals your company wants to achieve. All this will help to identify the correct approach. But it is not enough just selecting the method that will suit them best, the main focus should be on the way how this method would be applied. Some suggestions for improvements and adjustments were provided in the previous chapters.

Summary

The main purpose of this research was to compare two different approaches that can be used for planning and prioritization phases of building a Minimal Viable Product. The main goal was to understand what the main criteria are to achieve a high customer satisfaction level, appropriate quality, and profit for the company. For this research were selected two methods - the User Story Mapping technique and the requirement-based approach - PRD. The main problem was to understand how these techniques can be implemented and applied in a real-life case scenario - the mobile application for the Estonian sport club: Impuls, to achieve desired results - efficient product development, customers are satisfied, and costs are minimal. Advantages and disadvantages were identified, and best practices for improving each method were proposed. Applying these approaches to practise, the author concluded that user story mapping brought more positive output rather than the PRD approach. User stories are a user-centric approach, with a focus on actual user needs based on constant feedback loops. The story mapping technique also showed a positive result - it helps to illustrate the full picture and can be a powerful tool for identifying dependencies/risks and making the prioritization. However, this approach is mostly suitable for small user-centric applications/startup companies. Sometimes it is not possible to avoid requirements. The Product Requirement Document method also showed a positive outcome - it can be used for identifying and documenting technical aspects, with a focus on non-functional requirements - that will help to achieve a better quality of the product.

Regardless of which approach was selected for implementation, it is very important to build shared understanding. Author identified the main gaps and also provided suggestions/ best practices on how to achieve a common understanding between all involved parties for each method.

To draw the conclusion, it can be said that both methods have advantages and disadvantages, and it is very critical to understand what product you are building (the context, final goal, analyze market needs and situation), understand who your clients and final users are, teams involved in the implementation process, and what goals your company wants to achieve. These analyzes should be done first before choosing the appropriate method. It will help to identify which approach to choose next. Based on this, any of the methods provided above can help to achieve desired results.

Kokkuvõte

Selle uurimistöö peamiseks eesmärgiks oli võrrelda kahte erinevat lähenemisviisi, mida saab kasutada MVP plaanimise ja loomise etappidel. Peamiseks ülesandeks oli saada arusaam põhilisest kriteeriumist, mis tagab kõrget kliendi rahuolu, sobivat kvaliteeti ja kasumlikust ettevõttele. Selle uurimstöö läbiviimiseks oli valitud kaks lähenemisviisi - User Story Mapping tehnika ja requirement-based approach PRD. Põhiline probleem oli aru saada, kuidas neid tehnikaid saab rakendada päriselu stsenaariumis, luues mobiilrakendust Eesti spordiklubile Impuls, saavutades järgmisi tulemusi: tõhus tootearendus, kõrge kliendirahuolu ja kõik eelmainitud minimaalsete kuludega. Töö käigus olid tuvastatud meetodite puudused ja eelised. Oli pakkutud iga meetodi parandused. Rakendades neid tehikaid praktikas, töö autor jõudis järeldusele, et user story mapping sai rohkem positiivseid väljundeid, kui PRD lähenemisviis. User stories on kasutajapõhine lähenemisviis, mis põhineb kasutaja põhivajadustel. Vajadusi kaardistatakse pideva kasutaja küsitluse käigus ja kasutaja poolt antud tagasisidel. Story mapping tehnika näitas ka väga head tulemust. Selle abil saab illustreerida täieliku pildi toimuvast, millest kujuneb välja võimas tööriist riskide ja sõltuvuste tuvastamisel ning otsuste langetamisel. Kuid see lähenemine on enamasti sobilik kasutajale suunatud väikestele rakendustele või start-up ettevõtetele. Mõnikord ei ole võimalik nõudeid vältida. PRD meetod näitas ka positiivset tulemust. Seda lähenemist saab kasutada tehniliste aspektide tuvastamiseks ning dokumenteerimiseks, fokuseerides mittefuntsionaalsetel nõuetel, mis omakord aitab saavutada paremat toote kvaliteeti. Olenemata sellest, milline lähenemine oli valitud rakendamiseks, on väga oluline luua ühine arusaam. Autor tuvastas põhilisi lünkasid ja tegi ettepanekuid kuidas saavutada arusaam iga kaasatud osapoole vahel iga meetodi puhul.

Järelduseks võib öelda, et mõlemad meetodid omavad nii elliseid kui puudusi ja tuleb väga kriitiliselt hinnata toodet mida plaanis luua (sisu, eesmärk, analüüsida turu vajadust ja situatsiooni), tagada arusaam oma lõppkasutajast ja kliendist, millised osapooled on rakendamisprotsessi kaasatud ja millised eesmärke soovib ettevõtte saavutada. Eelanalüüs peaks olema tehtud enne lähenemistehnika valikut. See aitab tehnika valikul. Põhinedes eelmainitule saab iga meetod olla abiks soovitud tulemuse saavutamiseks.

References

Books

- [5] "User Story Mapping: Discover the Whole Story, Build the Right Product"
  Book by Jeff Patton
  Released September 2014
  Publisher(s): O'Reilly Media, Inc.

- [6] “Extreme Programming Installed”
  Book by Ron Jeffries, Ann Anderson, Chet Hendrickson
  Released October 2000
  Publisher(s): Addison-Wesley Professional

- "Learning Agile: Understanding Scrum, XP, Lean, and Kanban"
  Book by Andrew Stellman and Jill Alison Hart
  Released November 2014
  Publisher(s): O'Reilly Media, Inc.

Web Pages

- [1] Indicator of Startup Failure
  https://stumejournals.com/journals/i4/2017/5/238.full.pdf
- [2] Case Study Process https://www.digitalmethods.net/MoM/CaseStudy
- [3] When NASA Lost a Spacecraft Due to a Metric Math Mistake
  https://www.simscale.com/blog/2017/12/nasa-mars-climate-orbiter-metric/
- [4] User Story Mapping https://builtin.com/product-management/user-story-mapping
- [8] Story Map Concepts http://www.jpattonassociates.com/wp-content/uploads/2015/03/story_mapping.pdf
- [9] How you slice it https://www.jpattonassociates.com/wp-content/uploads/2015/01/how_you_slice_it.pdf
- [10] Proto.io application https://proto.io/
- Agile Manifesto https://agilemanifesto.org/
- Scrum Org https://www.scrum.org
- Product Requirements Document
  https://www.productplan.com/glossary/product-requirements-document/
- Product Requirements Document
  https://en.wikipedia.org/wiki/Product_requirements_document
- How to Write an Effective Product Requirements Document (PRD)
  https://fireart.studio/blog/how-to-write-an-effective-product-requirements-document-prd/

- Why Google Glass Failed and Biggest marketing lessons you can learn from it https://startuptalky.com/google-glass-failure-case-study/
- Failure of Apple Newton https://www.slideshare.net/nijansh/imen301-hw2
- Google Glass: A Case Study https://onlinelibrary.wiley.com/doi/abs/10.1002/pfi.21919
- Case study: The Design Process https://uxplanet.org/case-study-the-design-process-670dd3af4372
- Is the Product Requirements Document Dead? A Debate https://www.uservoice.com/blog/is-the-product-requirements-document-dead
- Top 5 takeaways from User Story Mapping https://medium.com/@mal.sanders/top-5-takeaways-from-user-story-mapping-by-jeff-patton-f8c80cf73750
- How Uber, Airbnb & Dropbox Released MVPs to Achieve Rapid Growth https://medium.com/@LoganTjm/how-uber-airbnb-dropbox-released-mvps-to-achieve-rapid-growth-d823ac6eaed5

Blog Posts

- [7] "From User Story Mapping to High-Level Release Plan"
  July 31, 2017
  By Basia Coulter & Charlotte Fouque & Daryl Katz Riethof
  https://www.caktusgroup.com/blog/2017/07/31/user-story-mapping-high-level-release-plan

Appendix 1 – Non-exclusive licence for reproduction and publication of a graduation thesis[1]

[1] The non-exclusive licence is not valid during the validity of access restriction indicated in the student's application for restriction on access to the graduation thesis that has been signed by the school's dean, except in case of the university's right to reproduce the thesis for preservation purposes only. If a graduation thesis is based on the joint creative activity of two or more persons and the co-author(s) has/have not granted, by the set deadline, the student defending his/her graduation thesis consent to reproduce and publish the graduation thesis in compliance with clauses 1.1 and 1.2 of the non-exclusive licence, the non-exclusive license shall not be valid for the period.