

Henrik August Burmann  
Marcus Johannessen

# Fullstack application for people in need looking for food

Bacheloroppgave i Dataingeniør  
Veileder: Donn Morrison  
Mai 2023



Henrik August Burmann  
Marcus Johannessen

# **Fullstack application for people in need looking for food**

Bacheloroppgave i Dataingeniør  
Veileder: Donn Morrison  
Mai 2023

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



Kunnskap for en bedre verden





## Abstract

This report provides an overview of the development process involved in creating a minimal viable product full-stack application for Matsentralen Trøndelag, aimed at enhancing the accessibility of food assistance for individuals in need. The primary objective behind the application was to make the services of non-profit organizations more available to the public. Additionally, the application aimed to streamline the administration of organization data.

The developed application encompasses three distinct interfaces or portals, each serving specific functions. Firstly, the user interface enables individuals to search for and identify various non-profit organization by employing search and filtering capabilities based on divers criteria.

Secondly, the organization portal empowers organizations to register themselves by providing relevant information, ensuring accurate and up-to-date data within the system.

Lastly, the regional portal caters to the different Matsentralen branches located across Norway, serving as a central hub for viewing requests from organizations seeking registration as partner organizations. This portal facilitates the management of organizations details, allowing for editing of information and granting the ability to accept or decline registration requests.

In the development of the application, the team utilized modern frameworks and technology standards to ensure the attainment of the desired outcomes outlined by the clients at Matsentralen. Notably, Spring Boot with Gradle, Next.Js, and OAuth 2.0 were instrumental in enabling the creation of a robust and feature-rich application that aligned with the specified requirements.

The achieved result is a minimal viable product that effectively lets the user look for organizations providing food help near them, fitting the user's needs. The application successfully streamlines the process of storing organization data, which has the potential to serve as a solution to Matsentralen's scaling problem.

## Sammendrag

Denne rapporten presenterer en oversikt over utviklingsprosessen bak utviklingen av et enklest brukbart produkt for Matsentralen Trøndelag. Denne applikasjonen hadde som mål å gjøre mat-tjenester mer tilgjengelige for mennesker som av ulike årsaker ikke har råd til mat. I tillegg var et mål med applikasjonen å effektivisere administreringen av organisasjonsdata.

Den utviklede applikasjonen består av tre ulike deler, hvor hver del har en spesifikk funksjon. Den første retter seg mot mennesker som trenger mathjelp. Her kan brukere søke etter og filtrere ideelle organisasjoner basert på ulike kriterier som har et samarbeid med Matsentralen.

Den andre delen lar organisasjoner søke om samarbeid med Matsentralen og vedlikeholde informasjon om organisasjonen etter godkjenning.

Den siste delen er rettet mot de ansatte i de ulike seksjonene av Matsentralen lokalisert rundt i Norge. Denne fungerer som en plattform for Matsentralen til å behandle søknader og ha en ryddig oversikt over organisasjoner i deres seksjon.

I utviklingen av applikasjonen benyttet gruppen seg av moderne rammeverk og teknologistandarder for å sikre at kravene satt av Matsentralen ble tilfredsstilt. Viktige teknologier brukt er Spring Boot med Gradle og OAuth 2.0. Disse teknologiene var essensielle for å kunne bygge en robust og funksjonsrik applikasjon.

Sluttresultatet er et enklest brukbart produkt som effektivt lar brukeren finne ideelle organisasjoner som tilbyr mat-hjelp og oppfyller brukerens behov. Applikasjonen effektiviserer prosessen med lagring av organisasjonsdata og har potensial til å være en løsning på Matsentralens problemer knyttet til organisasjonens vekst ved videre utvikling.

## Preface

This report is the result of the system development project for the final bachelor thesis of the study Software Engineering at the department of Computer Science at the Norwegian University of Science and Technology in Trondheim. The project is an assignment provided by Matsentralen Trøndelag, a non-profit organization, providing food aid to people in need. The students chose the assignment with a wish of developing a full-stack system. The process of working with this assignment has given the team valuable insight into system development and communication with client, to provide a system that meets their need. Further, it has provided the team with challenges that the team had no prior experience of. We would like to thank Jort Reitsman and Marte Bjørnsund for providing valuable feedback during the project period and for inviting us to their headquarters in Trondheim. Finally we would like to thank our supervisor Donn Morrison for the feedback considering questions and problems during our project.

# Assignment

## Original assignment

Matsentralen is a non-profit organisation that collects surplus food from grocery suppliers and donates it to charity organisations who help people in need. We are a national network of eight food banks. Our mission is to reduce food waste and simultaneously help people in need in our society.

The aim of this project is to develop an intuitive web application to connect people in need with charitable organisations that fit their situation. Additionally, the application should have a management back-end to allow easy maintenance by Matsentralen.

A successful design will help people in need get access to food more efficiently, as well as reduce the resources Matsentralen uses to achieve this with the tool we currently use. The current tool can be found here: <https://www.matsentralen.no/matkartet>

## Clarifications

The final assignment is very similar to the original assignment text. However, the original text is somewhat vague, mainly because of some misconceptions on Matsentralen's side, which have been unraveled during meetings.

The confusion mainly comes from a misunderstanding over the meaning of the terms backend and frontend. What Matsentralen is asking for is rather a "admin portal" for administration of organizations and information. A functioning backend is of course still necessary for the application to work, but not in the way asked for by Matsentralen.

## Abbreviations

- API - Application Programming Interface
- REST - Representational Programming Interface
- URI - Uniform Resource Identifier
- MVP - Minimal Viable Product
- WCAG - Web Content Accessibility Guidelines
- ICT - Information and Communications Technology
- CI/CD - Continuous Integration and Continuous Delivery
- DVCS - Distributed Version Control Systems

# Contents

<b>1</b>	<b>Introduction of Matsentralen and Problem Statement</b>	<b>1</b>
1.1	What is Matsentralen . . . . .	1
1.2	Current digital solution . . . . .	2
1.3	Matsentralen's desired solution . . . . .	2
1.4	Problem Statement . . . . .	3
1.5	Thesis structure . . . . .	3
<b>2</b>	<b>Theory</b>	<b>5</b>
2.1	Working Methods . . . . .	5
2.1.1	Scrum . . . . .	5
2.1.2	Wireframes . . . . .	5
2.1.3	Universal Design . . . . .	5
2.1.4	WCAG . . . . .	6
2.1.5	MVP . . . . .	6
2.2	Technological Concepts . . . . .	6
2.2.1	REST . . . . .	6
2.2.2	Dependency Injection . . . . .	6
2.2.3	Server Side Rendering . . . . .	7
2.2.4	Continuous Integration . . . . .	7
2.2.5	Version Control . . . . .	7
<b>3</b>	<b>Method and Choice of Technology</b>	<b>9</b>
3.1	Research Method . . . . .	9
3.2	Development Method . . . . .	10
3.2.1	Agile Development with based on Scrum . . . . .	10
3.2.2	Wireframes . . . . .	10
3.2.3	Wireframes - People in need of food help . . . . .	10
3.2.4	Wireframes - Organizations and Admins . . . . .	10
3.2.5	Development - Sprints . . . . .	10
3.2.6	Version Control during development . . . . .	11
3.3	Choice of Technology . . . . .	11
3.3.1	Back-end . . . . .	11
3.3.2	Front-end . . . . .	13
<b>4</b>	<b>Results</b>	<b>15</b>
4.1	Scientific Results . . . . .	15
4.1.1	Wireframes . . . . .	15
4.1.2	The finished application . . . . .	18
4.2	Engineering results . . . . .	29
4.2.1	Functional features . . . . .	29
4.2.2	Non-functional features . . . . .	30
4.3	Administrative Results . . . . .	31
4.3.1	Planning . . . . .	31
4.3.2	Time Management . . . . .	32
<b>5</b>	<b>Discussion</b>	<b>33</b>
5.1	Scientific Results . . . . .	33
5.1.1	Wireframes . . . . .	33
5.1.2	Application Views . . . . .	33
5.1.3	Validation of fields . . . . .	34

5.1.4	Documentation of code	34
5.2	Engineering Results	34
5.2.1	Functional features	34
5.2.2	Non-functional features	35
5.2.3	The setup of the project	35
5.2.4	Choices of technology	36
5.3	Administrative Results	36
5.3.1	Planning	36
5.3.2	Time Management	36
<b>6</b>	<b>Conclusion and further work</b>	<b>38</b>
6.1	Conclusion	38
6.2	Further work	38
6.2.1	Multiple Languages	38
6.2.2	Allowing users to search for cities and district	38
6.2.3	Donation Map	38
6.2.4	Divide the application in multiple application	39
6.3	Attachment B - Pre-work plan	42

## List of Figures

1	Example Matkartet	2
2	Chosen method of research	9
3	Sprint development	11
4	Wireframe - Startpage	15
5	Wireframe - Organization overview	15
6	Wireframe - Organization registration	16
7	Wireframe - Organization info views	16
8	Wireframe - Organization Matkartet and Donasjonskartet	16
9	Wireframe - Admin - Organizationlist	16
10	Wireframe - Admin organization details	17
11	Wireframe - Admin- Application List	17
12	Wireframe - Admin - Application	17
13	Product - User - Start page	18
14	Product - User - Start page mobile	18
15	Product - User - Organization Detail page	19
16	Product - User - Organization Detail page mobile	19
17	Product - Log in	20
18	Product - Organization - First Time Application	20
19	Product - Organization - Second Time Application	21
20	Product - Organization - Home View	21
21	Product - Organization - Edit home page data	22
22	Product - Organization - Service View	22
23	Product - Organization - Service Edit	23
24	Product - Organization - Service Edit	23
25	Product - Organization - Service Edit	23
26	Product - Organization - Donation View	24
27	Product - Organization - Donation Edit	24
28	Product - Admin - Overview organizations	25
29	Product - Admin - Application overview	25
30	Product - Admin - First Time Application View	26

31	Product - Admin - First Time Application Edit . . . . .	26
32	Product - Admin - Second Time Application View . . . . .	27
33	Product - Admin - Second Time Application Edit . . . . .	27
34	Product - Admin - Add and delete shared fields . . . . .	28
35	Hourly graph of hours worked . . . . .	32
36	Hours of work for each category . . . . .	32

## List of Tables

1	Functional features . . . . .	30
---	-------------------------------	----



# 1 Introduction of Matsentralen and Problem Statement

## 1.1 What is Matsentralen

Matsentralen is an umbrella organization for eight food banks spread across Norway. The food banks receive surplus food and beverage from partners in the food industry, that for different reasons cannot be sold in store, but is still of high quality and safe to consume. Matsentralen redistributes these products to non-profit organizations in their area. The non-profits use these products to help people in need.

Matsentralen has a big user base, consisting of many different types of people in different life situations. In common is that they are in need of food help. The users of Matsentralen's services are among many others, people who struggle with alcohol or drug addictions, poor families, people reintegrating to society after serving a sentence and asylum seekers. Matsentralen's partner organizations often have distinct target groups they focus on helping and different ways of providing the food.

Matsentralen has partnerships with over 480 non-profit organizations. The group of partners contains both organizations belonging to big national non-profit organizations such as Frelsesarmeen and Røde Kors and smaller independent organizations such as Omsorgskaffeen i Trondheim. [15].

There are three main ways the surplus food is used by the partner organizations: home delivery of food, distribution of groceries and serving of meals. The organizations themselves choose how they use the food.

Matsentralen has many partners in the food industry. This includes grocery wholesalers as Rema 1000 and Norgesgruppen, and others like Bama and Tine [10]. The food provided from these partners is as mentioned surplus food that for various reasons cannot be sold in stores, but is still of high quality. This could be products that have the wrong packaging or are seasonal products. The products also often are a result of overproduction, where the businesses would not be able to sell all of the products before the expiration date.

By distributing food that otherwise would have gone to waste, to people in need, Matsentralen achieves their two main goals: To reduce hunger and poverty and to reduce food waste. In 2018 Matsentralen won prizes for both cases [12].

Overall Matsentralen contributes to 7 of the 17 of the UN's Sustainable Development Goals. [11] These are:

- 1. No poverty.
- 2. No hunger.
- 3. Good health and well being.
- 10. Reduced inequalities.
- 11. Sustainable cities and communities.
- 12. Responsible consumption and production
- 17. Partnership for the goals.

## 1.2 Current digital solution

Matkartet is Matsentralen's current tool to display all their partner organizations for people in need looking to find food. It is a custom Google map, where all Matsentralen's partner organizations are manually plotted in. When a point is clicked, information about that organization appears on the left side of the screen. Below is an example taken from Matkartet.

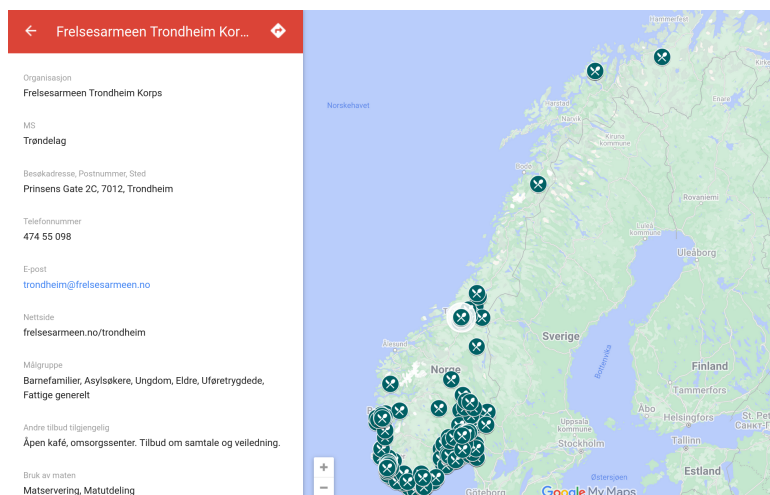


Figure 1: Example Matkartet

As for storage of data, each section of Matsentralen is responsible for storing and maintaining the information of their own local partners. Several organizations, including Matsentralen Trondheim, does this through an spreadsheet document, operating as a database, that has to be manually updated by the employees of Matsentralen.

As Matsentralen has grown a lot the last years, today's solution leads to some challenges. With every organization having to be manually plotted into Matkartet it creates a lot overhead for the employees of Matsentralen when new organizations become partners or information has to be updated.

Further, Matkartet itself is not easy to use for a lot of users. After navigating to Matkartet users are presented with a large map without any information except for the plots of each organization. There is no way for the user to filter on target groups nor the form of food help the organizations provides. Matsentralen spends a lot of time answering phone calls and emails from people in need looking for food help - time that could be spent on further developing Matsentralen. This could be improved with a better tool.

Since Matsentralen's current way of storing data is a static spreadsheet document, more overhead arises for the Matsentralen employees. If an organization for example wants to update opening hours or way of food usage, they have to contact Matsentralen and have them manually update the spreadsheet document. A lot of time could be saved if the organizations themselves could update their own information.

## 1.3 Matsentralen's desired solution

To face the challenges discussed above Matsentralen wanted a solution that would make it easier for people in need of food help to find organizations that fit their needs, as well to reduce some manual labor for the employees at Matsentralen.

To satisfy these needs a fullstack web application was needed. A database was also needed to replace the spreadsheet documents. There was no architecture to work build on, and everything therefore had to be built from scratch.

The application will have three different types of users: people looking for food help, the partner organizations and the employees of Matsentralen. Therefore the application needed to be split in three. The main points of each part is described below. More detailed requirement specifications are described in the system requirements.

- The first mentioned part of the application could be seen as a 1-to-1 replacement for Matkartet. Here Matsentralen wanted an interface where it was easier for users to browse through and search for different organizations. The users should also have the opportunity to filter organizations based on different criteria such as target groups and location.
- Further, there was need for the organizations themselves to have their own part of the application. The main requirements for this part was the organizations themselves should be able to apply for a partnership with Matsentralen through an application and have the ability to view and update their own information. They should get to choose what information they want to display for users and what only Matsentralen should see. A successful implantation of this would reduce a lot of the time Matsentralen spends on keeping track of all new data.
- The employees of Matsentralen themselves also needed their own portal. Here the wish was that the employees of Matsentralen could be administrators of their own section. They needed to have the ability to review and edit applications from organizations, view and edit all organizations in their sections and export wanted organization information to an spreadsheet document.

## 1.4 Problem Statement

Given the challenges and needs described earlier in the chapter the problem statement has evolved to be:

"How can a minimal viable product, full-stack application, be developed to help people in need find food help, while fixing the scaling problem of Matsentralen"

## 1.5 Thesis structure

The report is structured as follows:

### **Theory:**

In this section relevant theory required to understand the theoretical background of the project.

### **Method and Choice of Technology**

This section presents the research method chosen, work methodology used and choices of technology taken.

### **Results**

In this section the results of the project are presented. The results are split in three: scientific, engineering and administrative results

### **Discussion**

This section discussed the result presented

### **Conclusion and further work**

In the conclusion the project is summarized and further work is discussed.

**Societal Influence** At the end the application's societal impact is discussed.

## 2 Theory

During the development of a fullstack application there are many concepts, both technical and non-technical a team will have to use. In this chapter concepts essential to the team's work will be described.

### 2.1 Working Methods

This section will cover non-technical concepts of the process described, as far as possible, in a chronological order.

#### 2.1.1 Scrum

In the beginning of the project a team needs to agree on a how they want to work and therefore choose a development framework. A popular framework used is Scrum. Scrum is a framework for an iterative, agile work process[21]. The idea of Scrum is to split the working period into cycles called sprints, with every sprint having a set amount of tasks to finish.

A scrum project often involves daily morning meetings, called stand-up meeting. In these meetings the team members explain what they have done since the last meeting, and what they plan to do until the next meeting. In larger teams, stand-up meetings are especially useful to ensure that all team members have good overview of the entire project, while avoiding wasting time by having multiple people unknowingly working on the same task.

After each sprint there is often a larger meeting called a Sprint Retrospective. In this meeting the team reflects on the how the last sprint went and how they can improve in future sprints. The meeting is also used to plan what is to be done during the next sprint.

#### 2.1.2 Wireframes

Early in a project it is important for a software engineer to understand what the client envisions. A client often has a lot of ideas and dreams of how a product should look like and, without necessarily being able to describe them in detail. Therefore Wireframes are common to use by software developers in the start phase of a project.

Wireframes are visual, 2D illustrations of how a web application or app can look. This way developers can show their client how they have interpreted the client's wishes and get feedback. Wireframes are non-functional, meaning it is easy to change direction if the client does not like what the developers have made.

When working in larger teams, where there are both designers and developers, wireframes are a good tool for communicating how the product should function and look.

Fidelity varies in wireframes. Wireframes can have low, medium- and high-fidelity. [19]. The level of fidelity in this context defines to what degree the wireframes are an exact representation of how the product should look. Low-fidelity wireframes have a minimal amount of color and design, leaving the developer to interpret the wireframes. While high-fidelity wireframes will look very similar to the developed website.

#### 2.1.3 Universal Design

When developing a product of any kind, one must consider universal design. Universal design is a collective term used for designing a product in a way that it can be used by as many people as

possible, considering different disabilities [23]. Examples of such disabilities can be blindness and loss of lower extremity motor control.

#### **2.1.4 WCAG**

Universal design includes the design of websites. Norwegian law states that both public and private organizations must follow universal design for ICT solutions [9]. To help developers comply with universal design there are a set of guidelines called WCAG. In Norway as per February 1st, the public sector must follow 49 of the 78 of the WCAG 2.1 guidelines [3]. For the private sector that number is 35. [17]

Important rules to remember when developing a website are having clear colour contrasts, not having too small of a text size and present the contents in a meaningful order [17]

#### **2.1.5 MVP**

MVP (Minimal Viable Product) is, similarly to wireframes, useful early in development to get an indication of if a product idea is going to work or if product is working as a client wants [13]. An MVP has the features needed to fulfill the basic requirements for the product, but a lot of extra features still is to be implemented. This gives the team an opportunity to get early feedback, and change direction if the feedback is negative.

## **2.2 Technological Concepts**

This section covers several technical concepts used during the project.

### **2.2.1 REST**

REST, first introduced in 2000 by Dr. Roy Fielding, is a set of architectural design rules used for developing web services [18]. In a REST project the client only needs to know the URI for the resource needed to request a resource. Therefore, the server and the client are independent of each other, making it possible to develop the applications separately. The client and server should not be able to in any way modify or change the other.

With a HTTP request the client can add HTTP verbs, specifying what it wished to do, where the most common are the GET, POST, PUT, and DELETE verbs. These are used for getting requesting resources, posting a new resource, changing a resource and deleting a resource respectively. With the HTTP request body, headers and parameters are often added to provide additional information of what it wants the server to do or return.

Since the client and server are independent, all communication must be stateless. This means that a new request has no memory of the the last request, and all needed information needs to be passed in each request.

Using REST APIs are very popular currently for flexibility and the possibility to develop applications independently. [18]

### **2.2.2 Dependency Injection**

Dependency injection is a design pattern in object orientated programming where the goal is to achieve loose coupling in a program and/or make a class independent of its dependencies, resulting in reusable and testable code [20]. There are four main points essential to dependency injection:

- A service class: This is a standard model class, for example a class for a book with object variables.
- A client: This class uses the service to develop different functions.
- An interface: The interface works breaks the dependency between the client and the service, rather making the client dependent on the interface. The advantage with decoupling the client and services, is that changes to the service will not affect the client to the same degree.
- Last of all is the injector. The injector makes it so the service is not dependent of a specific type of interface and dependencies can be swapped easily.

Because the decoupling of dependencies between classes, dependency injection also makes it easier for developers to work on different classes separately.

### 2.2.3 Server Side Rendering

Server side rendering is a way of rendering a web page, where the HTML file is rendered on the web server, rather than the by the client. The HTML file is then sent to the web browser from the server. This way the client does not spend time waiting for JavaScript and CSS files, decreasing load time [22]. This can improve the user experience drastically.

Another advantage with server side rendering is that it helps search engines like Google index the HTML based website, making it more available for users. This is something search engines usually struggles with, where indexing for websites based on JavaScript can take a lot of time [14].

Since the page is already rendered by the server this also makes the page better to share via social media platforms, since it allows for the social media showing a preview of your page to the user. [14]

When the rendering is moved from the browser to the server this of course results in extra work for the server and may therefore be a more expensive solution [22].

### 2.2.4 Continuous Integration

Continuous Integration (CI) is the practise of that involves regularly integrating code changes from multiple developers into a shared repository. The main goal of CI is to detect and address integration issues and conflicts early in the development process. Each integration triggers an automated pipeline which may involve testing the code, building and compiling the code. The pipelines aims to ensure that integrated changes do not introduce errors or breaking functionality.

### 2.2.5 Version Control

Version Control Systems (VCS) is a fundamental practice in software development aimed at effectively managing the various iterations of software code. These systems facilitate the recording and tracking of changes made to different files within the code base, enabling developers to review previous functional versions if errors or issues arise during the implementation of new features.

The primary mechanism employed by VCS is the concept of commits, whereby each commit represents a snapshot of the code base at a specific point in time. By associating a unique identifier with each commit, VCS systems establish historical record of the software's evolution. This enables developers to refer back to earlier versions ensuring code stability [4].

Furthermore, VCS supports collaborative development by allowing developers to work concurrently on a shared code base. This is achieved through the creation of branches, which are distinct

versions of the code base that can be developed independently. These branches serve as separate lines of development and can include the implementation of specific features, bug fixes, or experimental changes. Ultimately, these branches can be merged back into the main code base, combining the changes with the latest updates.

In scenarios where multiple developers modify the same file or section of code, VCS systems are equipped to handle merge conflicts. These conflicts arise when conflicting changes need to be adapted during the merging process. VCS provides mechanisms for developers to resolve conflicts safely, ensuring that changes from different contributors are integrated seamlessly.



### 3 Method and Choice of Technology

This section is split into three subsections: research method, development method and the reasons behind our choices of technology. Combined the section covers how the team worked to fulfill the problem statement given in the introduction.

#### 3.1 Research Method

At the beginning of the project, neither team member had any knowledge of Matsentralen’s work nor structure. Therefore there was a need for research to explore the research question / problem statement created. Multiple in-person interviews with Matsentralen were conducted. In these meetings the team’s main contact at Matsentralen described, in detail, the problems they were having while the organization is growing bigger, and what they would need from the new system. This data analysis was therefore qualitative.

Even though Matsentralen Trøndelag is the client of this project, this application will be used by all eight sections, and a more quantitative approach was also taken to involve more people in the collection of data. A spreadsheet was created where all sections could fill out needs and wishes they had. This was important since the different sections face their own challenges.

As the prototype was created there was a need of a lot of in depth input from Matsentralen. Here two sets of interviews with meetings were held where a representative of Matsentralen Vestland was included in one, to get more data of higher quality. Generally data was continually collected by designing and developing and changing adaption to the responses gotten from Matsentralen.

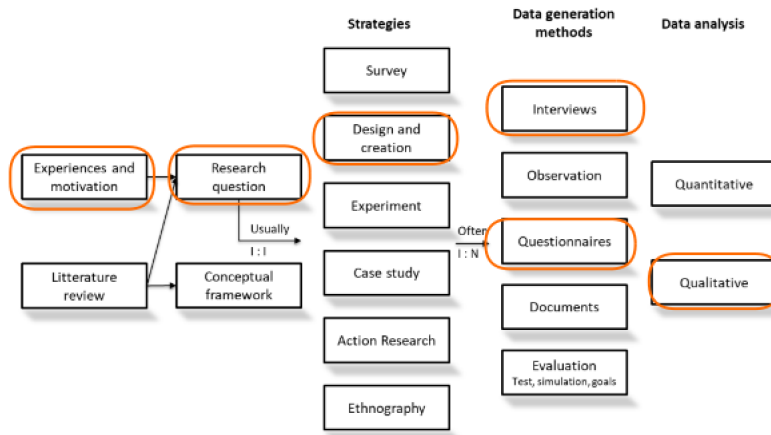


Figure 2: Illustration of chosen form of research. Template taken from compendium on Blackboard

A challenge encountered was that Matsentralen did not have a clear idea of what they specifically wanted and needed of the application. The initial ask of the project was something to replace Matkartet, as well as a solution to store and administer organizations. This research process was therefore integral for the team to gradually understand what Matsentralen needed to form the requirement specification. Understanding the the biggest strength of chosen research process.

Initially in the project there was a wish to perform user test on people not related to Matsentralen to get larger quantity of data for the part of the application. However, such user tests were not conducted because of time constraints. This is a clear weakness in the research done during the project as the application will be used by people that do not have in depth knowledge of Matsentralen.

## 3.2 Development Method

In this subsection the choices of development method while developing the minimal viable product will be described.

### 3.2.1 Agile Development with based on Scrum

Starting of the project the team needed to decide on the method to use, and went for a Scrum based method. Since the team only consists of two members, not all principles in Scrum were followed. Daily stand-up meetings were dropped as the team members communicated on a daily basis anyway.

The project as a whole was split into five sprints, stretching from the start of January until due date in May. This involved all mandatory assignments, not just the development itself. This is shown in the Gantt chart found in pre-project. The time estimation in this chart will differ from other time estimations in this document, as a result of time optimism early in during the project.

### 3.2.2 Wireframes

Before starting the development, Wireframes were needed. This was of great importance since the design was to be built completely from scratch were Matsentralen's official logo and colors where the only things pre-decided.

The tool used for the Wireframes was Figma, a very popular tool among designers. [2]. With non of the team members being experienced in the development of design, Figma's easy to use tools made it easy to make decent wireframes.

The wireframes for the application were split into the three sections discussed in 1.3. In the part for people in need of food help, the focus was slightly different compared to the latter two.

### 3.2.3 Wireframes - People in need of food help

Since many of these users will be people that often may have limited computer skills, it was important to have a design that was intuitive to use. Considerable amount of time was invested to make wireframes of medium to high fidelity. This was to ensure that a lot of information was presented without overwhelming the user. As there only were two main pages connected to this user view there were time for this.

### 3.2.4 Wireframes - Organizations and Admins

Further there were a lot more views related to the organization and admin views. In these sections the main purpose of the wireframes were to map how all the information and options were to be displayed. The detailed design were therefore less importance. Hence was a more low-medium fidelity approach was chosen for these parts.

### 3.2.5 Development - Sprints

The development of the application was split into three sprints, with the whole period stretching from March 21st to May 18th. Of early priority of was to build the essential parts of the server, with the ability to create and store organizations. Further, with the assignment emphasising that the application's goal was to help people in need find food help, that part was prioritized first on the front-end. Organizations were created using the API tool Postman to create organizations to view on the page early in the project. The rough structure of the sprints can be view in the image below.

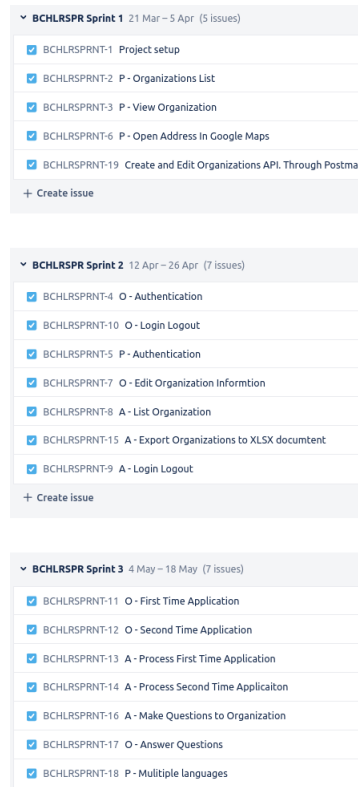


Figure 3: Sprints development: P - People in need of food help related view, O - For organization, A - For admins

### 3.2.6 Version Control during development

Throughout the developmental phase of the application, the team used Distributed Version Control Systems (DVCS) using Github to effectively manage and track distinct versions of both the back-end and front-end. Unlike traditional centralized version control systems, DVCS ensured that each team member possessed a copy of the entire code base rather than having access to fragmented portions. This allows the developers to have the same overview of project structure.

## 3.3 Choice of Technology

### 3.3.1 Back-end

#### Spring Boot

The first and most important technology based decision the team had to make, was how to develop the back-end. To have a fast as possible development tempo, a framework that allowed for developing the server and client separately was needed. The choice was therefore Spring Boot.

Spring Boot is a popular Java based framework used for developing REST APIs, as described in 2.2.1 allows applications to be developed separately since the client only needs to know the URI of the requested resource.

Spring boot offers a lot of annotations making development easier and offers extra information to the compiler of a class, variable or function. They can among other things be used to make a parameter required or not required in a function or define a path for an API endpoint. Other annotations describes a class' role in a REST API. Some of these are @Controller, @Service and @Entity. For example, will a class with the @Entity tag will if written correctly be added to the database.

Auto configuration is also offered in Spring Boot. This can be done in different ways, but in this project the annotation `@SpringBootApplication` is used[16]. Auto configuration configures many of the dependencies for the developer, which could be very complicated for the developer in a project with a lot of classpaths and dependencies.

Further Spring Boot allows for dependency injection, a big advantage with this framework. This is easily done by adding the "Autowired" annotation.

## **Gradle**

When building a Spring Boot project the developer has a choice between two build tools: Apache Maven and Gradle. Gradle was chosen for this project for two main reasons:

- In contrast to Maven, Gradle does not use a XML file to hold it's plugins and dependencies.. Both team members had a lot of experience with Maven and the POM (Project Object Model) file. As the project gets bigger, this XML file gets very hard to navigate in. Gradle's "build.gradle.kts" file achieves the same as the POM file, but in significantly fewer lines [1].
- Gradle also allows for incremental building. This means if the output of already compiled files have not changed, these files will not be compiled again [6]. This can save considerable compile time.

## **Kotlin**

Spring Boot is as mentioned a Java based framework. The team therefore had to decide which Java based programming language to use. Kotlin became the chosen language. Kotlin is very similar to Java in most ways, but has some advantages.

Kotlin implements the "data class". This is a class that auto-generates get and set methods, as well as constructors. In this project there are a lot of entity classes where only variables are defined. Therefore many fewer lines of code has had to be written. These classes are again used in the repository interfaces.

In addition to being an object oriented language, Kotlin is also a functional programming language. This also makes it easier to limit the amount of lines needed in the code base.

## **OAuth 2.0**

Concerning security OAuth 2.0 was chosen for authorization. The reasoning behind the choice was its ability to support multiple authentication protocols and grant types. Furthermore, OAuth 2.0 provides a standardized authorization flow and token-based authentication, which simplifies the management of access controls and ensures the privacy and security of user data. The protocol is also compatible with external authentication providers like Google, Facebook and Github. This allows users to authorize themselves without exposing user credentials.

## **Google Cloud**

The team opted to utilize two main features of the Google Cloud platform, specifically, Google Cloud Run and Google Cloud SQL. Google Cloud run was used for deployment and continuous integration (CI) for pipeline management. This was done to have a running backend available that was integrated with the teams main branch in Github. The CI pipeline was triggered every time there was a merge to this branch, and if successful, the backend would be updated with the latest features. Additionally, Google Cloud's support for MySQL 8 was utilized to establish a database connection for the project.

## MySQL

For the choice of a database management system MySQL was chosen. The team wanted to use SQL because of its flexibility with queries. MySQL was the chosen since it integrates well with Google Cloud.

## Cloudinary

Images were not an integral part of the application, however each organization has the opportunity to add an organization image. These images were saved using the image database Cloudinary. With Cloudinary it was easy to upload and fetch images. When uploading an image, Cloudinary returns the full URI for the image, which is saved in the SQL database. When fetching the image, the application uses the NextJs Image component and passes the component the URI for the image.

### 3.3.2 Front-end

#### NextJs

After deciding the backend stack, the team had to choose a frontend framework. The team wanted a flexible framework and both team members had experience with both Vue and some experience with React. For flexibility the team agreed on using React.

Considering React, in contrast to Vue, only being a library, the team also had the opportunity to choose a framework for React. NextJs was chosen for two main reasons for this:

- NextJs implements server side rendering, discussed in 2.2.3. Therefore NextJs could ensure faster loading time, than plain React.
- With NextJs being a framework and not just a library, it also offers pagination. The developer can define dynamic routes, making it a lot easier to get parameters from the URI, than in plain React. In this application there are a lot of dynamic routes using an organizations ID. To acquire the ID in NextJs it was only to ask for the specific parameter.

NextJs is a framework in rapid development. A recent update to NextJs is the new app directory architecture, implementing new strategies for routing and rendering, as well as giving the developer more tools related to the layout [5].

The team chose to use this new app directory architecture for these reasons and as well as to ensure the application technology was as modern as possible to deliver a long lasting product to Matsentralen.

#### Next-Auth

In order to facilitate authentication on the front-end, the implementation of Next-Auth was utilized. Next-Auth is an open source authentication framework designed specifically for Next.js applications. The reasoning behind selecting this particular solution stemmed from its comprehensive support for various authentication mechanisms, including OAuth 2.0, Json Web Tokens (JWT), and a multitude of sign-in services, such as Google [8].

#### TypeScript

When working with NextJs, the user can choose between two programming languages: Javascript and Typescript. Since TypeScript requires the user to identify the type of the variable being used, the group chose to use TypeScript to easily be able to debug errors.

#### TailwindCSS

For the design of the frontend, the team needed a tool that integrated well with NextJs. The

team went for TailwindCSS. Tailwind is a tool that allows the developer to insert CSS directly into HTML components, using the "className" property. The language is made more compact, making it easier to read and write. In the tailwind.config file developers can set all values of colors, sizes and fonts, to be used in the application. This way the team did not have to style the application through a normal CSS file, resulting in both saved time and a better result.

## 4 Results

### 4.1 Scientific Results

This section will display some of the wireframes made early in the project and show all pages from the finished application.

#### 4.1.1 Wireframes

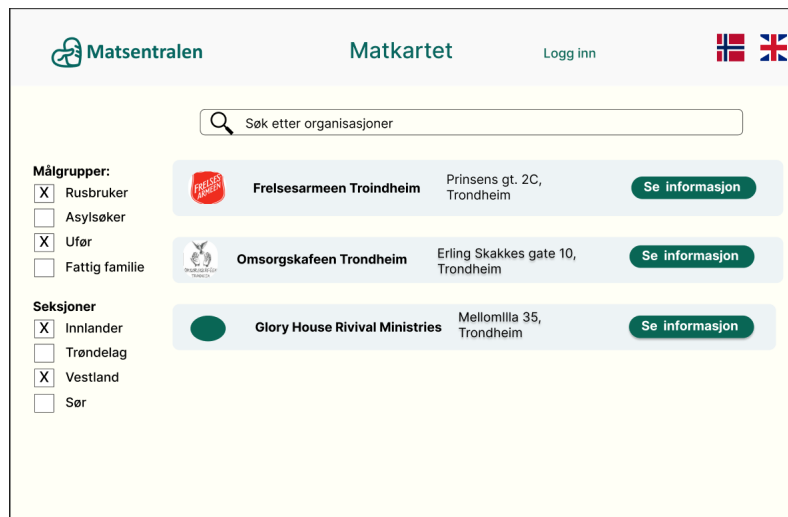


Figure 4: The start page of the application. Here users can search for and filter organizations.



Figure 5: User view of information about an organization

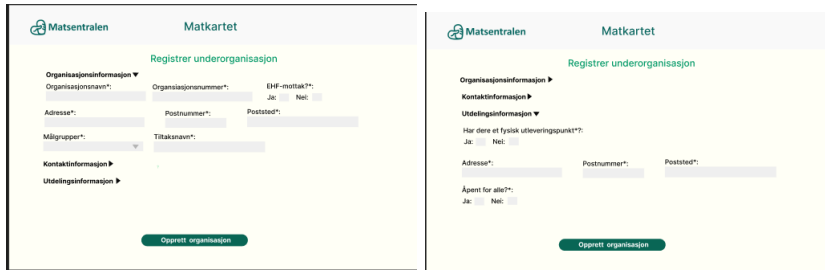


Figure 6: Some of the pages related to the registration of an organization

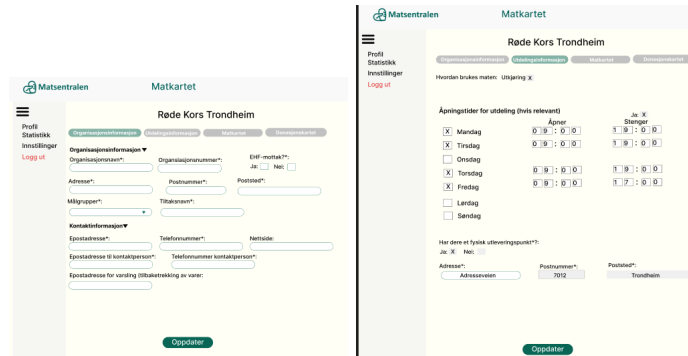


Figure 7: Organization view of data with the opportunity to update

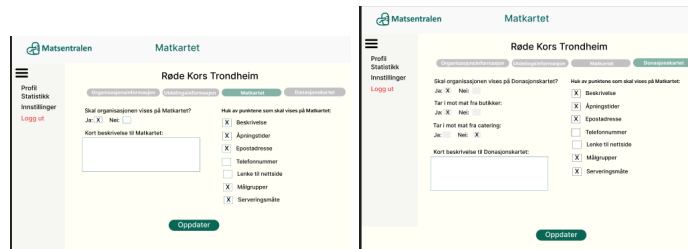


Figure 8: Organization view with options to set what the organization wants to be visible on Matkartet and Donasjonkartet



Figure 9: Admin view of the organizations in their section





Figure 10: Admin view of an organization in their section



Figure 11: Adminview of new application to their section



Figure 12: Admin view an application to their section

### 4.1.2 The finished application

The views of the application will be shown in three parts:

- The two view for people in need of food help, as well as the log in page.
- All view related to an organization - from application to editing your existing organization
- All views related to an admin user. This includes the page for the super admin.

#### Views for people in need of food help

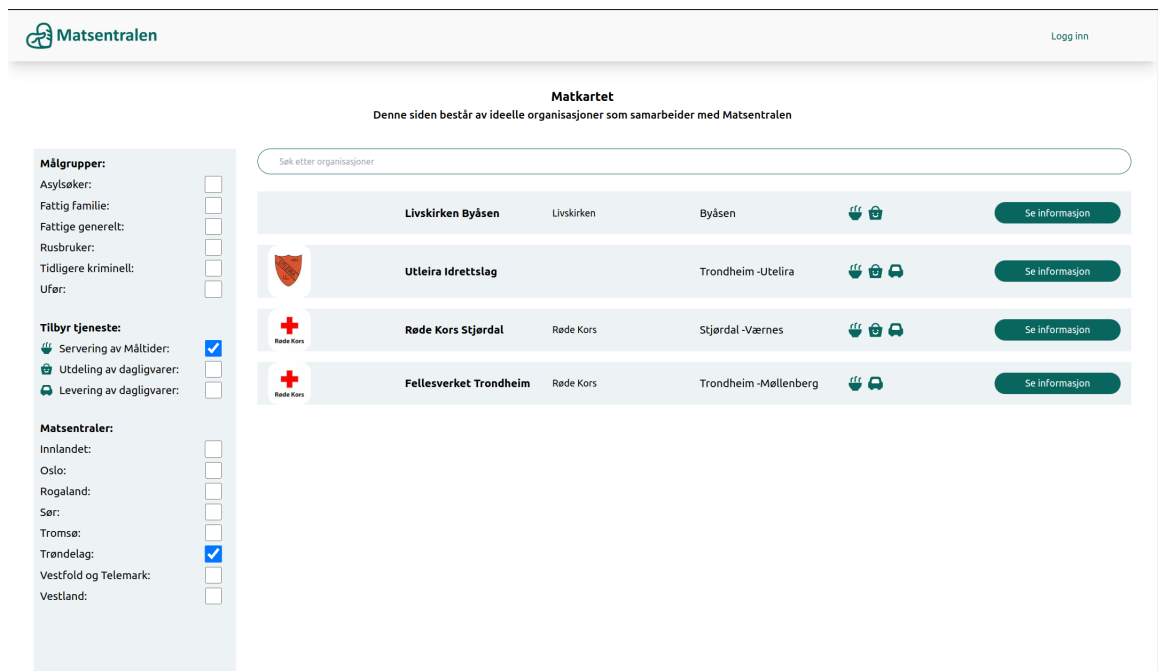


Figure 13: Start page - the matkartet replacement. The user can filter organizations based on organization name and parent organization name using the search bar and filter on the target groups the organizations focus on, the services they provide and which section they belong to

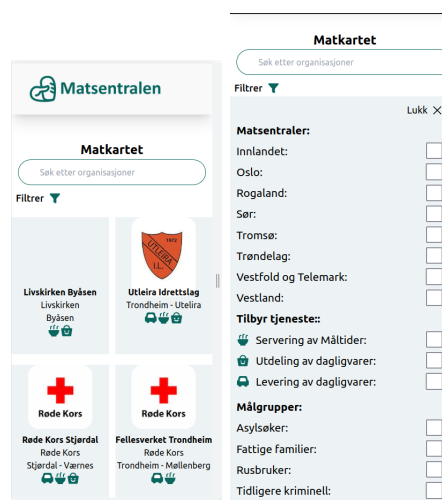


Figure 14: Start page for mobile

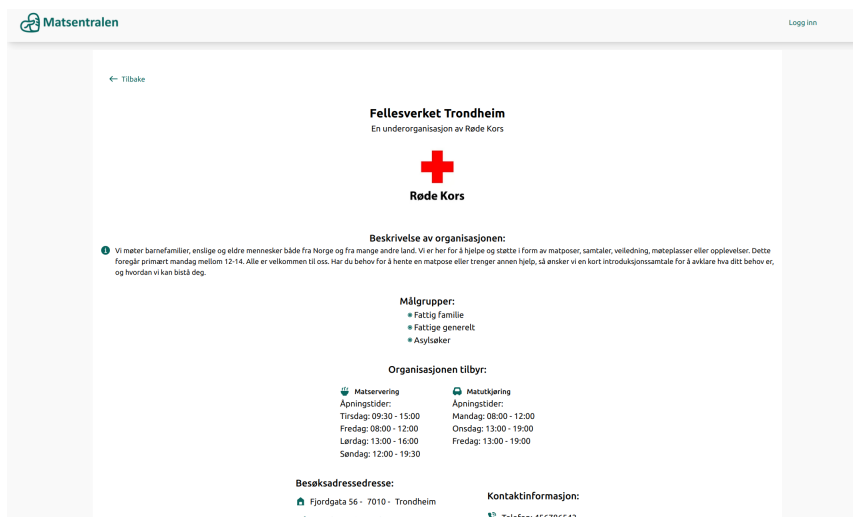


Figure 15: Organization information - When clicking "se informasjon" on the start page this opens

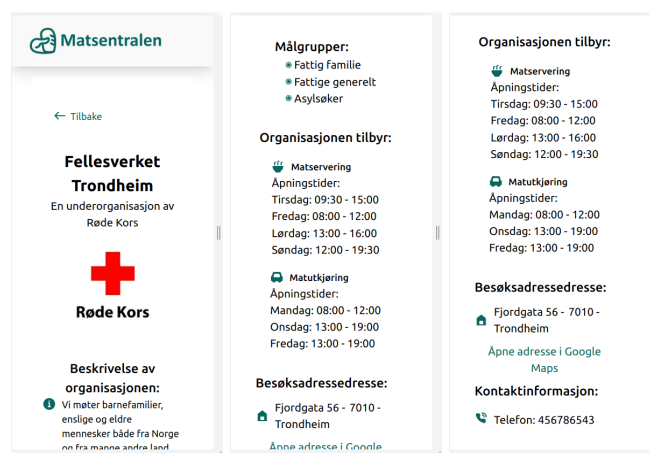


Figure 16: Organization detail page for mobile user

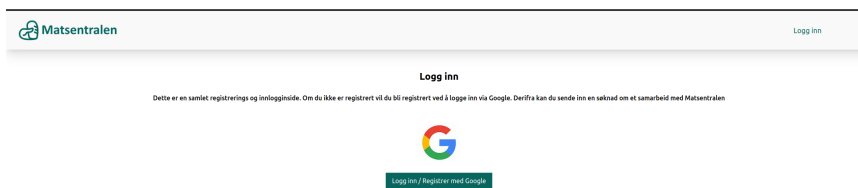


Figure 17: Log in page - Organizations, admins and managers all use the same login portal

## Views for organization

Figure 18: First time application

**Matsentralen** Evax Burrex - Organisasjonsansatt Se søknadsstatus Logg ut

### Andrestegsøknad samarbeid med Matsentralen

Dette er et skjema for å søke om samarbeid med Matsentralen. Fyll ut og send inn skjema. Når organisasjoner er godkjent vil du få mulighet til å fylle ut åpningstider, og sette hva som skal vises på Matkartet og Donasjonskartet. Tekstfelt som har en rød \*, er obligatoriske.

**Organisasjoninformasjon:**

Organisasjonsnavn \* (Fellesverket Trondheim)  
 Foreldreorganisasjon: (Røde Kors)  
 Er åpen for alle: Nei  Ja   
 Er organisasjonen et akuttmottak: Nei  Ja   
 Postnummer \* (7010)  
 Bydel hvis relevant (Møllenberg)  
 Nettside

Organisasjonsnummer \* (122232112)  
 Matsentral: (Trendelag)  
 Mottar økonomisk støtte til innkjøp av mat: Nei  Ja   
 Adresse \* (Fjordgata 56)  
 Poststed \* (Trondheim)  
 Målgrupper: Asylsøker:  Fattig familie:  Fattige generelt:  Rusbruker:   
 Tidligere kriminell:  Ufar:   
 Finansieringsform \* (Beskrivelse av hvordan organisasjonen finansieres. Kan brukes av administratorene i Matsentralen til å vurdere hvorvidt organisasjonen er skikket til et samarbeid)

**Kontaktinformasjon**

Epost \* (fellesverket@post.no)  
 Epost - Faktura \* (invoice@gmail.com)  
 Kontaktperson - Epost \* (navn@fellesverket.no)  
 Telefonnummer \* (92343212)  
 Kontaktperson - Navn \* (Navn Navnesen)  
 Kontaktperson - Telefon \* (32323232)

Legg til organisasjonsbilde  
 Choose File (RodeKors.png) Last opp bildet

**Tjenesteinformasjon**

Tilbyr matutkjøring: Nei  Ja   
 Tilbyr matservering: Nei  Ja   
 Besøststed - Adresse (Fjordgata 56)  
 Besøststed - Poststed (Trondheim)  
 Telefon (456786543)  
 Tilbyr utdeling av dagligvarer: Nei  Ja   
 Tilbyr drop-in kontakt: Nei  Ja   
 Besøststed - Postnummer (7010)  
 Epost (kontakt@fellesverket.no)

Bekreft endringer

Figure 19: Second time application

**Matsentralen** Evax Burrex - Organisasjonsansatt Gå til min side Logg ut

**Fellesverket Trondheim** Rediger organisasjon

**Organisasjoninformasjon:**

Organisasjonsnummer: 122232112  
 Paraplyorganisasjon: Røde Kors  
 Registreringsdato: 2023-05-20  
 Åpent for alle: Ja  
 Mottar økonomisk støtte: Nei  
 Er organisasjonen akuttmottak: Nei  
 Målgrupper: Fattig familie - Fattige generelt - Asylsøker  
 Åpent for alle: Ja  
 Adresse: Fjordgata 56  
 Postnummer: 7010  
 Poststed: Trondheim  
 Bydel: Møllenberg  
 Finansieringsform: Beskrivelse av hvordan organisasjonen finansieres. Kan brukes av administratorene i Matsentralen til å vurdere hvorvidt organisasjonen er skikket til et samarbeid

**Kontaktinformasjon**

Epost: fellesverket@post.no  
 Telefon: 92343212  
 Epost - Faktura: invoice@gmail.com  
 Kontaktperson - Navn: Navn Navnesen  
 Kontaktperson - Epost: navn@fellesverket.no  
 Kontaktperson - Telefon: 32323232

**Organisasjonsbilde**


  
**Røde Kors**

Figure 20: Home View Organization

Organisasjonsinformasjon

Tjenesteinformasjon

Matkartet

Donasjonskartet

Logg ut

Organisasjonsnavn \* Fellesverket Trondheim

Tilbake

Tilbakestill Bekreft endringer

**Organisasjoninformasjon:**

Organisasjonsnummer \* 122232112

Foreldreorganisasjon: Røde Kors

Registreringsdato: 2023-05-20

Er åpen for alle: Nei:  Ja:

Mottar økonomisk støtte: Nei:  Ja:

Er organisasjonen et akuttmottak: Nei:  Ja:

Målgrupper:

Asylsøker:  Fattig familie:  Fattige generelt:  Rusbruker:  Tidligere kriminell:

Adresse \* Fjordgata 56

Postnummer \* 7010

Poststed \* Trondheim

Bydel hvis relevant Møllenberg

Nettside

**Finansieringsform \***

Beskrivelse av hvordan organisasjonen finansieres. Kan brukes av administratorene i Matsentralen til å vurdere hvorvidt organisasjonen er skikket til et samarbeid

**Kontaktinformasjon**

Epost \* fellesverket@post.no

Telefonnummer \* 92343212


Epost - Faktura \* invoice@gmail.com

Kontaktperson - Navn \* Navn Navnesen

Kontaktperson - Epost \* navn@fellesverket.no

Kontaktperson - Telefon \* 32323232

**Organisasjonsbilde**



Røde Kors

Choose File No file chosen Last opp nytt bilde

**Finansieringsform \***

Beskrivelse av hvordan organisasjonen finansieres. Kan brukes av administratorene i Matsentralen til å vurdere hvorvidt organisasjonen er skikket til et samarbeid

**Kontaktinformasjon**

Epost \* fellesverket@post.no

Telefonnummer \* 92343212

Epost - Faktura \* invoice@gmail.com

Kontaktperson - Navn \* Navn Navnesen

Kontaktperson - Epost \* navn@fellesverket.no

Kontaktperson - Telefon \* 32323232

**Tjenesteinformasjon**

Tilbyr matkjøring: Nei:  Ja:

Tilbyr utdeling av dagligvarer: Nei:  Ja:

Tilbyr matservering: Nei:  Ja:

Besøststed - Adresse Fjordgata 56

Besøststed - Postnummer 7010

Besøststed - Poststed Trondheim

Epost kontakt@fellesverket.no

Telefon 456786543

Tilbyr drop-in kontakt: Nei:  Ja:

Figure 21: Home view organization edit page

Matsentralen

Organisasjonsinformasjon

Tjenesteinformasjon

Matkartet

Donasjonskartet

Logg ut

Evax Burrex - Organisasjonsansatt

Gå til min side

Logg ut

**Tjenesteinformasjon**

Rediger organisasjon

Tilbyr matkjøring: Ja

Tilbyr utdeling av dagligvarer: Nei

Tilbyr servering av måltider: Ja

Besøststed adresse: Fjordgata 56

Besøststed postnummer: 7010

Besøststed poststed: Trondheim

Epost: kontakt@fellesverket.no

Telefon: 456786543

Tilbyr drop-in kontakt: Ja

**Åpningstider**

**Kjøring:**

Mandag: 08:00 - 12:00

Onsdag: 13:00 - 19:00

Fredag: 13:00 - 19:00

**Utdeling av varer:**

**Servering:**

Tirsdag: 09:30 - 15:00

Fredag: 08:00 - 12:00

Lørdag: 13:00 - 16:00

Søndag: 12:00 - 19:30

Figure 22: The organization's view of the services they provide

← Tilbake Tilbakestill Bekreft endringer

**Tjenesteinformasjon**

Tilbyr matutkjøring: Nei:  Ja:

Tilbyr utdeling av dagligvarer: Nei:  Ja:

Tilbyr matservering: Nei:  Ja:

Besøststed - Adresse:

Besøststed - Postnummer:

Besøststed - Poststed:

Epost:

Telefon:

Tilbyr drop-inn kontakt: Nei:  Ja:

**Åpningstider**

**Kjøring:**

Mandag: 08:00 - 12:00

Onsdag: 13:00 - 19:00

Fredag: 13:00 - 19:00

**Utdeling av varer:**

**Servering:**

Tirsdag: 09:30 - 15:00

Fredag: 08:00 - 12:00

Søndag: 12:00 - 19:30

**Legg til nye åpningstider**

**Utkjøring**

▼ Apningstidspunkt:  :  Stengetidspunkt:  :

**Utdeling**

▼ Apningstidspunkt:  :  Stengetidspunkt:  :

**Servering**

▼ Apningstidspunkt:  :  Stengetidspunkt:  :

Figure 23: Edit page for organization service information

← Tilbake Rediger organisasjon

**Matkartet**

Vises på Matkartet: Ja

**Beskrivelse** Vi møter barnefamilier, enslige og eldre mennesker både fra Norge og fra mange andre land. Vi er her for å hjelpe og støtte i form av matposer, samtaler, veiledning, på møteplasser eller opplevelser. Dette foregår primært mandag mellom 12-14. Alle er velkommen til oss. Har du behov for å hente en matpose eller trenger annen hjelp, så Matkartet: ønsker vi en kort introduksjonssamtale for å avklare hva ditt behov er, og hvordan vi kan bistå deg.

Følgende ikke-obligatoriske felt vises på Matkartet: Åpningstider - Bruksmetoder - Besøkssted - Målgrupper - Nettside - Telefonnummer

Figure 24: Organization view for which fields are shown on the Matkartet replacement

, Nei: ), 'Huk av punktene som skal vises på Matkartet:' (Åpningstider: , Lenke til nettside: , Telefonnummer: , Epost: , Målgrupper: , Bruk av maten: , Besøkssted: ). There is also a 'Beskrivelse til Matkartet' section with a text area containing the same description as in Figure 24."/>

← Tilbake Bekreft endringer

**Skal organisasjonen vises på Matkartet**

Ja:

Nei:

**Huk av punktene som skal vises på Matkartet:**

Åpningstider:

Lenke til nettside:

Telefonnummer:

Epost:

Målgrupper:

Bruk av maten:

Besøkssted:

**Beskrivelse til Matkartet**

Vi møter barnefamilier, enslige og eldre mennesker både fra Norge og fra mange andre land. Vi er her for å hjelpe og støtte i form av matposer, samtaler, veiledning, møteplasser eller opplevelser. Dette foregår primært mandag mellom 12-14.

Figure 25: Organization view for editing which fields are to be shown

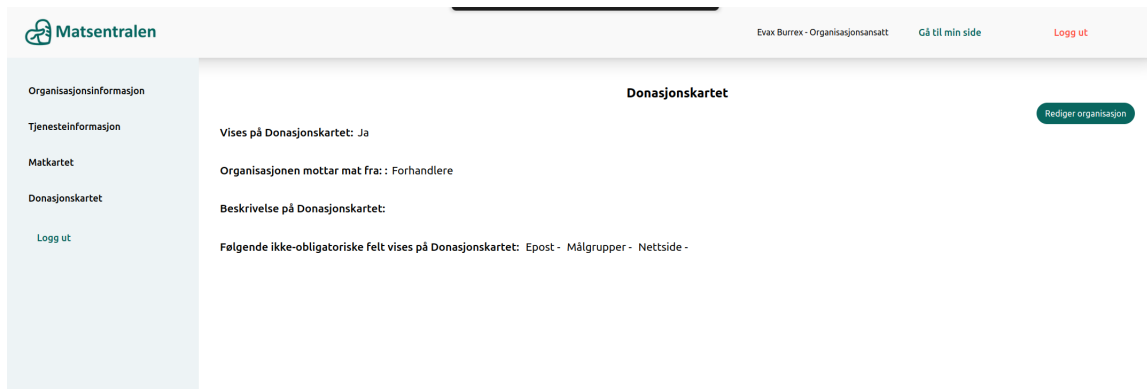


Figure 26: Organization view for which fields are to be shown when a similar version of Matkartet for businesses looking for organization is implemented

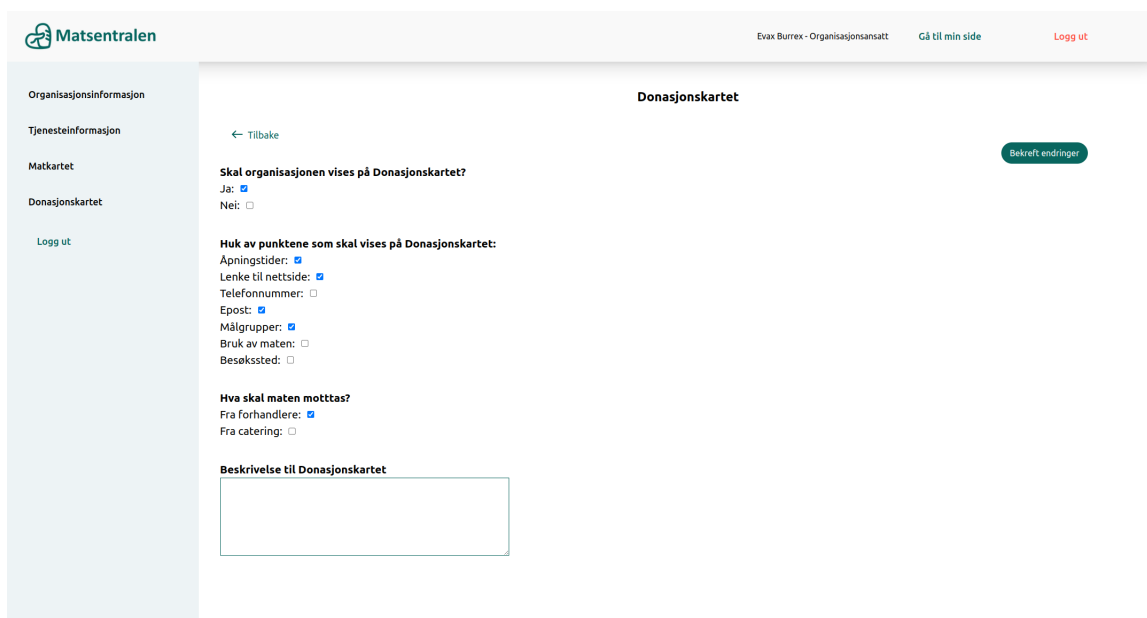


Figure 27: Organization view for editing which fields are to be shown on the donation map



## Admin Views

The screenshot shows the Matsentralen admin interface. The top navigation bar includes the Matsentralen logo, the user name 'Henrik Bur - Admin', a link to 'Gå til min side', and a 'Logg ut' button. The left sidebar contains navigation options: 'Liste over organisasjoner', 'Søknader', and 'Administrer felter'. The main content area is titled 'Organisasjoner i Matsentralen Trøndelag' and includes a sub-header 'Skriv ut organisasjoner til regneark'. Below this is a table of approved organizations.

Organisasjonsnavn	Paraplyorganisasjon	Telefon	Epost	Vises på Matkartet	Vises på Donasjonskartet	
Fellesverket Trondheim	Røde Kors	92343212	fellesverket@post.no	Ja	Ja	<a href="#">Vis mer</a>
Røde Kors Stjørdal	Røde Kors	344324324	rode@kors.no	Ja	Nei	<a href="#">Vis mer</a>
Livskirken Byåsen	Livskirken	456789567	livs@kirken.no	Ja	Nei	<a href="#">Vis mer</a>
Utleira Idrettslag		433433433	fotball@utleira.no	Ja	Nei	<a href="#">Vis mer</a>

Figure 28: Admin's view of the approved organizations in their section

The screenshot shows the Matsentralen admin interface for application requests. The top navigation bar is identical to Figure 28. The left sidebar is the same. The main content area is titled 'Søknader' and is divided into two sections: 'Førstegangssøknader' and 'Andregangssøknader'. Each section contains a table of application requests.

Organisasjonsnavn	Paraplyorganisasjon	Telefon	Epost	Mottar økonomisk støtte	Godkjent av Matsentralen?	
Frelsesarmeen Trondheim	Frelsesarmeen	191919191	frelsesarmeen@frelse.no	Nei	Nei	<a href="#">Videre til søknad</a>
Røde Kors Stjørdal	Røde Kors	344324324	rode@kors.no	Nei	Nei	<a href="#">Videre til søknad</a>
Utleira Idrettslag		433433433	fotball@utleira.no	Nei	Ja, og avventer andregangssøknad fra organisasjon	<a href="#">Videre til søknad</a>
Livskirken Byåsen	Livskirken	456789567	livs@kirken.no	Nei	Nei	<a href="#">Videre til søknad</a>

Organisasjonsnavn	Paraplyorganisasjon	Telefon	Epost	Mottar økonomisk støtte	Registreringsdato	
Fellesverket Trondheim	Røde Kors	92343212	fellesverket@post.no	Nei	2023-05-20	<a href="#">Videre til søknad</a>

Figure 29: Admin's view first and second time applications in their sections

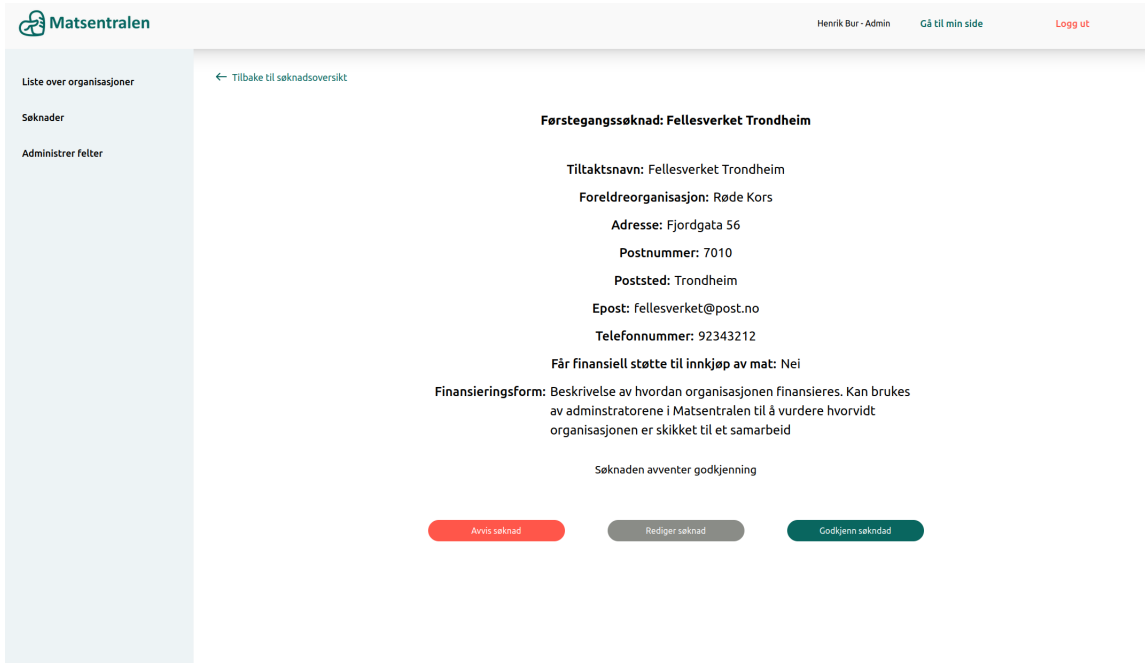


Figure 30: Admin's view of a first time application

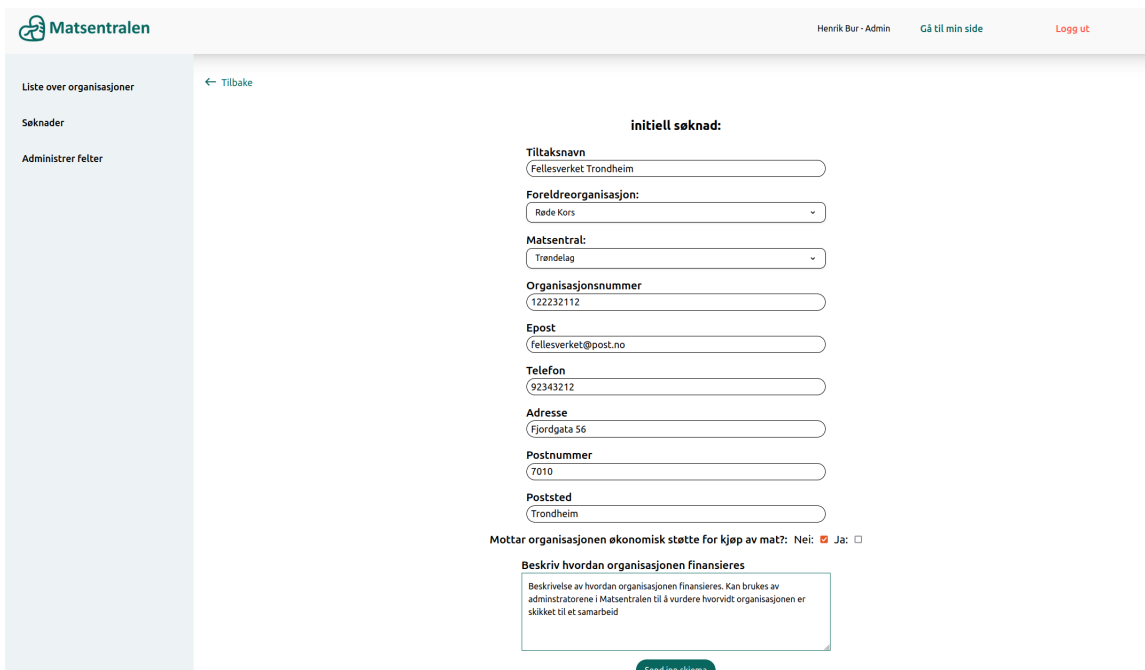


Figure 31: Admin's view to edit first time application

Liste over organisasjoner

Søknader

Administrer felter

### Andregangssøknad: Fellesverket Trondheim

**Organisasjoninformasjon:**

Organisasjonsnummer: 122232112	Paraplyorganisasjon: Røde Kors	Registreringsdato: 2023-05-20
Åpent for alle: Ja	Mottar økonomisk støtte: Nei	Målgrupper: Fattig familie - Fattige generelt - Asylsøker
Åpent for alle: Ja	Adresse: Fjordgata 56	Postnummer: 7010
Poststed: Trondheim	Bydel: Møllenberg	Finansieringsform: Beskrivelse av hvordan organisasjonen finansieres. Kan brukes av administratorene i Matsentralen til å vurdere hvorvidt organisasjonen er skikket til et samarbeid

Er organisasjonen et akuttmottak?: Nei

**Organisasjonsbilde**



**Røde Kors**

**Kontaktinformasjon**

Epost: fellesverket@post.no	Telefon: 92343212	Epost - Faktura: invoice@gmail.com
Kontaktperson - Navn: Navn Navnesen	Kontaktperson - Epost: navn@fellesverket.no	Kontaktperson - Telefon: 32323232

**Tjenesteinformasjon**

Tilbyr matkjøring: Ja	Tilbyr utdeling av dagligvarer: Nei	Tilbyr servering av måltider: Ja
Besøkssted adresse: Fjordgata 56	Besøkssted postnummer: 7010	Besøkssted poststed: Trondheim
Epost: kontakt@fellesverket.no	Telefon: 456786543	Tilbyr drop-in kontakt: Ja

Avvis søknad
Rediger søknad
Godkenn søknad

Figure 32: Admin's view of a second time application

Liste over organisasjoner

Søknader

Administrer felter

← Tilbake
Tilbakestill Bekreft endringer

**Organisasjoninformasjon:**

Organisasjonsnavn * <input type="text" value="Fellesverket Trondheim"/>	Organisasjonsnummer * <input type="text" value="122232112"/>
Foreldreorganisasjon: <input type="text" value="Røde Kors"/>	Matsentral: <input type="text" value="Trendelag"/>
Registreringsdato: 2023-05-20	Er åpen for alle: Nei <input type="checkbox"/> Ja: <input checked="" type="checkbox"/>
Er organisasjonen et akuttmottak: Nei: <input checked="" type="checkbox"/> Ja: <input type="checkbox"/>	Mottar økonomisk støtte: Nei: <input checked="" type="checkbox"/> Ja: <input type="checkbox"/>
Adresse * <input type="text" value="Fjordgata 56"/>	Postnummer * <input type="text" value="7010"/>
Poststed * <input type="text" value="Trondheim"/>	Bydel hvis relevant <input type="text" value="Møllenberg"/>
Målgrupper: Asylsøker: <input checked="" type="checkbox"/> Fattig familie: <input checked="" type="checkbox"/> Fattige generelt: <input checked="" type="checkbox"/> Rusbruker: <input type="checkbox"/> Tidligere kriminell: <input type="checkbox"/> Ufor: <input type="checkbox"/>	Nettside <input type="text"/>
Finansieringsform * <small>Beskrivelse av hvordan organisasjonen finansieres. Kan brukes av administratorene i Matsentralen til å vurdere hvorvidt organisasjonen er skikket til et samarbeid</small>	

**Kontaktinformasjon**

Epost * <input type="text" value="fellesverket@post.no"/>	Telefonnummer * <input type="text" value="92343212"/>
Epost - Faktura * <input type="text" value="invoice@gmail.com"/>	Kontaktperson - Navn * <input type="text" value="Navn Navnesen"/>
Kontaktperson - Epost * <input type="text" value="navn@fellesverket.no"/>	Kontaktperson - Telefon * <input type="text" value="32323232"/>

Figure 33: Admin's view for editing a second time application

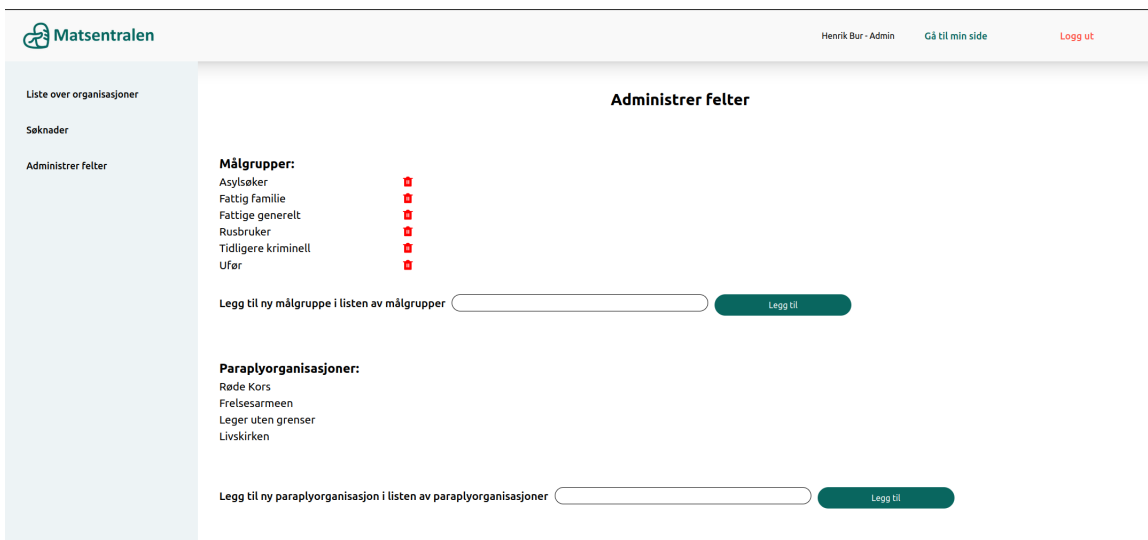


Figure 34: Admin's view adding and removing targetgroups and adding parent organizations. These could be used by all users of the applications

## **4.2 Engineering results**

As seen on the given slides, most of the features put in the vision document have been fulfilled. Here, the results of both functional and non-functional features will be described.

### **4.2.1 Functional features**

The following table describes the status of the functional requirements of the project. More detailed descriptions of the requirements can be found in the vision document.

<b>Feature</b>	<b>Priority</b>	<b>Status</b>
Functionality for storage of data in database	High	Completed
Functionality for people in need of food help to view partner organizations of Matsentralen.	High	Completed
Functionality for people in need to view detailed organization information.	High	Completed
Authentication and authorization functionality for organizations, admins and system administrators.	High	Completed
Functionality for partner organization applications	High	Completed
Functionality for organizations to administer their data	High	Completed
Functionality for admins to edit organization information	Medium	Completed
Functionality for food providers to find volunteer organizations.	Medium	Implemented on the backend, but not frontend
Functionality for multiple languages	Low	Dropped because of time constraints
Functionality to export organizations to spreadsheet	Low	Completed
Functionality for monthly surveys for service stats.	Low	Dropped by Matsentralen

Table 1: Functional features

#### 4.2.2 Non-functional features

##### Security

For security, it was important that the application followed basic security measures described in OWASP top 10, including Cross Site Scripting and SQL injection. Next.js, which is a React framework renders data through JSX rather than HTML. This means that it escapes any values embedded in the JSX and validates the input as a string, preventing an attacker from inserting

"`<script></script>`" tag to execute Javascript code making it vulnerable to cross site scripting. However, the application is not safe from cross site scripting attacks if it mutates the DOM directly using for example `innerHTML` [7]. Furthermore, as for SQL injection Spring Boot JPA prevents it by generating SQL queries and given that custom queries is placed in a `@Query()` annotation the application run no risk of security breached through SQL injection.

### **Usability**

Usability was important, with different focus points for the different parts of the application. In the part of the application for people in need of food help, WCAG principles were followed to make the application as easy to use as possible: All elements are wrapped in elements corresponding to their task in application, allowing for text-to-speech tools to interpret the site for people with visual impairments. Further, large font sizes is used and many fields have an additional icons, helping people that may struggle to read or have limited knowledge of Norwegian, to understand the basic information. As for the organization and admin parts of the application, a minimal amount of colors are used. The black text with a white background ensures a maximum amount of contrast.

### **Hosting of the application**

Early in the project hosting of the application was set up in Google Cloud. The server was connected to the main branch of the Github repository. In addition an SQL database was used in Google Cloud. At delivery the Google Cloud is taken down, but there will only be need to change propriety names to host it in another cloud. The main reason for the termination of the cloud was the price of the service, which was not a price worth to pay while developing. At the moment of delivery the application therefore runs locally with a local SQL database.

### **Supportability**

As discussed, a third party is set to continue development of the application after delivery. Therefore the application is developed allowing for the further features and adjustments. On the front-end components are extensively used. This will make it easy to change both design and functionality in the future without having to change a lot of files.

On the back-end further implementation of tables and entities are made simple by using the created Spring Boot functionality. Due to the loose coupling, the entities can easily be given more attributes without having to restructure code.

### **Physical Requirements**

For the application to be accessible to as many people as possible, the interface for people in need of food help also needed to be fit for mobile screens. The design of the mobile version manages to present the same information as the design for computer screens, while being easy to use. This can be seen in figures 14 and 16 .The rest of the application is fit for various sizes of computer screens.

## **4.3 Administrative Results**

### **4.3.1 Planning**

To ensure sufficient tracking of tasks and milestones during the project the team adapted a Scrum based methodology for agile development. A Gantt diagram was developed in the start phase of the project to give a roughly overview of the tasks distributed between different sprints. However, due to the small amount of information in the Gantt diagram, Scrum boards were utilized to give a more comprehensive overview of the tasks. Using boards and backlogs provided by Jira Software made it easier to customize task and divide them in smaller tasks. This approach resulted in more flexibility and opportunity to view all future tasks and all tasks relevant for the current sprint.

### 4.3.2 Time Management

The team continuously documented all time spent each day during the project in a timetable. The time table gave an overview of what each team member had worked on during the week, distributed between different categories such as development, research, main report and documentation. In addition, each team member wrote a weekly log explaining what had been worked on, what went great and not so great. The goal of the time management was to be within the required  $500 \pm 50$  hours, to secure consistent workload throughout the project. The figures below give an overview of the time management for each member and for each category.

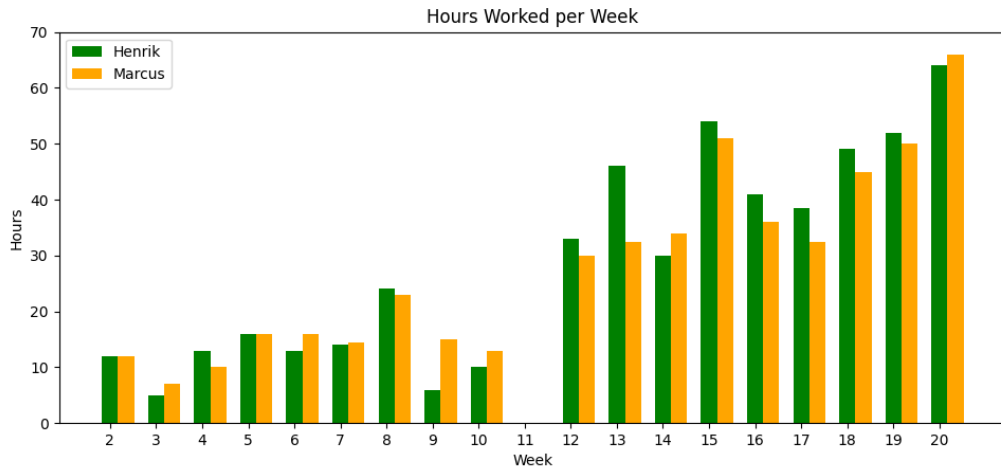


Figure 35: Hourly graph of hours worked

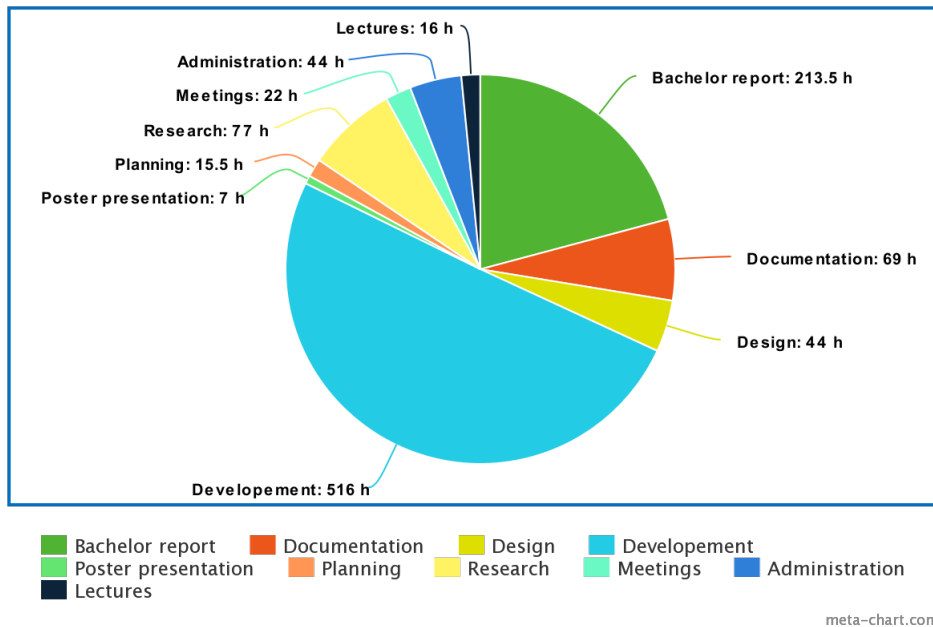


Figure 36: Hours of work for each category



## 5 Discussion

The team is extremely satisfied with the results of the application. The problem thesis of "How can a minimal viable product, full-stack application, be developed to help people in need find food help, while fixing the scaling problem of Matsentralen", has, in the opinion of the team, successfully been completed. In this section the results achieved will be discussed, as well as challenges that occurred and what is still to be implemented.

### 5.1 Scientific Results

As detailed in chapter 3.1, Matsentralen did not have a clear vision of what they wanted to achieve going into the project. Therefore the time spent on research took considerably more time than expected. During the project as the team had many interviews with Matsentralen, Matsentralen got a clearer view of what they wanted. Including the other sections of Matsentralen also helped in the achieve this.

Because of this time consume, the design of the wireframes was completed later than expected. Therefore, the team did not have time to conduct quantitative users tests on the views for people in need of food help, however multiple of Matsentralen's employees gave their opinion on the wireframes for the whole application. This testing was integral to help Matsentralen define what they wanted of features.

#### 5.1.1 Wireframes

At the time of the completion of the prototype, wireframes were made for nearly all views needed to complete the functional requirements. There were no existing design to base the wireframes on, and all the wireframes therefore had to be made fully from scratch. As the requirements changed during development, some views are missing and some of the wireframes are no longer relevant. The wireframes for the people in need of food help are very similar to the end product, as a result of using a high-fidelity approach. The other wireframes are in a larger degree different from the finished application when it comes to design, because of the low fidelity approach. Setting the structure for these pages were of larger importance, hence the structure and flow is similar to the final product.

#### 5.1.2 Application Views

As in section 4.1.2 the discussion of the application views is split into three parts.

##### **For people in need of food help**

In this part of the application it was integral to easily allow the user to find organizations that fit their needs. Therefore the user can filter by organization name, parent organization name, target groups, service methods and Matsentralen sections at the same time. The organization cards display the most important information about an organization.[36](#)

Further, when viewing the detailed information about an organization the team chose a simple as possible design to not overwhelm the user. [15](#)

##### **Organization views**

The views made for organizations presents the structure in an understandable way, where the organization first is presented with a first time application the organization must fill out. The application contains only a few fields with the most important information about an organization.

[18](#)

The second time application and other pages contains considerably more fields, which made it more challenging to present the entire form for the user. However, using simplistic design and informative descriptions the pages the elements are presented in an understandable way. 19

When the organization is approved the, four different information sorts organization information, service information, and what is to be shown to the public is split up to not push to much information on the user.

### **Admin views**

The admin portal gives the employees of Matsentralen a well organized system to view, edit and review applications and existing organizations<sup>28</sup>.Data that in the current solution has to be located and updated in multiple spreadsheets are now more simple to locate and read through the application.

#### **5.1.3 Validation of fields**

For Matsentralen it was important that data registered by the organizations was properly validated to ensure information was correct. Validation through React is therefore implemented for all necessary fields.

#### **5.1.4 Documentation of code**

As the project is to be inherited by a third party after delivery, documentation of code has been implemented on the back-end. Entities and relevant methods are documented to ensure to help the new developers understand the role of each entity.

## **5.2 Engineering Results**

### **5.2.1 Functional features**

Most of the features listed in the vision document have been implemented. These include all features of high importance and that fulfil the thesis statement. In general the completed features can be summarized to:

- People in need of food help can find and view organizations that provide food help.
- Matsentralen's various ways of storing and handling of organizations are centralized into one system that can handle Matsentralen's up scaling problems

Some features were though not completed.

### **Missing features**

The exclusion of the interface for the donation site is the most notable missing features. There is full back-end support for this feature. Organizations are also able to set which fields are to be shown, similarly to the Matkartet replacement solution. However, because of time constraints, the team prioritized to finish features of higher priority instead of implementing this interface.

Furthermore, during the research phase the team suggested that a feature to support multiple languages for parts of the application would make the site more available to a larger audience. This was an idea Matsentralen were enthusiastic about. This feature was dropped since it would be very time consuming.

At last there initially a plan to implement a feature where admins could post surveys for organizations to answer, about how many people they have helped and how much food they had distributed. The idea was that it would be easier for Matsentralen to keep track of how the food they passed on was used. However, the features was moved to another of Matsentralen's running project as the feature was in many ways unrelated to the problem statement.

### 5.2.2 Non-functional features

#### Security

Considering security the use of standardized frameworks and technologies reduce the amount of security features a developer has to implement to deal with security issues. The application utilizes these implementations to prevent vulnerabilities such as SQL injection and cross site scripting. Furthermore, the front-end part of the application do not mutate the DOM by for example calling functions as innerHTML, preventing cross site scripting vulnerabilities. Because the scope of the project was to develop a working MVP the security issues had reduced priority, focusing rather on usability and features instead of conducting an extensive security assessment. The use of social login through Google, using OAuth 2.0 standards removed the need for storing user password in a database. However, for deployment to production a greater security assessment would be needed to detect security vulnerabilities not detected by the developers of the application.

#### Usability

Usability is an important part of the application given that most of the user mass may be people with limited technical knowledge. One of the goals of the project was to reduce the amount of manual labor for the employees at Matsentralen. This includes one problem the client addressed through a meeting saying that a lot of people call Matsentralen's offices to clarify questions that the website did not provide any answer to. With great usability on the website users are able to navigate through the website clarifying any questions they may have. Furthermore, the application was designed using already existing color palettes used on Matsentralen's main website securing consistency between sites and following WCAG principles.

#### MVP

By developing an MVP there was an idea that third party would build upon the project to eventually make it an production ready application. This was later confirmed by the client during one meeting. Therefore, it was important that that the code would support this transition to a third party by structured code through components and file structure. Furthermore, the team offered to provide support for the third party, clarifying questions that they may have.

### 5.2.3 The setup of the project

There was no existing technology nor architecture to build on. Therefore, all architecture had to be built from scratch. This involved the NextJs front-end project, the REST API, database and hosting as well as the image database. The time invested in the architecture setup resulted in development starting nearly a month later than planned.

The setup of the architecture did however pay off when done. As described in 2.2.1, building a REST API allowed the team to develop the frontend and backend separately. This increased development speed substantially.

Though it must be mentioned that the time spent on setting up the Google Cloud did not match the value the project got from it as the cloud had to be taken down.

#### 5.2.4 Choices of technology

The choices of technology were in general very good and made it possible to develop the finished application. Many of the technologies were new to one or both team members, which led to a steep learning curve during development.

##### Spring Boot

The selection of Spring Boot as the framework for constructing the API proved to be very successful. With Spring Boot's auto configurations and provided services, the team managed to build a well functioning REST API, that will be highly expandable for the developers inheriting the project. Further, Spring Boot's security features helped the team build a backend that fulfills the security demands.

Using Kotlin for Spring Boot worked well. The data classes provided in Kotlin reduced the amount of lines of code that had to be written. As neither team member was experienced in Kotlin some time was spent getting used to the language.

##### NextJs

NextJs was a useful React framework. The dynamic routing offered in NextJs made routing better than the hard coded paths the user must set up in plain React. Further the application has a very fast response time, which partly can be attributed to NextJs server side rendering.

Though the choice of NextJs itself was unproblematic, an unexpected complication occurred related to the version of NextJs that was chosen. The team chose to use the experimental NextJs app directory that was introduced late 2022. As detailed in 3.3.1 this new implementation offers many useful features, however as the implementation was very new, it did not exist many sources that explained the differences between the old and new solution. Therefore a lot of time was spent finding solutions that were not deprecated.

### 5.3 Administrative Results

#### 5.3.1 Planning

Planning a system development project presents several challenges due to the complexity. The initial phase of the project demanded a substantial investment of time in acquiring a understanding of the project's scope, involving both system requirements and the technological framework. One noteworthy challenge regarding time estimation stemmed from the considerable workload associated with developing a system from scratch. Notably, considerable time was dedicated to activities beyond direct coding, such as engaging in client communication, designing the system architecture, establishing the database structure, and implementing the front-end and back-end architecture. These essential pre-coding tasks, although not immediately involving the actual coding process, were crucial for laying the foundation of the project and, consequently, demanded substantial time allocation.

Furthermore, the team decided that most of the work should be in person accommodating for better communication and planning.

#### 5.3.2 Time Management

In the beginning of the semester the team had to account for another course running parallel with the bachelor project. The plan was to devote most of the time Thursday and Friday working about 8 hours each day. In addition, the team also had to accommodate for submissions for the other curriculum which resulted in the necessity for flexibility in managing the workload. After the exam

of March 20th the team had more time working on the project as shown in figure 35 after week 11.

The plan after this exam was to devote weekdays to project work. However, recognizing the potential time constraints caused by work commitments and other external factors, the team reached a agreement that it might be necessary to extend working hours and allocate weekends to compensate for any potential time loss.

Furthermore, even though the team had planned an initial time frame for each task during development the team deemed it more important to complete the tasks rather than to comply with the time estimates. This allowed the team to rather move tasks back to the backlog or move it forward to another sprint.

## 6 Conclusion and further work

The goal of the project addressed in the problem statement "How can a minimal viable product, full-stack application, be developed to help people in need find food help, while fixing the scaling problem at Matsentralen" was based on the wish from Matsentralen to have a system that make it easier for people to locate non-profit organizations providing food services. Furthermore, this included relieving the employees from a lot of manual labor of register and updating organizations. This report presents the work of developing a MVP that would meet Matsentralens's needs, technological choices and findings obtained during the project work.

### 6.1 Conclusion

The developed product resulted in a successful MVP that met the requirements provided by Matsentralen. The MVP has been the result of great cooperation with Matsentralen giving the team valuable feedback about a lot that was uncertain in the start phase of the project. Furthermore, the achievement of the project can be contributed to key factors including time management, choices of technology, team work and agile development methodology.

The result of the project will be a system that can be built upon to eventually become a production ready application that Matsentralen can utilize, both improving efficiency and reducing the amount of manual labor. Furthermore, a production ready application will hopefully help people in need find food help by introducing a more user friendly user interface that lets users navigate and search on specific requests.

Throughout the course of the project, the team has gained valuable knowledge and expertise of various aspects of system development, client communication, and agile development methodologies. By creating a system from scratch, that includes communication with client, designing, architecture setup and developing both the frontend and backend of the application, has introduced the team to challenges that we have never encountered before.

### 6.2 Further work

For the MVP to become a fully functional application there are some features the team believes the third party inheriting the project should focus on.

#### 6.2.1 Multiple Languages

Given that a lot of users who look for food help are non-Norwegian speakers, support for multiple languages on the website should be a focus point during further development. In this project for multiple languages was not implemented due to time constraints and low priority.

#### 6.2.2 Allowing users to search for cities and district

To help users locate organizations in their proximity, search for cities and districts within larger cities should be implemented. This will be especially important as Matsentralen acquires more partner organization.

#### 6.2.3 Donation Map

The implementation of the donation map will be a complicated task, as there is full back-end support for the feature, however the interface will need to be implemented. This will let businesses in the food industry find non-profit organization seek their products.

#### **6.2.4 Divide the application in multiple application**

For practical reasons the application may need to be split into three smaller applications. Having people in need of food help, organizations and admins all using the same web page will not be sustainable in a long time perspective. In addition to the login being unnecessary complicated, this makes the site more vulnerable to downtime.

## Social Impact

The potential societal impact of this application is substantial. As already discussed in the introduction Matsentralen positively contributes to 7 of the 17 UN Sustainability Goals. By further distributing food to people in need, that otherwise would have gone to waste, Matsentralen has a large societal impact both environmentally and economically.

The role of this application in this environment is to make Matsentralen's services available to a larger user mass. If the application succeeds with this goal, Matsentralen will have be able to provide their service on a larger scale, and therefore have a larger social impact.

As the application does not store nor process any sensitive data there are no larger ethical problems that the group have identified. However, as the application has a big potential to do good, we as developers also has a responsibility to make the application usable for as many people as possible. Though the application takes multiple vision impairments into account there are as discussed other factors to consider.



## References

- [1] Alexandra Altvater. *Gradle vs. Maven: Performance, Compatibility, Speed, & Builds*. en-US. June 2017. URL: <https://stackify.com/gradle-vs-maven/> (visited on 05/01/2023).
- [2] Sean Dexter. *Figma continues to skyrocket — 63% reported it was their primary UI tool*. en. Dec. 2021. URL: <https://uxdesign.cc/figma-continues-to-skyrocket-63-reported-it-was-their-primary-ui-design-tool-in-2021-bb9390a8b96b> (visited on 04/24/2023).
- [3] *EUs webdirektiv (WAD) | Tilsynet for universell utforming av ikt*. nb. URL: <https://www.uutilsynet.no/webdirektivet-wad/eus-webdirektiv-wad/265> (visited on 04/19/2023).
- [4] *Git - About Version Control*. URL: <https://git-scm.com/book/en/v2/Getting-Started-About-Version-Control> (visited on 05/20/2023).
- [5] <https://www.smashingmagazine.com/author/atila-fassina>. *Understanding App Directory Architecture In Next.js*. en. Section: General. 0. URL: <https://www.smashingmagazine.com/2023/02/understanding-app-directory-architecture-next-js/> (visited on 05/02/2023).
- [6] *Incremental build*. URL: [https://docs.gradle.org/current/userguide/incremental\\_build.html](https://docs.gradle.org/current/userguide/incremental_build.html) (visited on 05/01/2023).
- [7] *Introducing JSX - React*. en. URL: <https://legacy.reactjs.org/docs/introducing-jsx.html> (visited on 05/20/2023).
- [8] *Introduction | NextAuth.js*. en. Apr. 2023. URL: <https://next-auth.js.org/getting-started/introduction> (visited on 05/22/2023).
- [9] *Lov om endringer i likestillings- og diskrimineringsloven mv. (universell utforming av IKT-løsninger) - Lovdata*. URL: <https://lovdata.no/dokument/LTI/lov/2021-06-11-77> (visited on 04/19/2023).
- [10] *Matbransjen*. URL: <https://www.matsentralen.no/matbransjen> (visited on 04/26/2023).
- [11] *Matsentralen Norge - FN's Bærekraftsmål*. URL: <https://www.matsentralen.no/baerekraftsmal> (visited on 04/27/2023).
- [12] *Matsentralen Norge - Om Oss*. URL: <https://www.matsentralen.no/om-oss> (visited on 03/29/2023).
- [13] *Minimum Viable Product (MVP)*. en-US. URL: <https://www.productplan.com/glossary/minimum-viable-product/> (visited on 04/18/2023).
- [14] Prerender. *Server-Side Rendering - Pros & Cons for Your SEO and Budget*. en-US. Feb. 2021. URL: <https://prerender.io/blog/what-is-srr-and-why-do-you-need-to-know/> (visited on 04/20/2023).
- [15] *Samarbeid med ideelle organisasjoner - Matsentralen*. URL: <https://www.matsentralen.no/ideelle-organisasjoner> (visited on 04/26/2023).
- [16] *Spring Boot Auto-configuration - javatpoint*. URL: <https://www.javatpoint.com/spring-boot-auto-configuration> (visited on 05/01/2023).
- [17] *WCAG-standarden | Tilsynet for universell utforming av ikt*. nb. URL: <https://www.uutilsynet.no/wcag-standarden/wcag-standarden/86> (visited on 04/19/2023).
- [18] *What is a REST API? | IBM*. en-us. URL: <https://www.ibm.com/topics/rest-apis> (visited on 04/17/2023).
- [19] *What is a Wireframe? Guide With Types, Benefits & Tips (2023)*. en-US. Apr. 2023. URL: <https://visme.co/blog/what-is-a-wireframe/> (visited on 04/18/2023).

- [20] *What is dependency injection in object-oriented programming (OOP)?* – TechTarget Definition. en. URL: <https://www.techtarget.com/searcharchitecture/definition/dependency-injection> (visited on 04/25/2023).
- [21] *What is Scrum?* en. URL: <https://www.scrum.org/learning-series/what-is-scrum> (visited on 04/17/2023).
- [22] *What is server side rendering: pros and cons | EPAM SolutionsHub.* en. URL: <https://solutionshub.epam.com/blog/post/what-is-server-side-rendering> (visited on 04/19/2023).
- [23] *What is Universal Design?* en-US. URL: <https://universaldesign.org/definition> (visited on 04/19/2023).

## Vedlegg

**Attachment A - Source code**

**6.3 Attachment B - Pre-work plan**

**Attachment C - Vision Document**

**Attachment D - Project Manual**

**Attachment E - Requirement specifications**

**Attachment F - System Documentation**

