

Lena Amdal-Larsen
Vilde Gunnes Bertelsen

The Development of "Tempero Beregningsmodeller"

The digitalization of a service

May 2022

NTNU

Norwegian University of Science and Technology
Faculty of Architecture and Design
Department of Design

Lena Amdal-Larsen
Vilde Gunnes Bertelsen

The Development of "Tempero Beregningsmodeller"

The digitalization of a service

Bachelor's thesis
May 2022

NTNU

Norwegian University of Science and Technology
Faculty of Architecture and Design
Department of Design



Norwegian University of
Science and Technology

Abstract

Title: The Development of “Tempero Beregningsmodeller”

Date: 13.05.2022

Participants: Lena Amdal-Larsen and Vilde Gunnes Bertelsen

Supervisor: Lefteris Papachristos

Employer: Tempero Energitjenester AS

Keywords: BWU, Web Development, calculation models, digitalization, design, development

Pages: 66

Number of attachments: 5

This project focuses on increasing efficiency through the digitalization of one of the consulting company Tempero Energitjenester AS’s services: their calculation models. Tempero’s current handling of this happens through their customers contacting them each time they need their service. This project aims to give Tempero an application that will allow their customers to use this service digitally, and consequently increase efficiency in both Tempero’s and the customers’ workplace. This project is based on previous work, where insights and research were gathered. In this report, the process of designing and developing the minimum viable product named “Tempero Beregningsmodeller” is presented through a design phase that follows principles of useability and accessibility, the concept of journey-driven design, as well as a development phase using the MERN Stack. Tempero Beregningsmodeller is an application that makes Tempero’s calculation models available to their clientele.

Sammendrag

Tittel: Utviklingen av “Tempero Beregningsmodeller”

Dato: 13.05.2022

Deltagere: Lena Amdal-Larsen and Vilde Gunnes Bertelsen

Veileder: Lefteris Papachristos

Oppdragsgiver: Tempero Energitjenester AS

Stikkord: BWU, Webutvikling, beregningsmodeller, digitalisering, design, utvikling

Sider: 66

Antall vedlegg: 5

Dette prosjektet fokuserer på effektivisering gjennom digitalisering av en av konsultentselskapet Tempero Energitjenester AS sine tjenester: deres beregningsmodeller. Temperos nåværende håndtering av dette skjer ved at kundene deres kontakter dem hver gang de trenger tjenesten deres. Dette prosjektet har som mål å gi Tempero en applikasjon som vil tillate kundene deres å bruke denne tjenesten digitalt, og dermed øke effektiviteten på både Temperos og kundenes arbeidsplass. Dette prosjektet er basert på et tidligere arbeid, hvor innsikt og forskning ble samlet inn. I denne rapporten presenteres prosessen med å designe og utvikle et minimumprodukt kalt “Tempero Beregningsmodeller” gjennom en designfase som følger prinsipper om brukervennlighet og tilgjengelighet, konseptet Reisedrevet design, samt en utviklingsfase ved bruk av MERN Stack. Tempero Beregningsmodeller er en applikasjon som gjør Temperos beregningsmodeller tilgjengelige for deres kunder.

Foreword

We would like to thank Tempero Energitjenester AS for giving us the chance to work on this exciting project, our supervisor, Lefteris Papachristos, for his guidance, and each other for being supportive and encouraging throughout the project.

Last but not least, we'd love to thank the cats in our neighborhood for bringing a smile to our faces during challenging times.

Gjøvik, 13 May 2022.

Lena Andal-Larsen

Vilde Gunnes Berntsen

Table of Contents

1 Introduction	1
1.1 Project description	1
1.2 Problem statement	3
1.3 Structure of the report.....	3
2 Background.....	4
2.1 Results from earlier work	4
3 Project organization	5
3.1 Organization tools.....	5
4 Understanding	7
4.1 Interview and survey	7
4.2 Payment methods.....	8
4.3 Functional requirements	9
5 Design	12
5.1 Non-functional requirements	12
5.1.1 Journey driven design	12
5.1.2 Usability principles	13
5.1.3 Accessibility principles	13
5.2 User testing	13
5.3 Heuristic evaluation	14
5.4 Low-fidelity prototype.....	15
5.4.1 Iteration one and two.....	15
5.5 High-fidelity prototype	17
5.5.1 Iteration one and two.....	17
5.5.2 Iteration three	21
6 Development	25
6.1 Technology stack	25

6.2 Tempero's Calculation models	26
6.3 Front-end.....	28
6.3.1 Folder structure	31
6.3.2 Sass	31
6.3.3 Redux	32
6.3.4 Axios	33
6.3.5 Toast.....	33
6.4 Back-end.....	34
6.4.1 Authentication and Authorization.....	34
6.4.2 API	34
6.4.3 Email system.....	35
6.4.4 Database.....	36
6.5 Stripe.....	38
7 The application "Tempero Beregningsmodeller"	41
7.1 User-guide	48
8 Discussion.....	50
8.1 A good user experience	50
8.1.2 Heuristic evaluation	51
8.1.3 Accessibility.....	52
8.1.4 Conclusion to sub question 1	53
8.2 Guidance in the system.....	53
8.2.1 Conclusion to sub question 2	53
8.3 Safe payment method	53
8.3.1 Conclusion to sub question 3	54
8.4 Sustainability	55
9 Conclusion.....	57
9.1 Further development.....	57

10 References	60
11 Figures	63
12 Tables.....	65
13 Appendix	66

1 Introduction

Digitalization is a term that has many different definitions. One definition can refer to social life being reconstructed into a digitalized environment, another a business being moved into the digital sphere (Bloomberg, 2018). In this report, the term is referred to as a business utilizing technology in areas like the work process, so that they can become more efficient.

Some of the benefits of digitalization are increased efficiency, increased productivity, and, if done correctly, a better user experience for the customer. The process of digitalization can take a lot of time and money, and if done badly, the customers and employees that are going to use the new system, may be unsure on how to use it and if they have the skills needed to do so (Bossard, 2020).

The purpose of this project is to digitalize the consulting company Tempero Energitjenester's current way of doing calculations for their customers. This will be done by making a user-friendly application where they have the ability to rent and use calculation models online.

1.1 Project description

This project's owner is Tempero Energitjenester AS. Tempero is a small consulting company, founded in 2012, with customers in Norway, Sweden, and Denmark. Their largest business area is finding solutions for energy supplies, efficient energy, and power use in buildings, as well as indoor climate.

Tempero works with both new constructions and restoration of existing buildings. Their role is to be the client's "competence" in meeting with contractors and consulting engineering firms in construction projects. To do this Tempero has developed a number of calculation models in Excel for calculating energy supply, heat, ventilation, and energy consumption in buildings, as shown in Figure 1 (Amdal-Larsen et al., 2021).

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1		LUFTHASTIGHET I VENTILASJONSKA Snutt: 263, versjon 1																			
2		Merk: For å unngå støy bør hastigheten i ventilasjonskanalen være lavere enn 2,5. Hastighet 3 kan aksepteres om andre ting ligger til rette																			
3																					
4			$V = A * U$																		
5			V = volumstrøm																		
6			A = areal på kanal																		
7			U = Hastighet på luft																		
8			d = diameter kanal m2																		
9																					
10			$V \text{ (m3/h)} = V/3600 \text{ (m3/sek)}$																		
11			V (m3/h) 300 Fyll inn luftmengde m3/h																		
12			A =			$A = \pi * r^2 = \pi * \text{diameter}^2 / 4$															
13			U (m/s) =			$U \text{ (m/s)} = V/A = (V/3600 \text{ [m/sek]})/(\pi * d^2/4) = 4 * V/3600 * \pi * d^2$															
14			d (m2) 0,2 Fyll inn diameter på kanalen																		
15			U = 2,65																		
16																					
17																					
18																					
19			Jo lavere lufthastighet, jo mere blir ventilasjonsluften påvirket av omgivelsene der kanalene ligger																		
20			Jo større dimensjon på kanal, jo mere blir luften påvirket av omgivelsene der kanalen ligger																		
21			Jo mindre kanal, jo høyere SFP (med samme luftmengde)																		
22			Isolering av kanaler medfører virkning på byggets totale klimagassutslipp (materialer), men vil også medføre spart energi pga lavere varmetap (og mindre kjøling)																		
23			NB! Det finnes svært mye annet som kan påvirke om det blir støy fra ventilasjonen.																		
24			Se Byggforsk kunnskapssystem - byggdetaljblad 552.306																		

Figure 1 - Example of one calculation model in Excel made by Tempero

In the current solution, if a customer needs to do a calculation, they have to ask Tempero for the specific calculation model. The customer then gets sent the Excel sheet they need, or Tempero will do the calculation for them, using the sheet, and give them the answer.

In the interview with the project owner, it was revealed that Tempero believes that with the right input data, these models can be used directly by their customers without them needing to consult with Tempero, especially for smaller projects. This way their customers' projects can be completed much faster since the person responsible for gathering the information from Tempero, alone can get the basic data needed to directly contact suppliers, electricians, plumbers, etc. to finish their ongoing construction projects.

Tempero wants to digitalize its current solution and make the calculation models available through an application. Tempero's requirement for the application is that their current customers should be able to rent calculation models and use them to get the answers they need. They also want to display information that easily explains what each model calculates. Some of the models are complex, so their last requirement is to implement a way for the customer to contact Tempero through the application if they need guidance (Amdal-Larsen et al., 2021). To begin with, there will only be three calculation models in the application, but if it gets used, more can be added later.

1.2 Problem statement

The problem statement focuses on fulfilling Tempero's and the end-users' wishes for the system. The statement is:

How can we design and implement a secure and user-friendly application that makes calculation models from Tempero Energitjenester AS available for its clientele?

To better answer the problem statement, the group created three sub questions:

1. How can we create a good user experience for the users using the application?
2. What is the best way for the end-user to contact Tempero Energitjenester AS for guidance when it comes to using the calculation models?
3. What is a safe way to implement a card payment method in the application?

Each sub question focuses on different aspects of the application. Sub question 1 focuses on how to give the users a good user experience while using the application. Sub question 2 is asked because some of the calculation models are very complicated, so a way for the customers to ask Tempero for guidance is needed. Sub question 3 is asked because Tempero does not want to give away these calculation models for free, the application needs a safe way for the users to pay.

At the end of this project, we will have a minimum viable product (MVP) that Tempero's customers can sign up for and use. The product was established through interviews with the project owner and a survey of the end-users.

1.3 Structure of the report

Chapter 1 introduces the project and the problem statement. Chapter 2 talks briefly about what the group did in the previous semester relating to this project. Chapter 3 presents how the project was organized. Chapter 4 shows the group's understanding of the project, and what was required before starting with the design and development process, which is presented in chapter 5 and 6. Chapter 7 shows the final product, and the result will be discussed in chapter 8. Lastly, chapter 9 will have a conclusion to the problem statement and proposals for further development.

2 Background

This project was started in the fall semester of 2021, in the course *IDG3101 Fordypningsprosjekt*. The findings made during the course acts as the foundation of this project. One of the members from the original group left, but the remaining group has gotten permission to use their findings.

2.1 Results from earlier work

The goal of the course *IDG3101 Fordypningsprosjekt* (see appendix A of Amdal-Larsen et al., 2021) was mainly to research the elements needed in order to develop a system for both renting and using Tempero's calculation models, and make a lo-fi prototype of that system.

The group focused on:

- How to give the users of the system a good user experience?
- What is the best way for Tempero to give their customers guidance through the system, when they need help with calculations related to their projects?
- What is a good and safe payment method for the system?

The group found answers to these questions through interviewing the project owner, surveying the end-users, and by researching user experience, a variety of different payment methods and Tempero's current calculation models in Excel.

From this insight, the group made an "MVP vs nice-to-have" diagram and came up with the functional requirements for the application. From this, a lo-fi prototype was made. Parts of this report will include the findings and results from this earlier work.

3 Project organization

The group for this project consists of two members. The way the group worked was a mix of physical and online meetings. Meetings with the project owner were held online, over Teams, because he lives in another city, while the meetings with our supervisor were mostly in person. Chapter 3.1 explains how the group worked together on this project, and what tools were used to make it as smooth as possible.

3.1 Organization tools

Good communication and routines are crucial for such a big project, so the first thing the group agreed on was a working schedule, Monday-Friday 10:15 - 16:00. The main way of communication within the group has been over Discord, a platform both group members are familiar with.

A server was made for this project (see Figure 2) where the group shared information in the chat channels and held meetings using the voice channel. Discord is a great communication platform, it lets the users store notes, videos and other helpful information organized in one place, through channels in the server. Files and documents that was too large to store in Discord got stored in a Google Drive. This let the group share documents and work on them together when needed.

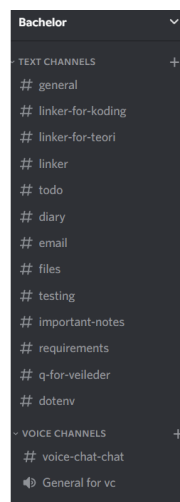


Figure 2 - Our Discord server for this project

The version control system GitHub was used to store the group's code. The way the group collaborated on the code was using the Live share extension in Visual Studio Code. It lets one person host the code while others can read and write the same files. The group members

A Gantt chart made with Team-Gantt was used to keep track of everything the group needed to do (see Figure 3). The chart was filled with tasks, milestones, and deadlines, and in what order they needed to happen. The chart also showed which group member was assigned to do the task which helped the group keep track of who did what.

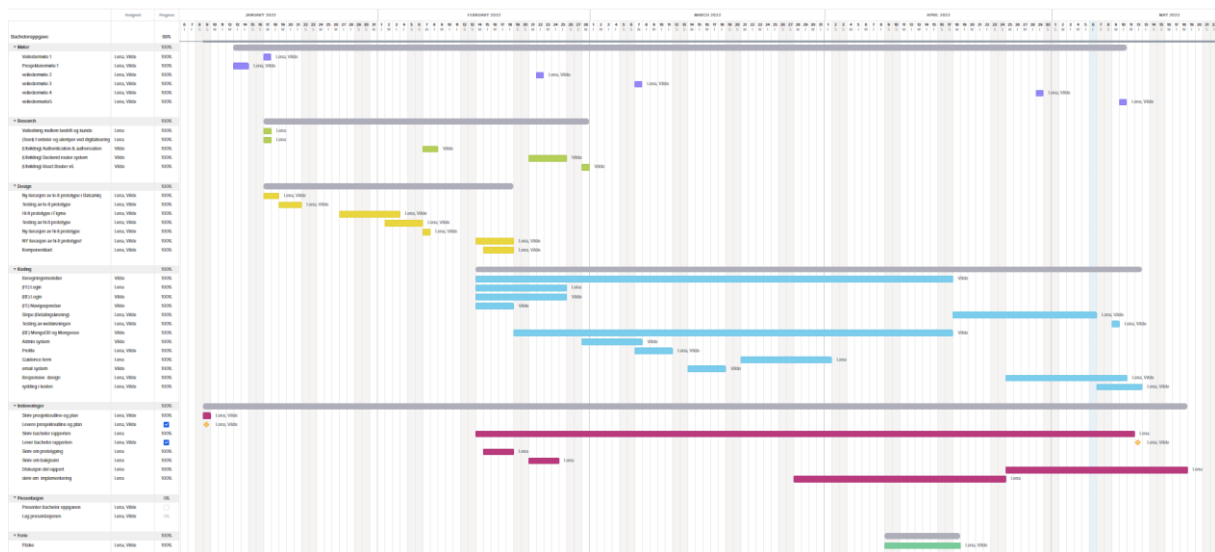


Figure 3- Screenshot of the Gantt chart for our project

4 Understanding

Before making an application, the group needed to gather information to better understand what the project owner and end-users wanted. Chapter 4.1 will show the results from the interview of the project owner and the survey taken by the end-users. Chapter 4.2 have research of different payment methods the group considering using for the system. Based on these findings chapter 4.3 have an MVP vs nice-to-have table, and a table with the functional requirements for the application. These chapters contain content from the report written in the previous semester.

4.1 Interview and survey

In the interview with the project owner, the group identified three user groups: Tempero admins, Firm admins, and Under users. The Tempero admin is an employee from Tempero, the Firm admin is a person from a firm that Tempero Energitjenester AS already has as a customer, and the Under users is other people from the same firm.

Some other things the project owner wanted for the system was:

- The design of the application to correspond to professionals; it shouldn't hinder the efficiency of the user.
- The application should be in Norwegian.
- A way to give the users guidance if needed.
- A subscription-based renting system, where the user can rent the calculation models for a limited time period.

Based on the information the group received from the interview, a survey was made and sent out to Tempero's customers. This was done to figure out who the user group is, and what their needs and preferences would be for a system like this. Seven people answered the survey.

The results were as follows:

- 7/7 of the respondents believed they would benefit from a system for renting and using calculation models online.
- The majority of the users are in the age group 60+.

- 7/7 said that they would use the system on the computer, and one person also said they would use it on a mobile phone (they could choose multiple options).
- Six people wanted guidance through face-to-face conversations, one through phone calls, three through email, and no one through a chat (they could choose multiple options).
- 7/7 felt safe using Vipps, card, and invoice as a payment method online.
- The majority wanted a light-colored website with minimal styling.

4.2 Payment methods

Being able to rent calculation models is a big part of this system, so some research into the most common payment methods in Norway was done. Based on the answers from the survey, four payment methods stood out from the rest: Vipps, Card payment (Stripe), PayPal, and Klarna. Table 1 shows a summary of the different methods.

	Vipps	Stripe	PayPal	Klarna
Price	* Payment only for use * 2.99% of the purchase price per transaction	* Payment only for use * 2 NOK fixed fee per transaction * 2.4% of the purchase price for Norwegian bank cards * 2.9% of the purchase price for international bank cards	* Prices vary based on selected service * 2.80 NOK fixed fee * 1.9% of the purchase price for buyers without an account * 3.4% of the purchase price for buyers with an account	* N/A Price based on company agreement.
Businesses can pay through the service	No	Yes (with a company card)	Yes	No
Open use without an agreement with the company	No	Yes	No	No

The customer must register an account via the service	Yes	No	Yes/No*	No
--	-----	----	---------	----

* Some of the solutions for PayPal do not require an account for payment, but the use of the portal, etc. will demand it.

Table 1 - Summary of different payment methods made by Ida M. R Gjeitsund (Amdal-Larsen et al., 2021).

Stripe (card payment) was chosen for the application because all the other methods had clear problems the group could not overlook, such as businesses not being able to pay through the service. Card payment was also one of the methods the end-users were comfortable using (Amdal-Larsen et al., 2021).

4.3 Functional requirements

Based on the interview with the project owner and the answers from the end-users an “MVP vs. nice-to-have” table of the system, shown in Table 2, was made. It differs what the system must have to work, from extra functionalities that would be nice to have, and which type of user it affects.

Users	MVP	Nice to have
Everyone	<ul style="list-style-type: none"> *Authentication and authorization *New firm request *Navigation *Responsive design 	<ul style="list-style-type: none"> *Change language *Change password *Dark/light mode *User Profile *Information page *Forgotten password
Tempero Admin	<ul style="list-style-type: none"> *Overview of all the calculation models *Overview of firms in the system *Accept/decline firm requests *Inbox 	<ul style="list-style-type: none"> *Change prices of the models *Add more Tempero admins
All customers (Under users and Firm admins)	<ul style="list-style-type: none"> *Contact form for guidance *Ability to see and use the rented calculation models 	<ul style="list-style-type: none"> *History of the calculations
Firm Admins	<ul style="list-style-type: none"> *Ability to rent models 	<ul style="list-style-type: none"> *Invoice

	*Card payment *Receipt *Ability to add Under users	*Buy models in packages *Preview of the models *Edit firm information
--	--	---

Table 2 - MVP vs Nice-to-have (Amdal-Larsen et al., 2021).

From the MVP table, the group made a sitemap of the application (see Figure 4). This helped visualize the information architecture of the system before the prototyping process started.

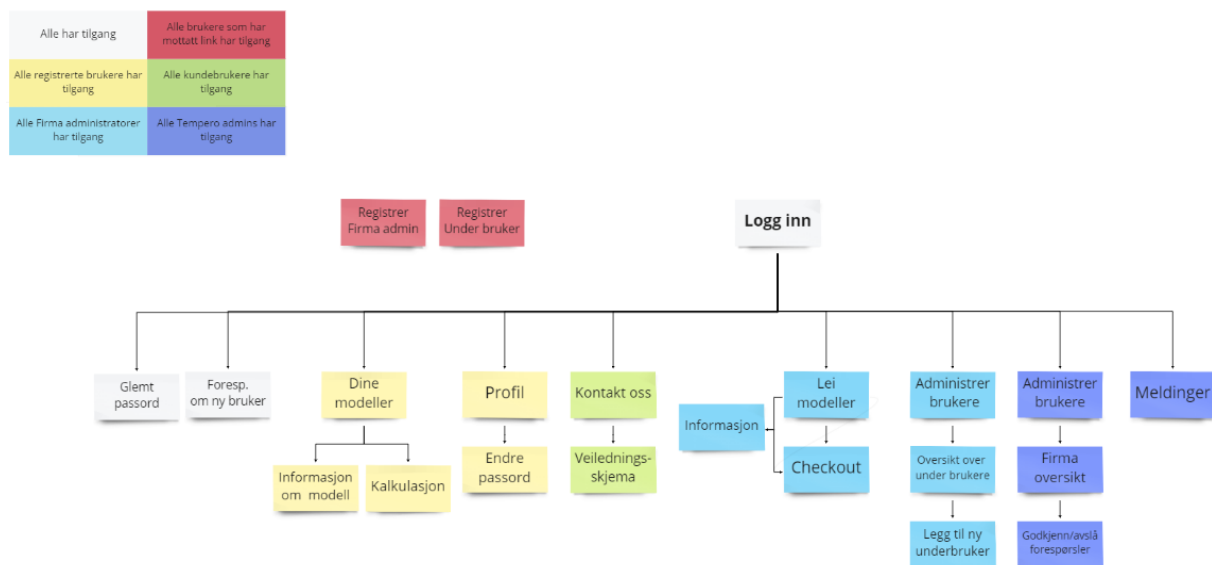


Figure 4 - Sitemap that shows what the different users have access

From the MVP table and the sitemap, the group came up with requirements necessary for the system to work. Table 3 below shows a summary of them.

Requirements	How it works
Registering of customers	* The user needs to apply for permission to create a user before registering. Tempero does not want to have unknown companies in the system * If Tempero approves; the user will get an email with a registering link
Type of users	* Tempero admin, Firm admin, Under user
Calculation models	* Will show an overview of every calculation model available to rent * The main page of the system will be “Dine Beregningsmodeller” which shows the calculation models the firm is currently renting

Renting calculation models	<ul style="list-style-type: none"> * Only Firm admins can rent calculations models, but the Under users can see and use the models their firm is renting * The Tempero admin can see and use all of the models * The models will be available to rent in increments of weeks or months
Payment method	<ul style="list-style-type: none"> * Payments happen through card using the payment service provider Stripe
Guidance	<ul style="list-style-type: none"> * The guidance will happen through a contact page with contact information and a form, so that the customers can contact Tempero in different ways * Tempero Admin have an inbox where the messages sent from the contact form will be displayed
Administration	<ul style="list-style-type: none"> * Administration page where Firm admins can invite Under users and have an overview of all the users in their firm * Administration page where Tempero admins can approve applying firms the right to register and have an overview of all the firms in the system

Table 3 – System requirements (Amdal-Larsen et al., 2021).

5 Design

After getting a better understanding of what the system should look like, the design process started. Two types of prototypes were made, low fidelity, and high fidelity. The lo-fi prototype is made in Balsamiq and the hi-fi prototype in Figma. During the prototyping process, different iterations of the prototypes were made, based on the feedback from the user tests held after every iteration.

The lo-fi prototype had two iterations and hi-fi had three. For the last two iterations of the hi-fi prototype, a UX expert did a heuristic evaluation. By doing the user tests and evaluation during the prototyping phase, the group saved time, because it's more efficient to make changes to a prototype than the coded application.

In this part of the report chapter 5.1 will show the design concepts the group used for the project. Chapter 5.2 will explain how the group did the user testing, and chapter 5.3-5.4 have the results of the different iterations of the lo-fi and hi-fi prototype.

5.1 Non-functional requirements

While designing the prototypes, the group had some design concepts in mind. Following these helped making the prototypes better. They will be discussed in connection with the project in chapter 8.

5.1.1 Journey driven design

In recent years, it has become very popular to design for "mobile-first", probably because more than 3.8 billion people worldwide owns a smartphone in 2022 (Statista, 2022). One reason for doing "mobile-first" is that a small screen forces the designer to prioritize the most important content, which then can be scaled upwards when designing for tablets and desktops, making a responsive design which provides the users a better experience (Morales, 2021).

An alternative to this method is "Journey driven design". The first step in "Journey driven design" is to uncover the journey, this starts with the research phase (Mesibov and Levin, 2017). The way this is done is by asking the end-users what device they would want to use the application on and where they would use it.

The next step is to design the journey. Here the designer needs to ask:

- Have I designed for the most critical device?
- What other devices might come into play here?
- What is the context for this interaction? Where will the user physically be?

After each iteration of the prototype, one should do user testing to make sure it works as intended.

5.1.2 Usability principles

Jakob Nielsen, one of the creators of the Nielsen Norman Group, has presented ten general principles for good interaction design. They are heuristic, i.e., simple procedures or strategies, and not specific rules you must follow (Nielsen, 2020b). These include among others: visibility in system status, consistency and standards, error prevention, and aesthetic and minimalist design. Following these heuristics will make the user experience better and less confusing.

To assure the designer is doing it right, it is possible to have an interaction design expert do a heuristic evaluation to reveal insight that can help the designer make the interface more usable (Interaction Design Foundation, 2022).

5.1.3 Accessibility principles

In Norway, we have laws in place that will ensure that all web applications are adapted for as many people as possible. «Regulations on universal design of information and communication technology (ICT) solutions» (2019), requires that all network solutions must meet 35 specific criteria in the standard "Guidelines for accessible web content (WCAG) 2.0" (W3C, 2011a).

WCAG's goal is to “provide a single shared standard for web content accessibility that meets the needs of individuals, organizations, and governments internationally” (Initiative (WAI), 2022).

5.2 User testing

According to Jacob Nielsen, testing on more than five people is enough to identify the majority of usability issues, because you keep seeing the same things over and over again and won't learn anything new (Nielsen, 2000a). The group did user tests on five people, mostly in the age range 60+ as the survey showed that this was the main target group. In addition, these people had different technological skills.

The user testing was done after each new iteration of the prototype, through “scenarios”. Five different scenarios were made, which together would guide the test person through a normal use of the application. The testers were observed while they went through them. The feedback from these tests helped the group improve the design.

The five scenarios consisted of tasks that were: realistic, actionable, and not too descriptive, as this would be too leading (McCloskey, 2014). Table 4 below shows the five scenarios used in the test.

#	Scenario
1	Send a request to have your company registered in the system. Then register a new user (Firm admin) and log in
2	Find and rent the calculation model named “Lufthastighet i ventilasjonskanaler og støy”, and rent it for 1 week
3	Find the calculation model you just rented (“Lufthastighet i ventilasjonskanaler og støy”) and use the model to do a calculation
4	Find the contact form page and send a message to Tempero, requesting guidance
5	Add a user from your firm (Under user) into the system. After you have done this, log out

Table 4 – The five scenarios used in the user-tests

5.3 Heuristic evaluation

Heuristic evaluation is a process where a usability expert measures the usability of the user interface. This evaluation is done by using already established heuristics, to reveal insight that can help the designers make the product more usable (Interaction Design Foundation, 2022).

Since usability experts were not available to us, we used an interaction design student at NTNU that was well versed in the usability heuristics to act as an expert. They did the evaluation for this project on the two final iterations of the hi-fi prototype. The evaluation was done by having the expert go through Nielsen’s 10 heuristics and as problems within each heuristic were discovered, they were documented (Nielsen, 2020). These problems were also given a level of severity ranging from 1 to 4, 1 being a “cosmetic problem” and 4 being a “usability catastrophe”.

5.4 Low-fidelity prototype

The lo-fi prototype was made in Balsamiq, which is a web-based tool where the user can make simple sketches. The first iteration of the prototype was made for a computer screen, as this was the most critical device based on the feedback from the end-users.

The context the end-user will be in when they use the application is either in an office or at a building site. It is easier to bring a phone to a building site, so a mobile version of the application is needed. Having a responsive design, meaning a system that looks great on all devices, is also very important for the user experience. The second iteration of the lo-fi prototype is made for mobile devices.

5.4.1 Iteration one and two

Based on our understanding of the system, the lo-fi prototype was made (see Figure 5). The first iteration, made in the previous semester, included the application's main functions: The request and registering pages, the overview of the available calculation models and the use of a specific model.

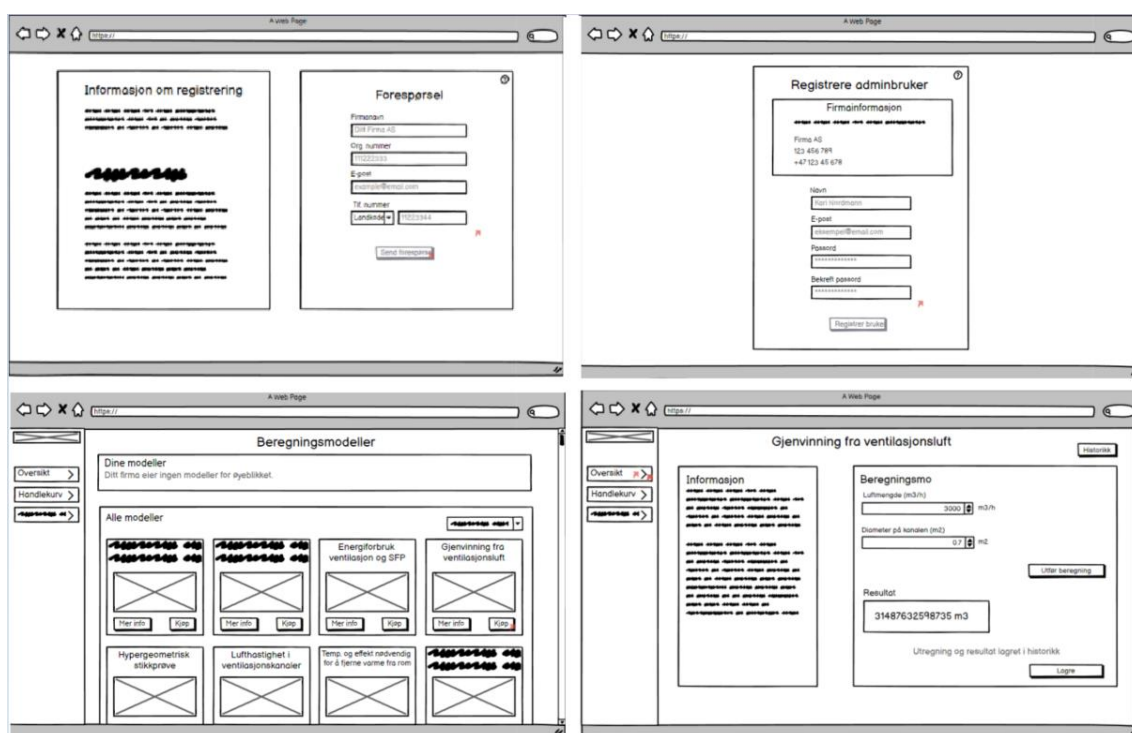


Figure 5 - From the left: Add a new firm request form, register Firm admin form, an overview of all the calculation models, using a specific model from the lo-fi prototype in Fordypningsprosjekt

After the first iteration was made, it was user tested. The group went through the feedback received from the testers and the observations done under the tests and discussed which parts

of the prototype were good and bad. This was put into a simple diagram, shown in appendix B.1.

Some of the changes made from iteration one to two (see Figure 6 below) were: to include a back button on each page, add a logout button, change the name of the main page from "Oversikt" to "Dine Beregningsmodeller" and create an indication in the menu so the user can see which page they are at.

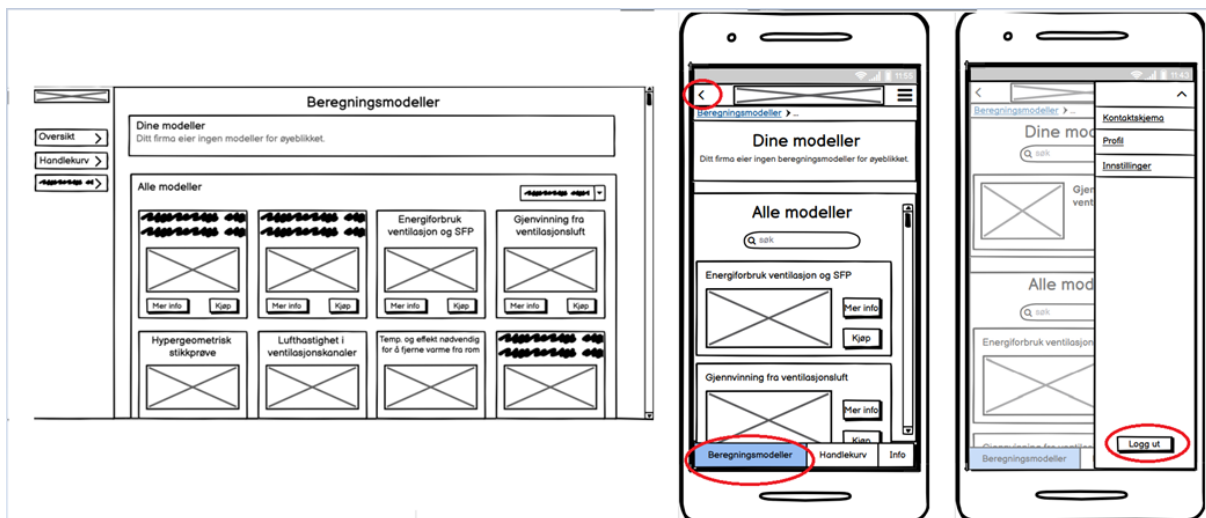


Figure 6 - Changes made from iteration one (left) to two (right)

The first iteration of the prototype was made to fit a computer screen because of the group's choice of following the design principle "Journey driven design". The second iteration was made for mobile to check if there was enough space for the most important content on a smaller screen before starting hi-fi prototyping. Because of the already simple design, the content had no problem fitting a mobile screen.

After the second iteration was done, the prototype was user tested once again. Appendix B.2 shows a diagram with feedback from the testers. The main things they wanted to change were the text on the buttons of the models from "buy" to "rent", the hamburger menu was too small, the firm request needed to look like a link, and the most important text on the application needed a bigger font. After two iterations of the lo-fi prototype, the next step in the design process was to make a hi-fi prototype. The first iteration of the hi-fi prototype was based on the feedback from the second user test of the lo-fi prototype.

5.5 High-fidelity prototype

To create our hi-fi prototypes, the group utilized the application Figma, which is a web-based graphics editing and user interface design app. The colors of the prototype followed a simple color palette of white and blue, the shade of blue being the same as the font color on Tempero's logo, as seen in Figure 7.



Figure 7 - Tempero's logo, from 2012

“Arial” was chosen as the font for the site, as it is listed as being accessibility friendly (Siteimprove, 2022). The font is kept well above the minimum font size of 12 pixels and 16 pixels for body text, recommended by WCAG (W3C, 2011a).

The design of the prototype follows the preferences of the target group, gathered from the user survey carried out by the group in the fall semester of 2021. It gave some general insight into what the target group preferred when it came to website design. They wanted a light-themed site, with simple styling. In other words, a site for efficient work (Amdal-Larsen et al., 2021).

5.5.1 Iteration one and two

The system has three types of users: Firm admin, Under user, and Tempero admin. These users have access to different pages and see different things on the site. Because of this, one prototype for each user was made. Figure 8 shows what content the different users have.

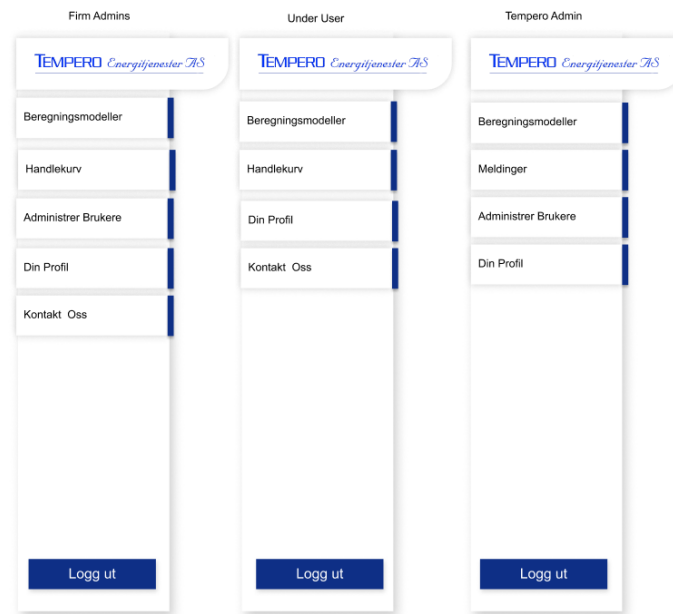


Figure 8 - The navigation bars of the different user types, from iteration one

The main pages in the hi-fi prototype are the login and register pages, an overview of the calculation models the user owns and calculation models available to rent, the calculation page where the model gets used, the user administration page, and the contact page.

The hi-fi prototype was tested through scenarios, same as with the lo-fi prototypes. Appendix B.3 shows the diagram with the feedback from the first user test. The second iteration was made based on this feedback. The following part of the report shows the results, and what was changed from iteration one to two.

The biggest change made was to split “Beregningsmodeller” into two different pages: “Dine Beregningsmodeller” and “Lei Beregningsmodeller” (see Figure 9). This made each page less packed, and it is easier to see which models are owned by the user and which ones aren’t. When the user logs in, they now land on the page: “Dine Beregningsmodeller”.

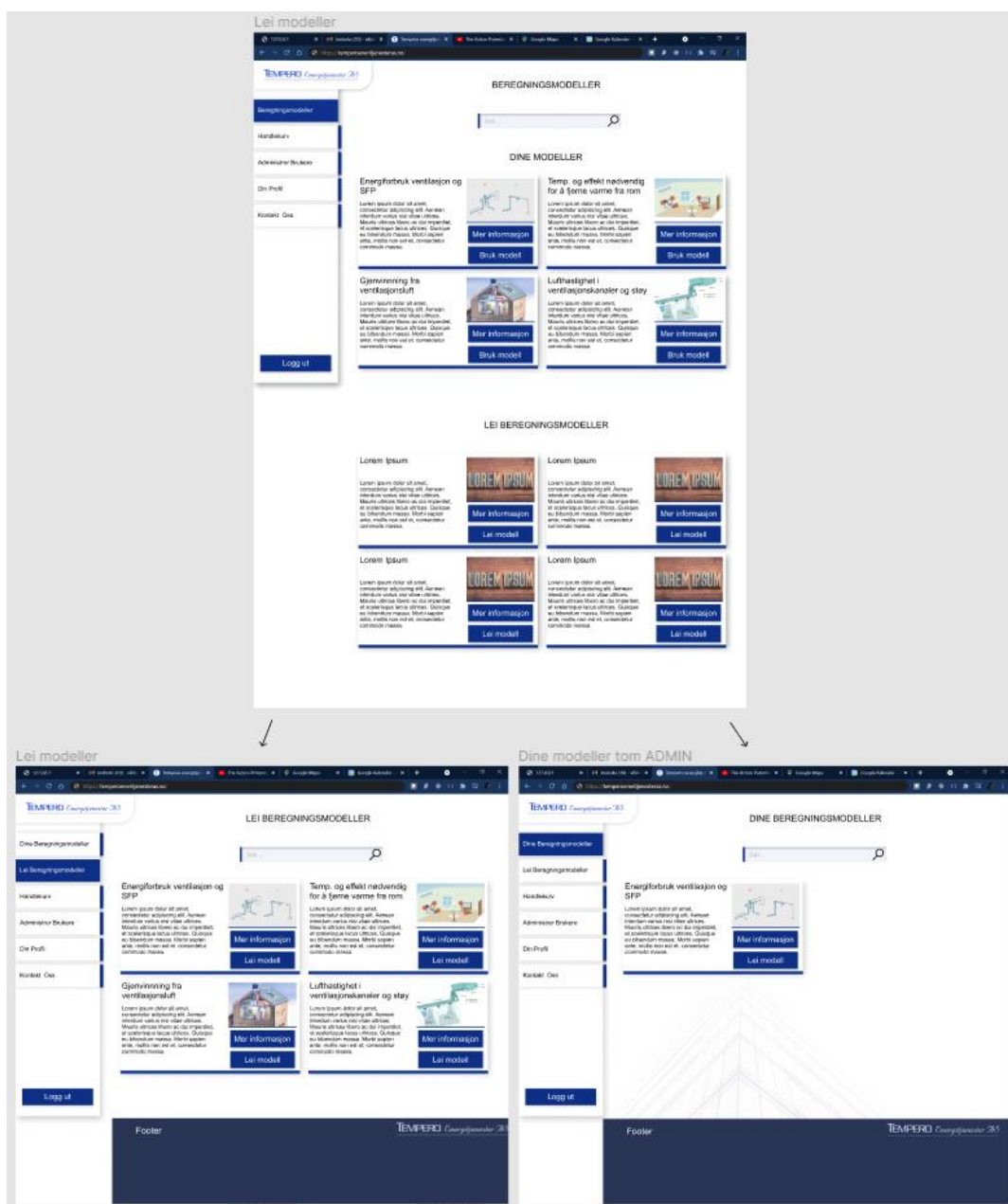


Figure 9 – Models page split into two; one for owned models, one for rentable models

Another big change was adding categories to both the model pages. Tempero has three different groups of calculation models: Energy services, Productivity, and Projects. This change makes it easier for the users to find what they need (see Figure 10). It will also make the page less crowded since the user has the ability to collapse the categories and hide the type of models they don't want to see.

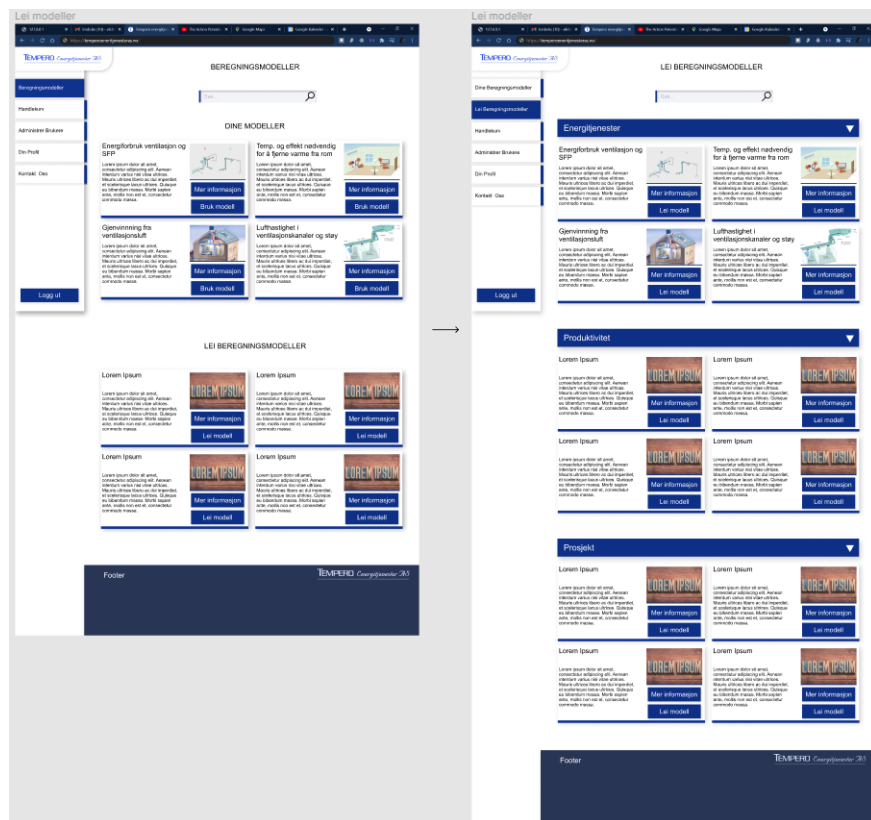


Figure 10 – Changes done to the model overview page

Some smaller changes included adding an email, phone number, and a job title in the Firm admins registering page so the Temporo admin can see what kind of job the users have (see Figure 11). These three things will be helpful to know for Temporo when guidance is needed.

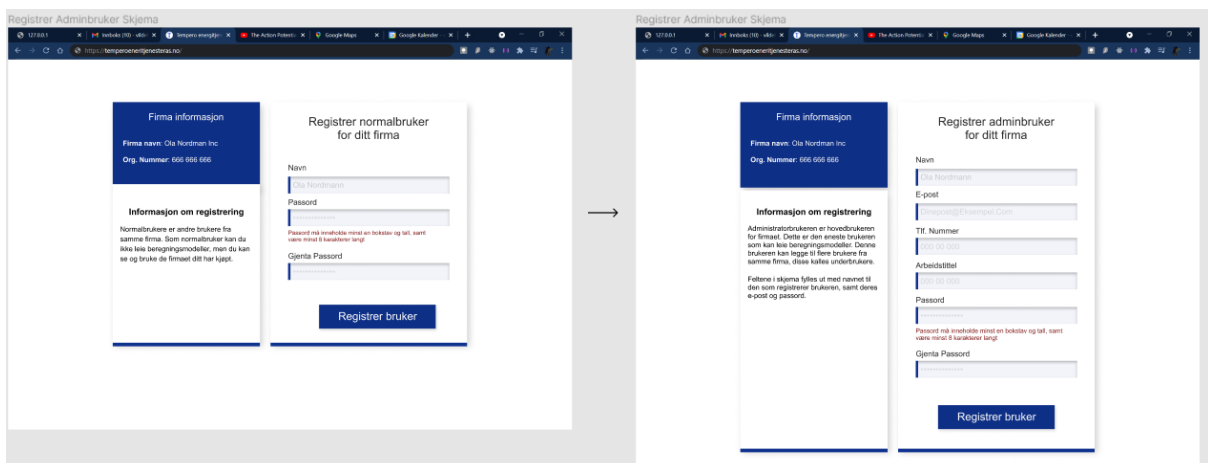


Figure 11 - Added phone number, email, and job title input fields

The “history” button, used for getting to the calculation history of specific models, was changed from grey to blue (see Figure 12), to make it more consistent with the rest of the site since that was the only grey button anywhere in the design.

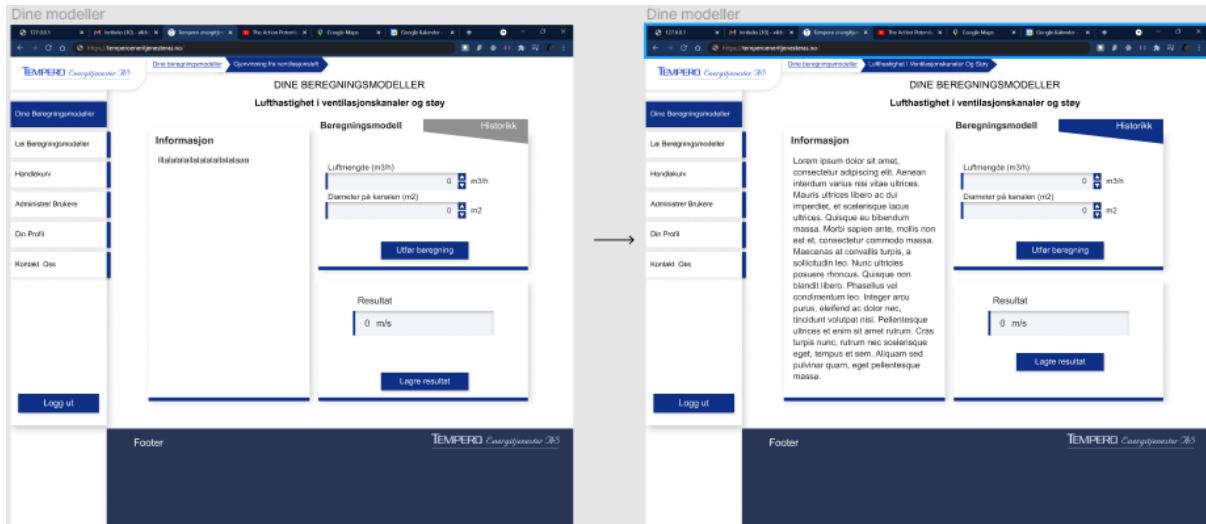


Figure 12 - Changed the history button from grey to blue

The trashcan icon, showing that something could be removed or deleted, was changed to a red box with a white X in the middle, which made it take up less space on the page (see Figure 13 below).

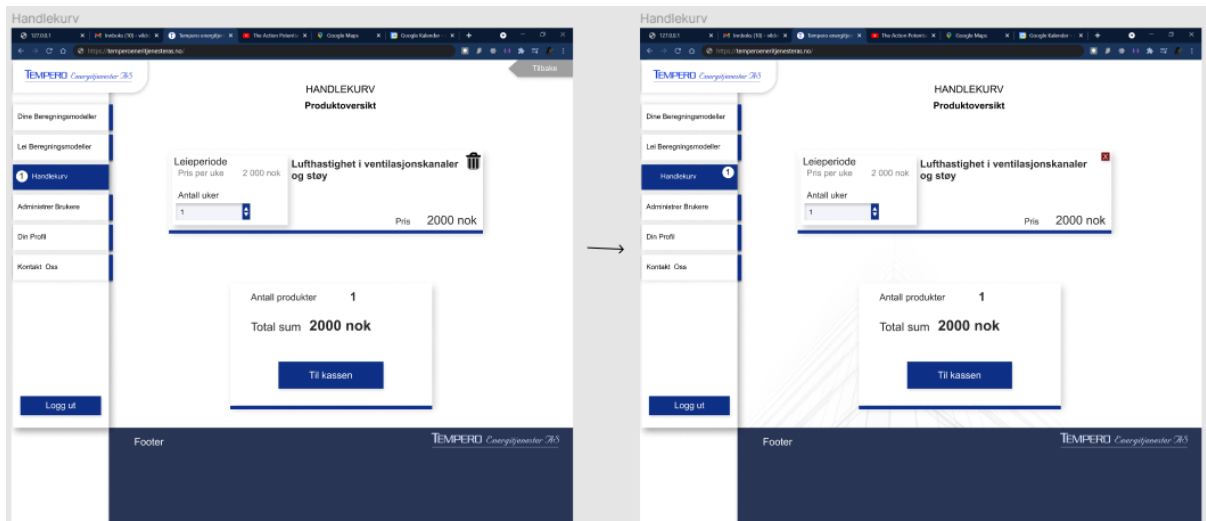


Figure 13 - Changed trashcan to a red box with a white X

5.5.2 Iteration three

The third iteration was the final one the group made. After iteration two was done, it was user tested once again. However, this time the testers didn't have much feedback. Instead of

making another iteration based on the small amount of feedback, the group arranged a heuristic expert evaluation to see if the interface was usable enough.

Appendix C shows a table with the results from the heuristic evaluation of the second and third iterations. All the problems that were found were on a severity scale of 1 and 2 (cosmetic problems and minor usability problems).

In the evaluation, there were found three problems about heuristic #4 - consistency and standards, and six about heuristic #8 - Aesthetic and minimalist design.

5.5.2.1 Consistency and standards

The first two problems for heuristic #4 were some cosmetic inconsistencies. The first problem was the top of the navigation bar and the second was the history button. The logo in the navigation bar had the only rounded shape in all of the system and the history button was the only button that wasn't shaped like a rectangle. This was changed to match everything else in the final iteration (see Figure 14 and 15).



Figure 14 – Changes in the logo design

The third problem was a minor usability problem. In the navigation bar, the page the user is at is colored blue, but the history button was blue when the user was not on it. This could be a

reason for confusion for the users. The design was changed to be consistent with the navigation bar.

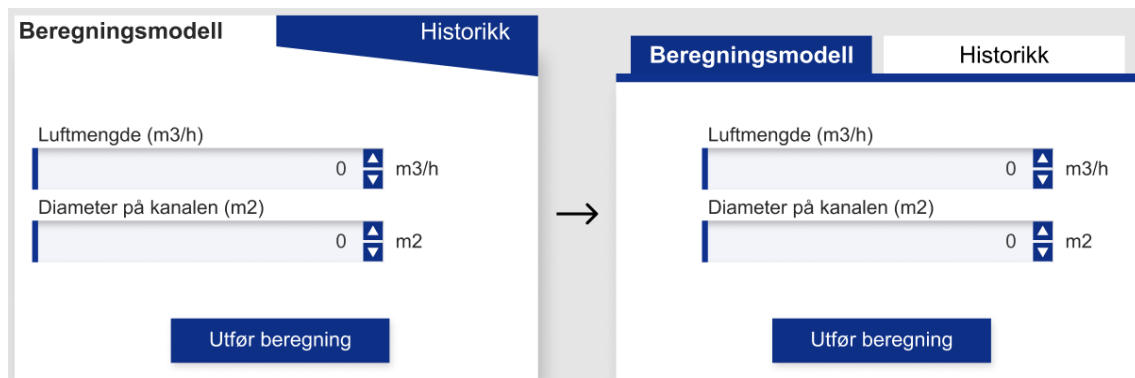


Figure 15 - Changes on the history button

5.5.2.2 Aesthetic and minimalist design

The interview and survey showed that both the project owner and the end-users wanted a simple and minimalistic design. A lot of information is needed to use the system and the calculation models, but it takes up a lot of space, and it will also give the users a bigger cognitive load, that can lead to information overload, making the user not want to read it at all (Krug, 2014).

The heuristic evaluation showed that the information might not be as needed as first thought. The first time a user uses the page, the information is good to have at hand, but when the user comes back to the same page for the 10th time, they already know what the information box says, and it will just be a distraction that takes unnecessary space.

To solve this problem the information was hidden behind an information icon or collapsible element (see Figure 16). This way the user can read it if they want to, while it won't take up a lot of space on the page.

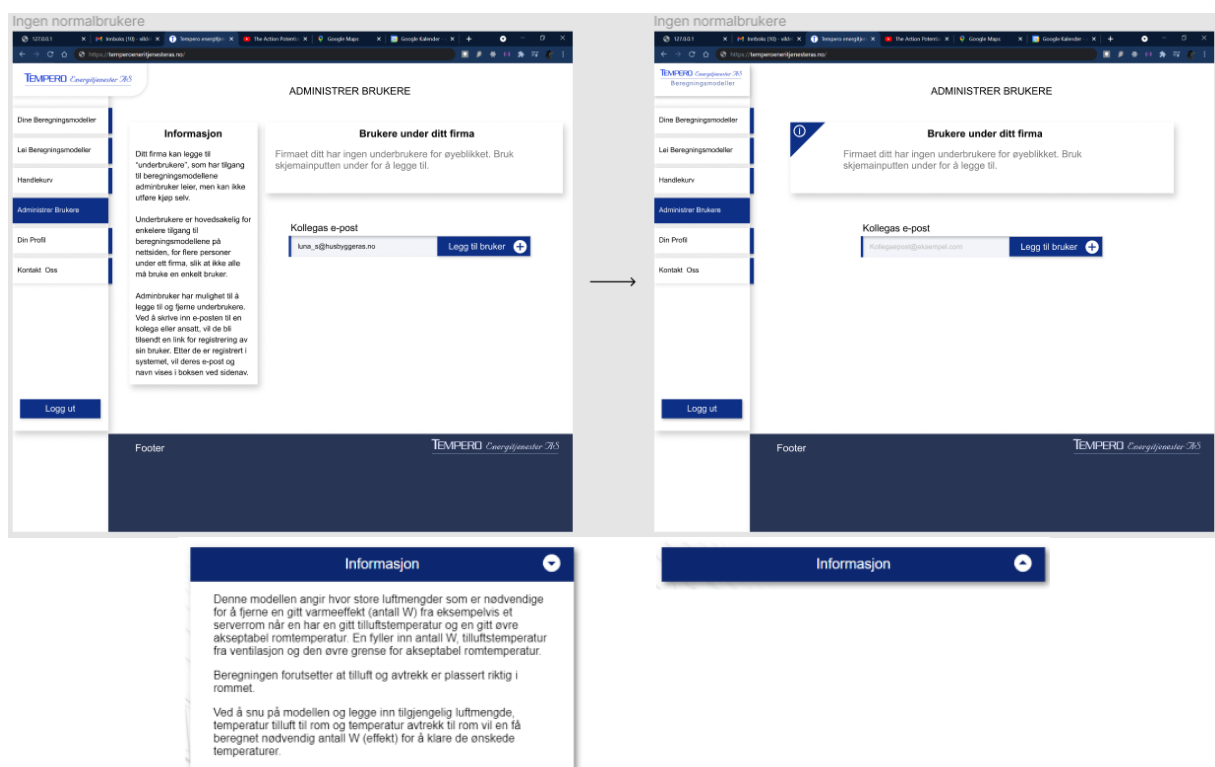


Figure 16 – Information box changes. The right picture shows an info icon the information is hidden behind. The bottom picture shows a collapsible element with information

6 Development

After the final iteration of the hi-fi prototype was made, the development process could start. In this phase, the group used the MERN stack and other technologies to develop the application (see chapter 6.1). The group's focus was on implementing the minimum viable product.

This phase was split into front-end (see chapter 6.3), back-end (chapter 6.4), and Stripe (chapter 6.5). The front-end is the user interface with which the users will interact, the back-end is the server-side, which the user can't see, where data gets arranged and stored (Kenzie Academy, 2020). Stripe is its own chapter as it requires both front- and back-end to be explained together.

6.1 Technology stack

The group discussed different alternatives that made it possible to build a full-stack application with only JavaScript and decided on using the MERN stack. The main reason for doing this was efficiency through prior experience with the technologies it consists of. Other similar stacks that were considered, was the MEAN stack which replaces React with Angular, and the MEVN stack, which replaces React or Angular with Vue.

MERN was chosen as our group is small, and time-usage is a priority. Choosing to stick to a familiar front-end framework, let us focus more on other parts of the project, like learning how to implement the payment service provider Stripe.

The MERN stack includes MongoDB, Express, React, and Node. Figure 17 shows the 3-tier architectural pattern MERN is made up of. It includes a front-end display tier, application tier, and database tier.

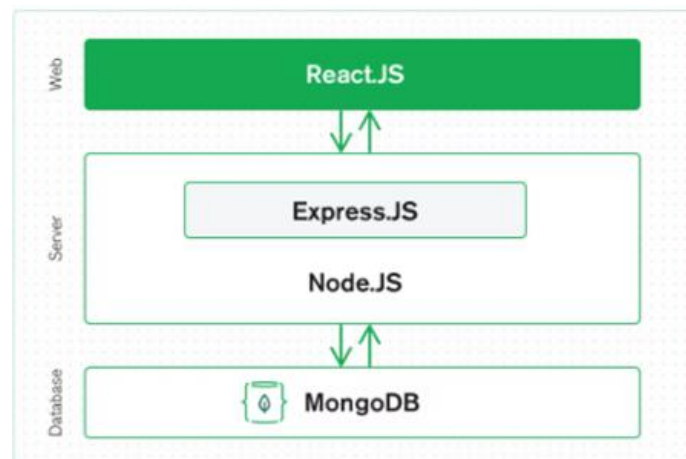


Figure 17 – MERN 3-tier architecture pattern (MongoDB, 2021a).

The top and front-end tier of MERN is made by using React.js, which was the most commonly used web framework in 2021 (Stack Overflow, 2021). It lets the developer make dynamic client-side user interfaces from simple components, connect them to the backend server and render them with HTML.

The middle level is the server tier with the server-side language Express running inside a Node.js server. This tier handles the server-side logic of the application.

The bottom tier is the database MongoDB which is needed for storing data. In addition to this, Mongoose was used to create a connection between the database and Express, and to make different schemas for the data needed (MongoDB, 2021a).

6.2 Tempero's Calculation models

The main function of the application is to use the calculation models. Tempero has a lot of models in Excel with different variations of data inputs needed to use them. These Excel sheets needed to be transferred to code, in a way that made them easy to use. To summarize, the group's findings from the previous semester, this included error limitation, and research into mathematical calculations in JavaScript.

Error limitation is part of making a good user experience. To limit the user's ability to make mistakes while doing a calculation the group used already existing functions in HTML.

All the input fields need to be of the type “number”, which makes it impossible for the user to input letters and other symbols into the fields. Each field also needs to have the attribute “required” so calculations can’t be done without filling in all the fields. Each calculation model has different minimum and maximum values that are logical for each field (see Figure 18). These values need to be set so the users can’t input data that is way off what it should be.

Modell	Input/felt	Min verdi	Max verdi	Enhet	
Temp. og effekt nødvendig for å fjerne varme fra rom					
	Watt (mengde energi) tilført	1	10 000 000	Watt	1000 Watt = 1 kW
	m ³ /h tilluft	1	500 000	m ³ /h	Kubikkmeter luft pr.time
	Temp. tilluft til rom	1	100	grader C	
	Temp. avtrekk i rom	1	100	grader C	
Gjenvinning fra ventilasjonsluft					
	Energipris kr/kWh	0	100	kr	Pris for 1 kWh
	Luftmengde m ³ /h	1	60 000	m ³ /h	Kubikkmeter luft pr.time
	Temp. avtrekksluft	1	100	grader C	
	Temp. uteluft	1	100	grader C	
	Gjenvinningsgrad i %	0	100	prosent	
	Lengde begrensingsperiode	1	8784	timer	8760 timer er et år. Men kan også bruke andre timetall
Lufthastighet i ventilasjonskanaler					
	V (m ³ /h)	1	60000	m ³ /h	Kubikkmeter luft pr.time
	d (m)	0,1	10	meter	Diameter på en sirkulær ventilasjonskanal målt i meter
Energiforbruk ventilasjon og SFP					
	SFP planlagt	1	10	P[kW]/V[m ³ /h]	Effekt som viften i ventilasjonsanlegget trenger dividert på luftmengde
	Luftmengde m ³ /h	1	50 000	m ³ /h	Luftmengde som skal gjennom ventilasjonskanaler pr. time
	SFP i aktuelle prosjekt	1	10	P[kW]/V[m ³ /h]	
	Timer i aktuelle tidsperiode	1	8 784	timer	8760 timer er et år. Men kan også bruke andre timetall

Figure 18 - Min and max inputs for each input calculation model made by Tempero

The group also found that JavaScript has a problem with floating point precision. Meaning it is imprecise when it comes to decimals in mathematical calculation (w3schools, n.d.). Figure 19 shows an easy calculation that gives the wrong answer.

```
> 16:08:10.359 0.1 + 0.2
< 16:08:10.367 0.30000000000000004
```

Figure 19 - Example of an easy calculation done in JS that gives the wrong answer

A solution to this problem, is to use some of JavaScript built-in functions in the calculation. As shown in Figure 20 a combination of «parseFloat()», «toFixed()», or «toPrecision()» will make sure the answer is correct. Having JavaScript do the math correctly is crucial for the calculation models to work as intended. With all this figured out the real development could start.

```
> 16:09:17.334 parseFloat((0.1 + 0.2).toFixed(2))
< 16:09:17.336 0.3
> 16:10:17.053 parseFloat((0.1 + 0.2).toPrecision(12))
< 16:10:17.060 0.3
```

Figure 20 - Example of an easy calculation done in JS, with functions giving the right answer

6.3 Front-end

The first step in the development process was to identify what could be made into components from the final iteration of the hi-fi prototype. The group identified what would be components and what would be “pages” in the application.

Components are JavaScript classes or functions that accept inputs (e.g., props) and returns a React element that describes how a section of the User Interface should appear (Kagga and Atto, 2020). Functional components were used to make the application because of their predictability and conciseness. Properties (props) are what make the React component dynamic and reusable. They provide a way of passing data down from one component to another (Kagga and Atto, 2020). Table 5 below shows the components the group has coded and explains the functionality and usage of them.

Component	Functionality	Uses
Header	User navigation	Used on every page, except for the pages where the user is not logged in
Footer	Displays contact information	Footer is used on every page
Spinner	Displays a spinner when a page is loading information	Used on every page that needs to use the API to load content
Collapsible	A collapsible container that takes in a label and children. The children will be hidden when the collapsible is clicked	Every page that holds information has a collapsible information container. An exception is the contact page, due to the other containers on the page and design balance
DropdownModels	Uses the Collapsible component. Takes in a category, an array of models and a type of either ‘dashboard’ or ‘catalogue’. Loops through the array of models and renders each one. Depending on type, each model has a set of buttons relating to renting or using the model	Used in both the dashboard (“Dine Beregningsmodeller”) and the catalogue (“Lei Beregningsmodeller”) pages
AdminReqs	Fetches all new user requests from the database and displays them. Tempero admins can accept or	The first half of the administration page for Tempero admins

	deny the requests in this component	
AdminFirms	Fetches all firms from the database and displays them	The second half of the administration page for Tempero admins
FirmAdminNav	Navigation specific for Firm admins	For when the role of the user is Firm admin
TemperoNav	Navigation specific for Tempero admins	For when the role of the user is Tempero admin
UnderUserNav	Navigation specific for Under users	For when the role of the user is Under user
SuccessDisplay	Displays a link to a Stripe hosted site for managing the user's subscriptions, a link to the dashboard, and a link to the then purchased calculation model	For after a user has successfully subscribed to a model
ProductDisplay	Displays the prices (week, month) and a short description of a model, along with two buttons for selecting a plan of either a weekly or a monthly subscription. The buttons take the user to a checkout page hosted by Stripe	Displays when a user selects a model to rent

Table 5 – The applications components and their functions and uses

The pages within the App component, renders based on React Router DOM routes. React router is a fully featured client server-side routing library for React. It allows the developer to display pages that the users can navigate (React Router, 2022). The components mentioned above are used on different pages, as seen in Table 6 below. Together these make up the User Interface of the application.

Page	Functionality	Components
Login	Simple login form with links to the request new user page	
Request	Form for requesting access to the system, for new firms	
Register	Form for registering a new user. Only accessible when a Tempero admin accepts a new firm's request	

	for access to the system	
Model (Single model)	Fetches calculation model, displays calculation model, and handles calculations done by the user, by sending the input values to the backend for calculation, and displays the results	*Collapsible
ModelInfo	Fetches model information and displays it. Contains a link in the form of a button, which leads to the page for renting models	
AdministrateT	Administration page for Tempero admins. Administrates new user requests. Overview over all firms and the users listed under the firms	*AdminFirms *AdminReqs
AdministrateF	Administration page for Firm admins. Administrates Under users in the firm, including adding and removing users	
Catalogue (For renting models)	Page displaying models not owned by the user. Each model has a link for more information, and another for renting the model	*DropdownModels
ContactForm	Contact page. Contains contact information and a form for sending Tempero messages through the system	
Dashboard (Owned models)	Page for displaying models owned by the user. Each model has a link in the form of a button to use the model	*DropdownModels
Profile	User profile page, where the user can edit their information, including name, phone number, job title and their password	
RentModel	Page handling model renting. First renders the ProductDisplay component, then on a successful checkout, renders the SuccessDisplay component	*SuccessDisplay *ProductDisplay
Messages	Fetches and displays all messages sent by the users of the system. Only Tempero admins has access. Messages can be deleted from here. Each message holds contact information from the user, along with the message	

All pages contain a Footer component and most contain a Header component. Pages that interact with the API, use a Spinner component.

Table 6 – The applications pages with their components and functions

6.3.1 Folder structure

The folder structure is split into front-end and back-end. It is done like this because it makes it easier to debug, and it gives the choice to host either the front-end and back-end separately or both at the same time. Figure 21 shows the structure of the front-end folder.

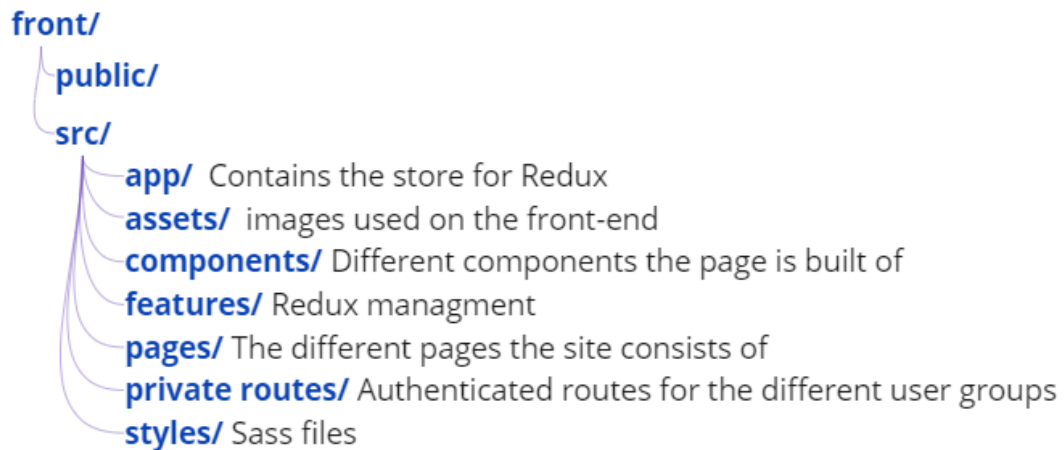


Figure 21 - front-end folder structure

6.3.2 Sass

To make the application look visually pleasing, the group used Sass (Syntactically Awesome Stylesheets). It is a “CSS pre-processor that lets you use variables, mathematical operations, mixing, loops, functions, imports, and other interesting functionalities that make writing CSS much more powerful” (Mauri, 2018).

The main reason for using Sass for this project is that its syntax and nesting ability makes it easy to follow the HTML hierarchy. The syntax (see Figure 22) is also shorter than with normal CSS because it doesn’t use brackets and semicolons. Sass is instead white space sensitive (Giraudel, 2022) .

```

1 $tempero: #0E2F86
2 $shadowColor: rgba(30, 30, 30, 0.4)
3 $deleteRed: #7A1212
4
5
6
7
8 @import "variables"
9
10 *
11 -webkit-box-sizing: border-box
12 -moz-box-sizing: border-box
13 box-sizing: border-box
14
15 body
16   margin: 0
17   font-family: Arial, Helvetica, sans-serif
18   background-color: white
19   font-size: 16px // Rem
20
21   main
22     width: 100%
23     min-height: 100vh

```

Figure 22 - Sass syntax

The style folder consists of sass files, the main files are organized so pages with similar content share a file. In addition to this, one file has all the media queries needed to make responsive design and another file has the most used variables in the application.

6.3.3 Redux

One important function in our application is authentication. The three types of users see different things on the website (see chapter 5.5.1). The information about the different users has to be shared with many different components that may or may not directly interact. Redux creates a single data store that can be accessed from anywhere in the application, as shown in Figure 23 (Castro, 2022). Redux makes it easier for the developer to pass the state to the components that requires it.

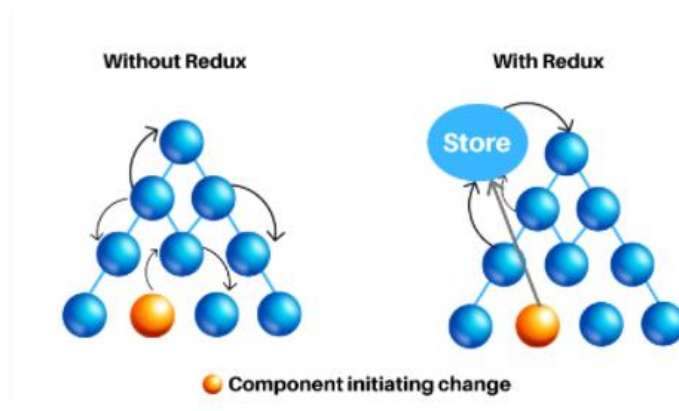


Figure 23 - How Redux stores the state (Totla, n.d.).

6.3.4 Axios

Axios was used in the application, to get the front-end to communicate with the back-end. Axios makes it easy to send CRUD (Create, Read, Update, and Delete) requests to the back-end API. It is promise-based, which lets the developer use JavaScript async and await to make a readable and asynchronous code. Figure 24 shows an example of how Axios is used in this project to *get* all the available models for a user under a firm, from the backend.

```
const getModels = async (token) => {  
  const config = {  
    headers: { Authorization: `Bearer ${token}` }  
  };  
  
  try {  
    await axios  
      .get('/api/calculation', config)  
      .then((res) => {  
        setModels(res.data)  
        toast.info('Alle beregningsmodeller hentet')  
      })  
  } catch (error) {  
    toast.error(error.response.statusText)  
    console.log(error.response.statusText)  
  }  
}
```

Figure 24 – Screenshot showing a Axios get call

6.3.5 Toast

An important factor that makes the user experience better is to always inform the user of what is going on. To do this the group used the notification library Toast. A toast is a customizable non-blocking notification pop-up that shows the user a readable message and then disappears after a few seconds. In this project, it was used to alert the user about information being sent, updated, or errors when they occur, as shown in Figure 25.

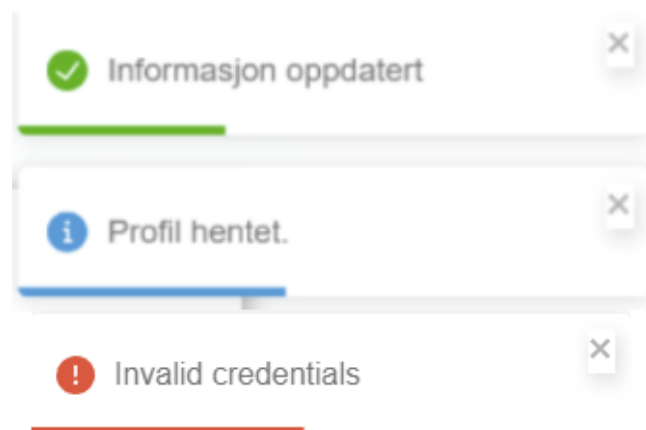


Figure 25 - Toast that shows information to the user

6.4 Back-end

The group identified a variety of technologies needed to get the application to work as intended. Authentication and authorization, APIs, a way to send emails with the links for registering pages, and a database to store the data, are the main parts of the server-side of the system.

6.4.1 Authentication and Authorization

The three types of users in the system have different access levels that are protected. Authorization happens for various routes, to determine whether the user is supposed to see data specific for Tempero users, Firm admins, or Under users.

Whenever a user's password is involved, from creation to editing to usage, the npm package "bcryptjs" is used. The package allows for creating hashed passwords and comparing that hashed password to an input to check for matches. Hashing makes it so no one can see the user's actual password, and only the hashed passwords get stored in the database. The "hash" is impossible for a human to read and make sense of (Arias, 2019).

Authentication (the process of verifying who someone is) happens for most routes, in the form of a middleware checking if the user has a bearer token. The npm package "jsonwebtoken" generates a token for the user at login, which is set to expire three days after creation, whereupon the user will be logged out automatically. This token is attached to the user and stored in local storage. When a route is called from the front-end, the token is sent as a header and verified in the middleware. The token is set to last for three days so the users don't have to enter their login details every time they exit and re-enter the application, which again makes for a better user experience.

6.4.2 API

When the front-end loads up certain pages that require dynamic and changing content, calls are made to the API. Here, data is fetched from a database and sent back to the front-end to render. Table 7 shows the five main API routes in the system.

Route	Functionality
Calculations	Handles CRUD interactions with the database, takes in values, and runs calculations
Firm	Handles CRUD interactions with database

Messages	Handles CRUD interactions with database
User	Handles CRUD interactions with database
Stripe	Payment handling. Handles interactions with Stripe API

Table 7 - main APIs in the system and their functionalities

6.4.3 Email system

To make it possible for the customers to register Firm admins and Under users, they need to be sent the links that take them to the registering pages. These URLs need a parameter which holds a request id, allowing the front-end to make a call to the back-end, which uses the request id to validate the user.

For this to work, the back-end of the system needs to be able to *send* emails. This was done using Nodemailer. Nodemailer allows the developers to create their own email templates using HTML, where style is added through style tags within the HTML. In this application, four different email templates are needed:

- An email is sent to Tempero, informing them when a new firm is requesting access to the system.
- Two different email templates are used with registration, whereas one handles a firm being declined and the other when they are accepted. A link to the registration page is only sent to those accepted.
- A last email is sent to a person when a Firm admin adds them under their firm as an Under user, as shown in Figure 26.

TEMPERO *Energitjenester AS* Beregningsmodeller

Hei

Ditt firma ønsker å legge deg til som underbruker i Tempero sitt systemet for Beregningsmodeller.

Klikk knappen under for å registrere din bruker.

Registrer bruker

Kontakt Tempero:

Telefon 951 84 774

Besøksadresse Havnegata 9, 7010 Trondheim

Figure 26 - Screenshot of email template made with Nodemailer

The outlook account “Temperoberegningsmodeller@outlook.com”, specifically made for the application, sends out these emails from the system to the email address provided by the back-end.

6.4.4 Database

The database chosen for this project was MongoDB since it’s part of the MERN stack. It is a database based on a non-relational document model, meaning that the data gets “stored in documents similar to JSON objects. Each document contains pairs of fields and values. The values can typically be a variety of types including things like strings, numbers, booleans, arrays, or objects” (MongoDB, 2022b).

The MongoDB database is built using Mongoose schemas. The schemas in our applications are as follows:

- User
- Firm
- Available models
- Calculation model
- Calculation model information
- Request
- Message

The user schema consists of a user's name, job title, their role in the system, email, phone number, password, firm id, and customer id if provided. All but customer id are required, and both email and phone number are set as unique. The roles are enumerated, being "Tempero" "Firm admin" or "Under user", as per the roles in the system determining access levels. The firm id binds the user to their firm.

The firm schema consists of information about a firm, including the firm's name and organization number.

The available model's schema consists of a firm id and a model id, linking a model to a firm.

The calculation model schema consists of a model name, category, two price ids created by Stripe (week plan, month plan), a short description of the model, and an array of inputs, which is set in a subschema. The subschema consists of the label for the input, a shortened label, min- and max values, and an output name for the calculation function, working as a reference. All are required, and both the calculation model name and product id are unique.

The calculation model information schema consists of a model id to bind it to its relevant model, and a full description holding an array of paragraphs.

The request schema consists of the requesting user's firm name, organization number, the user's phone number, email, their role in the system, and a Boolean value specifying whether the request is for a new firm or a new user under an existing firm.

The last schema, messages, consists of the sender's name, email and phone number, their firm name and id, and the message itself. The message schema can be seen in Figure 27.

```

1  const mongoose = require('mongoose')
2
3  const messageSchema = mongoose.Schema({
4    firmid: {
5      type: mongoose.Schema.Types.ObjectId,
6      ref: 'Firm',
7      required: true
8    },
9    firmname: {
10     type: String,
11     required: true
12   },
13   name: {
14     type: String,
15     required: [true, 'User needs a name'],
16   },
17   email: {
18     type: String,
19     required: [true, 'User needs an email'],
20   },
21   tlf: {
22     type: Number,
23     required: [true, 'User needs a phone number'],
24   },
25   msg: {
26     type: String,
27     required: true
28   }
29 }, { timestamps: true })
30
31 module.exports = mongoose.model('Message', messageSchema)

```

Figure 27- Message schema

6.5 Stripe

Stripe is a payment service provider that accepts credit cards and other payment methods. It supports many different currencies, including NOK, which was used for Tempero’s products. Stripe has a subscription function that allows for different payment plans, and lets the customer “make recurring payments for access to a product” (Stripe, 2022a).

The application has two payment plans. Customers can choose between paying for a week or a month at a time. This is due to the differing projects they work on – Some projects only require one or two uses of a model, meaning paying for a full month would be a waste of money.

To implement Stripe into the project, both front- and back-end work was necessary. The front-end handles the users’ interactions with the model subscription system, sending and receiving calls and data necessary to process payments. The back-end handles i.e., a set of webhooks and interactions with Stripe’s API.

When a Firm admin decides on renting a calculation model, they have to choose a plan of either weekly or monthly payment from the application’s front-end. Before the user is sent to the Stripe hosted checkout page (see Figure 28), the application’s back-end either generates a customer object and ties the customer id within it to the user or finds the user’s customer object if they already have a customer id tied to them. The customer id is sent with the checkout call to Stripe’s API, so that Stripe can hold onto their information and automatically fill out their details at checkout. This way, the user only has to fill out their card details once, which enhances the user experience.

←

Tempero | Beregningsmodeller


TEST MODE

Abonner på Energiforbruk ventilasjon og SFP

kr 1 600,00

hver måned

Denne modellen viser hvor mye energiforbruket (kWh forbruket) øker eller minker om en endrer på ventilasjonsanleggets SFP faktor.



Powered by stripe

Vilkår

Personvern

Betal med kort

E-post

Kortinformasjon

Fortsett å bruke •••• 0007

4000 0057 8000 0007

VISA

11 / 24

123

Navn på kortholder

test

Land eller region

Norge

▼

Abonner

🔒

Ved å bekrefte abonnementet ditt tillater du Tempero | Beregningsmodeller å belaste kortet ditt for denne betalingen og fremtidige betalinger i samsvar med vilkårene deres.

Figure 28 - Screenshot of Stripe's checkout (Card details for testing).

Once a user has completed the checkout and successfully paid, Stripe sends a message to the application's back-end through a webhook. A webhook is an automated message sent from an application when something happens (Guay, 2020), and the application runs different functions depending on which event type Stripe's webhook holds in its payload. For example, the event type stating a subscription has been created and paid for, will take in the model id tied to the metadata of the specific product (listed in Stripe), and use the email of the customer to find the user and their firm id, both in order to bind the model to the user's firm.

After this is completed, the user is sent to a success page with three links (see Figure 29). One link takes the user to the calculation page for the model, another to the dashboard and the last to a Stripe hosted subscription management portal, which works similarly to checkout.

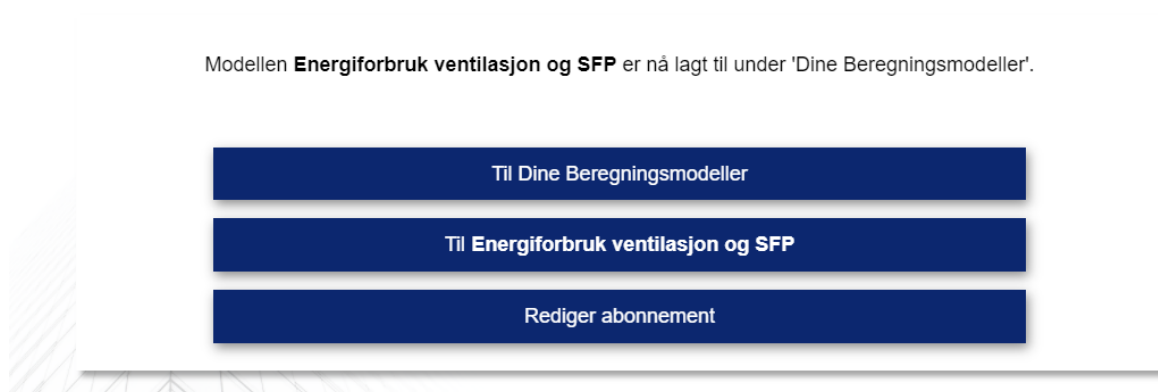


Figure 29 - Screenshot of the application's "successful payment" page

7 The application “Tempero Beregningsmodeller”

The goal of this project was to digitalize Tempero Energitjenester AS’ current solution by making an application where their customers can rent their calculation models online, without needing to talk directly to them.

All the functions in the MVP table (see chapter 4.3) were implemented in the application. In addition to this, two of the nice-to-haves; profile and change password, also got implemented. The final application is named “Tempero Beregningsmodeller”, and it can be reached through this link: <https://tempero-beregningsmodeller.herokuapp.com/>.

Tempero Beregningsmodeller lets a person from each of Tempero’s customer's firms register an admin user, which again can add Under users from their firm and rent calculation models from three different categories. The rented models can then be used by anyone in that firm to do calculations with. If they for any reason need help, the application has a contact form and contact information so the user can get in touch with Tempero.

There are three different user types available, Firm admin, Under user, and Tempero admin. Figure 30 shows what pages each type of user can access. The different color Post-its show who has access to what.

- White - Unregistered users
- Yellow - All registered users
- Green - All registered customers (Firm admin and Under users)
- Blue - Firm admins
- Purple - Tempero admins
- Red - Links (sent over email)

Everyone has access	All users with a link has access
Alle registered users have access	All customers have access
All Firm admins have access	Alle Tempero admins have access

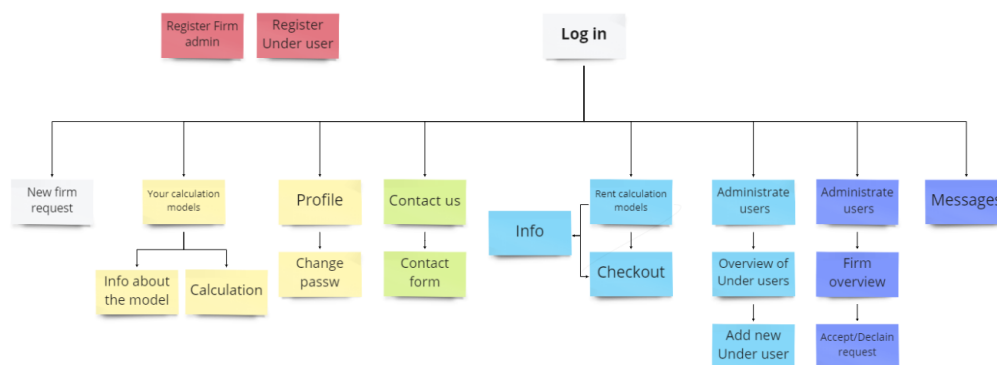


Figure 30 - Overview of the different pages Tempero Beregningsmodeller has and who has access

After the user logs in, the first thing they see is “Dine Beregningsmodeller” (see Figure 31). This page has an overview of the calculation models the firm has rented. For the Tempero admin, the “Beregningsmodeller” page shows all the models Tempero owns, and they are always available to use.

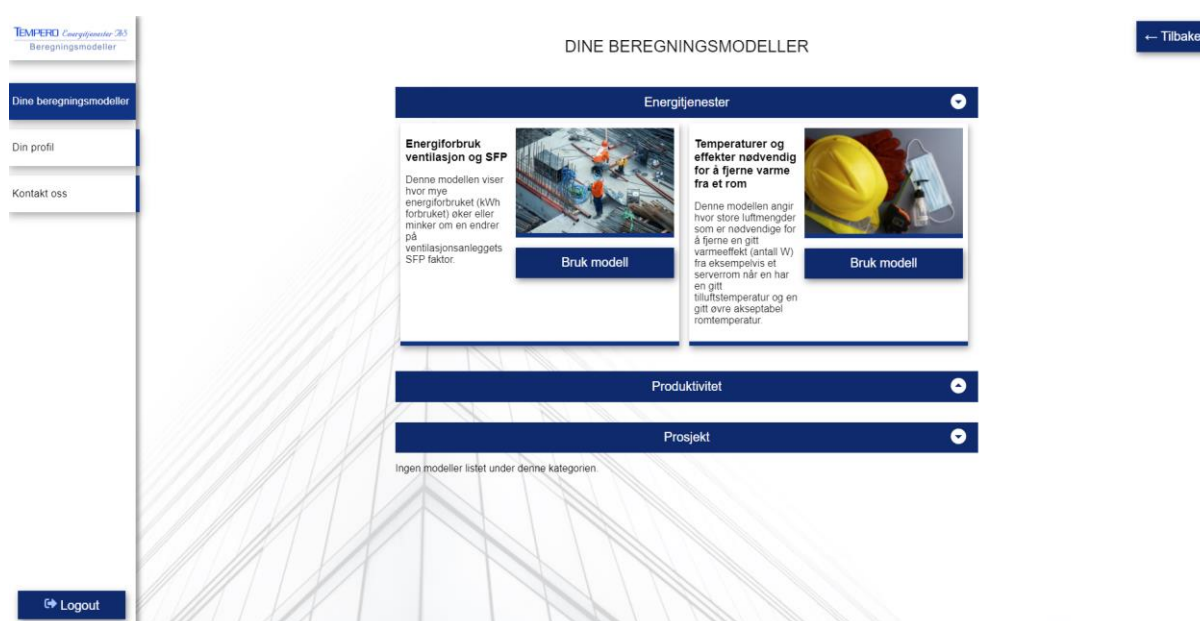


Figure 31 - Screenshot of overview of rented models

Each model has a button that takes the user to a calculation page for that model. The models are sorted into three different categories: Energy services, Productivity, and Projects. This makes it easier for the users to find what they need.

The calculation page for each model is the main functionality in Tempero Beregningsmodeller. It consists of a collapsible element, with basic information about how to use the model, input fields where the user inputs the data they have, and a result field where they get the answers, which appears when they click the “Utfør beregning” button (see Figure 32 below).

The figure shows two side-by-side screenshots of the 'DINE BEREGNINGSMODELLER' (Your Calculation Models) page in the Tempero application. Both screenshots have the title 'Temperaturer og effekter nødvendig for å fjerne varme fra et rom' (Temperatures and effects necessary to remove heat from a room).

Left Screenshot (Model with information):

- Modell** (Model):
 - Watt (mengde energi) tilført rommet: [Empty input field] Watt
 - m3/h tiluft: [Empty input field] m3/h
 - Temp. tiluft til rom: [Empty input field] °C
 - Temp. avtrekk i rom: [Empty input field] °C
 - [Utfør beregning] (Perform calculation) button
- Informasjon** (Information) - Collapsible section:
 - Denne modellen angir hvor store luftmengder som er nødvendige for å fjerne en gitt varmeeffekt (antall W) fra eksempelvis et serverrom når en har en gitt tiluftstemperatur og en gitt øvre akseptabel romtemperatur. En fyller inn antall W, tiluftstemperatur fra ventilasjon og den øvre grense for akseptabel romtemperatur.
 - Beregningen forutsetter at tiluft og avtrekk er plassert riktig i rommet.
 - Ved å snu på modellen og legge inn tilgjengelig luftmengde, temperatur tiluft til rom og temperatur avtrekk til rom vil en få beregnet nødvendig antall W (effekt) for å klare de ønskede temperaturer.

Right Screenshot (Shows a calculation being done with the result):

- Modell** (Model):
 - Watt (mengde energi) tilført rommet: [100] Watt
 - m3/h tiluft: [220] m3/h
 - Temp. tiluft til rom: [15] °C
 - Temp. avtrekk i rom: [28] °C
 - [Utfør beregning] (Perform calculation) button
- Resultat** (Result):
 - Nødvendig luftmengde for å fjerne en gitt varmeeffekt: [22 m3/h]
 - Hvor stor effekt kan en gitt luftmengde fjerne ved gitte tilufts og avtrekkstemperaturer: [1001 W]

Figure 32 - Screenshot of the models from the application. Left: Model with information. Right: Shows a calculation being done with the result

The “Lei Beregningsmodeller” page looks a lot like “Dine Beregningsmodeller” (see Figure 33). The difference between them is that this page shows an overview of the models that the user can rent. These models have two links, one takes the user to a page with information about the model, and the other to the checkout where they can rent the model.

The Firm admin is the only one that can rent models, but Under users can use all the models the admin has rented. This way, in case more than one person from the same firm is working on a project that needs the same calculation models, the users don't have to rent more than one.

The checkout page consists of a picture, the name of the model, and the price of renting it. It works like a subscription, the user pays for a week or a month at a time, and when the time is up, their card gets charged for another week/month if they don't cancel it. The user will get an

email telling them their card will get charged a few days before it happens. If they cancel the subscription, they lose the access to the calculation model. When the purchase is done the model is available from the “Dine Beregningsmodeller” page.



Figure 33 - Screenshot of "Lei Beregningsmodeller" page from the application

Both Firm admins and Tempero admins have access to a page named “Administrer brukere”, but these pages look different depending on what role the user has. The Firm admin's page (see Figure 34) lets the admin add Under users from their firm into the system. The Firm admin writes the email address of the person they want to add, and a link to a registration page for Under users gets sent to that email. When the person has registered their account, the Firm admin will see that on the same page. It is also possible for the Firm admin to delete Under users from the system.

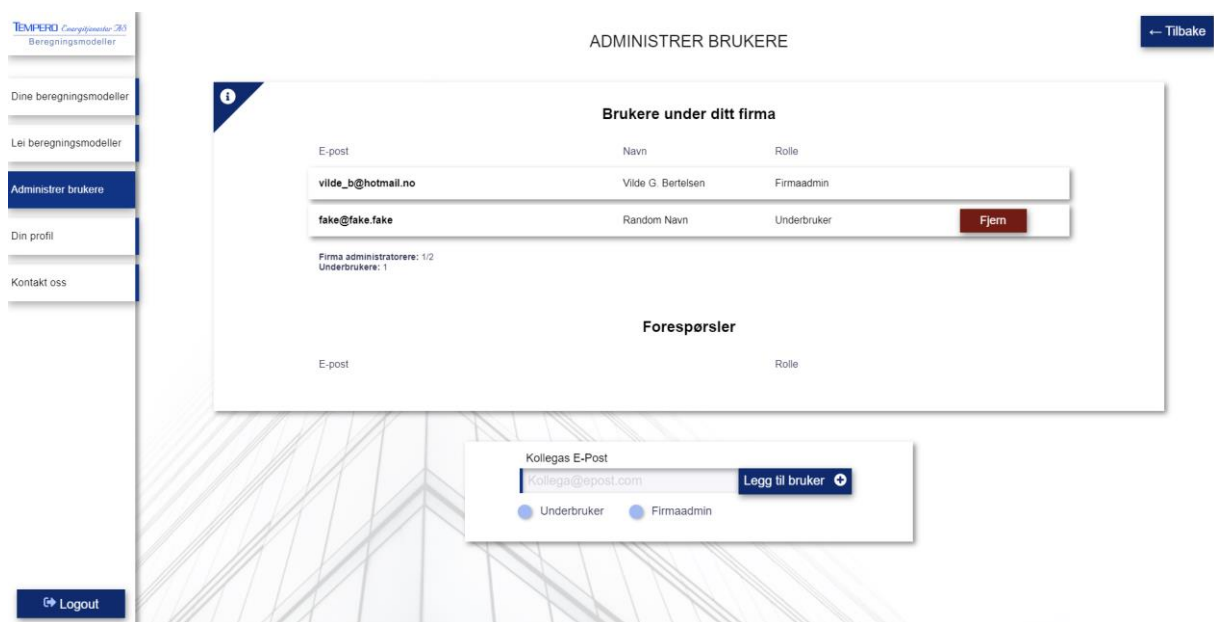


Figure 34 - Screenshot of the Firm admin page from the application

The Tempero admin page shows an overview of all the firms in the system and all the users in each firm. The page also has an overview of requests from firms asking to join the system (see Figure 35). It shows the name, organization number, email, and phone number of the firm. It is possible to click the organization number to see if the firm is legitimate. It takes the user to *Brønnøysundregistrene* which holds information about all firms in Norway. From this information, the Tempero admin can choose to accept or decline the firm. If they accept, the person that sent the request will get an email with a link that takes them to a Firm admin registration page. When the Firm admin gets registered the firm will show up for the Tempero admin in the firm overview.

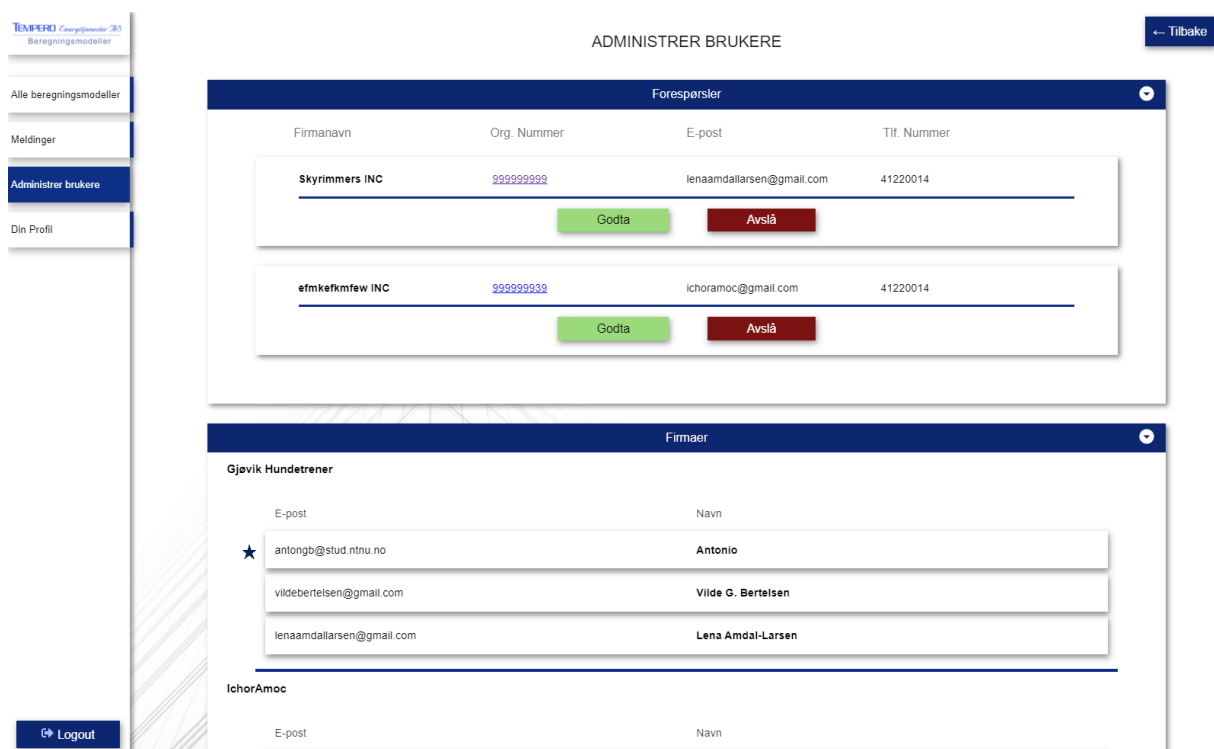


Figure 35 - Screenshot of the Tempero admin pages from the application.

The “Profil” page has the same information for every type of user. It shows the logged-in user's name, phone number, and job title, which they can edit (see Figure 36). In addition to this they can also edit their password. Firm admins and Under users also have the possibility to delete their own users.

Trying to change a password or delete a user will trigger an alert that asks the user if they want to proceed. As a precaution, a Firm admin can only delete their user if there is more than one Firm admin for the firm in the system.

Figure 36 - Screenshot of the profile page from the application

Firm admins and Under users have access to the “Kontakt oss” page where they can find Tempero’s contact information and a contact form. Through the contact form, the users can tell Tempero what they need help with and how they want to get guidance (email, phone, or in-person) and Tempero will respond, in the customers preferred way, as soon as possible.

Figure 37 - Screenshot of the contact form from the application

Tempero admins have a “Melding” page where all the guidance requests from the users in the system are sent to. Tempero admins will also be sent an email notifying them that they have gotten a new message.

A message holds information about the name of the sender, what firm they are from, their phone number, email address, and the message text. This way the Tempero admin can easily contact them.

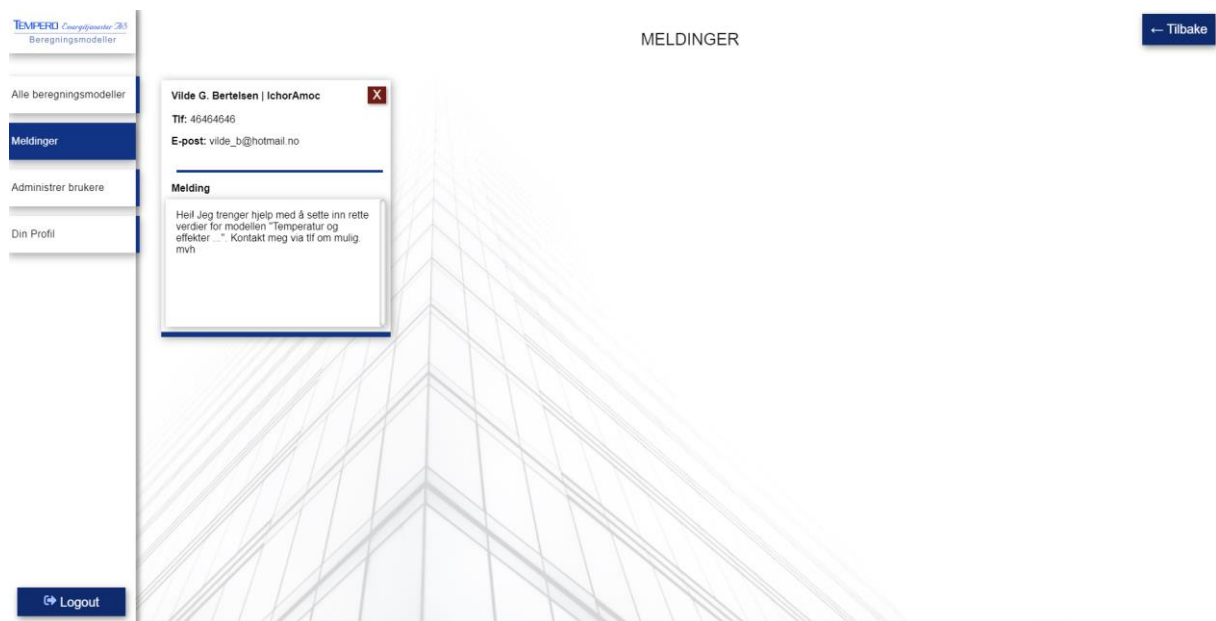


Figure 38 - Screenshot of the message inbox from the application

7.1 User-guide

This guide will demonstrate how the reader can use the application with all three user types. It will showcase the main functionalities of Tempero Beregningsmodeller, which can be reached from here: <https://tempero-beregningsmodeller.herokuapp.com/>.

The application is set to test mode, meaning both the payment method and actual calculation models are fake. Stripe does not charge the user, and the calculation models return only the inputted values added together, due to these models being actual products, Tempero don't want anyone to use for free.

To make the process simpler, three users are already registered in the system, they are available in Table 8 below.

Role	Email	Password
Tempero admin	ellinorr@fake.no	sNH9Xp2s7QJJ98zKMfEG
Firm admin	ovep@fake.no	2yVJBxBVQrfaVuaNzYFq
Under user	davidl@fake.no	bENhR9k7Mk8T2LaUKrAX

Table 8 - Login information

Some functionalities depend on other user types doing something, so following the guide is crucial to try every functionality. The guide will be split into multiple parts where different user types are used.

Part 0 will demonstrate how to make a request to Tempero asking for your firm to get added to the system. *Make sure to fill in an email you have access to, so you can receive the registration link. You might receive the email in your spam folder.*

Part 0 - Request to add a new firm

Step 1	Click the link: “Ny bruker? send forespørsel”
Step 2	Fill in your firm's information Org.nr: Any 9 digits Phone nr: Any 8 digits

Table 9 - User guide part 0

Part 0.1 - Tempero admin

Step 1	Log in as Tempero admin
Step 2	Go to “Administrer brukere” and accept the new firm request
Step 3	Log out

Table 10 - User guide part 0.1

A new firm has now been accepted to the system, and you will get an email with a link that takes you to the Firm admin registration page. As a Firm admin, you can now add Under users to the firm. For the full user guide, see appendix D.

8 Discussion

At the beginning of the project the following problem statement was asked:

How can we design and implement a secure and user-friendly application that makes calculation models from Tempero Energiteknister AS available for its clientele?

To better answer the problem statement, three sub questions were made. These questions were built on the previous questions asked in our report for the class *IDG3101*

Fordypningsprosjekt. In this part of the report, we will discuss our research and the results we have gotten from this project to answer the problem statement.

The main reason for making “Tempero Beregningsmodeller” was to digitalize the way Tempero helps its customers with the calculations needed for their building projects. The problem Tempero have with their current approach is that it takes them a lot of time to do something for their customers, that the customers could be able to do themselves if they had access.

The survey done by the end-users showed that they were very interested in getting a digitalized system, but for an application to be better than the current approach, it needs to be safe and user-friendly. Without a good user experience, people wouldn't want to use it.

8.1 A good user experience

To make a good user experience you have to focus on the users, how they complete their tasks and goals, and what their strengths and weaknesses are when it comes to technology (Miller, 2005). Not everyone thinks the same, so a good user experience is subjective depending on the person.

Concepts like “Journey driven design”, usability and accessibility principles, as well as user testing, and the group's survey of the end-users contribute to answering sub question 1: *How can we create a good user experience for the users using the application?*

8.1.1 Journey driven design

Today “mobile-first” is the most popular design method because of how many people have a mobile phone. In this project, however, we are making a system for Tempero's existing customers, which in the big picture is a small number of people. The best way to make a good

user experience for them would be to ask what they want. This is the first step in an alternative design method called “Journey driven design”.

The end-user survey showed that the most critical device to make the application for was PC and that the users wanted a simple design. This was the main focus when designing Tempero Beregningsmodeller. We prioritized finishing a desktop version first, then adjusted the design to fit for mobile afterwards.

8.1.2 Heuristic evaluation

We wanted to make the task of using a calculation model as simple as possible while still fulfilling Tempero's requirements. During the prototyping phase, a heuristic evaluation of our prototype was done by a UI expert. It followed Jakob Nielsen's ten principles for a good interface design (Nielsen, 2020), to see if Tempero Beregningsmodeller had a good user interface. The main heuristics focused on was visibility in system status, consistency and standards, error prevention, and aesthetic and minimal design.

Visibility in system status refers to always keeping the user informed of what is going on through feedback. We implemented Toasts (see chapter 6.3.5) to do this. It gives the user information about the result when an action is performed.

After using Tempero Beregningsmodeller, we found that a potential problem with the consistency of the design is that it might become *too* consistent. “Dine Beregningsmodeller” and “Lei Beregningsmodeller” looks pretty similar, in fact they are almost identical. The only way to differentiate them from one another is to read the main heading and the highlighted part of the navigation bar, which shows which page they are on. If the users just skim the page, they might not notice if they are on the wrong page or not. This wouldn't create any real problems except maybe losing the users some time, because after a few click they would either see a calculation model or a payment page, but it is something to take into consideration if the application gets developed further.

Error prevention is very important for this project because of how complicated some calculation models can be. If the user gets the wrong answer from the calculation it can lead to a number of bad events. In the chapter 6.2 we looked at different things that could help prevent human errors. Simple HTML functions like “number”, “required” and logical min/max values in each input field drastically decrease the user's possibility to input the

wrong data. For other parts of the application, simple alert pop-ups will show before a user can do drastic changes like deleting something or changing a password.

The heuristic: aesthetics and minimalist design, say that “Interfaces should not contain information which is irrelevant or rarely needed. Every extra unit of information in an interface competes with the relevant units of information and diminishes their relative visibility.” (Nielsen, 2020).

While the design started very minimalistic, the large number of information boxes needed made for a very packed site. We couldn't just remove the information text because it is important to prevent human errors, so we found that hiding the text behind a collapsible element was a good solution. It lets the user choose what they want to see while making the page look clean.

8.1.3 Accessibility

By following the Web Content Accessibility Guidelines (WCAG) (W3C, 2019b), *Tempero Beregningsmodeller* should be accessible to as many as possible. Accessibility makes for a great user experience. Because most of the end-users were over 60 years old, having easily readable text and good contrast on the site was something the group focused on.

As mentioned in chapter 5.5 all our font sizes are at minimum 12 pixels and generally above 16 pixels, making the text readable for the users.

The contrast of the pages follow guideline 1.4.3 Contrast (Minimum) that say the visual presentation of text and images of text should have a contrast ratio of at least 4.5:1. The web accessibility tool Wave (WebAIM, 2022) shows that the blue on white color *Tempero Beregningsmodeller* uses, passes both the AA and AAA requirements, with a contrast of 8.59:1.

We also made sure to follow guideline 1.1.1 Non-text Content and provide alternative text to pictures and input fields. *Tempero Beregningsmodeller* has pictures on the calculation model overview, and all of these have “alt tags” that describe the picture. All the input fields in the calculation models also have corresponding labels to show what they are. In addition, buttons with Sass-rendered text have aria-labels explaining what the button does. All this is done so that users who can't see these elements will have alternatives that lets them use the application.

8.1.4 Conclusion to sub question 1

To answer sub question 1, the group has focused on creating a good user experience through surveying the end-users, running user tests, and focusing on accessibility and usability through the entire design process. All this was used in the development of the application.

8.2 Guidance in the system

Sub question 2 asks: *What is the best way for the end-user to contact Tempero Energitjenester AS for guidance when it comes to using the calculation models?* The answers we got from the user survey showed that 100% were happy with the current guidance system, which included emailing Tempero or meeting with them face to face.

In our report from the previous semester (see appendix A) we concluded with keeping the current way of guidance, but also adding a contact form in the application. This way the users can chose if they want to contact Tempero the old way or through the new application.

We didn't want to complicate the process further by adding e.g., a chatbot. It would have been a bad solution because of the complexity of the questions that might be asked. Getting any complex question answered face to face is more efficient, because the conversation will be more detailed. By making the application the way we have, we hope that the need for help might not be as big as before, making both Tempero and the end-user's jobs easier.

8.2.1 Conclusion to sub question 2

To answer sub question 2, the best way to contact Tempero for guidance would be to keep doing what they currently do; let the user contact them through email or phone. In addition to this, the application has a contact form so the customers can send messages to Tempero through the system if they wish. Adding this provides them another option for asking minor questions, without having to leave the page.

8.3 Safe payment method

Our MVP vs nice-to-have table (see chapter 4.3) shows that we needed to implement a card payment method in the system, and that invoice would be nice-to-have. The current application only takes card payment, but if this application gets developed further, we will also add an invoice option, because not all firms necessarily have a company card available to

use. Sub question 3 asks: *What is a safe way to implement a card payment method in the application?*

Implementing a payment method into the system was a big and daunting task. People are going to use real money to pay for the calculation models, so the payment method must be safe. We could not do this on our own, so we started researching the four different payment systems that we knew the end-users were familiar with (see chapter 4.2). We wanted to only look into familiar payment methods because it is logical that people feel safer paying in a way they have done earlier on other websites.

Based on the system's needs and how well it was documented, our choice on payment system ended up being Stripe. Some of Stripe's users are large companies like Ford and Lyft (Stripe, 2022b), so we believed Stripe was a very secure service to use.

Stripe can be used without Tempero having an agreement with them and the customers do not need to register a user to pay through the service. All the payment methods we looked into had an additional fee that the user need to pay in order to use it. Stripe's fee is a fixed amount of 2 NOK per transaction. This seemed worth it for how secure the system is.

In the interview with Tempero, we were told that they wanted a subscription-based system in the application. Stripe has a subscription function that let the user pay in increments of days, weeks, or months, depending on what the developers choose. In this application we saw it best to let the customer choose if they wanted to rent the models in increments of weeks or months. Giving both these options will make the products more appealing. If they only need the model for a short amount of time they can chose a week, and then pay again the week after if they didn't finish the work in time. If they need the model for a long time, paying each week would become annoying, so then it's better to choose the monthly option.

The payment system is functional in the application, but Stripe is a very complex system for beginners, so there are better ways to implement it, such as adding more webhooks to keep better track of events, like a user changing their payment plan. This is something we will look into if the application gets developed further.

8.3.1 Conclusion to sub question 3

To answer sub question 3, a safe way to implement a card payment system is to use the third-party service Stripe. The reason for this is that the security surrounding the payment system

will be handled by a well-trusted software, instead of our application. As displayed in Figure 28, what the user sees is a standardized checkout page, which is familiar to them and can consequently make them feel safer when paying.

8.4 Sustainability

Since sustainability is more important than ever, NTNU wanted their students to reflect on how their projects could contribute to a “better world”. Sustainability is the concept of development being able to satisfy the needs of the current human population, while not damaging the world for the future generations to come. In other words, sustainability refers to developing something in a way where it can be sustained over a long period of time (Tjernshaugen, 2022).

A potential positive impact our project might have, is augmenting Tempero’s work, as their largest business area is finding solutions for energy supplies, efficient energy, and power usage in buildings, as well as indoor climate. By digitalizing their work, the consulting they do could reach out to more people and firms, thus increasing the amount of new or restored buildings that could become more sustainable in terms of resource use, like energy consumption. This applies to UN’s sustainability goal nr. 12: “Ensure sustainable consumption and production patterns” (UN, n.d.).

Looking at it from a development point of view, different strategies could be used to make the application itself more sustainable. Transmaterialization is one strategy for sustainable website design. It is the process of transforming a product into a service, making it less resource intensive (Frick, 2016). This is exactly what we have done with Tempero Beregningsmodeller, it takes Tempero’s product; the calculation models, and make them available through a subscription-based service.

Before this application was created, Tempero would receive a request for a calculation model through one email then return the answer through another. With this application, emails are required in order for the users to register, but when they are in the system, the uses of emails for getting the calculation models will decrease, and thus decreasing energy consumption.

Sending and receiving emails uses electricity, most likely produced through methods which releases CO₂ (Abdallah and El-Shennawy, 2013). Say Tempero sends and receives in total 10

emails per day, this would equate to 2500 emails a year which would create around 12kg of CO₂. This equates to driving 27km in an average nonelectric car (CwJobs, n.d.).

Another sustainability strategy is dematerialization, meaning simplifying websites to have just enough content to achieve their goal (Frick, 2016). Tempero Beregningsmodeller is a very simple website, but it has a few pictures it might not need. If we test the application further, we can see if the users actually need the pictures. If they don't, removing them would make a very small change, but would still improve the sustainable of the application.

As this is the first version of the application, our focus was to get it online and functioning, to test how it performs among Tempero's clientele. Due to this, it is hosted through a free hosting service, but in the future, choosing the right service would make the application more sustainable. One option is the company GreenGeeks (2022), who emphasize on leaving a positive energy footprint on the environment.

9 Conclusion

Efficiency is vital in any work environment and is something that can be enhanced by digitalizing certain tasks. This project was created in collaboration with Tempero, in order to augment their work in the form of digitalizing their services. This would allow for more people and businesses to utilize their services, as well as removing Tempero as the middleman when they are not specifically needed.

We chose to *design and implement a secure and user-friendly application that would make Tempero Energitjenester AS' calculation models digitally available for their clientele*. To do this, we created three sub questions. Chapter 8, which discusses the questions asked for this project, shows how, in the process of designing and developing the application, we focused on the end-users' needs, and the application's security, usability and accessibility.

We concluded that the best way for the users to ask for guidance would be by contacting Tempero through email, a phone call or the application's contact form. Through this, they can personally decide on the best way for them to receive guidance.

The security surrounding the application's payment system is handled by Stripe. This allows the customers to feel safer when putting their card details into an unfamiliar website in order to rent the calculation models.

These three sub questions together contribute to answering the problem statement. Our solution is the application, Tempero Beregningsmodeller. It is a user-friendly application which makes Tempero's calculation models available to rent and use online, for their customers. Tempero Beregningsmodeller's easy access to calculation models gives their customers the ability to do the calculation needed for their project on their own, making the process move faster and more efficient.

9.1 Further development

The current application is a good starting point, but we have some suggestions for further development. However, before any big changes are made, it would be smart to check if the application gets used by the customers.

Whether digitalization is a success or not can be measured through Key Performance Indicators (KPI) (Marr, 2012). These indicators can measure, among other things, the customer perspective: how often the application is used, and the change in productivity due to this. This is something that happens over a longer period of time, but Tempero should be able to use this method to see if the digitalization of the calculation models is a success or not.

Chapter 8.4 discussed some future changes for the application surrounding sustainability. Further development of the application would include adding the functions mentioned in the nice-to-have part of Table 2, renting models in packages, model history, invoices, enhanced Stripe functionalities, choice of language and a dark/light theme.

The current application only has three working calculation models, so a package deal won't make much sense at the moment. However, Tempero has a lot of calculation models, so if they decide they want more of them added to the system, it would be nice for the users to be able to rent similar models in packages, instead of one at a time.

Implementing the calculation models' "history function" would also be a nice touch. The final iteration of the hi-fi prototype has a history button, as mentioned in chapter 5.5, but the function was never development in the current application, as it was not a required functionality. This function would let anyone that owns a model see previous calculations saved to that specific model. This way the users can look up old answers instead of doing the same calculation over again. It would also be useful if errors arise, as the user can check what data was put in previously, to see if any input contributed to the faulty result.

Invoice is another thing that would be good for the application. As mentioned, all of Tempero's customers might not have a company card that can be used to rent models. An invoice system would make it possible to pay in another way, making the application accessible to more users. The way we'd go about adding invoice is through Stripe, as it provides many payment options for businesses, such as invoicing and receipts.

In addition, we would improve our application's webhooks for communication with Stripe, as it is currently in its simplest form. For example, there are many different types of events, such as "subscription has been updated", "subscription has been paused", etc., all of which should be covered by the application's back-end to some degree.

Some smaller changes that could improve the application are the choice to change languages and a dark-/light-mode switch. The project owner informed us that some of Tempero's customers are from other Scandinavian countries. Adding a second language like English, could make the application more usable for people whose main language is not Norwegian. Dark-/light-mode would also be good because different users might prefer one over the other, or their surroundings might call for different brightness levels to see properly on their screens. If Tempero's customers like the application and use it regularly, a final touch would be to integrate the system into Tempero's future website.

10 References

- Abdallah, L., El-Shennawy, T., 2013. Reducing Carbon Dioxide Emissions from Electricity Sector Using Smart Electric Grid Applications. *Journal of Engineering* 2013, e845051. <https://doi.org/10.1155/2013/845051>
- Amdal-Larsen, L., Bertelsen, V.G., Gjeitsund, I.M.R., 2021. *Webløsning for Tempero Energitjenester AS*.
- Arias, D., 2019. How to Hash Passwords: One-Way Road to Enhanced Security. *Auth0 Blog* Available at: <https://auth0.com/blog/hashing-passwords-one-way-road-to-security/> (accessed 4.27.22).
- Bloomberg, J., 2018. Digitization, digitalization, and digital transformation: confuse them at your peril. *Forbes*. August 28.
- Bossard, 2020. *The Pros and Cons of Digitalization*. Available at: <https://provenproductivity.com/the-pros-and-cons-of-digitalization/> (accessed 1.17.22).
- Castro, S., 2022. Why and When You Should Use Redux. *Jobsity*, April 29. Available at: <https://www.jobsity.com/blog/why-and-when-you-should-use-redux> (accessed 4.1.22).
- CwJobs, n.d. *The Email CO2 Calculator*. Available at: <https://www.cwjobs.co.uk/insights/environmental-impact-of-emails/> (accessed 5.13.22).
- Frick, T., 2016. *Designing for Sustainability: A Guide to Building Greener Digital Products and Services*. O'Reilly Media, Inc.
- Forskrift om universell utforming av IKT-løsninger (2019) *Forskrift om universell utforming av informasjons- og kommunikasjonsteknologiske (IKT)-løsninger*. Available at: <https://lovdata.no/dokument/SF/forskrift/2013-06-21-732> (accessed 2.16.22).
- Giraudel, K., 2022. *Sass Guidelines*. Available at: <https://sass-guidelin.es/> (accessed 3.29.22).
- GreenGeeks, 2022. *Fast, Secure and Eco-friendly Hosting*. Available at: <https://www.greengeeks.com/> (accessed 5.13.22).
- Guay, M., 2020. What are webhooks? *Zapier*, September 12. Available at: <https://zapier.com/blog/what-are-webhooks/> (accessed 5.10.22).
- Initiative (WAI), 2022. *WCAG 2 Overview*. Available at: <https://www.w3.org/WAI/standards-guidelines/wcag/> (accessed 4.29.22).

- Interaction Design Foundation, 2022. *What is Heuristic Evaluation?* Available at: <https://www.interaction-design.org/literature/topics/heuristic-evaluation> (accessed 2.16.22).
- Kagga, J., Atto, E., 2020. *Understanding React Components*. Available at: <https://medium.com/the-andela-way/understanding-react-components-37f841c1f3bb> (accessed 4.27.22).
- Kenzie Academy, 2020. Front End vs. Back End: What's the Difference? *Kenzie Academy*, August 17. Available at: <https://kenzie.snhu.edu/blog/front-end-vs-back-end-whats-the-difference/> (accessed 3.31.22).
- Krug, S., 2014. *Don't Make Me Think, Revisited. A common sense approach to web usability*. New Riders.
- Marr, B., 2012. *Key Performance Indicators (KPI): The 75 measures every manager needs to know*. Pearson UK.
- Mauri, C., 2018. 7 benefits of using SASS over conventional CSS. *Mugo Web*, March 14. Available at: <https://www.mugo.ca/Blog/7-benefits-of-using-SASS-over-conventional-CSS> (accessed 3.29.22).
- McCloskey, M., 2014. *Task Scenarios for Usability Testing*. Available at: <https://www.nngroup.com/articles/task-scenarios-usability-testing/> (accessed 2.2.22).
- Mesibov, M., Levin, J., 2017. *Mobile First Is Just Not Good Enough: Meet Journey-Driven Design*. Available at: <https://www.smashingmagazine.com/2017/02/mobile-first-is-just-not-good-enough-meet-journey-driven-design/> (accessed 11.25.21).
- MongoDB, 2022b. *What Is NoSQL? NoSQL Databases Explained*. Available at: <https://www.mongodb.com/nosql-explained> (accessed 5.4.22).
- MongoDB, 2021a. *What Is The MERN Stack? Introduction & Examples*. Available at: <https://www.mongodb.com/mern-stack> (accessed 3.30.22).
- Morales, J., 2021. Mobile First Design Strategy: The When + Why. *Adobe XD Ideas*, February 16. Available at: <https://xd.adobe.com/ideas/process/ui-design/what-is-mobile-first-design/> (accessed 11.25.21).
- Nielsen, J., 2020b. *10 Usability Heuristics for User Interface Design*. Available at: <https://www.nngroup.com/articles/ten-usability-heuristics/> (accessed 10.4.21).
- Nielsen, J., 2000a. *Why You Only Need to Test with 5 Users*. Available at: <https://www.nngroup.com/articles/why-you-only-need-to-test-with-5-users/> (accessed 2.2.22).

React Router, 2022. *React Router / Tutorial*. Available at:
<https://reactrouter.com/docs/en/v6/getting-started/tutorial> (accessed 4.27.22).

Siteimprove, 2022. *Accessible fonts: How to choose a font for web accessibility*. Available at:
<https://siteimprove.com/en-us/accessibility/most-accessible-fonts/> (accessed 2.16.22).

Stack Overflow, 2021. *Stack Overflow Developer Survey 2021*. Available at:
https://insights.stackoverflow.com/survey/2021/?utm_source=social-share&utm_medium=social&utm_campaign=dev-survey-2021 (accessed 3.29.22).

Stripe, 2022a. *How subscriptions work*. Available at:
<https://stripe.com/docs/billing/subscriptions/overview> (accessed 4.5.22).

Stripe, 2022b. *Stripe Customers*. Available at: <https://stripe.com/en-no/customers/all>
 (accessed 5.3.22).

Tjernshaugen, A., 2022. bærekraft. *Store norske leksikon*.

Totla, J., n.d. The Advantages of Using Redux along with React. *Synergy* Available at:
<https://synergytop.com/blog/the-advantages-of-using-redux-along-with-react/>
 (accessed 5.12.22).

UN, n.d. *Sustainable consumption and production. United Nations Sustainable Development*.
 Available at: <https://www.un.org/sustainabledevelopment/sustainable-consumption-production/> (accessed 5.10.22).

W3C, 2019. *How to Meet WCAG (Quickref Reference)*. Available at:
<https://www.w3.org/WAI/WCAG21/quickref/?showtechniques=111#contrast-enhanced> (accessed 4.28.22).

W3C, 2011. *Retningslinjer for tilgjengelig webinnhold (WCAG) 2.0* Available at:
<https://www.w3.org/Translations/WCAG20-no/> (accessed 12.2.21).

w3schools, n.d. *JavaScript Numbers*. Available at:
https://www.w3schools.com/js/js_numbers.asp (accessed 5.3.22).

WebAIM, 2022. *WebAIM: Web Accessibility In Mind*. Available at: <https://webaim.org/>
 (accessed 4.28.22).

11 Figures

Figure 1 - Example of one calculation model in Excel made by Tempero	2
Figure 2 - Our Discord server for this project	5
Figure 3- Screenshot of the Gantt chart for our project	6
Figure 4 - Sitemap that shows what the different users have access.....	10
Figure 5 - From the left: Add a new firm request form, register Firm admin form, an overview of all the calculation models, using a specific model from the lo-fi prototype in Fordypningsprosjekt.....	15
Figure 6 - Changes made from iteration one (left) to two (right).....	16
Figure 7 - Tempero's logo, from 2012	17
Figure 8 - The navigation bars of the different user types, from iteration one	18
Figure 9 – Models page split into two; one for owned models, one for rentable models	19
Figure 10 – Changes done to the model overview page	20
Figure 11 - Added phone number, email, and job title input fields	20
Figure 12 - Changed the history button from grey to blue	21
Figure 13 - Changed trashcan to a red box with a white X	21
Figure 14 – Changes in the logo design	22
Figure 15 - Changes on the history button	23
Figure 16 – Information box changes. The right picture shows an info icon the information is hidden behind. The bottom picture shows a collapsible element with information.....	24
Figure 17 – MERN 3-tier architecture pattern (MongoDB, 2021a).....	26
Figure 18 - Min and max inputs for each input calculation model made by Tempero	27
Figure 19 - Example of an easy calculation done in JS that gives the wrong answer.....	27
Figure 20 - Example of an easy calculation done in JS, with functions giving the right answer	27
Figure 21 - front-end folder structure.....	31
Figure 22 - Sass syntax.....	32
Figure 23 - How Redux stores the state (Totla, n.d.).	32
Figure 24 – Screenshot showing a Axios get call	33
Figure 25 - Toast that shows information to the user.....	33
Figure 26 - Screenshot of email template made with Nodemailer	36
Figure 27- Message schema	38

Figure 28 - Screenshot of Stripe's checkout (Card details for testing).	39
Figure 29 - Screenshot of the application's "successful payment" page.....	40
Figure 30 - Overview of the different pages Tempero Beregningsmodeller has and who has access.....	42
Figure 31 - Screenshot of overview of rented models	42
Figure 32 - Screenshot of the models from the application. Left: Model with information. Right: Shows a calculation being done with the result	43
Figure 33 - Screenshot of "Lei Beregningsmodeller" page from the application	44
Figure 34 - Screenshot of the Firm admin page from the application.....	45
Figure 35 - Screenshot of the Tempero admin pages from the application.	46
Figure 36 - Screenshot of the profile page from the application.....	47
Figure 37 - Screenshot of the contact form from the application.....	47
Figure 38 - Screenshot of the message inbox from the application	48

12 Tables

Table 1 - Summary of different payment methods made by Ida M. R Gjeitsund (Amdal-Larsen et al., 2021).....	9
Table 2 - MVP vs Nice-to-have (Amdal-Larsen et al., 2021).....	10
Table 3 – System requirements (Amdal-Larsen et al., 2021).....	11
Table 4 – The five scenarios used in the user-tests	14
Table 5 – The applications components and their functions and uses	29
Table 6 – The applications pages with their components and functions.....	30
Table 7 - main APIs in the system and their functionalities	35
Table 8 - Login information	49
Table 9 - User guide part 0.....	49
Table 10 - User guide part 0.1	49

13 Appendix

Appendix A: The report from IDG3101 Fordypningsemnet

Appendix B: Overview of feedback from all user-tests

Appendix C: Results from the heuristic evaluation

Appendix D: User guide for the system

Appendix E: The source code of Tempero Beregningsmodeller