FNH-CRM
Management tool for fitness studios

Bachelor Thesis

Department of Computer Science
OST – University of Applied Science
Campus Rapperswil-Jona

Autumn Term 2020

Author(s):	Michael Güntensperger
	Raphael Rechsteiner
	Lorenzo Schumacher
Advisor:	Prof. Frank Koch
Project Partner:	FNH Personal Training GmbH
External Co-Examiner:	Prof. Hansjörg Huser
Internal Co-Examiner:	Laurent Metzger

Abstract

For fitness studios and studio chains, customer loyalty is getting more and more important, as the prices for subscriptions cannot get much lower. For a studio to achieve this, a CRM is necessary. The tools on the market are either fitness studio softwares that offer appointment booking but are missing a lot of a CRM functionality or CRM software that misses the needed fitness studio functionality. FNH wants to change this and build a fitness studio software with the CRM at its core.

When a personal trainer works with the current solution, he has to constantly switch between multiple devices, PDF forms and applications. This makes work tedious and costs a lot of time, which could be spent on the customer. In order to optimize these processes, a CRM should be created that combines all the required functions to manage a lead's or customer's information digitally in one tool and on one device. This simplifies the analysis with business intelligence tools significantly, because all the information is in one place.

A web-based CRM was developed for the client, which contains the basic functionality. The application makes it possible to follow a potential customer, called lead, through the entire customer acquisition process to the management of a customer. This includes recording various measurements and documentation of the customer, the training or the nutrition, as well as customer interaction reporting. Several functions were implemented to improve customer loyalty. The administrator can create individual processes, so-called workflows, that can be assigned to the leads depending on their requirements. All the lead process steps are recorded for future analysis.

The prototype contains two core components, frontend and backend. Both parts of the application provide all the desired basic functionality and can be easily extended due to the use of well-known web technologies. This includes React for the frontend and Node.js for the backend. For the database, MySQL was used. All application components are hosted as containers in the cloud. The modular design and use of the flexible REST API allows for new functionality to be added to the application in the future.

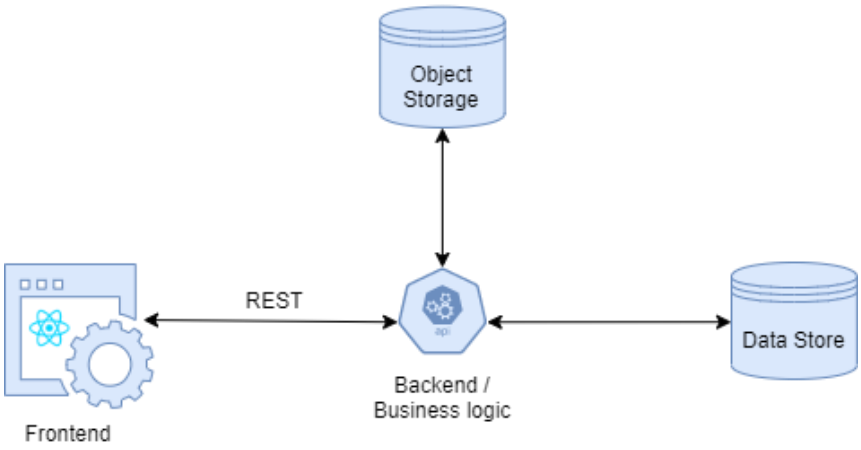

Figure 1 - Architecture diagram - Overview

Management summary

About the customer

FNH is a personal training studio and is short for Fitness Nutrition Health. They have three studio locations around the upper lake Zürich: Siebnen, Jona and Grünfeld, with Grünfeld being the most recent addition. There are ten personal trainers working for the studios. With more than 800 customers, a lot of organization is involved to keep all the customers happy.
The cheaper fitness chains are making their money by selling a huge number of cheap subscriptions and offering a very limited customer support. On the other hand, FNH sells more expensive subscriptions but accompanies and pushes the customers to a healthier lifestyle, which involves more personalized customer interactions. A study in 2018 showed that customers find the consulting in the studios to be very important. [1] As such, offering personalized and high-quality client services is one of their key business drivers. Digitalizing this aspect of their business has been an ongoing process for some time and has not yet yielded the expected results.
For these reasons, FNH has decided to build their own application even though it is not an IT-company. Their focus is on developing a simple, user-friendly application, that is limited to the core functionalities required.

Initial situation

In order to justify subscription fees that are a multiple of the competition, FNH decided to look for a tool that can help them be even better at their job.
At the core of the customer interaction is a CRM. FNH is using Virtuagym, which is a fitness studio software that is more of an appointment-booking platform than a CRM. Virtuagym is missing some very important features, one of which is the tracking and logging of customer interactions. The “KK-Liste” is being used to do this because Virtuagym’s API is unfinished and offers no automatic synchronisation.
Before choosing to build a new CRM, other solutions were considered as well. This includes e.g. Salesforce which turned out to be too expensive as licenses for every employee would have to be bought. Additionally, it offers a large amount of functionality that is not needed.

Procedures, technologies

To support the client’s future expansion plans, an important requirement is to develop flexible product. Because of this the project was split into a frontend and a backend application. To have the components as uniform as possible and allowing them to run on different platforms web technologies were chosen. For the frontend, the JavaScript software library React is used. The backend runs on Node.js. This way, JavaScript is used for both the front- and backend. The application has a modular structure and can easily be extended with additional functionality. This enables the implementation of optional features in the future and thus to continuously extend and improve the application with new useful features.
For the design, the goal was to realize the client’s vision, while also bringing in new ideas. This was achieved with the extensive use of mockups in the first phase of the project. There was a lot of client feedback and input during the project. To handle these inputs the development had to be agile.

Figure 2 - Related technologies

Results

A web-based CRM prototype was developed that meets the customer's desired business cases. A web application with different roles to allow different tasks to be performed. It allows the customer to work more process oriented in the future and have a leads journey be logged through the entire process: From leads registering on a website or manual registration by a personal trainer in the studio, to converting the lead to a customer. To build the application as extendable as possible was one of the top priorities.

Due to the choice of using web technologies, the application can be run on the Windows tablets used in the studios as well as the computers in the back office. Both front- and backend as well as other infrastructure are hosted in the cloud for easy scalability and extensibility.

Outlook

All functions requested by the customer, that were deemed feasible, were implemented. This makes it possible to map a leads journey until he becomes a customer with custom workflows. After that, all a customer's data including measurements, contracts and interactions can be added and managed. FNH had many new ideas along the way, which led to many optional features. These were continuously documented to be implemented outside of the thesis scope.

The flexible and expandable design of the architecture makes it possible to easily add new features or components later on, e.g. a mobile app for the booking of appointments by the studio clients themselves as described in the implementation study in appendix J.

Content

1. Introduction and overview

1.1 Description of the initial situation

As with every company, a central CRM is the core of good customer management. FNH's personal trainers and management have a lot of tools in use for different tasks. The problem is that a lot of tasks need the customers information. Because this information is not in one place, either it is being accepted that the newest data is not included, or exports and import have to be performed before every action e.g. sending a newsletter to existing clients.
With online sales becoming more and more important, tools for the sending of newsletters (Mailchimp) and the creation of sales pages (Click Funnels) gained a lot of relevancy. The software-landscape inside FNH changed so much, that Mailchimp took over some part of the CRM functionality. To automate the communication between different applications, Zapier is being used.

FNH's goal is to unify and simplify their processes. The aim is to create a central software that makes all of the important data available in a single database. This will also make it easier to analyse the data and use this information to optimise processes and improve customer loyalty. As part of the bachelor thesis, FNH commissioned a web-based application and supporting infrastructure that could fulfil their business case.
To successfully complete this task, a requirements analysis should first be conducted and discussed with the client. The aim is to understand FNH's ideas as accurately as possible and to steer them in a direction that can be realized. The client wants a very flexible and modular application, that he can extend at any time. It is also important to him that after the completion of the bachelor thesis, it is as easy as possible to find a software developer, who can quickly familiarise himself with the application. To support this, a flexible infrastructure must be developed around the core, the CRM, and a programming language that is as popular as possible must be used.

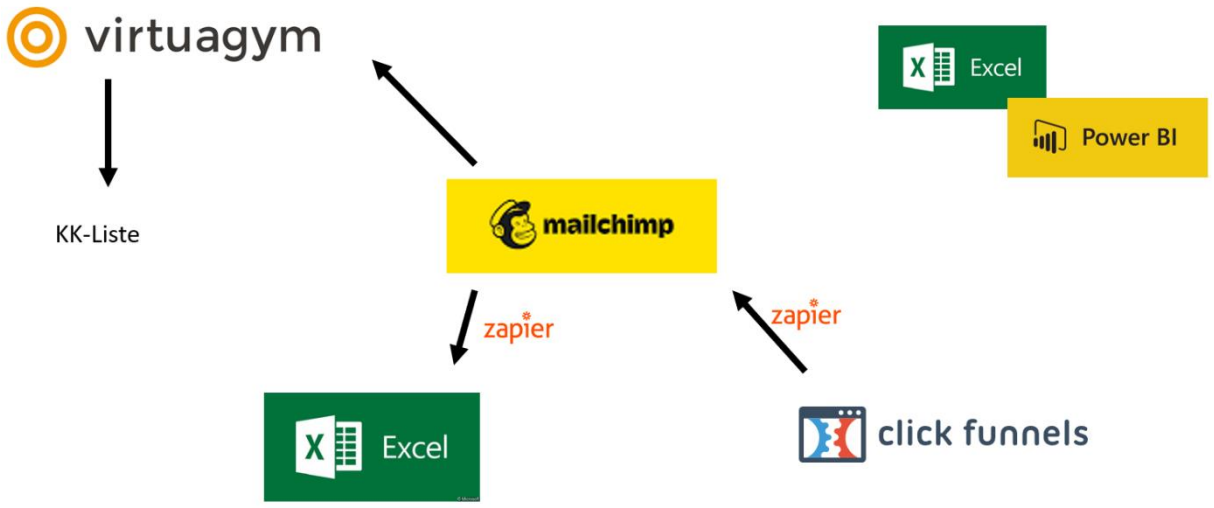

Figure 3 - Current FNH landscape

1.2 Preliminary work

There was no preliminary software development work on this project.
The client FNH has been working in the fitness business for years and has tested a lot of different fitness studio applications as well as CRM-systems but was never happy with it and therefore changed it every few years. FNH sees an opportunity to stand out of the competition by offering a more personalized service to their customers. To do so, they need an application that exactly fits their needs.

Analysis of existing products
Different studio software and CRM tools were analysed, a few of them will be explained here:

- Virtuagym
  Tool to manage everything inside a fitness studio. This includes the booking of appointments by the customer themselves through an app and documenting the client's activity.
  Problem: Marketing is an important means to acquire new customers. Virtuagym does not offer any connection to other tools like Zapier to integrate marketing tools e.g. ClickFunnels, Mailchimp, that would allow the automation of sales processes and decrease the number of leads that are being lost in the process.
- Salesforce
  Is a great and very popular CRM. It allows to track a leads journey, convert it to a customer and manage it thereafter.
  Problem: Because every trainer needs to have access to all the data, every employee would need a license. This yearly cost would be too expensive for the studio. It has many features that are not needed by the studio and therefore adds a lot of unnecessary complexity.
  Because the studio needs an appointment booking platform and the possibility to register certain measurements, a separate application would need to be integrated.

This is why FNH decided to build their own application that exactly fits their needs.

1.3 Vision

The vision is to provide a simple CRM with a scope of functions limited to a fitness studio's needs. It allows to log all the data in a simple GUI but make complex analysis for the studio management. Because it is web-based, it can be used on a handheld device to document the trainings in the studios. As the processes are structured, no leads will be lost and the profit with fitness subscriptions can be increased. Thanks to all the easy availability of user information, it is possible to offer the leads/customers a more personalized experience.
FNH's advantage over a standard software development company is their know how in the fitness business. Therefore, it would be the next natural step for them to sell the tool to competitors as soon as the development process is completed.

2. Requirements analysis & design

The requirements analysis is an important step before starting with the software architecture. It is essential that the customer's idea and wishes are understood and that all parties involved have the same understanding of the requirements.
Based on the mission statement and the first meetings with the customer, the requirements analysis was conducted. This resulted in the User Stories, with the help of which it was possible to derive the functional requirements, such as the Use Cases and the first Optional Features, as well as the non-functional requirements. These mockups made it easy to visually record and convey the requirements that were developed. This ensured that there was a uniform understanding of the functions. To visualize the interaction of the individual screens and components, the domain model including a sequence diagram for a complex process was created.

2.1 User stories

User stories describe features and user interactions in an informal way. The goal is to understand the users and their motivation in using the different features [2]. There are different user types, that interact with the application. The owner manages the application and can create new locations and personal trainers. The personal trainer is interested in his customers data, which he wants to have at hand as quickly and easily as possible.

US01: Studio management
As the owner of a personal training studio, I need to manage the studios online. I need to manage my personal trainers, the physical training locations and more. I can create, view, edit and delete this information using the web application.

US02: Data analysis
The CRM of a personal training studio stores a lot of data about the customers, as well as the operation. As owner I want to analyse this data directly on the web application or using an external tool. The analysis might be simple, like seeing how many customers there are in specific age range or more complex, like finding the correlation between advertisement and money spent in the studio shop.

US03: Customer relation
As a personal trainer I need to have a close connection to my clients. I want to make sure that they train regularly and achieve their goals, such as weight loss or muscle gain. Before starting my workday, I open the web application and look at the list of the customers, that are assigned to me. The application tells me if a lead / customer has not been contacted for a while and reminds me to do that. I can check the customer’s training data, in-studio sales, as well as the interaction history. During the day I can then contact the user and talk to him about his training. I make sure that he is following the nutrition plan and maybe book a personal training at the studio for him. After contacting the customer, I can track the conversation in the application, so no information is lost for future interactions.

US04: Customer acquisition
Acquiring new customers can be a long and hard process. It starts with a potential customer being registered as lead. This can happen through many ways, like the customer clicking on an advertisement of the studio and filling out a form or being recommended by an existing customer. Converting a lead to a customer can consist of many different steps like calling him, mailing him or setting a first appointment. There can be many kinds of clients, so the process is not the same for everyone. As an owner I can create tailored workflows for different client categories. As a chief personal trainer, I can assign the right predefined workflow to a lead. As a personal trainer, I go

through the workflow. The application guides me through it, and I register every step. Once the lead is converted to a customer, I still know what process he went through and how he became a lead in the first place.

2.2 Functional requirements

2.2.1 Overview

This is an overview of all use cases and the optional features that have been developed, including their prioritization. The goal is to implement all the use cases with priority one to three. Priority one is a must goal, priorities two and three are desirable.

Use case	Description	Prioritization [1 – 3]
UC1	CRUD users (owner, chief personal trainer, personal trainer)	1
UC2	CRUD fitness studios	1
UC3	CRUD lead/customer	1
UC4	Filter customers	3
UC5	Customer-Trainer assignment	1
UC6	Fitness studio assignment	1
UC7	CRUD customer interactions	2
UC8	CRUD customer documents	3
UC9	View customer report	1
UC10	Customize workflow	2
UC11	Manage workflow	1
UC12	CRUD contracts	2
UC13	Add measurements	3
UC14	Add customer training notes	1

Table 1 - Use case overview

The optional features are categorized by priorities of one to three as well. These are desired goals that will be implemented if time permits.

Optional features	Description	Prioritization [1 – 3]
OF1	Archive customer / lead	1
OF2	CRUD products	1
OF3	Show contract that expire soon	1
OF4	Contact customer	3
OF5	Bring a friend / brought by personal trainer	2
OF6	CRUD in-studio purchases	3
OF7	View customer billing status	3
OF8	Personal Trainer dashboard	3
OF9	Customer billing	3
OF10	CRUD checklist	3
OF11	Additional workflow information	3
OF12	Global search	3
OF13	Measurement external devices	3
OF14	Visual customer progress	3
OF15	Add note as PDF	3

OF16	Consultant form	3
OF17	Contract as webform	3
OF18	Profile picture	3
OF19	Fitness-device settings	3
OF20	Timestop a subscription	2
OF21	Timer in “FNH Fitnesstest”	3
OF22	Automatically generate chief personal trainer	2
OF23	Door opening system	3
OF24	Information button on measurements	3
OF25	Show a user’s birthday	2
OF26	Body fat calculator on calipometry measurement	3
OF27	Lead information when converted to customer	1
OF28	Collection basin leads	2
OF29	Mailchimp connection	3

Table 2 - Optional features overview

Restrictions

The following restrictions were agreed upon with the customer:

- Booking platform will not be developed in the scope of this bachelor thesis.
- An app for the customer to e.g. make bookings will not be part of the scope.
- Door opening system will not be part of the scope.
- The configuration and connection of Power BI will not be part of the scope.

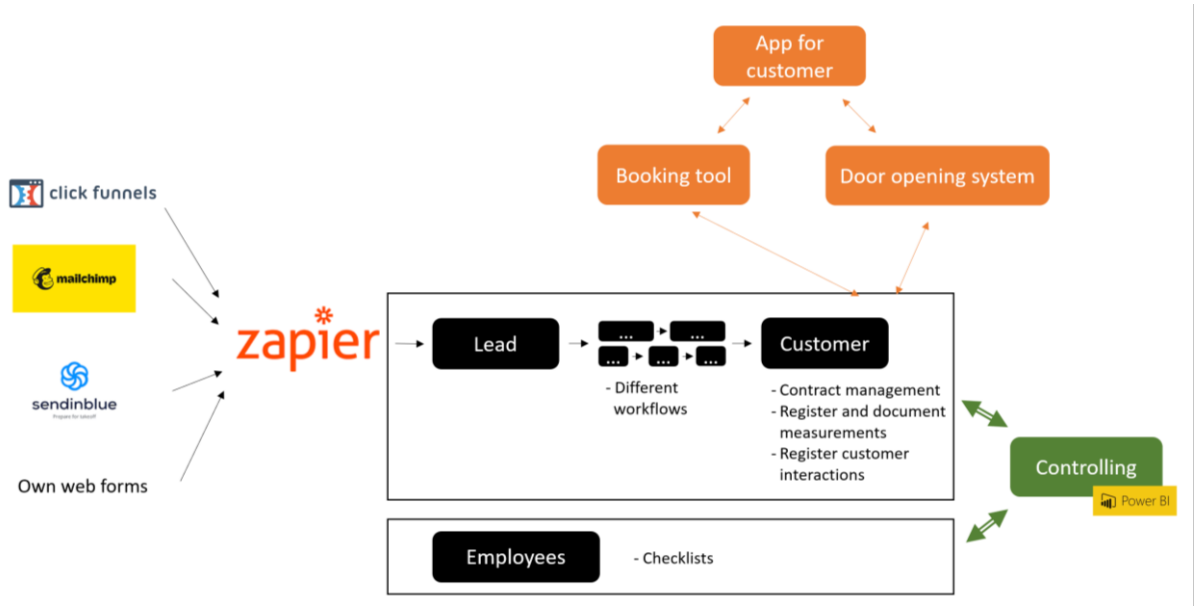

Figure 4 - System overview

Dependencies

The following dependencies exist:

- External Zapier API
- Hardware used by the studio staff

2.2.2 Use cases

Actors

A1	Owner	Owner of the software, has full access (including A2 and A3)
A2	Chief personal trainer	Chief personal trainer per location, has the right to set a workflow to a lead and all the privileges of A3.
A3	Personal trainer	The main user of the application

Table 3 - Actors

Use case diagram

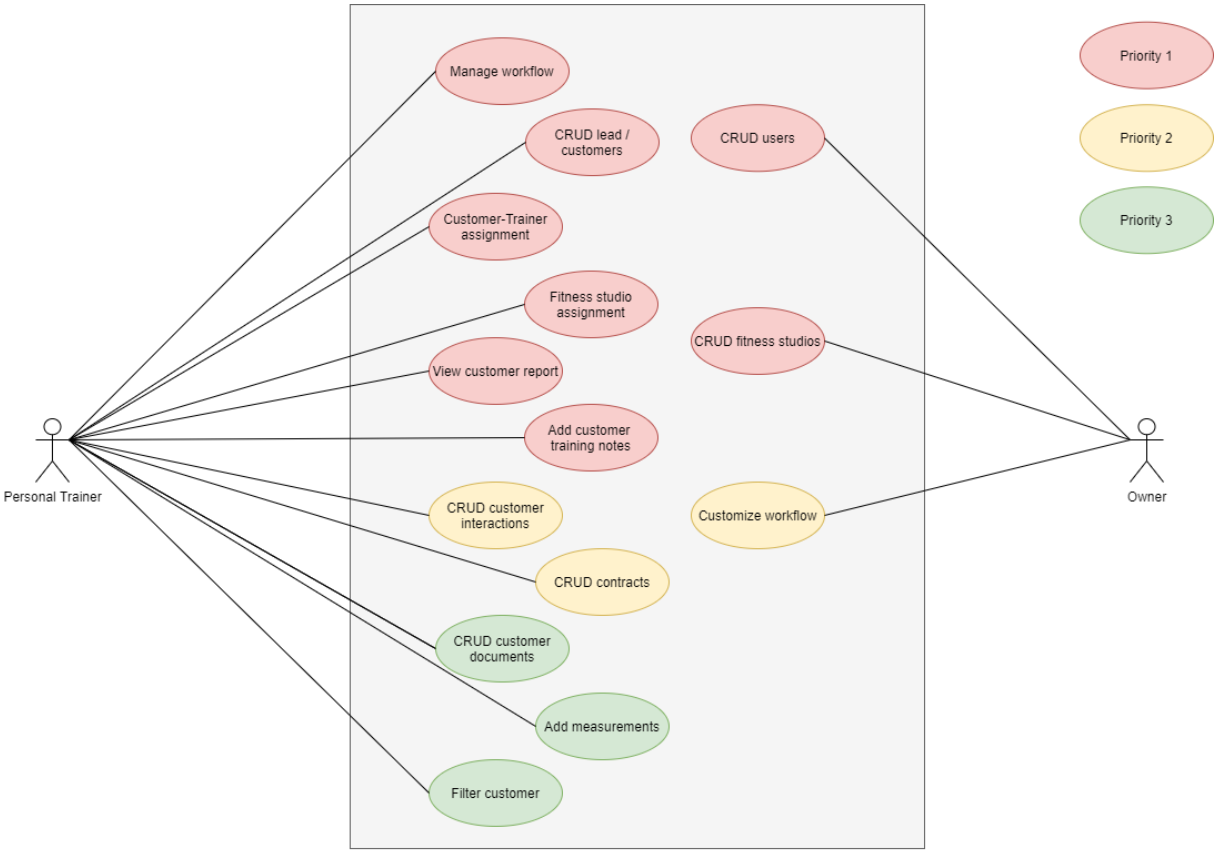

Figure 5 - Use case diagram

Use case description

UC1 - CRUD users (owner, chief personal trainer, personal trainer)
The personal training studio administrator can manage his personal trainers using the web application.

UC2 - CRUD fitness studios
The personal training studio opens a new location. The administrator can register this new studio location using the web application.

UC3 – CRUD lead/customer
There is some basic information linked to every customer: name, age, address, interests and more. A personal trainer can manage the customer information using the web application.
Show the customers / leads in colour (orange, red) according to the time to their last interaction.

UC4 – Filter customers
The customer list can be very long. The personal trainer needs to be able to filter and sort the list according do different criteria.

UC5 – Customer-Trainer assignment
Every customer is assigned to a personal trainer. Depending on preferences, working hours or other reasons, a personal trainer can assign a customer to a different trainer.

UC6 – Fitness studio assignment
Customers can change the studio to attend the personal trainings at a different location. The personal trainer can manage the studio assignment.

UC7 – CRUD customer interactions
For a personal trainer, it is very important to keep track and document all customer interaction. He might forget some things about a customer, or the customer might be assigned to a different trainer. To avoid the loss of information, the personal trainer registers every customer interaction in the application.

UC8 – CRUD customer documents
Important customer documents like contracts, notes or invoices are stored in the application. Documents can be uploaded as pdf and downloaded again later.

UC9 – View customer report
Before contacting a customer or preparing for a personal training, the personal trainer can check the customer report. It shows the most important customer documentation, such as name, address, telephone number etc. and also problems, targets and measurements in compact form. This allows the instructor to quickly prepare for customer interactions.

UC10 – Customize workflow
Converting a lead to a customer requires a customizable workflow. The owner can create and save workflows using an intuitive tool. He can choose which steps are necessary and what order they have to be done in.

UC11 – Manage workflow
The personal trainer is responsible for converting a lead into a customer. Using the application, the trainer can track the workflow visually and see which step the user is currently at. Once a step is done, he registers the interaction and the workflow is updated. The workflow a customer came from needs to be tracked for future use.

UC12 – CRUD contracts
To have all the customers data digitally in one place, the contracts have to be online as well. The personal trainer can download a contract and let the customer sign it. Information like expiration date and price of the sale have to be registered manually.

UC 13 – Add measurements
To track the customers progress, certain measurements will be performed regularly. These have to be registered in the application to evaluate the performance over time.

UC14 – Add customer training notes
Because every customer is different and has different problems and goals, the trainings are protocolled. This way, every training can be built upon the previous one and progress can be achieved. If the customer should change the trainer, he already has all the necessary information.

2.2.3 Optional features

During the project, the customer as well as the developers had some ideas about how the applications could be extended. They were documented with a short description. The result is a list of possible optional features that would be implemented based on the customer's prioritization if there was time left at the end of the project.
Below is a list of all the features with the highest priority, including the ones that were implement. The full documentation of the optional features can be found in the appendix.

OF1 – Archive customer / lead
When a customer cancels his subscription, he might not be gone forever. If he comes back before a set amount of time, his past training data and interactions still need to be around. To achieve this, a personal trainer can archive a user, when he is temporarily unsubscribed.

OF2 – CRUD products
The various products of a fitness studio can be created, edited and deleted by the owner. This includes for example the different subscriptions or the personal trainings.

OF3 – Show contracts that expire soon
When a customer's contract is about to expire or has expired, this should be clearly indicated so that the personal trainer knows he should contact the customer.

OF15 – Add note as PDF [implemented]
Attach handwritten notes and drawings as PDF to a client under the submenu "Notes". This enables to save notes that cannot be specifically assigned.

OF22 – Automatically generate chief personal trainer [implemented]
Automatically convert to chief personal trainer if responsible for a studio is assigned

OF27 – Lead information when converted to customer [implemented]
Lead is being informed when converted to a customer. This could e.g. be used to send him login data for an appointment booking app.

2.3 Non-Functional requirements

2.3.1 Preliminary remarks

The non-functional requirements (NFR) for this project are specified according to the ISO standard 25010. [3]

2.3.2 Requirements

Functional suitability

NFR1 The development team defines the functionality described by the client. The development team develops the functionality according to prioritization agreed upon by client, dev team and supervisor.

Performance efficiency

NFR2 When clicking on a customer in the application, it should take less than 2 seconds to display customer information on a DSL connection in Switzerland.

NFR3 The backend should be able to process and respond to 100 requests a minute.

NFR4 Every page of the web application should not take more than 200ms to load on a DSL connection in Switzerland.

Compatibility

NFR5 Zapier sends an HTTP REST call to the system, with information about a new possible customer. The system records this as a new lead.

NFR6 The database needs to be accessible to be analysed with Power BI.

NFR7 The Web Application should run on Chrome, Edge and Safari on a Microsoft Surface Tablets with OS Version 10, on Microsoft personal computers with OS Version 10 and on Mac personal computers with OS Version 10.15 “Catalina”.

Usability

NFR8 The frontend needs to be reachable over the internet on a domain provided by the client.

NFR9 Three of four test users need to be able to execute two predefined and specific functionalities from the following list, without prior knowledge:
- Signing a contract
- Register a customer interaction
- Add a workflow to a new lead

NFR10 Three of four test users should rate their UI experience (criteria: layout, responsiveness, colour, content) of the Web application using a tablet with a grade of at least 7 of 10 where 10 is the best.

Reliability

NFR11 The database should be able to handle up to 10'000 customer entries.

NFR12 Errors should not cause system failure but cause an error message and revert the system to the previous state.

NFR13 The stability of the application is guaranteed according to the SLA of the cloud service provider agreed upon by the dev team, client, and supervisor. (DigitalOcean)

NFR14 Every error should be logged by the system.

Security

NFR15 All communication to and from the front- and the backend needs to be encrypted using an SSL certificate provided by the client.

NFR16 Data entered into the web application input fields, will be validated before being processed by the system. SQL Injection tests on the input fields should not show vulnerabilities.

NFR17 No user passwords are stored in plaintext in the database.

NFR18 When a call is made to the backend API by an authenticated user, only functionalities permitted to his role are executed by the system.

NFR19 When a user is logged in on the web application, he can only access pages, that his role is permitted to.

Maintainability

NFR20 The business logic in the backend should be modular, so that it can be expanded in the future.

NFR21 Business logic functionality needs to be testable using unit tests.

NFR22 The backend API needs to be testable using API testing tools.

Portability

NFR23 Database, backend and frontend should all be deployed on different instances.

2.4 Domain analysis

Contains the domain model and the most important sequence diagrams. It provides an overview of all concepts and system relationships. Thanks to the domain analysis, the interaction of the individual components and screens can be visualized in the domain model. It shows which components are needed and how they work together. With the help of the domain model and the sequence diagram, the database model can be created, and the interfaces defined in the architecture phase.

2.4.1 Domain model

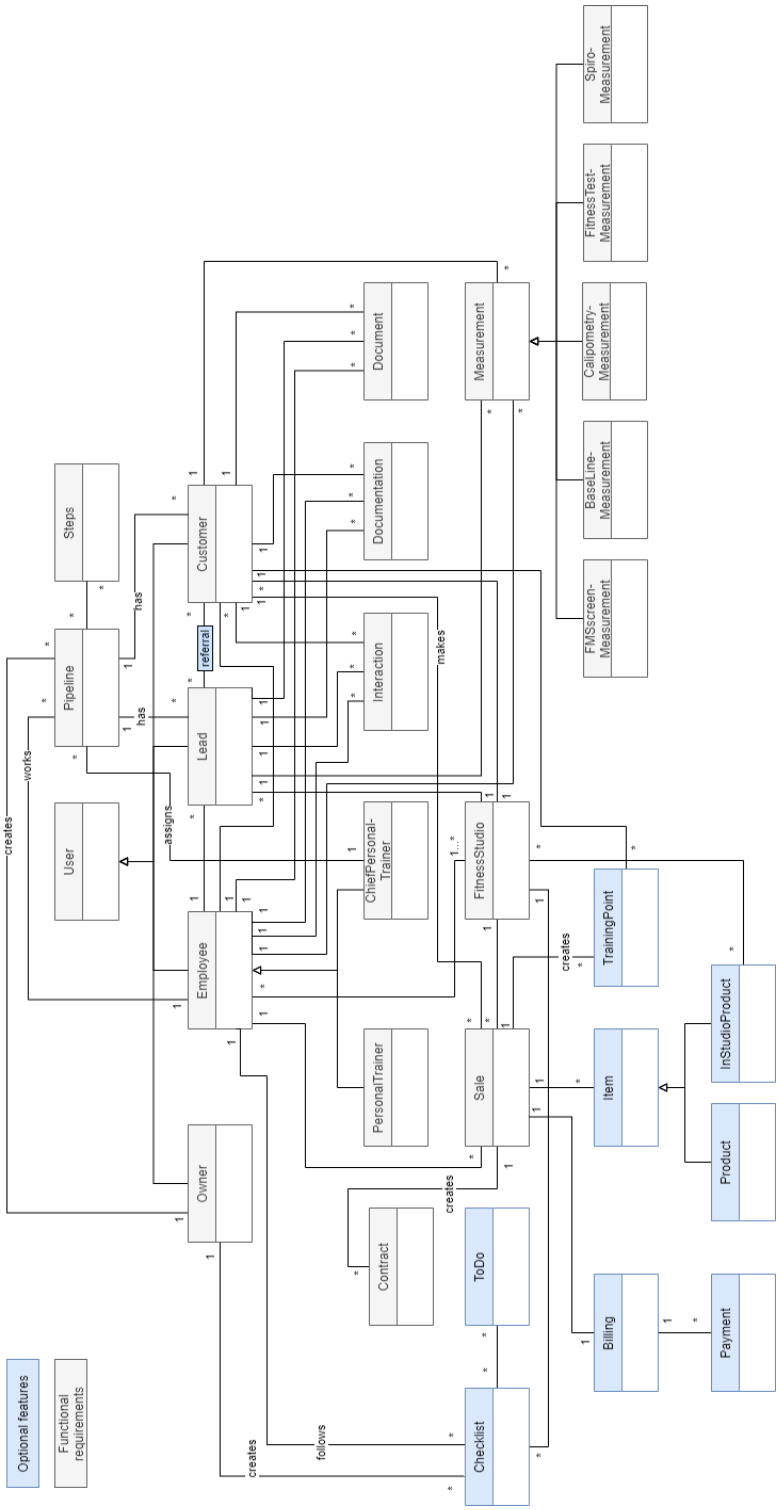

Figure 6 - Domain model

In the following table the most important objects are highlighted and provided with a short description.

Lead	A lead is somebody interested but did not purchase a subscription yet. As soon as he signs a contract, he becomes a customer. Every lead goes through a predefined process called workflow before becoming a customer. A lead contains all necessary data of the customer such as first name, last name, e-mail address, etc.
Customer	To become a customer, one has to buy a subscription, before that, he is a lead. The customer keeps all the data when converting from a lead. A customer can bring new leads by referring the studio to somebody else e.g. a “bring a friend”-campaign. All the leads brought by the customer will be listed on his profile.
Measurements	A measurement is one or many values e.g. body weight. A personal trainer can take different measurements. They will be documented to analyse later and for tracking a customer’s progress.
Interaction	An interaction is a communication between a customer and an employee. This can take place in different ways, by telephone, SMS, WhatsApp, email, etc. Interactions should be simple to record and easily accessible for further use.
Documentation	A documentation belongs to a customer and contains important information about him/her. Like injuries, preferences, or information about the trainings. An employee can create notes.
Document	Documents such as notes, invoices, consultation notes can be uploaded to a customer. Contracts are a special type of document. The contract will be downloaded as a standardized / static pdf, filled out on the tablet and uploaded again. The difference to a normal document is that additional fields like value (in CHF) and the expiration date have to be filled out.
Workflow	For a lead to become a customer, he goes through a workflow. A workflow consists of several steps. One such step can be for example making a consulting appointment. The owner can create workflows and arrange these steps as desired. An employee can only use the existing workflows.
Sale	A sale is a transaction, where a customer buys something from the studio. The items that can be sold are either in-studio purchases, such as protein powder or fitness bands or studio products, such as 10 personal training points. Every transaction is supervised and assigned to a personal trainer. Things like proteins and fitness bands are sold as extras inside the studios. To track the sales, these can be assigned to the customer that bought them.
Checklist	The checklist can be created by the owner and the employee can apply it. A checklist consists of various todos, e.g. contact a specific customer, clean the fridge. It is there to support the employee in his daily work. It must be possible to "tick off" the checklist and to read and evaluate them again at a later date. An employee can have several checklists for different areas/activities.

Table 4 - Domain model description

2.4.2 System sequence diagrams

The following sequence diagram should give a better understanding of how the components are working together to manage the contracts in the frontend by the personal trainer. This was done by the example for UC12.

UC12 - CRUD contracts

The personal trainer is in a consulting meeting with a lead that decides to buy a membership and signs the contract right away. After signing it, the personal trainer has to upload the contract and fill in some additional information for analysis because there is no text recognition at this point. Preconditions are that the personal trainer is logged in.

The following procedure is illustrated:
1. Navigate to the lead
2. Displaying the leads information
3. Downloading the contract
4. Uploading the contract

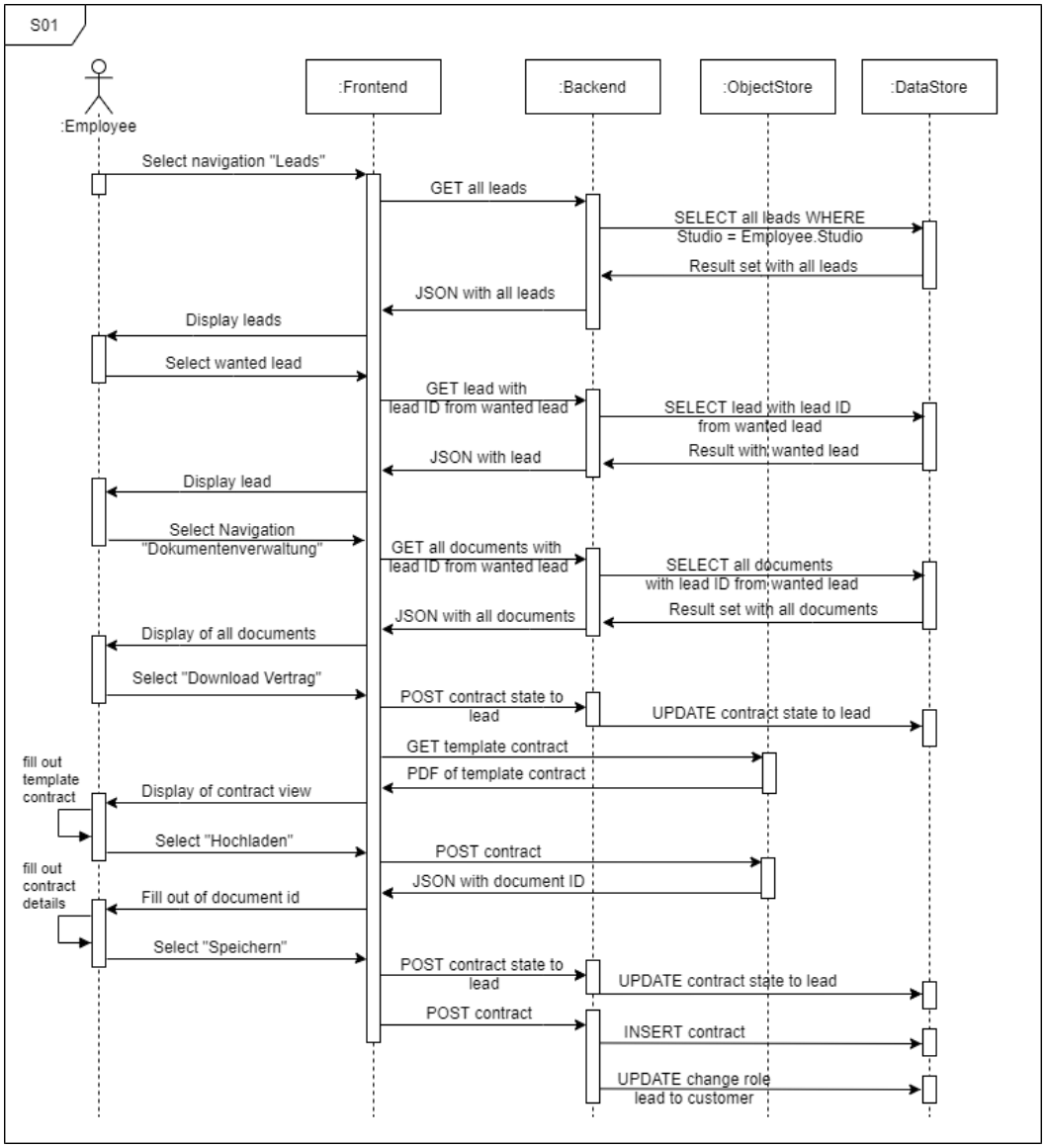

Figure 7 - System sequence diagram S01

2.5 Mockups

The style of the final product may deviate from the mockup shown below. It is used to show the functionality and helps to set clear requirements. The mockup was created with Adobe XD. The latest version is being compare to the actual implementation, where it is interesting to explain the design process[1].

2.5.1 Frontend screens

The mockup serves as a common basis for discussion about the structure and functionalities of the frontend. This gives the customer an exact idea of what he can expect at the end of the bachelor thesis. The following mockups do not contain any optional features. Except for the search and navigation to the personal trainer dashboard, they have been included because it was important for the customer to see that they are globally placed and accessible from every screen. The mockups with optional features that have been created can be found in the appendix. In order to give an idea of how the mockups were implemented, some are compared to the final design implementation. The lead and customer screens are almost identical, with small differences, which is why only one of the two screens is shown.

Login screen

Because only authorized users are allowed to use the application, a login is required. This way different roles can be setup to have different permissions. E.g. create new personal trainers, only see customers of one studio.

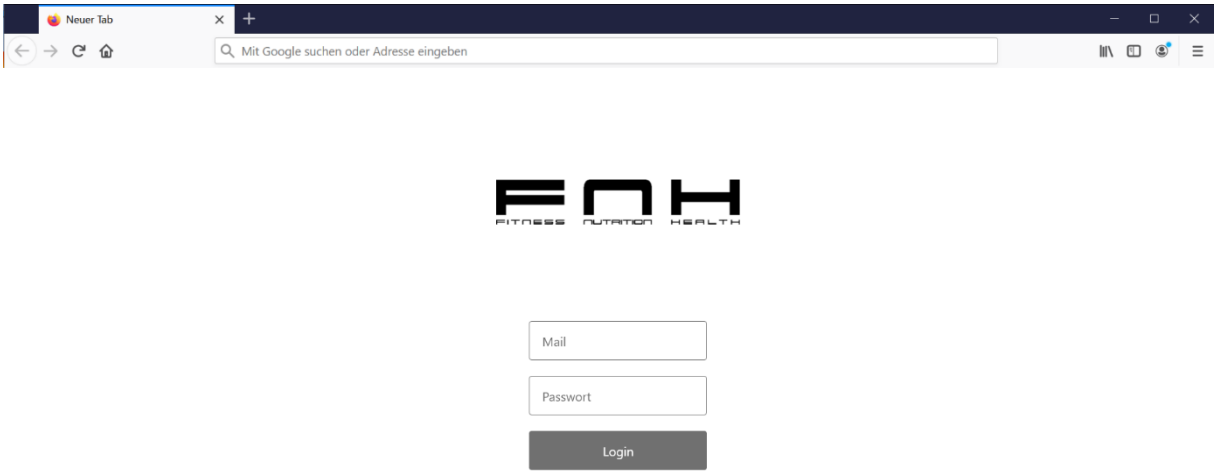

Figure 8 - Mockup – Login screen

[1] https://xd.adobe.com/view/b2d76610-6c41-4a49-8b0a-9ca3895a0f2a-e784/?fullscreen

Lead screen - Overview

On this screen the overview of all leads is shown. In a later stage, it will be integrated into the personal trainer's dashboard. The recorded leads are displayed in a table. By clicking on one, its detail page with all the information and actions will be shown. A new lead can be generated via the "Neu" button or entered into the system via the Zapier interface. A lead is highlighted if it has not yet been contacted or the last contact was more than two weeks ago. Leads can be filtered by studio, personal trainer and type. This reduces the effort to have a simple overview of only some leads.
The customer overview screen looks the same, except the missing "Neu" button, because a customer has to be a lead first. This is a requirement from FNH so that they can always track how a lead entered the system. The rest of the screens are identical with the difference that only clients are listed.

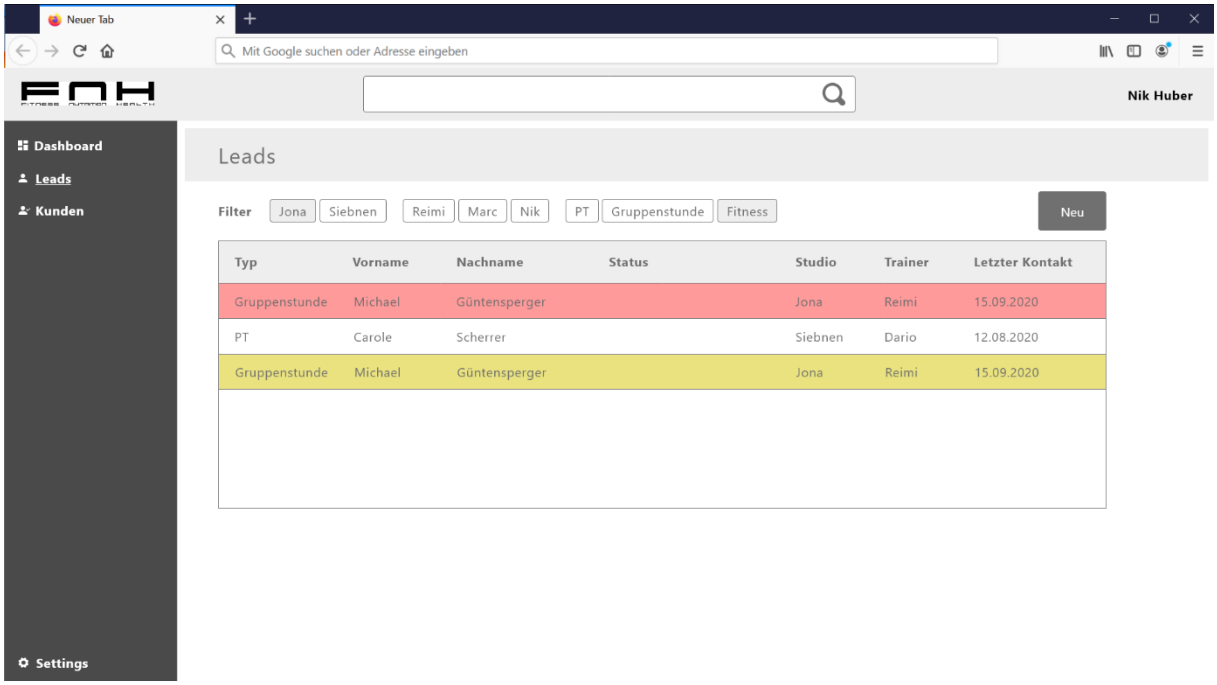

Figure 9 - Mockup – Lead screen - Overview

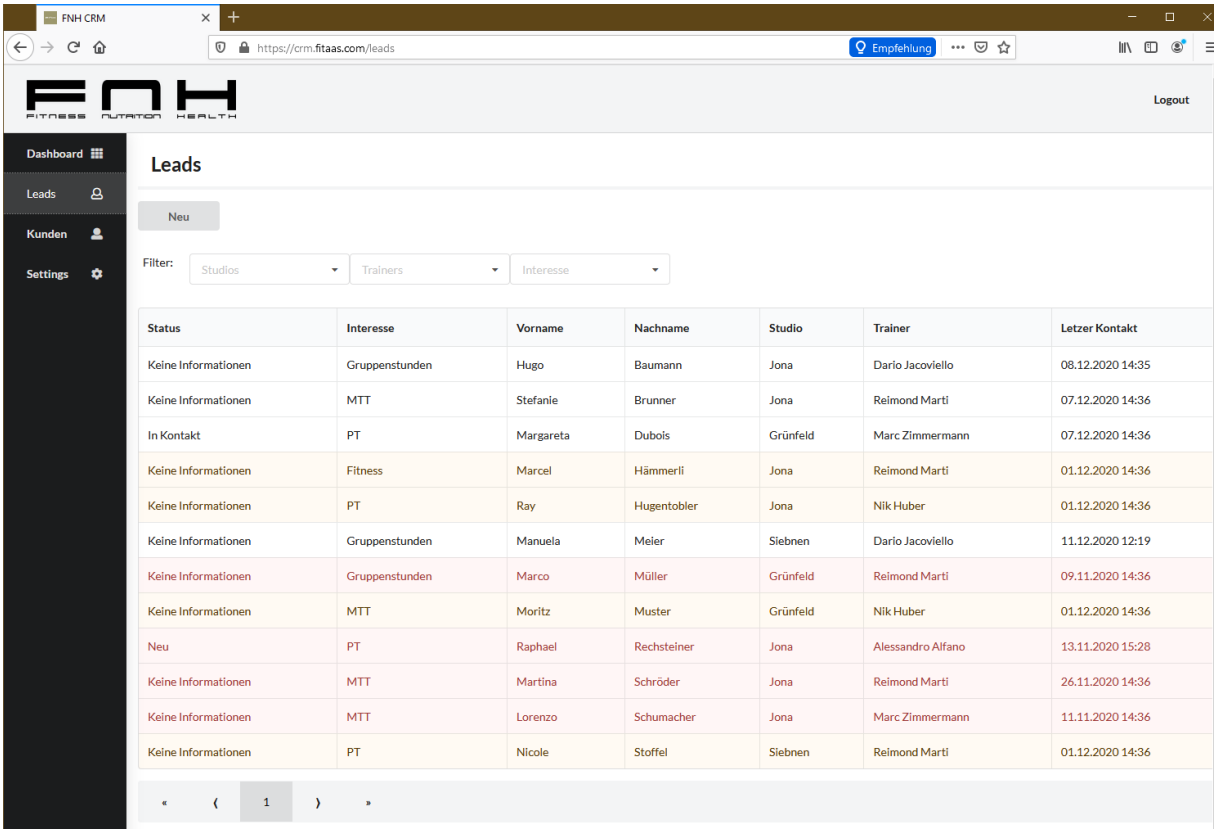

Figure 10 - Final version – Lead screen - Overview

Comparison to the final version of the frontend

The lead screen is optically almost identical to the mockup, only small differences are present. It is not the start page after login, this became the dashboard with a short welcome text. The filters were realised with dropdowns, which makes the whole thing clearer, especially if there are many entries. At the request of the customer, the column order was changed and the column "Type" was renamed to "Interest". In addition, it is now possible to filter by interest. To improve the performance when loading the page, pagination was introduced so that only 20 entries are loaded at a time.

Lead screen – Detail view - general

All the information about a lead can be found on this page. The data can be added, changed or deleted. The dashboard is only available to customers. When managing a lead, the focus lies on the workflow, which can be individually assigned according to requirements. The workflow can be assigned via a dropdown. The step in which the lead is located is clearly indicated by colour. The button to the right of it can be used to advance one step. Once the workflow is complete, a lead becomes a customer. A single workflow step has one of the following statuses: “New”, “In contact”, “Appointment set”, “Did not show up”, “Test for free”, “Not interested”, “No information”. In the future, “Exkunde” (OF 20) will be implemented, so that leads and customers can be archived instead of be permanently deleted from the system. In the customer view the workflow is not shown.

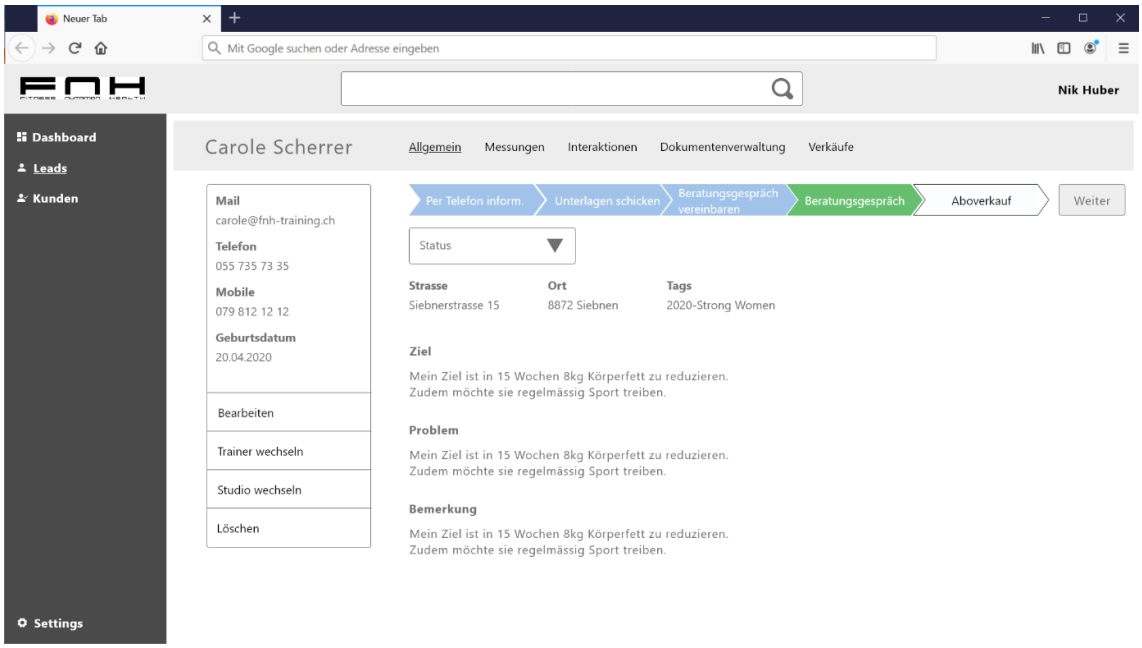

Figure 11 - Mockup – Lead screen – Detail view – General

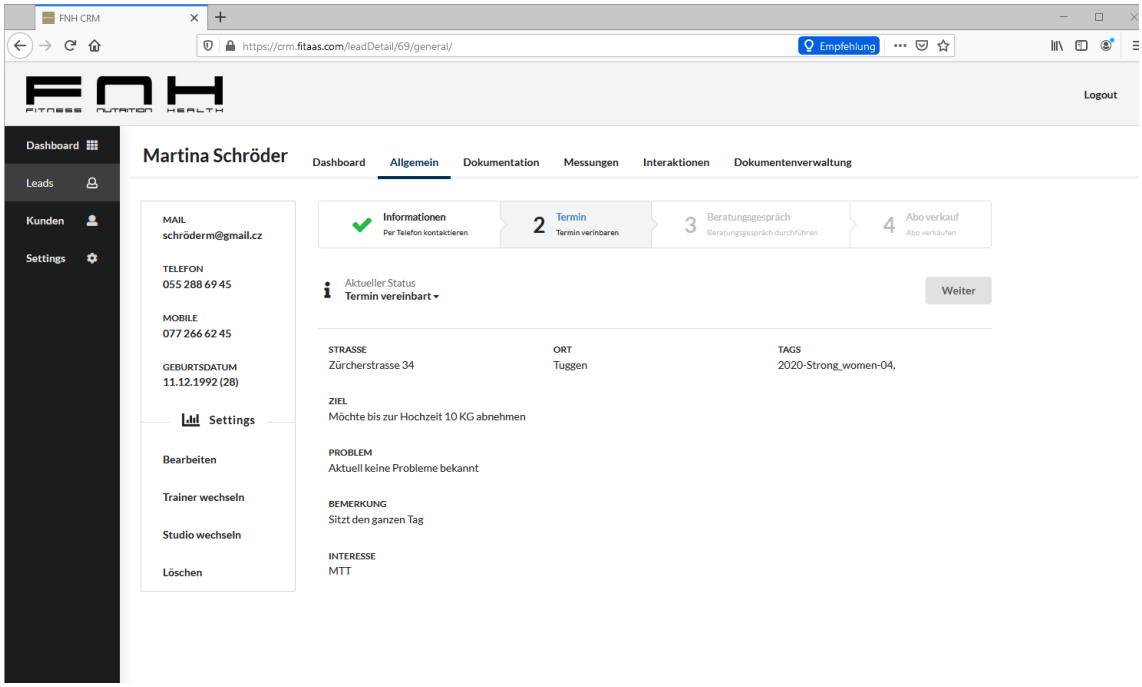

Figure 12 - Final version – Lead screen – Detail view – General

Customer screen – Detail view - Dashboard

When a specific customer's profile is opened, one lands on the dashboard. The most important information about the customer, such as goals, problems and measurements are clearly displayed. In a first step, the graphics are only displayed in a simplified form. Further actions like editing, deleting a customer, assigning him a new trainer or studio can be performed on this page.

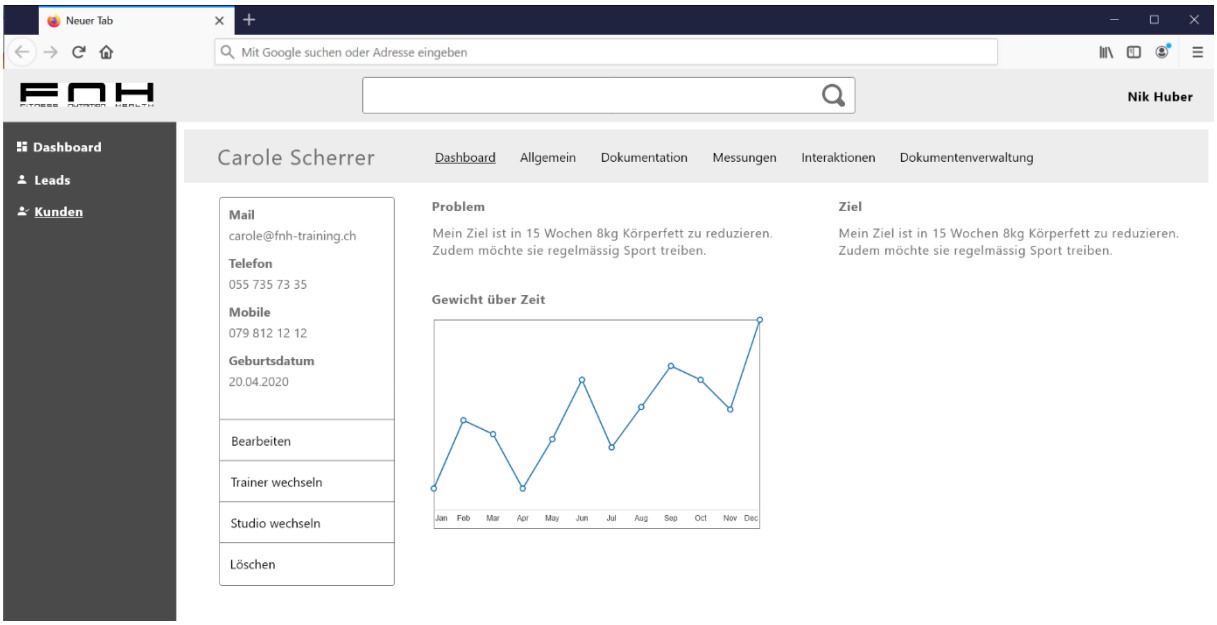

Figure 13 - Mockup – Customer screen – Detail view – Dashboard

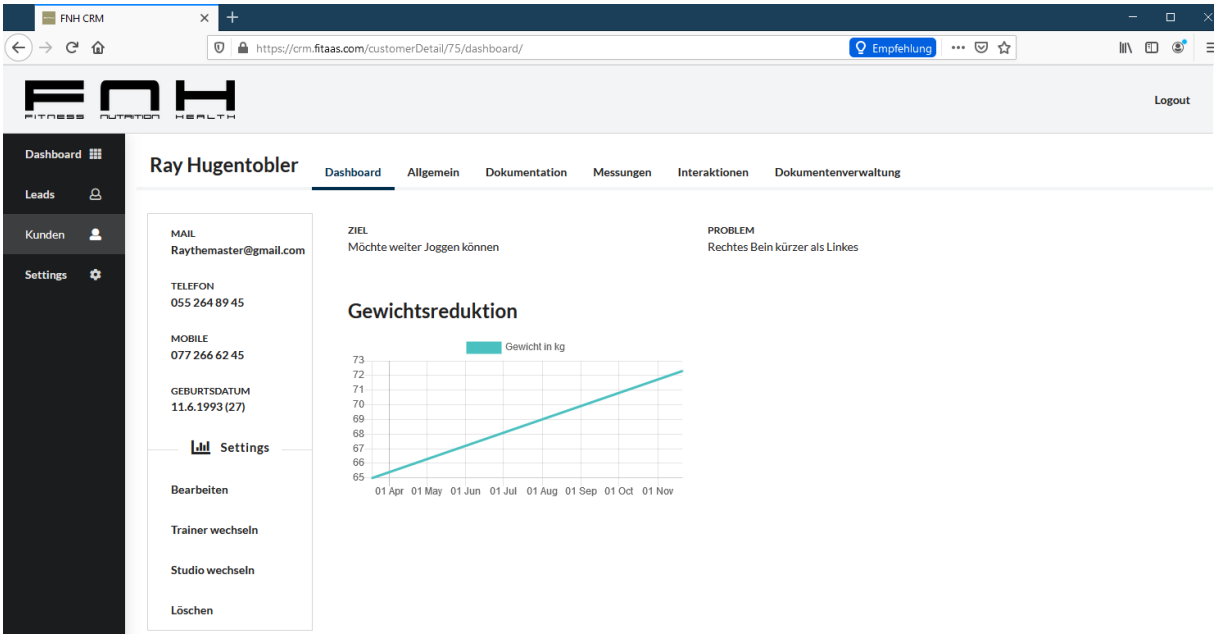

Figure 14 - Final version – Customer screen – Detail view – Dashboard

<u>Comparison to the final version of the frontend</u>

The customer requested two small changes to this screen. The order of "Ziel" and "Problem" was inverted, because the goal is more important to them. Additionally, the age is calculated automatically from the birthday date and shown next to it.

Customer screen – Detail view - Documentation

Documentation on a customer can be registered on this page. Via dropdown the personal trainer, that wants to register this documentation, can be selected. This helps to track who recorded an entry, if there should be any questions. Because there are different Documentation types, they can be selected as well. Currently there are three documentation types: general, training and nutrition. Beneath the input mask, the latest entries are shown.

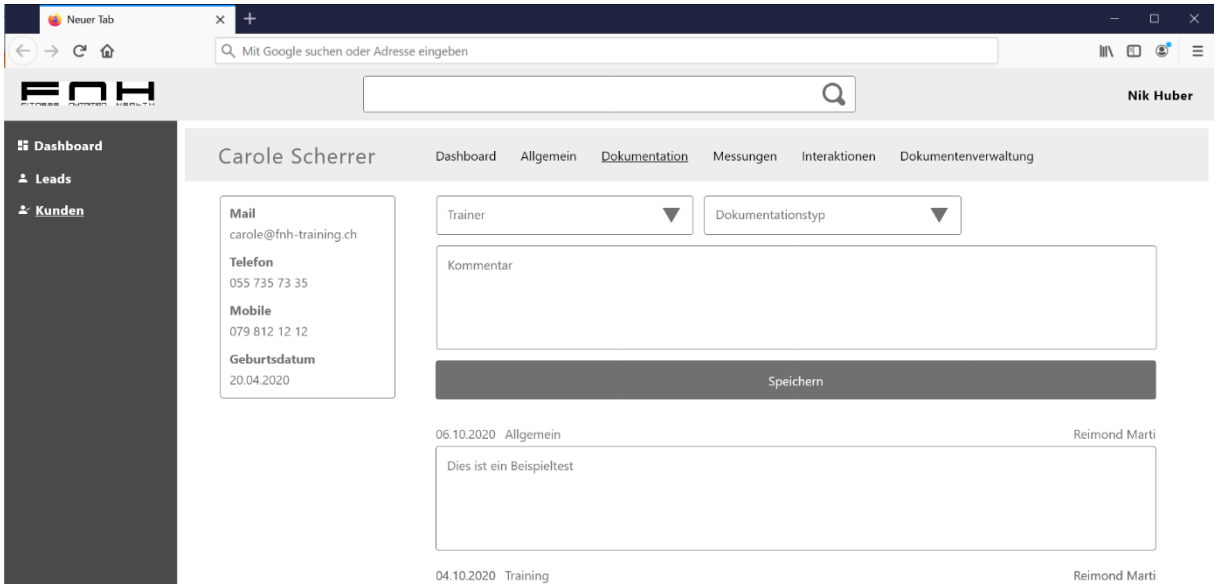

Figure 15 - Mockup – Customer screen – Detail view – Documentation

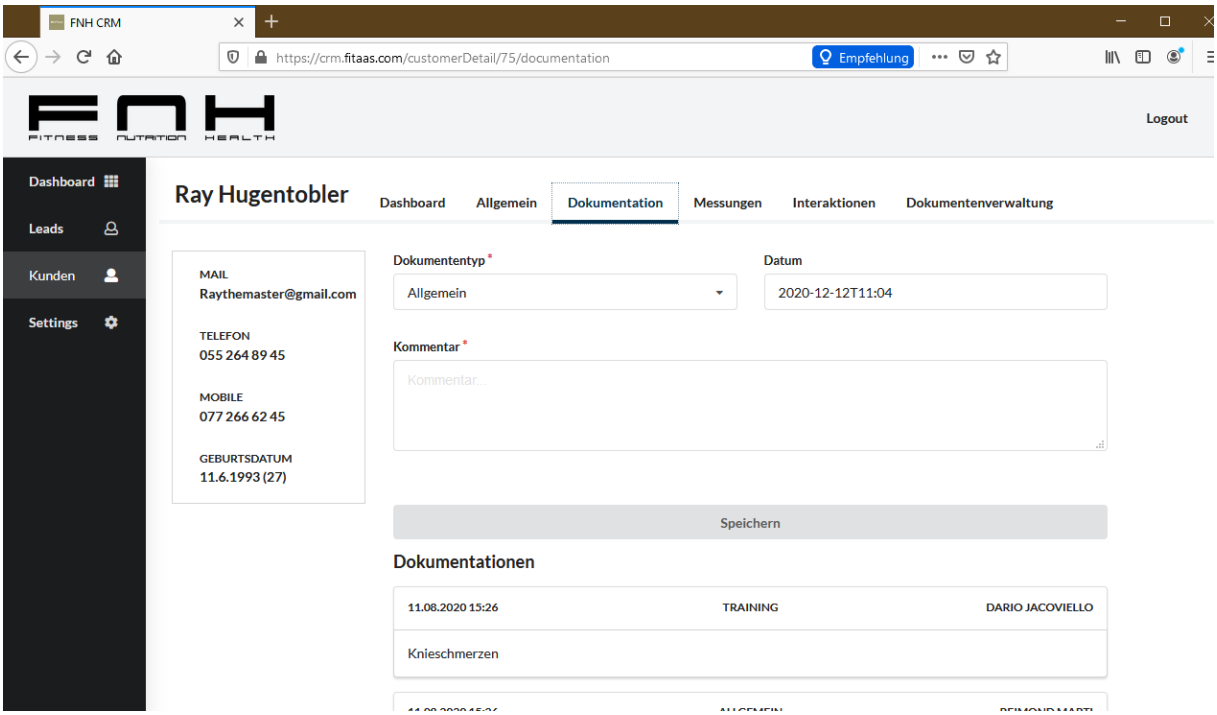

Figure 16 - Final version – Customer screen – Detail view – Documentation

<u>Comparison to the final version of the frontend</u>

The trainer dropdown has been removed, the registering trainer is identified based on the login, thus ensuring that incorrect data cannot be recorded in the name of other trainers. A date field has been added so that one can specify when the documentation took place.

Customer screen – Detail view - Measurements

In the submenu "Measurements" different types of measurements can be recorded, which are assigned to a customer. The most recent measurement is always displayed at the top. Depending on the type of measurement, different fields are required. Mockups of two of the five measurements are shown here. In the appendix F screenshots of all five measurements can be found.

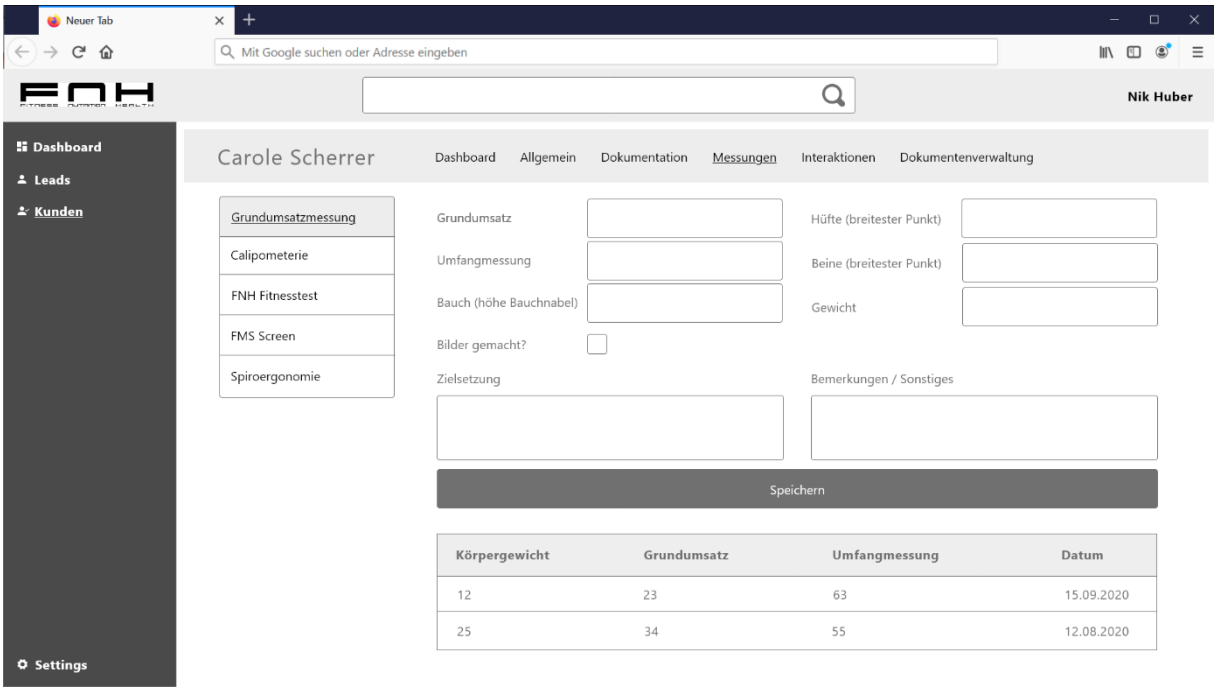

Figure 17 - Mockup – Customer screen – Detail view – Measurement – Basic turnover

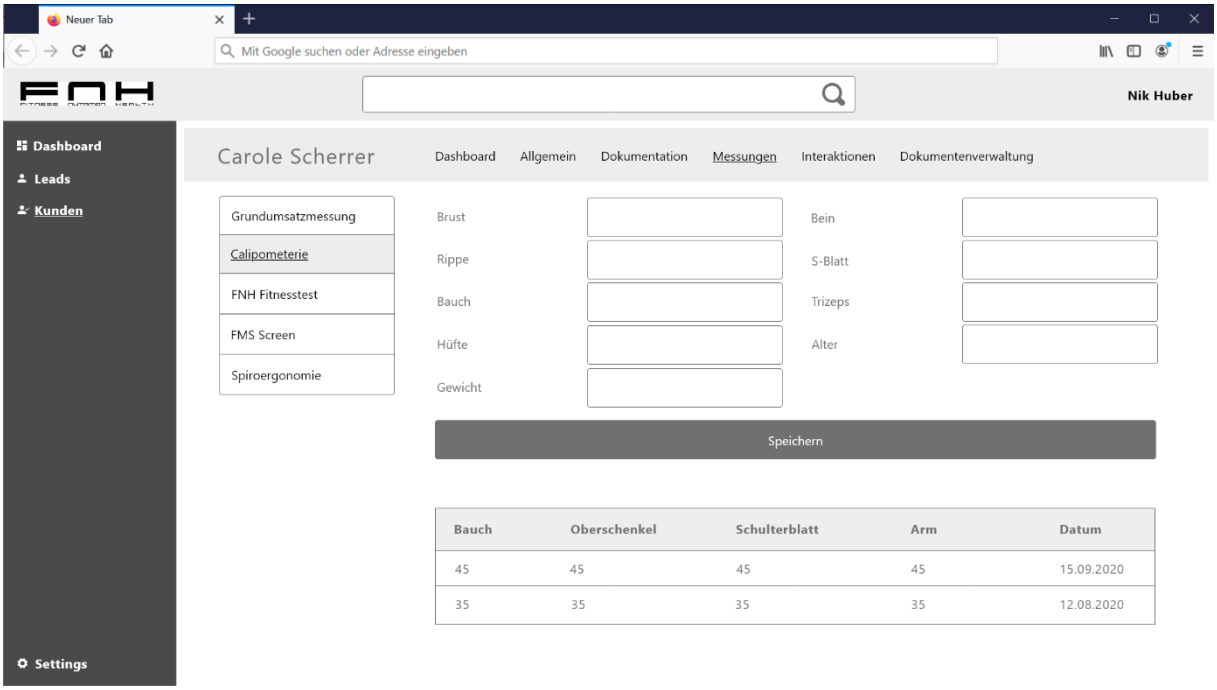

Figure 18 - Mockup – Customer screen – Detail view – Measurement – Calipometry

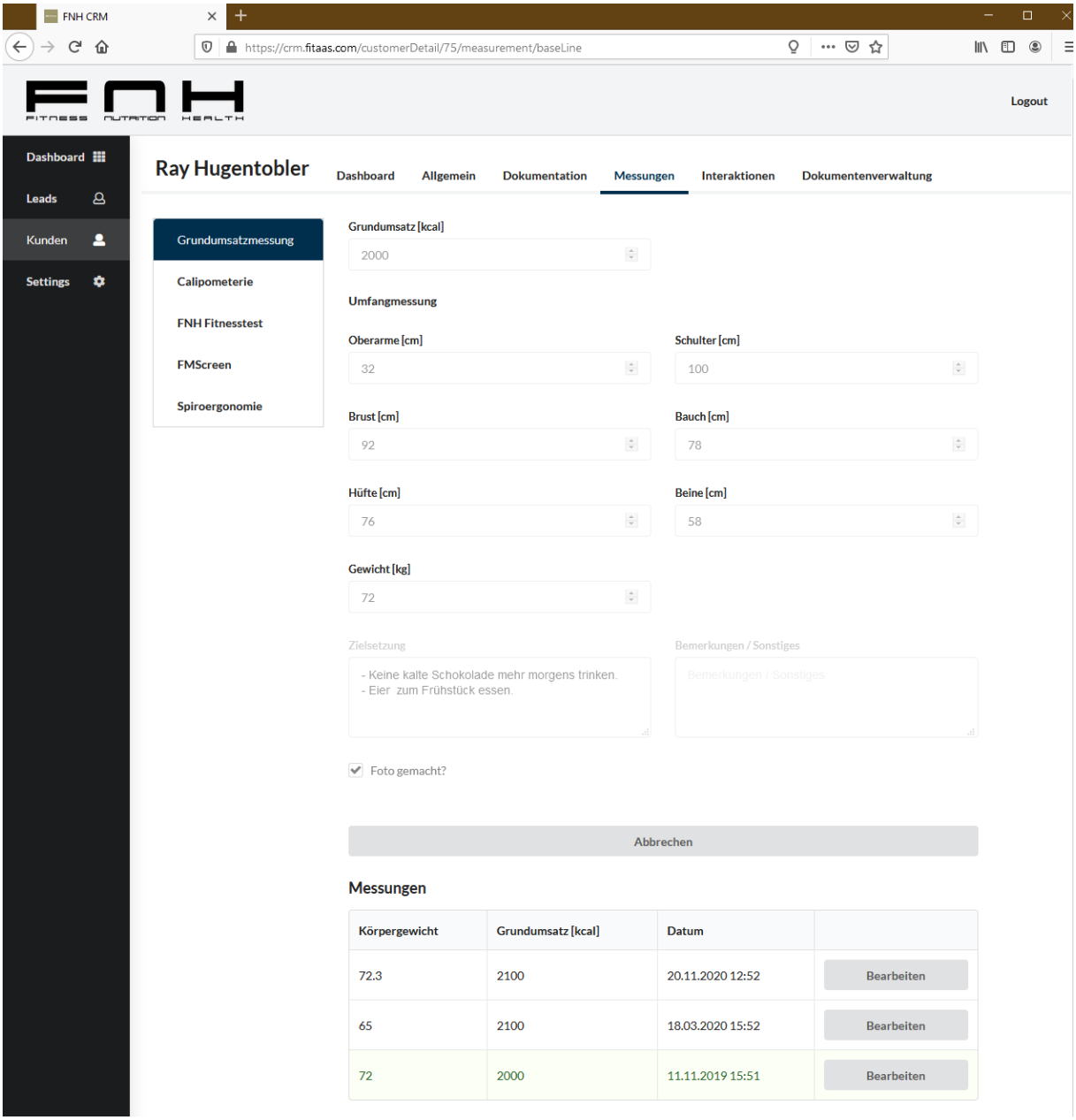

Figure 19 - Final version – Customer screen – Detail view – Measurement – Basic turnover

Comparison to the final version of the frontend

The fields in the final version have clearer labelling than in the mock-up, so that there is no confusion on what format data has to be entered in. In addition, an edit button was introduced. The entry viewed or edited is highlighted in the measurement list.

Customer screen – Detail view - Interactions

Under "Interactions", customer contacts can be recorded and logged. The channel on which the customer was contacted (WhatsApp, phone, email, etc.) must be specified. In addition, a comment that describes the interaction can be recorded. To avoid having to switch to the overview page, the required data such as mail, phone number, etc. are displayed on the left side. Every interaction is saved and displayed in a list below. The newest ones are always at the top.

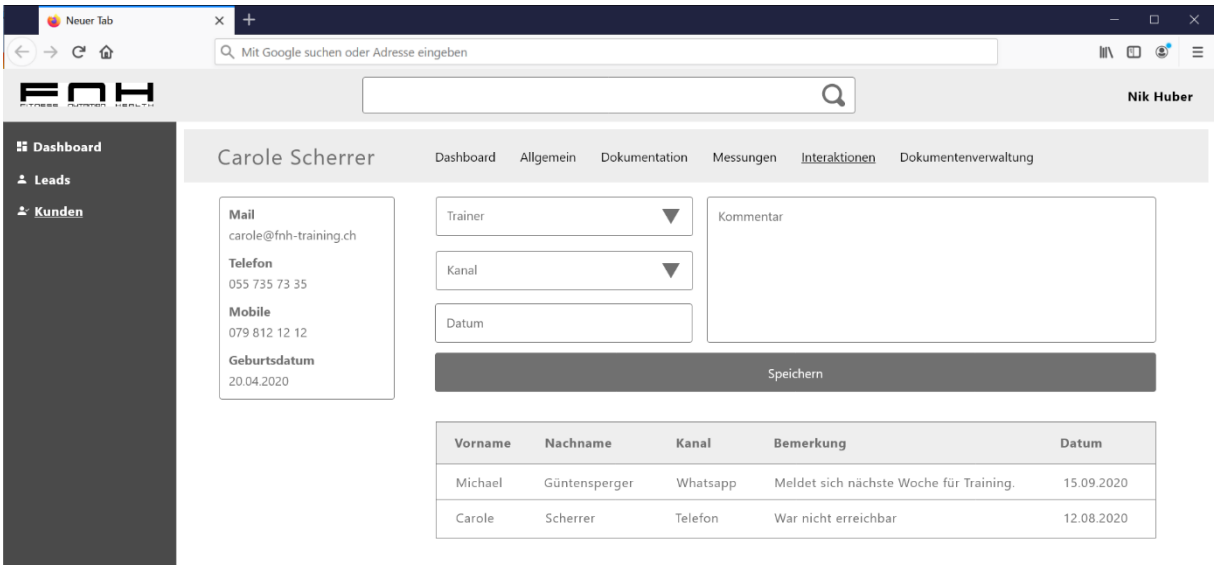

Figure 20 - Mockup – Customer screen – Detail view – Interactions

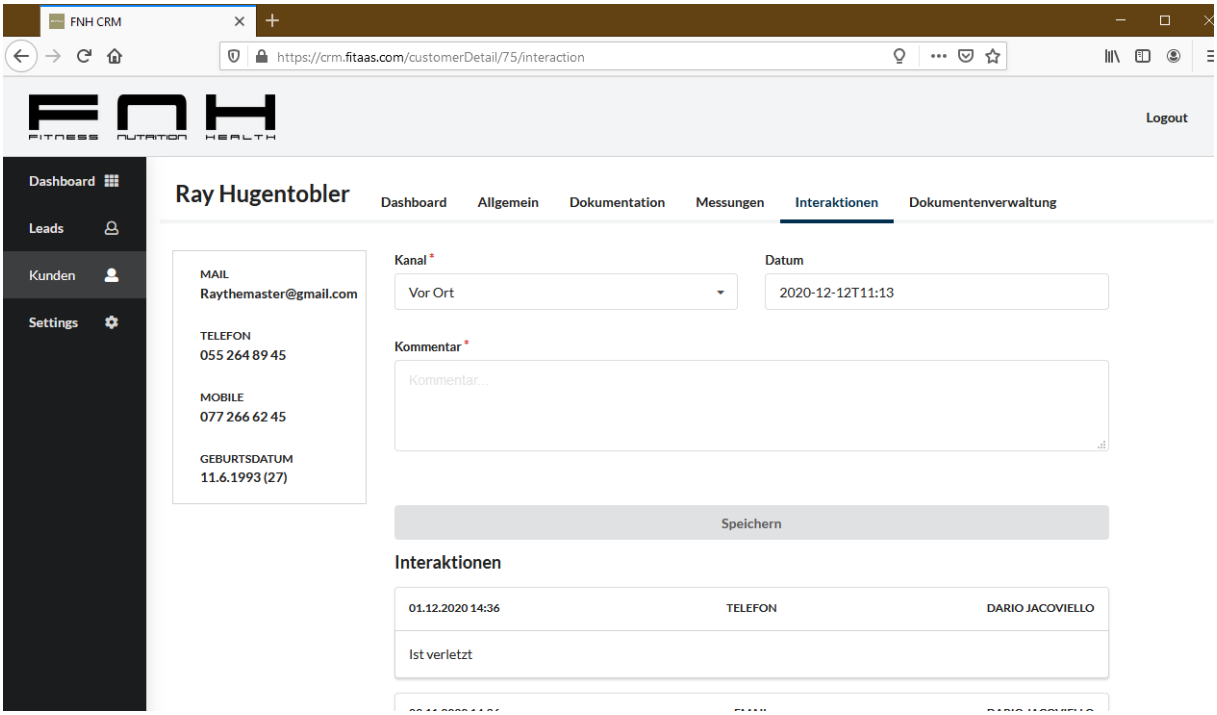

Figure 21 - Final version – Customer screen – Detail view - Interactions

<u>Comparison to the final version of the frontend</u>

As with the documentation, the trainer dropdown has been removed for the same reasons as well. In addition, the segment view was introduced instead of the table so that the entire text of the interaction is always visible.

Customer screen – Detail view – Document management
Any documents, such as contracts or notes, that can be assigned to a customer can be uploaded here as a PDF and viewed again later.

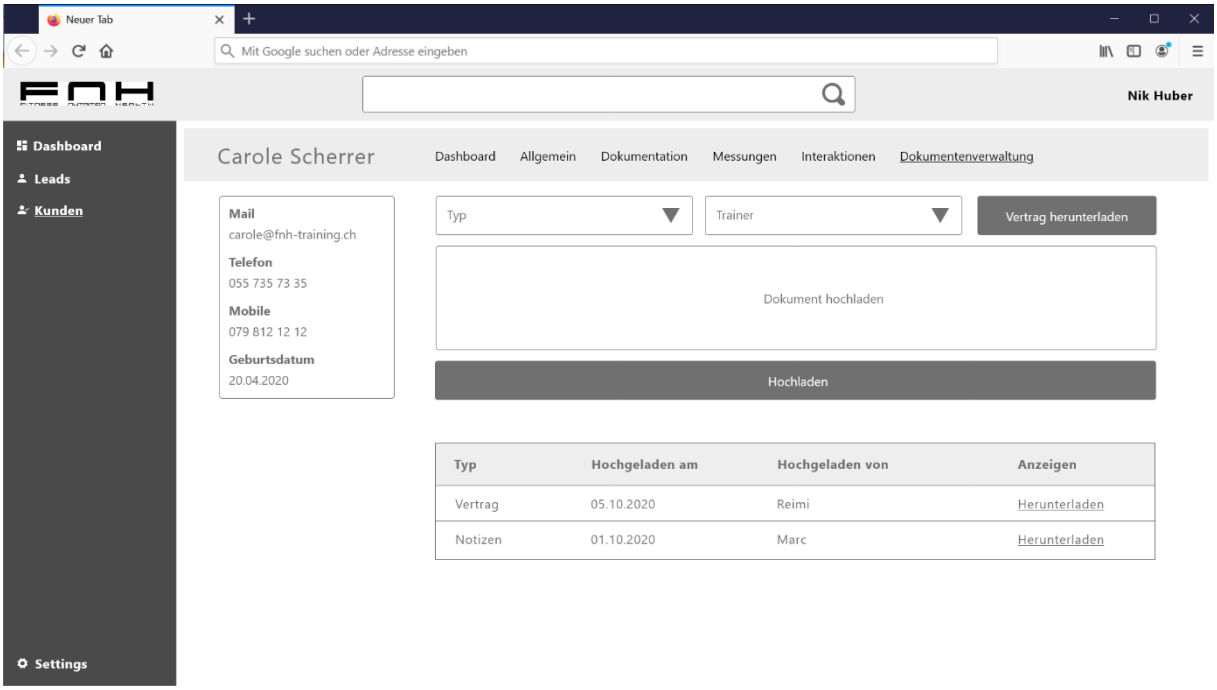

Figure 22 - Mockup – Customer screen – Detail view – Document management

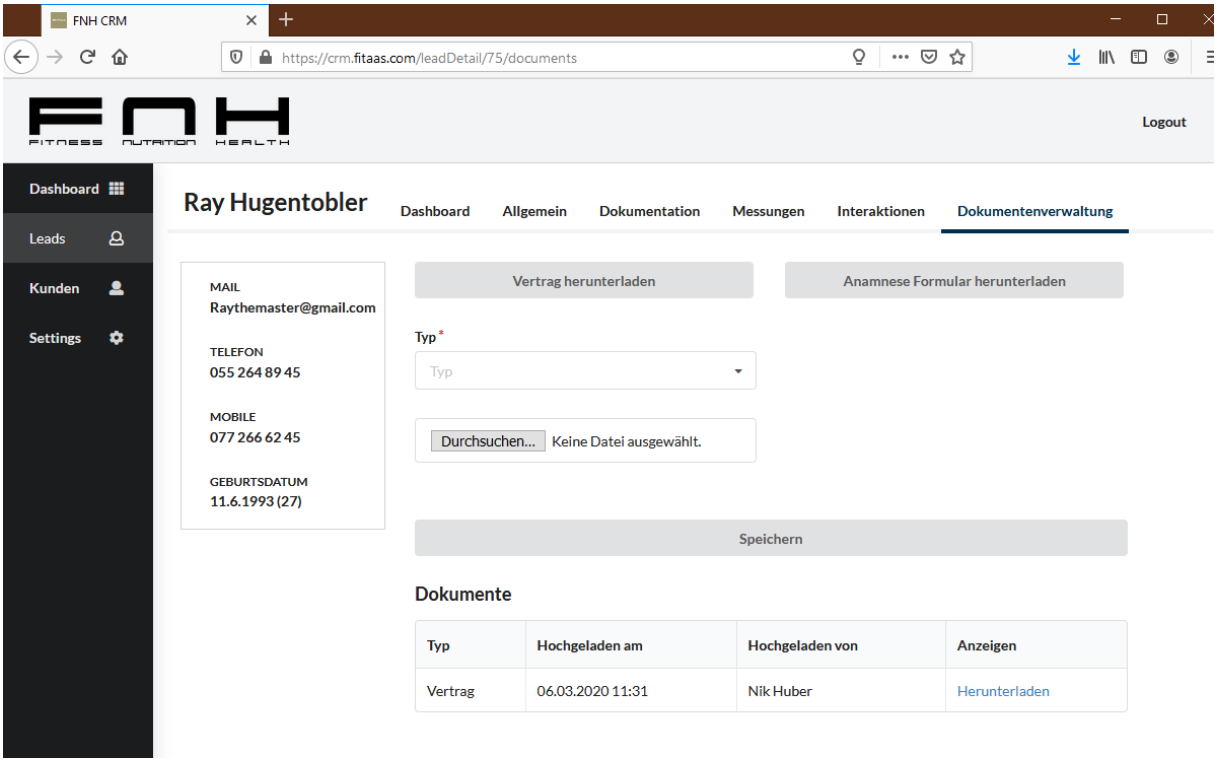

Figure 23 - Final version – Customer screen – Detail view – Document management

The contract is a special document. For analysis, additional information is needed and because the contract is being down- and uploaded as static document, these have to be entered manually. The expiration date is required at a later stage to display the contracts that expire soon (OF3), the value in CHF is used to analyse the sales over a certain period of time.

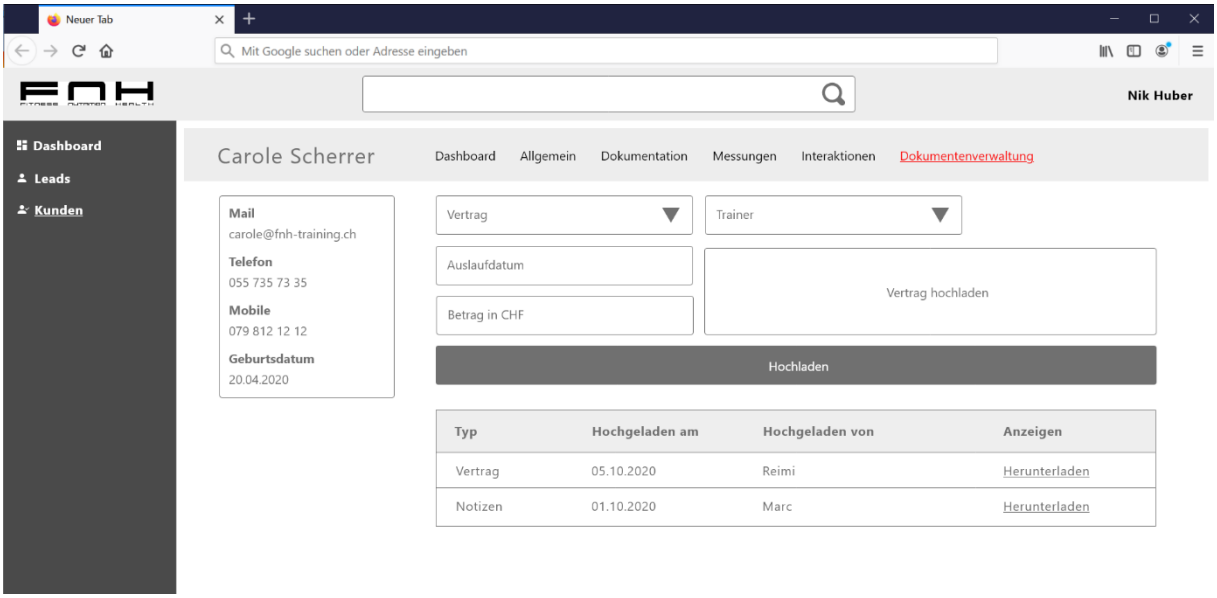

Figure 24 - Mockup - Customer screen - Detail view - Contract downloaded

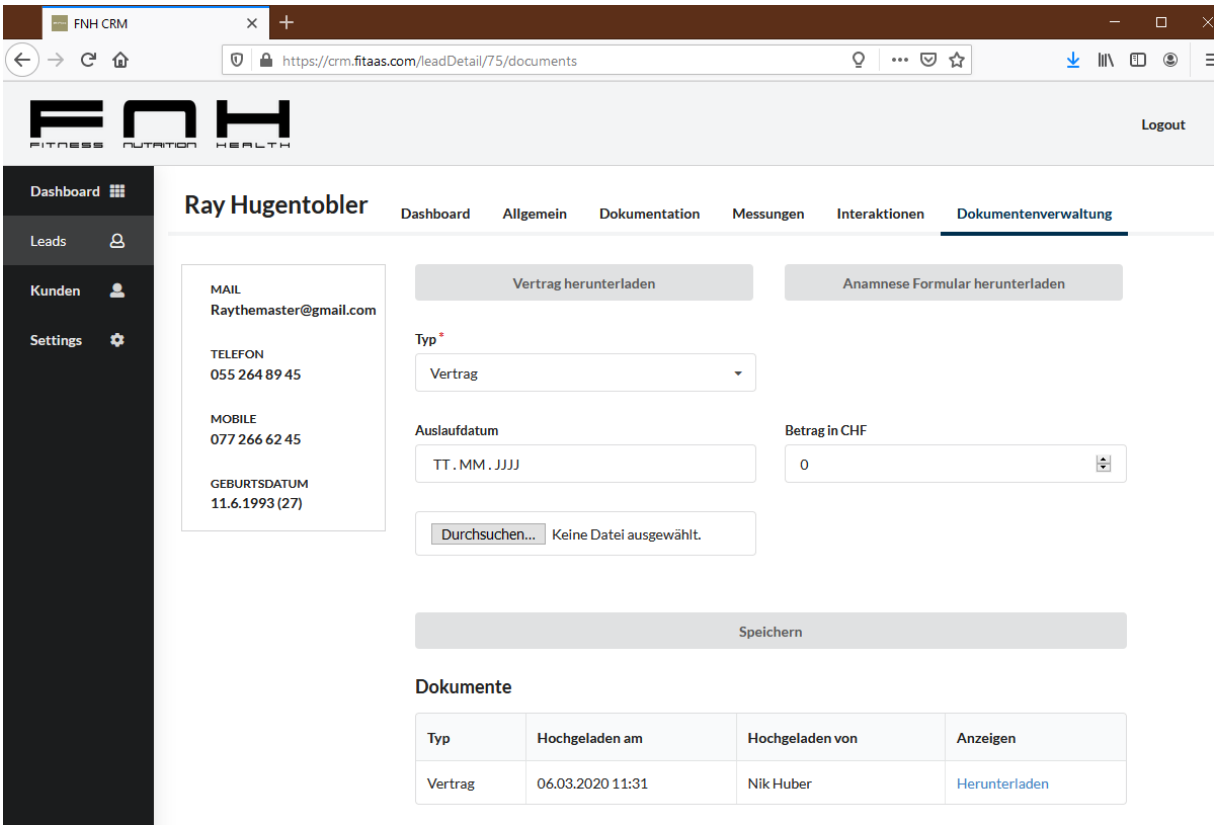

Figure 25 - Final version - Customer screen - Detail view - Contract downloaded

Settings screen – Detail view - Users

The settings are only visible to users with the role "owner". Under "Benutzer" he can create new users and edit or delete existing ones. The users' password can also be changed here. The deletion must be confirmed after pressing the button to prevent accidental deletion.

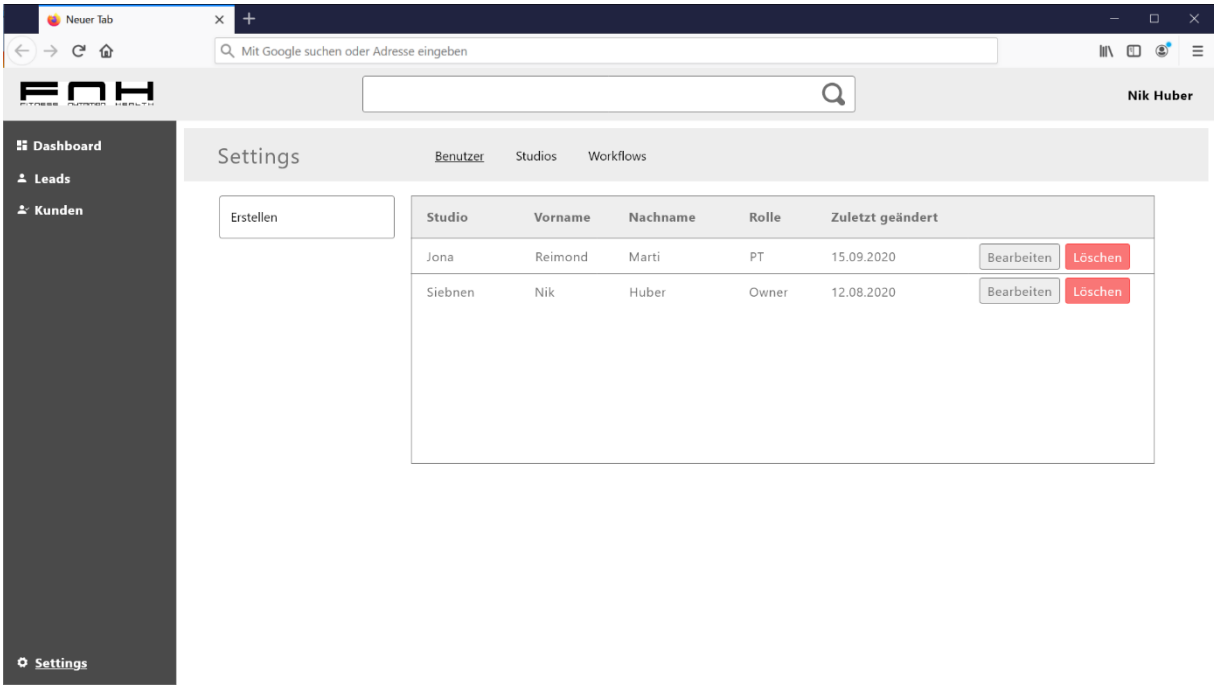

Figure 26 - Mockup – Settings screen – Detail view – Users

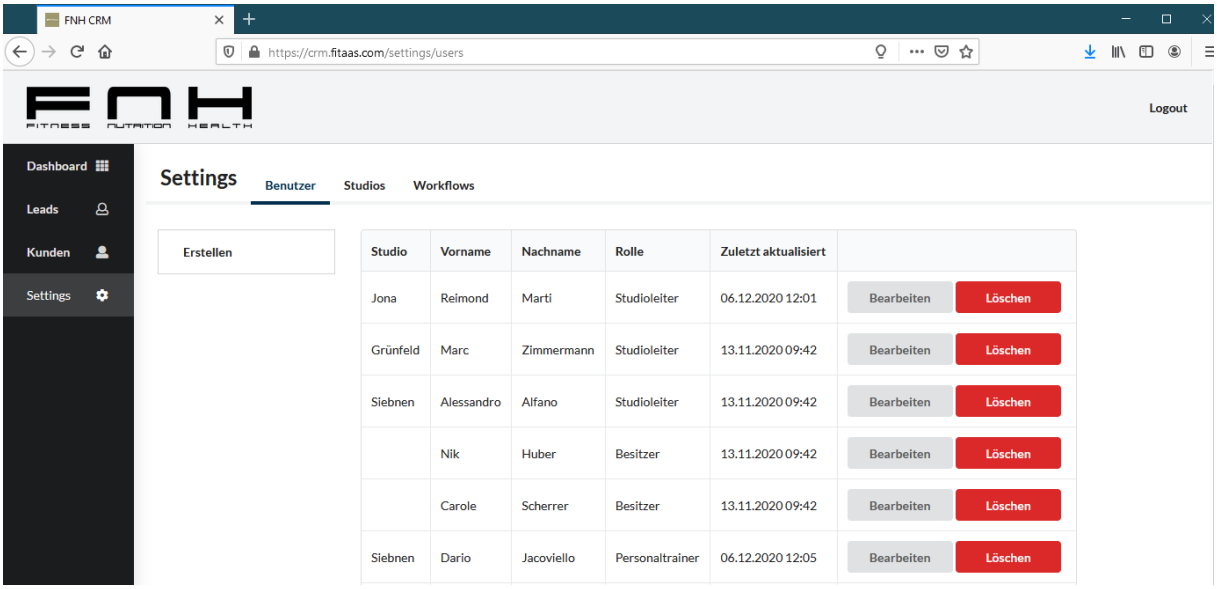

Figure 27 - Final version – Settings screen – Detail view – Users

Settings screen – Detail view - Studios

In the settings, one can also create new studios or edit existing ones. The deletion must be confirmed to prevent accidental deletion.

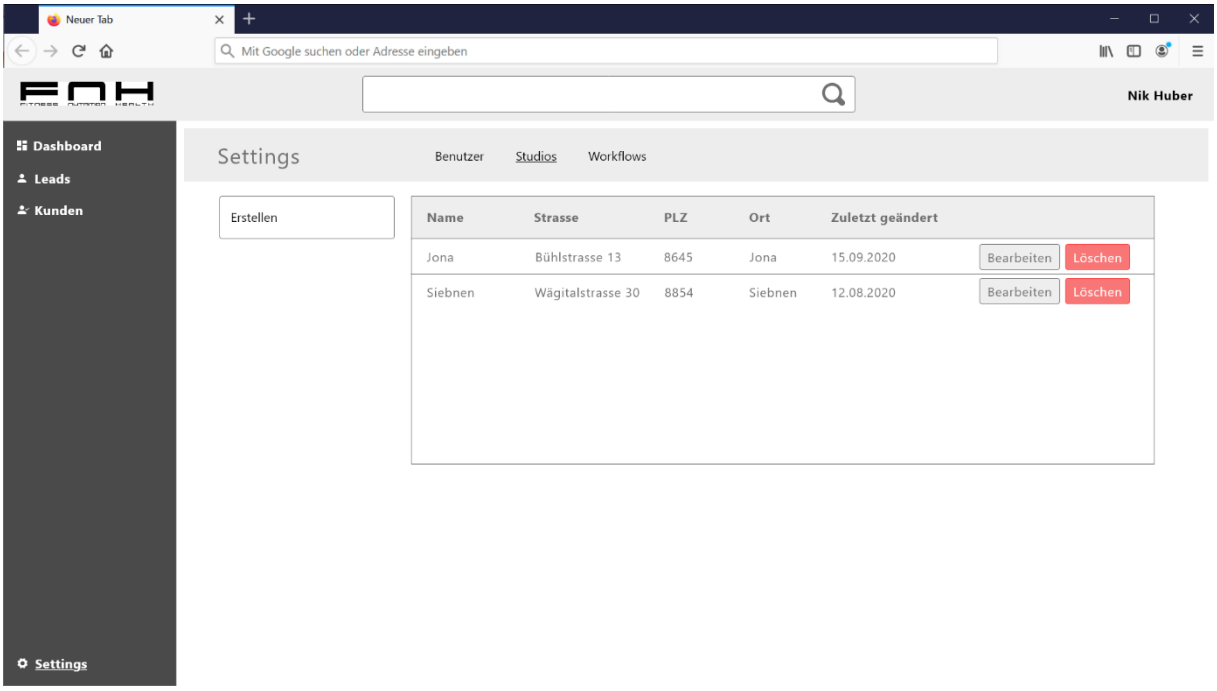

Figure 28 - Mockup – Settings screen – Detail view – Studios

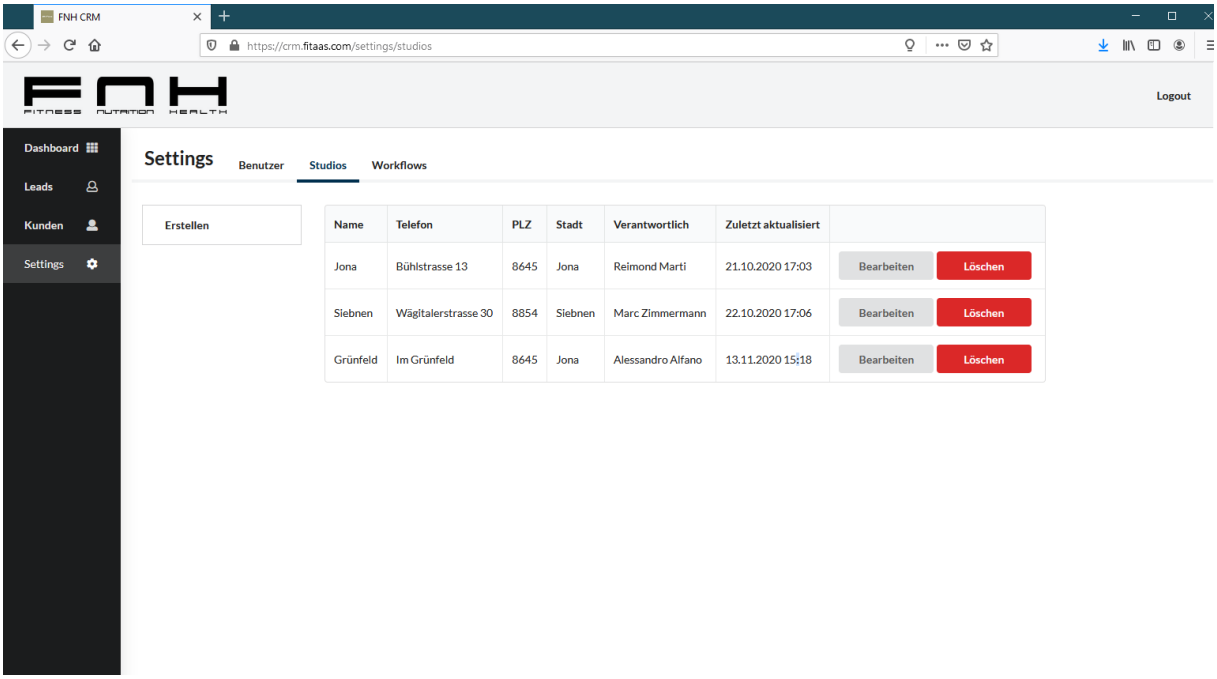

Figure 29 - Final version – Settings screen – Detail view – Studios

<u>Comparison to the final version of the frontend</u>

The studio settings have been extended with "Responsible". When a studio is created, a chief personal trainer, who is responsible for the studio, can be assigned. If the assigned trainer is still a personal trainer, he is automatically converted to chief personal trainer (OF22).

Settings screen – Detail view – Workflows

In order to be able to respond to a customer's requirements as well as possible, there are various workflows. A workflow consists of several steps, which have to be processed one after the other in a certain order. The workflow can be created, edited or archived in this submenu by the user with the role "Owner". New building blocks can be created while building a new workflow. They can then be assigned to a step in the workflow. To make this process intuitive, the building blocks can be dragged and dropped to the right position.

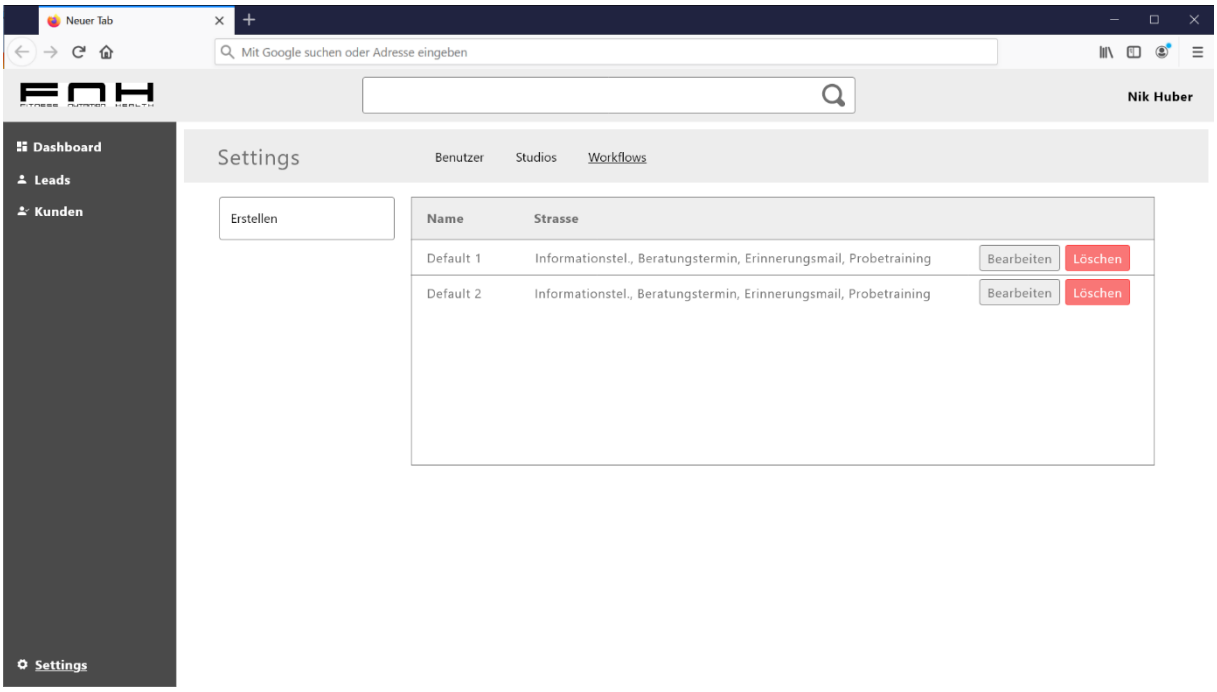

Figure 30 - Mockup – Settings screen – Detail view – Workflows

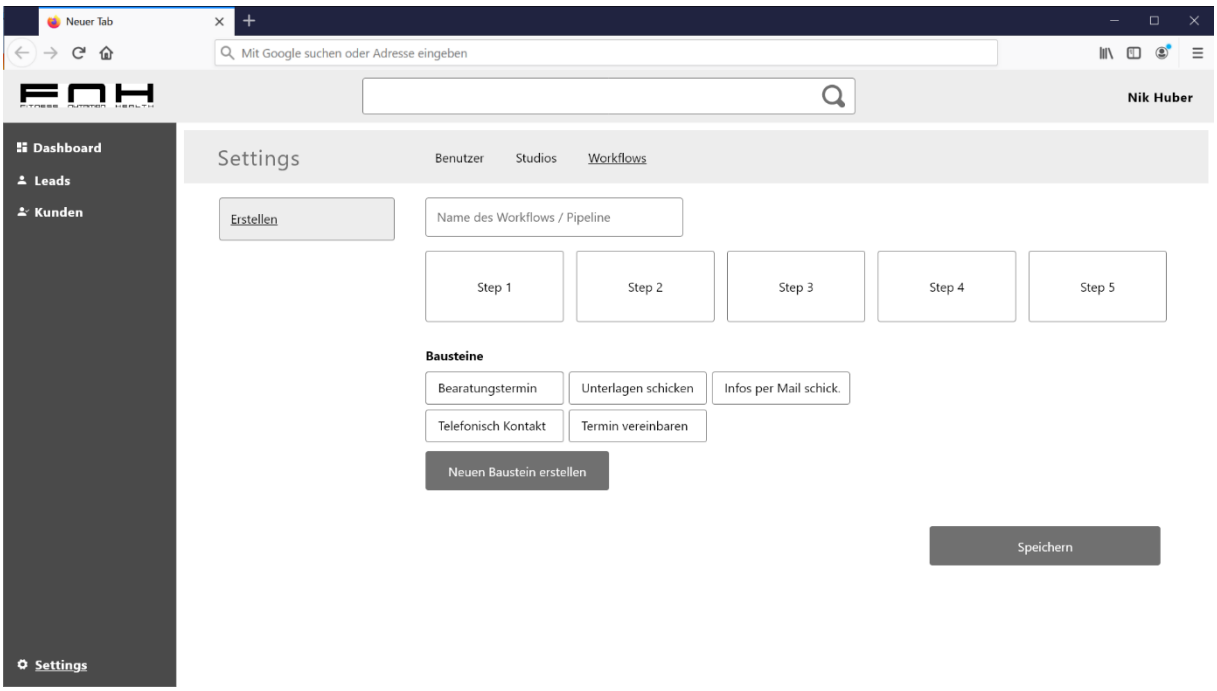

Figure 31 - Mockup – Settings screen – Detail view – Create workflow

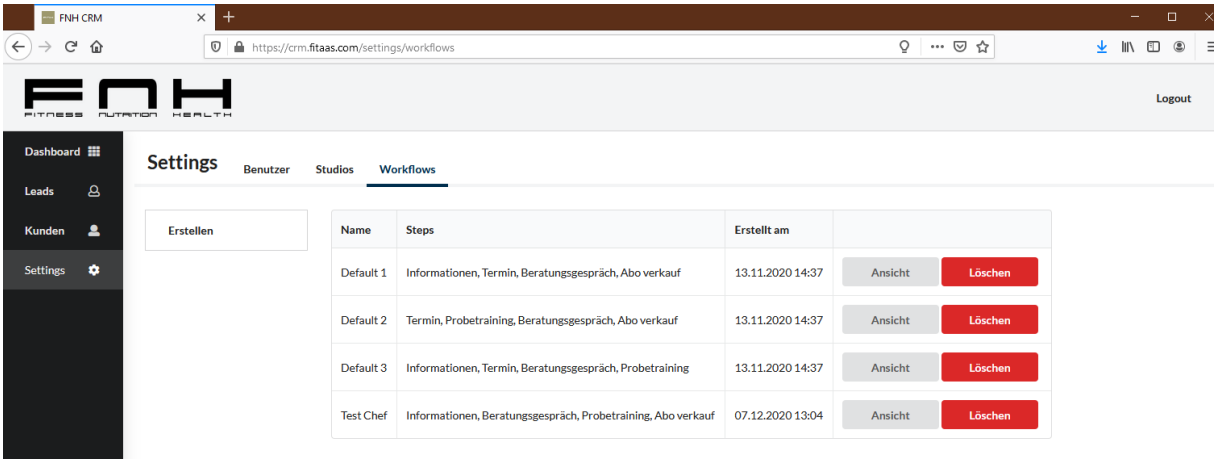

Figure 32 - Final version – Settings screen – Detail view – Workflows

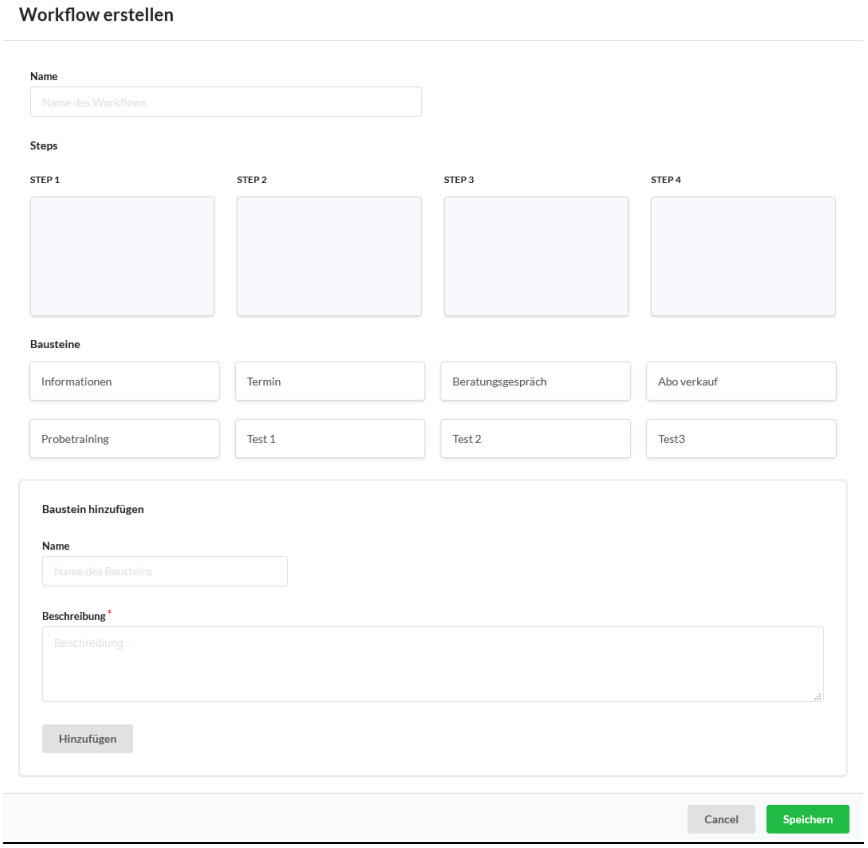

Figure 33 - Final version – Settings screen – Detail view – Create workflow

<u>Comparison to the final version of the frontend</u>

The creation of the workflow was solved with a dialogue window, a so-called "modal". This way, it is consistent with the rest of the application. In addition, the creation of the building blocks was visually separated from the creation of the workflow, so that it is clear to the user that this is not part of the workflow itself.

2.5.2 Frontend screenflow

In order to visualize the interaction of the individual screens, a screenflow was created. Like the individual mockups, this was very practical to see the structure of the screens and their dependencies during development. It allowed to create the navigation and routing of the application right from the start and served as the basic structure of the application.

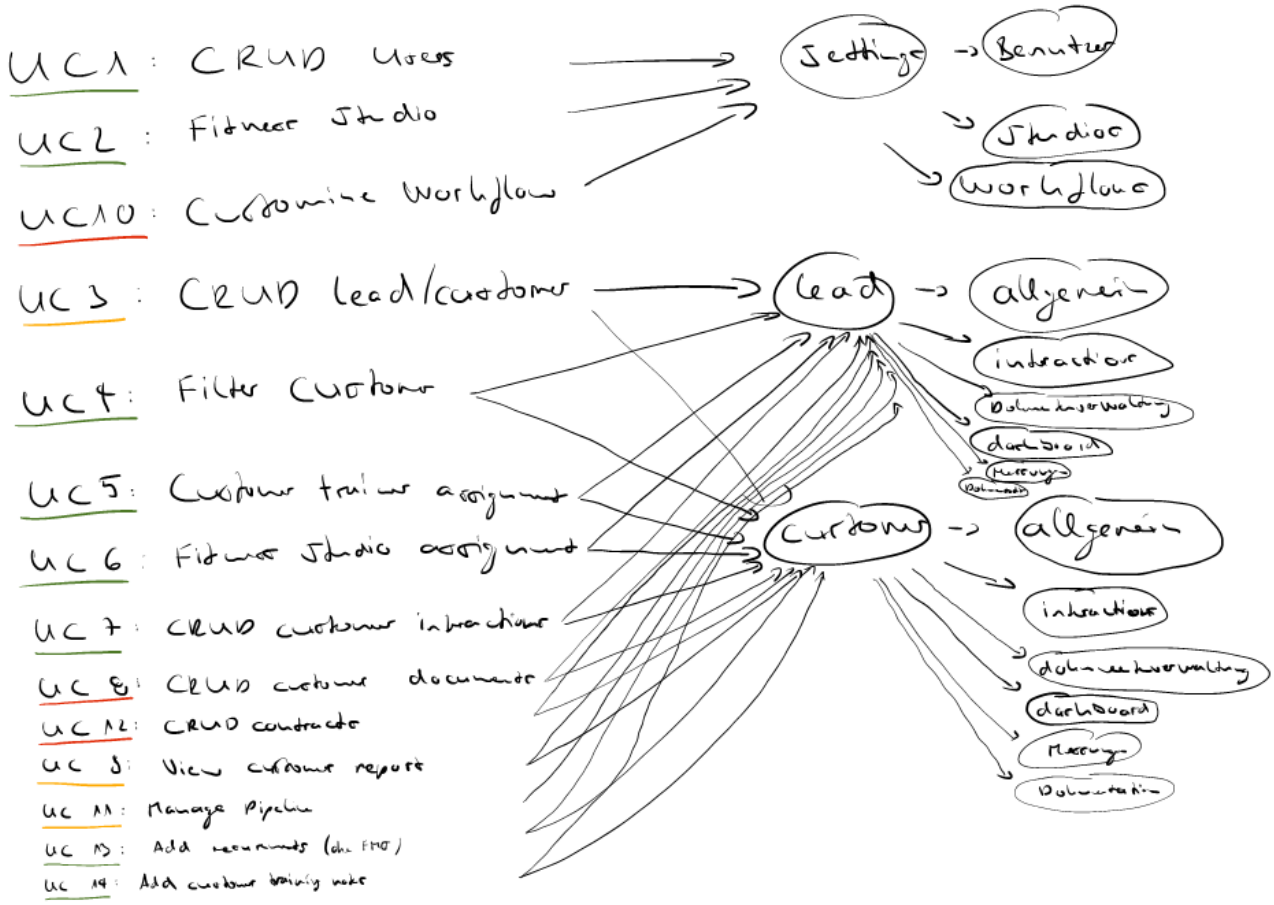

Figure 34 – Screenflow

The image on top displays a screenflow. It helped to clarify which screens belong to what menu option. On the left are all the use cases, in the middle the menu point and on the right the needed screens to display all the use cases. The colours on the use cases reflect a first assessment of the difficulty. Red is the most and green the least difficult to implement. It was used to set the scope of the project (functional requirements). Settings, lead and customer became the main (vertical) menu points, the ones on the right became the sub (horizontal) menu points.

3. Software architecture

Contains a description of every component used to build and run the software. All of it is hosted on DigitalOcean. It offers all the functionality that is needed, the deployment is simple and the cost is comparatively low.

3.1 Design decisions

JavaScript

Allows to use the same programming language on the font- and backend. This way, everyone can work on both applications without having to change the development environment. There is a big community around JavaScript, so a lot of content for problem solving can be found online (not always an advantage) [4].
The client plans to continue the project after the bachelor thesis. Because there are a lot of JavaScript developers out there, it should be possible to continue the development at a reasonable price.

Node.js

Because there are no computation intensive tasks and Node.js is very good at handling huge amounts of simultaneous connections, it is the perfect technology to work with for the backend in this project [5]. The entry hurdle is low and projects can be set up very quickly.

React

Is one of the most popular frontend frameworks for the web. Compared to its biggest competitor Angular, it has a lower entry hurdle and allows to start developing more quickly. Because it is maintained by Facebook and has a big community, it can be expected to be around for some time [6].

Redux

Redux is used for the React frontend. Without Redux the state is distributed everywhere. Redux stores a global state, that is accessible from everywhere in the frontend. Since the state change is explicit, debugging is made easier [7].

Postman

Is used to document API's. In this case it is used in the backend to easily document the interface to the frontend. It visually displays the API and helps with the testing. Postman automatically creates an API-documentation and allows the user to mock a demo API online.
First, it was planned to use Swagger for the documentation, in order to learn a new tool [8]. The setup for Swagger turned out to be more difficult than expected. Because of this the switch back to the well-known tool Postman was made [9].

MySQL DataBase

Is a relational database based on SQL. MySQL runs on virtually all platforms and is a popular choice for web-based projects that require a database for simple data transactions. MySQL is reliable and works well with business intelligence applications because they are typically very read-heavy. If the app will have a lot of traffic in the future, MySQL will be easily scalable [10]. The choice to use a relational database over a NoSQL one, seemed natural. For a CRM, the data integrity and relationships are very important. Not only for the application to work correctly, but also to analyse data for business purposes with tools like Power BI. The flexibility and scalability of a NoSQL database is not necessary to a high enough degree to justify its choice. While NoSQL databases are generally faster, a relational database is still fast enough for this application [11].

Sequelize OR Mapper

Sequelize maps the database to the Node.js application. It offers a lot of functionality and a promise-based access to the database. Using an OR Mapper guarantees the replaceability of the database. Furthermore, Sequelize allows working with associations, which represent relations between database entities and simplifies interactions significantly compared to raw SQL data access [12].

Semantic UI React

Semantic UI React is used to simplify the styling of the application enormously. With the help of semantic UI's theming, elements such as buttons can be defined globally in one place and restyled very easily at any time [13]. There are many different UI component libraries, a detailed evaluation of why Semantic UI React was chosen can be found in appendix K.

3.2 System overview

The diagram provides an overview of the system with a detailed description of all the individual components. They are all hosted on DigitalOcean.
The developer documentation offers a more detailed description on how to setup the environment locally and on DigitalOcean and can be found in appendix **Fehler! Verweisquelle konnte nicht gefunden werden.**.

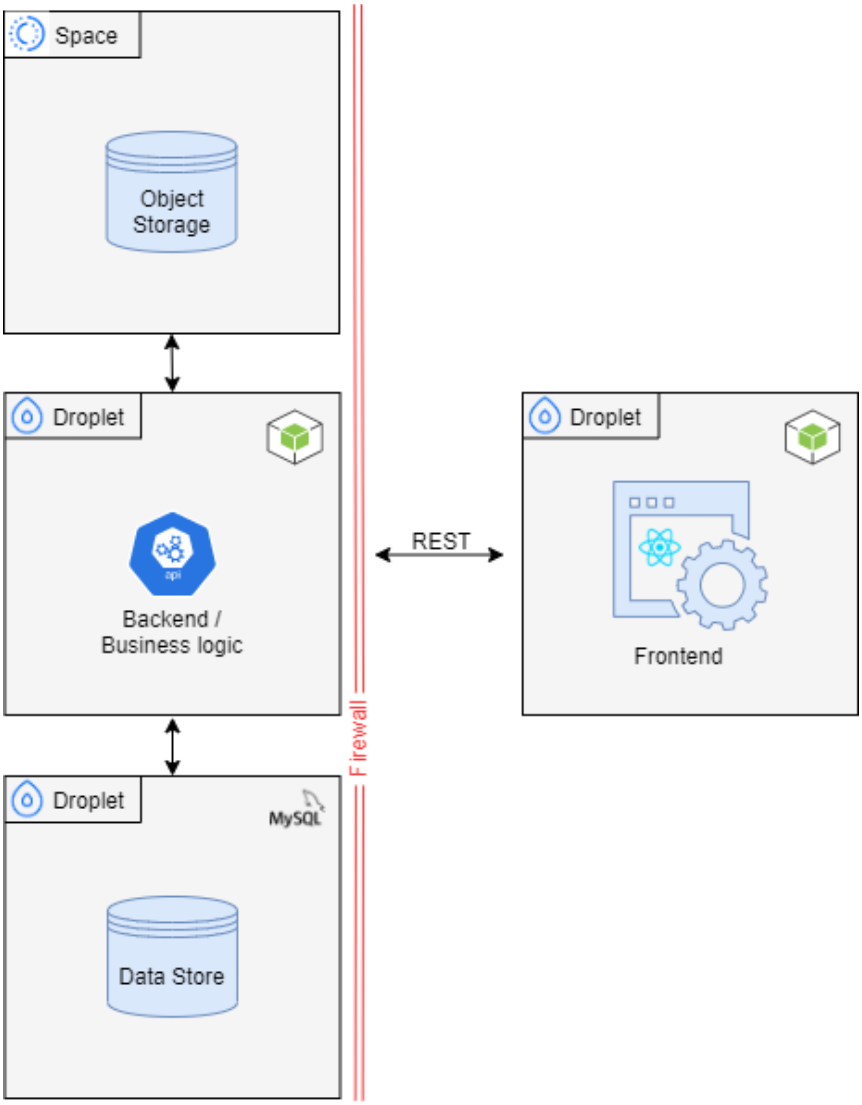

Figure 35 - System overview

Frontend

The web application / user interface for the users (owner, personal trainer) is implemented in React. It provides CRUD operations for the users to work with the data. There is an account per user so that it is simple to track the activities and manage the permissions. A role based authentication is implemented to limit the tasks a user is allowed to perform.

Data Store

DigitalOcean offers a managed MySQL database that is used to run the database. The advantage is that it can be scaled very easily. The MySQL database could also be run in a droplet. To do so, it would have to be setup and managed by the developers. To save some time on the setup and in return have more time for the development, it was decided to work with the managed MySQL database service. For local development, a simple MySQL docker container can be run. The database cannot be accessed directly.

Backend

The backend is realized with Node.js and it is the interface between the frontend and the database. It offers an HTTP REST-API which makes it possible to add further functionality or connect applications later on.

Object Storage

The object storage is a special service provided by DigitalOcean, that allows the simple storage of unstructured data, such as pdf's or images. The object store is needed to store contracts and notes. To make sure, that object store access parameters are not exposed to the client, all interactions pass through the backend.

3.3 Architectural goals & restrictions

3.3.1 Goals

Expandability

Because a REST-API is being used, it will be easy to add new features in the future. The API is well documented, so it is easy to add new functionality to the backend or just use the API for new projects, like e.g. an online booking platform.
The front-, backend, object storage and database are all deployed on DigitalOcean. Because of the competitive pricing, the easy setup and good documentation, future extensions or new versions can be deployed without problems. Another advantage of the cloud hosting is that if the traffic should spike, the only thing to do is an upgrade of the selected Droplet size. State of the art tools are used for the development and deployment. Those tools will likely be around for a long time and there are a lot of skilled developers in this area.

Usability

An optimal user interface and usability is of great importance. Through meaningful naming of the individual functions, suitable icons and a very simple structure, the application is made more comprehensible for the end user. On every screen, it should be clear what tasks can be performed. By using different, easily distinguishable colours, it is ensured that the user can easily differentiate between the individual elements and recognise clickable areas.

Privacy and security

Since a customer's sensitive personal data is being stored, the general data protection regulations must be complied with. This includes the storage of passwords only in hashed form and that no user has access to functionality that his role is not allowed to.

3.3.2 Restrictions

Application uptime is dependent on the cloud service provider, on which it is hosted. DigitalOcean guarantees an uptime of 99.99% in its SLA's for all Droplets.
What a downtown of 99.99% means:

- Daily: 8s
- Weekly: 1m 0s
- Monthly: 4m 22s
- Yearly: 52m 35s

Frontend

Because all the data is being stored in the backend, a stable internet connection is always needed.

Backend

If there are big spikes on the needed performance, the droplets on DigitalOcean have to be upgraded manually.

3.4 Tools

These are the main tools, that were used for development:

WebStorm

WebStorm is the web IDE by JetBrains. It offers excellent support for JavaScript like intelligent code completion. WebStorm is well supported and documented [14].

GitLab

GitLab is the source code management and CI/CD tool of choice. It is a complete DevOps platform, that offers multiple services [15].

Postman

Is used to manage and test API-requests. It is easy to use and offers a simple dashboard with an overview of all the registered API calls [9].

Docker

Docker runs virtualized software packages called containers. In this project Docker is used to run the database during development. Docker allows to start the container from a predefined state, which enables reproducible tests using a prepared database image [16].

DataGrip

Like WebStorm, this is a JetBrains product. It is a database-IDE that is designed for relational databases and allows to easily check database interaction of the application during development, by showing the current state [17].

Jira

Jira is a team collaboration tool by Atlassian used for project management. The central features are time and issue tracking. It is a very intuitive and flexible tool, that supports agile development [18].

3.5 Logical architecture

The backend is divided into 3 layers. The API layer routes all of the REST calls to the correct controller. The business layer is divided into different modules, with distinct responsibilities. Every module contains a controller for every responsibility it has. Every controller accesses the database directly, using an OR Mapper.

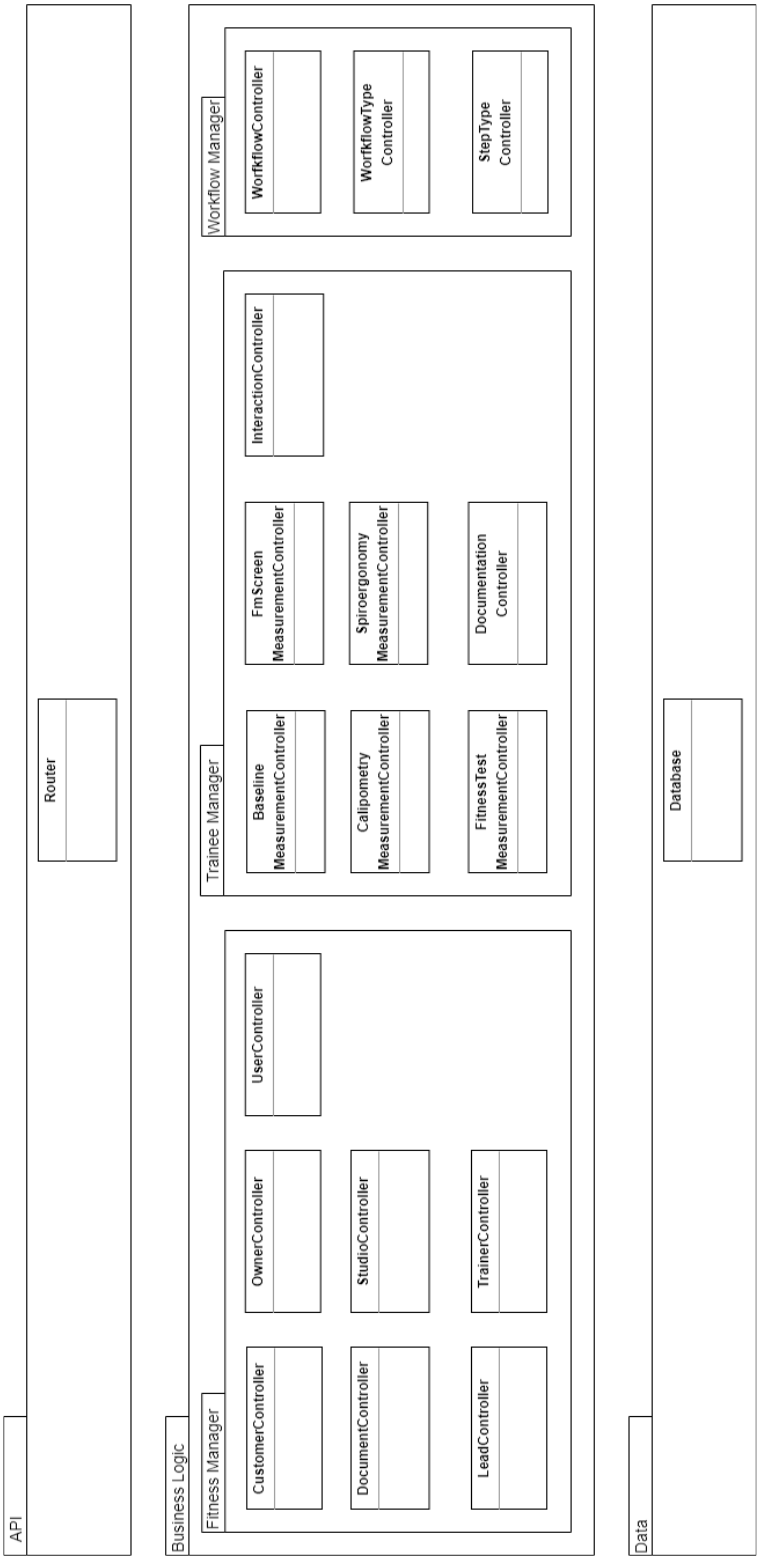

Figure 36 - Logical architecture

3.6 Important processes

In the following subchapters the designs of a selected number of important processes and the architectural motivation behind them is described.

3.6.1 Role based access

Every user of the system can have multiple roles. The role determines what can be accessed. In the backend the routes are protected and can only be accessed by predefined roles. To ensure that a user cannot pretend to have another role, the user id is hashed into the Access Token when logging in. When data is accessed using that Bearer Token, the system knows, who it is and what roles that user has. In the frontend some functionalities are only available to specific roles. In the database this is achieved with multiple table inheritance. The parent is the User table and every role is a child. General information and associations are stored in the parent, while the role specific properties and associations are saved in the child. This means that the role objects are not only identifiers, but also carry information about the user, that is specific to that role. This modular setup makes role changes, like transforming a lead into a customer very easy.

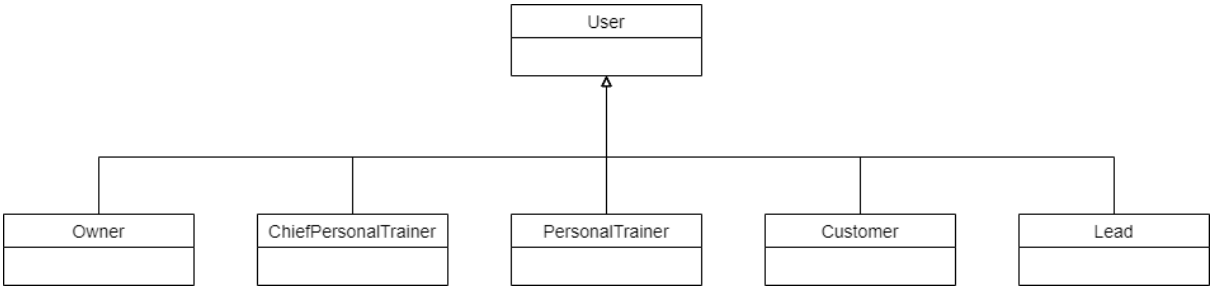

Figure 37 - Role based access

Because associations are applied to the role when possible, a user needs to have the correct role. E.g: If the studio owner wants to do personal trainer work, like recording an interaction with a customer, he needs to have the PersonalTrainer role. When the interaction is saved to the database it is associated to the PersonalTrainer object and not to the User directly.
The roles Lead and Customer can only access the login route. Currently, they are only used as data objects to store information and relations. Because future expansions, such as appointment booking, would allow customers and leads to interact with the system, this architecture makes sense. Customers and Leads already have a login and the corresponding routes can be opened to them.

3.6.2 Workflow

A workflow is the work procedure, which is applied to a lead in order to transform him into a customer. Every workflow currently consists of up to for steps, but expansion is possible in the future. The owner can create new steps and workflows, which are then saved as new templates. When a chief personal trainer applies a workflow to a lead, a new instance of the workflow is created. It inherits the steps and their order from the template but is independent after creation. When the template is deleted or changed, the workflow instance stays unaffected. This allows to analyze later, how leads were converted to customers, even when the respective workflows have been changed or do not exist anymore.

The following figure shows what happens if a WorkflowType is deleted.

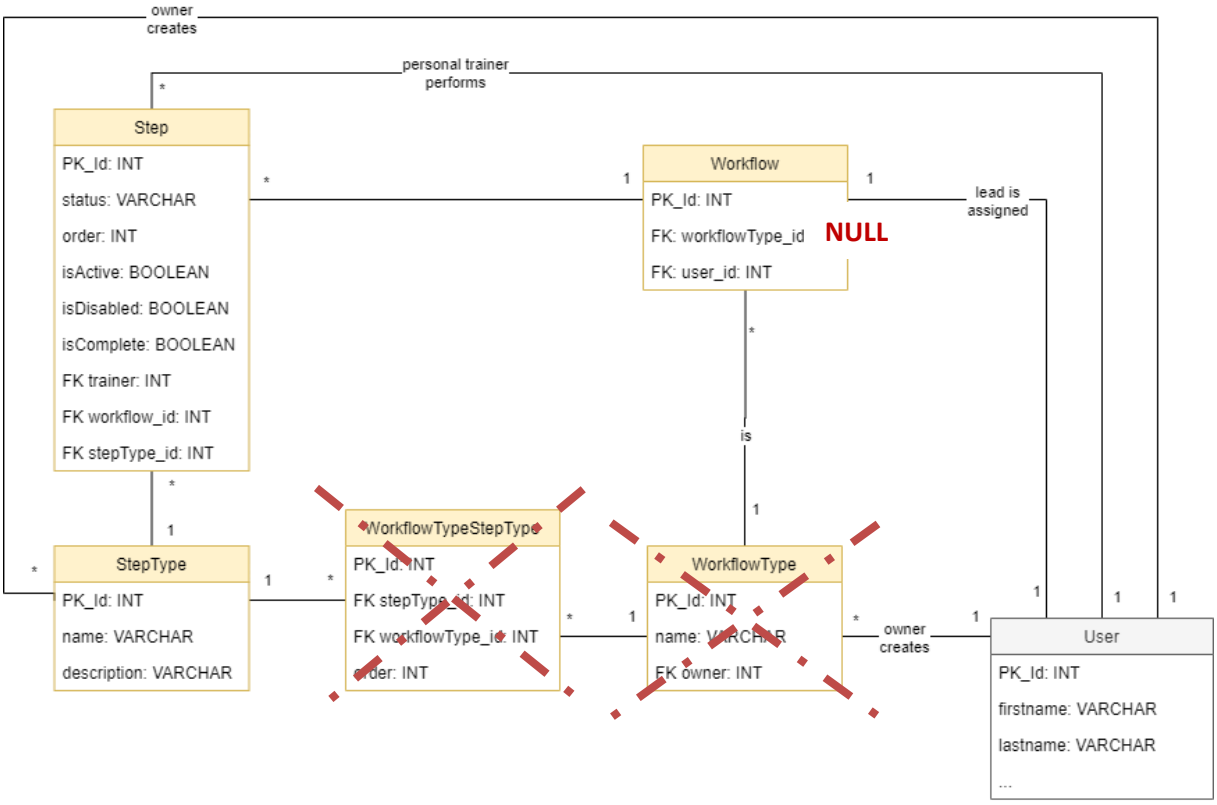

Figure 38 – WorkflowType deletion

Every step has a status which can be set by the personal trainer that works on the workflow. This status describes more precisely what is going on with the step. E.g: The step “test training” could have one of the following statuses “appointment booked”, “training held successfully” or “lead did not show up”. The steps have another state attribute, which describes, whether they are active, disabled or complete. Only one step is always active. After completion it is set to complete and the next step is changed from disabled to active. When the last step is finalized, the lead is automatically converted into a customer. The completed workflow stays associated to the user so it can be used for analysis.

3.6.3 Document management

Documents, such as contracts or manual notes are saved as pdf's on a DigitalOcean Space. DigitalOcean Spaces is and object store, which can be accessed using Amazon's AWS S3 API. The npm package "aws-sdk"[2] offers this functionality for Node.js. In order to not expose the Spaces access credentials to the client, the file up- and download needs to pass through the backend. This also allows to use the role based authentication of the backend for the document access. To keep track of the documents, their access key, as well as some other information is saved to the database.

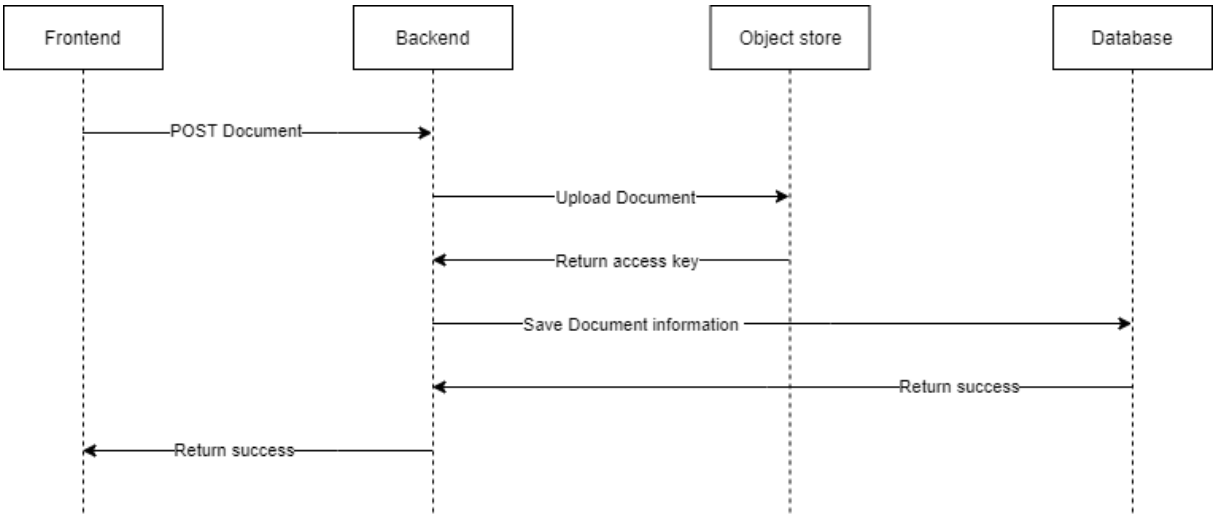

Figure 39 – Sequence diagram - Document management

3.6.4 Colouring customer by last interaction

To determine how long ago the last contact with a lead or customer was, the entries in the frontend are coloured. The time since the last interaction is calculated using the recorded interactions with a specific user. The colouring signals to the personal trainer, where action is required. Red means immediate action is required, orange that the customer or lead has to be contacted soon and white stands for ok, no action required. After contacting a customer or lead, the personal records this interaction, which is then being saved to the system.

Status	Interesse	Vorname	Nachname	Studio	Trainer	Letzer Kontakt
Keine Informationen	Gruppenstunden	Hugo	Baumann	Jona	Dario Jacoviello	08.12.2020 14:35
Keine Informationen	MTT	Stefanie	Brunner	Jona	Reimond Marti	07.12.2020 14:36
Keine Informationen	PT	Margareta	Dubois	Grünfeld	Marc Zimmermann	07.12.2020 14:36
Neu	PT	Lena	Gerber	Siebnen	Marc Zimmermann	09.11.2020 14:36
Keine Informationen	Fitness	Marcel	Hämmerli	Jona	Reimond Marti	01.12.2020 14:36
Keine Informationen	Gruppenstunden	Manuela	Meier	Siebnen	Dario Jacoviello	11.12.2020 12:19

Figure 40 - Customer list by last interaction

[2] https://www.npmjs.com/package/aws-sdk

3.6.5 Pagination and filtering

Requests for all customers or leads in the system can return a lot of entries. Loading so many entries can take a noticeable amount of time and slows down the web application frontend. To make the data more readable pagination could be implemented in the frontend, but the loading of the site would still take too long. Because of this, the pagination is done in the backend. When requesting all the leads or customers from the API, a maximum number of entries as well as the page offset can be specified. In order to make the work with these entries even easier a number of filters are provided. Applying the filter in the frontend would not work with pagination, as it would only return part of all the entries. Because of this, the filtering has also to be completed by the backend.
To achieve pagination and filtering, requests to the customer and leads route can pass filters, maximum number of entries and the page offset as query parameters. These are then parsed and passed on to the OR Mapper, which supports them.
The following figure shows how filters and pagination are added to the HTTP REST GET request as query parameters.

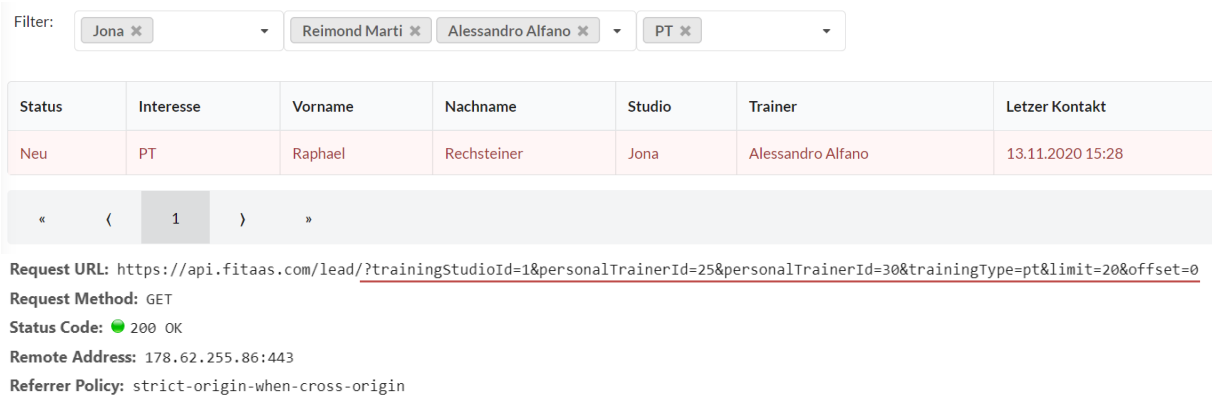

Figure 41 - Pagination and filtering example

3.7 API description

The API is the interface between the backend, the frontend and the database. All the requests to the backend will be responded to in JSON format.

Because the requests can only be performed by an authenticated user, one always has to include the access token as Bearer-Token for every request to the backend.

The following endpoints are defined in the backend. More information on the API-documentation can be found in the appendix **Fehler! Verweisquelle konnte nicht gefunden werden.**.

Endpoint	Options
/user	GET, POST
/user/{id}	GET, DELETE
/user/login	POST
/owner	GET, POST
/owner/{user_id}	GET
/customer	GET, POST
/customer/{user_id}	GET, PUT, DELETE
/customer/studioId/{user_id}	PUT
/customer/personalTrainerId/{user_id}	PUT
/lead	GET, POST
/lead/{user_id}	GET, PUT, DELETE
/lead/studioId/{user_id}	PUT
/lead/personalTrainerId/{user_id}	PUT
/trainer	GET, POST
/trainer/{user_id}	GET
/studio	GET, POST
/studio/{id}	GET, PUT, DELETE
/interaction	POST
/interaction/byuserid{user_id}	GET
/interaction/{id}	DELETE
/documentation	POST
/documentation/{id}	GET, PUT, DELETE
/documentation/byuserid{user_id}	GET
/document	POST
/document/{id}	GET
/document/byuserid{user_id}	GET
/document/contractTemplate	GET
/document/anamneseFormular	GET
/workflow/byuserid{user_id}	GET, POST
/workflow/{id}	PUT

/workflow/stepType	GET, POST
/workflow/workflowType	GET, POST
/workflow/workflowType/{id}	GET, DELETE
/measurement/BaseLineMeasurement	POST
/measurement/BaseLineMeasurement/{id}	GET, PUT, DELETE
/measurement/BaseLineMeasurement/byuserid{user_id}	GET
/measurement/FitnessTestMeasurement	POST
/measurement/FitnessTestMeasurement/{id}	GET, PUT, DELETE
/measurement/FitnessTestMeasurement/byuserid{user_id}	GET
/measurement/CalipometryMeasurement	POST
/measurement/CalipometryMeasurement/{id}	GET, PUT, DELETE
/measurement/CalipometryMeasurement/byuserid{user_id}	GET
/measurement/FmScreenMeasurement	POST
/measurement/FmScreenMeasurement/{id}	GET, PUT, DELETE
/measurement/FmScreenMeasurement/byuserid{user_id}	GET
/measurement/SpiroergonomyMeasurement	POST
/measurement/SpiroergonomyMeasurement/{id}	GET, PUT, DELETE
/measurement/SpiroergonomyMeasurement/byuserid{user_id}	GET

Table 5 - API routes

3.8 Deployment

A web browser (Edge, Chrome or Safari) is needed to run the frontend. The application is mostly used on Surface Go 2 Tablets with a screen size of 10.5 inch. Of course, it can also be run on other devices such as Microsoft personal computers with OS Version 10 and on Mac personal computers with OS Version 10.15 “Catalina”. Detailed information can be found in chapter 2.3 .

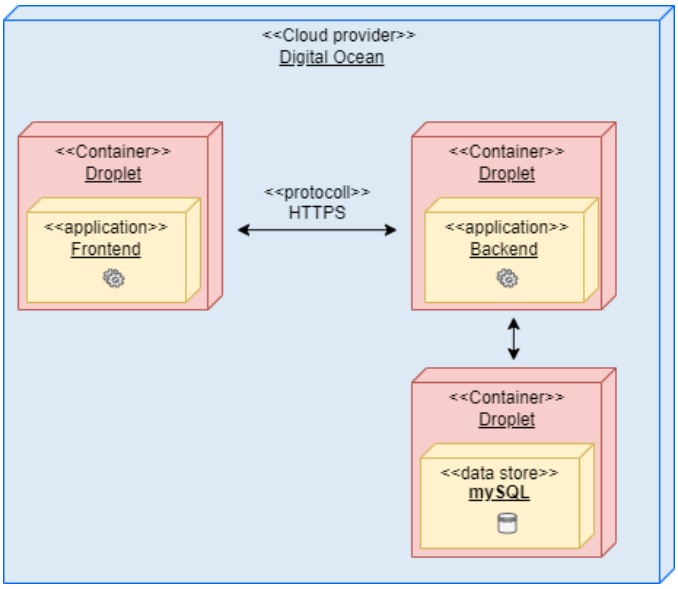

Figure 42 - Deployment diagram

All the components run on droplets which can be accessed with SSH. Because of the automated deployment, this is only rarely needed. The database is behind an extra firewall and only reachable with its internal IP-address.
During the development, the Droplet had to be upgraded due to the high utilization of the RAM when building the application. Resizing was straight forward and only took a couple of minutes.

3.8.1 Automated deployment

To avoid deploying the applications manually every time changes to the code are made, deployment pipelines are set up that are executed whenever a commit to the production branch is published. First, the application is tested and only if the tests succeed the application deployed. This helps to focus on the programming and not the publishing itself. Whenever a pipeline fails, the user that made the commit is informed by email.

The pipelines run on GitLab CI/CD. The front- and backend are deployed to their respective droplets on DigitalOcean. After the code on the Droplets is updated, the processes are automatically restarted, and the services are available again almost instantly.

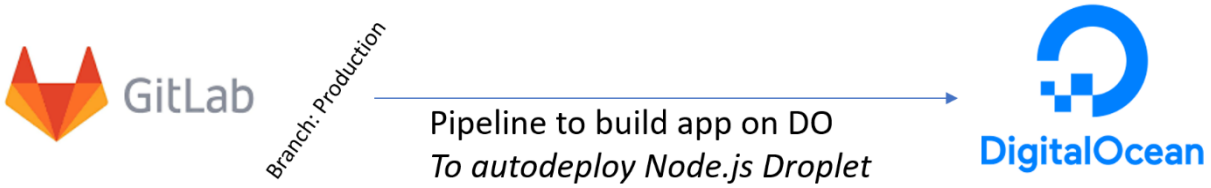

Figure 43 - Automated deployment

3.9 Data storage

Based on the domain model, a database model was created, which is described by the following diagram.

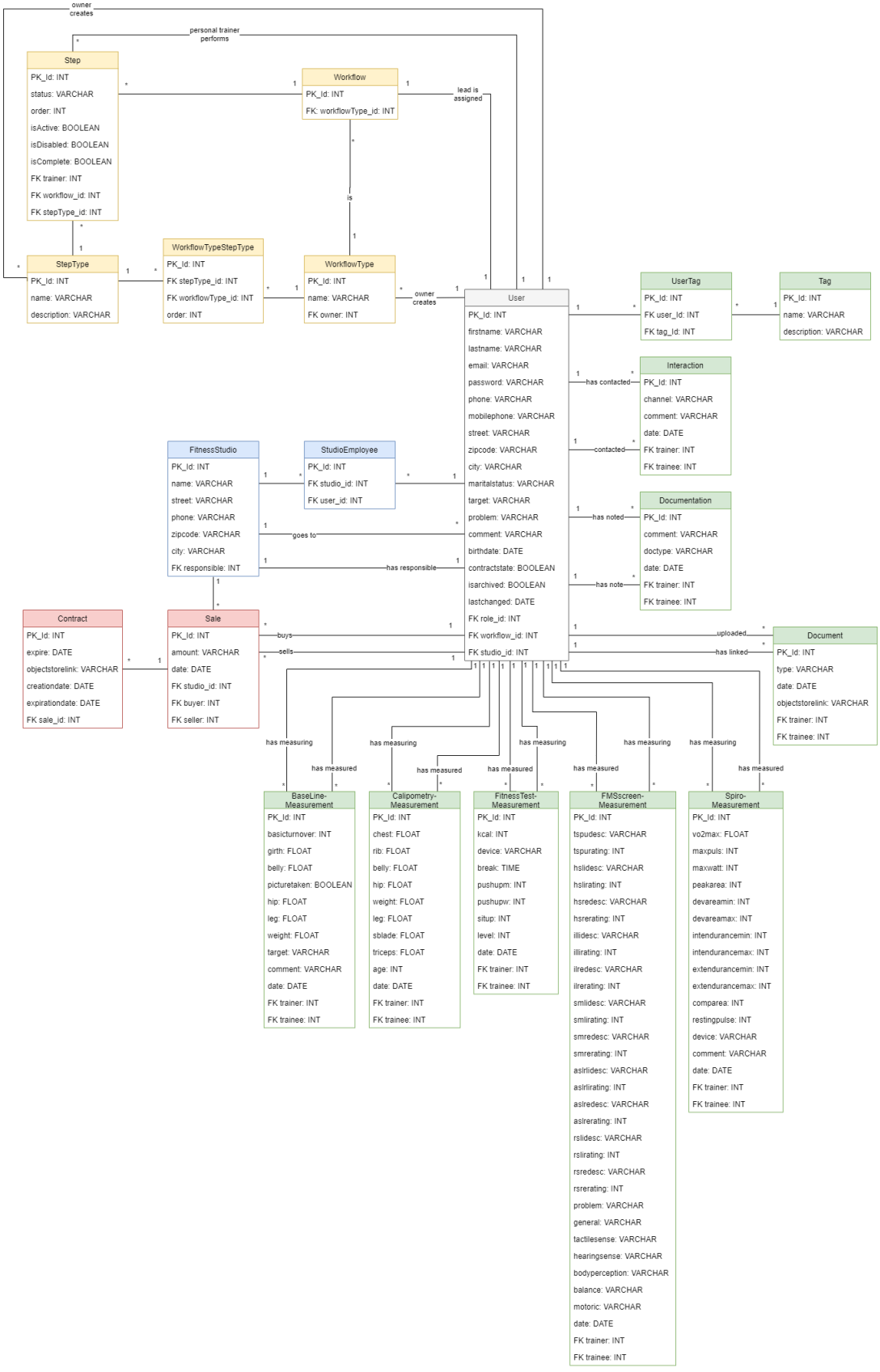

Figure 44 - Database model

It was used to build the database and to develop the API of the backend. Because the database is created using an OR-Mapper, intermediary tables as well as foreign keys are generated automatically using associations and the resulting database looks slightly different.
During the project, several changes to the data structure were made (e.g. new fields on the measurements). The final version can be found in appendix H.

Light grey	User itself. Owners, chief personal trainers, personal trainers, leads and customers are all users. The different roles are implemented with multiple table inheritance from the user basis class. To simplify this graphic, the subclasses are combined in the User. Detailed explanation can be found in chapter 3.6.1 .
Green	All the information that is shown in a lead's / customer's detail view.
Yellow	All components that belong to the workflow. That is used to track the steps a lead takes before becoming a customer.
Blue	Studio information and which employee works in which studio.
Red	Sales including the contract. The implementation was not part of the functional scope and therefore not implemented.

In order to describe fields that relate to both the lead and the customer, we have agreed to name these trainees, so that it is clear that it does not matter whether it's a lead or a customer. For example, the training studio id for User is called trainee id.

3.10 Sizes and performance

DigitalOcean's droplets are very flexible and can be configured to one's specific needs. The marketplace provides one click configurations for popular containers. This also makes it easier for the client to maintain the setup after the project. The advantage of deploying in the cloud is that if there should ever be the need for more size or performance, it is "click to scale". If needed, the containers can be hosted in different counties. For this project, the containers are hosted in a datacentre in the Netherlands, which has very low latency for users in Switzerland. The way the API is designed, only a minimal number of requests are needed to display data on the frontend.

To scale the deployment, two load balancers would be added to decouple the front- from the backend. Multiple front- and backends would allow dividing the work. The MySQL database would be scaled with more memory and CPU power.

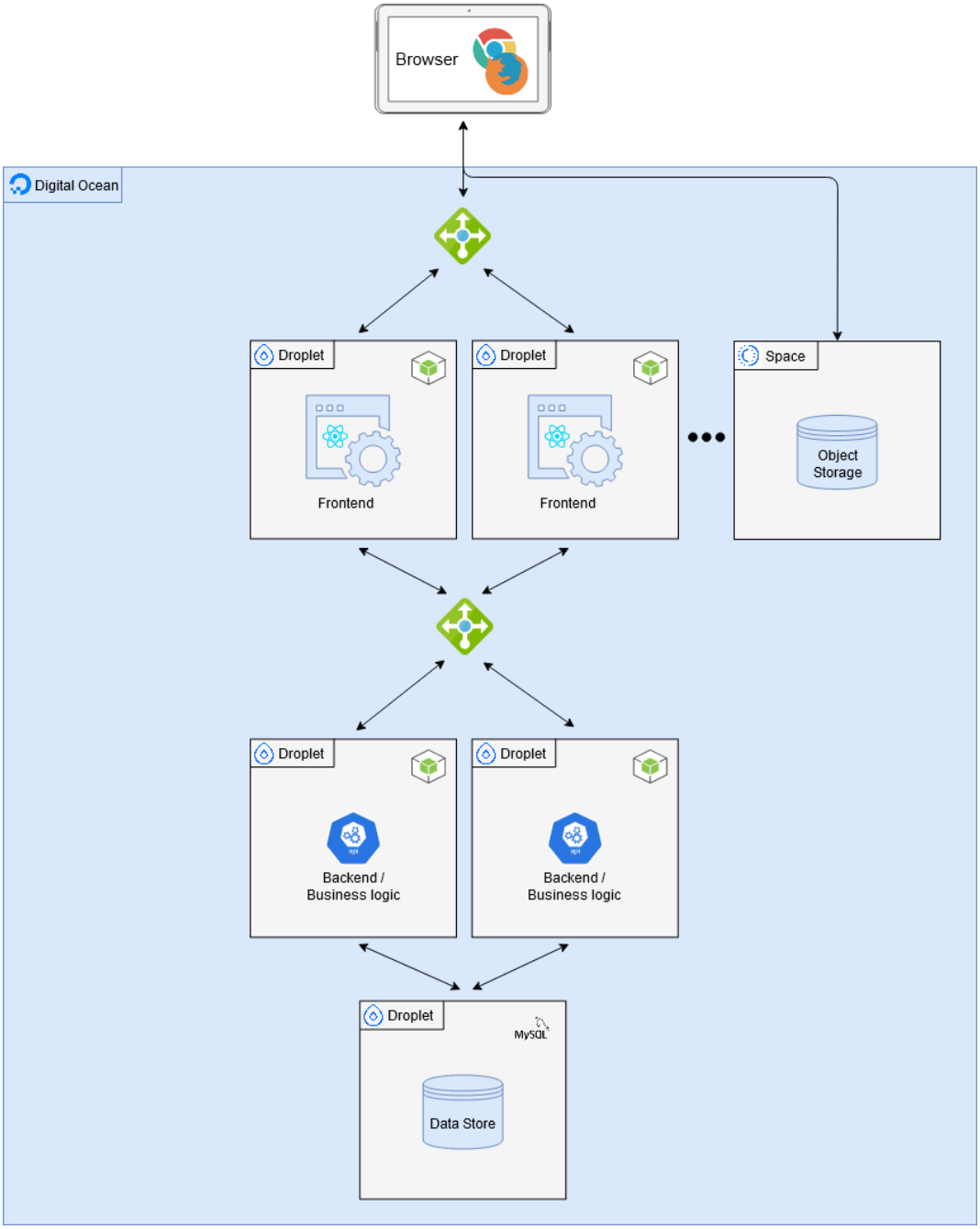

Figure 45 - Size and performance scaling

3.11 Refactoring

One challenge that was encountered during the project, was the fact that the objects lead and customer have very similar functionality. Because of this some associations between leads or customer and other entities, that were identical for both, were moved to the User superclass. This was done to avoid losing information of a lead, when converting him to a customer. This way, when a lead is converted the subclass is changed but the user with the associations persists, which is easier than transferring all associations from the old entity to the new one. This solution is not very elegant, because users that are not leads or customer could theoretically have associations, that are exclusive to leads and customers. On the other hand, it is very easy to implement and works well.
A possible solution would be to add a new entity “trainee” between leads and customer and user. This entity would contain all the shared fields and associations of lead and customer, but introduce an additional level of inheritance. This solution would also allow to merge functionalities from the customer and lead controller, which are very similar. But it would make the data structure and the role based access more complicated than it is now and potentially hard to understand for new developers.

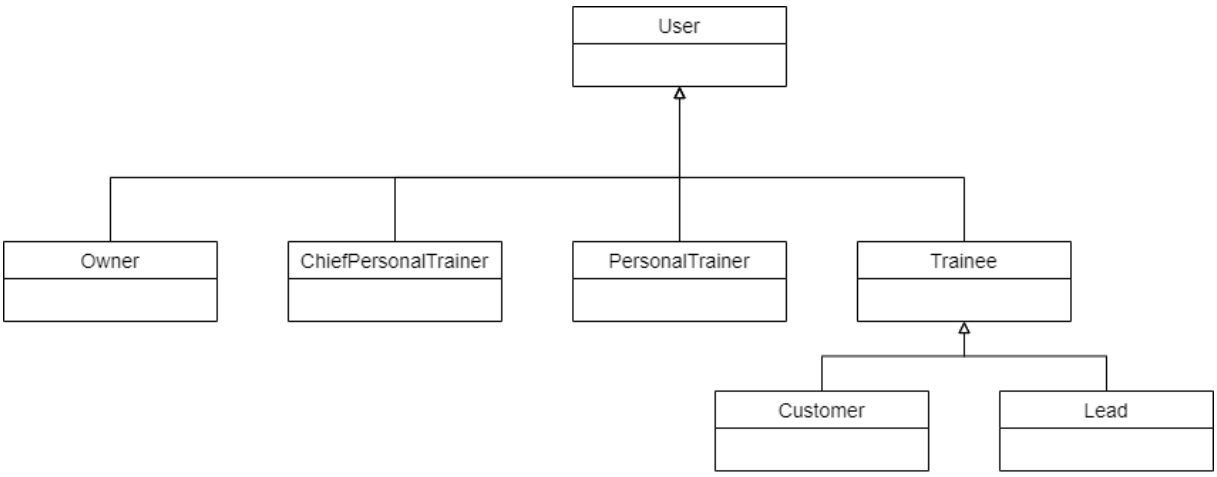

Figure 46 - Possible refactoring of roles

Whether such a refactoring should be made needs to be decided by the client, when development on the application is continued.

4. Quality assurance and testing

Several measures were taken to ensure the high quality of the end product. One part of these was to set up the development processes to ensure that the code was written correctly. The other part was to perform a variety of tests on different aspects of the product.

4.1 Performance tests

The performance of the backend was tested using a simple command line tool called ApacheBench. It is an HTTP server benchmarking tool, which can send many concurrent requests to a server and report on the response times [19]. When benchmarking the customer and lead routes with 1200 entries and no pagination the recorded response times were around two seconds, which is way too high. This was one of the reasons why pagination was added. Performance tests with 10 concurrent requests and a limit of 20 entries returned, showed that 1000 requests per minute could be successfully processed.

```
Benchmarking api.fitaas.com (be patient)

Server Software:        nginx/1.17.10
Server Hostname:        api.fitaas.com
Server Port:            443
SSL/TLS Protocol:       TLSv1.2,ECDHE-RSA-AES256-GCM-SHA384,2048,256
TLS Server Name:        api.fitaas.com

Document Path:          /lead/?limit=20&offset=0
Document Length:        12606 bytes

Concurrency Level:      10
Time taken for tests:   114.061 seconds
Complete requests:      2000
Failed requests:        0
Total transferred:      25934000 bytes
HTML transferred:       25212000 bytes
Requests per second:    17.53 [#/sec] (mean)
Time per request:       57.030 [ms] (mean, across all concurrent requests)
Transfer rate:          222.04 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:       46   96  59.3     60     276
Processing:    89  473 104.4    476    1012
Waiting:       89  472 103.7    476     998
Total:        158  569 120.2    563    1070

Percentage of the requests served within a certain time (ms)
  50%    56.3
  66%    61.1
  75%    64.7
  80%    66.4
  90%    71.1
  95%    74.4
  98%    78.8
  99%    85.2
 100%   107.0 (longest request)
```

Figure 47: Benchmarking of 10 concurrent requests of /leads to the backend

These results have to be taken with a grain of salt, because it is hard to make a quantitative statement from the response times as they also depend on the server hardware and the internet connection of the tester. But it is possible to make a qualitative assessment and rule out any major problems with response times.

4.2 Metric analysis

The use of metric analysis allows developers to find problematic parts of the code to revise. In addition, it gives an overview of the code base. To get all the metrics data, the Metrics Reloaded plugin for WebStorm was used. The tool allows to determine the number of lines of code and the number of files by file type.
The lines of code indicate how many characters are physically present in the source code. This can be derived by the number of line breaks in the project. The non-comment lines of code correspond to the effective lines of code, these are the lines of code minus all comment lines, empty lines and lines with only one opening or closing bracket.

Metrics	Size
Lines of code	7'316
Non-comment lines of code	7'275
JavaScript lines of code	7'270
Non-comment JavaScript lines of code	7'229
CSS lines of code	46
Non-comment CSS lines of code	46
Number of JavaScript files	63
Number of CSS files	2
Number of tests	11 Integration Tests

Table 6 - Metric analysis – Frontend

It takes very few lines of CSS code to style the whole application. This is because of the Semantic UI React library and the theming used. This simplifies the styling heavily.
It can be seen that comments where intentionally kept to a minimum. This was possible thanks to the use of short and meaningful names so that commenting on a method or function is not necessary. Furthermore, filename and variable names were kept as short and clear as possible.

Metrics	Size
Lines of code	7'791
Non-comment lines of code	7'725
JavaScript lines of code	7'791
Non-comment JavaScript lines of code	7'725
Number of JavaScript files	69
Number of tests	73 Unit Tests + 77 Postman

Table 7 - Metric analysis – Backend

In contrast to the frontend, which also contains CSS lines of code, the backend is pure JavaScript code. This is because the frontend contains a user interface, which needs some styling, while the backend exposes all of its functionality over an API. Other than that, the front- and backend are almost identical in scope looking at the lines of code and number of files.

Metrics	Size
Lines of code	15’107
Non-comment lines of code	15’000
JavaScript lines of code	15’061
Non-comment JavaScript lines of code	14’954
CSS lines of code	46
Non-comment CSS lines of code	46
Number of JavaScript files	132
Number of CSS files	2
Number of tests	73 Unit Tests + 77 Postman + 11 Integration Tests

Table 8 - Metric analysis – entire project

When looking at the two projects as a whole, it can be seen from the table above that an average of 112 lines of code per file were written, which is a good average. There are many different rules on how many lines of codes should be written per file, but everything under 120 should be ok [20].

4.3 Unit tests

The unit tests were performed on the backend on every route. The test framework Mocha was used to perform the tests and Chai as the assertion library. Before every test the database is reset to a predefined state. Because the entire business logic is implemented as express middleware, express has to be mocked for the unit tests. The unit tests run on the Gitlab CI runner using a local database on every push.

Figure 48 - Unit test success on GitLab CI

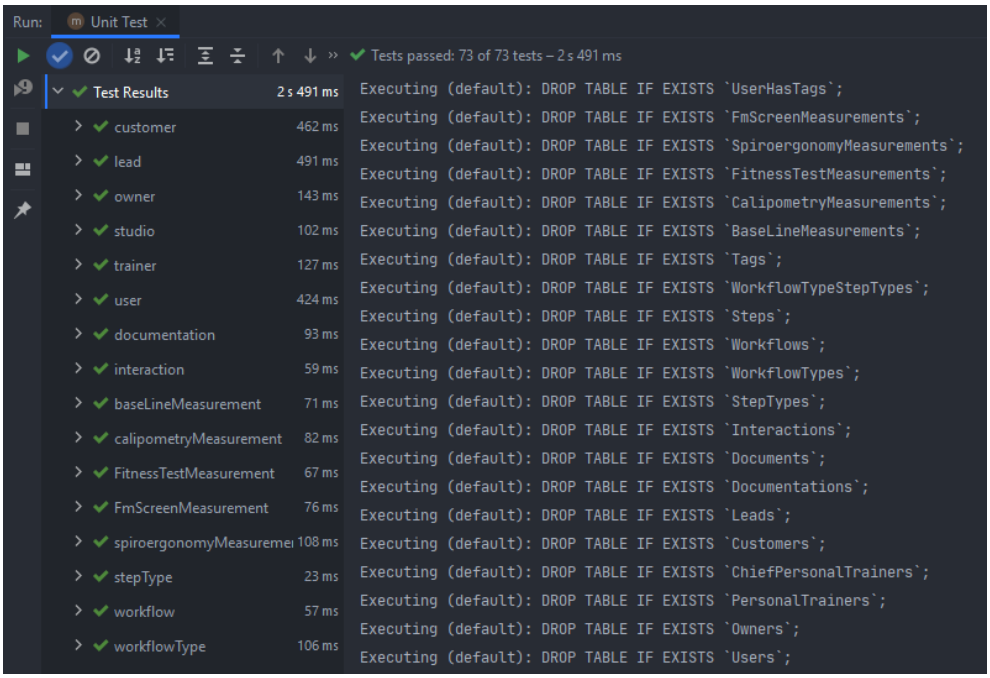

Figure 49: Unit test success in WebStorm

4.4 Integration tests

4.4.1 Postman

Postman was used to manually test the API while working on the front and backend. This is easier than writing unit tests while working on a feature, so that the test can be written as soon as a first version is working. In the appendix “API Documentation” is a more detailed documentation on how to use it. Additionally, there is also an export of all the routes with demo requests already prepared. Postman was used because of its easy setup, overview of all the requests, automatic generation of online documentation and simple starting of requests.

4.4.2 Katalon Recorder

To perform integration tests directly on the website, Katalon Recorder was used. It allows to record a series of actions and the software will repeat it later to check if the behaviour is the same [21]. When running the tests, a browser window opens, and the tests are performed in sequence, where the user is being logged out and logs in again after every test. In order for the tests to succeed the database has to be prepared using a prepared image. These tests are used to check if all the buttons and fields still are working after changes are made to the frontend.

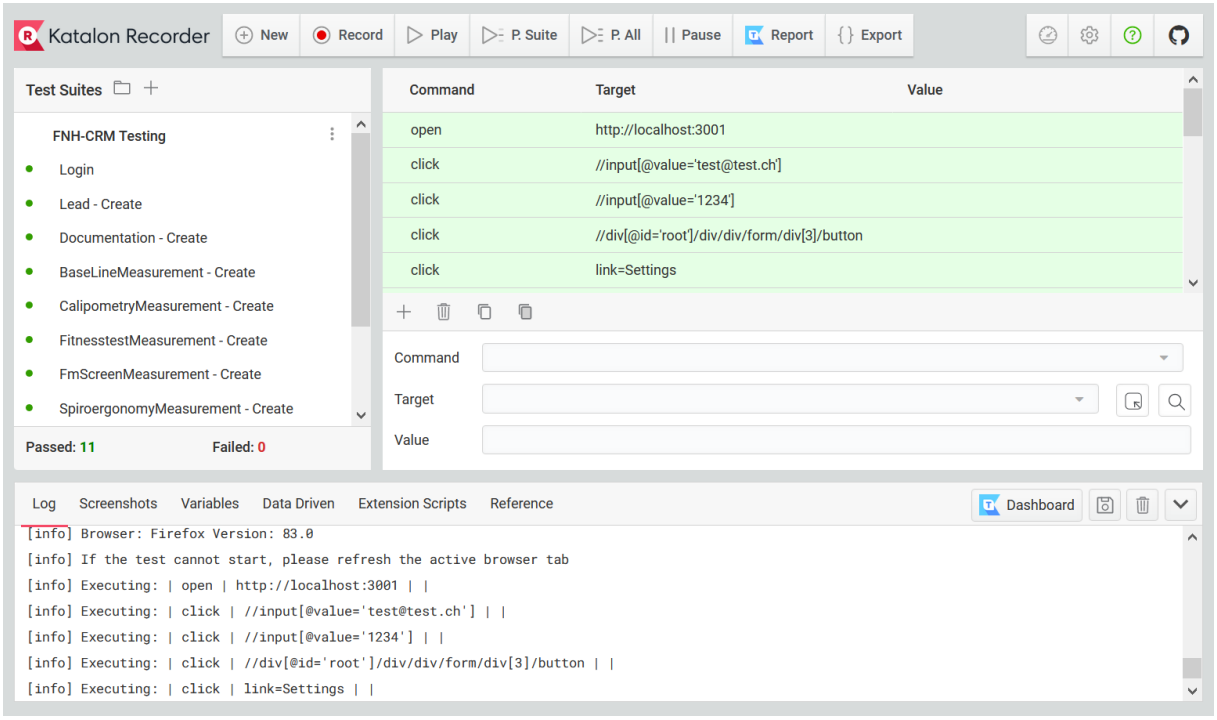

Figure 50 - Katalon Recorder with all tests succeeding

4.5 Usability tests

In order to get feedback as early and regularly as possible, the CRM was shown to the customer and project supervisor at the end of milestones. To test the usability and user-friendliness of the CRM, tests with project-independent personal trainers from the FNH were conducted.

4.5.1 General conditions

Based on the user stories and scenarios defined in the document Requirements Specifications, concrete test cases were created and played through with every test person. For the test supervisor to understand what the test person is looking for or what to expect, they were asked to think out loud. To have the same conditions for every test, a test database with all the necessary data to solve the different scenarios was set up.

Selection of test person

Usually, the test subjects should be as diverse as possible. In this case all of them have to be either personal trainers or owners of a fitness studio, because the application requires some domain knowledge to use it. One example of this would be the fitness-tests.

Test device

The users are using either a laptop or an FNH-tablet. The application can be run on Edge, Chrome or Safari.

4.5.2 Test cases

The defined scenarios should simulate the use of the app in everyday life as well as possible. Most of the functions of the CRM are covered with the following five test cases that were given to the testers.

Test Case 1: Create a new workflow

As an owner, you want to create new workflows that can be assigned to leads. This allows to have different customer journeys.
Log in as an owner and create a workflow. If necessary, create some workflow steps as well. Log out again

Test Case 2: Create a lead

As an interested person walks into the fitness studio, you want to register him as a lead.
To do so, log in as personal trainer and create a new lead filling out all the fields.
After that, log out again.

Test Case 3: Assign a workflow to a new lead and set a state

As a chief personal trainer, you want to assign a workflow to a lead, so that another personal trainer can start the process of moving / converting the lead to a customer.
To do so, log in to the application as chief personal trainer and add a workflow to a new lead. After adding the workflow, change the state to “In Kontakt”.

Test Case 4: Create an interaction

You just had a call with a customer about the bill, so you log the interaction in the customers’ profile.
To do so, you change to the customer “Michael” and register an Interaction.

Test Case 5: Manage a contract

As a lead is ready to purchase, you want to fill out a contract and let him sign it.
To do this, a few steps are required. Download the contract. Fill it out and upload it again and do the necessary steps to finish the purchase.
When all the tasks are finished, you log out of the application.

4.5.3 Evaluation

The usability test was performed between the 03.12.2020 and 05.12.2020. Following is a list with all the weaknesses that were found and the measures to solve the issues. The Complete Usability Tests can be found in the appendix.

Test case	Observations	Measures
TC1	Creating a workflow is not that intuitive because the creation of steps is in the same view. It is not clear, which input fields belong to the workflow and which ones to the creation of steps.	Moved the creation of steps into a separate box (with border around), so that it is visually better separated
TC1	When logging in, the keyboard writes the first letter of the mail address upper case on default.	Disable it and allow for the mail address to be converted to lower case.
TC2	It was not clear to the users why a password had to be entered when creating a lead.	This will not be changed because it will be necessary in the future when the customers can login themselves using an app or website.
TC3	Users did not find the workflow assignment immediately and were searching through the horizontal navigation.	When selecting a lead from the list, it will change directly to general, where the workflow is located. (Compared to the customer, where the dashboard is being shown.)
TC3	A user found out that when hovering over an element that can be clicked, the cursor does not change, so it's not that clear what can be clicked on	Was solved by the mouse pointer now changing when hovering over an element.
TC4	A user tried to create an interaction without selecting an option in the dropdown, which lead to an error message.	The dropdown has the option "Vor Ort" preselected so that this cannot happen again.
TC5	When the contract is being uploaded, the table with uploaded contracts does not auto update in the frontend, therefore users clicked on the upload-button multiple times.	The functionality of auto refreshing the table was added.

Table 9 - Test case evaluation

The test subjects also found the navigation to be consistent and clearly understandable. The design is simple but beautiful and tidy. They quickly found their way through the application after checking out the individual pages.
Because an hour was invested every few weeks to test the application and click through all the functionality, the bugs were kept to a minimum. Additionally, personal trainers working in the FNH Jona were consulted, to check the functionality especially of the tests, so that the logic is working correctly.

5. Results and outlook

5.1 Achieved goals

The achieved objectives can be measured on the basis of the functional and non-functional requirements. The functional requirements were classified as Use Cases and Optional Features with a priority of one to three. During the project, the features were subject to change by the developers and the client. Prioritizations were changed with the goal to satisfy FNH's needs and have a presentable working prototype at the end of the project. To achieve some stability over the development time, we had to fixate the core Use Cases at the start of the project. Some use cases were not fully completed.

UC7 The client decided that it does not make sense to be ablet to edit and delete interactions. This prevents the data from being manipulated.

UC8/12 Because personal trainers should not be able to delete contracts and documents in general after they are uploaded, it is not yet possible.

The changes were discussed with the customer early on and removed from the functional requirements during the project.

Use case	Description	Prioritization [1 – 3]	Implemented
UC1	CRUD users (owner, chief personal trainer, personal trainer)	1	Yes
UC2	CRUD fitness studios	1	Yes
UC3	CRUD lead/customer	1	Yes
UC4	Filter customers	3	Yes
UC5	Customer-Trainer assignment	1	Yes
UC6	Fitness studio assignment	1	Yes
UC7	CRUD customer interactions	2	Partially
UC8	CRUD customer documents	3	Partially
UC9	View customer report	1	Yes
UC10	Customize workflow	2	Yes
UC11	Manage workflow	1	Yes
UC12	CRUD contracts	2	Partially
UC13	Add measurements	3	Yes
UC14	Add customer training notes	1	Yes

Table 10 - Compliance with the functional requirements (use cases)

Additionally, three optional features were implemented.

OF15 **Add note as PDF**
It was discussed with the client, that it would makes sense to be able to upload notes from consultations as pdf, because there is no form for it yet. Therefore, it was decided to add this functionality to the document management.

OF22 **Automatically generate chief personal trainer**
If a personal trainer is assigned to a studio, his role changes automatically. This way, the owner does not have to change the personal trainer's role manually.

OF27 Lead information when converted to customer
Because the customer communication is important in a CRM, the lead gets an automatic email when converting to a customer. This could e.g. be used to send a customer information on how to use the appointment booking app that will be developed in the future.

It is to note, that even though some possible functionalities of the backend, like all the delete functions (e.g. interactions, measurements) were not implemented in the frontend, they are still implemented on the backend and available by using the REST API. The API is well documented and additional functionality can easily be added to the backend at a later stage.

Optional features	Description	Prioritization [1 – 3]	Implemented
OF1	Archive customer / lead	1	no
OF2	CRUD products	1	no
OF3	Show contract that expire soon	1	no
OF4	Contact customer	3	no
OF5	Bring a friend / brought by personal trainer	2	no
OF6	CRUD in-studio purchases	3	no
OF7	View customer billing status	3	no
OF8	Personal Trainer dashboard	3	no
OF9	Customer billing	3	no
OF10	CRUD checklist	3	no
OF11	Additional workflow information	3	no
OF12	Global search	3	no
OF13	Measurement external devices	3	no
OF14	Visual customer progress	3	no
OF15	Add note as PDF	3	yes
OF16	Consultant form	3	no
OF17	Contract as webform	3	no
OF18	Profile picture	3	no
OF19	Fitness-device settings	3	no
OF20	Timestop a subscription	2	no
OF21	Timer in “FNH Fitnesstest”	3	no
OF22	Automatically generate chief personal trainer	2	yes
OF23	Door opening system	3	no
OF24	Information button on measurements	3	no
OF25	Show a user’s birthday	2	no
OF26	Body fat calculator on calipometry measurement	3	no
OF27	Lead information when converted to customer	1	yes
OF28	Collection basin leads	2	no
OF29	Mailchimp connection	3	No

Table 11 - Compliance with the functional requirements (optional features)

Except of one non-functional requirement, they could all be completed. The requirement of every page to load within 200ms could not be achieved on every page. As it sometimes took 350ms for the leads overview page to load. Because this still is fast enough to work with the application, it was discussed with client and agreed on moving the limit to 350ms.

Non-Functional requirements	Fulfilled	Comment
Functional suitability		
NFR1	Yes	Not directly measurable
Performance efficiency		
NFR2	Yes	The customer pages show immediately, the data is being loaded right away
NFR3	Yes	Tested with performance test
NFR4	Partially	Because of the filtering and the amount of data, it sometimes takes the lead overview page 350ms to load
Compatibility		
NFR5	Yes	Not directly measurable, continuous manual testing through a Click Funnels page connected to Zapier
NFR6	Yes	It is a MySQL database, for which Power BI offers connectors
NFR7	Yes	Was tested on notebooks and tablets
Usability		
NFR8	Yes	The web application is available on: https://crm.fitaas.com/
NFR9	Yes	Additionally, two more cases were tested - Create workflow - Create lead
NFR10	Yes	The test users rated the application with 8 or higher
Reliability		
NFR11	Yes	Guaranteed by the cloud service provider
NFR12	Yes	
NFR13	Yes	
NFR14	Yes	Pm2 takes care of the logging. The size of the log file is limited to 50mb.
Security		
NFR15	Yes	All communication is SSL encrypted using a domain validated certificate from “Let’s Encrypt” [22], which is applied to the Nginx reverse proxy.
NFR16	Yes	
NFR17	Yes	All passwords are hashed
NFR18	Yes	Covered with the role concept
NFR19	Yes	
Maintainability		
NFR20	Yes	The logic was put into separate functions
NFR21	Yes	Covered with 73 Units test
NFR22	Yes	Postman was used to test the backend
Portability		
NFR23	Yes	All hosted on different droplets / managed database

Table 12 - Compliance with the non-functional requirements

5.2 Outlook and extension

The following section explains how the application could be further extended, modified and improved with four cases as example. Additionally, to the two presented features, there is an entire list of additional features in the appendix. Rough concepts are presented. The feasibility of these extensions was not tested but only theoretically worked out. The first extension “Collection basin for leads” is a possible extension that was proposed to the client. The second one, to automate the interactions with a customer, was discussed with the client early on but had to be left out because of the limited time. The third one would add an entire mail-marketing functionality to the application and spare the owner of having to login to Mailchimp, every time he wants to send a mass mail to a lot of customers / leads. The last feature described is the appointment booking, which is important to the customer.

5.2.1 Collection basin leads

Thanks to Zapier, it is possible for leads to sign up through different channels. Because not all of them can return the same amount of data. A website might only return the first-, last name and the phone number.

Figure 51 - ClickFunnels example form

This is still enough to contact the lead but not enough to add it to the system as a lead. To work a lead, a personal trainer and studio have to be set. To guarantee this, every lead would enter the collection basin, where a chief personal trainer or owner would assign the needed data before it is converted to a lead inside the system. This additional layer guarantees that only clean data is inside the system, no matter, from where the leads signed up.
Later on, workflows could be assigned automatically depending on the channel they arrived from. If the capacity of a personal trainer could be logged inside the application as well, the personal trainer could also be assigned automatically. This way, the lead basin would only be needed for special cases.

5.2.2 Customer interactions

This issue was registered as OF4. At this point, all the interactions can be registered manually in the system. The plan is to have a button, that when clicked generates an automated mail or WhatsApp message to a lead or customer, that e.g. asks him if he would like to book a new personal training session or group training. This message would include a link to the website, where the session can be booked. A similar thing was done on the workflow using SendGrid, where a lead gets a welcome mail as soon as he is converted to a customer.
A personal trainer has to contact his leads / customers every few weeks, adding this basic interaction would save a lot of time. An input field could be added, for the personal trainer to append some personal words to the message.

5.2.3 Mailchimp connection

This issue was registered as OF28. As a user enters a system or signs up for come campaign, tags are added. Mailchimp uses tags to perform actions as soon as a tag is set to a user in their system. These could be sending them to a newsletter, a single information mail or just removing them from a mailing-list. An example tag would be 2020-Strong_women-04, which would be the “Strong women”-campaign that was launched in April 2020. Every user that has this tag would then receive a few mails tailored to their interest in “Strong Women”.
To add this principle to the CRM, tags are part of the current system. At this point, they can be stored in the database and displayed in the frontend. In the future, they could be linked and synchronized using the Mailchimp API and be managed in the CRM as well. This would make it possible to use a lot of Mailchimp functionality inside the application. For example, if a lead converts to a customer, it could automatically be added to the internal newsletter for customers and be removed as soon as he is not a customer anymore (at the moment, a comparison of the effective customers and the ones in Mailchimp is performed manually twice a year).
This could then easily be extended to e.g. inform a customer if his subscription is about to end, send some information concerning the exact problems the customer has.

5.2.4 Appointment booking

The fitness studio could not work without an appointment booking app for the customers. The current CRM is missing this essential component. The customer decided to build this as soon as possible, therefore it is analysed in more detail in the form of an implementation study, which can be found in appendix J.

Appendix

A. Glossary

Term	Description
Adobe XD	User experience design-tool by Adobe to create mockups and test them on devices e.g. phone.
Angular	Frontend framework to build mobile and desktop web applications
ApacheBench	For http server benchmarking, in our case speed testing the backend.
API	An **A**pplication **P**rogramming **I**nterface is a program part that is made available by a software system to other programs for connection to the system.
AWS S3	**A**mazon **W**eb **S**ervice, **S**imple **S**torage **S**ervice, is an object storage service technology by Amazon.
BA	Brief for "**B**achelor **A**rbeit", means bachelor thesis.
Bearer-Token	A security token, a kind of non-transparent value that is created by the authentication server. It is a hash, generated using the user id and a random seed. Grants access to the application and identifies the user.
Chai	Assertion library for Node.js, can be paired with any test framework.
CHF	**C**onfoederatio **H**elvetica **F**ranc, is the abbreviation for the Swiss franc.
CI/CD	**C**ontinuous **I**ntegration **/ C**ontinuous **D**eployment, by enforcing automation in building, testing and deploying applications, it bridges the gaps between development and operations activities and teams.
ClickFunnels	Is a marketing tool that helps to acquire customers.
CPU	**C**entral **P**rocessing **U**nit, main part of a computer system, generally consists of main memory, control unit and arithmetic-logic unit.
CRM	**C**ustomer **R**elationship **M**anagement, describes a tool for customer relationship management. It focuses on customers and customer relationship processes.
CRUD	**C**reate data record, **R**ead data set, **U**pdate record, **D**elete data record.
CSS	**C**ascading **S**tyle **S**heets is a stylesheet language for electronic documents and one of the core languages of the World Wide Web.
DataGrip	A cross-platform IDE for databases by JetBrains.
DevOps	DevOps consists of the terms development and it operations. It describes the approach of improving software development and IT operations.
DigitalOcean	US cloud infrastructure provider often used for the simple setup of cloud services.
Docker	Free software that offers OS-level virtualization to e.g. run an isolated database on a local machine.
Droplet	Different types of virtual machines by DigitalOcean, hosted in one of their datacentres.
DSL	**D**igital **S**ubscriber **L**ine, high speed bandwidth connection technology to transfer data over telephone lines.
FNH	**F**itness **N**utrition **H**ealth, name of the client of this thesis.
GitLab	Project- and source code management software.

GUI	A **G**raphical **U**ser **I**nterface a type of user interface through which users interact with electronic devices via visual display representations.
HTML	The **H**yper**T**ext **M**arkup **L**anguage is a text-based markup language for structuring electronic documents such as texts with hyperlinks, images and other content.
HTTP	**H**yper**T**ext **T**ransfer **P**rotocol is a stateless protocol for transmitting data on the application layer over a computer network.
IDE	**I**ntegrated **D**evelopment **E**nvironment is an application used for developing software. It offers a source code editor, build automation tools, a debugger, source control integration and more.
IP	**I**nternet **P**rotocol, a network protocol.
ISO	The **I**nternational **O**rganization for **S**tandardization is the international association of standardisation organisations.
IT	**I**nformation **T**echnology is the use of computers to store, retrieve, transmit, and manipulate data or information
JavaScript	Programming language mostly used for Web-development.
JetBrains	Is a multinational software company.
Jira	Project management tool by Atlassian.
JSON	**J**ava**S**cript **O**bject **N**otation is a compact data format in an easily readable text form and serves the purpose of data exchange between applications.
Katalon Studio	An automation test solution developed by Katalon LLC.
Lead	In marketing, a person interested to buy a product but did not yet buy one.
Let's Encrypt	Free, automated and open certificate authority provided by the Internet Security Research Group (ISRG).
Mailchimp	A marketing automation platform and email marketing service.
Mocha	JavaScript test framework for Node.js programs.
MySQL	Is one of the most common relational database management systems.
Nginx	Free and open-source web server, that can be used as reverse proxy, load balancer, mail proxy and HTTP cache.
NFR	**N**on-**F**unctional **R**equirements is a requirement that lays down criteria by which the operation of a system, rather than a specific behaviour, can be evaluated.
Node.js	Server-side JavaScript platform, developed by Google, most often used for Webservers.
NPM	**N**ode **P**ackage **M**anager is a package manager for the JavaScript programming language.
Object Storage	Is a computer data storage architecture that manages data as objects. Used to store unstructured data like images and pdf's.
OF	**O**ptional **F**eature, a functionality that can be implemented, but is not mandatory.
OR-Mapper	**O**bject **R**elational **M**apper, a software that maps object-oriented data to a relational representation in a database.
OS	**O**perating **S**ystem
PDF	**P**ortable **D**ocument **F**ormat
Postman	Is a Tool for developing and testing API's.

Power BI	**Power B**usiness **I**ntelligence is a tool from Microsoft to provide interactive visualizations and business intelligence capabilities within one interface.
RAM	**R**andom **A**ccess **M**emory
React	JavaScript library for building user interfaces.
Redux	Is an open source JavaScript library for managing state information in a web application.
REST-API	**RE**presentational **S**tate **T**ransfer-**A**pplication **P**rogramming **I**nterface Programming interface that explains the behaviour between the client and server. Is used to describe a restful HTTP API.
Salesforce	Is an international provider of cloud computing solutions that provides a customer relationship management software.
Semantic UI	Semantic is a development framework that helps create beautiful, responsive layouts with human-friendly HTML.
SendGrid	Tool to send customized mails.
SendinBlue	Is an online marketing tool for relationship marketing.
Sequelize	Sequelize is a powerful library in JavaScript that makes it easy to manage an SQL database. At its core, Sequelize is an object-relational mapper.
SLA	A **S**ervice **L**evel **A**greement is a basic agreement between the client and the service provider for recurring services. Often used in the context of cloud service providers.
SMS	**S**hort **M**essage **S**ervice, a system for sending text messages from one mobile phone to another.
SQL	The **S**tructured **Q**uery **L**anguage is a database language to define data structures in relational databases and for editing (inserting, changing, deleting) and querying databases based on them.
SSH	**S**ecure **SH**ell describes a cryptographic network protocol for the secure operation of network services over unsecured networks.
Swagger	Is a collection of open source tools to design, build, document and use HTTP web services.
UC	**U**se **C**ases describe the behaviour of a system from the user's point of view.
UI	**U**ser **I**nterface
URL	A **U**niform **R**esource **L**ocator identifies and locates a resource via the access method to be used and the location of the resource in computer networks.
Virtuagym	A comprehensive all-in-one fitness studio / personal training software solution.
VM	A **V**irtual **M**achine is an emulation of a computer system and offers the functionality of a physical computer.
Webstorm	Is an integrated development environment from the company JetBrains for web development.
Whatsapp	Is an instant messaging service by Facebook.
Zapier	Is an online service provider that manages interfaces. It moves data between different applications and services.

B. References

[1] Statista, "Statista - Health & Fitness Clubs - Statistics & Facts," [Online]. Available: https://www.statista.com/topics/1141/health-and-fitness-clubs/. [Accessed 14 12 2020].

[2] M. Rehkopf, “User Stories with Examples and Template,” [Online]. Available: https://www.atlassian.com/agile/project-management/user-stories. [Accessed 20 12 2020].

[3] ISO/IEC, 9126, Software engineering - Product quality, 2001.

[4] E. Elliott, “How Popular is JavaScript in 2019?,” Medium, 11 05 2019. [Online]. Available: https://medium.com/javascript-scene/how-popular-is-javascript-in-2019-823712f7c4b1. [Accessed 20 12 2020].

[5] Atul, “When should I use Node JS for backend ?,” Medium, 5 11 2016. [Online]. Available: https://medium.com/@atulanand94/when-should-i-use-node-js-for-backend-8379d6b224d2. [Accessed 20 12 2020].

[6] Facebook Inc., "React," [Online]. Available: https://reactjs.org/. [Accessed 20 12 2020].

[7] D. Abramov, «React Redux,» [Online]. Available: https://react-redux.js.org/. [Zugriff am 20 12 2020].

[8] Swagger, "https://swagger.io," [Online]. Available: https://swagger.io. [Accessed 20 12 2020].

[9] Postman, Inc., [Online]. Available: https://www.postman.com/. [Accessed 20 12 2020].

[10] MySQL, “https://www.mysql.com,” [Online]. Available: https://www.mysql.com. [Accessed 20 12 2020].

[11] Microsoft, "Relational vs. NoSQL data," [Online]. Available: https://docs.microsoft.com/en-us/dotnet/architecture/cloud-native/relational-vs-nosql-data. [Accessed 20 12 2020].

[12] Sequelize, "Sequelize ORM," [Online]. Available: https://sequelize.org/. [Accessed 20 12 2020].

[13] Semantic Org, "Semantic UI React - The official Semantic-UI-React integration," [Online]. Available: https://react.semantic-ui.com/. [Accessed 20 12 2020].

[14] JetBrains s.r.o, [Online]. Available: https://www.jetbrains.com/webstorm/. [Accessed 20 12 2020].

[15] GitLab Inc., [Online]. Available: https://about.gitlab.com/. [Accessed 20 12 2020].

[16] Docker, Inc., [Online]. Available: https://www.docker.com/. [Accessed 20 12 2020].

[17] Jetbrains, "https://www.jetbrains.com/de-de/datagrip," [Online]. Available: https://www.jetbrains.com/de-de/datagrip. [Accessed 20 12 2020].

[18] Atlassian, [Online]. Available: https://www.atlassian.com/software/jira. [Accessed 20 12 2020].

[19] Apache Software Foundation, “ab - Apache HTTP server benchmarking tool,” [Online]. Available: https://httpd.apache.org/docs/2.4/programs/ab.html. [Accessed 20 12 2020].

[20] D. Keller, Software Engineering 1, Rapperswil: HSR Script, 2019.

[21] Katalon LLC., “Katalon Recorder - A lightweight extension for Record and Playback,” [Online]. [Accessed 20 12 2020].

[22] Internet Security Research Group (ISRG), "Let's Encrypt - How It Works," 18 10 2019. [Online]. Available: https://letsencrypt.org/how-it-works/. [Accessed 20 12 2020].

C. Table of figures

D. Table List

E. Project plan

1. Project Overview

FNH is using a lot of different tools which combined offer the CRM functionality they need.

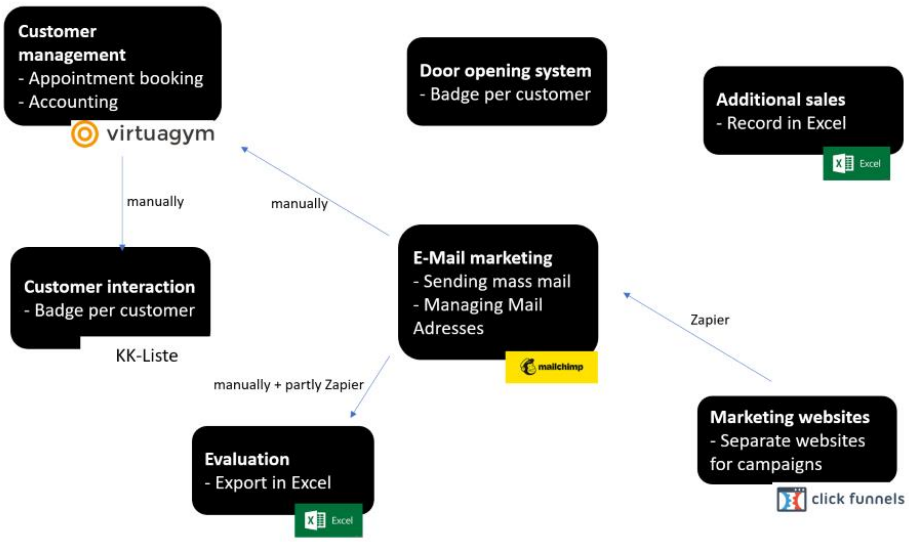

Figure 1 - Current solution

The tools that would combine most of these functionalities are very expensive and complex. Managing customers over different applications causes a lot of overhead. The interfaces between the tools do not work perfectly together. FNH wants a centralized application, which is tailored to their specific needs. The application needs to be a web application that can run on tablets. It should also be easy to use, so that the personal trainers can easily work with it on their tablets.

1.1 Purpose and goal

This is a first overview of the planned solution to digitize FNH's processes.

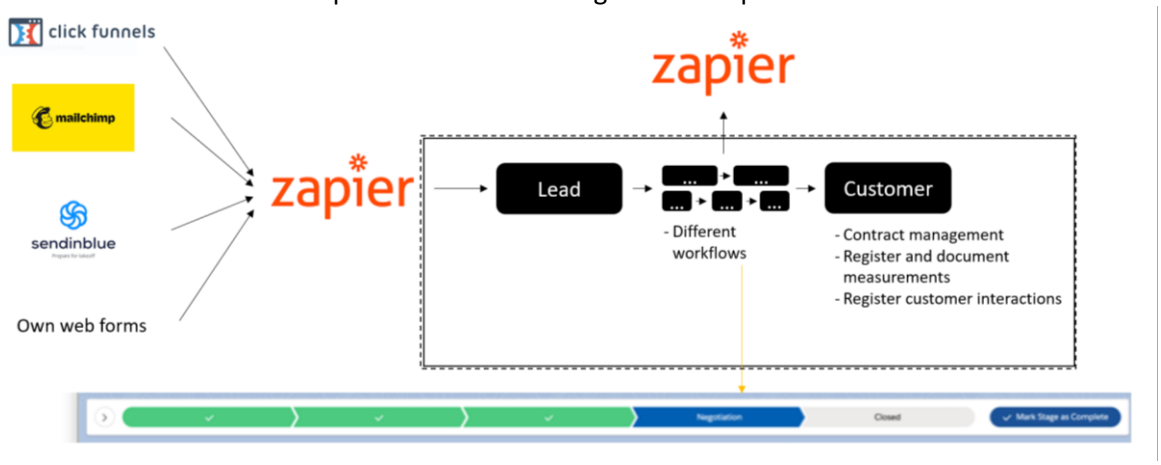

Figure 2 - Future solution

Develop a web application that is tailored to the specific need of FNH and easy to use for the personal trainers. The application should be light weight, so that it can work on tablets. Furthermore, the system needs to be extensible for future expansions.

1.2 Scope of delivery

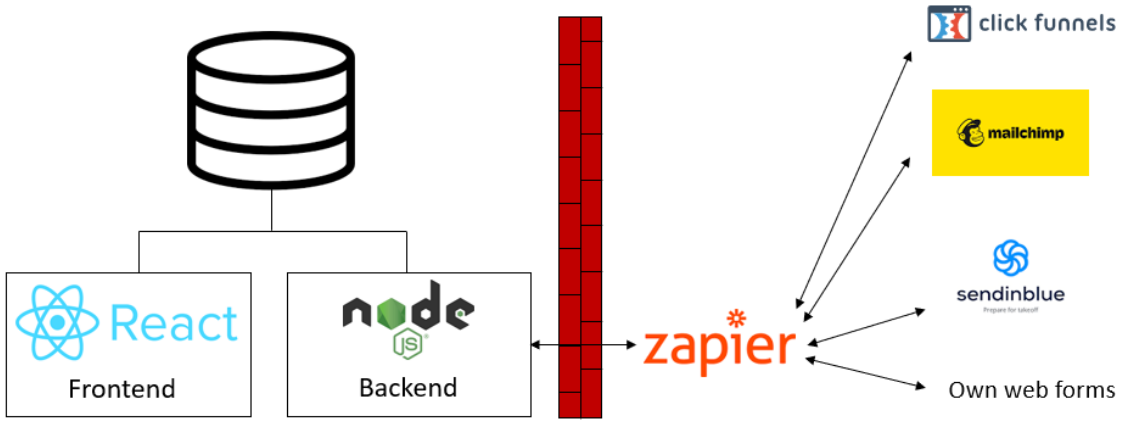

Figure 3 - Scope of delivery

Three different components will be delivered:
- A Node.js Server managing the data access.
- A React web application for personal trainers.
- A relational database.

1.3 Technologies

React

React is a technology from Facebook and is very popular. This should guarantee long term support. There is a lot of documentation online and the learning curve is very fast.

Node.js

Node.js allows to start working very quickly, due to its simplicity. It is easy to host a Node.js server, because the technology is very popular. Because of its popularity there are a lot of open-source libraries for various applications.

Javascript

The developing language for both React and Node.js is JavaScript. Using the same language in both components makes it much easier for the developers. JavaScript is a very popular language, with a big community.

1.4 Assumptions and limitations

The workload should amount to about 24 hours per team member and week. Due to the large scope of the project, the focus will be on implementing the core functionality. A lot of time will be spent on planning (organisation, time management, architecture and design). This will ensure that the rest of the project will run smoothly and the application will be extensible in the future.

To have the possibility to outsource further development, the entire work (code, documentation, …) will be done in English. This is not the mother tongue of the team members and will require additional time when working.

2. Project organization

The project is carried out according to the Scrum-Plus[3] method, which we use in the 'Software-Engineering' course. The project is divided into the four phases inception, elaboration, construction and transition according to the 'Unified Process'[4]. From the elaboration phase on, the Phases run according to the Scrum method and are divided into sprints.

2.1 Organizational structure

The project will be realized within a flat organizational structure.

Name	Position	E-Mail	Responsibility
***	Developer	***	Documentation QA & Testing
***	Developer	***	Continuous Integration & Deployment
***	Developer	***	Project administration & Customer communication

Table 1 - Organizational structure

2.2 External interfaces

The following external interfaces are defined for the project.

Name	Position	E-Mail	Responsibility
***	Supervisor	***	Consulting Supervision
***	External Co-Examiner	-	Examine the work
***	Customer	***	Product owner
***	Customer	***	Product owner

Table 2 - External interfaces

3. Management procedures

3.1 Time planning

The project officially started on September the 14th. Every student will invest 360 hours. The project has to be finished by the 25nd of December.

Phase	Effort (Team)	Weeks	Milestones
Inception	144	2	End of Inception
Elaboration	144	2	End of Elaboration
Construction	648	9	End of Construction
Transition	144	2	End of Transition

Table 3 - Time planning phase

[3] Chapter 12 from https://skripte.hsr.ch/Informatik/Fachbereich/Software-Engineering_2/SE2/hsr-se2-skript-2019-v1.2.pdf
[4] Chapter 9 from https://skripte.hsr.ch/Informatik/Fachbereich/Software-Engineering_1/SE1/hsr-se1-skript-2018-v1.9.pdf

3.1.1 Phases / Iterations

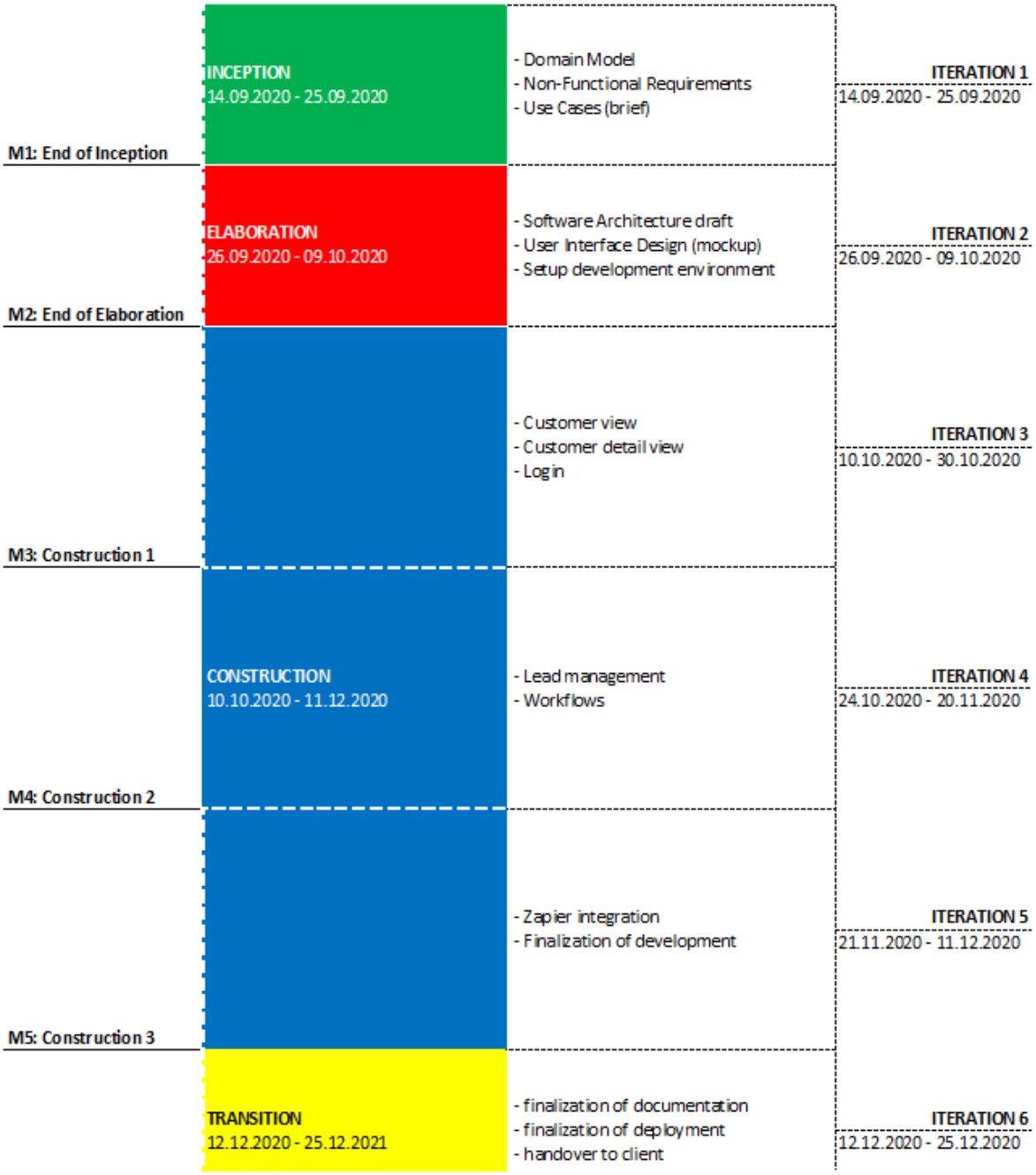

Figure 4 - Iterations

3.1.2 Milestones

M	Deadline	Name	Tasks
M1	25.09.2020	End of Inception	- Domain Model - Non-Functional Requirements - Use Cases in brief format
M2	09.10.2020	End of Elaboration	- Software Architecture draft - User Interface Design (mockup) - Setup development environment - Study: appointment booking by studio clients
M3	30.10.2020	Construction 1	- Customer view - Customer detail view - Login
M4	20.11.2020	Construction 2	- Lead management - Workflows
M5	11.12.2020	Construction 3	- Zapier integration - Finalization of development
M6	25.12.2020	End of Transition	- finalization of documentation - finalization of deployment - handover to client

Table 4 - Milestones

The description of the milestones represents where the development focus lies and does not mean that all features belonging to this topic will be finished. The detailed planning will be agile and defined as use cases.

M1 – End of Inception

Design of the domain model, of non-functional requirements, use cases in short form and use cases in fully dressed form if required.

M2 – End of Elaboration

Designing the software architecture, designing the user interface (mockup), setting up the development environment. In order to make sure, that the architecture is extensible, a study on a concrete expansion is done: appointment booking by studio clients.

M3 – Construction 1

In this first short construction phase, the goal is to have a simple application with core functionality, on the basis of which the further development can be coordinated with the customer.

Customer management

Is an overview of all the registered Customers.

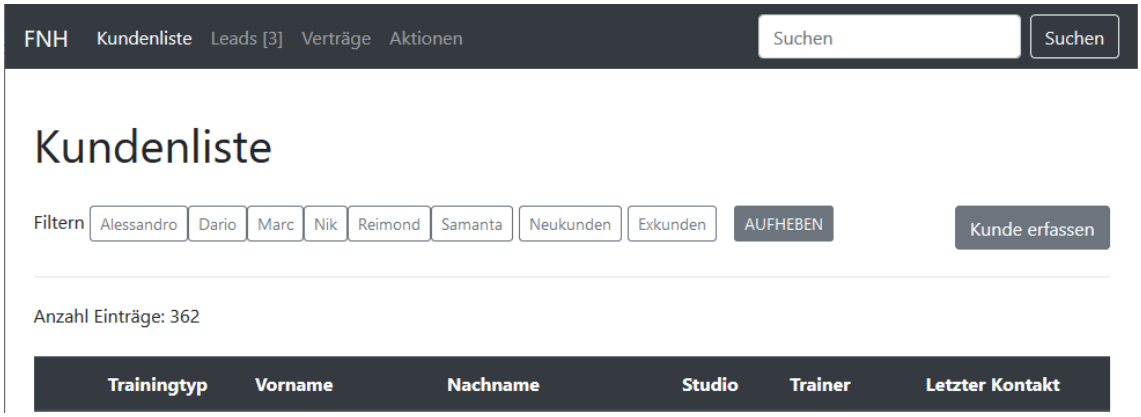

Figure 5 - Old solution of customer list «KK-Liste»

Customer view

Shows all the customers of both studios (Jona, Siebnen). An entry can have different colours:

- White: no action required
- Yellow: customer has not been contacted for the last seven days
- Red: customer has not been contacted for two weeks

Customer detail view

Different actions can be performed on a customer. Because of all the different information there should be registers, the user can click through.

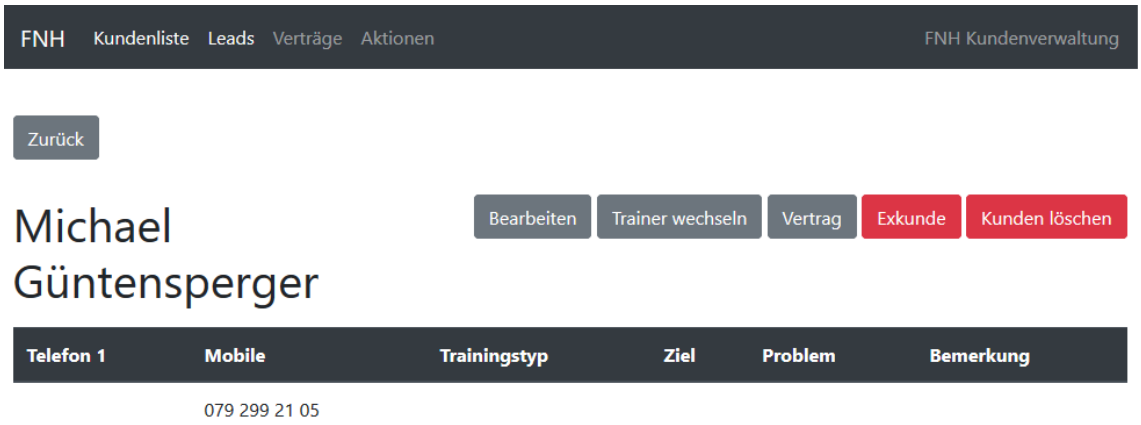

Figure 6 - Old solution of customer detail view «KK-Liste»

M4 – Construction 2

In this phase, the goal is to implement all the needed core functionalities, including a big part of the business logic.

Lead management

A lead can come from different channels / marketing campaigns. Different actions are performed by the personal trainer on a lead, before it is converted to a customer.

Tracking a lead

A lead can enter the system through different channels. This history has to be tracked, so that the user always knows, where a lead came from.

Workflows

Not all the leads are handled the same way, there are different workflows. The user needs to be able to create and manage these workflows. When tracking a lead, the user sees the current workflow step visually.

Figure 7 - Sample workflow

M5 – Construction 3

The goal of the last construction phase is to integrate Zapier, to be able to work with external tools.

Zapier integration

Example for the connection of Mailchimp to the FNH-CRM (outside-in) via Zapier (used for E-Mail marketing and the subscription to newsletters).

Figure 8 - Process illustration (outside-in)

Example for the connection of the FNH-CRM to Zapier for sending a mail (inside-out). It should be possible to setup a Zapier-action for every Workflow-step.

Figure 9 - Process illustration (inside-out)

Register the API to Zapier. This helps to connect further applications to the FNH-CRM without touching the code. For this to work, the FNH-CRM-API has to be optimized to work with different data inputs / outputs.

M6 – Transition

Completion of the documentation and handover of the customer.

3.2 Meetings

Meeting notes will be saved in pdf to the Teams available to all involved in the project.
The purpose of these meetings is to provide information, exchange information, make decisions and discuss problems. Important matters to be discussed with FNH must involve the supervisor. Minor issues and questions are discussed by the dev team and FNH without supervisor and documented. The work is presented and discussed at the end of every milestone with FNH and the supervisor.

4. Risk management

4.1 Risks

Nr	Title	Description	Preventive measures	Behaviour in case of occurrence
R1	Unknown technologies	It can take a lot of time to get in to unfamiliar technologies. It is not easy to predict how much time this familiarization takes.	Prevention almost impossible. Minimize risk by using well known technologies with strong documentation.	Discussion with team members, knowledge transfer
R2	Wrong priorities (GUI)	In this project the focus is on usability. This can be dangerous if the team invests too much time to develop a perfect GUI, without being able to judge how much time is needed. This can compromise the project, because the time will be missing at the end.	Plan well, stay realistic, analyse in the team	Analyse, rethink
R3	Too ambitious	Too much time is used for one thing. This time is then missing at other points.	Short iterations with feedback	Cut features
R4	Wrong technologies	Choosing wrong technologies, which are either unsuited or cause problems, can cost a huge amount of time. Realising such a problem too late could compromise the entire project	Strong prototype	React as quick as possible
R5	Data loss with Jira or GitLab	The possibility of data loss on the platforms Jira and/or GitLab.	Create local backups regularly.	Restore from local backups and redo lost work as quick as possible.
R6	Authentication and authorization	Implementation of role-based authentication and authorization is more complex than expected	Evaluate several frameworks and methods during the elaboration phase	Consult with experienced developers. Implement only basic authorization

Table 5 - Risks

The evaluation of the risks is as follows.

Nr	Max. damage [h]	Probability of occurrence	Weighted damage	Probability of occurrence after preventive measures	Weighted damage after preventive measures
R1	48	5%	2.4	2%	0.96
R2	72	5%	3.6	2%	1.44
R3	72	5%	3.6	2%	1.44
R4	96	5%	4.8	2%	1.92
R5	72	1%	0.72	0.5%	0.36
R6	48	8%	3.84	1%	0.48
Sum	**408**		**18.96**		**6.6**

Table 6 - Risk evaluation

The weighted damage before the preventive measures is 18.96, after applying the preventive measures it is only 6.6, which is a very low value. This is due to the team already working together in the past and the well-known technologies.

4.2 Dealing with risks

An important point in dealing with risks is that they are always present and will be involved in the relevant decisions. The defined preventive Measures must be consistently applied and adhered to in order to minimize the probability of occurrence as far as possible. Reserves are included in the estimation of the work packages and the activities of adapted to the corresponding package.

In order to prevent the risk of miscalculations and unforeseen work, we include a buffer in each work package. This buffer is about 5% of the estimated time.
During the project, this value can be adjusted if experience shows that the estimation is inaccurate. This value adjustment takes place at the beginning of the iteration planning.

5. Work packages

The project management will be done in Jira.

Only the developers and the supervisor have access.

For this project the agile framework Scrum+ is used and therefore the work packages are recorded, estimated, prioritized and scheduled in an iteration. These packages are called tasks and are estimated using story points. The idea of story points is to estimate how hard a task is to complete. The points have a Fibonacci-like format: 0, 0.5, 1, 2, 3, 5, 8, 13, 20. No task should take longer than 16 hours. If a task seems to big, it has to be broken up. Over time the team should get a feeling of how many story points they can complete in a certain amount of time.[5]

[5] https://www.atlassian.com/agile/project-management/estimation

6. Infrastructure

6.1 Developer tools

Gitlab	Code versioning
Webstorm	Programming
Gitlab CI/CD	Continuous Integration/ Deployment
DigitalOcean	Deployment
Docker	Container
EsLint	Code Quality
Mocha	Unit Testing
Katalon Recorder	Integration Test
Postman	API Test

Table 7 - Developer tools

6.2 Project management

Jira	Time tracking / Issue management
Microsoft Teams	Communication and collaborative editing of documents
Jira Add-on « Time Reports »	So that time evaluation can be run per student per day
Draw.io	Cloud tool for the creation of diagrams

Table 8 - Project management

7. Quality measures

To ensure a high quality of the final product, we will take the following measures:

- In-person dev team meetings: weekly (every Friday afternoon)
- Code presentation: code is presented regularly in the team
- Pair programming: to ensure knowledge transfer we will do pair programming
- Reviews with FNH: at the end of every milestone
- Testing: continuous unit and integration tests

7.1 Documentation

The documentation files are shared with Teams (Office files).
The MS-Office files are edited directly on SharePoint so that multiple people can work on it at the same time and the files are synchronized with Teams. SharePoint automatically keeps a change history which allows us to access older versions of the files.

7.2 Project Management

Jira is used as the project management tool as described in chapter 6.2.

7.2.1 Definition of Done

An issue is closed as soon as the following requirements are met:

Code

- Code is correctly commented, where necessary
- Only tidied up code (e.g no commented code)
- Code compiles without errors and warnings
- Unfinished parts are marked with 'TODO:'

Testing

- API-requests and Integration tests run without errors.
- Six eyes principle: The code is discussed and approved in the team.
- Test coverage will be defined in a separate document.

Jira

- Issue is marked as closed and all the time spent on it has been logged.

7.3 Development

To ensure proper version control and teamwork, the code files are managed with git[6] and stored online on a GitLab[7] Repository.
For every feature, a new branch is created. A feature branch can only be pushed to the main branch if all the requirements of chapter 7.2.1 “Definition of Done” are met. The branches are merged using pull requests and the four-eye principle.
For every part of the project, there is a separate repository:

- Frontend
- Backend

7.3.1 Procedure

Every commit has a meaningful comment. A commit should contain only a single functional change and be self-contained.

7.3.2 Unit Testing

Unit tests are created for the most important functions. The application fetches the data from the database server. We will concentrate only on unit tests that check the business logic.

7.3.3 Code Style Guidelines

For React and Node.js we will be using the default ESLint rules[8].

7.4 Testing

All manual tests that are performed are recorded in the respective test protocols.

7.4.1 Unit Tests

Unit tests as defined in 7.3.2 are created and run on a build server after every push to GitLab.

7.4.2 Integration Tests

Complex business logic applications are tested using integration tests, if necessary.

7.4.3 API Tests

API tests are used to ensure responses by the backend API are correct and in the right format.

7.4.4 System Tests

System tests are performed by FNH personal trainers following instructions and specifications by the dev team to test the functionality of the business logic.

7.4.5 Usability Tests

Usability tests are already carried out during development. This should help us to identify problems in the UI and the developed business processes as quickly as possible.

[6] https://git-scm.com/
[7] https://about.gitlab.com/
[8] https://eslint.org/docs/rules/

F. Final application

Login details
There are different logins, depending on the role with different rights.

Personal Trainer:

Mail: *******	Password: ******

Chief Personal Trainer:

Mail: *******	Password: ******

Owner:

Mail: *******	Password: ******

Link details
The web-based application can be accessed through the following link:
https://crm.fitaas.com/

Screenshots of the Application
The following screenshots should give an overview of the application without logging in. The aim was to show as much functionality in one screenshot as possible.

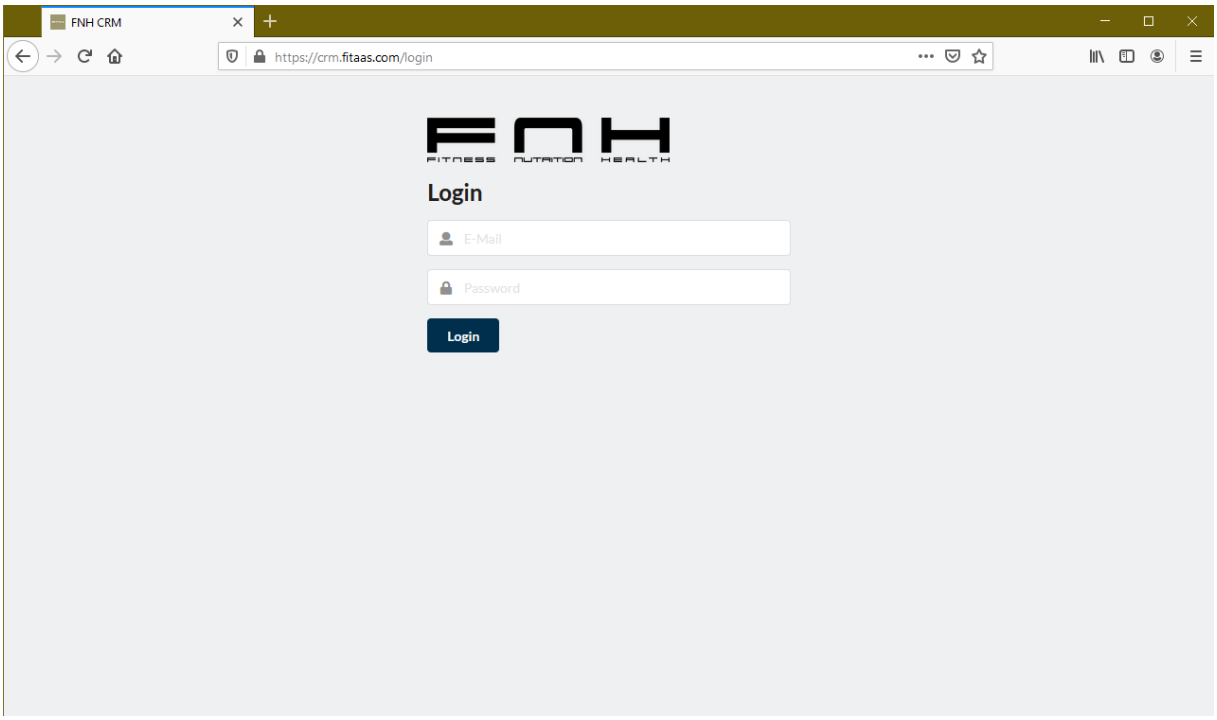

Figure 1 - Final application - Login screen

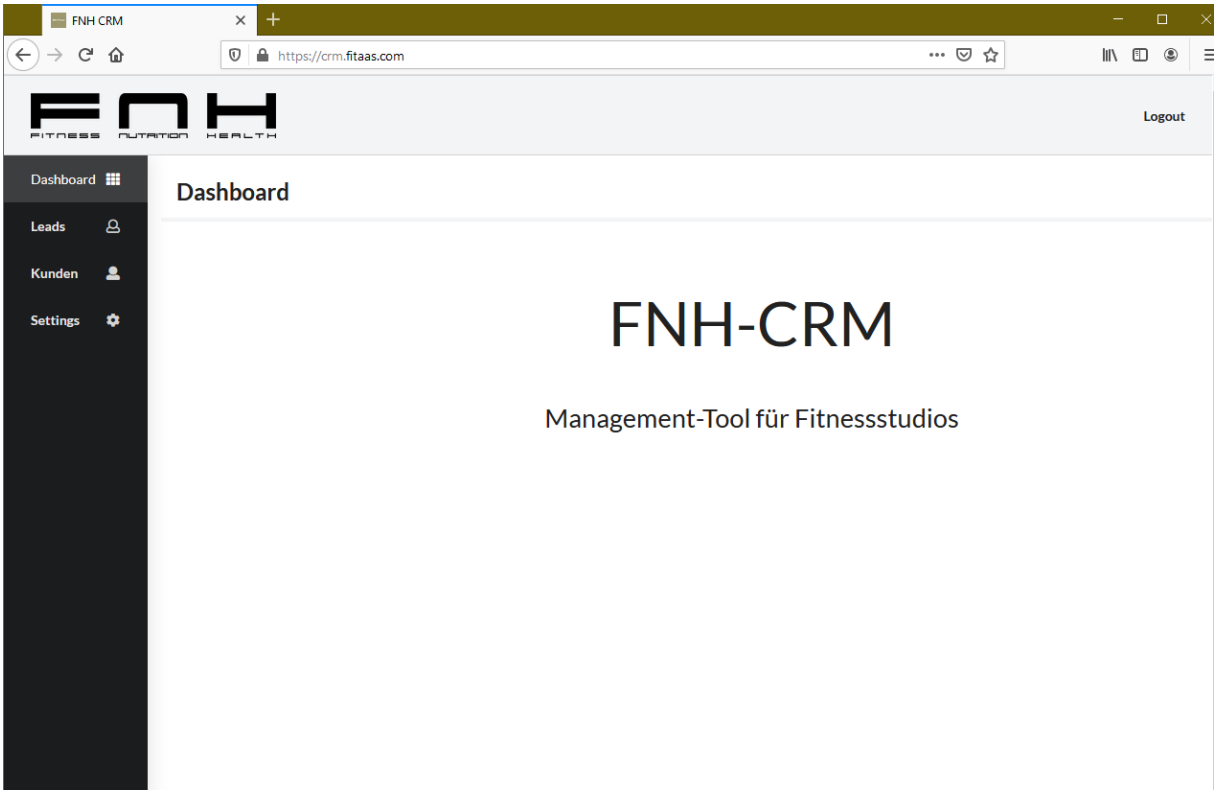

Figure 2 - Final application - Dashboard

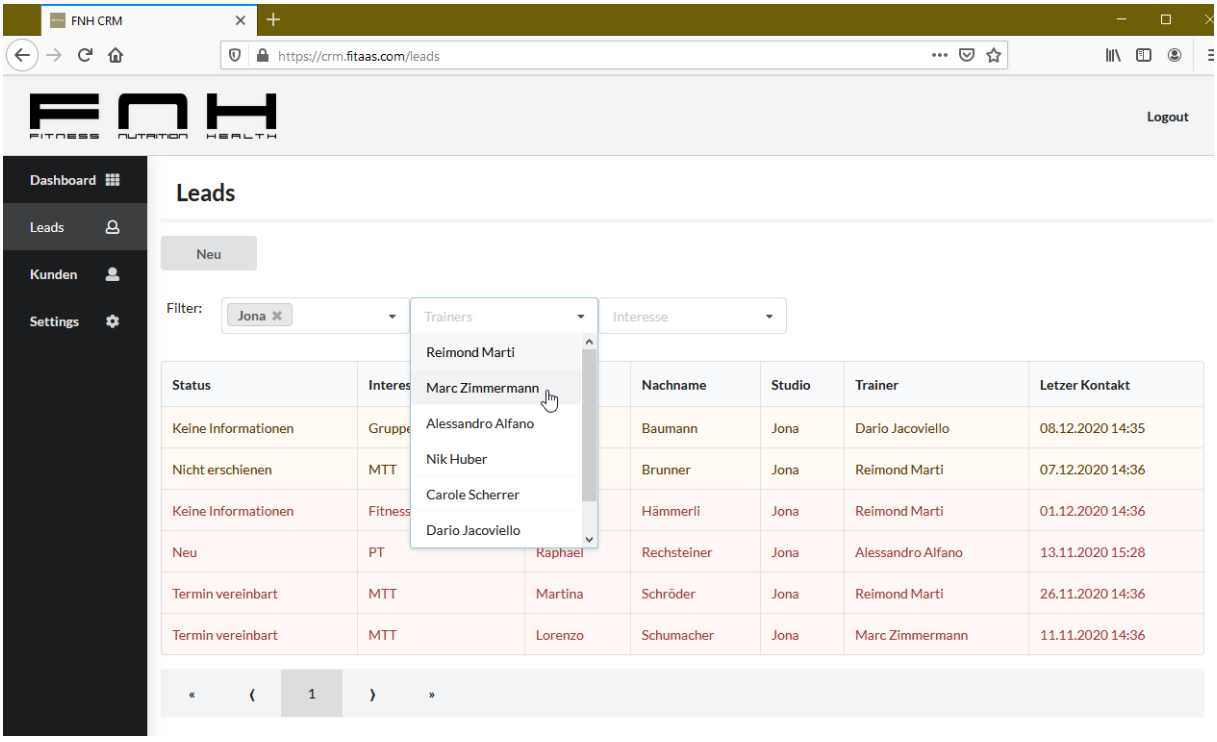

Figure 3 - Final application - Lead screen - Overview

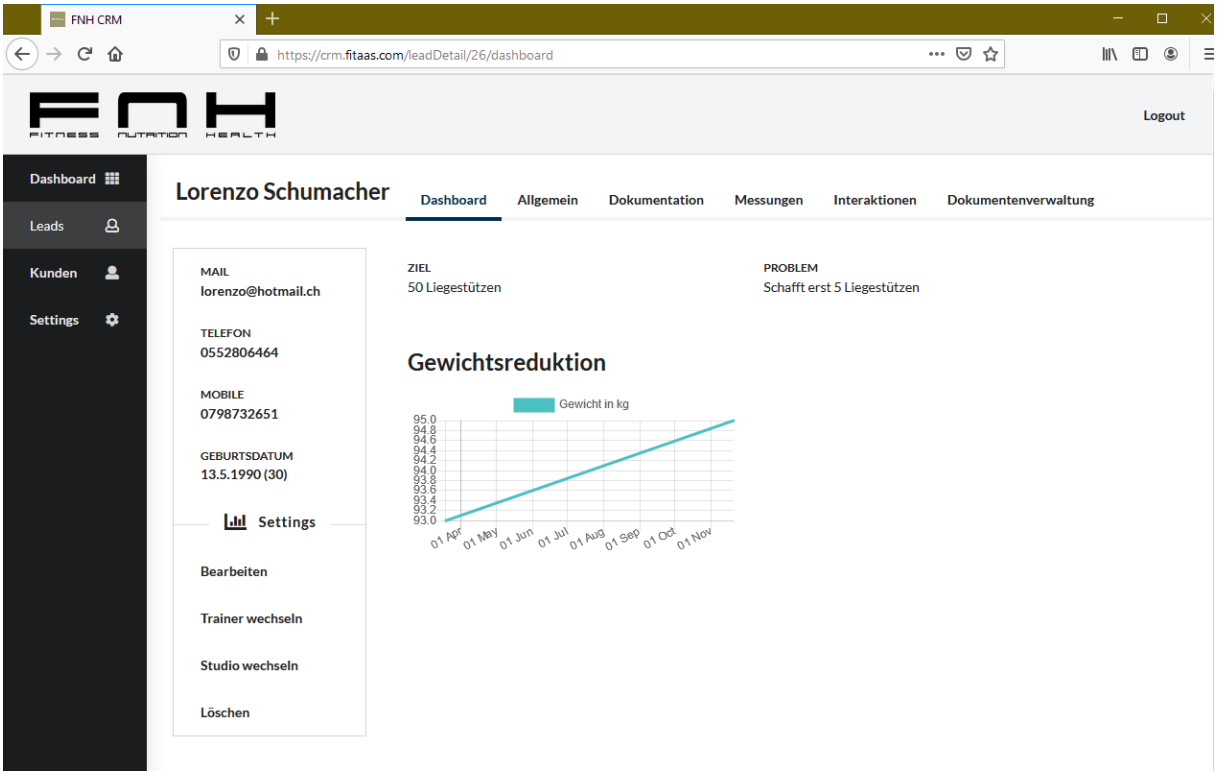

Figure 4 - Final application - Lead screen - Detail view - Dashboard

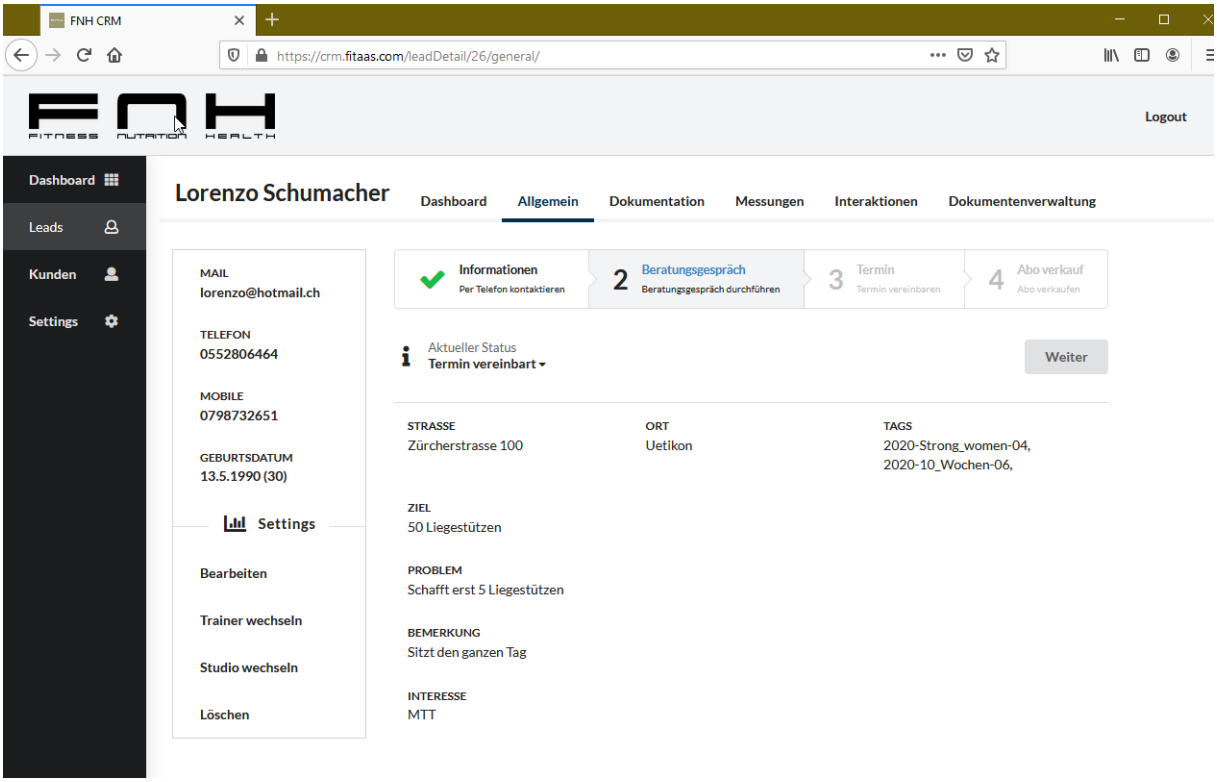

Figure 5 - Final application - Lead screen - Detail view - General

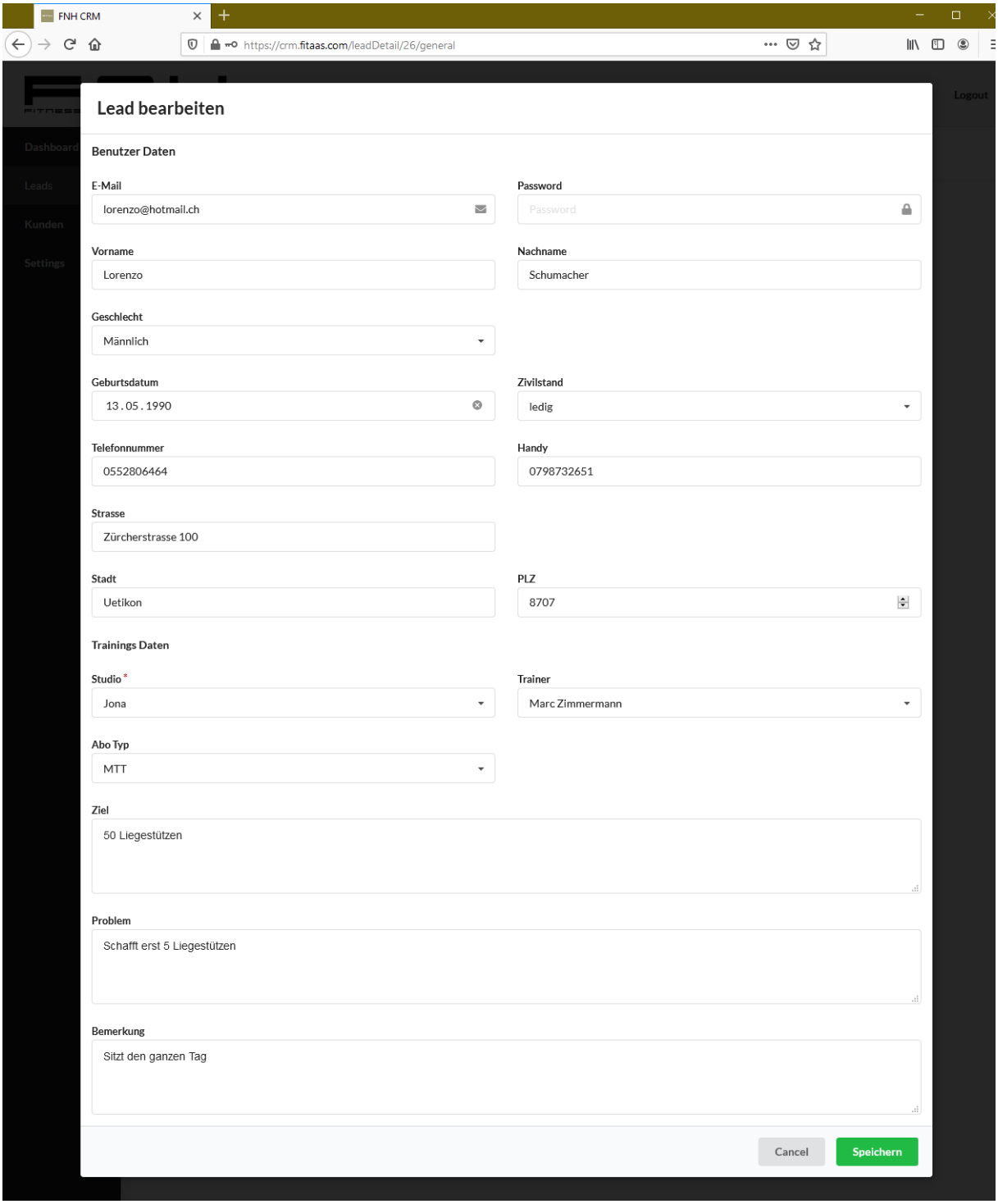

Figure 6 - Final application - Lead screen - Detail view – Edit lead

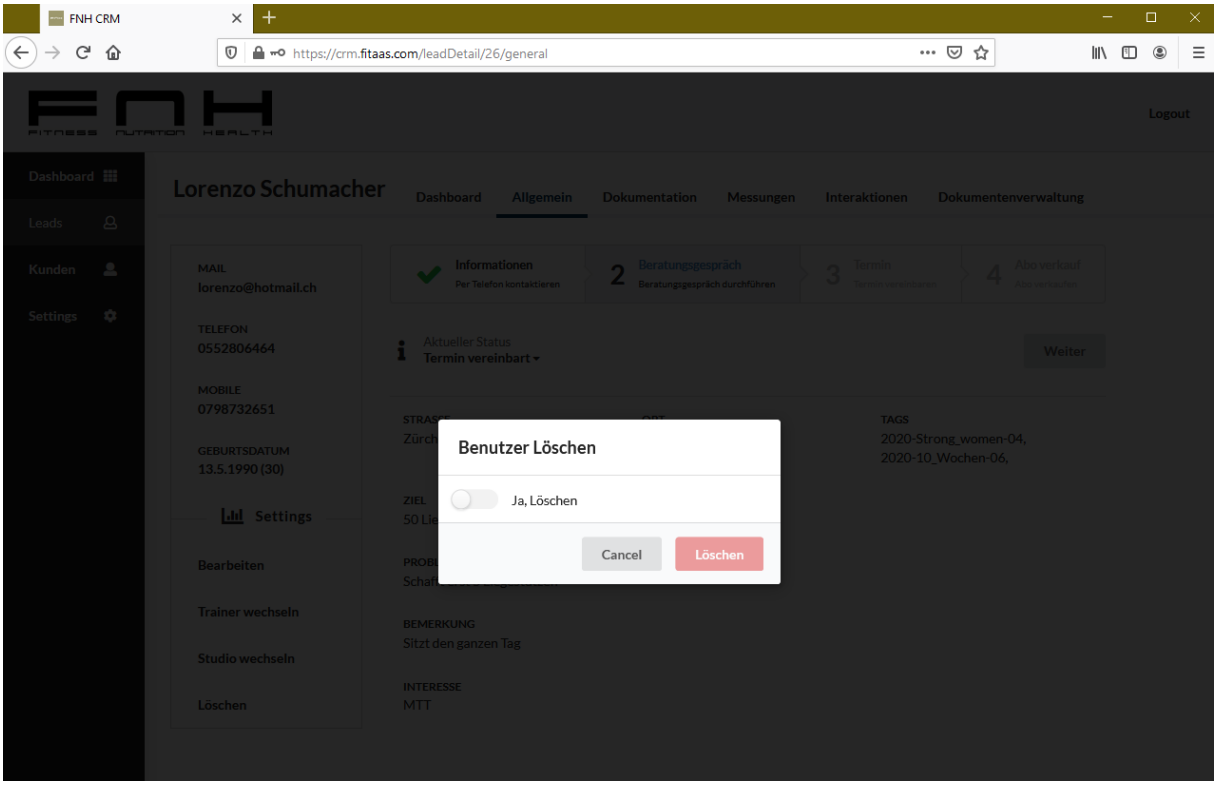

Figure 7 - Final application - Lead screen - Detail view – Delete user

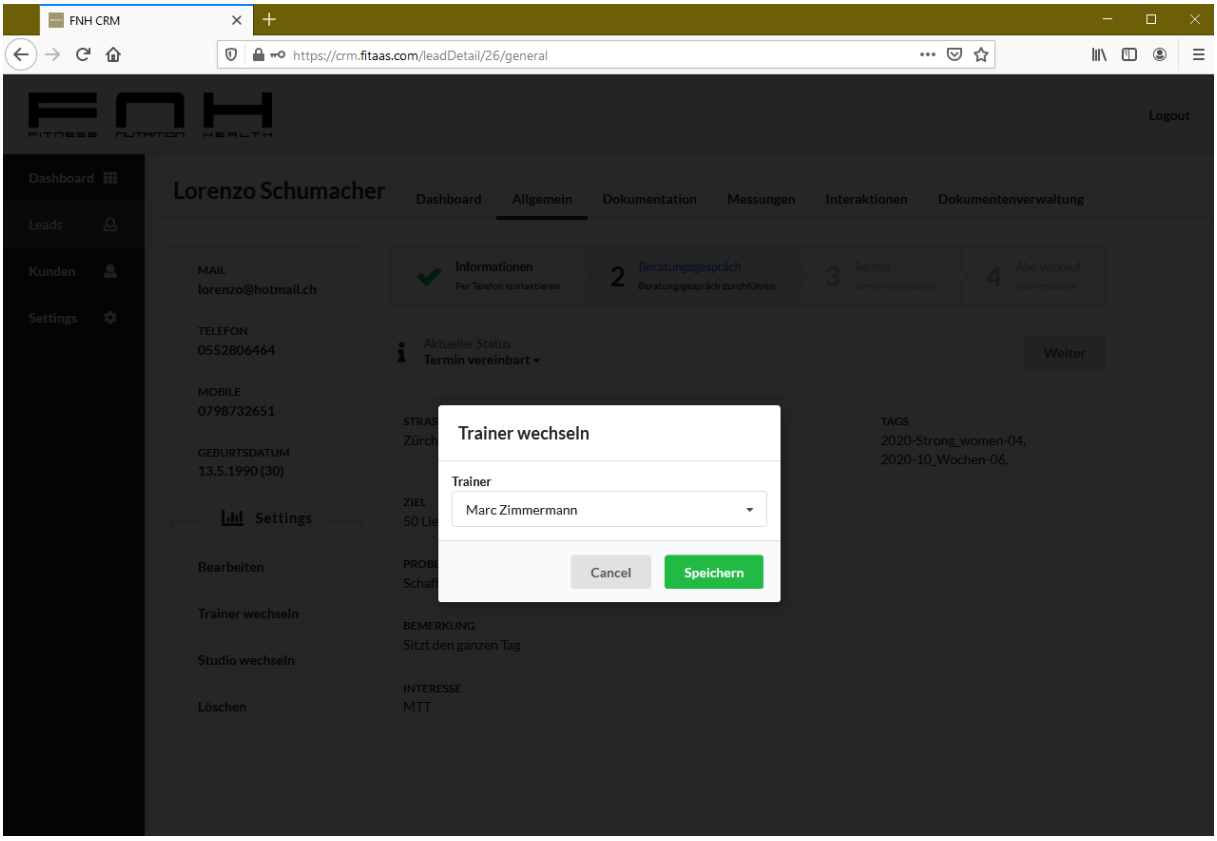

Figure 8 - Final application - Lead screen - Detail view – Trainer change

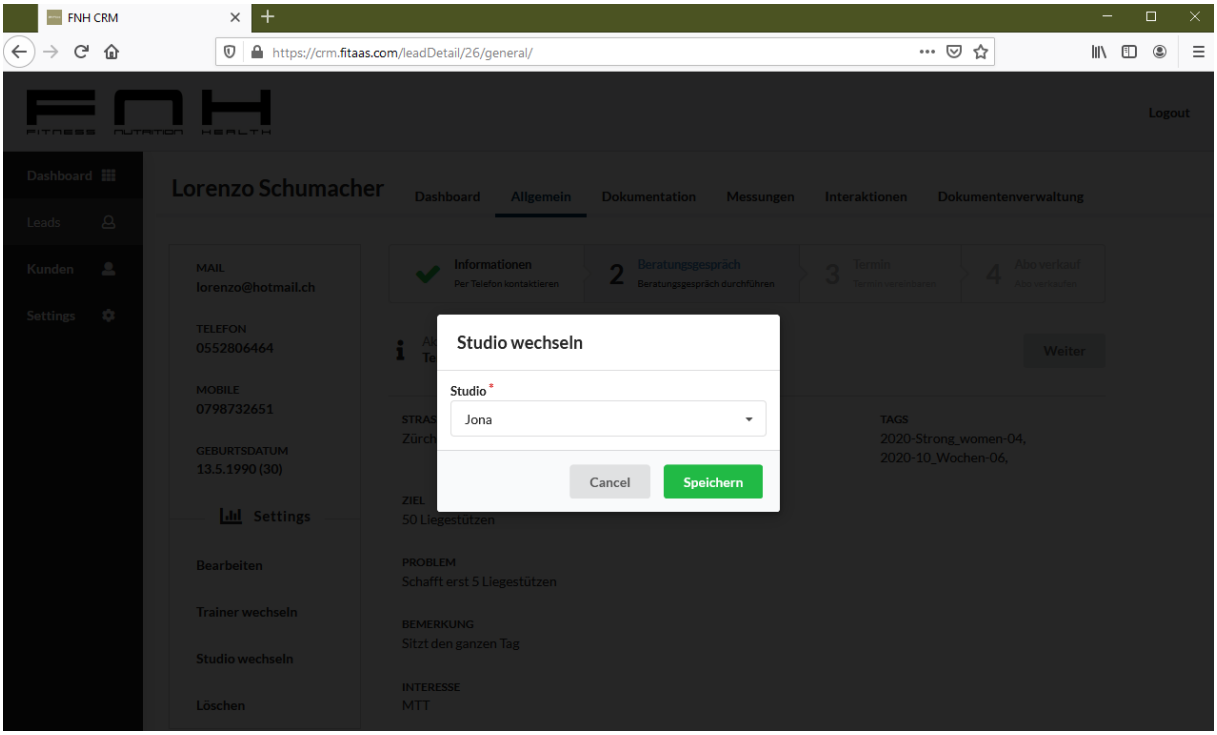

Figure 9 - Final application - Lead screen - Detail view – Studio change

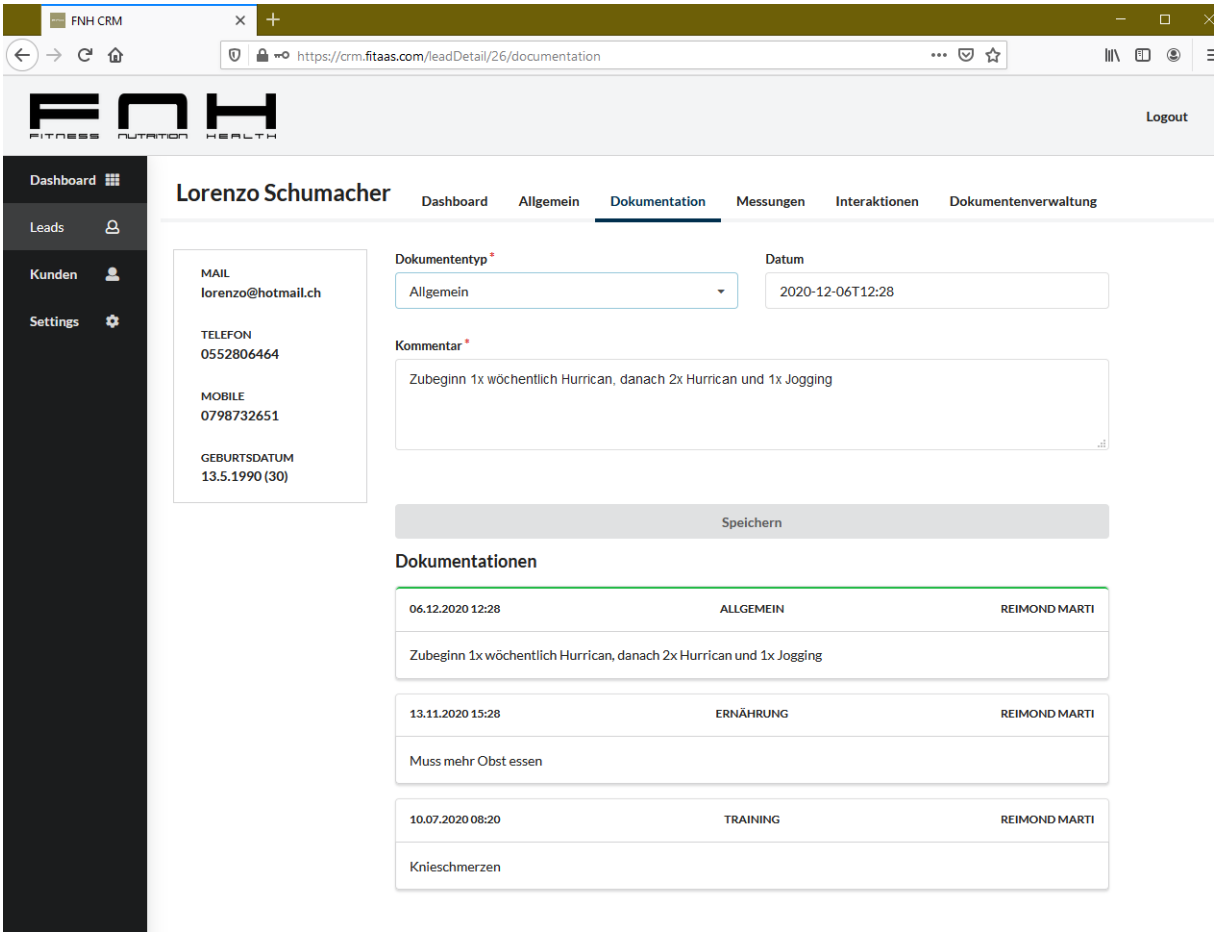

Figure 10 - Final application - Lead screen - Detail view - Documentation

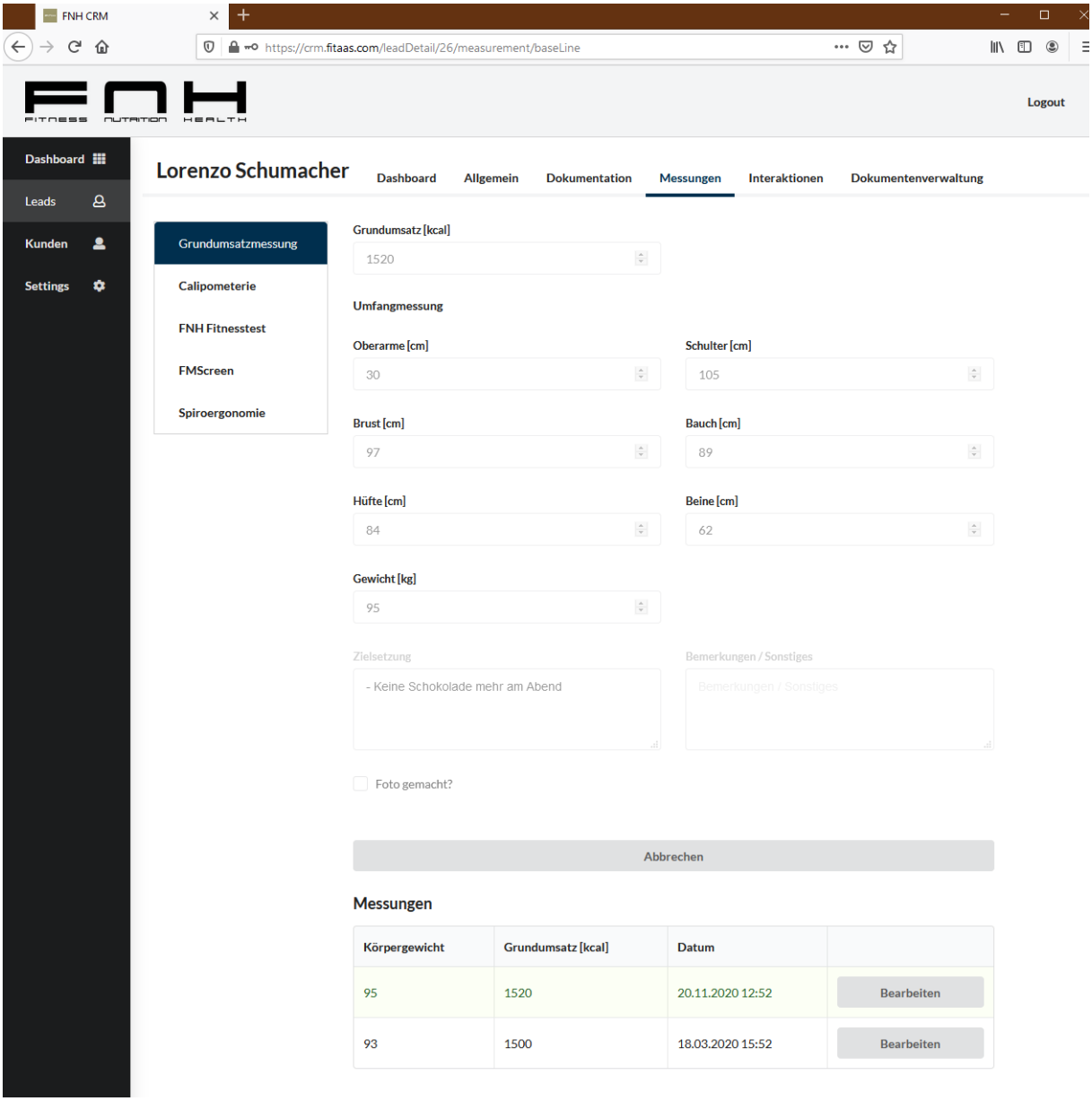

Figure 11 - Final application - Lead screen - Detail view - Measurement - Basic turnover

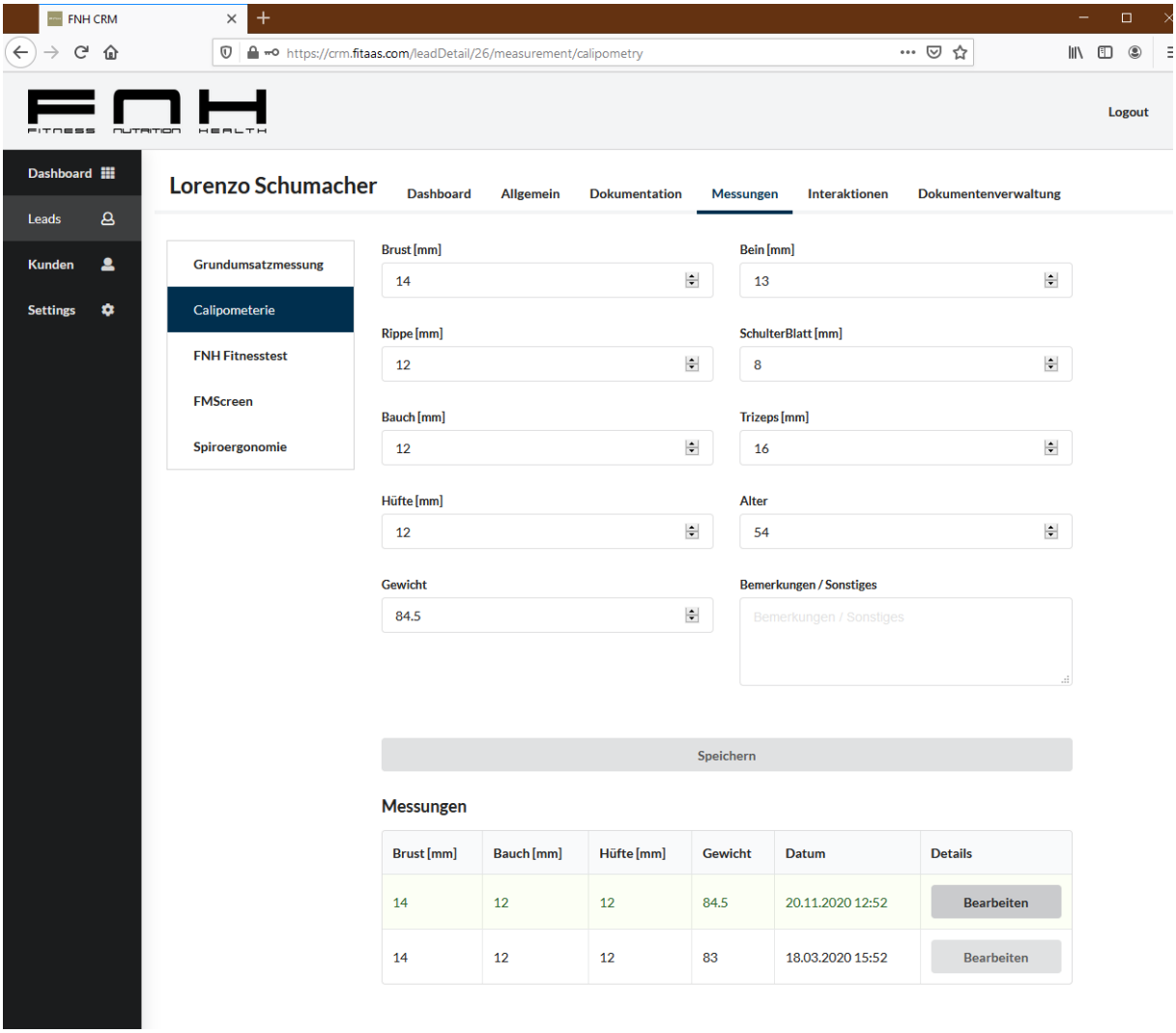

Figure 12- Final application - Lead screen - Detail view - Measurement - Calipometry

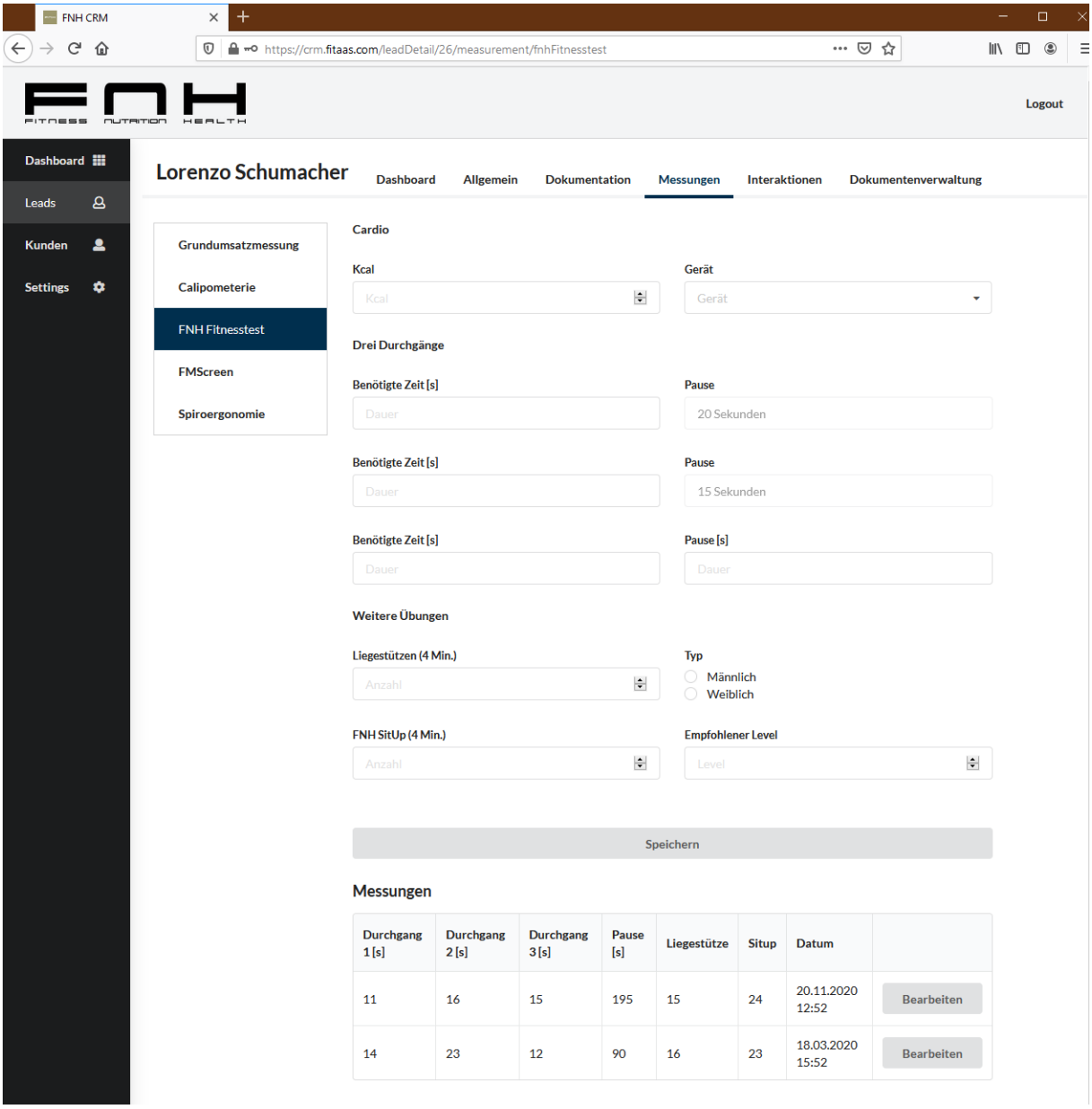

Figure 13 - Final application - Lead screen - Detail view - Measurement - Fitness test

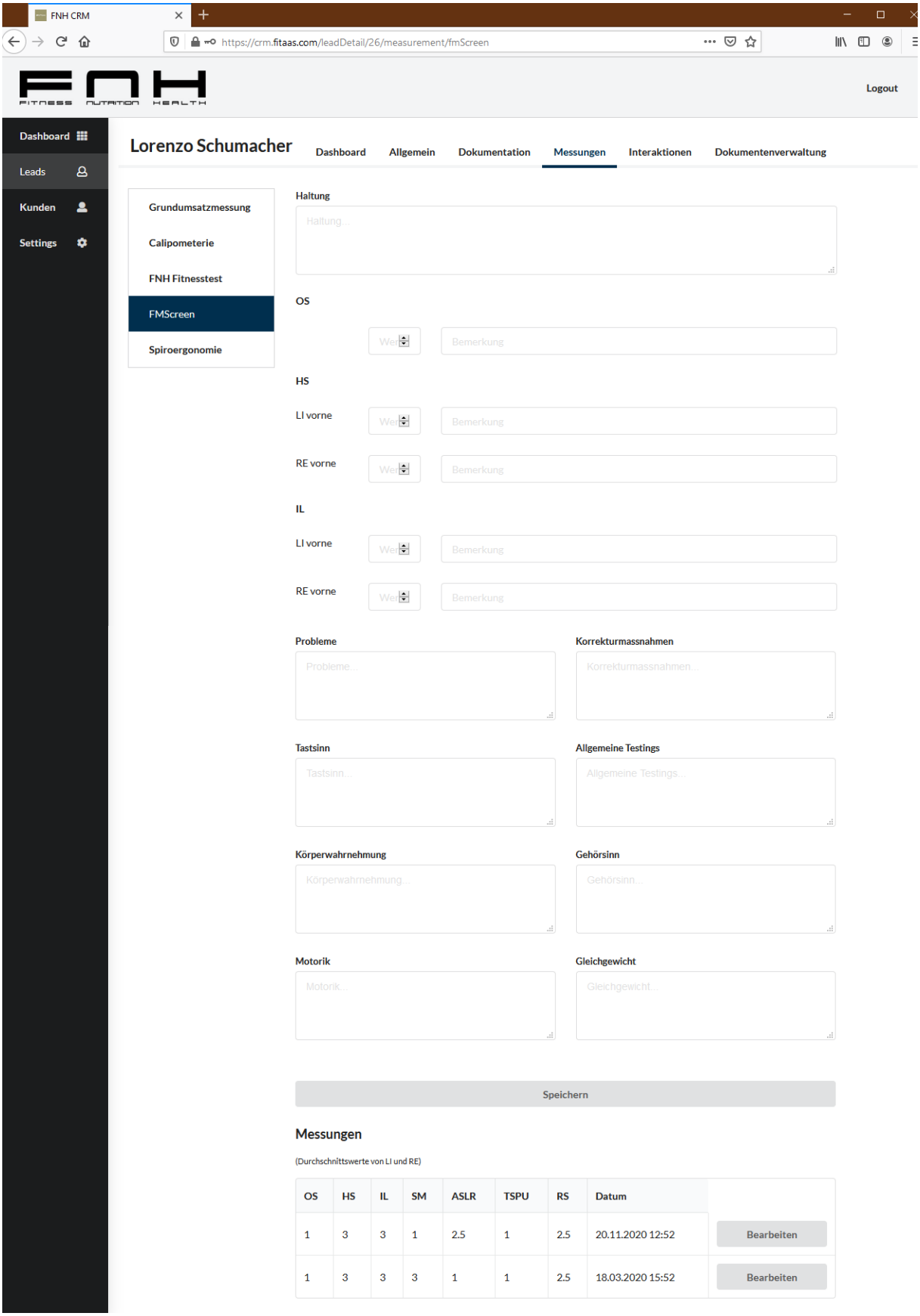

Figure 14 - Final application - Lead screen - Detail view - Measurement - FMScreen

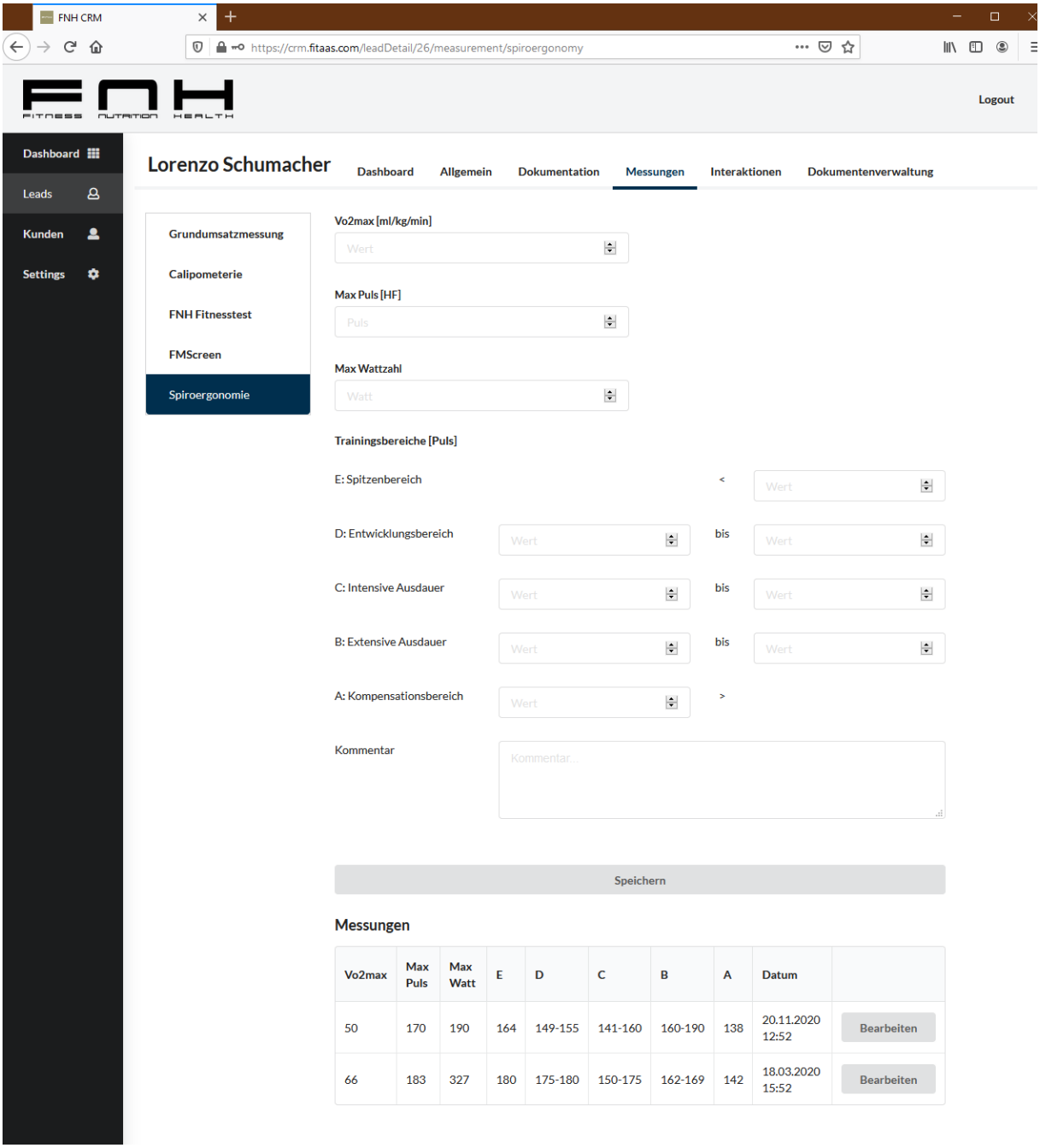

Figure 15 - Final application - Lead screen - Detail view - Measurement - Spiroergonomy

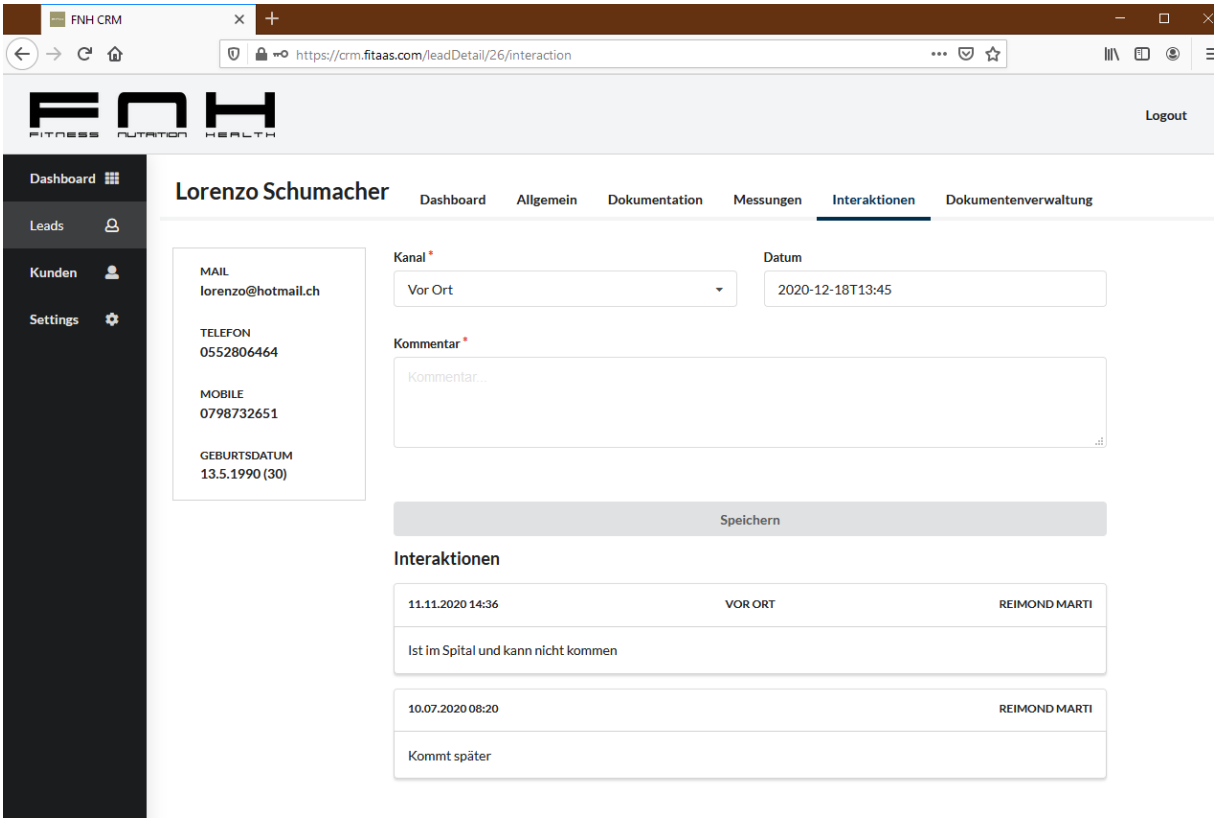

Figure 16 - Final application - Lead screen - Detail view - Interactions

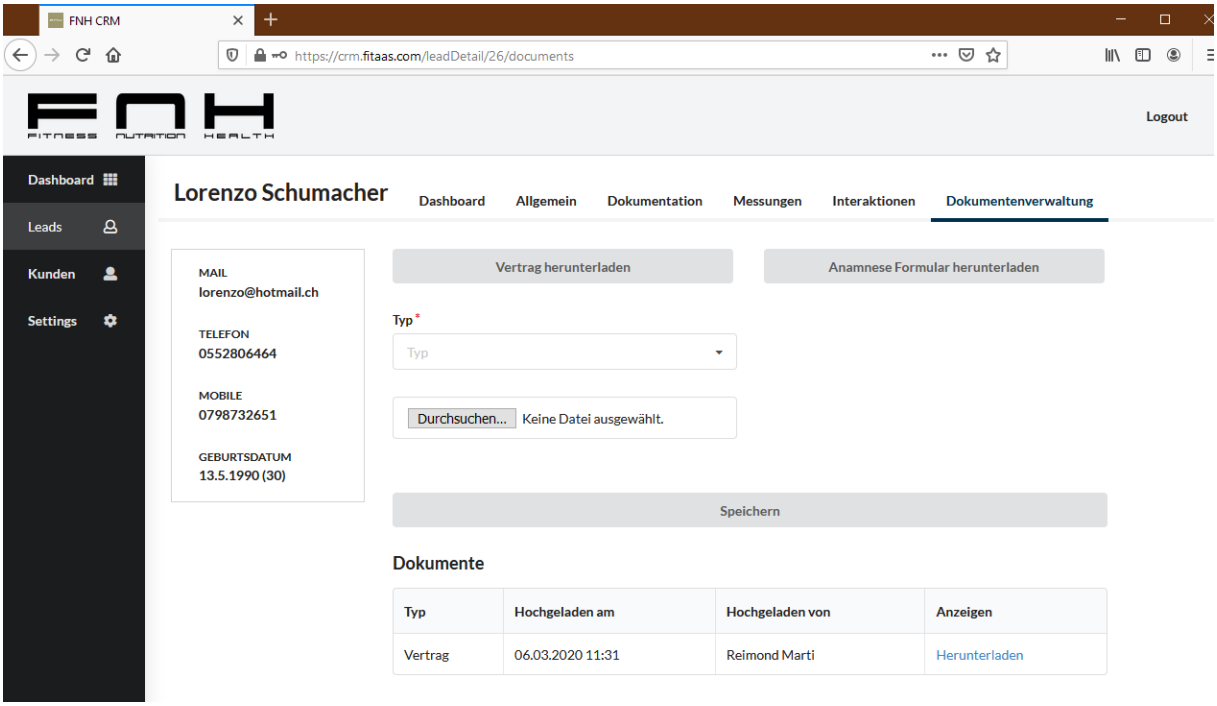

Figure 17 - Final application - Lead screen - Detail view - Documents

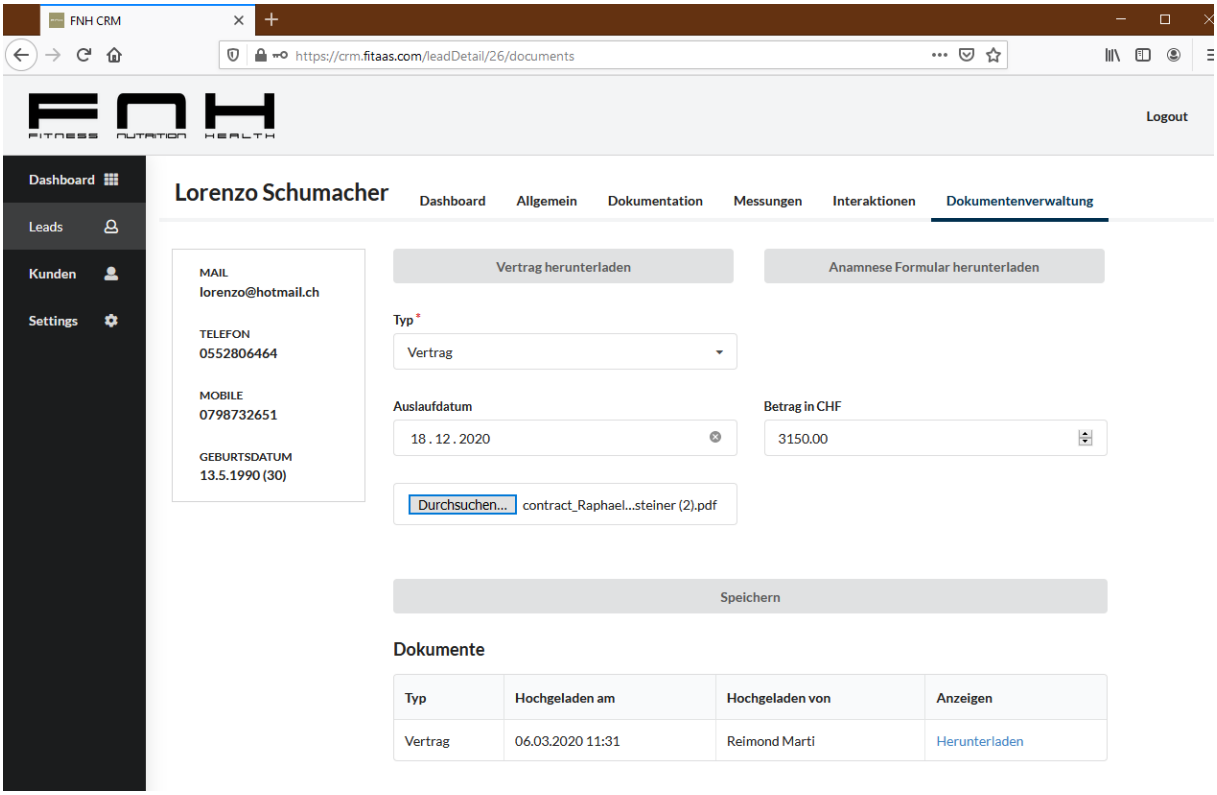

Figure 18 - Final application - Lead screen - Detail view - Documents - Contract

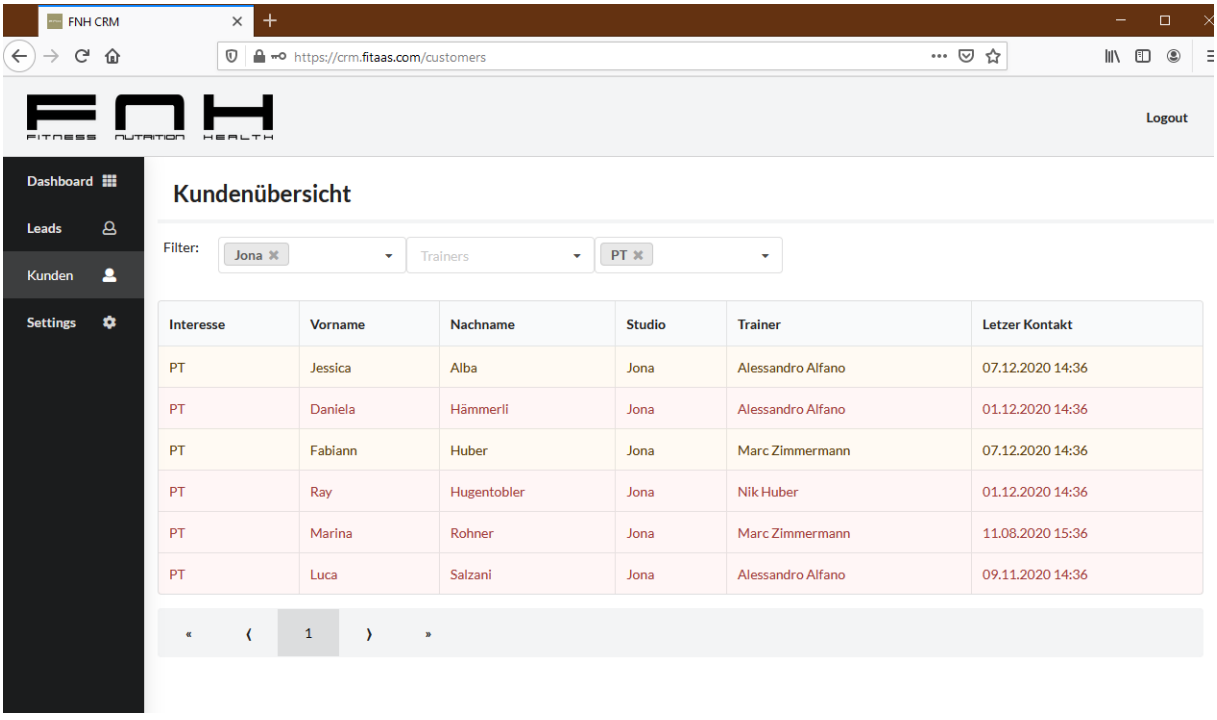

Figure 19 - Final application - Customer screen - Overview

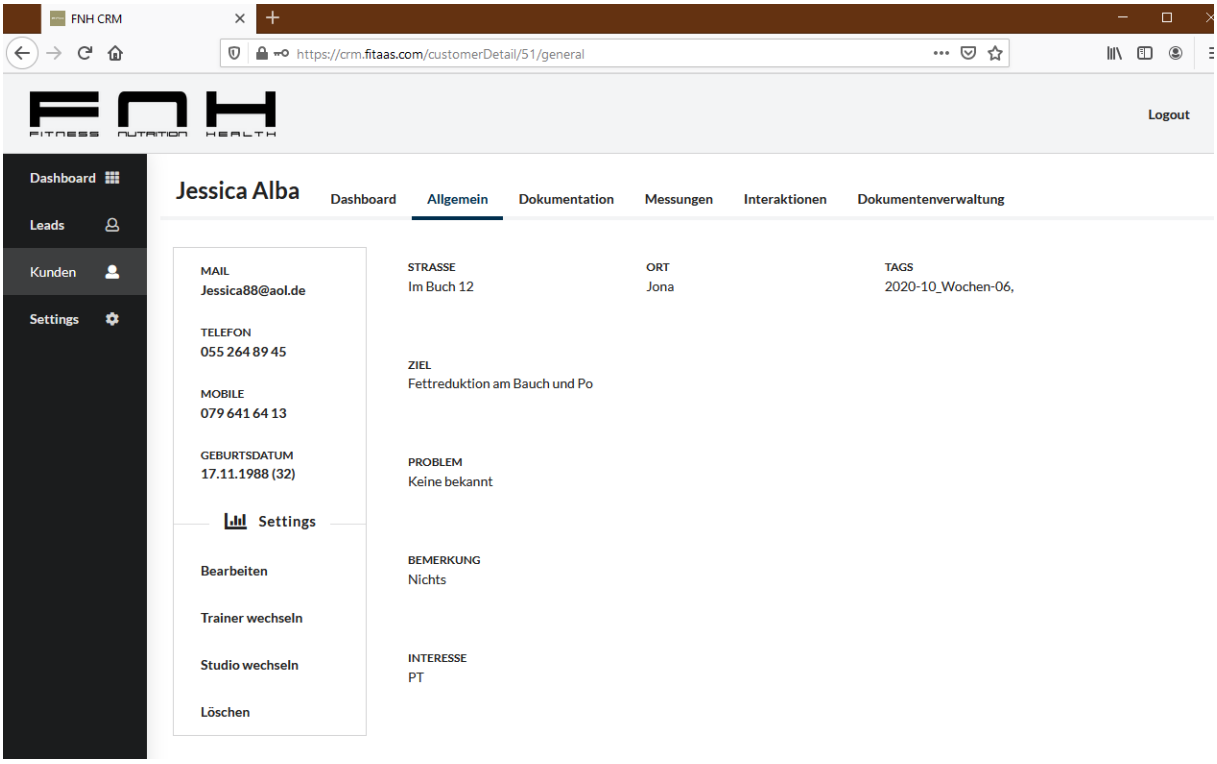

Figure 20 - Final application - Customer screen - Detail view - General

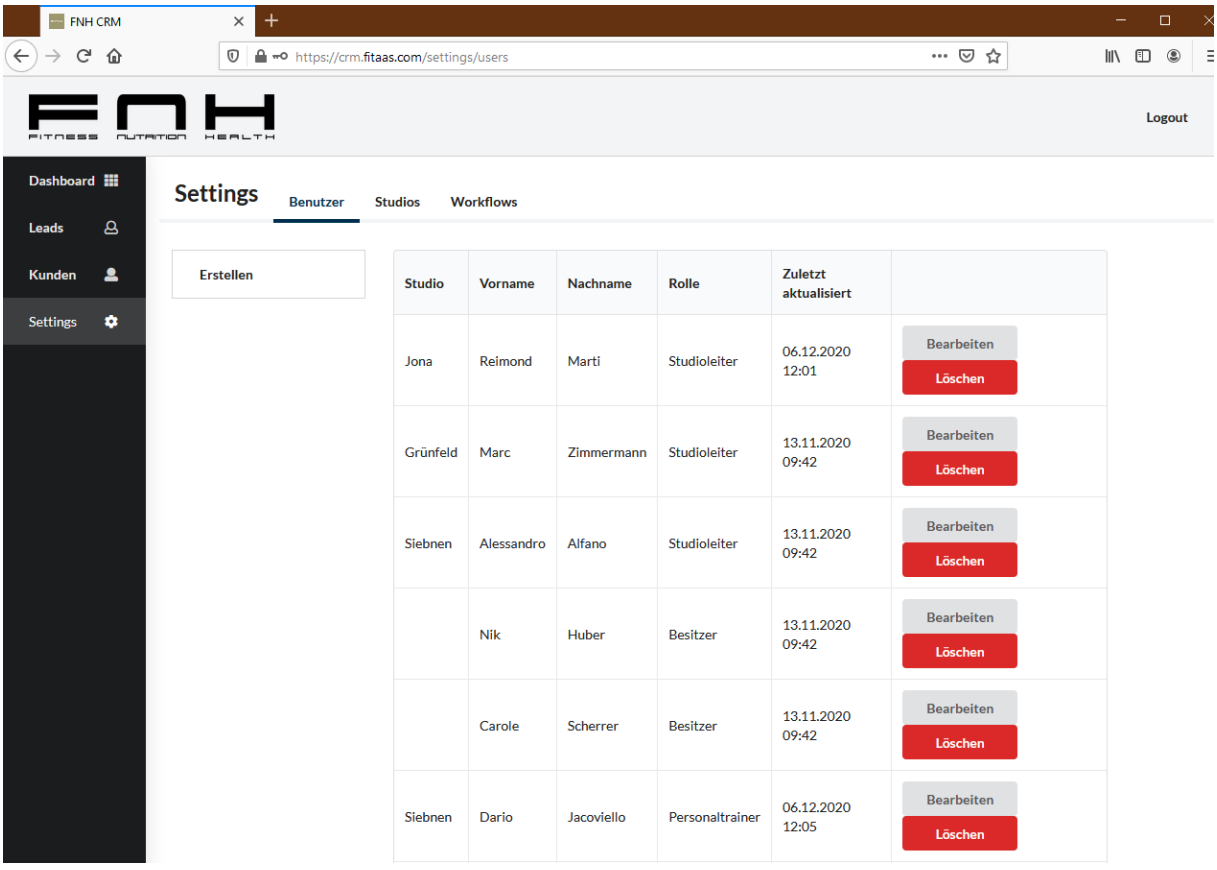

Figure 21 - Final application - Settings screen - User - Overview

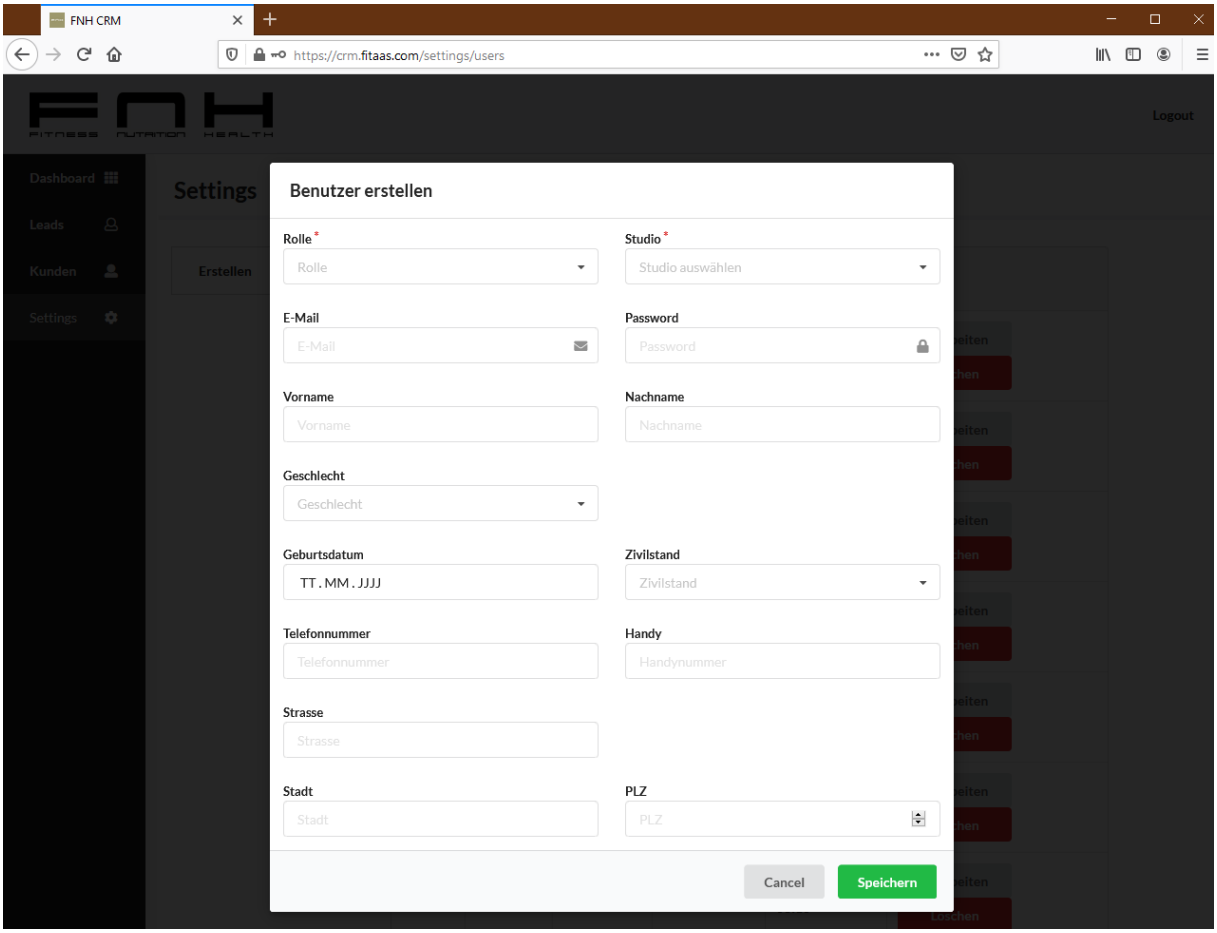

Figure 22 - Final application - Settings screen - User - Create

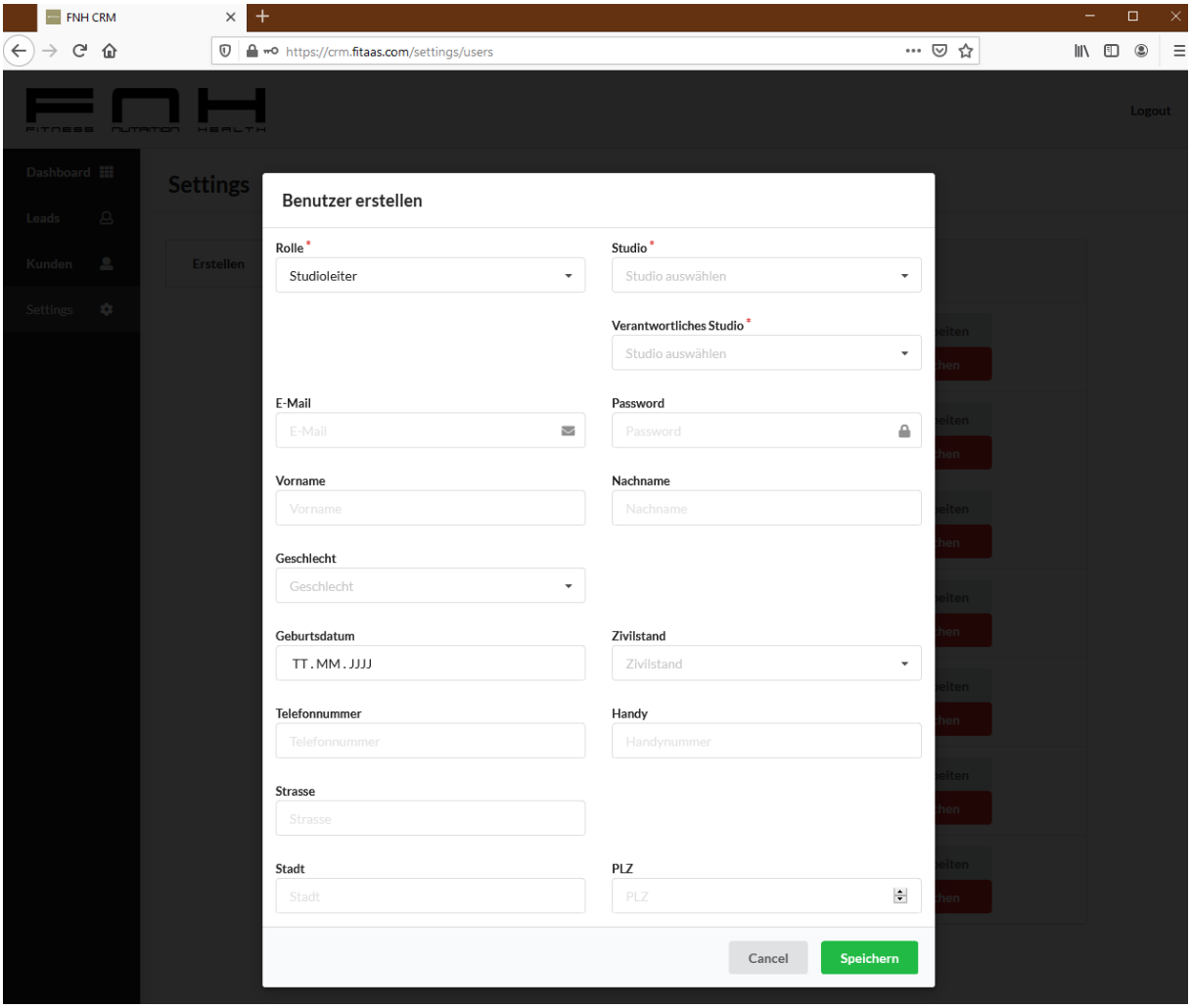

Figure 23 - Final application - Settings screen - User - Create - Chief personal trainer

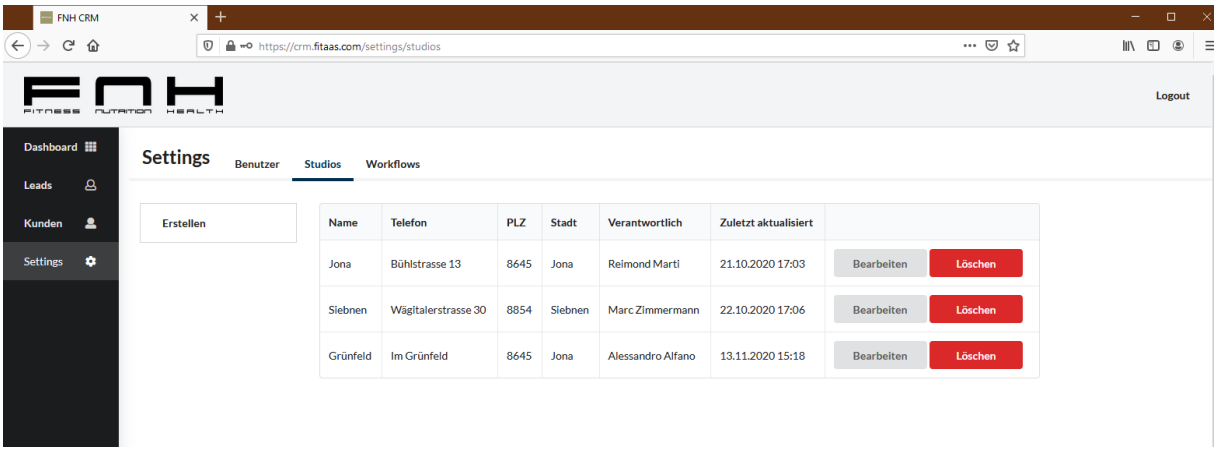

Figure 24 - Final application - Settings screen - Studios - Overview

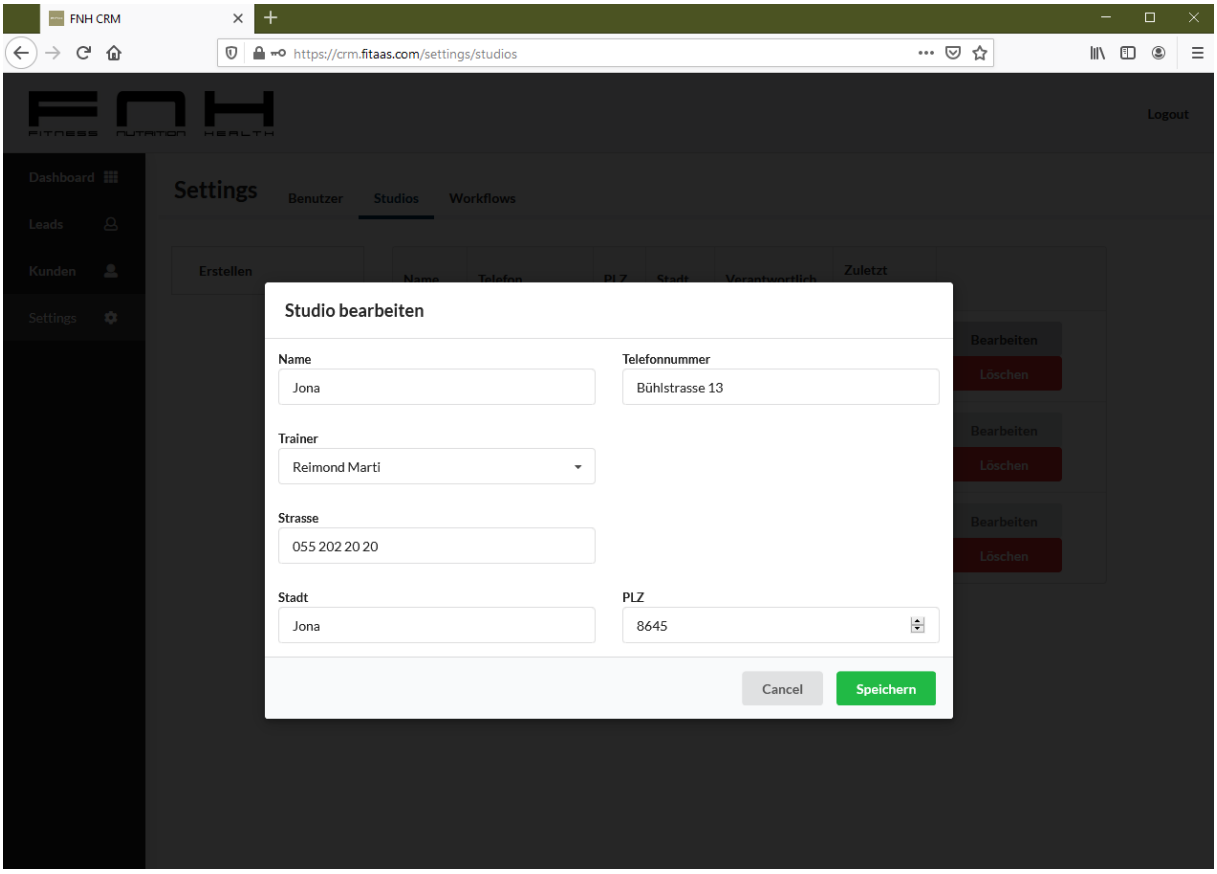

Figure 25 - Final application - Settings screen - Studios - Edit

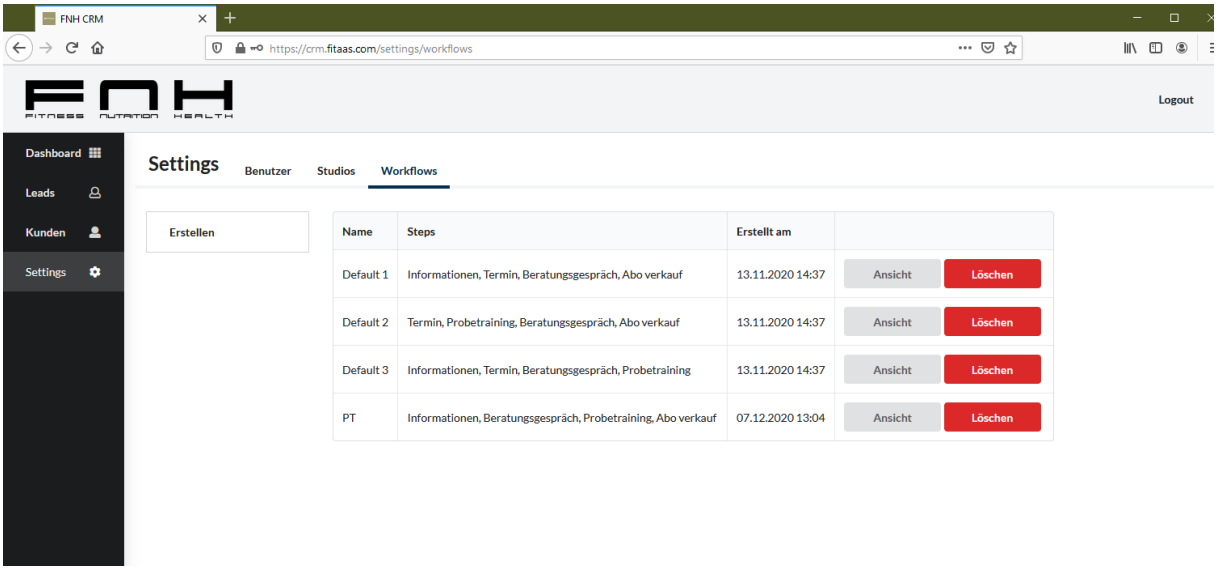

Figure 26 - Final application - Settings screen - Workflows - Overview

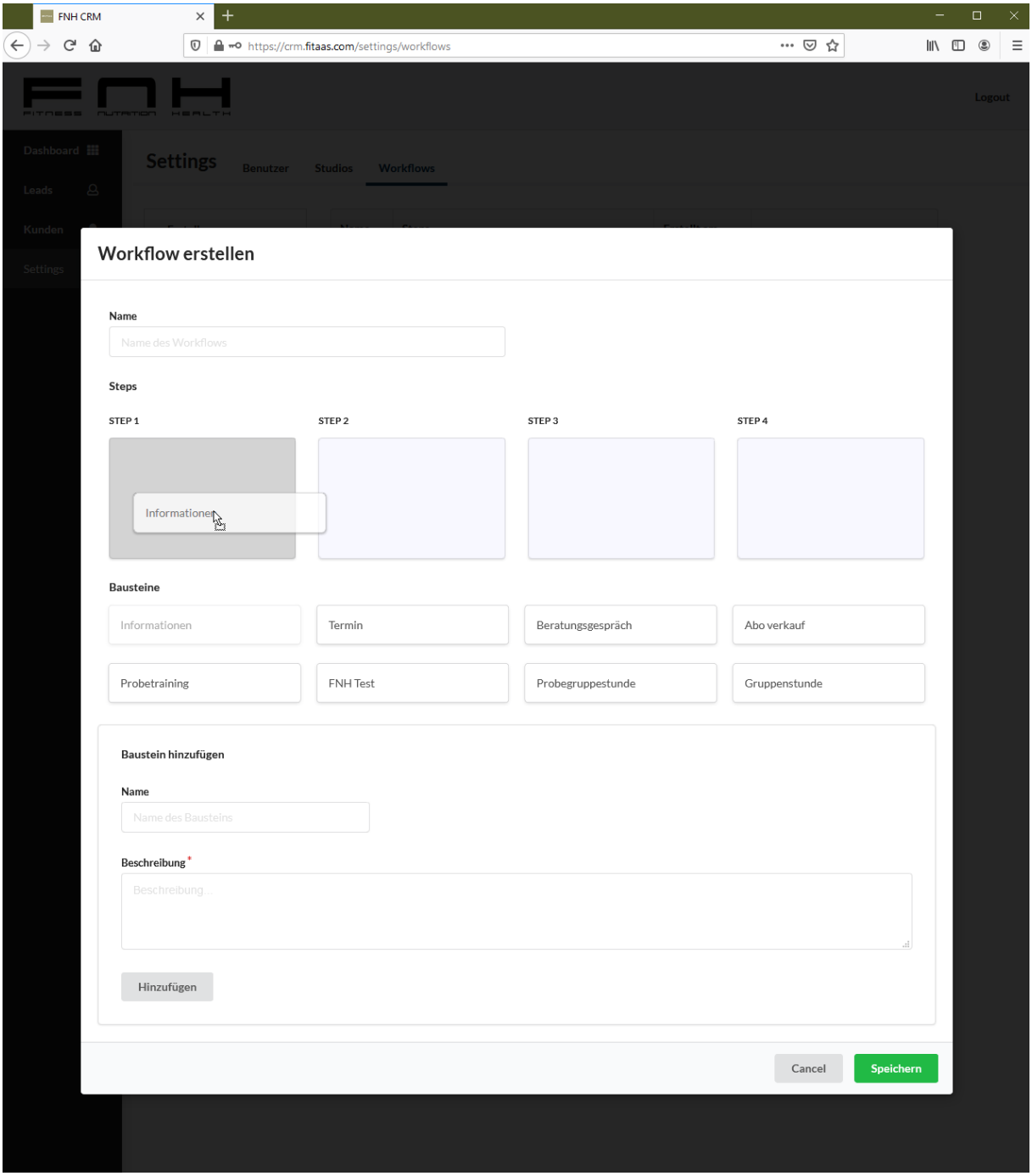

Figure 27 - Final application - Settings screen - Workflows - Create - Drag building block

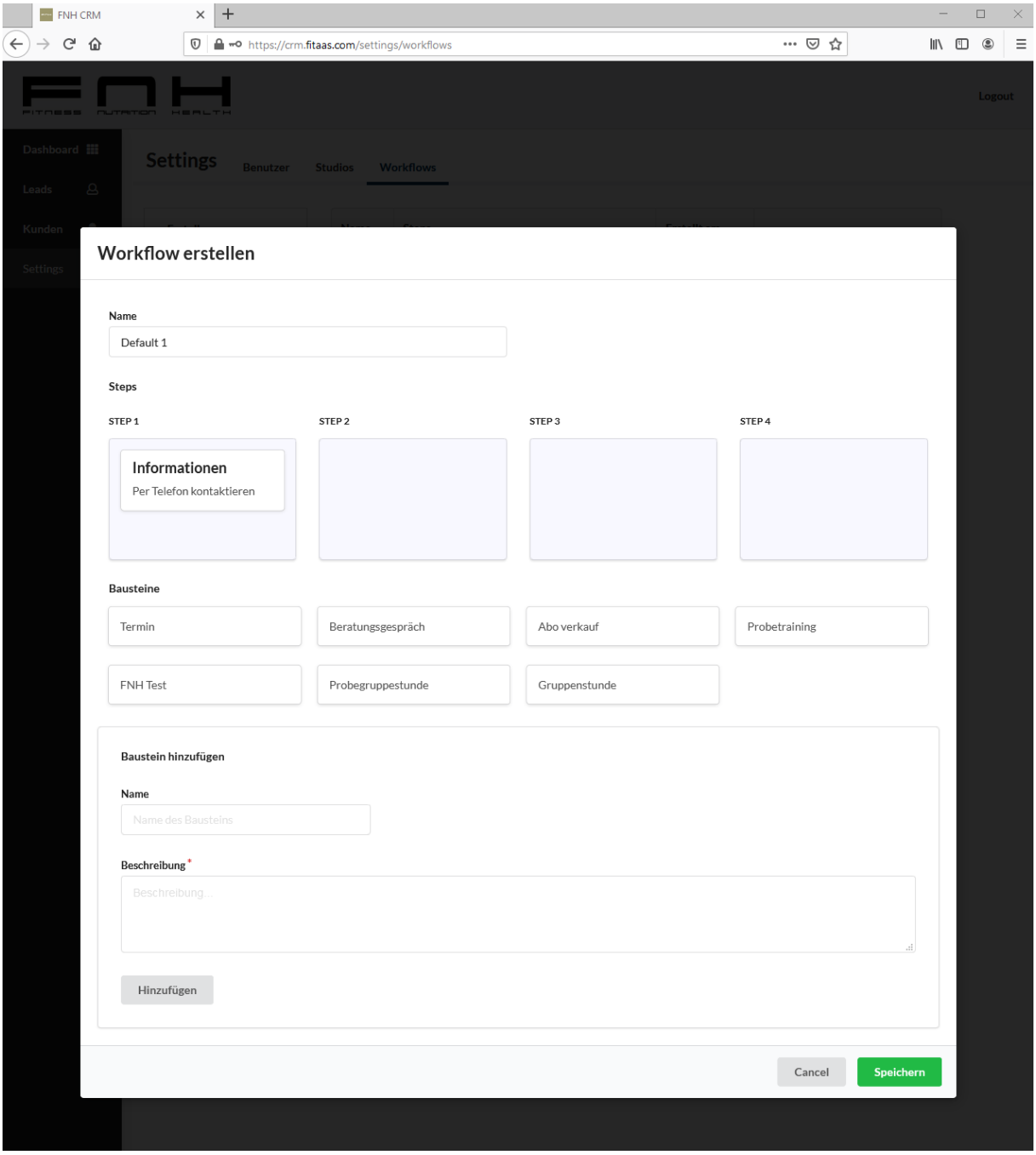

Figure 28 - Final application - Settings screen - Workflows - Create - Drop building block

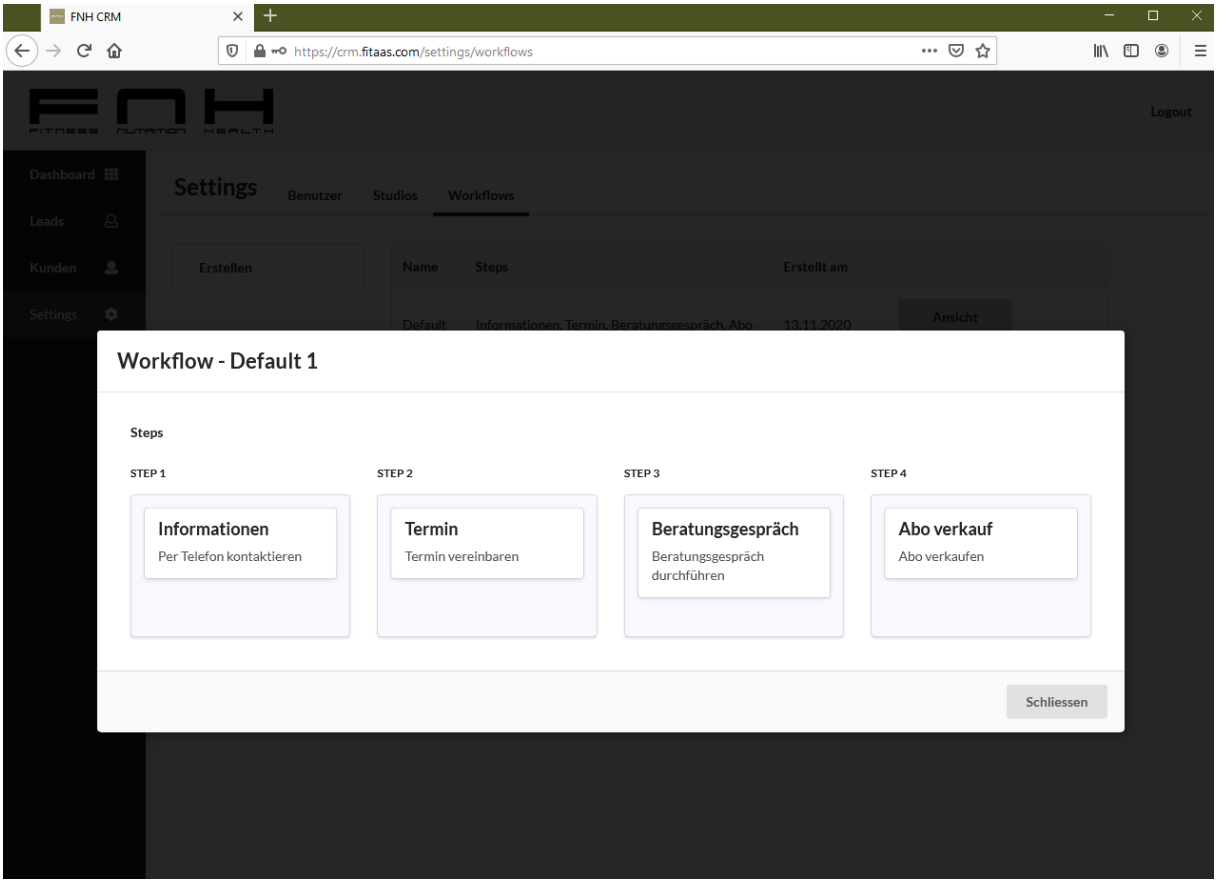

Figure 29 - Final application - Settings screen - Workflows - View

G. Optional features

This is a list of all the optional features with a short description.

OF1 – Archive customer / lead
When a customer cancels his subscription, he might not be gone forever. If he comes back before a set amount of time, his past training data and interactions still need to be around. To achieve this, a personal trainer can archive a user, when he is temporarily unsubscribed.

OF2 – CRUD products
The various products of a fitness studio can be created, edited and deleted by the owner. This includes for example the different subscriptions or the personal trainings.

OF3 – Show contracts that expire soon
When a customer's contract is about to expire or has expired, this should be clearly indicated so that the personal trainer knows he should contact the customer.

OF4 – Contact customer
The personal trainer sees in the customer interaction list, that a customer needs to be contacted. With the click of a button he can open the communication directly from the application.

OF5 – Bring a friend / brought by personal trainer
When a customer proposes a friend to join up with the studio, his friend is registered as a lead. To make this attractive, if such a lead can be converted to a customer, there is a reward for the customer who brought him in. The administrator can set these rewards in the application. When a lead is brought in through the "Bring a friend" program, the initiating customer is registered. If the process succeeds and the lead joins up, the reward is automatically assigned.

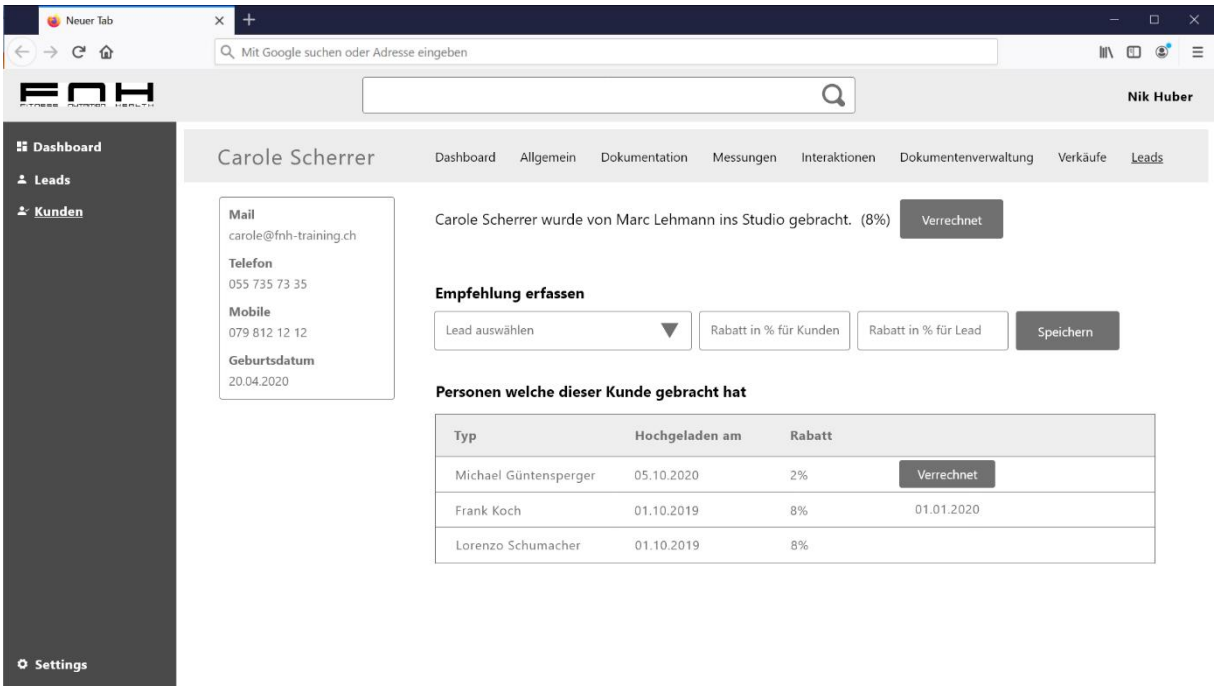

Figure 1 - Mockup - Customer screen - Detail view - Leads

OF6 – CRUD In-studio purchases
The additionally sold products such as T-shirts, protein powder, etc. in a fitness studio must be manageable. This also includes the inventory management.

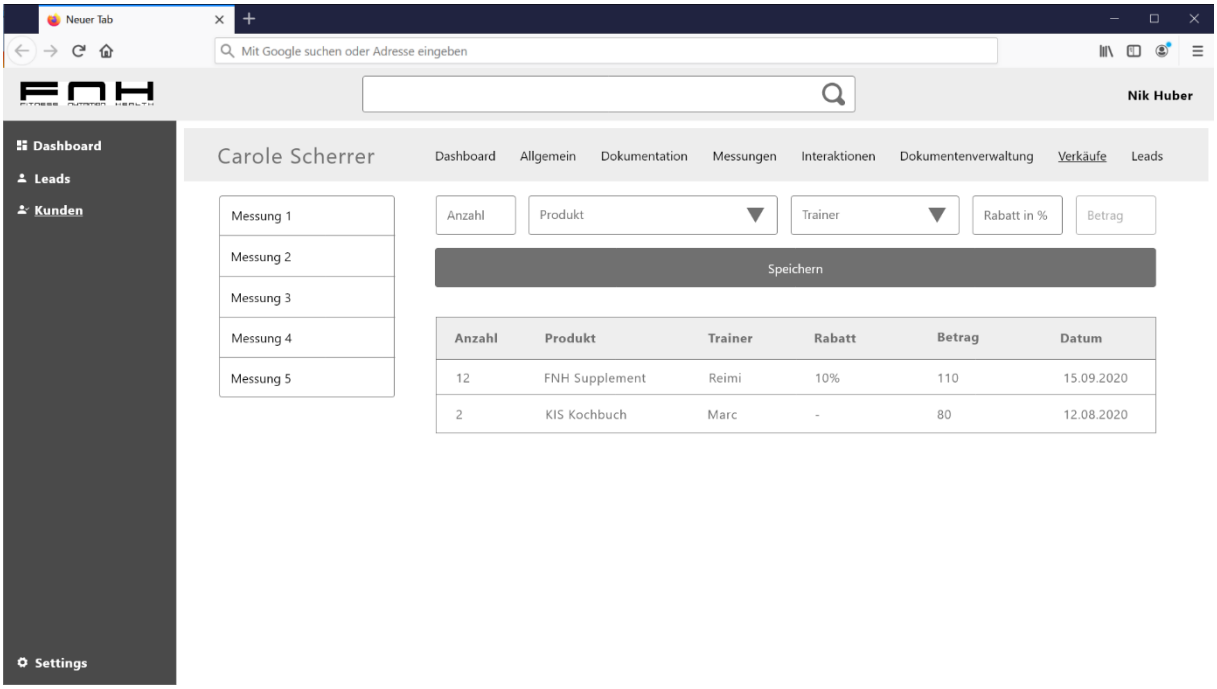

Figure 2 - Mockup - Customer screen - Detail view - Sales

OF7 – View customer billing status
The personal trainer can check if a customer has paid his invoice, or if he has any outstanding rates.

OF8 – Personal trainer dashboard
Every personal trainer has his own dashboard with the most important information for the current workday. This includes e.g. the contracts that expire (OF3), birthdays (OF25), checklists (OF10).

OF9 – Customer billing
The system automatically generates the bill for every user and sends it to their inbox including the letter for the health insurance.

OF10 – CRUD checklist
For the personal trainers to have a clear todo-list, checklists can be created by the owner. These can be filled out by the personal trainers e.g. every day before finishing up to make sure all the tasks are completed.

OF11 – Additional workflow information
The personal trainer has the possibility to log additional information like resources and costs invested in a workflow step.

OF12 – Global search
The personal trainer has the possibility to search for customers or leads. This allows the personal trainer to get the data of a customer or lead faster.

OF13 – Measurement external devices
New screen to edit incoming measurement from external sources to e.g. add weight to the customer automatically through a connected scale.
In the future, there will be new machines purchased to do measurements with the studio clients. These machines will be able to send the data to other services. This way, some measurements can be recorded automatically. Because the backend has a public API, it will be possible to automatically create measurements. The problem is that there is no customer assigned to the data at this point. Therefore, a collection basin for measurements would be created, where the measurements would have to be assigned manually to a customer.

OF14 – Visual customer progress
Measurement to add images of customers. Is used to show a customer's progress visually.

OF15 – Add note as PDF
Attach handwritten notes and drawings as PDF to a client under the submenu "Notes". This enables to save notes that cannot be specifically assigned.

OF16 – Consultant form
Consultation as form to fill out instead of a pdf document that has to be uploaded and can not be analysed.

OF17 – Contract as webform
The personal trainer can enter the contract information online and sign it directly on the website without downloading and uploading as pdf manually.

OF18 – Profile picture
The personal trainer can add a profile picture to a customer/lead. This makes it easier for the personal trainer to recognize the customer.

OF19 – Fitness-device settings
Store a customer's device settings like e.g. handlebar height for the measurement and display them at the specific measurement. Dropdown with training device, different fields for different devices.

OF20 – Timestop a subscription
The personal trainer can stop the subscription for a certain time span, if for example, a client is ill or injured for a longer period.

OF21 – Timer in "FNH Fitnesstest"
A timer is to be built in to automatically record the exercise duration and the breaks between the exercises. This allows the personal trainer to concentrate on the customer. (Especially helpful on the "FNH Fitnesstest")

OF22 – Automatically generate chief personal trainer
Automatically convert to chief personal trainer if responsible for a studio is assigned.

OF23 – Door opening system
Only allow active customers to access the studio and track the entries. Every entry creates an interaction. For the interactions not to become cluttered, tabs to separate the different interactions will be added.

OF24 – Information button on measurements
Add a button that displays information on how to perform a measurement when clicked on for every measurement.
OF 25– Show a user's birthday
Display a user's birthday, so that the personal trainer can congratulate him.

OF 26– Body fat calculator on calipometry measurement
Usually, the body fat is being calculated with a website after the calipometry measurement is finished. Because all the data is already in the system, this could automatically be calculated.

OF27 – Lead information when converted to customer
Lead is being informed when converted to a customer. This could e.g. be used to send him login data for an appointment booking app.

OF28 - Collection basin leads
Add new leads to an extra list, where the missing information (e.g. responsible personal trainer and workflow) can be set, before they enter the list with all the leads.

OF29 – Mailchimp connection
Connect Mailchimp, to manage the Newsletter/info mail recipients in the CRM as well.

H. Diagrams

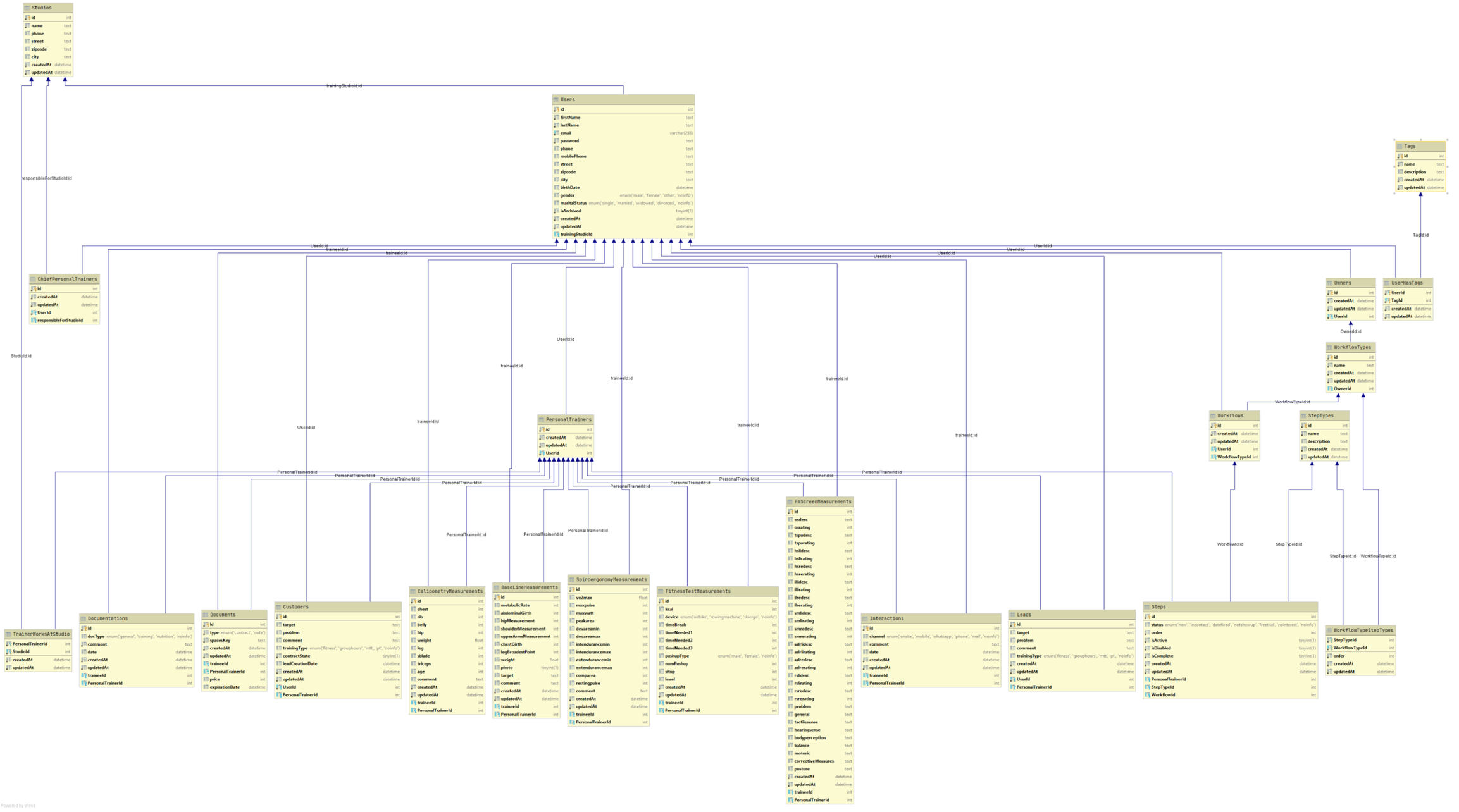

Figure 1 – Domainmodel generated

I. User documentation

FNH-CRM

Management tool for fitness studios

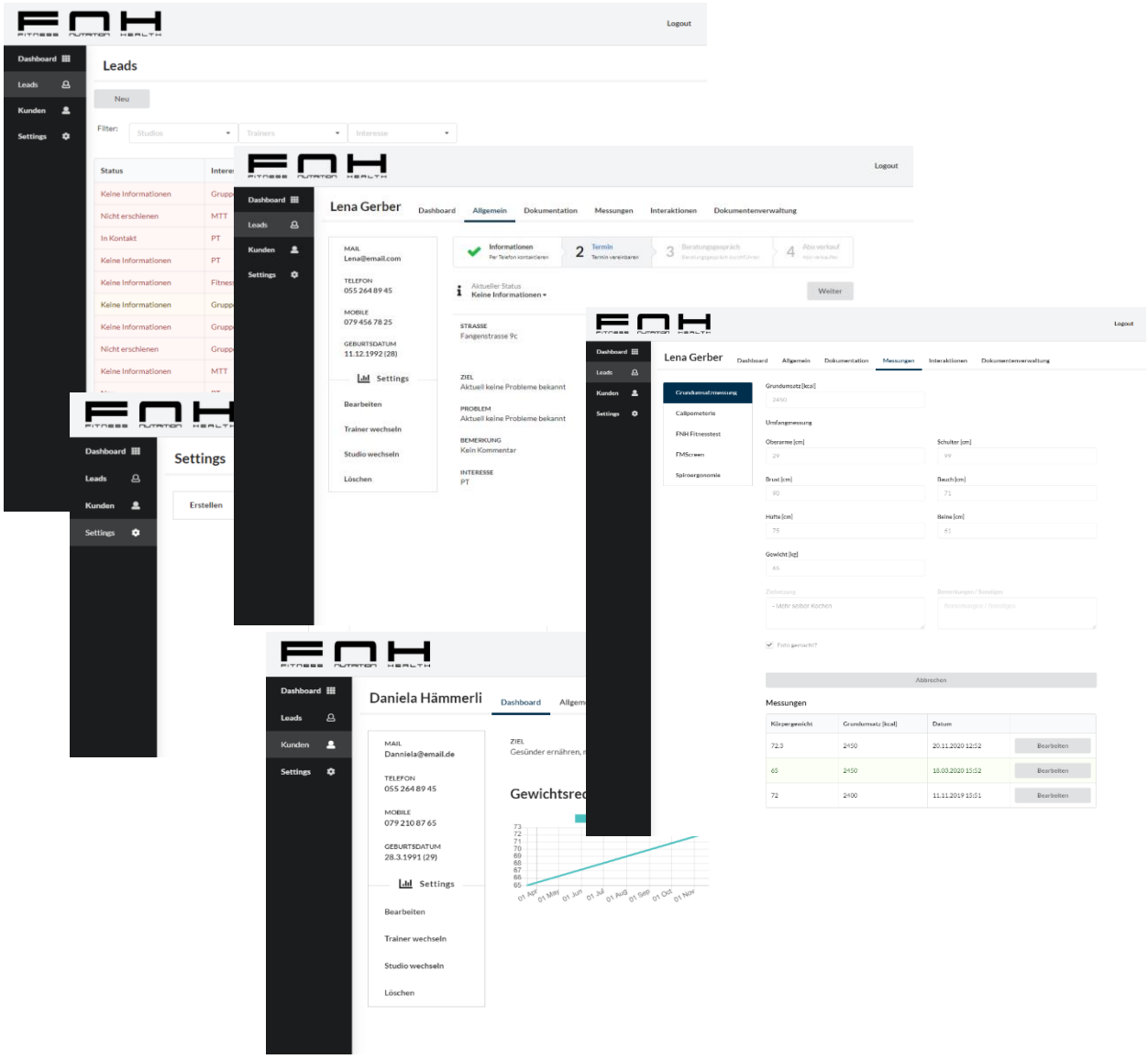

Figure 1- FNH-CRM - Overview

1. Welcome

The cloud-based web application FNH-CRM is a CRM tool specially designed and customised for fitness studios. Not only can all data from lead to customer be managed, one can also record various measurements directly in the application. All data can be analysed and evaluated thanks to a central database.

URL to the application: https://crm.fitaas.com/login

2. Navigating the CRM

Dashboard [Not yet implemented]
The Dashboard gives the personal trainer a central overview of the tasks to complete this day.

Leads

In addition to assigning a workflow, all leads and its associated data can be managed in this view. This includes editing the lead, recording or editing documentation, recording interactions or measurements, as well as the up- and downloading of contracts and notes.

Customers

Here, customers and the associated data can be managed. This includes editing the lead, recording or editing documentation, recording interactions or measurements, as well as the up- and downloading of contracts and notes.

Settings

In this view, the owner can manage all settings such as the users and their roles, studios and workflows. The chief personal trainer and personal trainer do not have access.

3. Using the software

3.1 Settings

Only the owner has access.

3.1.1 Users

Manage all the users of the application, this includes the owner, chief personal trainer and personal trainer.

Log in to the application and navigate to “Settings”.

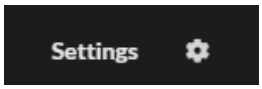

Figure 2 – Settings - Navigation

Navigate to «Benutzer» and you will see a list of all current users.

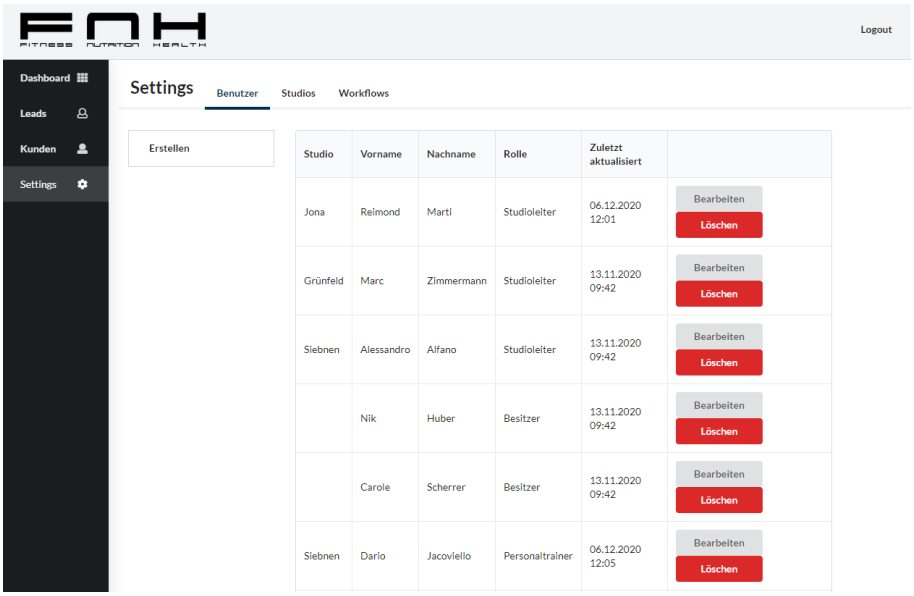

Figure 3 - Settings – User list

Create a new user

With a press of «Erstellen» you can create a new user

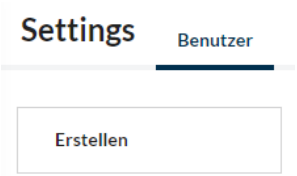

Figure 4 - Settings - Create button

A pop-up window opens. On the picture below, the required fields are inside the green rectangle. The remaining fields are optional. It is possible to edit all a user’s fields after he was created.

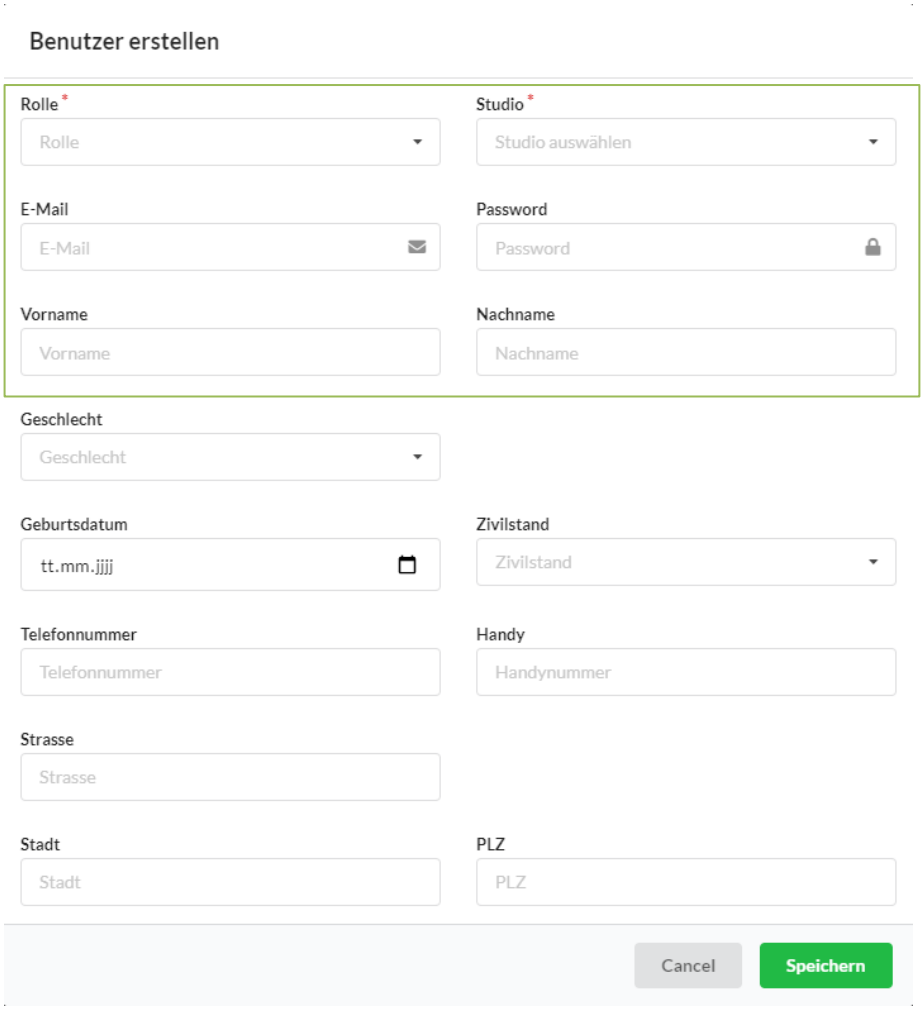

Figure 5 - Settings - Create user pop-up

For the role chief personal trainer ("Studioleiter"), an additional field called «Verantwortliches Studio» defines the studio for which he is responsible.

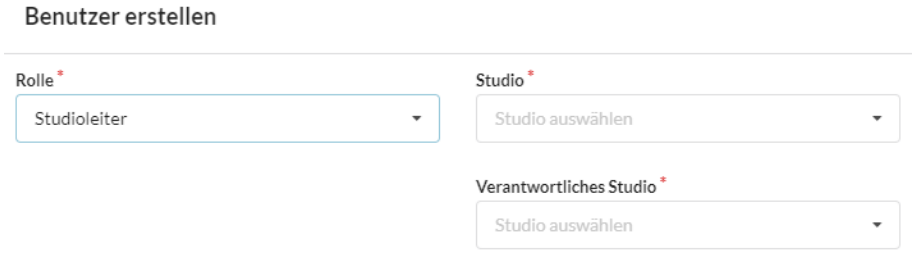

Figure 6 - Settings - Create user pop-up chief personal trainer

By clicking on save, the user is created and appears in the user list.

Edit a user

To edit a user, press the "Bearbeiten" button.

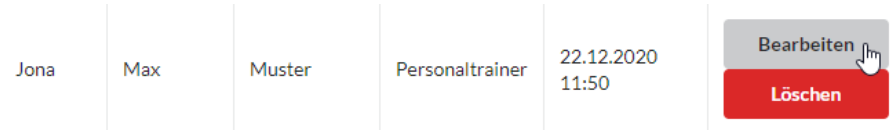

Figure 7 - Settings - Edit user

A pop-up will open. Here all data can be edited.

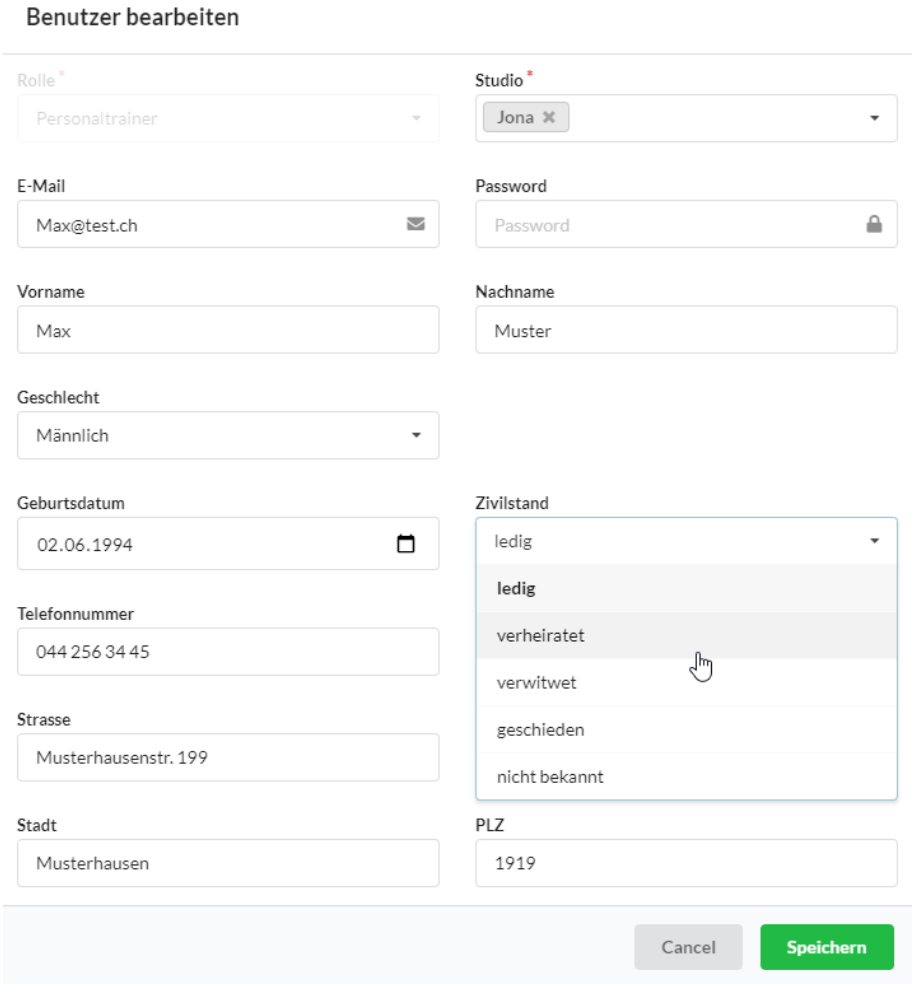

Figure 8 - Settings - Edit user pop-up

By clicking on ”Speichern”, the changes are saved.

Delete a user

Any user can be deleted. To do so, press the delete button.

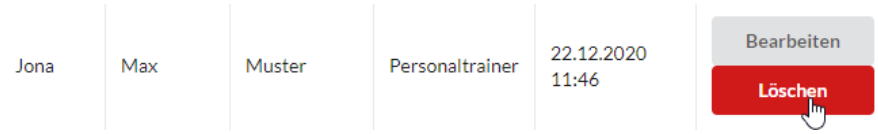

Figure 9 - Settings - Delete user

A pop-up opens in which the deletion must be confirmed.

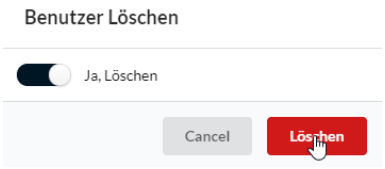

Figure 10 - Delete user pop-up

Important:
After confirming and pressing “Delete”, the user is deleted irrecoverably.

3.1.2 Studios

Log in to the application and navigate to “Settings”.

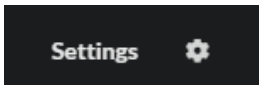

Figure 11 - Settings - Navigation

Navigate to «Studios» and where a list of all the studios is displayed.

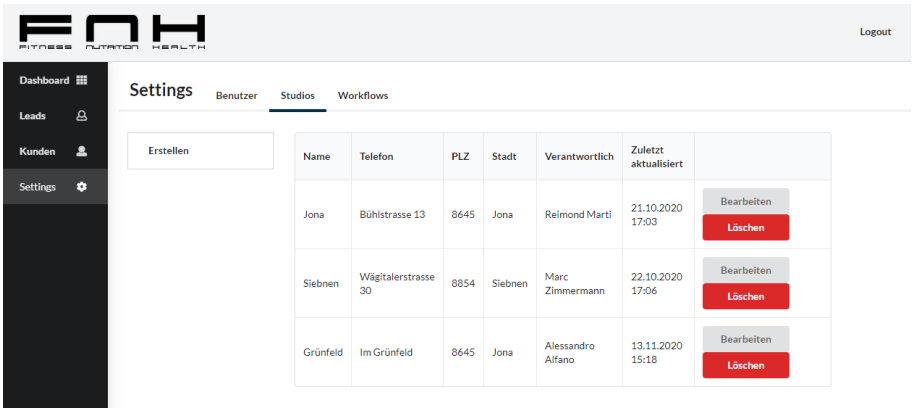

Figure 12 - Settings - Studio list

Create a new studio

With a press on «Erstellen» a new studio can be created.

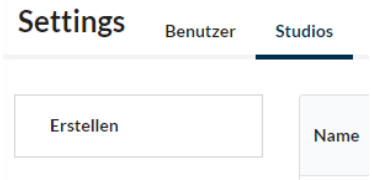

Figure 13 - Settings - Create button

A pop-up window opens. All fields except «Trainer» are required. It is possible to edit all a studio's fields after it was created.

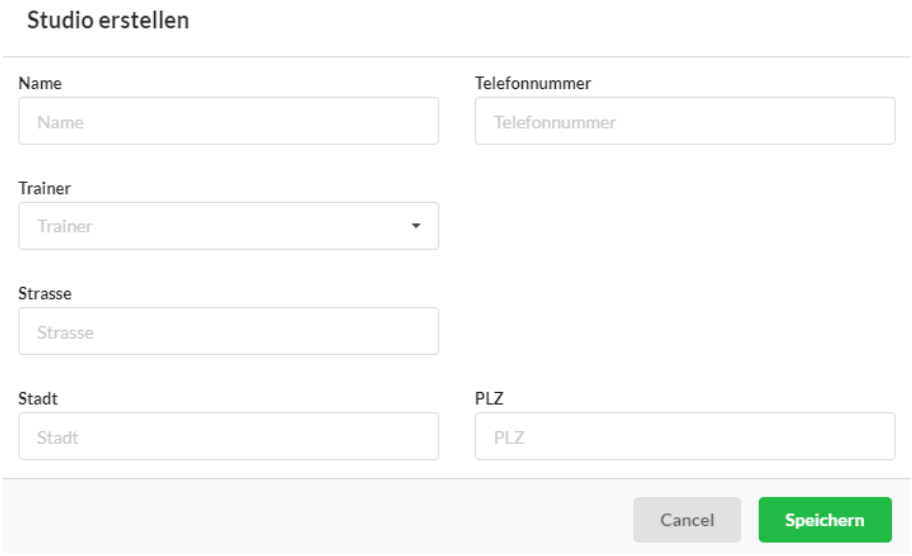

Figure 14 - Settings - Create studio pop-up

By clicking on save, the studio is being created and shown in the studio list.

Edit a Studio

To edit a studio, press the "Bearbeiten" button.

Figure 15 - Settings - Edit studio

A pop-up will open. All the data can be edited here.

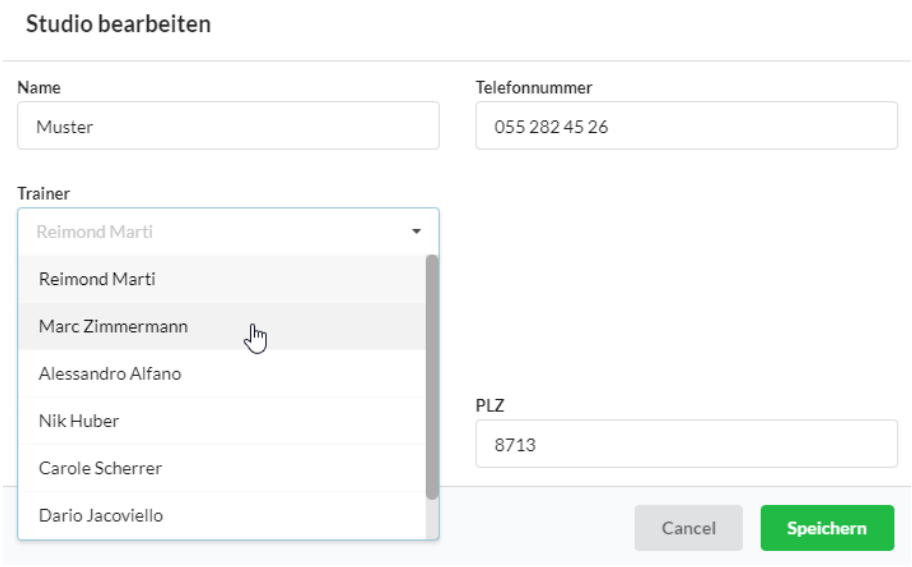

Figure 16 - Settings - Edit studio pop-up

By clicking on save, the changes are updated.

Delete studio
Studios can also be deleted. To do so, press the delete button.

Figure 17 - Settings - Delete studio

A pop-up opens in which the deletion must be confirmed.

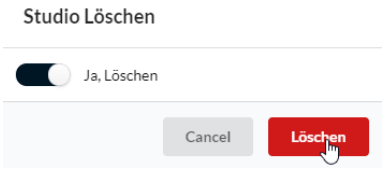

Figure 18 - Delete studio pop-up

Important:
After confirming and pressing “Delete”, the studio is deleted irrecoverably.

3.1.3 Workflow

Log in to the application and navigate to “Settings”.

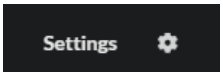

Figure 19 - Settings - Navigation

Navigate to «Workflows», a list of all the current workflows will be displayed.

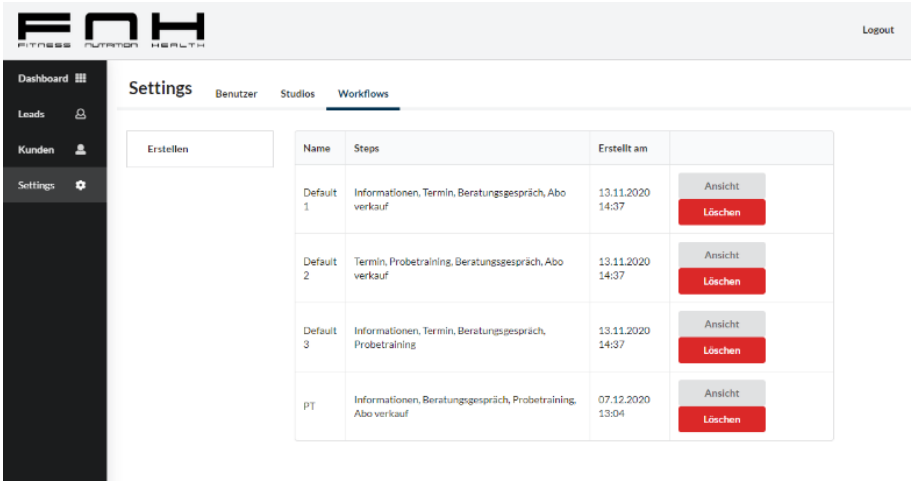

Figure 20 - Settings - Workflow list

Create a new workflow

With a press on «Erstellen» a new workflow can be created.

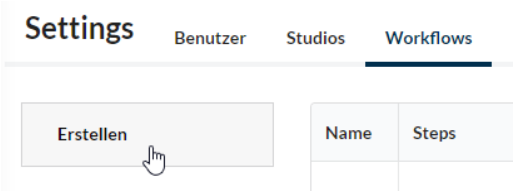

Figure 21 - Settings - Create workflow

A pop-up window opens. The field «Name» has to be filled out and at least one step has to be filled. With simple drag and drop, the “Bausteine”can be dragged in the desired order into the ”Steps”:

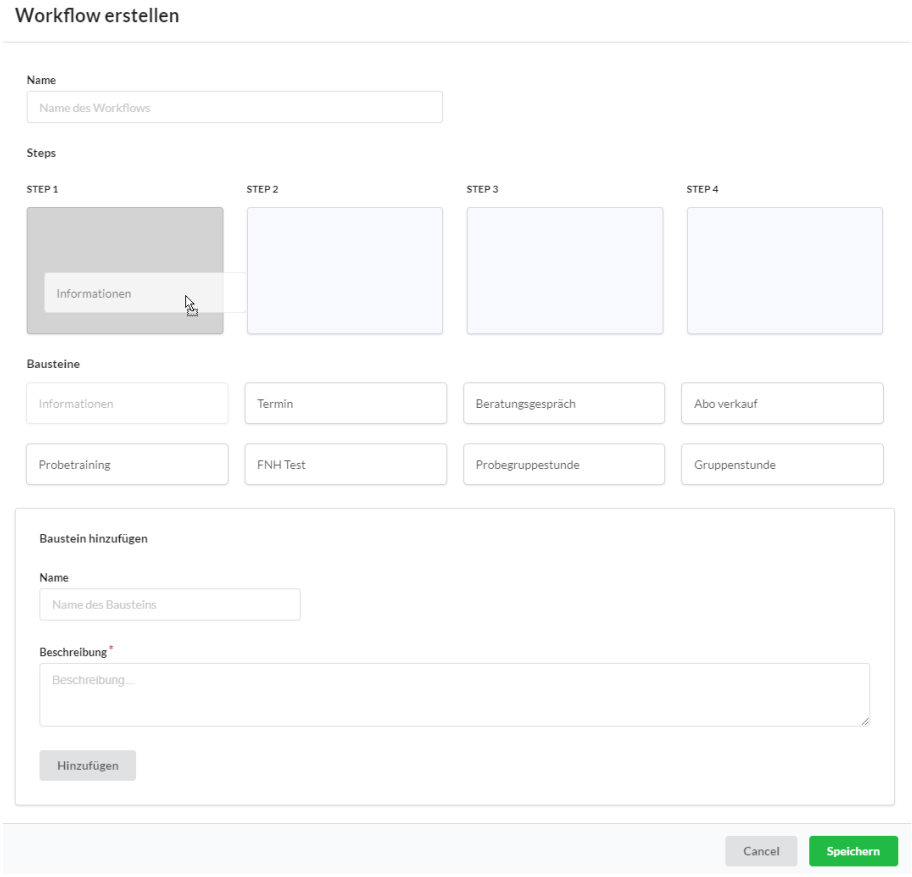

Figure 22 - Settings - Create workflow pop-up

New building blocks can be created a bit further down on “Baustein hinzufügen”. by pressing "Hinzufügen" the new building block appears immediately in “Bausteine”. By clicking on “Speichern”, the workflow is created and appears in the workflow list.

View a workflow

To view a workflow, press the "Ansicht" button.

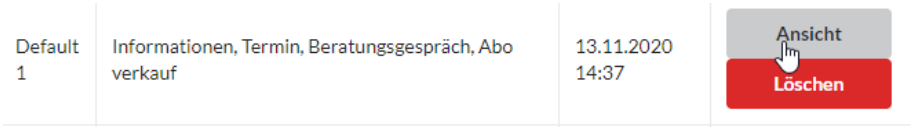

Figure 23 - Settings - View workflow

In the pop-up that opens up, all the workflows steps are displayed.

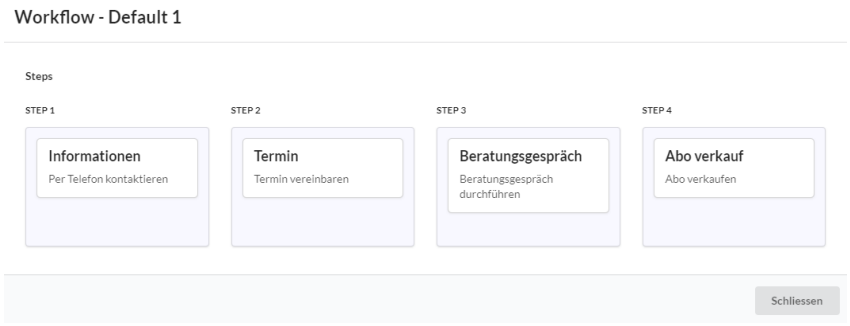

Figure 24 - Settings - View workflow pop-up

Delete a workflow
Any workflow can be deleted. To do so, press the delete button.

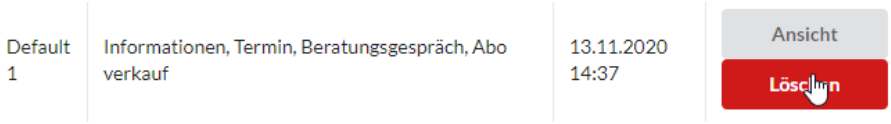

Figure 25 - Settings - Delete workflow

A pop-up opens in which the deletion must be confirmed.

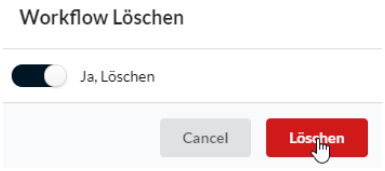

Figure 26 - Settings - Delete workflow pop-up

<u>Important:</u>
After confirming and pressing delete, the workflow is deleted irrecoverably.

<u>Nice to know:</u>
If the workflow is already assigned to a lead/customer, it will not be deleted from the lead/customer!

3.2 Leads

3.2.1 How to assign a workflow

Note:
Only a chief personal trainer or owner can assign a workflow to a lead.

To assign a workflow to a lead, the desired lead must be opened. this is done with a simple click on the lead in the lead list

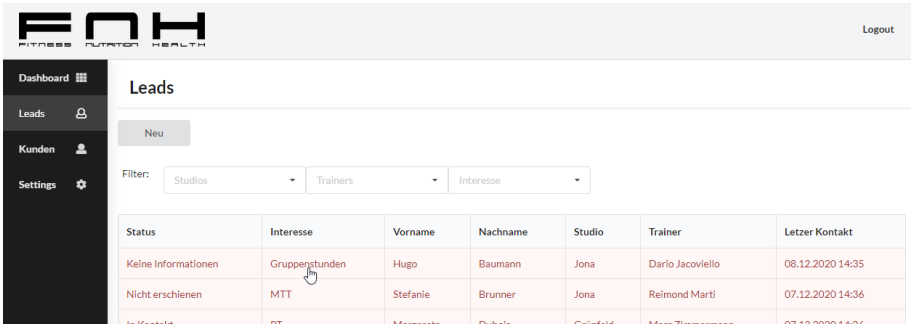

Figure 27 - Lead list

This takes you directly to the leads general page in which the workflow can be assigned. Select the desired workflow in the dropdown and confirm with the "Zuweisen" button.

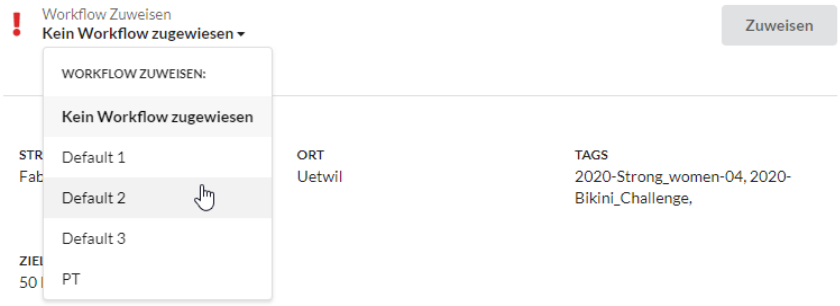

Figure 28 - Lead - General - Assign a workflow

A pop-up opens, in which the assignment can be confirmed or cancelled.

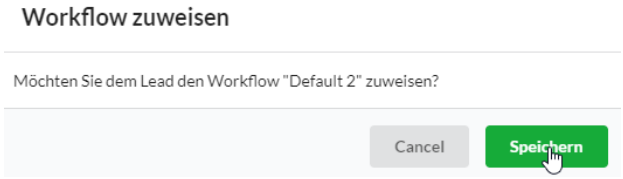

Figure 29 - Lead - General - Assign a workflow pop-up

By pressing Save, the workflow is being assigned and cannot be changed later on.

Figure 30 - Lead - General - Workflow assigned

3.2.2 How to change the status of a workflow

Navigate to the general page of the lead. There, the assigned workflow with the “Aktueller Status” is being displayed.

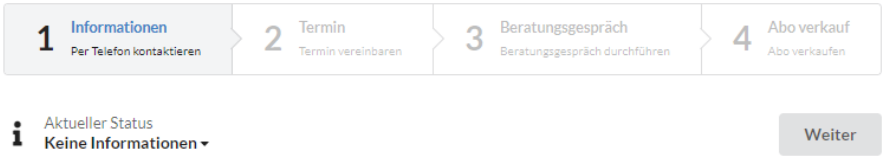

Figure 31 - Lead - General - Workflow

The current status of the lead can be adjusted below. Simply select the desired status in the dropdown

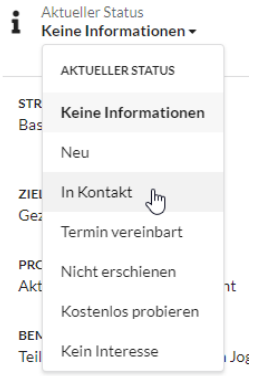

Figure 32 - Lead - General - Workflow status

3.2.3 How to complete a workflow step / How to convert a lead to a customer

Navigate to the workflow of the lead, the assigned workflow is being displayed.

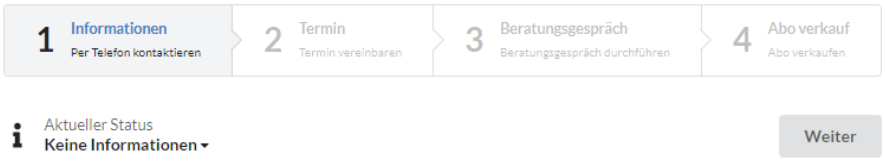

Figure 33 - Lead - General - Workflow

By pressing "Weiter" a pop-up that has to be confirmed to finish a step and start the next one.

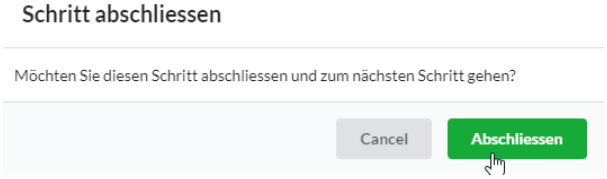

Figure 34 - Lead - General - Complete a step

By pressing "Abschliessen", the current step is completed and the next one becomes active.

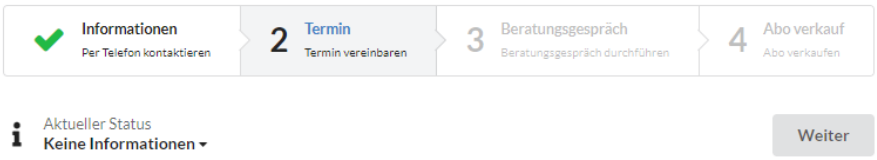

Figure 35 - Lead - General - Workflow step completed

Important:
If the last step is completed, the lead is converted into a customer and the login data is automatically sent to him by email.

3.3 Customer / lead

3.3.1 How to edit a trainee (customer or lead)

Navigate to the desired trainee and select the submenu «Dashboard» or «Allgemein».
On the left side you will find the vertical submenu to edit a trainee:

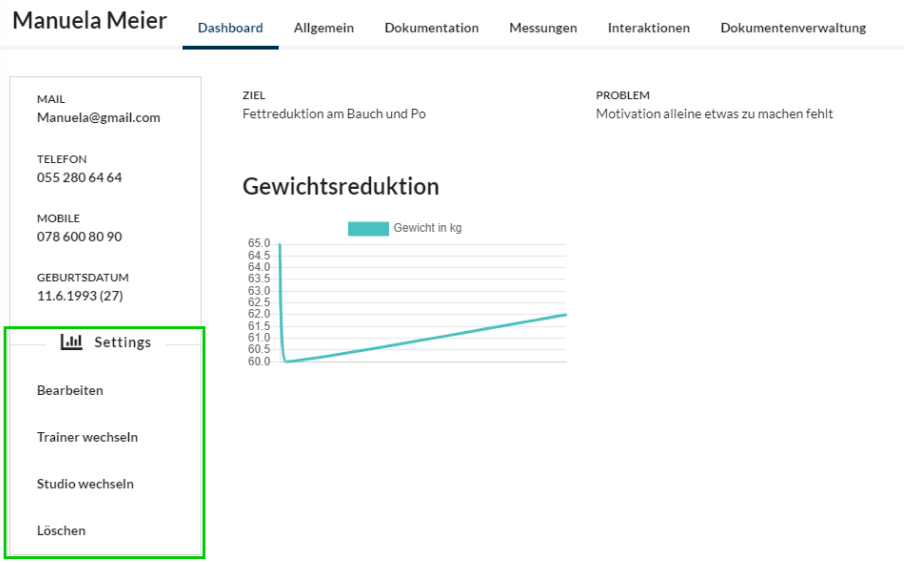

Figure 36 - Trainee settings

Edit – "Bearbeiten"
All trainee-related data such as name, first name, goals, problems, etc. can be edited.

Change Trainer – "Trainer wechseln"
The assigned trainer can be adjusted.

Change Studio – "Studio wechseln"
The assigned studio can be changed.

Delete – "Löschen"
The trainees and all with him associated data will be deleted irreversibly. A pop-up to confirm the deletion will appear before it is finally being deleted.

3.3.2 Documentation

In this horizontal submenu, «Dokumentation» such as general, training or nutrition can be recorded for a trainee.

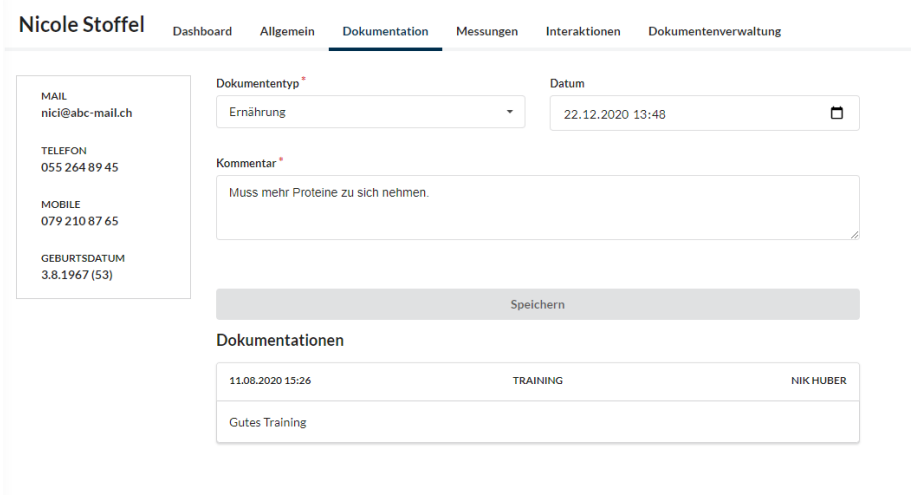

Figure 37 - Trainee - Documentation

To edit, the desired entry must be selected, it is then being highlighted in colour.

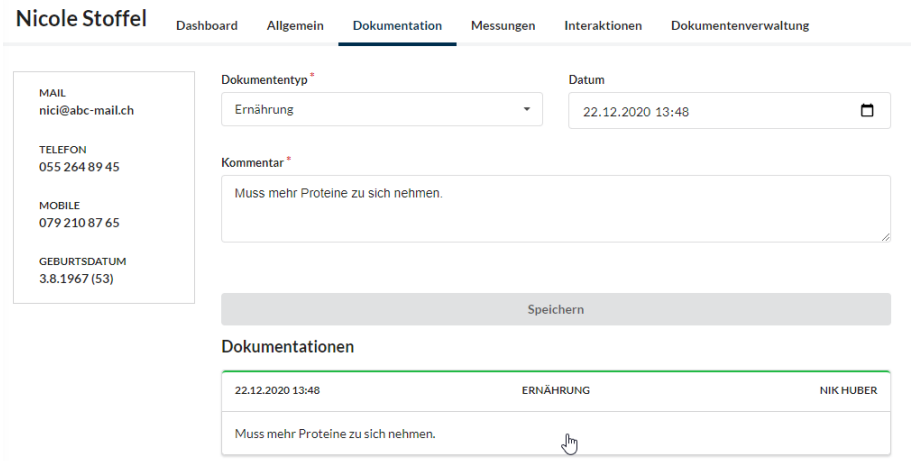

Figure 38 - Trainee - Documentation - Edit

The entry can now be edited and saved again by pressing Save.

3.3.3 Measurements

In this horizontal submenu, «Messungen» such as basic turnover, calipometry, FNH fitness test, FMScreen or spiroergonomic scan, measurements can be recorded or edited for a trainee.

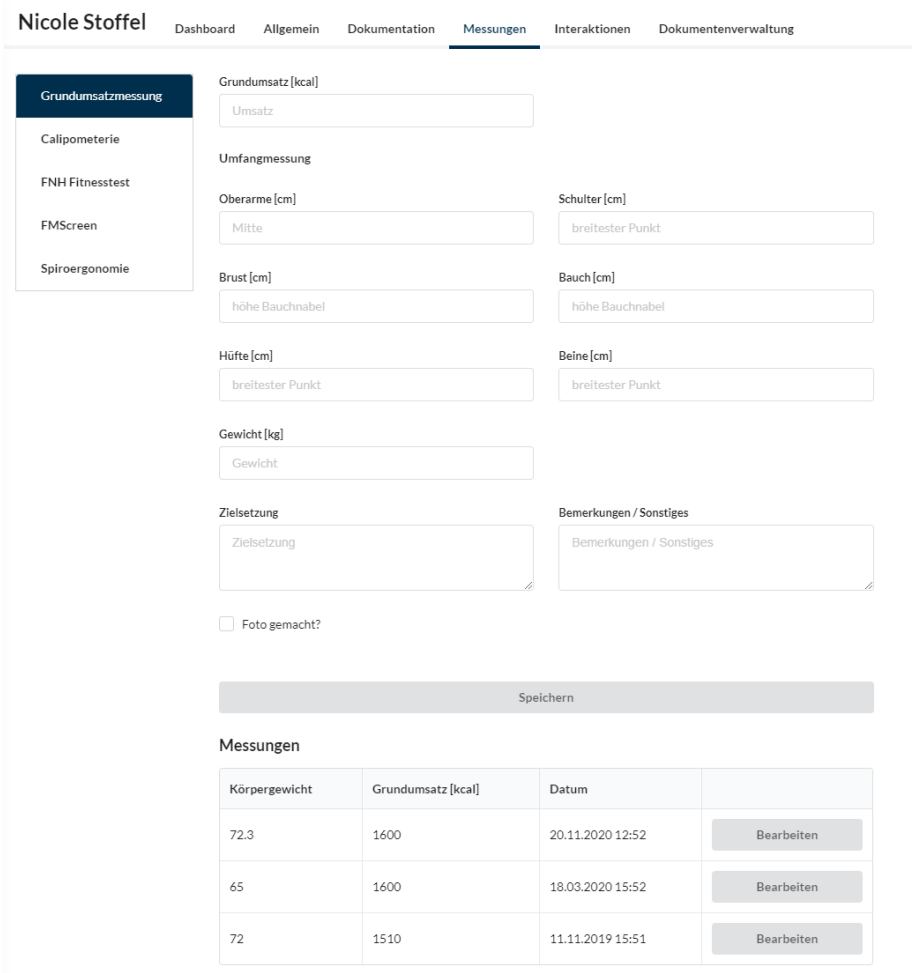

Figure 39 - Trainee - Measurement

To view a recorded measurement again, click on the measurement in the table and the data will be filled in the corresponding fields in the form above the table. The selected entry is highlighted in colour.

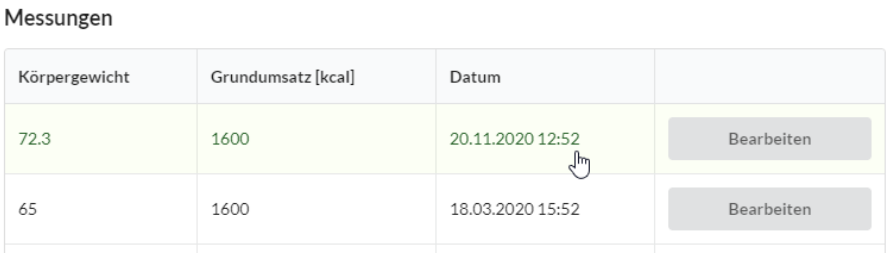

Figure 40 - Trainee - Measurement - View

A measurement can be edited. To do this, simply press the corresponding button "Bearbeiten".

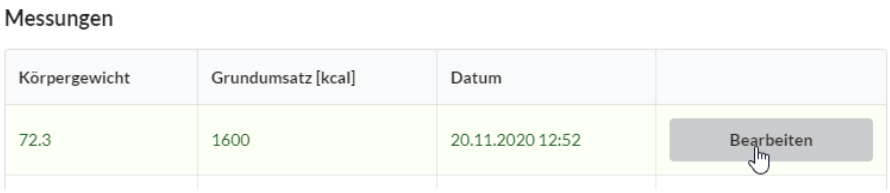

Figure 41 - Trainee - Measurement - Edit

The data is filled into the corresponding fields on in the form above the table and can be edited.

3.3.4 Interactions

In this horizontal submenu, «Interaktionen» such as on site, email, mobile, phone or WhatsApp can be recorded for a trainee.

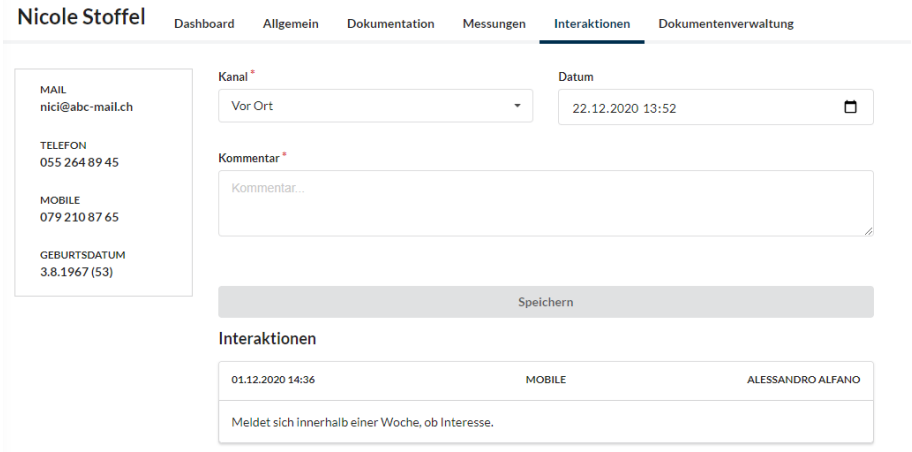

Figure 42 - Trainee - Interaction

<u>Important:</u>
A saved entry cannot be edited!

3.3.5 Document management

In this submenu, notes such as anamnesis forms or contracts can be downloaded and stored.

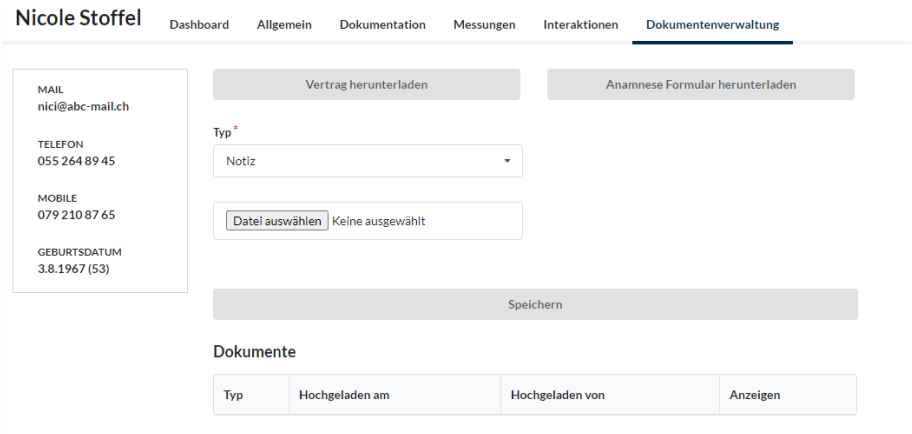

Figure 43 - Trainee - Document management

In the case of a contract, the expiration date and price must also be entered before the signed document is uploaded by pressing “Speichern”.

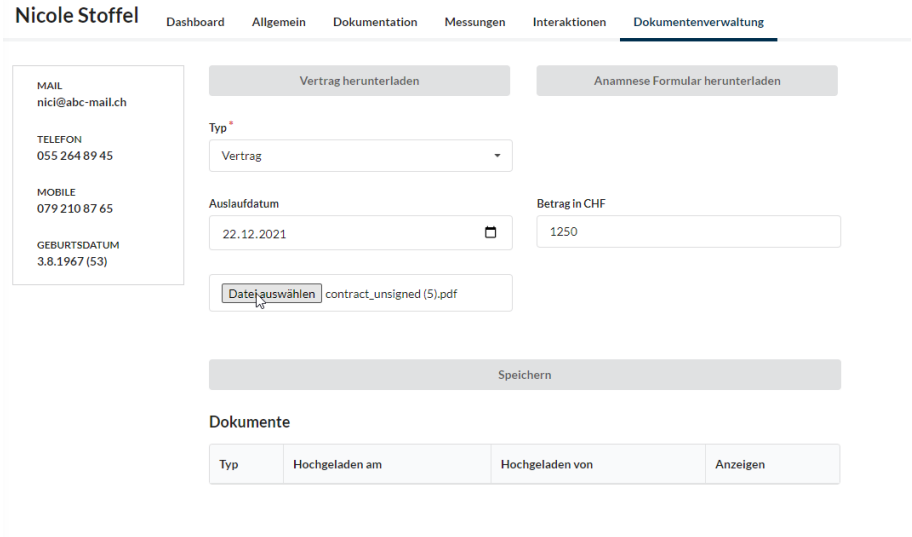

Figure 44 - Trainee - Document management - Contract

Documents can be viewed again at any time by pressing "Herunterladen".

Figure 45 - Trainee - Document management - Download

J. Implementation study appointment booking by studio client

1. Intro

The idea of this study is to find out what effort it would be to implement the appointment booking for customers, which would be the best suited technology and how to integrate it into the work done in the scope of the bachelor thesis. Because this would be a huge amount of work, it will only give a rough overview of the functionality. This study serves the customer with the basic information to outsource this part of the application in parallel or after the bachelor thesis.

1.1 Description

In the scope of this BA, the key components of the CRM will be implemented. To actually use the tool in the studios, the personal training appointments have to be managed inside the application as well.

Studio customer:	Can book appointments with their mobile phone (mobile app / website)
Personal trainer:	Have to be able to manage the appointment (blocking slots, working days, …) and have an overview of their workday in the FNH-CRM.
Billing:	When a customer books an appointment, its credits have to be deducted. This way, the billing can be performed automatically.

Table 1 - Description

1.2 Precondition

The Web Application should run on Edge, Chrome and Safari, on a Microsoft Surface Tablets with OS Version 10, on Microsoft personal computers with OS Version 10 and on Mac personal computers with OS Version 10.15 “Catalina”.
The time and feasibility estimation are done considering the experience and capabilities of the current BA Team members.

2. Current system

2.1 Customer app

The customers get access to the Virtuagym app that lets them book appointments and manage their training points (credit system to manage personal trainings).
Slots that are already booked are displayed in light grey. The ones written in black can be booked if the user has enough Personal Training points. A trainer is assigned to every training. There are different types of slots. E.g. personal training, group trainings.

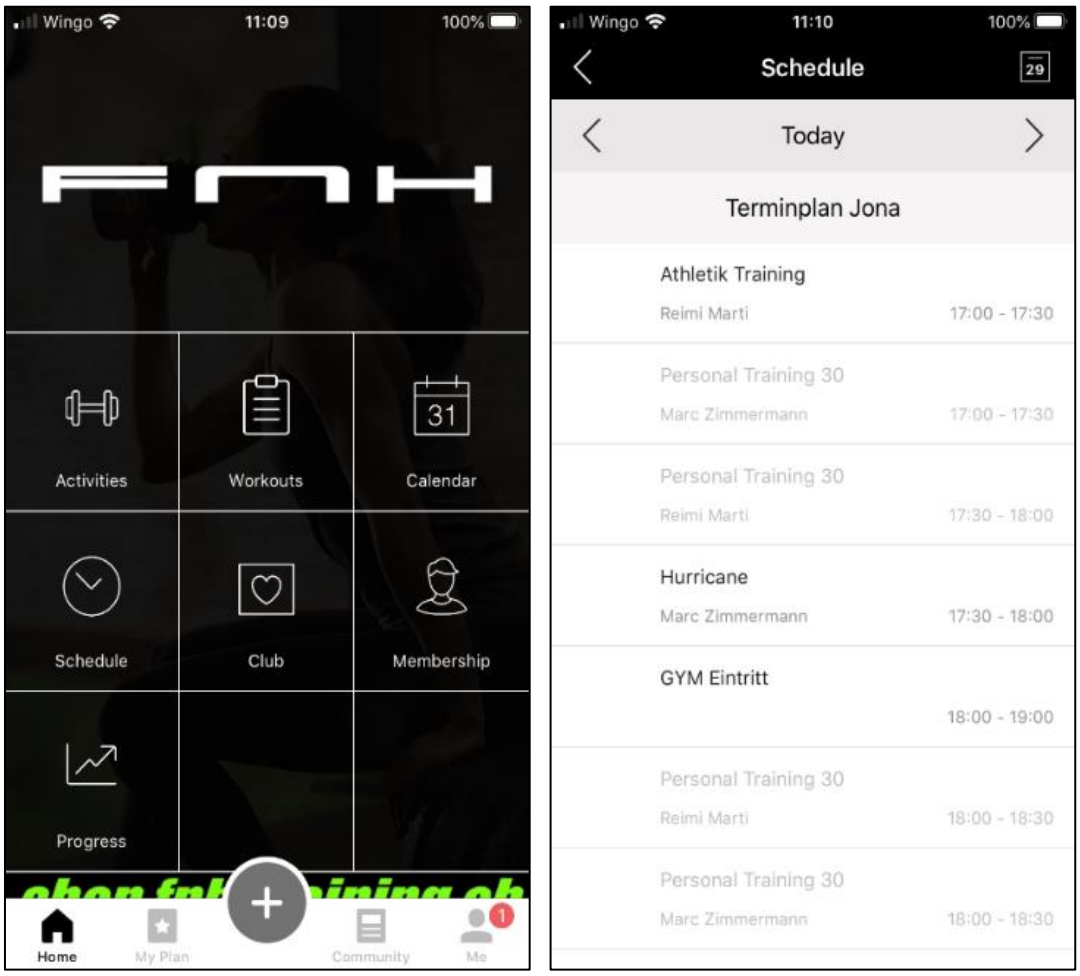

Figure 1 - Virtuagym - Customer app to book appointments

2.2 Personal trainer website

On the personal trainer's website, they have an overview of all the appointments (open and booked). New slots can be opened, existing ones can be deleted or booked manually. The slots are being booked for the next three years so they do not have to be booked manually day by day. After three years, the slots have to be extended for the next few years.

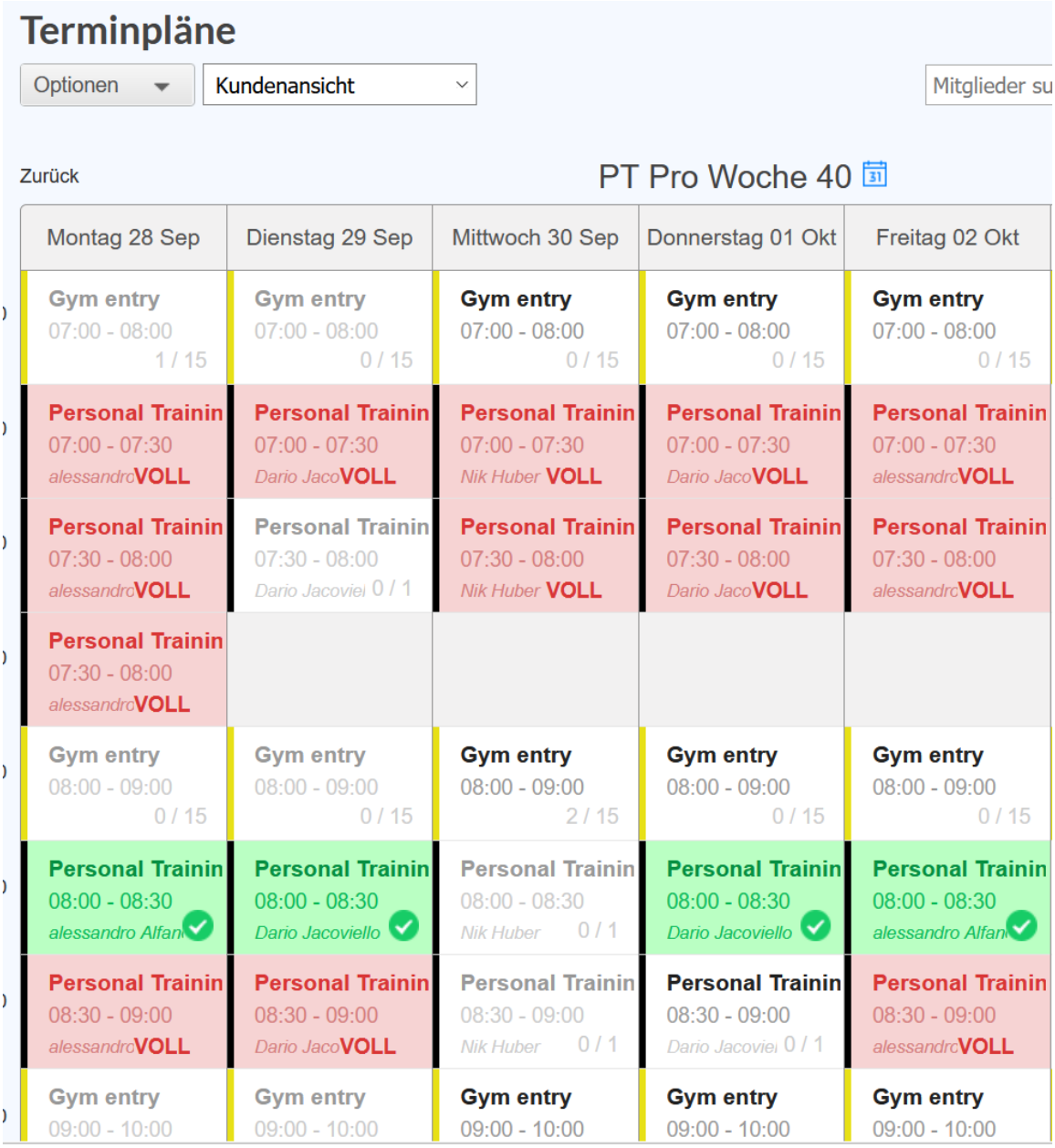

Figure 2 - Virtuagym – Manage appointments

3. Possible solution

Instead of an app, a web application, optimized for mobile devices is the cheapest and best solution. A website will be needed anyways and this way the only additional work will be to make it look like an app on the mobile phone.

3.1 Required functions

3.1.1 Customer

The website / app should be easy to use, the time to book an appointment should be as short as possible.

This is an example of an app from a different provider to show how it could work.

1. Click on the date when the customer would like to have the appointment.
2. Select a free appointment in the list below with the personal trainer the customer'd like.
3. "Beginn" will not be needed because the slots already have a start time. "Ende" should stay, so that a longer training can be booked.

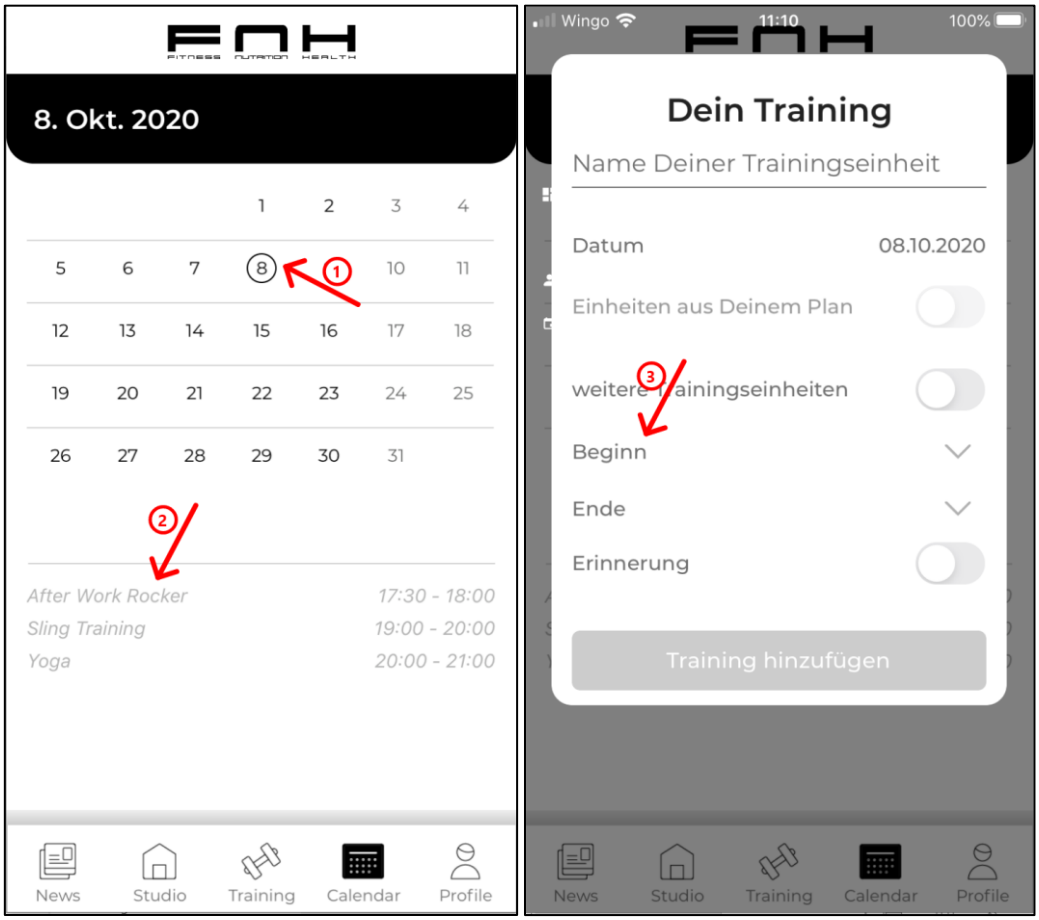

Figure 3 - App for customer to book appointments

3.1.2 Personal trainer

In the main menu on the left will be a new entry “Planung”, where a personal trainer’s daily calendar will be shown. With additional buttons, the trainer can also see the calendars of the entire studio and his working colleagues from the same studio.
The chief personal trainer can change (e.g. delete, lock or create) training slots.

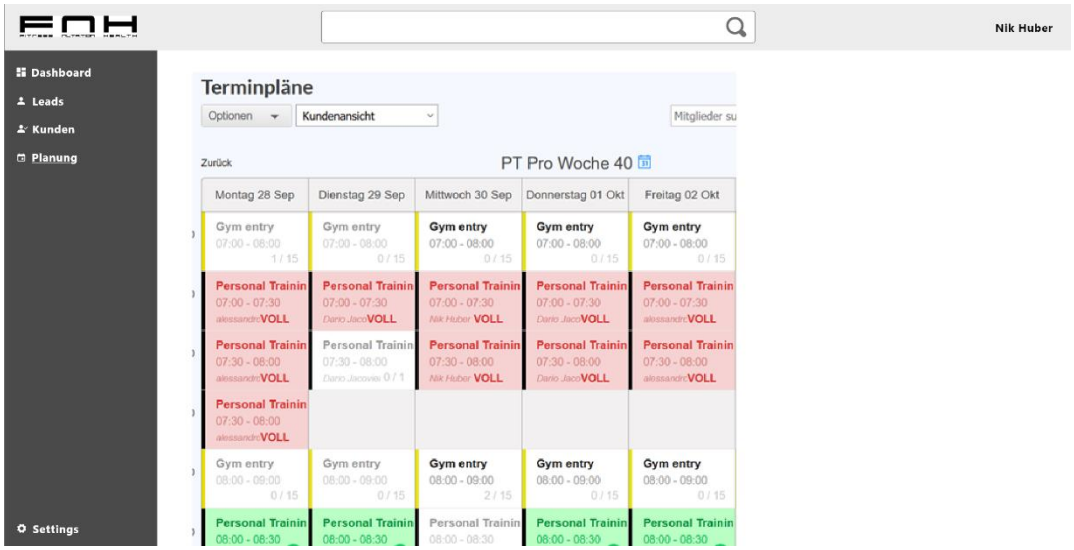

Figure 4 - Booking appointments in FNH-CRM

3.1.3 Billing

Whenever a customer books a personal training, personal training points have to be deducted. Personal training points have a certain value in CHF and are used to pay for personal and group trainings.
These have to be managed inside the application to check whether a customer can book trainings or not.

3.2 Required environment

To run the customer interface, an extra droplet will be needed. Everything else can be extended in the already running environment. Because only very little resources will be added for the additional functionality, scaling up will not be needed. The personal trainer's additional functionality will be added on the Admin-Frontend.

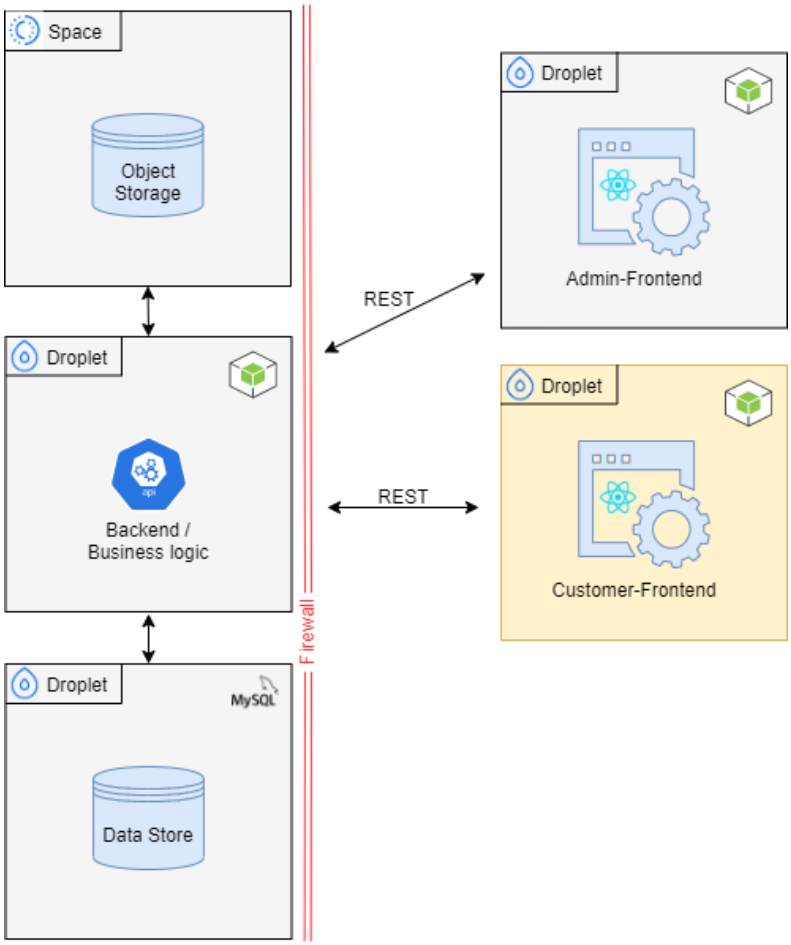

Figure 5 - Extended system overview

3.3 Technical realization

The technologies will stay the same as with the core features of the CRM. This way the application can be developed further by e.g. one person or a few people where everybody can work on every part of the application because the stack is the same.
Technologies: JavaScript, React (with Redux), MySQL, Docker, Swagger, DigitalOcean.
For more details, check out chapter 1.3 in the project plan.

3.3.1 Interface to FNH-CRM

The new functionality can be integrated into the existing Domain very easily by adding the appointment entity.

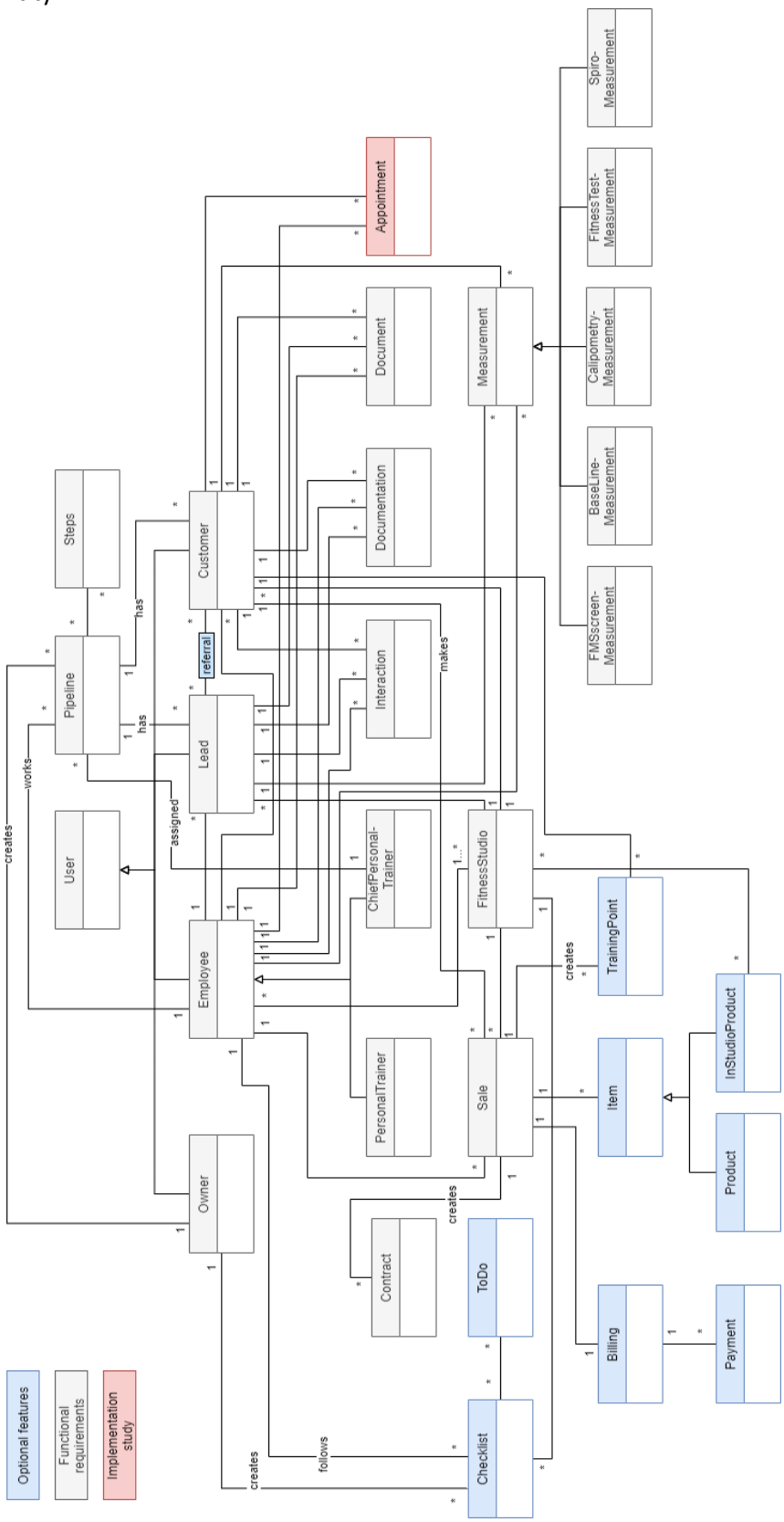

Figure 6 - Domain model extended

3.3.2 Integration in FNH-CRM

The new Business Logic can easily be integrated into the existing backend as a new Module. In addition, some new routes and a Database entry have to be added.

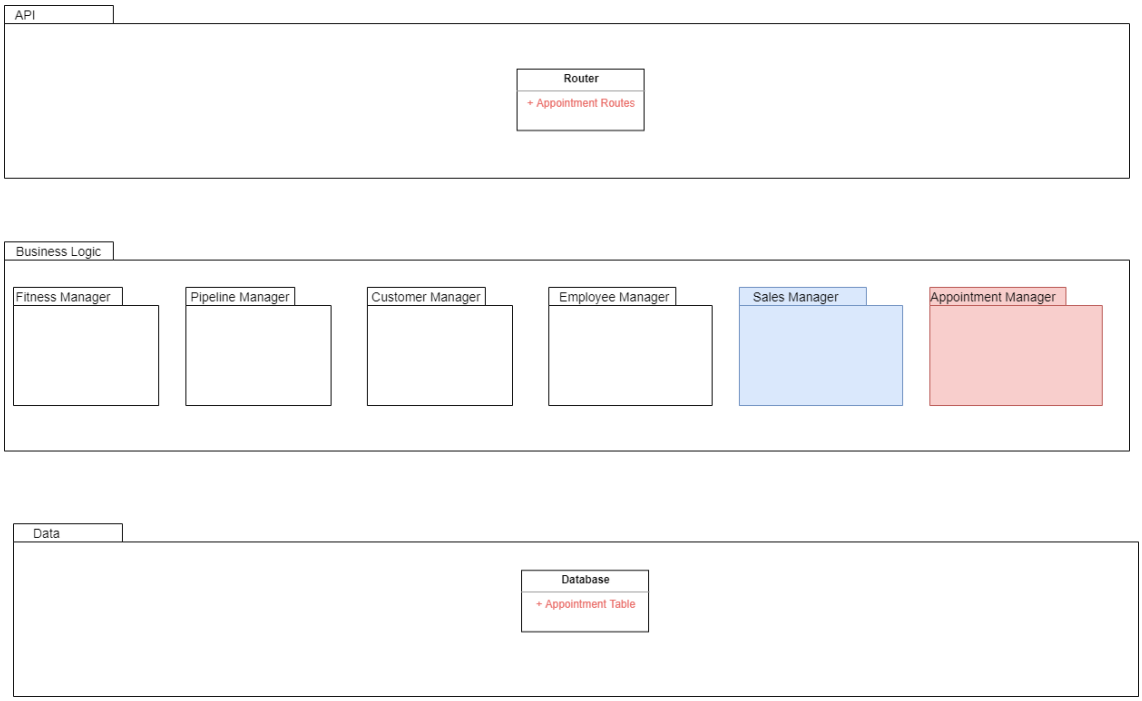

Figure 7 - Architecture

4. Estimation

The split into milestones allows to develop all the three components (Backend, Admin-Frontend, Customer-Frontend) at the same time and finish the project faster because the development can be done in parallel. Estimated time in hours is shown in brackets.

Preparations

- Setup development environment [24]
- Document the logic for the backend-implementation [40]

Milestone 1

Customer-Frontend	Creation of responsive Customer-Frontend-Website (no logic) - Design by hand [4] - Create Mockup [4] - Convert Mockup to Website [12]
Backend	Return appointment slots - Implement the authentication for users, including role extension [8] - Authentication-Mail logic and integration in lead to customer transition [8] - Add personal training points to a user [4]
Admin-Frontend	Create new page called “Planung” with a weekly schedule [16]

Milestone 2

Authentication and display of appointments.
Detail:

Customer-Frontend	Implement authentication and display appointments when clicking on date - Login logic [4] - Show agenda filled with appointments delivered by backend [16]
Backend	Booking of appointments - Allow booking of appointments and deduct personal training points [6] - Log whenever points are being deducted [4]
Admin-Frontend	Display the personal trainers weekly schedule with the slots delivered by the backend. [24] - Manage daily schedule plan (working hours by personal trainer) [32]

Milestone 3

Booking and deletion of appointments

Customer-Frontend	Booking of appointments if the user has enough personal training points [8]
Backend	Making changes to the schedule by the personal trainer (including holidays,..) [32]
Admin-Frontend	Book and delete appointments [8]

Milestone 4

Special cases and extended functionality

Customer-Frontend	- Forgot password [2] - Select preferred trainer [4]
Backend	- Forgot password mail and functionality [6]
Admin-Frontend	- Delete an appointment that user already signed up for [6]

4.1 Costs

This is a gross time estimation. We calculated with a salary of CHF 160 per hour for a supervisor in Switzerland and CHF 30 per hour for the designers and developers abroad.

Milestone	Time needed [h]	Additional* [h]	M&O* [CHF 160p.h.]	Cost per hour [CHF]	Total cost [CHF]
Preparation	64	83.2	0.00	160.00	13'312.00
MS1	56	72.8	2'912.00	30.00	5'096.00
MS2	86	111.8	4'472.00	30.00	7'826.00
MS3	48	62.4	2'496.00	30.00	4'368.00
MS4	18	23.4	936.00	30.00	1'638.00
	272	353.6	10'816.00		32'240.00

Table 2 – Cost calculation

*Additional
Includes the Time needed

+ 10% Testing and Documentation
+ 20% planning uncertainty

*M&O – Management and oversight
Of the additional time.

+ 25% Management and Oversight to ensure quality of the code and testing of the functionality. Costs are CHF 160 per hour

K. React UI libraries comparison

1. Intro

1.1. Description

The user interface (UI) is a large part of any application. There are several ways the styling of the applications can be implemented. One can create and design any element from scratch, such as buttons, or use familiar frameworks or libraries. The goal is to use a framework or a library that simplifies the styling of the applications and enables a visually appealing design.

1.2. Precondition

The frontend is being developed with React[9] and will run on different Web browsers. The selected library must be compatible with Redux[10] intuitive to use and open source. The library must make the development process more convenient, so that the developers can concentrate on the core of the application.

2. UI component libraries

React offers various libraries to simplify the styling of the user interface. Some of the libraries have been compared and evaluated.

2.1. Material UI

Material-UI[11] offers a comprehensive base of components and modules that provide perfect synergy with the ReactJS library. The design follows Google's material design guidelines[12]. It is a comprehensive guide to visual, motion and interaction design across platforms and devices. The components are self-supporting and only inject the styles they need to display, which could lead to performance enhancements in your application. Material UI is an open source tool with 61.6K GitHub stars and 18.45K GitHub forks. More than 140 companies use Material-UI, including well-known companies such as Nasa, Amazon, unity, Shutterstock and Netflix.

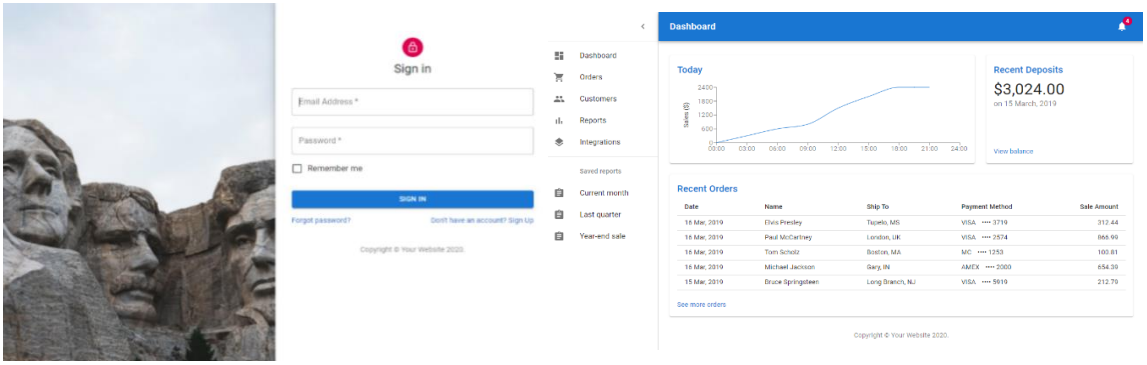

Figure 1 - Material-UI - Sample screens

[9] https://reactjs.org/
[10] https://redux.js.org/
[11] https://material-ui.com/
[12] https://material.io/design/

2.2. React Bootstrap

React Bootstrap[13] replaces the JavaScript in the regular Bootstrap components with React code. Each component was built from scratch as a true React component, without unnecessary dependencies like jQuery. As one of the oldest React libraries, React-Bootstrap has evolved along with React and has grown with it, making it a good UI base. Because React Bootstrap does not differ much from its bootstrap origins, developers can choose from the thousands of bootstrap themes that are immediately available. React Bootstrap has 18.4K GitHub stars and 2.9K GitHub forks.

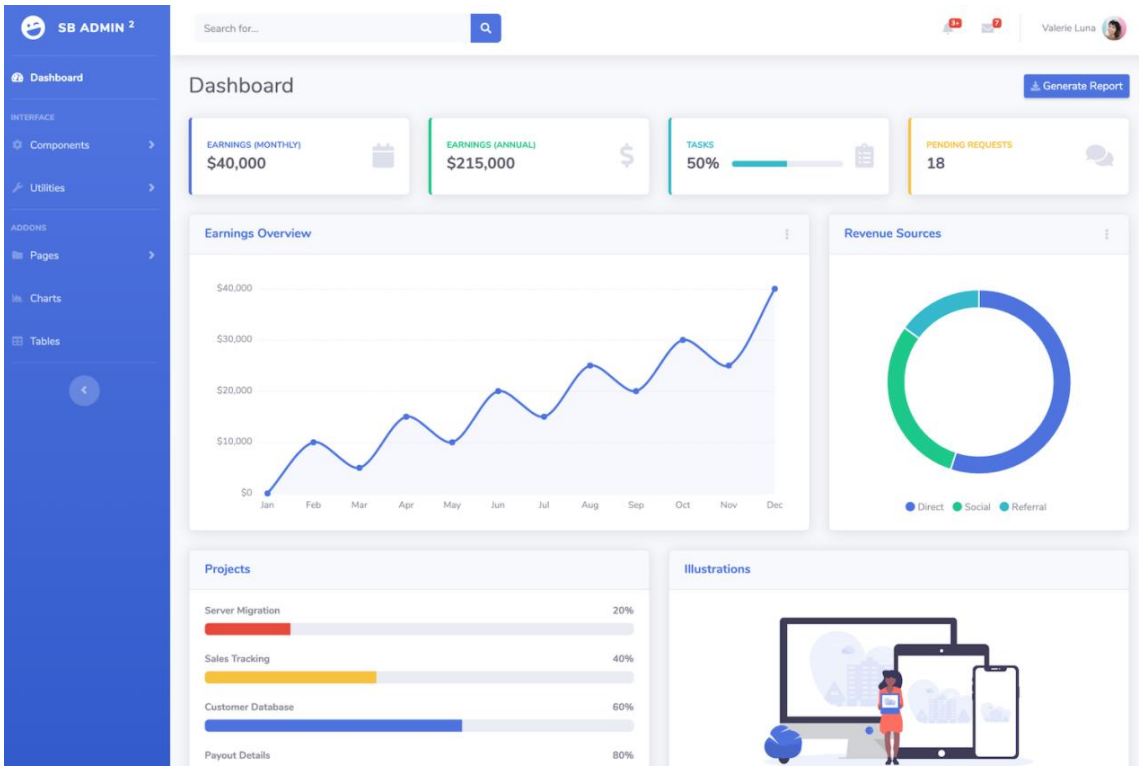

Figure 2 - React Bootstrap - Sample screen

[13] https://react-bootstrap.github.io/

2.3. Semantic UI

Semantic UI[14] supports designers and developers in creating a common vocabulary for UI. It is a jQuery based library that adds additional functionality to your pipeline. With Semantic UI React, all additional functions have been rewritten in React code. JSX code is used to define the components directly and bind them with the React component code. A huge list of pre-built components is provided, which are specifically designed to use and produce semantic-friendly code in a meaningful way. Semantic UI is an open source tool with 11.8K GitHub stars and 3.4K GitHub forks. About 115 companies use Semantic UI including Snapchat, Accenture, Kmong and DeepNatural.

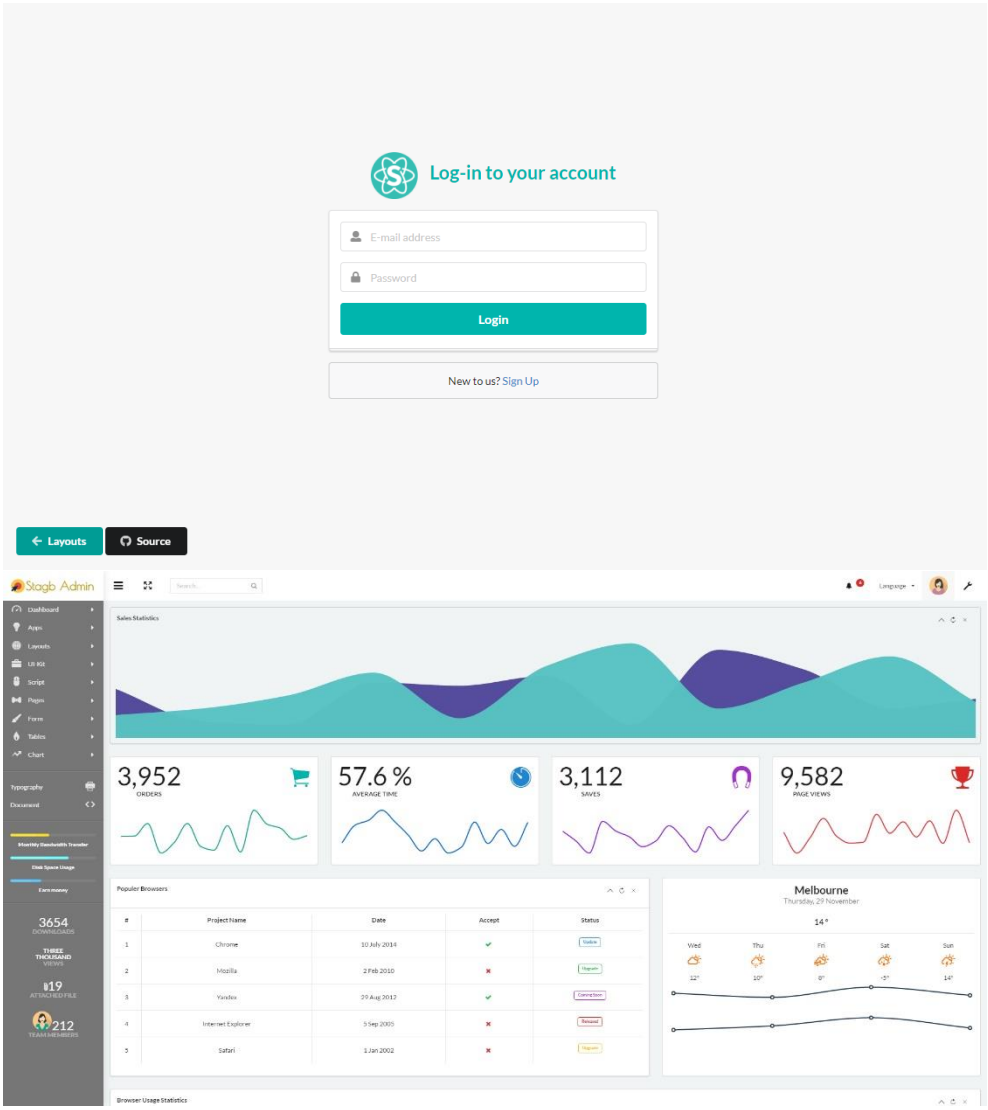

Figure 3 - Semantic UI - Sample screen

[14] https://react.semantic-ui.com/

2.4. Grommet

Grommet[15] is a react-based framework that provides accessibility, modularity, responsiveness, and theming in a tidy package. Grommet has a large library of components in its kit and counts big names like Netflix and Boeing among its users. The Grommet Design Kit is a drag-and-drop tool that makes designing your layout and components a breeze. It features sticker sheets, app templates, and icons galore. Grommet is still growing and already has 6.7K GitHub stars and 808 GitHub forks

Figure 4 - Companies that use Grommet

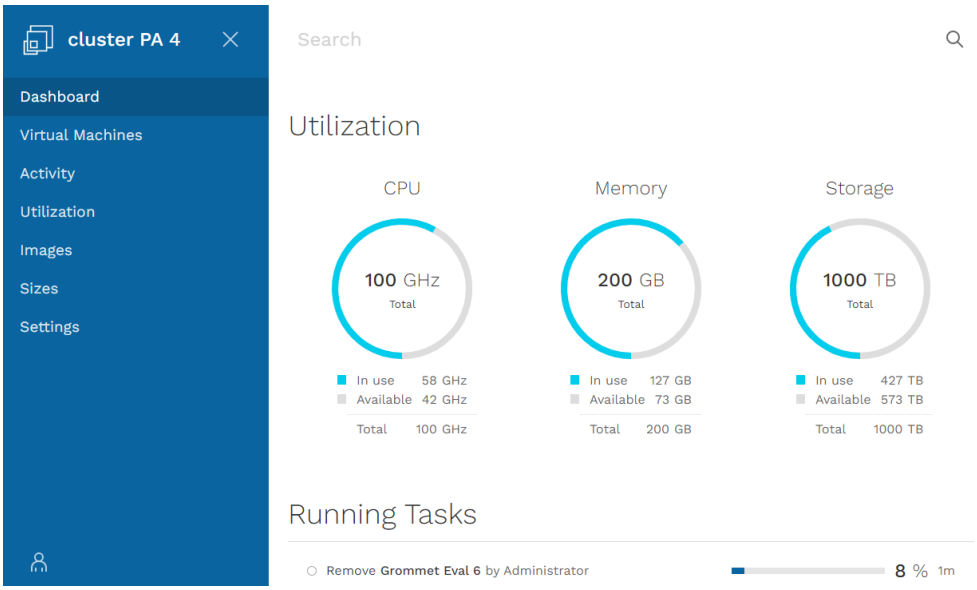

Figure 5 - Grommet - Sample screen

[15] https://v2.grommet.io/

2.5. Ant Design

Ant Design[16] provides more than 50 customizable components that can be used to create beautiful applications. Ant Design Pro offers out of the box UI solution for enterprise applications. Ant Design is a popular React UI library with over 63.9k stars and 24.5K forks on GitHub.

It was designed specifically for internal desktop applications and is based on several principles and uniform specifications. It makes design and prototyping easier and accessible to all members of a project.

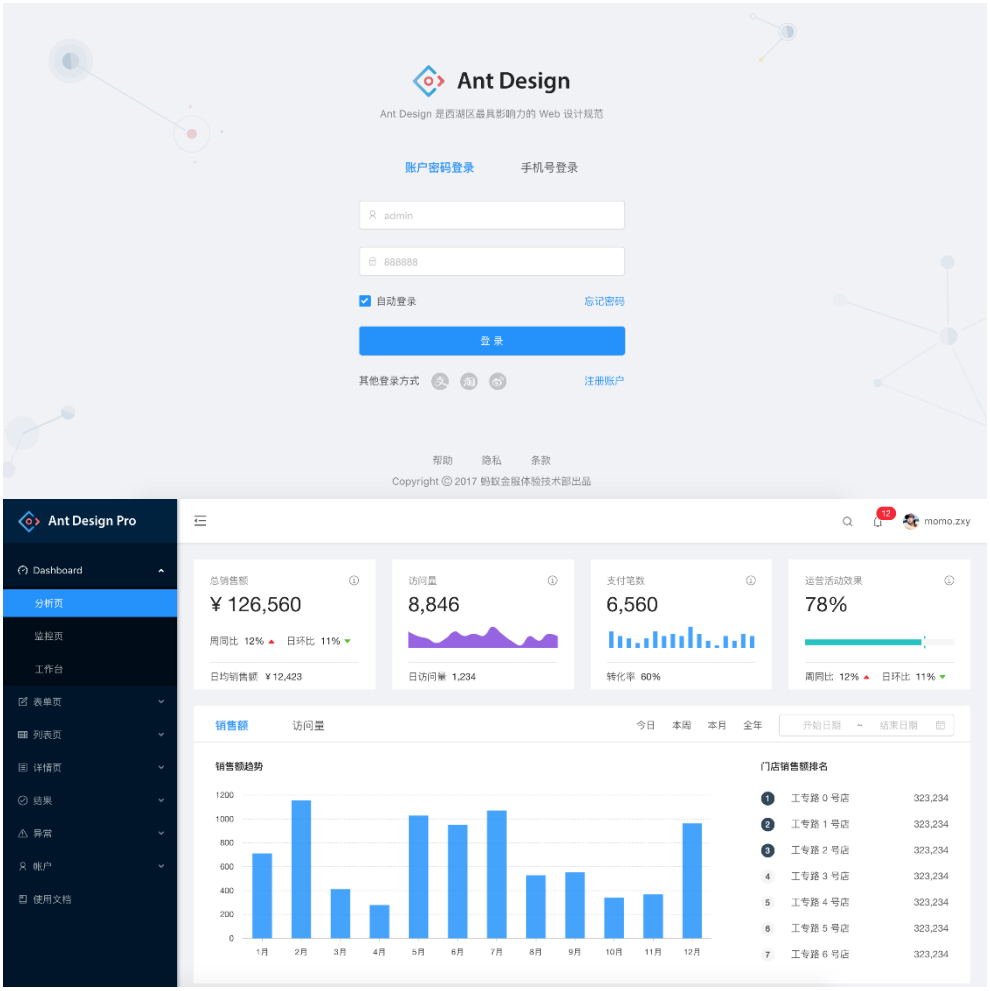

Figure 6 - Ant Design - Sample screens

[16] https://ant.design/

3. Analysis and decision

There are many different UI Component Libraries, all of them have advantages and disadvantages depending on the intended use. In the context of React and Redux, all analysed libraries / frameworks can be used.

Our goal is to keep the application as platform independent as possible. An important aspect is that a sufficiently large community is available. This allows us to find solutions and support for problems more quickly. The framework with the smallest community is Grommet, as it has not existed for as long as the others and is still growing. Not only by using a platform independent framework like React, but also the design should be as consistent as possible and look the same on all devices, so that it integrates well into the overall system. Therefore, the libraries Material UI, which is based on Google Material Design, is not an option. Semantic UI and React Bootstrap are both very popular UI component libraries. React Bootstrap is the more popular of the two libraries, with 18,400 stars on Github. However, the team already has experience with Semantic UI and it has many different components with well documented code examples where it is ahead of the competition. This makes the development process much faster and more convenient. For this reason, and because it meets all our requirements, Semantic UI was chosen.