Software Design

Carlo van Kessel
S-DB-IP3 and S-DB-GP3
S3-DB03
Hans van Heumen, Marc van Grootel
02-01-2023
V4

Version control

Version	Changes
1	Initial version.
	Added 1. User Stories
2	Added 2.1 Discover
3	Added 2.2 Explore
4	Added 2.3 Test
	Added 2.4 Listen

Contents

1. User Stories

For my application I made user stories. User stories are like the requirements for the application, it describes what the user can do on the application.

1.1 Functional

- As a user I should be able to log in.
- As a user I should be able to create an account.
- As a user, I must be able to start a workout.
- As a user, I must be able to add an exercise to my workout.
- As a user, I must be able to add a set to an exercise.
- As a user, I must be able to record the number of repetitions, kilos and rest in the set.
- As a user, I must be able to see per exercise whether I have gone up in weight or repetitions so I could see if I made progress.
- As a user, I must be able to delete an exercise.
- As a user, I must be able to delete a set from an exercise.
- As a user, I must be able to see the totals of an exercise (total kilos, total reps and total exercises).
- As a user, if I do an exercise for the second time, I want the previous data to be included so I know what I did last time.
- As a user I want to link my smart watch so I can track my calories.
- As a user I want to be able to record my food.

1.2 Non-Functional

- As a client, I want the website to load within 1 seconds so that users stay on the website.
- As a user, I want the website to work as well on a phone as it does on a laptop.
- As a user, I want my data to be stored securely.
- As a user, I want the website to be live 99% of the year.
- As a software developer, I want the code to be maintainable and readable by any other software developers.
- As a user, I want the website to be at least runnable on Firefox and Safari.

2. Design

When starting with my design I followed UX design cycle. The cycle is broken up in 4 parts: discover, explore, test, and listen. Each of these 4 parts have methods to use.

Discover:
- Field study
- Diary study
- User interview
- Stakeholder interview
- Requirements & constraints gathering

Explore:
- Competitive analysis
- Design review
- Persona building
- Task analysis
- Journey mapping
- Prototype feedback & testing (clickable or paper prototypes)
- Write user stories
- Card sorting

Test:
- Qualitative usability testing (in-person or remote)
- Benchmark testing
- Accessibility evaluation

Listen:
- Survey
- Analytics review
- Search-log analysis
- Usability-bug review
- Frequently-asked-questions (FAQ) review

2.1 Discover

For the discover part I used the field study and user interview methods.

2.1.1 My experience

2.1.1.1 What do I use for tracking my workout?

For my workout tracking I use Google Docs. I make tables and fill in everything that I want which looks like this:

Week 6
Upper: 1

Oefening	Sets	Reps	kg
Bench	4	5	
Incline bench press	3	12	
T bar row	3	12	
Pull Ups	3	6	
Dumbell shoulder press	3	12	
Cable row	3	12	
Side lateral raises	3	12	
Barbell curls	3	12	
French press	3	12	

Lower: 1

Oefening	Sets	Reps	kg
Squat	4	5	
Lunges	2	12	
Leg curl	2	12	
Leg press	3	12	
Stiff legged deadlift	3	12	

Upper:2

Oefening	Sets	Reps	kg
Incline bench	3	12	
Benchpress	4	4	
Dumbell bend over row	3	12	
Machine chest fly	3	12	
Standing military press	3	12	

This is not efficient because every week I need to copy this table and then delete every set, rep, and kg I filled so that I can fill them again. For me this is the best I can find where I have most control over what I can fill in and change.

2.1.1.2 What do I want to see in a workout tracking app?

For me personally I really want to fill the workout name I am about to do or choose a workout I did in the past. Then if I created a new one, I want to create exercises and add sets to them where I can easily fill in the reps and rest time and just by a clicker add weight. when I pick a workout, I have already done I want those last numbers to be like greyed out so that I can see what I have done the last time I did that workout. This way I can easily track progressive overload which is my workout goal.

2.1.2 Survey

For a different opinion about workout tracking in general I did a survey on 3 people who use the gym.

2.1.2.1 What do these people use the gym for?

2 out of the 3 people use the gym for their own. They go about 3 to 5 times a week. They go for muscle gaining and to this without the help of a coach/trainer. The other person is a health coach and helps people with losing weight and gaining muscle. He is a professional and has his own company.

2.1.2.2 Do they use an app for tracking? And what is good and bad about it.

They all use an app to make or track their workout. For the 2 who just go to the gym they use Google Docs. They don't have discovered an app where they can easily track their workout and see a clear view of their exercises. As for the coach he uses an app where he makes his own workouts for clients. He can easily add exercises and the clients can see a video of how to perform the exercise. This app is especially for making workouts for other people like this health coach/trainer. He says that the clients are very happy with the app and so is he. For him the app works perfectly with what he wants for his clients.

2.1.2.3 Conclusion

For my app I want to focus on the people who just go to the gym and want to track their workout. As the survey check out that there are different apps for a coach/trainer. This is not where my app will compete against, I want to make my app easy for tracking and to have a clear view of the workout.
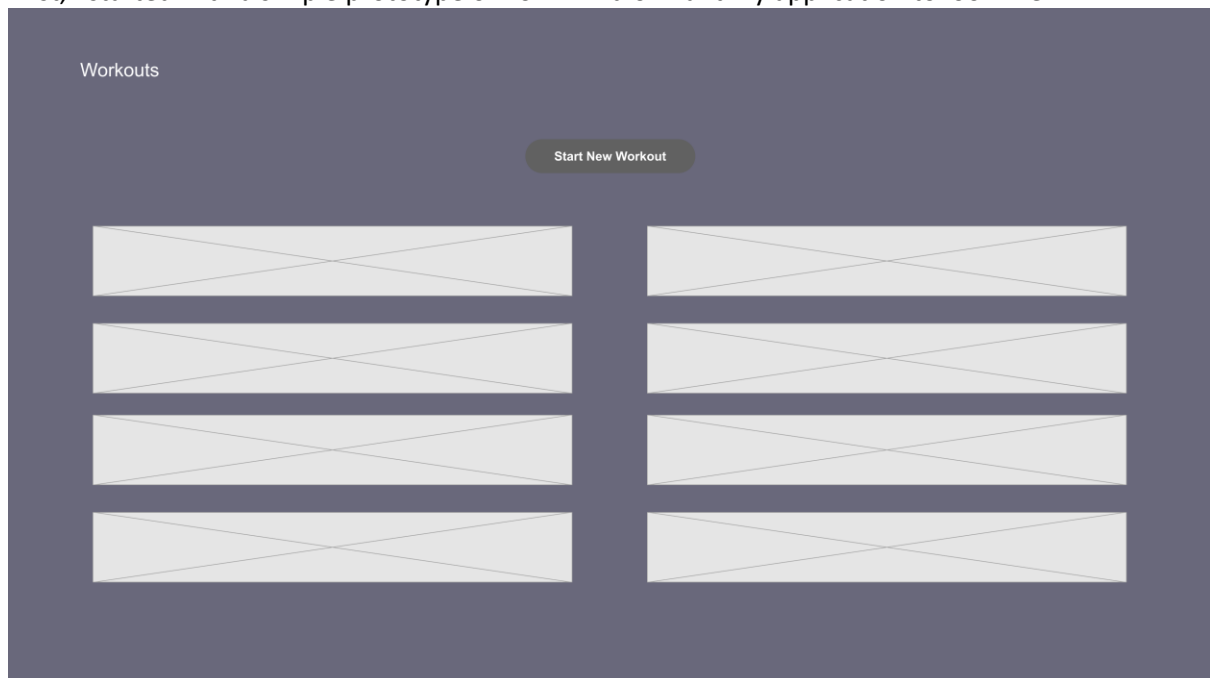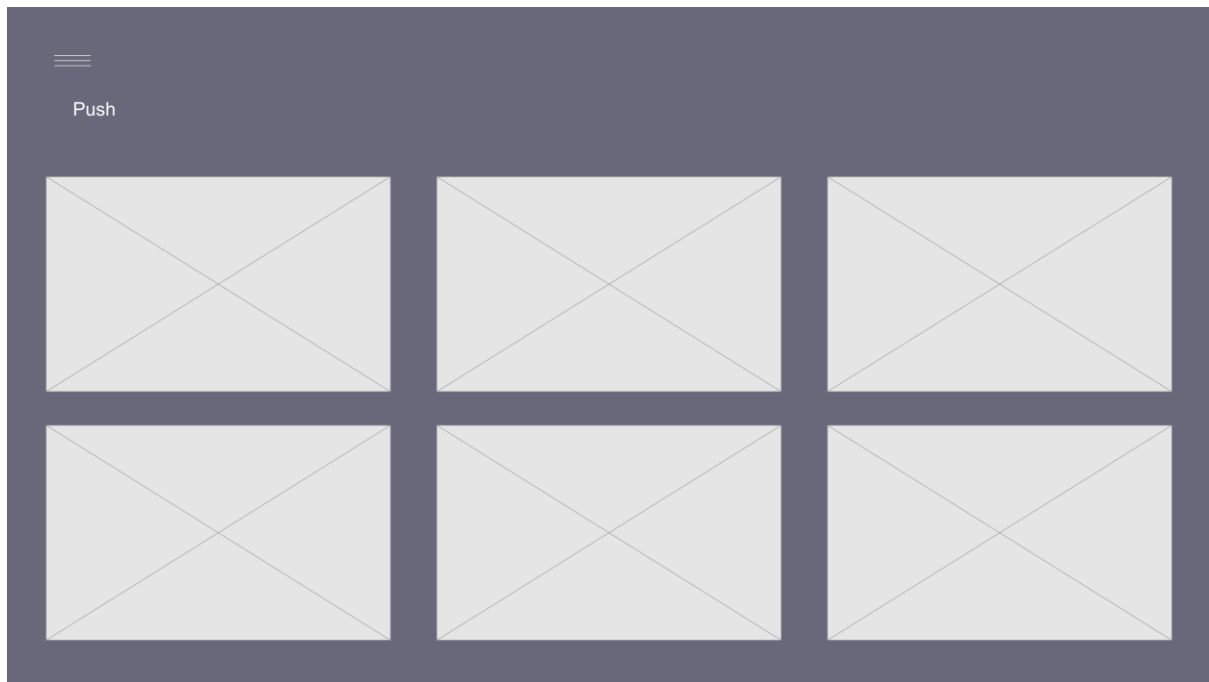
2.2 Explore

For the explore part I used different methods: prototype feedback & testing and user stories. The user stories can be read here: 1. User Stories.

2.2.1 Prototype feedback & testing

2.2.1.1 Prototype 1

First, I started with a simple prototype of how I kind of want my application to look like:

Here you can see a simple design with on the first image an overview of all the workouts. Where in the second image there is an overview of all the exercises in the workout.

2.2.1.2 Prototype 2

For the second prototype I made some changes to the overview of both pages:

Here I added some details of what I want the user to see in the overview such as: workout date and time, the exercises that are in the workout and all of the total kilo's reps and sets.

For the second view I added some details and added a total at the top of the screen. Here users can see all the totals of their workout. The exercises are split up into reps, sets, weight, rest, delete, and update. At the bottom of the page there is a save button where the user can save the workout.
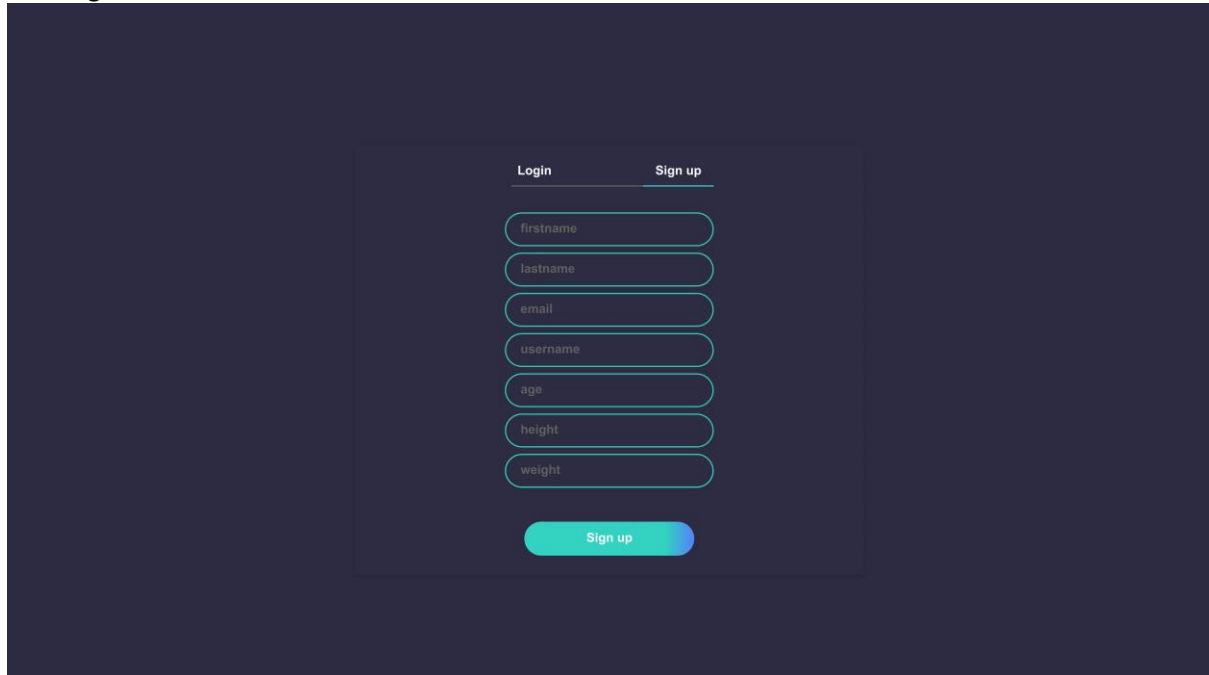
2.2.1.3 Design Desktop

For my design I started with picking colors. I wanted a dark background with brighter tones on the front. So, for my login page I made this design:
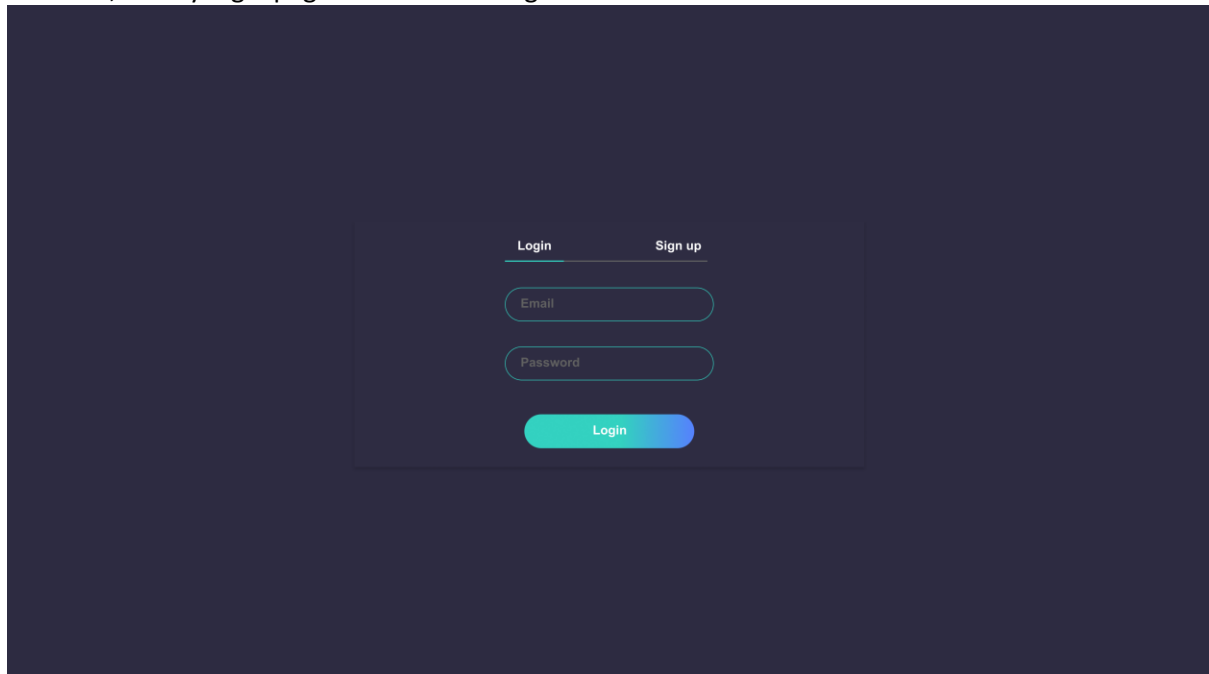
Here the user can login and you can see the bright color coming back in de button and in de form rounding's.
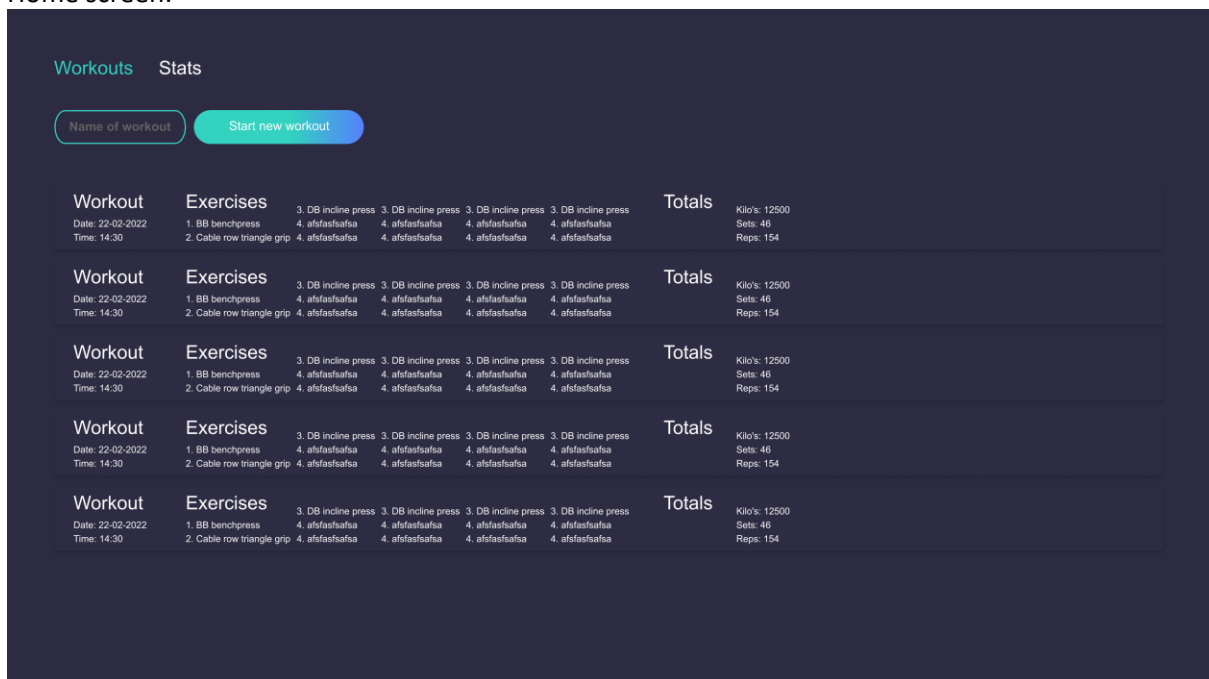
User sign in:

If a user does not have and account, he can make one with this screen.

Home screen:

For the home screen I made a table of all the previous workouts. Here you can see the date and time of the workout, the exercises that are done and all the totals of the workout. With the 'start new workout' button you can start a new workout. On the left of the button, you add your workout name. Then you come to the next screen:

Workout overview:
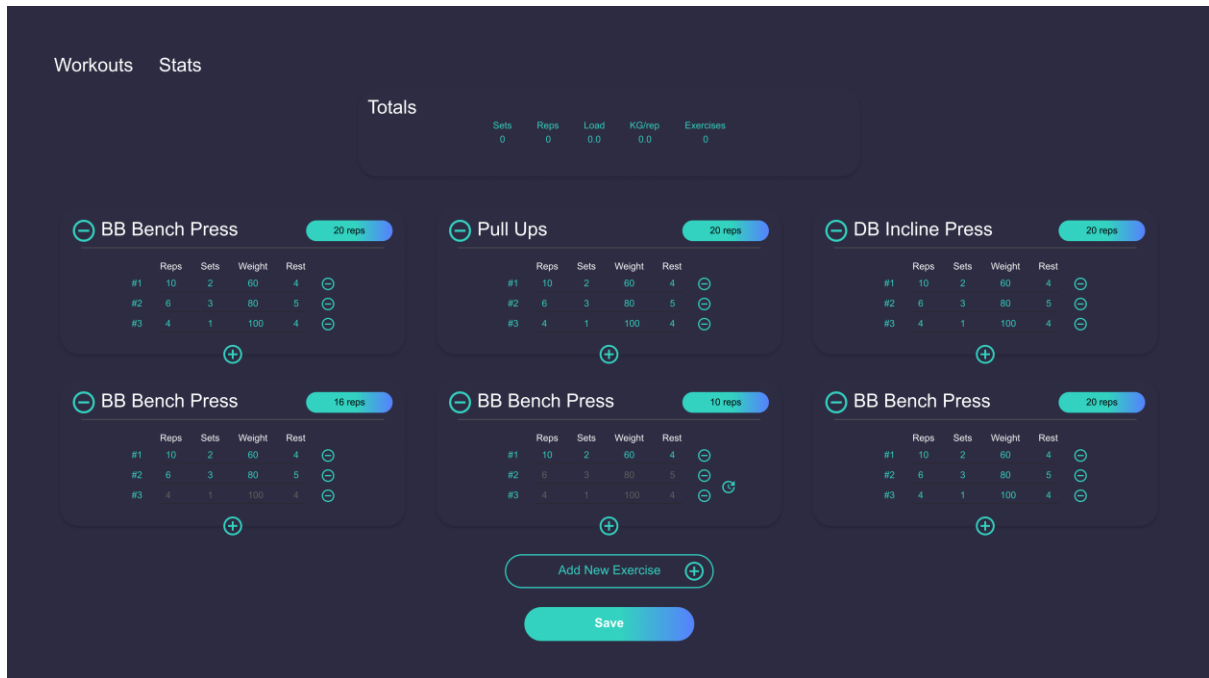
At the start if the user chooses a new workout this will all be blank, and the user can click on the 'Add New Exercise' button to add a exercise. Each set can be deleted and updated. Each exercise can be deleted as well. The total score of everything at the top will be automatically calculated. Most of the user stories are completed in this screen.

Stats:

For the stats page a user can click an exercise and a date. The graph will automatically change if these variables change.

2.2.1.4 Design Phone

For the phone I designed the main page. Here you can see the totals and the exercises in a smaller screen. The user can do everything the same as on desktop.

For the menu I asked a different user what would be better, a hamburger menu or just a normal menu. She said that the hamburger menu would give her an extra click which is unnecessary because I only have 2 links, I can use just a normal menu.

2.3 Test

2.3.1 In person usability study

For testing I visited a friend of mine. With my design I asked if he could use it. I didn't explain anything I just let him use my design. He was clicking easily through the application and didn't have points where he was searching for a button.

He liked the design a lot with the blue colors. He didn't have a problem with the application and said he can't wait to use it in person.
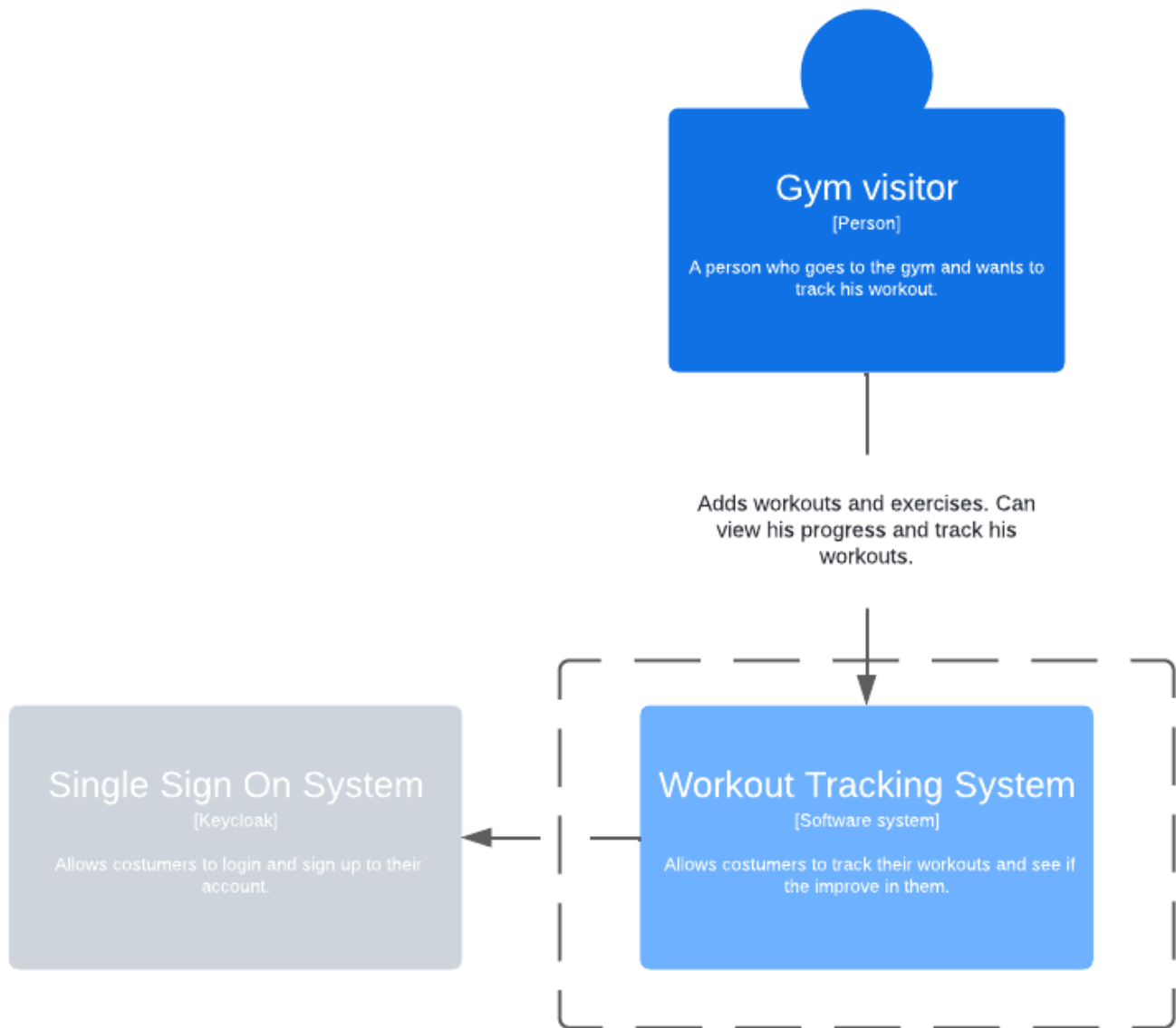
2.4 Listen

This heading I can't really fill in because I haven't fully made the application yet. When I am done with the application, I want it to test on multiple people and see if they all like it. When they have some problems, I want to fix them. This process is endlessly and always in motion.

3. Software Architecture

3.1 C4 Model

3.1.1 System Context diagram

3.1.2 Container diagram

Gym visitor
[Person]

A person who goes to the gym and wants to track his workout.

Adds workouts and exercises. Can view his progress and track his workouts.

PWA
[Container: Vue]

Provides all of the workout tracking functions to the customer via their web browser.

Makes API calls to [JSON/HTTPS]

Single Sign On System
[Keycloak]

Allows costumers to login and sign up to their account.

Makes API calls to [JSON/HTTPS]

Database
[Container: PostgreSQL]

Stores user information, workout data, acces logs, etc.

Reads from and writes to [JDBC]

API Application
[Container: Quarkus]

Provides workout tracking functionallity via a JSON/HTTPS API.

3.1.3 Component diagram

3.2 External API: Keycloak

The application involves an outside API for authentication, authorization, and delicate client information. I have decided to carry out this since it is most likely more secure than any implementation that I could make myself, and furthermore because it permits clients to sign in with just information I want or another service, like Google, Microsoft, Github or 13 other services. The disadvantage of implementing an outside API is that your application relies upon its continuous availability to work. As such, on the off chance that a service like Keycloak has an issue and is encountering margin time, your application would likewise experience the ill effects of corrupted usefulness or maybe it wouldn't function at all. In my utilization case notwithstanding, I feel that the advantages from using this external API offset the potential disadvantages.

3.3 Async

3.3.1 What is Async?

Async is doing multiple things at once instead of waiting on the last task. For example:
Imagine you're organizing a party and you want to send out invitations to all of your friends. You could send each invitation one at a time, but that could take a long time if you have a lot of friends. Instead of sending the invitations one at a time, you could use async to send them out in parallel. This means that you could start sending invitations to multiple friends at the same time, rather than waiting for one invitation to be sent before moving on to the next one.
This would allow you to send out all of the invitations more quickly, because you're not waiting for each one to be sent before moving on to the next one.

3.3.2 Async in software

When using async in software it means that you are doing multiple functions at once. Let's say the user does something and you need to process that, using async the user can be notified instead of waiting for the thing to happen. This gives a better user experience, and this can help the CPU and memory as well.

I my own project I use async for everything:

```
async logout() {
  await this.$keycloak.logout();
}
```

Here I wait for the SSO service to do its thing.

```
await axios.get( url: `${process.env.VUE_APP_BACK_END_API_URL}/workouts/user/`+this.$keycloak.profile.email, config)
  .then(async (response) => {
    this.workoutWithExercises = await response.data;
    this.workoutWithExercises.workoutIsValid = true;
  })
  .catch(async (error) => {
    this.workoutWithExercises.workoutIsValid = false;
    await this.showSnackBar( type: 'error', title: 'Error!', error.response.data);
  });
```

In this image you can see that I make an Axios request to my backend. Everything is async and await.