

# DHBWorkout - WebApp

## Team members

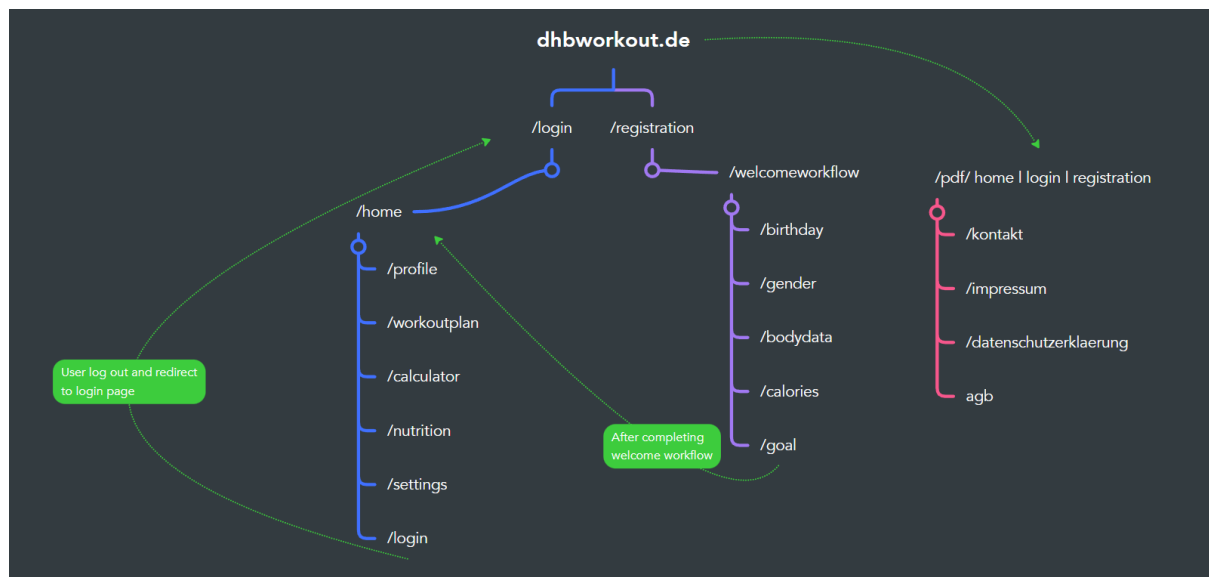
Leon Richter	Working on Login and Session Management via Cookies
Marc Grieser	Login view + basic logic, Registration view + basic logic, Created view after login, home view, profile view implemented Routing, created css for (almost) every page
Alex Schmidt	BMI calculator logic, website layout and design
Marvin Rieple	Working on making the WebApp more responsive and configured jenkins / docker for building / deployment.
Nico Merkel	Trainingsplan view and logic, trainingsplan css, website design in figma

## What is our website for?

It is for accessing the DHBWorkout service. It is a website that can be used to track your weight, set yourself a fitness goal and plan your exercises.

# Structure of our website:

The routing structure is seen in the graphic below:



If a user types dhwworkout.de in his browser, he will be redirected to dhwworkout.de/login.

From that point on the user will be able to further navigate to /registration.

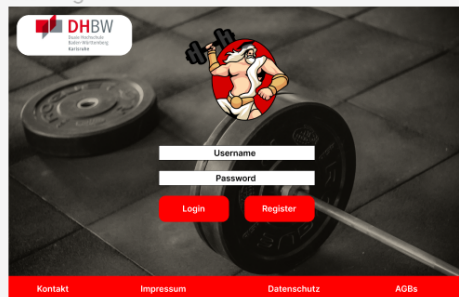
After login or registration the user will find himself on the /home view from where he can choose between profil, trainingsplan, bmi rechner, ernährung and einstellungen view with every view accessible from the navigation sidebar.

On every single view there is a footer from where the user can access PDFs about legal topics just like our legal notice, privacy policy or DHBWorkouts AGB's.

## Abstract Layout of our web page:

DHBWorkout is designed for a seamless user experience. With its intuitive single-page layout, users can easily navigate through the app's extensive features. Our simplified onboarding process allows users to define their profile based on key metrics like height, weight, and fitness goals, ensuring a personalized experience. Despite the app's diverse functionalities, we have organized it to be visually clear and user-friendly. DHBWorkout offers a range of features which are available in a clearly structured app.

## StartingPoint



**DHBW**  
Hochschule für Angewandte Wissenschaften  
Bamberg

**maxVorkout**

Möchtest du mich kennen?  
Ja, gerne!

Username  
Password

Login Register

Kontakt Impressum Datenschutz AGBs

## Register1



**DHBW**  
Hochschule für Angewandte Wissenschaften  
Bamberg

**maxVorkout**

Möchtest du mich kennen?  
Ja, gerne!

E-Mail  
Vorname  
Nachname  
Passwort  
Passwort wiederholen

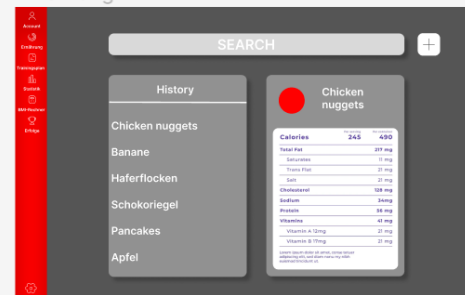
Zurück Weiter

Kontakt Impressum Datenschutz AGBs

## Account



## Ernährung



**SEARCH**

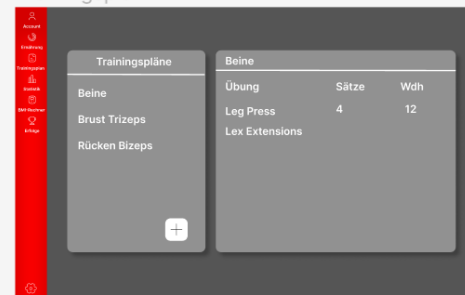
**History**

- Chicken nuggets
- Banane
- Haferflocken
- Schokoriegel
- Pancakes
- Apfel

**Chicken nuggets**

Calories	243	490
Total Fat	237 mg	
Saturated Fat	12 mg	
Trans Fat	21 mg	
Salt	21 mg	
Cholesterol	58 mg	
Sodium	24 mg	
Protein	28 mg	
Vitamin A	48 mg	
Vitamin A (IQR)	21 mg	
Vitamin A (IQR)	21 mg	

## Trainingsplan



**Trainingspläne**

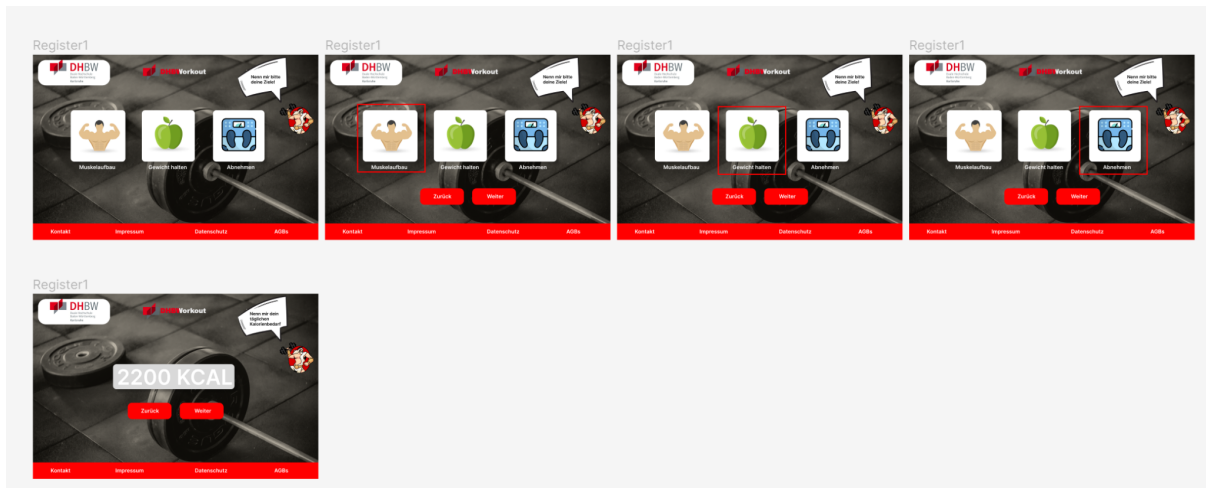
Beine Brust Trizeps Rücken Bizeps

**Beine**

Übung	Sätze	Wdh
Leg Press	4	12
Lex Extensions		

## BMI Rechner





Figma has been instrumental in transforming our design concept into a tangible reality. By utilizing its collaborative features, we have effectively communicated and implemented our layout goals, ensuring a shared understanding among the team. Figma's real-time collaboration capabilities have facilitated open communication, allowing us to fine-tune the user interface and create an exceptional user experience. With Figma as our design ally, we are confident in delivering a visually stunning and intuitive fitness app layout.

## Key-functions implemented by our back-end:

The backend is implemented as microservices. There are several APIs that can be called from the frontend either to login, register or enter your data (welcome-workflow). This enables the frontend to obtain data from or write data into our database in a secure way.

In order to reduce friction and deployment time, we decided to automate the building and deployment process of our web app. We utilize Jenkins and Docker for this purpose. Whenever a commit is pushed to the master branch on GitHub, a webhook is triggered. Jenkins then initiates a pipeline, which first retrieves all the changes from GitHub. It proceeds to build the Docker image, and during this process, the React app is also built and set up to be served via 'serve'. Once the image is built, it is uploaded to our Nexus instances. Subsequently, the test cases are executed within a separate throwaway Docker container, using a dedicated test script. This approach ensures portability across other Jenkins servers and eliminates the need to install npm on the Jenkins instance, thereby keeping it clean. After the tests are completed, the old WebApp container is stopped, and the new one is started.

The WebApp itself is not directly exposed to the internet. Instead, it is positioned behind an Nginx reverse proxy, which manages the domain and handles TLS encryption.

## How did we organize our source code?

The source code is organized and stored in a github repository. If someone wants to make a major change, a new branch is opened and later merged into the main branch. Before opening a branch for a change, the other team members are asked if someone else is already working on a similar problem, so bigger merge conflicts and unnecessary work is prevented.

## Any highlights?

The website is written with the react framework. This makes it easy to re-use certain components for different pages such as the sidebar or the footer. This also leads to the code being better organized and easier to work with.

Furthermore we implemented some test cases using the playwright framework by Microsoft. It enables us to run automated tests for multiple platforms (chrome, firefox, safari and mobile) after new source code is pushed to the master branch. After that, test reports are generated and if the tests are successful, the code is deployed.