Projekt Softwaretechnik / Medieninformatik

Meilenstein 2

15. November 2023

Projekt

eLunch

Firma: Steinbeis Embedded Systems Technologies GmbH

M. Kaan Asik	766381
Esra Balci	766561
Selim Cetin	766916
Vivian Schmiss	766236
Imran-Nur-Reyhan Sevinc	766668

Inhaltsverzeichnis

1 Vorwort

In einer Zeit, in der die Bedeutung von gesunder Ernährung und ausgewogenen Mahlzeiten immer mehr an Bedeutung gewinnt, haben wir uns das Ziel gesetzt, die Essensplanung zu revolutionieren und den Mitarbeitern der Steinbeis Embedded Systems Technologies GmbH dabei zu helfen, sich gesünder und energiegeladener zu fühlen.

Wir haben uns für die Firma Steinbeis, die sich das Projekt eLunch, ein Tool zur Planung eines Speiseplans überlegt hat, entschieden. Dafür hat die Firma bereits ein Produkt, das nun durch unseres abgelöst werden soll.

2 Generell

2.1 Zielgruppe

Grundsätzlich ist unsere Zielgruppe durch den Kunden des Projektes definiert. Unser Speiseplan-Tool richtet sich exklusiv an die Mitarbeiter der Firma und wird entwickelt, um deren Essenserlebnis zu verbessern und einen gesunden Lebensstil zu fördern. Diese Software wird maßgeschneidert für die Bedürfnisse und Anforderungen der Mitarbeiter. Unser Speiseplan-Tool wird für alle Mitarbeiter der Steinbeis GmbH zugänglich sein. Spezifischer ist unsere Zielgruppe also durch die Mitarbeiter und den Admin des Unternehmens definiert.

2.2 Probleme

Als grundsätzliches Problem stellt sich uns die Übersichtlichkeit der gesamten Oberfläche. Die Veranschaulichung der Gerichte innerhalb des Monats, das Eintragen bzw. das Ändern von Gerichten und die Eingabe der Zutaten in eine einfach lesbare Form, stellt also unsere größte Herausforderung dar. Hinzuzufügen ist, dass wir keine Experten für die von uns verwendeten Technologien sind. Zweifelsohne haben wir uns im Vorfeld für Technologien und Programmiersprachen entschieden, die uns liegen und bei deren Verwendung wir uns sicher fühlen. Dennoch können wir nicht mit Sicherheit voraussagen, dass wir im Projekt nicht Problemen gegenüberstehen werden, die unser aktuelles Können herausfordern und deren Lösung mehr Zeit als geplant in Anspruch nehmen wird.

2.3 Eigenschaften

Wir schreiben eine Web-Applikation, die das gemeinsame Kochen und dessen Planung in einer Firma simplifizieren soll. In einer MongoDB werden alle Daten gespeichert, welche mit Node.js implementiert werden. Die genaue Darstellung der Webseite werden wir anhand HTML, CSS und React.js veranschaulichen. Es gibt genau zwei Nutzergruppen, darunter sind die Mitarbeiter und der Admin. Deren Rechte und Anwendungsfälle werden unter Kapitel 4 User Stories genau erläutert.

3 Team-Organisation

In unserem Team soll jeder bei allem mal dabei gewesen sein und alles mal gesehen haben. Jeder muss Code geschrieben haben, muss wissen, wie unser Backend funktioniert, sollte die Dokumentation kennen und einen Teil davon geschrieben haben und jeder sollte in Figma mitgewirkt haben.

Dennoch gibt es grobe Rollen in unserem Team. Für das Backend sind Selim und Kaan zuständig. Das bedeutet, dass sie dafür verantwortlich sind, diese zu koordinieren und aufzusetzen. Zusätzlich kümmern Sie sich um die API.

Esra ist im Team die Projektleiterin. Sie koordiniert das gesamte Team und ist zuständig für die Kommunikation mit Steinbeis und organisiert die Meetings.

Für das Protokollieren der Meetings werden wir uns jede Woche abwechseln. Die Protokolle müssen immer aktuell und so ausführlich geschrieben sein, dass ein Projektfremder sofort weiß, worum es geht und auf welchem Stand das Projekt aktuell ist. Für die Dokumentation ist jeder zuständig, die einzelnen Dokumentationsteile werden von jedem Teammitglied einzeln zusammengestellt. Esra, Imran und Vivian formatieren, überprüfen und ergänzen diese anschließend.

Für Figma sind Imran, Esra und Vivian zuständig. Die benötigten Mockups waren zu koordinieren, zu erstellen und zu einem einheitlichen Design zusammen zu fügen.

Das Kanban-Board haben wir auf GitHub hinzugefügt. GitHub liegt in Esras Verantwortung. Sie hat das Projekt aufgesetzt und wird es pflegen. Die Zuweisung von Issues, Post-Requests bearbeiten, Merge-Requests bearbeiten bzw. Personen zuweisen und allgemein darauf achten, dass das Projekt strukturieren auf Git gepflegt ist, liegt in Esras Hand.

Beim Coden des Frontends sind ebenfalls Imran, Esra und Vivian zuständig.

Jeder arbeitet an allem mit und leistet prozentual gleich viel für das Projekt.

4 User Stories

Als Mitarbeiter möchte ich auf die Wochen- und Monatsansicht vom Speiseplan zugreifen können, um beim Speiseplan einsehen zu können, was es zum Essen gibt.

Als Mitarbeiter möchte ich in der Wochenansicht neue Gerichte hinzufügen, wenn noch keine eingetragen wurden, um dieses Gericht an dem eingetragenen Tag kochen zu können.

Als Mitarbeiter möchte ich mich bei einem Gericht zum Mitessen ein- oder austragen, um mitzuessen.

Als Mitarbeiter möchte ich auf meine Finanzübersicht zugreifen können, um meine Schulden bei meinen Kollegen zu begleichen, falls diese vorhanden sind.

Als Admin möchte ich eingetragene Gerichte bearbeiten oder neue hinzufügen, um zum Beispiel eine Zutat beim Gericht zu ändern.

Als Admin möchte ich die Ein- und Ausgaben meiner Mitarbeiter bearbeiten können, um ihre Konten auszugleichen.

Als Admin möchte ich die Statistik- und Mitarbeiteransicht einsehen können, um einen neuen Mitarbeiter hinzuzufügen.

5 Flow Chart

Wir haben ein Flow Chart vorbereitet, siehe Abbildung 1, welches aufführt, wie die verschiedenen Funktionen auf der Webseite mit anderen Screens verknüpft sind.

Abbildung 1: Flow Chart zum Ablauf der Steuerung der Webseite

6 Funktionsumfang

Aufgabe	Aufwand in Stunden
API-Definition in Open API Format	10 h
Backend Routing implementieren	60 h
Backend Verbindungsaufbau zur DB implementieren	10 h
Backend Tests (Unit Tests)	30 h
Frontend Grundgerüst für UI erstellen	40 h
Frontend UI abschließend implementieren, Buttons, Daten visualisieren…	60 h
UI/Mockup Design	10 h
Frontend Tests	20 h
User Testing	10 h
Projektmanagement in Git mit Branches, Mergekonfliktlösung…	20 h
Dokumentation	20 h
Präsentationsvorbereitung	10 h
Gesamtaufwand	300 h

7 User Interface

7.1 Desktop Ansicht

Im Folgenden sind unsere Mockups in den Abbildungen 2, 3, 4 und 5 zu sehen.

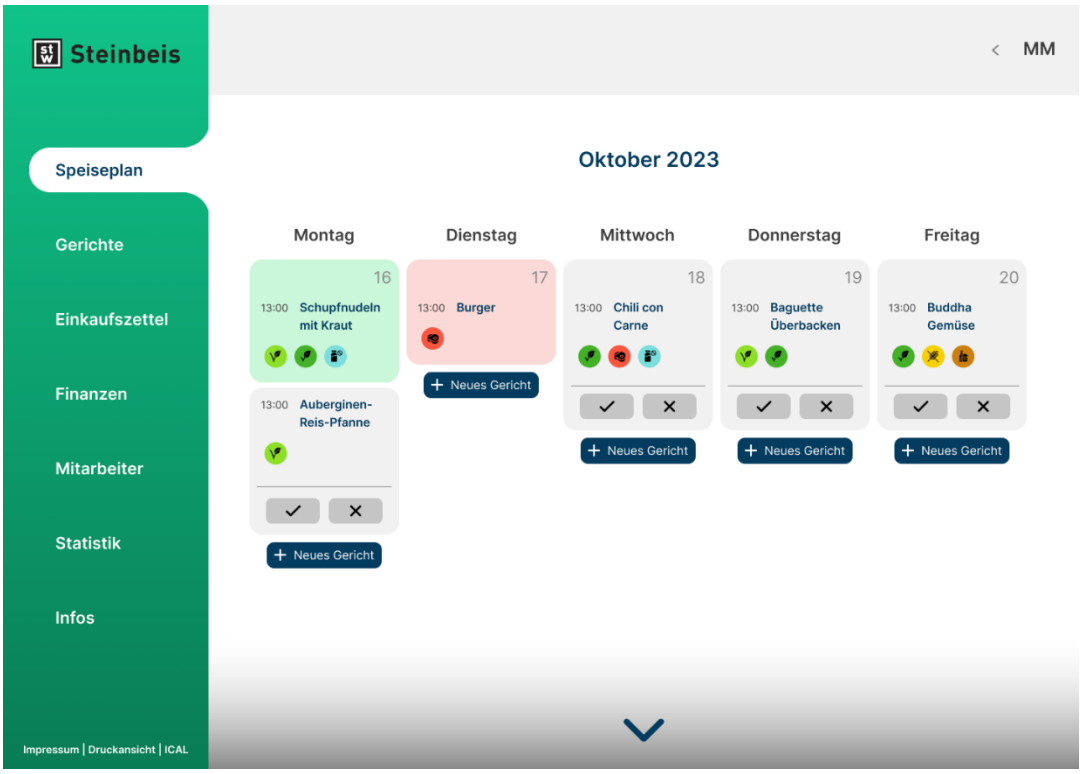

Abbildung 2: Wochenansicht vom Speiseplan

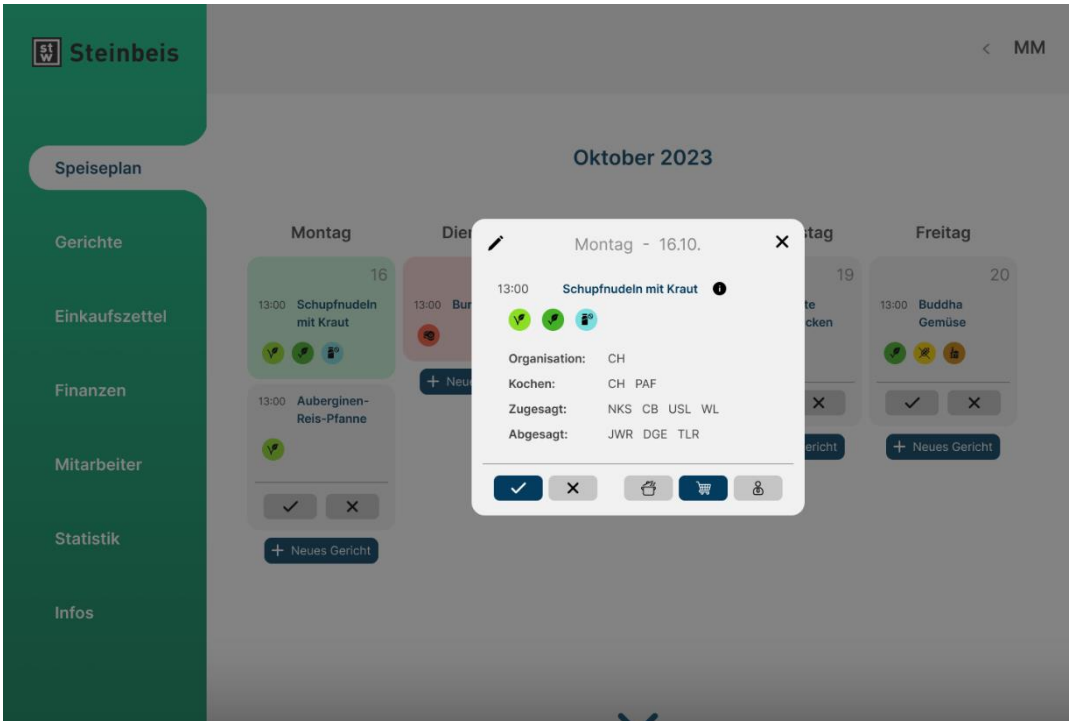

Abbildung 3: Detailansicht von einem Tagesgericht

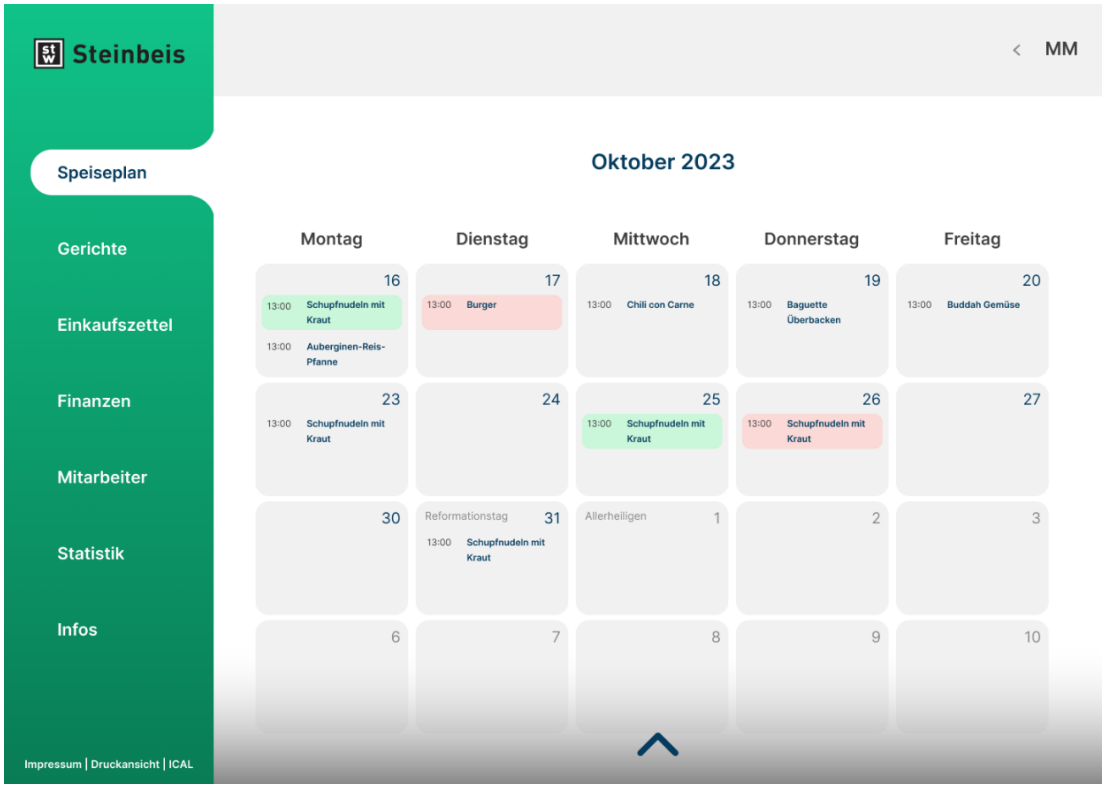

Abbildung 4: Monatsansicht vom Speiseplan

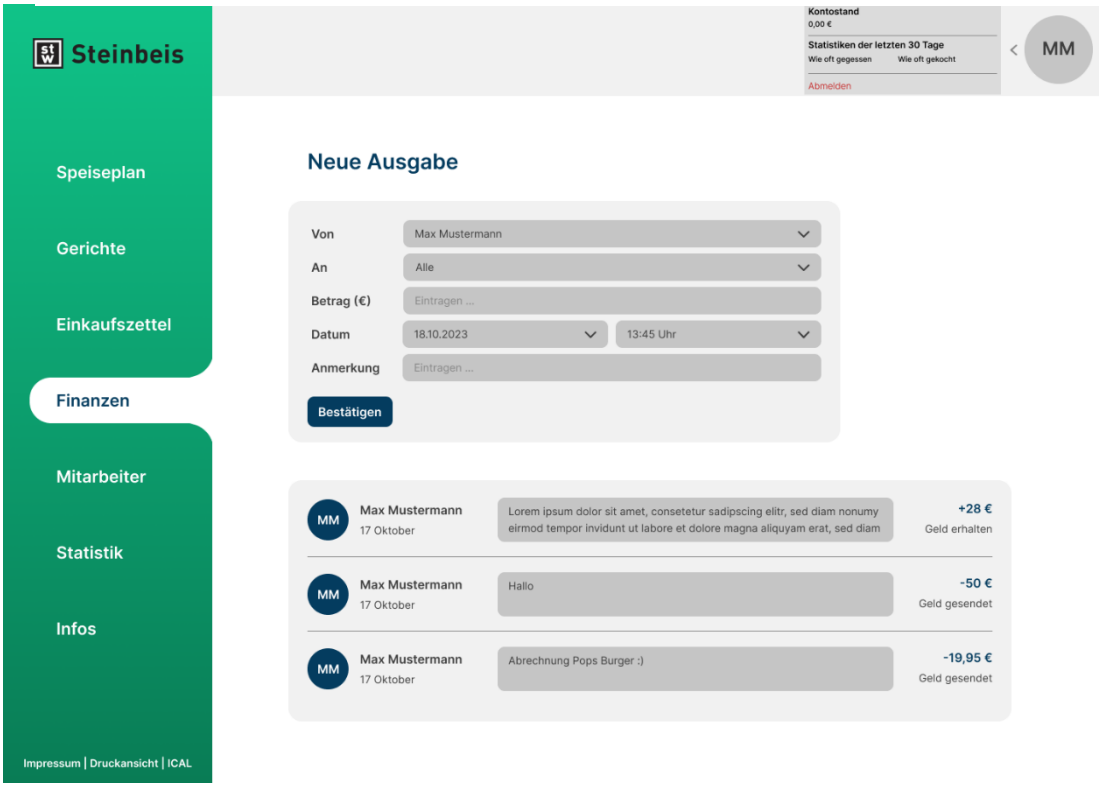

Abbildung 5: Finanzansicht

7.2 Mobile Ansicht

Im Folgenden sind unsere mobilen Mockups in den Abbildungen 6 bis 14 zu sehen.

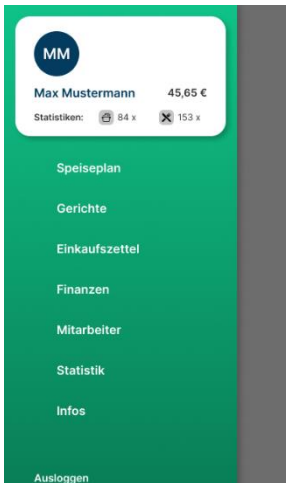

Abbildung 6: Navigationsbar

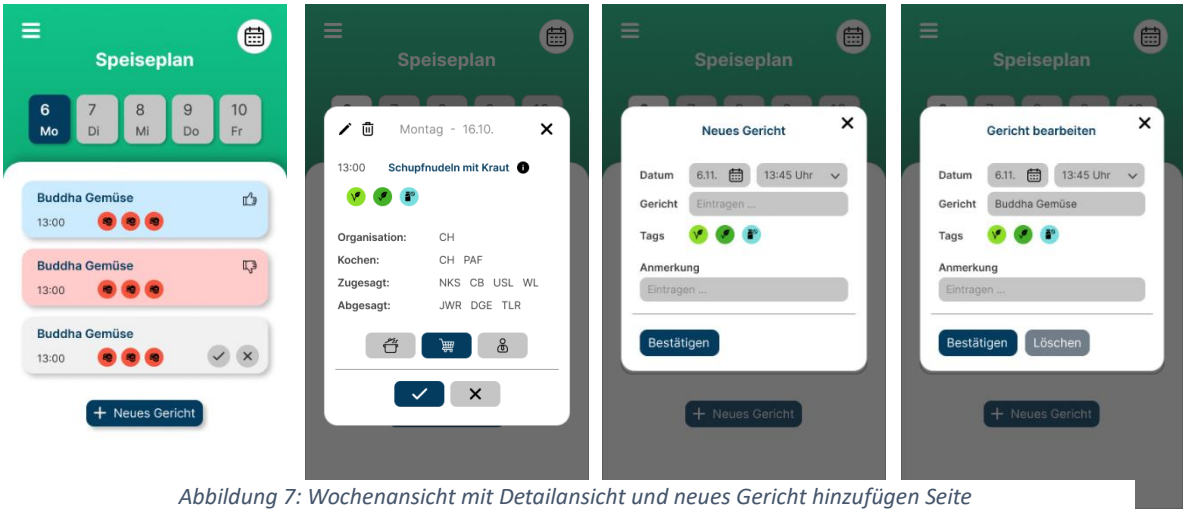

Abbildung 7: Wochenansicht mit Detailansicht und neues Gericht hinzufügen Seite

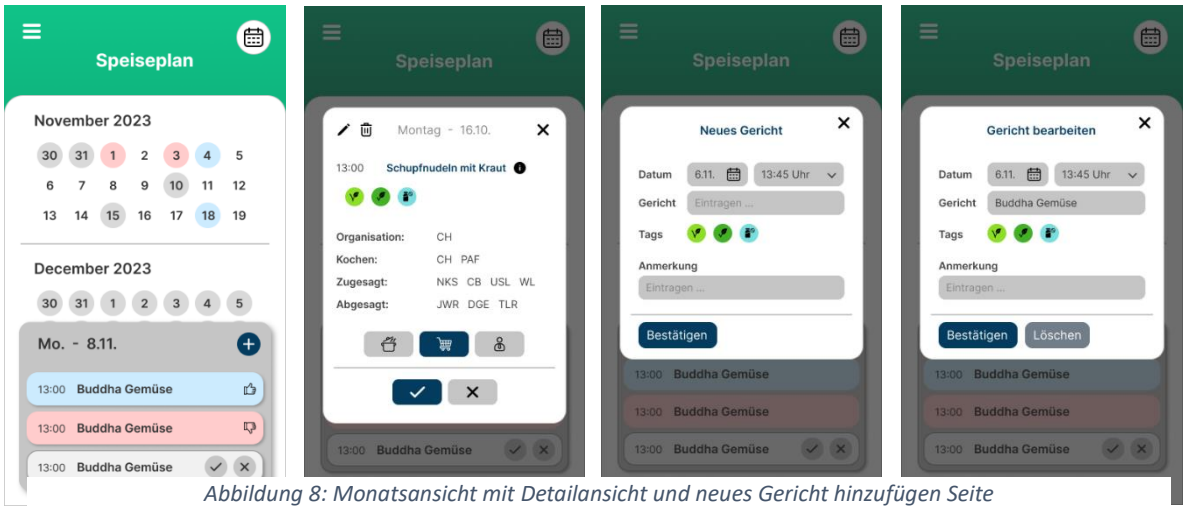

Abbildung 8: Monatsansicht mit Detailansicht und neues Gericht hinzufügen Seite

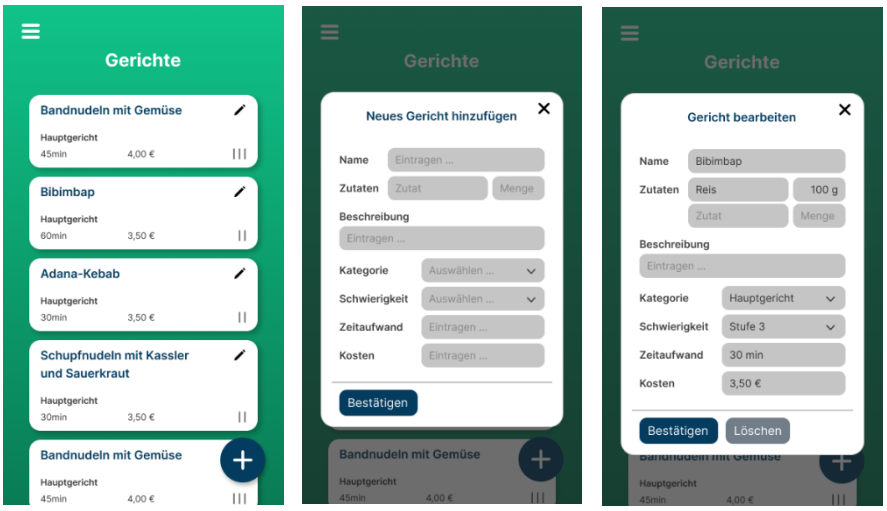

Abbildung 9: Gerichte Seite mit neuem Gericht hinzufügen Seite

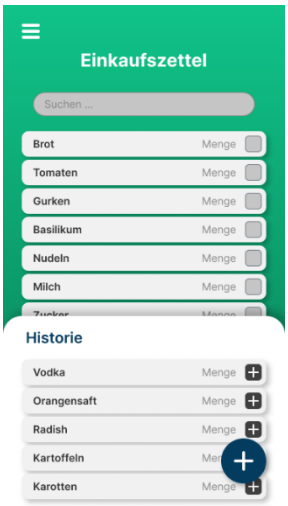

Abbildung 10: Einkaufszettel

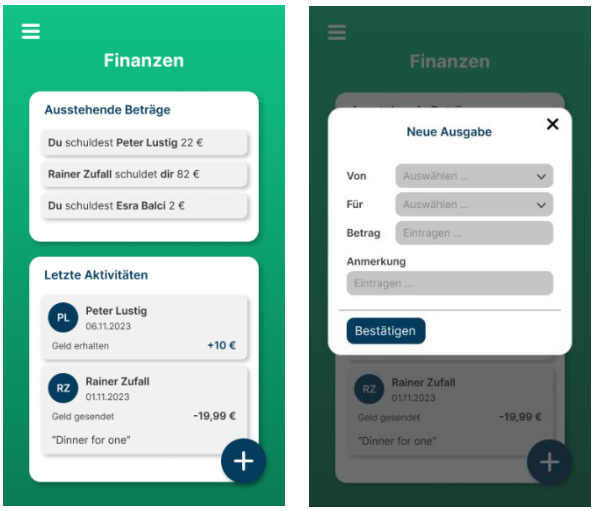

Abbildung 11: Finanzansicht mit neuen Ausgaben

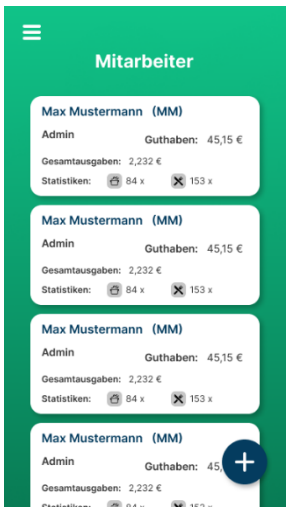

Abbildung 12: Mitarbeiteransicht

Abbildung 13: Statistikansicht

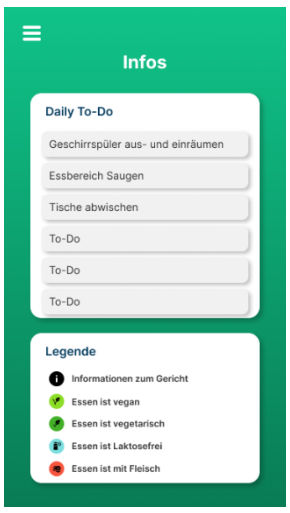

Abbildung 14: Infoseite

8 Technologien

Für die Architektur bzw. Technologien möchten wir sowohl für das Backend als auch für das Frontend JavaScript als Programmiersprache nutzen. Dabei nutzen wir Node.js als Runtime Environment für unser Backend und React.js als Frontend User Interface. Für die Datenhaltung möchten wir die MongoDB in Kombination mit der Mongoose Library nutzen.

8.1 Frontend

8.1.1 React.js

Für das Frontend haben wir uns für React.js entschieden. React.js ist eine JavaScript Library, die von Facebook entwickelt wurde. Mit ihr ist es möglich, interaktive, dynamische und komplexe Webseiten zu gestalten. Sie ist bekannt für das Rendering-Prinzip, welches dynamische Seiten ermöglicht, ohne die gesamte Seite neu laden zu müssen.

8.1.2 Material UI

Zudem verwenden wir Material UI, welches uns vorgefertigte Komponenten anbietet, die wir anschließend auf unsere Bedürfnisse anpassen. Dies ermöglicht uns ein schnelles und effektives Erstellen des User-Interfaces. Die umfangreiche Dokumentation und die gute Benutzerfreundlichkeit unterstützt uns dabei

8.2 Backend

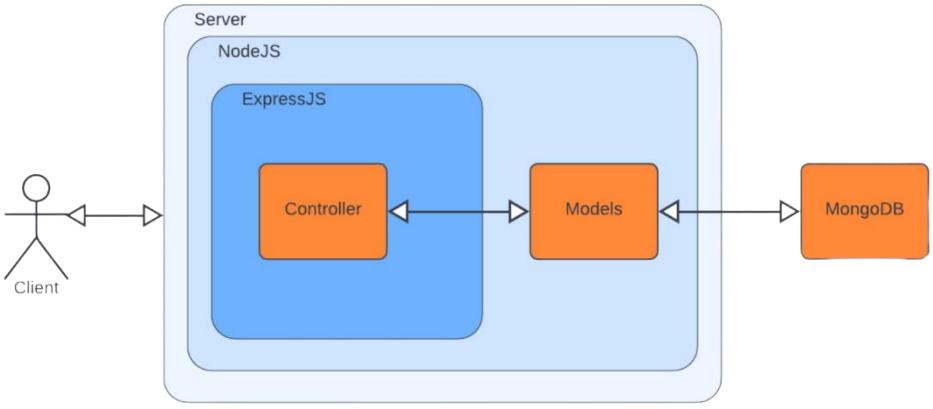

Abbildung 15: Beschreibt den Aufbau unseres Backends

8.2.1 Node.js

Für das Backend verwenden wir Node.js. Es ist ein Open-Source Runtime Environment für JavaScript, die es ermöglicht, serverseitige Anwendungen mit JavaScript zu entwickeln. Node.js verwendet eine ereignisgesteuerte, nicht blockierende I/0-Architektur, die es dem Entwickler ermöglicht, skalierbare und schnelle Web-Applikationen zu erstellen. Es wird meistens für die Erstellungen von Webanwendungen und APIs verwendet.

Wir haben uns für Node.js entschieden, da es sehr verbreitet und beliebt ist. Außerdem gibt es eine große Auswahl an Third-Party-Tools, die sehr einfach zu installieren und nutzen sind.

8.2.2 Express.js

Express ist ein Web-Application-Framework für Node.js. Es ermöglicht den Entwicklern, robuste APIs und Webserver auf saubere und einfachere Weise zu erstellen. Es ist ein lightweight Paket, dass die Hauptfunktionen von Node.js nicht verdeckt.

8.2.3 MongoDB

MongoDB ist eine NoSQL-Datenbank, die als Datenbanktechnologie weit verbreitet ist. Im Gegensatz zu traditionellen Datenbanken, speichert MongoDB Daten in ein dokumentorientiertes Format. Dies ermöglicht mehr Flexibilität und Skalierbarkeit. MongoDB wird oft bei modernen Webanwendungen und mobilen Apps verwendet. Deswegen ist es auch bekannt in einem Stack namens “MERN”: MongoDB, Express, React.js, und Node.js.
Als Unterstützung für die Nutzung der MongoDB via Backend werden wir die Mongoose Library verwenden.
Aufgrund der Popularität und somit vorhandener Dokumentation und Support haben wir uns für MongoDB entschieden.

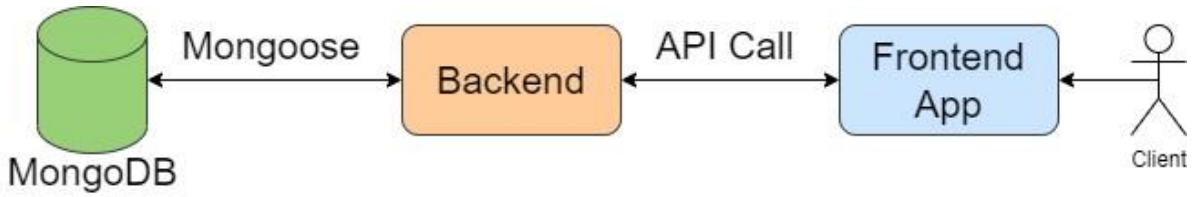

Abbildung 16: MERN Struktur

8.2.4 API

Für die Dokumentation unserer API nutzen wir OpenAPI.

8.3 Authentifizierung und Autorisierung

8.3.1 Keycloak

Es ist eine Open-Source-Software zur Identitäts- und Zugriffsverwaltung, die sich an moderne Anwendungen und Dienste richtet. Es bietet eine Vielzahl von Funktionen, darunter Benutzerföderation, starke Authentifizierung, Benutzerverwaltung, feingranulare Autorisierung und mehr. Mit Keycloak können Entwickler die Authentifizierung in Anwendungen integrieren und Dienste sicher bereitstellen, ohne sich mit der Speicherung von Benutzern oder der Authentifizierung von Benutzern befassen zu müssen.

8.4 Containerisierung

8.4.1 Docker

Docker ist eine Open-Source-Software, die es Entwicklern ermöglicht, Anwendungen in Containern zu erstellen, zu verteilen und auszuführen. Container sind eine Art von Virtualisierung, die es ermöglicht, Anwendungen in einer isolierten Umgebung auszuführen, die unabhängig von der zugrunde liegenden Infrastruktur ist. Docker bietet eine Reihe von Tools und Diensten, die es Entwicklern erleichtern, Container-Anwendungen zu erstellen, zu testen und bereitzustellen.

Eine der wichtigsten Funktionen von Docker ist die Möglichkeit, Images zu erstellen und zu verteilen. Ein Image ist eine Vorlage, die verwendet wird, um einen Container zu erstellen [1]. Docker bietet eine Möglichkeit, Images in einem zentralen Repository zu speichern und zu teilen, das als Docker Hub bezeichnet wird [1]. Entwickler können

Images von Docker Hub herunterladen und als Basis für ihre eigenen Container-Anwendungen verwenden [1].

Eine weitere wichtige Funktion von Docker ist die Möglichkeit, Dockerfiles zu verwenden, um Images zu erstellen [1]. Ein Dockerfile ist eine Textdatei, die eine Reihe von Anweisungen enthält, die Docker verwenden kann, um ein Image zu erstellen. Dockerfiles sind eine effektive Möglichkeit, Images zu erstellen, da sie es Entwicklern ermöglichen, Images zu erstellen, die genau auf die Bedürfnisse ihrer Anwendungen zugeschnitten sind.

8.5 Softwarearchitektur

8.5.1 Strukturübersicht

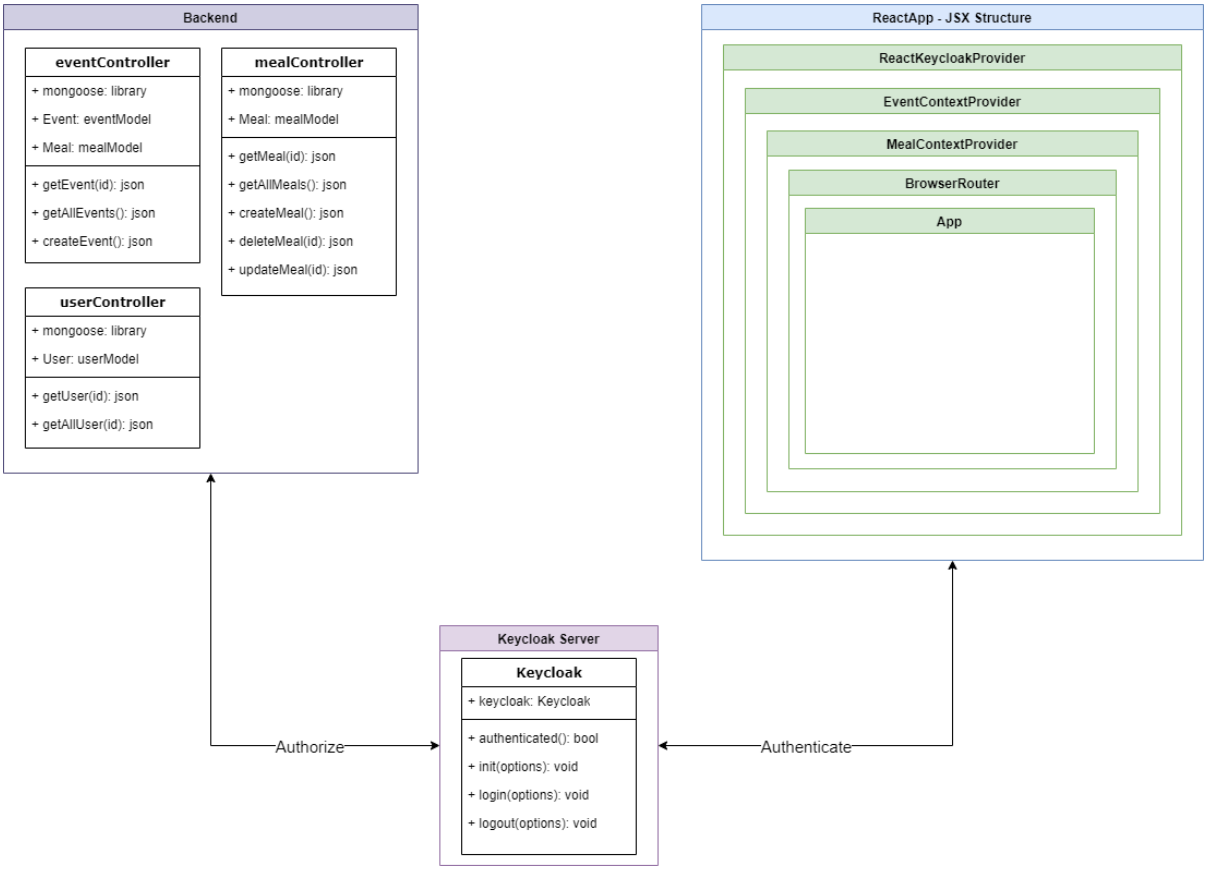

Abbildung 17: Strukturübersicht

8.5.2 Verhaltenssicht

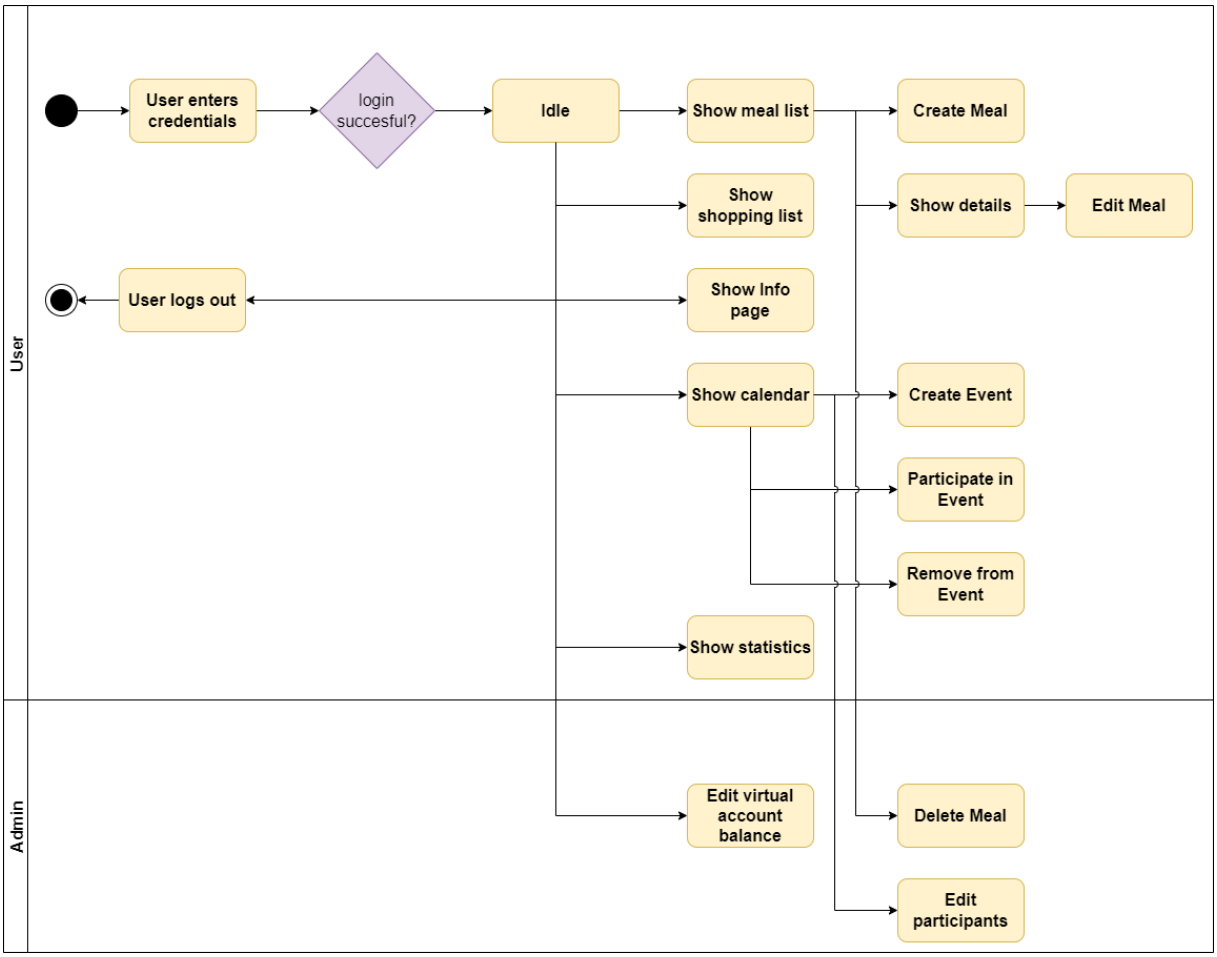

Abbildung 18: Verhaltenssicht

8.5.3 Verteilungssicht

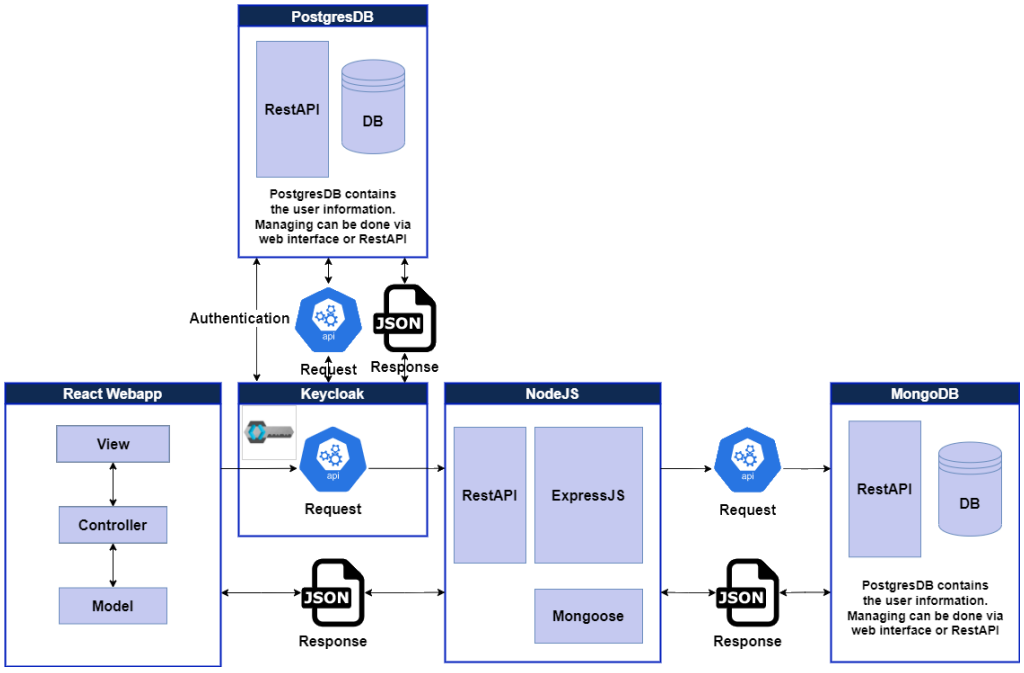

Abbildung 19: Verteilungssicht

8.6 Schnittstellentechnologien

8.6.1 Restful-API

RESTful-API ist eine Schnittstellenbeschreibung, die auf dem REST-Architekturstil basiert. REST steht für Representational State Transfer und ist ein Softwarearchitekturstil, der die Bedingungen festlegt, unter denen eine API arbeiten sollte. RESTful-APIs sind eine Möglichkeit, Webdienste zu implementieren, die auf den REST-Architekturstil folgen. Sie ermöglichen es Entwicklern, auf einfache und flexible Weise auf Webdienste zuzugreifen, ohne dass eine Verarbeitung erforderlich ist.

RESTful-APIs verwenden ein zustandsloses Protokoll, um textuelle Darstellungen ihrer Online-Ressourcen zum Lesen und Verarbeiten zur Verfügung zu stellen. Die APIs sind sicher, zuverlässig und effizient und ermöglichen es Entwicklern, auf verschiedene Funktionen zuzugreifen, z.B. Benutzer- und Gruppenverwaltung, Authentifizierung und Autorisierung.

8.6.2 Unsere API Endpoints (CRUD Functions)

POST Requests

- **/api/meals/** Creates a new meal with attributes in DB (Name, isVegan, etc.)
- **/api/events/** Creates a new event with attributes in DB (Meal, Date, User, Rollen)

PATCH (UPDATE) Requests

- **/api/meals/:id** Updates an existing meal with the given id in DB
- **/api/events/:id** Updates an existing event with the given id in DB

GET Requests

- **/api/meals/** Retrieves all meals from DB
- **/api/meals/:id** Retrieves the meal with the given id from DB
- **/api/events/** Retrieves all events from DB
- **/api/events/:id** Retrieves the event with the given id from DB
- **/api/users/** Retrieves all users from DB
- **/api/users/:id** Retrieves the user with the given id from DB

DELETE Requests

- **/api/meals/:id** Deletes a meal with the given id
- **/api/events/:id** Deletes an event with the given id

8.7 Datenbank

8.7.1 Logisches Datenmodell

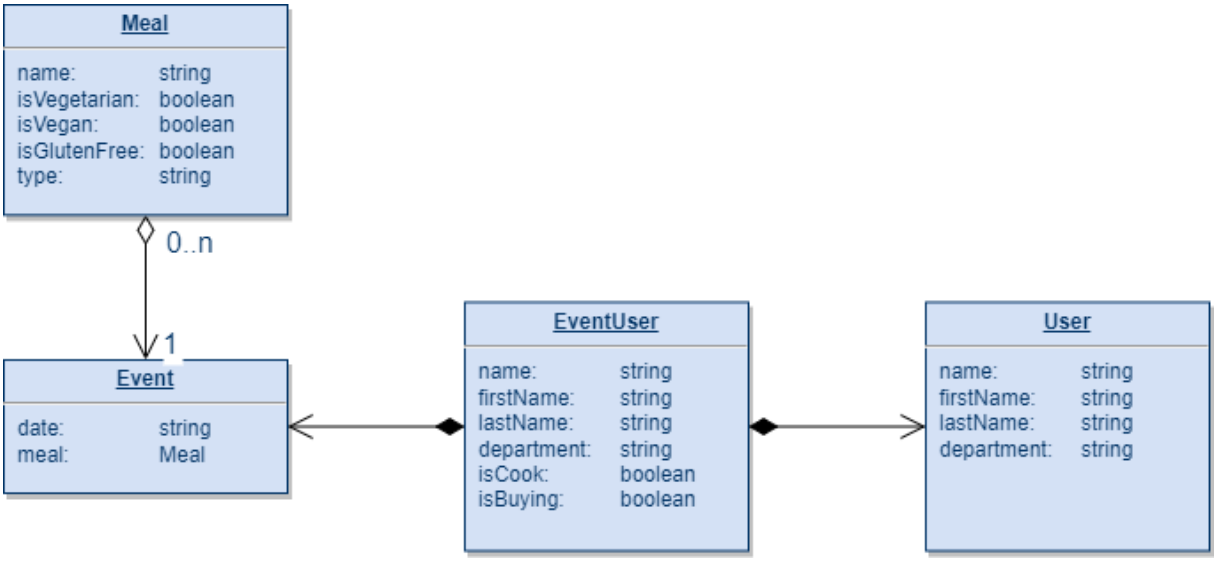

Abbildung 20: Logisches Datenmodell

8.7.2 Physisches Datenmodell

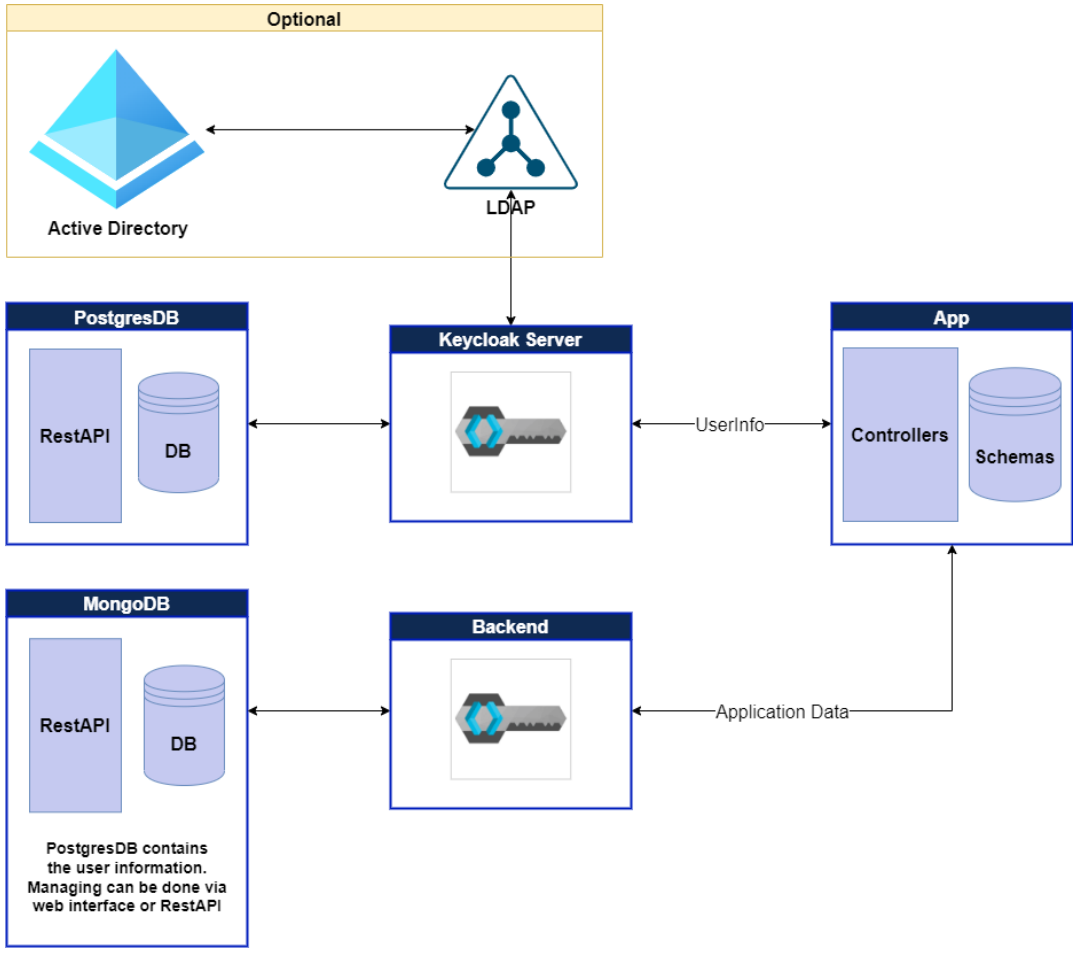

Abbildung 21: Physisches Datenmodell

9 Quellenverzeichnis

9.1 Online-Quellen

- https://www.hs-esslingen.de
- https://www.figma.com
- https://app.diagrams.net/
- https://www.lucidchart.com