Event-Planer

Felix Hoffmann, Baran Bickici, Sami Gökpinar, Ergün Bickici

29. April 2025

Inhaltsverzeichnis

1 Einführung **5**

2 Technologien **5**
2.1 Full-Stack Framework 5
2.1.1 Frontend 5
2.1.2 Backend 5
2.2 Datenbank 6
2.3 Authentifizierung 6
2.4 Deliverable 6

3 Requirements Specification **7**
3.1 Zielgruppe 7
3.2 Kundenbefragung 7
3.3 Feature-Map 7
3.4 User Stories 8
3.4.1 Anmeldung 8
3.4.2 Events 8
3.4.3 Wünsche 8
3.4.4 Umfragen 9

4 Funktionsumfang **10**
4.1 Anmeldung und Registrierung 10
4.2 Events 10
4.3 Wünsche 11

5 UI Entwürfe **12**
5.1 Login 12
5.2 Events 12
5.2.1 Event-Feed 12
5.2.2 Meine Events 13
5.2.3 Event erstellen 13
5.2.4 Umfrage erstellen 14
5.2.5 Umfrage bearbeiten 14
5.2.6 Meine Umfragen 15
5.2.7 Event bearbeiten 15
5.2.8 Event löschen 16
5.3 Wünsche 16
5.3.1 Wünsche-Feed 16
5.3.2 Meine Wünsche 17
5.3.3 Wunsch erstellen 17

6 Aufwandsschätzung **18**
6.1 Anforderungsanalyse 18
6.2 Entwurfsphase 18
6.3 Implementierung & Testing 18
6.4 Bereitstellung und Projektabschluss 18

7 Projektmanagement: "A la Scrum" 19

8 Systemarchitektur 20
8.1 Allgemein 20
8.2 Struktursicht 21
8.3 Verhaltenssicht 22
8.4 Verteilungssicht 23

9 Schnittstellentechnologien 24
9.1 Schnittstellenbeschreibung 24

10 Datenbank 25
10.1 Logisches Datenmodell 25
10.1.1 Entity Relationship Modell 25
10.1.2 Nutzerverwaltung mit Supabase Auth 25
10.1.3 Haupttabellen und ihre Aufgabenbereiche 26
10.1.4 Hilfstabellen für komplexe Beziehungen 26
10.2 Physisches Datenmodell 26

11 Literaturverzeichnis 27

Abbildungsverzeichnis

1 Feature-Map 7
2 Login 12
3 Event-Feed 12
4 Meine Events 13
5 Event erstellen 13
6 Umfrage erstellen 14
7 Umfrage bearbeiten 14
8 Meine Umfragen 15
9 Event bearbeiten 15
10 Event löschen 16
11 Wünsche-Feed 16
12 Meine Wünsche 17
13 Wunsch erstellen 17
14 Systemarchitektur 20
15 Struktursicht 21
16 Verhaltenssicht 22
17 Verteilungssicht 23
18 Logisches Datenmodell 25

1 Einführung

Im Rahmen des Softwareprojekts im 4. Semester des Studienganges Softwaretechnik und Medieninformatik wird über das ganze Semester ein Projekt mit einer Partnerfirma, die als Kunden agieren durchgeführt. Dieses Projekt wird mit der Firma Pep-Digital die sich ein Produkt wünschen mit dem Titel "Event-Planer". Dieses Produkt soll eine Web-Applikation werden, in denen Mitarbeiter der Firma Pep-Digital Events erstellen und das Event planen können z.B. mit Umfragen, Datum des Events und Teilnehmer. Dazu hat man auch die Möglichkeit Wünsche zu äußern aus denen man Events erstellen kann. Als erstes wird sich erst auf Schulungsevents konzentriert und in der Zukunft hat man die Möglichkeit auch andere Arten von Events zu erstellen.

2 Technologien

2.1 Full-Stack Framework

Für die Umsetzung des Projekts wurde Next.js als Full-Stack-Framework gewählt. Next.js basiert auf React und bietet eine vollständige Lösung für die Entwicklung von Webanwendungen, indem es sowohl Frontend- als auch Backend-Funktionalitäten integriert. Mit Next.js können Entwickler sowohl serverseitiges Rendering (SSR) als auch statische Seitengenerierung (SSG) nutzen. Zudem bietet es eine einfache Möglichkeit, API-Routen zu erstellen, was es zu einer hervorragenden Wahl für Full-Stack-Entwicklungen macht. Die Verwendung von TypeScript sorgt für eine typsichere und fehlerarme Entwicklung, sowohl im Frontend als auch im Backend.

2.1.1 Frontend

Im Frontend nutzt Next.js React. React ist eine quelloffene JavaScript-Bibliothek, die das Erstellen der Benutzeroberfläche schnell und dynamisch macht. Anhand von React können Web- und Mobile-Anwendungen mit derselben Codebasis erstellt werden, ohne jegliche Formatierung. Die Codierung erfolgt in TypeScript, was durch statische Typisierung und bessere Fehlererkennung das Entwickeln sicherer und effizienter macht.

2.1.2 Backend

Das Backend in Next.js wird durch das integrierte API-Routing realisiert, das auf Node.js basiert. Durch diese Integration können Entwickler API-Routen direkt innerhalb der Next.js-Anwendung erstellen, ohne einen separaten Server benötigen zu müssen. Diese Routen ermöglichen es, serverseitige Logik auszuführen, wie etwa das Abrufen von Daten aus einer Datenbank oder das Bearbeiten von Anfragen. Da Next.js auf Node.js aufbaut, können alle leistungsfähigen Funktionen von Node.js genutzt werden, während TypeScript die Entwicklung durch statische Typisierung und Fehlererkennung verbessert. Dies sorgt für eine konsistente und sichere Entwicklung sowohl im Frontend als auch im Backend innerhalb derselben Codebasis.

2.2 Datenbank

Für die Datenbank wurde PostgreSQL in Kombination mit Supabase gewählt. PostgreSQL bietet zahlreiche Vorteile für die Nutzung in Webanwendungen. Es ist besonders leistungsfähig und skalierbar, was es ideal für die Verwaltung großer Datenmengen und viele gleichzeitige Nutzer macht. Mit seiner hohen Erweiterbarkeit ermöglicht es die Anpassung der Datenbank an spezifische Anforderungen, etwa durch benutzerdefinierte Datentypen und Funktionen. Die robuste Datenintegrität sorgt dafür, dass die Daten sicher und konsistent bleiben, auch bei komplexen Transaktionen. Dank fortschrittlicher Indexierungs- und Abfrageoptimierungen können Webanwendungen schnell auf große Datenmengen zugreifen. Zudem ist PostgreSQL ACID-konform, was bedeutet, dass es zuverlässige und fehlerfreie Transaktionen garantiert. Diese Eigenschaften machen PostgreSQL zu einer bevorzugten Wahl für Webanwendungen, die hohe Leistung, Flexibilität und Datenintegrität erfordern. Supabase ist eine Open-Source-Plattform, die Entwicklern hilft, moderne Anwendungen schnell und einfach zu erstellen. Sie bietet eine Vielzahl von Funktionen wie Datenbanken, Authentifizierung, Echtzeit-Updates und APIs, um skalierbare und sichere Anwendungen zu entwickeln, ohne viel Aufwand in die Backend-Programmierung investieren zu müssen.

2.3 Authentifizierung

Supabase wird für die Authentifizierung genutzt, da es eine einfache und sichere Möglichkeit bietet, Nutzer in eine Anwendung zu integrieren. Es unterstützt Single Sign-On (SSO) und externe Identitätsanbieter wie Google und Microsoft Azure, sodass sich Benutzer bequem mit ihren bestehenden Konten anmelden können. Durch die Integration von Microsoft Azure können sich insbesondere Mitarbeiter direkt mit ihren Microsoft-Accounts authentifizieren. Dies erleichtert den Zugriff auf die Anwendung, da keine separaten Anmeldeinformationen erstellt werden müssen, und sorgt gleichzeitig für höhere Sicherheit und bessere Verwaltungsmöglichkeiten durch zentrale Benutzerkontrollen in Azure.

2.4 Deliverable

Als Deliverable wird in diesem Projekt Docker verwendet, da so sicher gegangen werden kann, dass die Anwendung in einer konsistenten Umgebung ausgeführt wird, unabhängig von den zugrunde liegenden Betriebssystemen oder Hardwarekonfigurationen.

3 Requirements Specification

3.1 Zielgruppe

Mit dem Endprodukt des Projekts, wird den Mitarbeitenden der Firma Pep-Digital eine Web-Applikation für die Erstellung und Planung von Events zur Verfügung gestellt. Da diese Web-Applikation firmenspezifisch und nicht für den offenen Markt entwickelt wird, ist die Zielgruppe dieser App sehr reduziert. Dies bedeutet im Umkehrschluss, dass ähnliche Applikationen und Konkurrenten bei der Entwicklung dieser App keine Rolle spielen. Durch die Analyse bereits bestehender Apps, wurden die gelungenen Eigenschaften und Macken dieser Apps herausgefiltert und dadurch weiß man nun, worauf man bei der Entwicklung dieser App achten soll.

3.2 Kundenbefragung

Eine Kundenbefragung bei der Softwareentwicklung ist wichtig, um die Bedürfnisse und Erwartungen der Zielgruppe zu verstehen. Durch direktes Feedback von potenziellen Nutzern können Entwickler ihre Produkte besser anpassen und optimieren. Im Gespräch mit dem Kundenstellverträter der Firma Pep-Digital ergaben sich beispielsweise Informationen über den Prozess der Erstellung der Events und Wünsche.

3.3 Feature-Map

Aus der Kundenbefragung ergab sich diese Feature-Map:

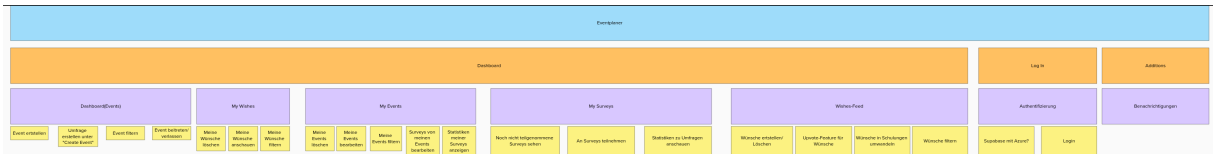

Abbildung 1: Feature-Map

3.4 User Stories

3.4.1 Anmeldung

- **Als Benutzer**, möchte ich mich mit einem bestehenden Account (z.B. Microsoft) anmelden können, **damit ich** mich nicht neu registrieren muss und die Applikation direkt mit meinem persönlichen Profil nutzen kann.
- **Als Benutzer** möchte ich mich mit meiner eigenen E-Mail-Adresse und einem selbstgewählten Passwort registrieren können, **damit ich** ein individuelles Benutzerkonto erstellen und die Applikation nutzen kann.
- **Als Benutzer** möchte ich mich mit meiner bei der Registrierung erstellten E-Mail-Adresse und meinem Passwort anmelden können, **damit ich** auf mein persönliches Benutzerkonto in der Applikation zugreifen kann.

3.4.2 Events

- **Als Benutzer** möchte ich Events erstellen können, **damit ich** eigene Veranstaltungen organisieren und veröffentlichen kann, die für alle Nutzer sichtbar sind.
- **Als Benutzer** möchte ich eine Übersichtsseite mit allen geplanten Events sehen können, **damit ich** schnell einen Überblick über bevorstehende Veranstaltungen erhalte und mich bei Interesse direkt dafür anmelden kann.
- **Als Benutzer** möchte ich einem Event beitreten können, **damit ich** meine Teilnahme an Veranstaltungen bestätigen und sicherstellen kann, dass ich für das Event registriert bin.
- **Als Benutzer** möchte ich ein Event, dem ich bereits beigetreten bin, wieder verlassen können, **damit ich** meine Teilnahme zurückziehen kann, falls ich doch nicht an der Veranstaltung teilnehmen kann.
- **Als Veranstalter** möchte ich Events, die ich erstellt habe, bearbeiten können, **damit ich** Informationen wie Titel, Datum oder Beschreibung aktualisieren und Änderungen an der Veranstaltung kommunizieren kann.
- **Als Veranstalter** möchte ich Events, die ich selbst erstellt habe, löschen können, **damit ich** fehlerhafte oder nicht mehr relevante Veranstaltungen aus dem System entfernen kann.

3.4.3 Wünsche

- **Als Benutzer** möchte ich einen Wunsch für eine Veranstaltung zu einem bestimmten Thema erstellen können, **damit ich** Vorschläge für neue Events einbringen kann.
- **Als Benutzer** möchte ich eigene erstellte Wünsche wieder löschen können, **damit ich** nicht mehr relevante oder falsch erstellte Wünsche entfernen kann
- **Als Benutzer** möchte ich Wünsche anderer Nutzer upvoten können, **damit ich** Interesse an vorgeschlagenen Veranstaltungen zeigen und unterstützen kann, dass diese umgesetzt werden.

- **Als Veranstalter** möchte ich mich zu einem Wunsch bereitstellen und daraus ein neues Event erstellen können, **damit ich** gezielt Veranstaltungen nach den Interessen der Nutzer anbieten kann.

3.4.4 Umfragen

- **Als Veranstalter** möchte ich innerhalb eines Events eine Umfrage für die Teilnehmer erstellen können, **damit ich** Feedback oder Meinungen zu bestimmten Themen einholen kann

- **Als Veranstalter** möchte ich eine von mir erstellte Umfrage bearbeiten können, **damit ich** Fragen oder Antwortoptionen anpassen kann, bevor die Umfrage abgeschlossen ist.

- **Als Veranstalter** möchte ich bei jedem Event die statistischen Auswertungen der Umfragen einsehen können, **damit ich** schnell einen Überblick darüber bekomme, zu welchen Optionen oder Themen die meisten Teilnehmer tendieren.

- **Als Benutzer** möchte ich einen Bereich haben, in dem ich alle meine noch nicht abgeschlossenen Umfragen sehen kann, **damit ich** den Überblick über offene Umfragen behalte und diese bei Bedarf weiterbearbeiten oder abschließen kann.

4 Funktionsumfang

Der Funktionsumfang vom Event-Planer wurde sorgfältig entwickelt, um eine umfassende und benutzerfreundliche Erfahrung für Mitarbeitende der Firma Pep-Digital zu gewährleisten. Im Folgenden sind die Hauptfunktionen im Detail aufgeführt:

4.1 Anmeldung und Registrierung

Mitarbeitende können sich mühelos mit ihrem bestehenden Firmenaccount über Microsoft anmelden. Die benutzerfreundliche und intuitive Oberfläche optimiert den Anmeldeprozess. Zusätzlich haben auch externe Nutzer die Möglichkeit, sich unabhängig von einem Unternehmen zu registrieren.

4.2 Events

User haben die Möglichkeit, eigene Events zu erstellen, zu bearbeiten und zu löschen. Jedes Event umfasst einen Titel und eine Beschreibung, in der Details zur Veranstaltung festgehalten werden können.

Nach der Erstellung können Events jederzeit vom Ersteller angepasst werden, sei es zur Änderung des Titels, der Beschreibung oder weiterer relevanter Informationen. Falls ein Event nicht mehr benötigt wird, kann es auch gelöscht werden.

Für eine bessere Übersicht gibt es einen **Event-Feed**, in dem alle Events angezeigt werden. Hier können User schnell durch die verschiedenen Veranstaltungen stöbern und sich inspirieren lassen.

Zudem gibt es eine eigene Sektion **„Meine Events“**, in der User ihre eigenen erstellten Events verwalten können. Hier können sie ihre Events ansehen, bearbeiten oder löschen.

Nach dem Event können individuelle Umfragen erstellt und jederzeit vom Ersteller bearbeitet werden. Diese Umfragen dienen dazu, Feedback und Meinungen der Teilnehmer zu sammeln. Mögliche Umfragethemen sind:

- **Terminfindung**: z.B. An welchen Freitagen im Monat können die meisten Teilnehmer teilnehmen?
- **Formatwahl**: Soll das Event digital oder in Präsenz stattfinden?
- **Sonstige Präferenzen**: Weitere individuelle Fragen, die der Ersteller festlegt.

Nach Abschluss des Events können die Teilnehmer das Event mit einer einfachen Sternebewertung (1-5 Sterne) bewerten. Diese Funktion hilft den Veranstaltern, schnell zu sehen, wie das Event bei den Teilnehmern ankam.

4.3 Wünsche

User haben die Möglichkeit, Wünsche zu erstellen, in denen sie äußern können, welche Events sie sich wünschen oder vorschlagen möchten. Jeder Wunsch enthält eine kurze Beschreibung des gewünschten Events.

Für eine bessere Übersicht gibt es einen Wünsche-Feed, in dem alle Wünsche angezeigt werden. Hier können User durch die verschiedenen Vorschläge stöbern und ihre Interessen zeigen.

Zudem gibt es eine eigene Sektion Meine Wünsche, in der User ihre eigenen erstellten Wünsche verwalten können. Hier können sie ihre Wünsche ansehen oder löschen.

Andere User können Wünsche upvoten, um zu zeigen, dass sie Interesse an diesem Event haben. Es gibt keine Mindestanzahl an Upvotes unabhängig von der Anzahl kann sich jemand bereitstellen, den Wunsch in ein tatsächliches Event umzusetzen.

Der Ersteller kann seinen Wunsch jederzeit bearbeiten oder löschen, falls sich die Idee ändert oder nicht mehr relevant ist.

Dieses Feature hilft dabei, Events zu organisieren, die wirklich von der Community gewünscht werden, und fördert die aktive Teilnahme aller User.

5 UI Entwürfe

Anmerkung: Die UI-Entwürfe sind keine endgültigen Designentscheidungen, sondern dienen lediglich zur Veranschaulichung der Anwendungsstruktur.

5.1 Login

Abbildung 2: Login

5.2 Events

5.2.1 Event-Feed

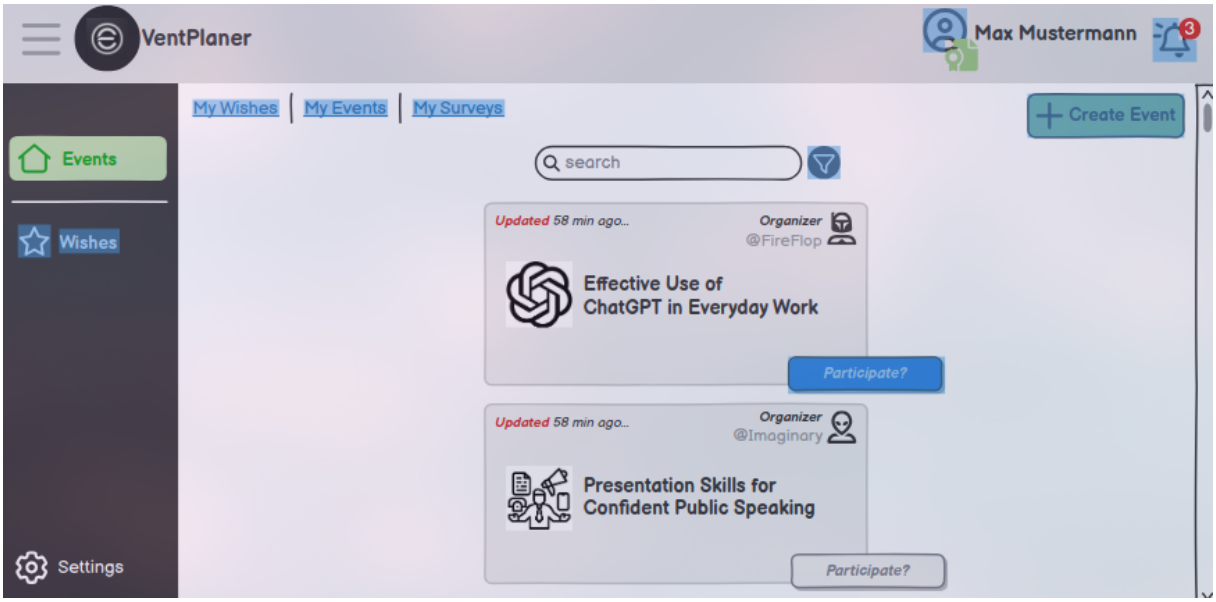

Abbildung 3: Event-Feed

5.2.2 Meine Events

Abbildung 4: Meine Events

5.2.3 Event erstellen

Abbildung 5: Event erstellen

5.2.4 Umfrage erstellen

Abbildung 6: Umfrage erstellen

5.2.5 Umfrage bearbeiten

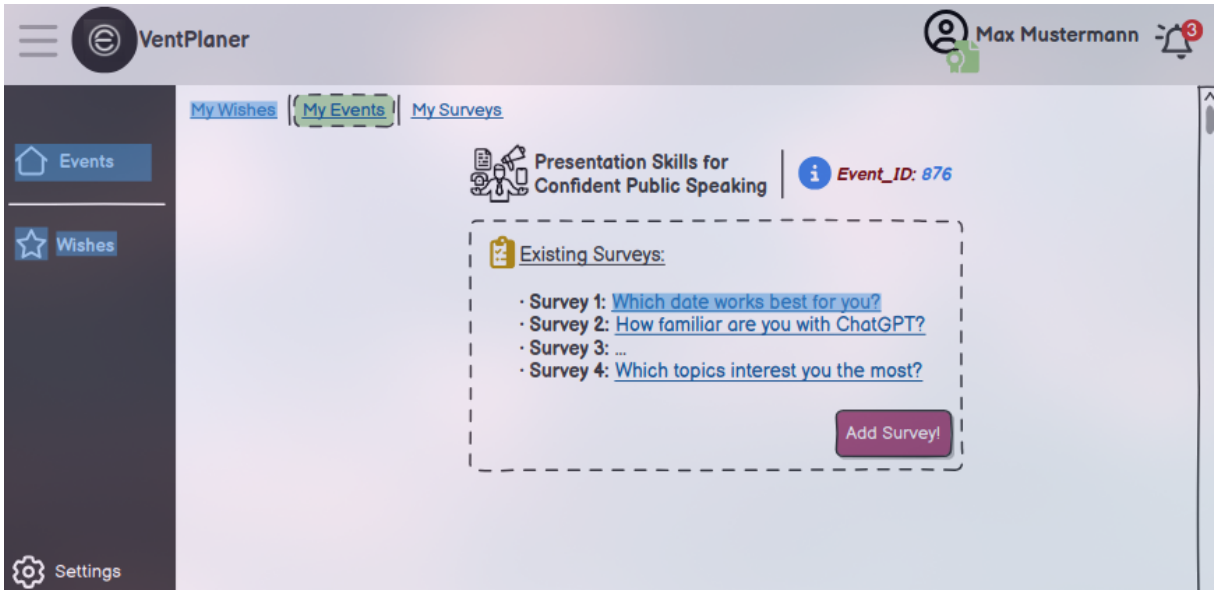

Abbildung 7: Umfrage bearbeiten

5.2.6 Meine Umfragen

Abbildung 8: Meine Umfragen

5.2.7 Event bearbeiten

Abbildung 9: Event bearbeiten

5.2.8 Event löschen

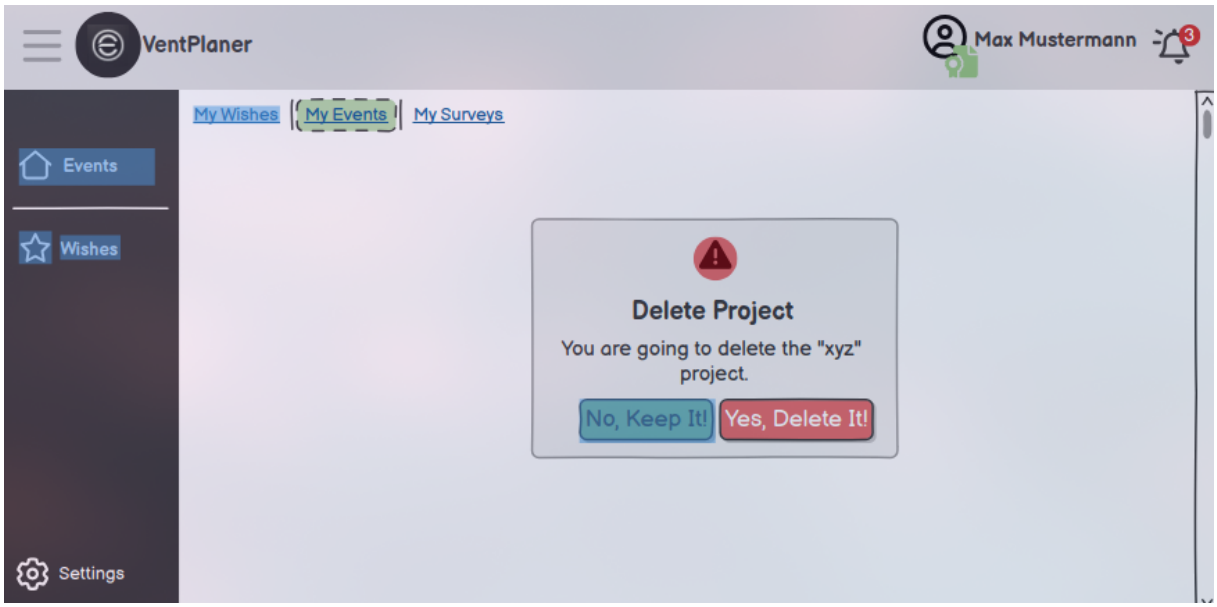

Abbildung 10: Event löschen

5.3 Wünsche

5.3.1 Wünsche-Feed

Abbildung 11: Wünsche-Feed

5.3.2 Meine Wünsche

Abbildung 12: Meine Wünsche

5.3.3 Wunsch erstellen

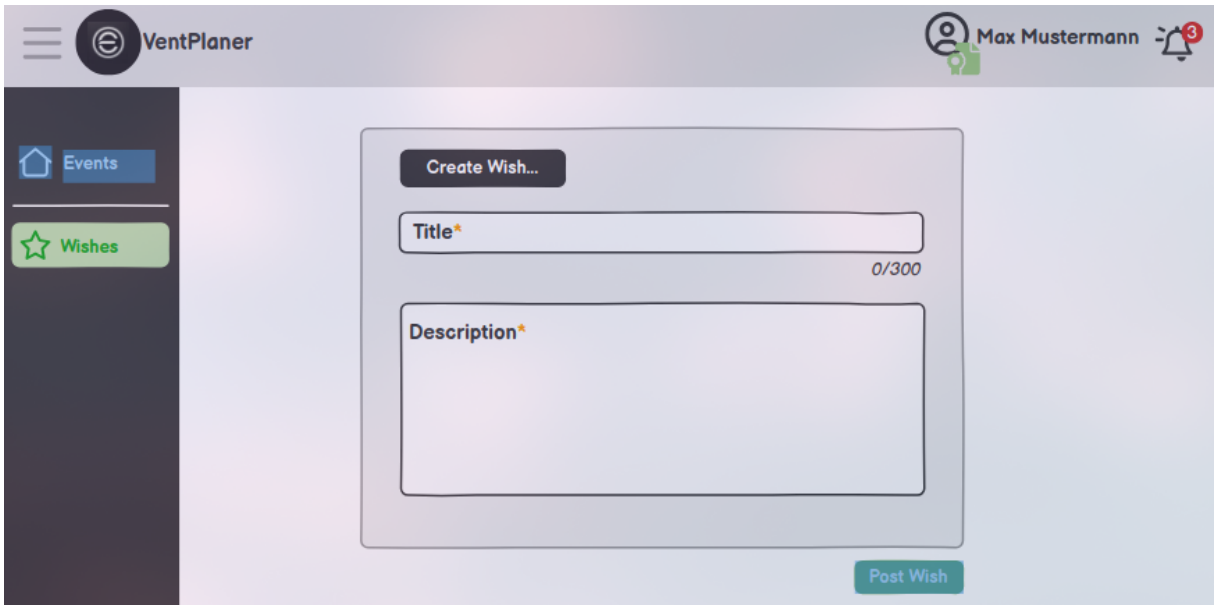

Abbildung 13: Wunsch erstellen

6 Aufwandsschätzung

6.1 Anforderungsanalyse

In dieser Phase werden die Anforderungen des Event-Planer festgelegt, einschließlich der Features und Funktionalitäten

- Dauer: 1 Woche

6.2 Entwurfsphase

Es wird das Design bzw. die Struktur der Anwendung erstellt, einschließlich der Benutzeroberfläche, der Datenbankstruktur und der Systemarchitektur.

- Dauer: 2 Woche

6.3 Implementierung & Testing

Entwicklung der Anwendung basierend auf den festgelegten Anforderungen und dem Design bzw. der Struktur. Applikation soll direkt nach jedem Feature auf Funktion getestet werden

- Dauer: 10 Wochen
  - Einarbeitung Technologien: 2 Wochen
  - Implementierung & Testing: 8 Wochen

6.4 Bereitstellung und Projektabschluss

Bereitstellung der Anwendung nach erfolgreicher Implementierung & Testphase.

- Dauer: 1-2 Wochen

7 Projektmanagement: "A la Scrum"

Für dieses Projekt werden gewisse Apekte der agilen Methode Scrum verwendet. Dabei werden die Scrum-Zyklen auch bekannt als Sprints in 1 Wochen abschnitten eingeteilt, die sich durch schrittweise Entwicklung und durch regelmäßige Feedbackschleifen auszeichnet. Alle Aufgaben eines Sprints werden in Jira in einem Backlog gespeichert, beschrieben und unter dem Team verteilt.

Das Team hat wöchentliche Meetings mit dem Kunden und danach mit dem Betreuer des Projektes auch bekannt als Spint-Review. Dieser Termin findet immer Montags statt und markiert das Ende des aktuellen Sprints. Dieser Termin wird genutzt um die Ergebnisse des Kundes vom Sprint zu präsentieren und Feedback einzuholen, danach werden die nächsten Anforderungen vom Kunden besprochen. Im Anschluss darauf findet das Treffen mit dem Betreuer statt, bei denen offene fachliche Fragen geklärt werden.

Die Sprints beginnen immer am Dienstag nachdem alle Anforderungen und Feedbacks eingesammelt worden sind. Des weiteren wurde der Sonntag als teaminternen Tag gekennzeichnet, bei dem weitere Fragen und Vorgehensweisen besprochen werden. Zudem werden die bearbeiteten Aufgaben durchgegangen und geklärt, was mit dem Kunden am Folgetag gesprochen wird. Nach dem Kundentreffen ist ein teaminternes Meeting angesetzt, bei welchem reflektiert wird, wie der letzte Sprint lief und mögliche Verbesserungen angesprochen werden.

Im Ablauf der Sprints werden 2 Daily Standup-Meetings inkludiert, da mehrere Dailys im Rahmen dieses Projekts unrealistisch sind. Allerdings sind die Teammitglieder die ganze Woche erreichbar, um Unterstützung zu leisten und spontane Treffen einzurichten. Als Kommunikationsmittel wird teamintern ein Discord-Channel verwendet und für die Betreuer- sowie Kundentreffen hingegen Microsoft-Team.

8 Systemarchitektur

8.1 Allgemein

Die Architektur des Gesamtsystems ist in einer Full-Stack Komponente unterteilt. Die Datenbank und der Authentifizierungsservice erfolgt über Supabase und einem Provider im Falle des Projektes über Microsoft, dass auch SSO unterstützt.

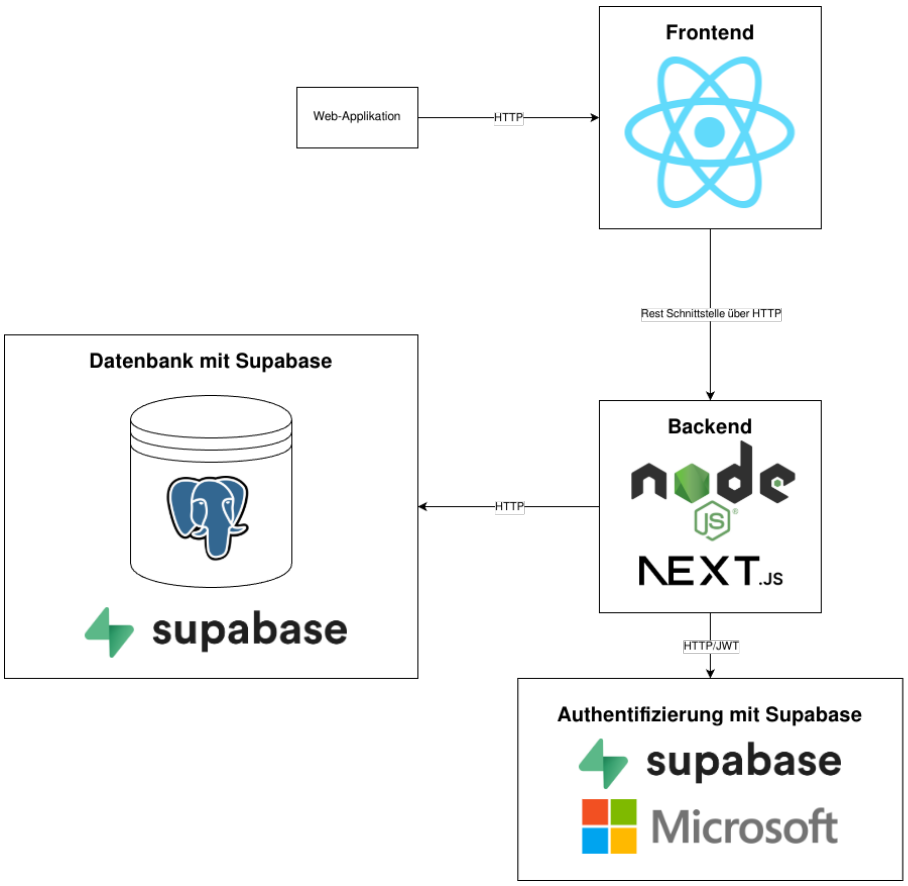

Abbildung 14: Systemarchitektur

8.2 Struktursicht

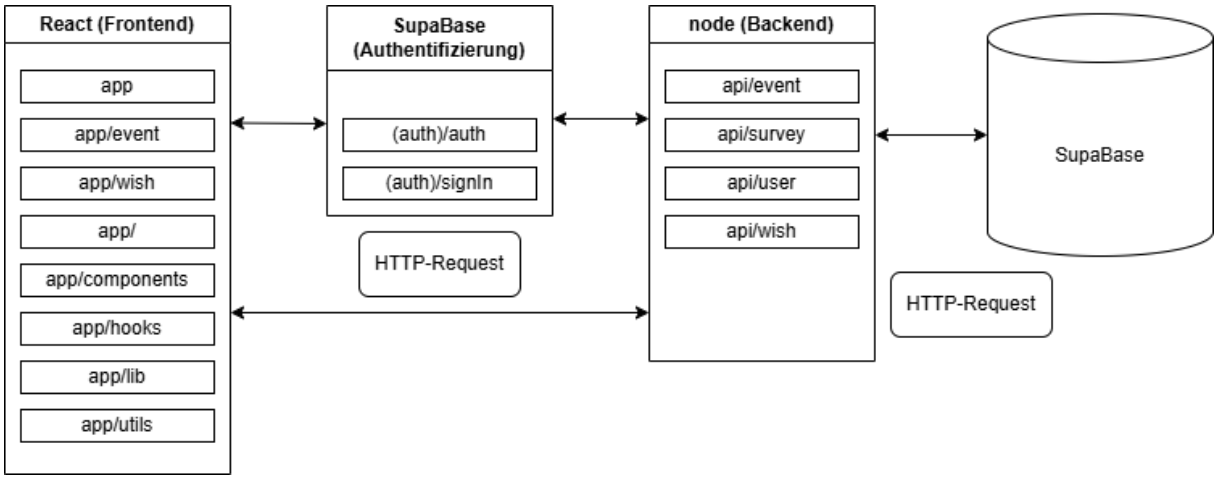

Abbildung 15: Struktursicht

Die Struktursicht zeigt den Aufbau des Projekts, das mit Next.js als Fullstack-Framework umgesetzt wurde. Die Anwendung ist in drei Hauptbereiche unterteilt: Das Frontend basiert auf React und ist modular strukturiert. Für die Authentifizierung wird SupaBase direkt vom Frontend aus genutzt. Die Backend-Logik ist in Node.js implementiert und über die integrierten API-Routen von Next.js (z.B. api/event, api/user) realisiert. Diese kommunizieren per HTTP mit der SupaBase-Datenbank. Die Architektur trennt somit klar zwischen Benutzeroberfläche, Geschäftslogik und Datenhaltung.

8.3 Verhaltenssicht

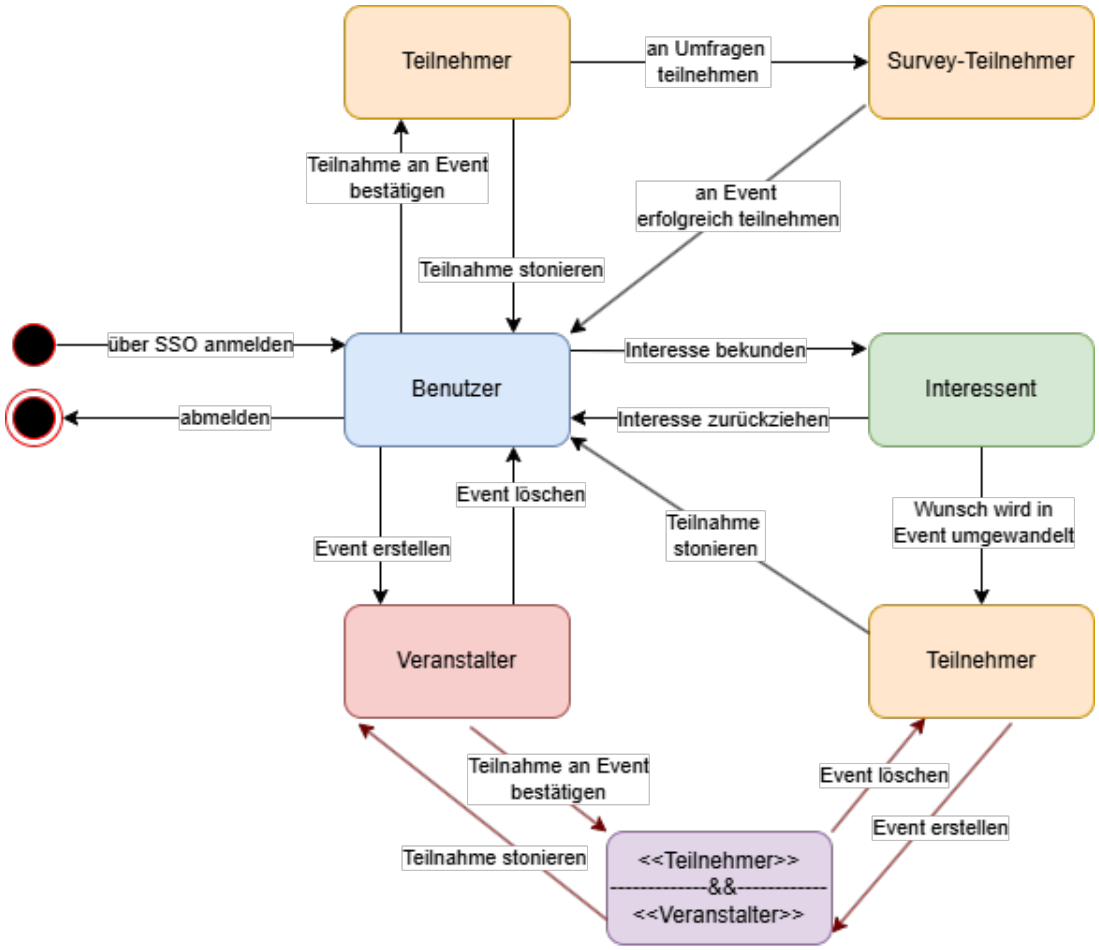

Abbildung 16: Verhaltenssicht

Die Verhaltenssicht beschreibt die Rollen und Interaktionen der Nutzer innerhalb der Anwendung. Ausgangspunkt ist der Benutzer, der sich über SSO anmelden und verschiedene Rollen einnehmen kann. Als Interessent kann er Interesse an einem Event bekunden oder zurückziehen. Sobald Interesse in eine Teilnahme übergeht, wird er zum Teilnehmer und kann Events bestätigen, stornieren oder an Umfragen teilnehmen. Wird ein Benutzer selbst aktiv, kann er als Veranstalter Events erstellen und verwalten. Die Grafik veranschaulicht diese dynamischen Zustandswechsel und Aktionen im System.

8.4 Verteilungssicht

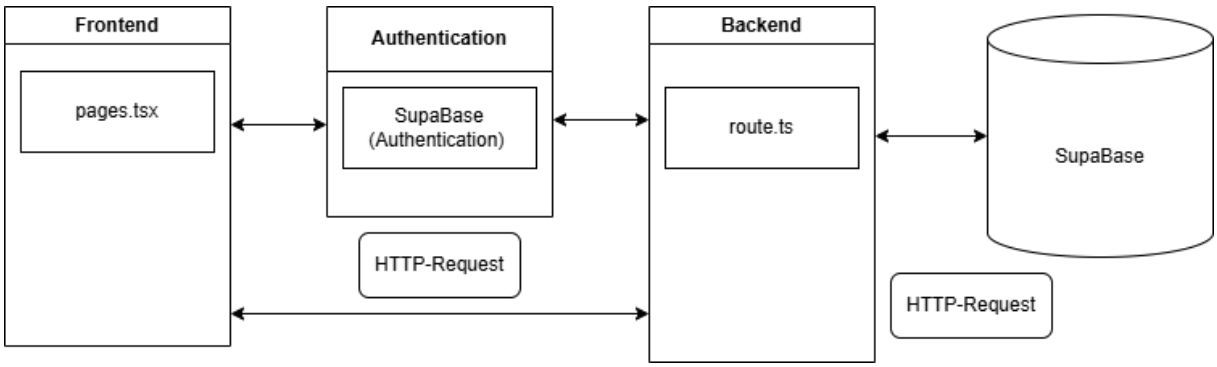

Abbildung 17: Verteilungssicht

Die Verteilungssicht zeigt, wie die verschiedenen Teile der Anwendung auf technische Komponenten verteilt sind. Das Frontend besteht aus React-Komponenten, die in pages.tsx implementiert sind. Die Authentifizierung erfolgt über SupaBase, das direkt vom Frontend per HTTP-Request angesprochen wird. Das Backend nutzt eine Datei wie route.ts, um serverseitige Logik bereitzustellen, und kommuniziert ebenfalls per HTTP mit der SupaBase-Datenbank. Die Datenhaltung sowie Authentifizierung sind somit ausgelagert und zentral über SupaBase realisiert. Die Struktur ermöglicht eine klare Trennung zwischen Präsentation, Authentifizierung, Logik und Daten.

9 Schnittstellentechnologien

9.1 Schnittstellenbeschreibung

Schnittstellentechnologien bilden das Fundament für die reibungslose Kommunikation zwischen den verschiedenen Komponenten unseres Softwaresystems. Im Rahmen dieses Projekts dient eine RESTful-API als zentrale Vermittlungsinstanz zwischen dem Next.js-basierten Frontend, der Prisma-gestützten Datenbankebene und der Authentifizierungslösung von Supabase. Als Datenformat kommt JSON zum Einsatz, während die Übertragung über das HTTPS-Protokoll erfolgt. Diese Kombination gewährleistet nicht nur eine hohe Effizienz, sondern auch eine plattformunabhängige und sichere Kommunikation zwischen den Systemkomponenten.

Die Architektur der API orientiert sich am App-Router-Konzept von Next.js, wobei jede Route in einer eigenständigen Datei (z. B. app/api/.../route.ts) implementiert wird. Die Gestaltung der Schnittstellen folgt dabei den etablierten REST-Prinzipien, insbesondere den Aspekten der Zustandslosigkeit, Adressierbarkeit, Verwendung standardisierter HTTP-Methoden und einer konsistenten Nutzung von HTTP-Statuscodes.

Ein zentrales Merkmal der gewählten Architektur ist die Zustandslosigkeit (Statelessness), was bedeutet, dass jede eingehende Anfrage sämtliche für die Verarbeitung notwendigen Informationen - einschließlich der Authentifizierungsdaten - selbst enthalten muss. Die Verwaltung der Benutzersitzungen erfolgt dabei durch Supabase, das JSON-Web-Tokens (JWT) in Form von Cookies bereitstellt. Zusätzliche clientseitige Zustände, wie beispielsweise der lokale Status von Upvotes, werden durch den Client selbst verwaltet, wodurch eine klare Trennung der Verantwortlichkeiten gewahrt bleibt.

Das Prinzip der Adressierbarkeit manifestiert sich in der strukturierten und eindeutigen Identifizierbarkeit aller Ressourcen über entsprechende URIs. So ermöglicht beispielsweise der Endpunkt GET /api/wish den Abruf einer vollständigen Liste aller Wünsche, während eine neue Ressource durch eine Anfrage an POST /api/wish angelegt werden kann. Spezifische Aktionen, wie das Setzen eines Upvotes, werden über definierte Routen wie POST /api/wish/wishId/upvote ausgelöst.

Die verwendeten HTTP-Methoden folgen dem etablierten CRUD-Paradigma (Create, Read, Update, Delete), wobei die Leseoperationen durch die GET-Methode realisiert werden. Das Anlegen neuer Ressourcen oder das Auslösen von Aktionen erfolgt mittels POST, während PUT und PATCH für Aktualisierungen und DELETE für das Entfernen von Ressourcen vorgesehen sind.

Die API kommuniziert den Erfolg oder Misserfolg einer Operation durch entsprechende HTTP-Statuscodes. Erfolgreiche Anfragen werden mit den Codes 200 OK oder 201 Created quittiert, während Validierungsfehler eine Antwort mit dem Status 400 Bad Request und detaillierten Fehlerbeschreibungen zur Folge haben. Unberechtigte Zugriffsversuche führen zur Rückgabe des Status 401 Unauthorized, und unerwartete Serverfehler werden durch den Code 500 Internal Server Error signalisiert.

10 Datenbank

10.1 Logisches Datenmodell

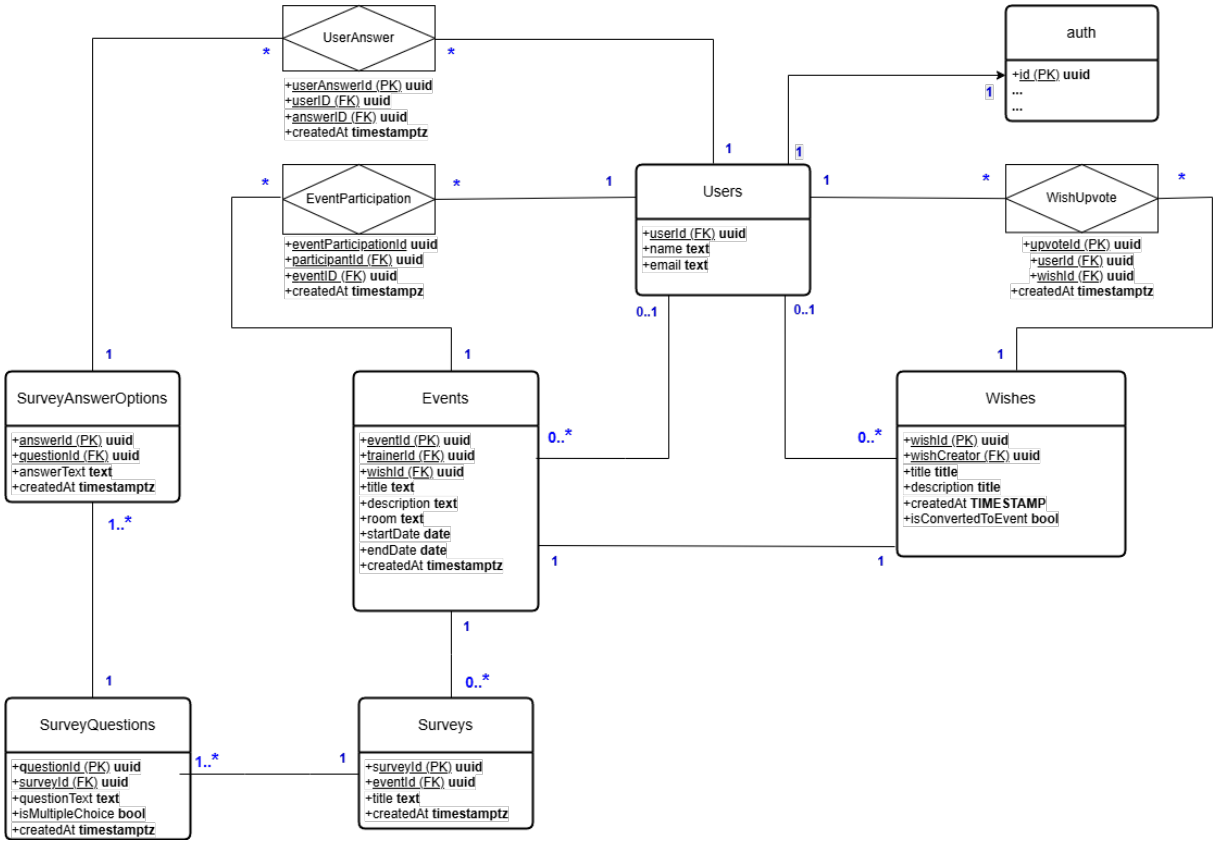

Abbildung 18: Logisches Datenmodell

10.1.1 Entity Relationship Modell

In der Anwendung übernimmt die Datenbank eine tragende Rolle - sie bildet das Fundament, auf dem sämtliche Datenflüsse und Systemlogiken aufbauen.Mithilfe ihrer Struktur lassen sich unterschiedliche Entitäten und deren Beziehungen modellieren, wodurch Funktionen wie Eventmanagement, Umfragen oder das Einreichen von Nutzerwünschen umgesetzt werden können. Technisch basiert das System auf Supabase, das neben der Bereitstellung der relationalen Datenbank auch mit Supabase Auth eine sichere und zugleich benutzerfreundliche Authentifizierungslösung integriert.

10.1.2 Nutzerverwaltung mit Supabase Auth

Ein zentraler Bestandteil dieser Architektur ist die enge Verknüpfung zwischen der eigens entwickelten Users-Tabelle und der von Supabase bereitgestellten Tabelle auth.users. Während sicherheitskritische Informationen wie Passwörter ausschließlich in auth.users abgelegt und geschützt bleiben, enthält die eigene Users-Tabelle ergänzende Angaben wie E-Mail-Adresse oder Anzeigename. Die Verbindung beider Tabellen erfolgt über das id-Feld, das in auth.users als Primary Key definiert ist. Ein automatisch ausgelöster Trigger sorgt dafür, dass diese ID unmittelbar nach der Registrierung auch in der Users-Tabelle

als Foreign Key gespeichert wird. Dadurch wird jeder Nutzer eindeutig identifizierbar und eine klare Trennung zwischen sicherheitsrelevanten und anwendungsbezogenen Informationen geschaffen.

10.1.3 Haupttabellen und ihre Aufgabenbereiche

Die Datenbankstruktur gliedert sich in Haupt- und Hilfstabellen. Zu den Haupttabellen zählen neben Users auch Wishes, Events sowie Surveys und SurveyQuestions. Über Wishes lassen sich Vorschläge und Ideen - etwa für künftige Schulungsveranstaltungen - einreichen. Diese Vorschläge können von anderen Nutzern geupvoted oder in tatsächliche Events umgewandelt werden. Solche Events können jedoch auch unabhängig von Wünschen erstellt werden. Surveys ermöglichen es, Rückmeldungen einzuholen, beispielsweise zur Qualität von Events, und sind direkt mit den dazugehörigen SurveyQuestions verknüpft, die die konkreten Fragestellungen enthalten.

10.1.4 Hilfstabellen für komplexe Beziehungen

Die Hilfstabellen dienen vor allem der Abbildung komplexer, meist Viele-zu-Viele-Beziehungen. So wird in UserAnswer gespeichert, welche Antwort ein bestimmter Nutzer auf eine Umfragefrage gegeben hat. Die Tabelle EventParticipation dokumentiert, welche Nutzer sich für welche Events angemeldet haben. WishUpvote wiederum ermöglicht es, Schulungswünsche zu unterstützen, indem man signalisiert, dass man die Idee für sinnvoll hält und sich eine Umsetzung in Form eines Events wünscht.

10.2 Physisches Datenmodell

Für das physische Datenmodell wird Supabase verwendet, das auf einer PostgreSQL-Datenbank basiert. Supabase hostet die gesamte Datenbank sowie die Benutzer-Authentifizierung. Alle Tabellen und Beziehungen folgen dem zuvor definierten logischen Modell. Zusätzlich sorgen serverseitige Funktionen wie Trigger und Zugriffspolicies für Sicherheit und Datenkonsistenz.

11 Literaturverzeichnis