

Multiuser Applikation

Damien Flury

04. November 2019

Inhaltsverzeichnis

1	Anwendungsfälle	2
1.1	Akteure	2
1.2	Anforderungen	2
1.2.1	Create Account	2
1.2.2	Login	3
1.2.3	Logout	3
1.2.4	Manage Entries	3
1.3	Nicht funktionale Anforderungen	3
1.3.1	Performance	3
1.3.2	Design	3
1.3.3	Simple Authentifizierung	3
1.3.4	Sicherheit	3
2	Datenhaltung	3
3	Architektur	5
3.1	Packagediagramm	5
3.2	Klassendiagramm	5
3.3	Deploymentdiagramm	5
4	Testfälle	5
4.1	Funktionale Testfälle	5
4.1.1	Account erstellen	5
4.1.2	Anmeldung	7
4.1.3	Abmeldung	7
4.1.4	Entry erstellen	7
4.2	Nichtfunktionale Anforderungen	7



Abbildung 1: Use Case-Diagramm (Entries und Autorisierung)

1 Anwendungsfälle

Ein Anwendungsfalldiagramm finden Sie in Abbildung 1.

1.1 Akteure

Benutzer können sich anmelden, abmelden und Entries verwalten.

1.2 Anforderungen

1.2.1 Create Account

Neue Benutzer können einen eigenen Account erstellen. Dazu brauchen sie eine Email-Adresse und ein Passwort, welches den Anforderungen entsprechen.

1.2.2 Login

Bestehende Benutzer können sich anmelden. Dazu werden wiederum die Email-Adresse, welche sie zur Erstellung verwendet wurde, und das dazugehörige Passwort benötigt. Ein Aktivitätsdiagramm dazu finden Sie in Abbildung 2.

1.2.3 Logout

Eingeloggte Benutzer können sich wieder abmelden. Dazu müssen sie zunächst angemeldet sein.

1.2.4 Manage Entries

Eingeloggte Benutzer können neue Entries erstellen, lesen, bearbeiten und wieder löschen.

1.3 Nicht funktionale Anforderungen

1.3.1 Performance

Die Datenbankabfragen müssen möglichst performant ablaufen, um die Benutzerfreundlichkeit nicht einzuschränken.

1.3.2 Design

Einheitliches Design in der Webapplikation, um die Benutzerfreundlichkeit zu optimieren.

1.3.3 Simple Authentifizierung

Die Applikation benötigt Authentifizierung. Die Tokens werden im Frontend gespeichert, sodass der Benutzer sich nicht jedesmal erneut einloggen muss. Für das Login werden lediglich Email und Passwort benötigt.

1.3.4 Sicherheit

Um die bestmögliche Sicherheit zu garantieren, wird HTTPS verwendet und ein ORM verwendet, um Database Injection zu vermeiden.

2 Datenhaltung

Die Applikation besteht aus vier Datenklassen (siehe Abbildung 3):

- ApplicationUser
- Entry
- Position
- Department

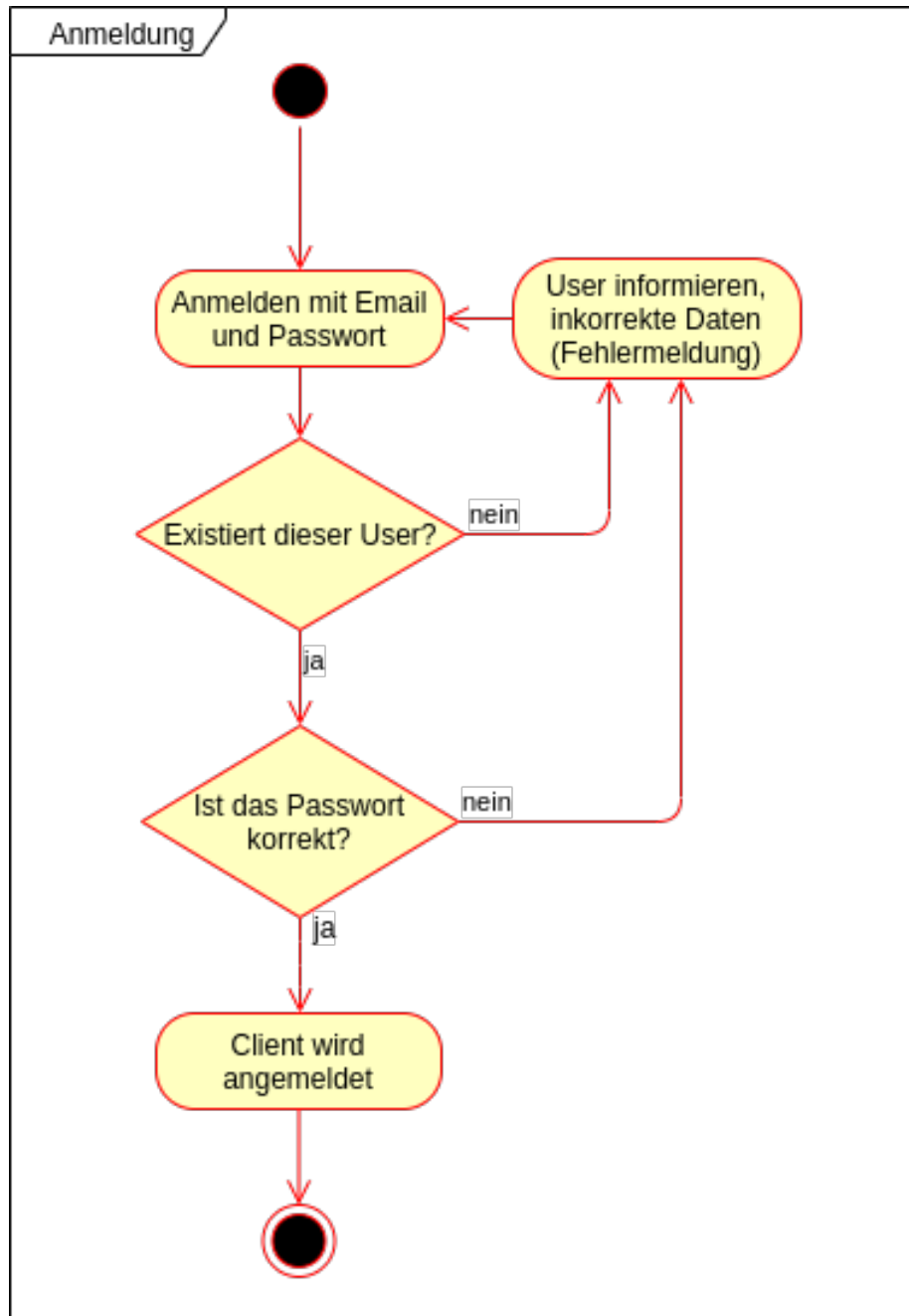


Abbildung 2: Aktivitätsdiagramm (Anmeldung)

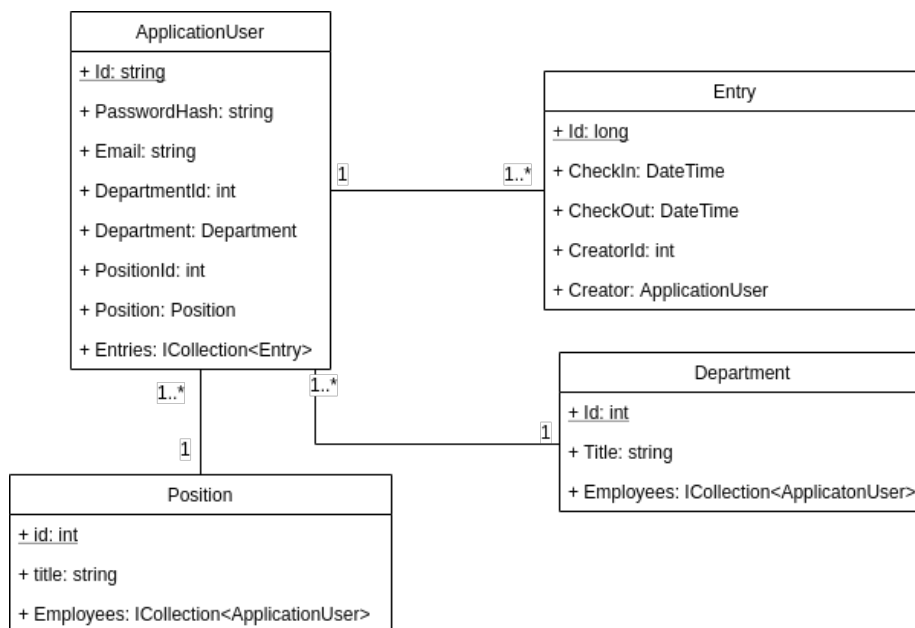


Abbildung 3: Fachklassendiagramm

3 Architektur

3.1 Packagediagramm

Ich verwende sechs Namespaces, zwei für die Datenbank und vier für die GraphQL API (siehe Abbildung 4).

3.2 Klassendiagramm

Ich verwende Klassen für die Datenbank-Entitäten und für die GraphQL Types. Ausserdem gibt es eine Startup-Klasse für die Projektkonfiguration (siehe Abbildung 5).

3.3 Deploymentdiagramm

Da wir lediglich auf localhost deployen, entstehen zwei Ports und eine Datenbank, welche als einfaches File eingesetzt wird. Auf Port 5001 wird eine GraphQL Api mit .NET Core eingesetzt, auf Port 3000 entsteht eine React Applikation (siehe Abbildung 6).

4 Testfälle

4.1 Funktionale Testfälle

4.1.1 Account erstellen

Siehe Tabelle 1

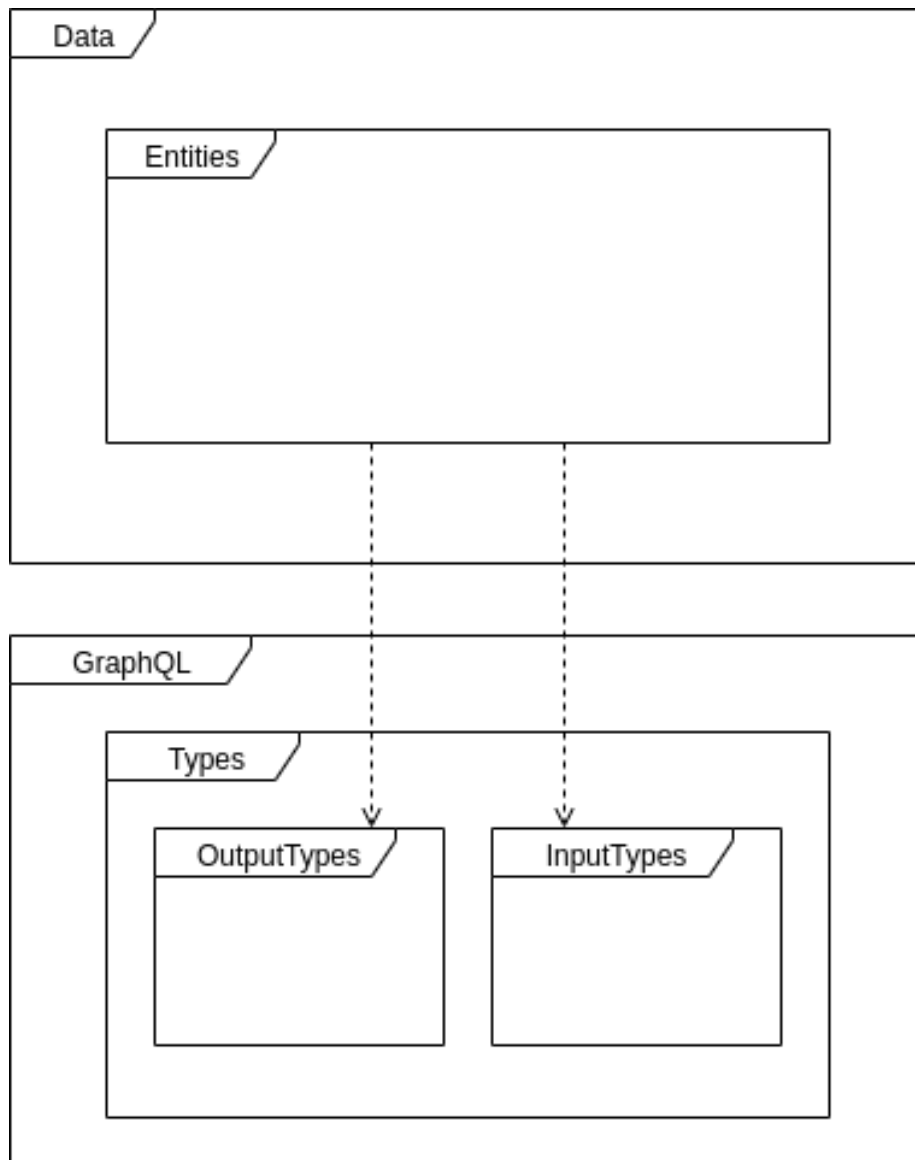


Abbildung 4: Packagediagramm

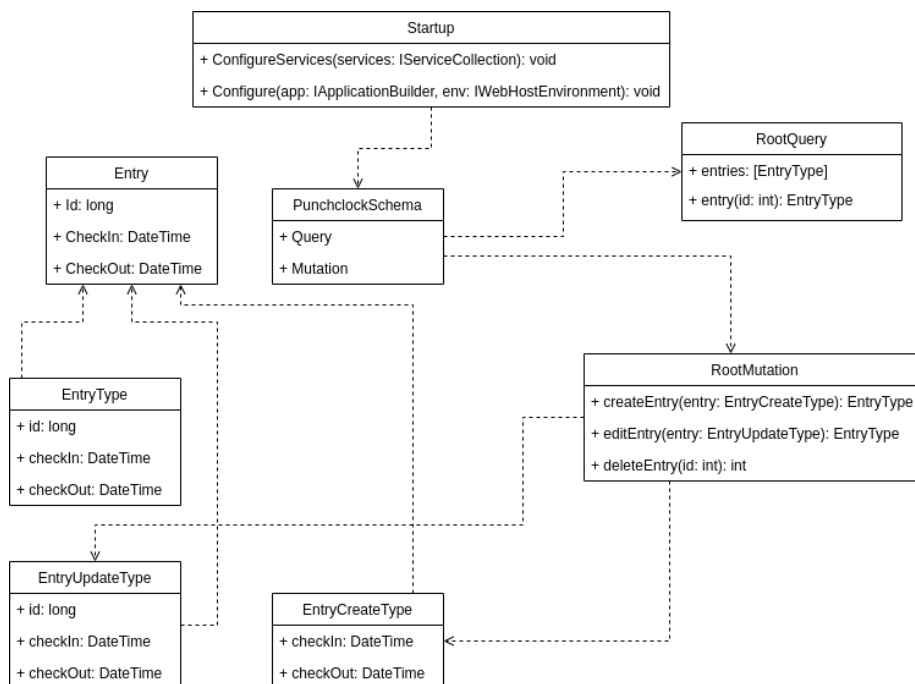


Abbildung 5: Klassendiagramm

Beschreibung	Der Benutzer kann einen neuen Account erstellen
Voraussetzung	Die Website muss funktionieren
Testschritte	<ol style="list-style-type: none"> 1. SSIGN Up"klicken 2. Email: "test1@test1.com" 3. Passwort: "P@ssw0rd!" 4. Submit klicken
Erwartetes Ergebnis	Der Account wird erfolgreich erstellt, der Benutzer wird erfolgreich angemeldet

Tabelle 1: Account erstellen

4.1.2 Anmeldung

Siehe Tabelle 2

4.1.3 Abmeldung

Siehe Tabelle 3

4.1.4 Entry erstellen

Siehe Tabelle 4

4.2 Nichtfunktionale Anforderungen

Eine Übersicht zu den nichtfunktionalen Anforderungen finden Sie in Tabelle 5.

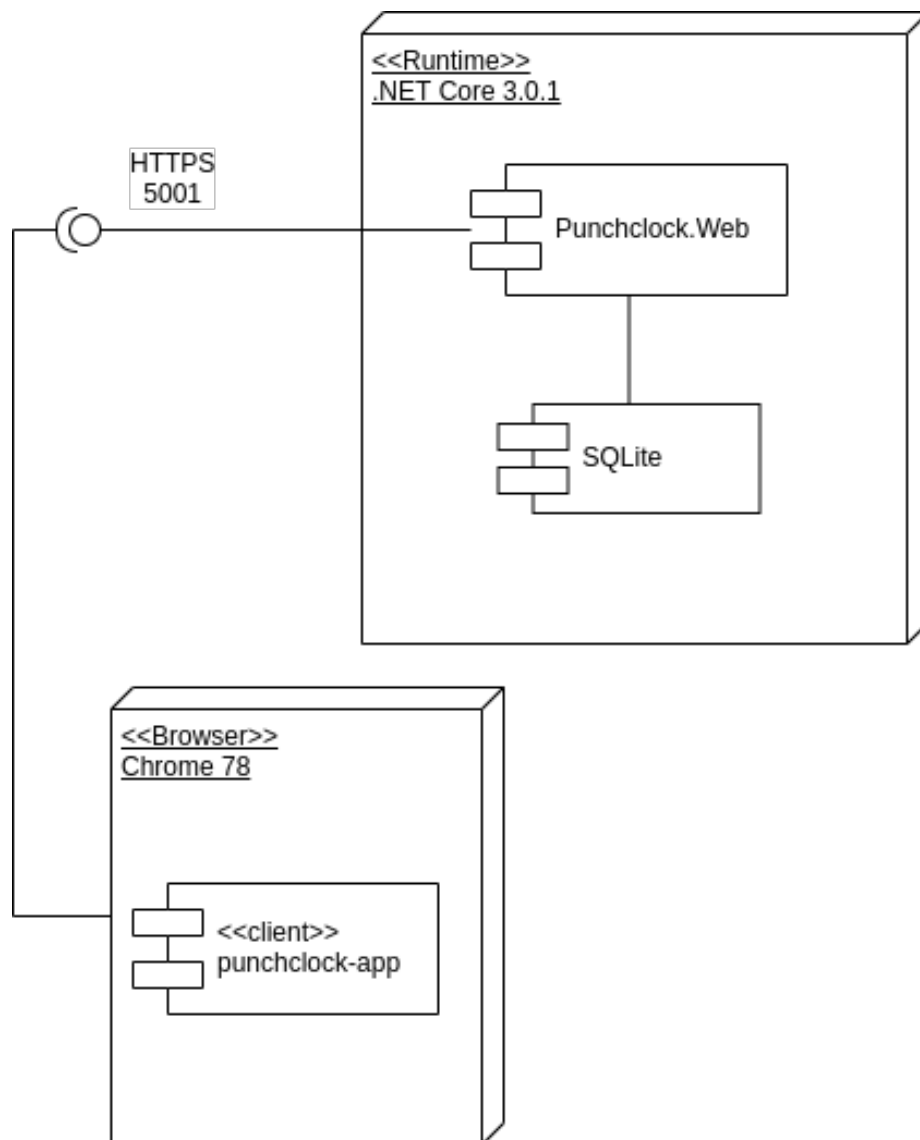


Abbildung 6: Deploymentdiagramm

Beschreibung	Der Benutzer kann sich anmelden
Voraussetzung	Die verwendete Email muss registriert sein
Testschritte	1. Loginseite aufrufen 2. Email: "test@test.com" 3. Passwort: "P@ssw0rd!" 4. Submit klicken
Erwartetes Ergebnis	Der Benutzer wird erfolgreich angemeldet und auf Index umgeleitet

Tabelle 2: Anmeldung

Beschreibung	Der Benutzer kann sich abmelden
Voraussetzung	Der Benutzer muss angemeldet sein
Testschritte	1. "Abmelden" klicken
Erwartetes Ergebnis	Der Benutzer wird erfolgreich abgemeldet und auf Index umgeleitet

Tabelle 3: Abmeldung

Beschreibung	Der Benutzer kann neue Entry erstellen
Voraussetzung	Der Benutzer muss angemeldet sein
Testschritte	1. "Check In" klicken
Erwartetes Ergebnis	Der Benutzer wird erfolgreich für den Tag eingchecked. Visuelles Feedback erfolgt.

Tabelle 4: Entry erstellen

Testfall	Result
Einheitliches Design	Der Benutzer findet sich zurecht.
Einfache Authentifizierung	Der Benutzer braucht nur Email und Passwort, um sich anzumelden.
Performance	Die Entries werden schnell zurückgegeben, Wartezeiten müssen <1 Sekunde betragen.

Tabelle 5: Nichtfunktionale Testfälle