**1. Ziele der Mini-App**

Das Ziel der Mini-App „Geräteausleihe“ ist es, eine effiziente und benutzerfreundliche Lösung für das Verwalten und Ausleihen von IT-Geräten wie Laptops, Smartphones, Tablets, und mehr zu bieten. Benutzer sollen die Verfügbarkeit von Geräten prüfen, Reservierungen tätigen und die Ausleihe- und Rückgabeprozesse einfach abwickeln können. Ein Kalender für bessere Übersicht der Buchung soll auch verfügbar sein.

**2. Funktionale Anforderungen**

1. Admin kann die Geräte verwalten, löschen und hinzufügen
2. Benutzer können alle verfügbaren Geräte anzeigen lassen und nach danach suchen/ Benutzer sollten eine grafische Benutzeroberfäsche bedienen
3. Benutzer können Geräte ausleihen und zurückgeben. Die App aktualisiert automatisch den Status der Geräte.
4. Benutzer sollten die verfügbaren Geräte sehen können
5. Die Dateb sollten in einer Datenbank gespeichert werden
6. Benutzer können Geräte für bestimmte Zeiträume reservieren und geplante Ausleihen anzeigen/Benutzer sollten die Daten in einem Kalender visualisieren.

**3. Nicht-funktionale Anforderungen**

1. Die Suchergebnisse sollten sortiert werden können
2. Die Architektur der App soll so aufgebaut sein, dass sie bei einer steigenden Anzahl von Benutzern und Geräten einfach skaliert werden kann.
3. Die Ausleihdauer sollte begrenzt werden
4. Benutzer sollten die Anwendung intuitiv und ohne lange Einarbeitung nutzen können. Die Benutzeroberfläche sollte einfach zu bedienen und gut strukturiert sein.

**4. Technische Spezifikationen**

* **Frontend**: Die Benutzeroberfläche wird mit **Java und Vaadin** erstellt, das auf Java basiert und eine benutzerfreundliche GUI-Entwicklung für Webanwendungen ermöglicht.
* **Backend**: Das Backend basiert auf **Java** und verwendet das **Vaadin-Framework**. Java bietet Zuverlässigkeit, hohe Sicherheit und ist für Unternehmensanwendungen gut geeignet.
* **Datenbank**: **PostgreSQL** wird als Datenbank verwendet. PostgreSQL bietet gegenüber MySQL erweiterte Funktionen für komplexe Datenverarbeitung, ist sehr stabil und auf hohe Skalierbarkeit ausgelegt.

**Vorteile der gewählten Technologien**:

* **Java**: Java wurde aufgrund seiner Stabilität, hohen Performance und bewährten Sicherheitsstandards für unsere Enterprise-Anwendung gewählt. Die breite Community-Unterstützung und regelmäßigen Updates machen Java langfristig wartbar und zukunftssicher. Die strikte Typisierung und die strukturierte Codebasis tragen zur Fehlervermeidung bei und erleichtern die Wartung. Java ermöglicht zudem eine einfache Skalierung für wachsende Anwendungen und bietet als kompilierte Sprache schnelle Antwortzeiten. Außerdem bringt Zichao viel Erfahrung mit Java mit, was die Entscheidung gegenüber PHP zusätzlich begründet.
* **Vaadin**: Unterstützt die Entwicklung von Webanwendungen auf Basis von Java und bietet eine Vielzahl vorgefertigter UI-Komponenten, was die Entwicklungszeit verkürzt.
* **PostgreSQL**: Unterstützt komplexe Abfragen, hohe Skalierbarkeit und erweiterte Sicherheitsfunktionen.
* **GitHub**: Ermöglicht Versionskontrolle, einfache Zusammenarbeit und Sicherung des Quellcodes.

**5. Architektur der App**

Die Anwendung folgt einer **dreischichtigen Architektur**:

1. **Frontend (UI-Schicht)**: Vaadin-basierte Benutzeroberfläche, die auf die REST-API zugreift, um Daten anzuzeigen und Benutzereingaben zu verarbeiten.
2. **Backend (Logikschicht)**: Java mit Vaadin, das die Business-Logik verarbeitet und Sicherheitsmechanismen bereitstellt.
3. **Datenbank (Datenhaltungsschicht)**: PostgreSQL speichert alle Daten zu Benutzern, Geräten und Buchungen.

**6. Datenbankmodell**

Geräte-class:  
- id: int // eindeutige ID  
- type : string // Typ des Geräts (Phone/Camera/Keyboard...)  
- model: string // Modell des Geräts (iPhone 13 mini 128gb red/Canon EOS 2000D Kit 18-55mm IS II/...)  
- status: enum // Status des Geräts (available/rented/missing)  
- max\_days: int // Maximale Ausleihdauer  
  
Buchung-class:  
- id: int // eindeutige ID  
- device\_id: int // ID des Geräts  
- user\_name: string // Name des Benutzers  
- borrow\_date: date // Ausleihdatum  
- return\_date: date // Rückgabedatum  
  
CRUD:  
 create:  
 Geräte (admin)  
 Buchung  
  
 read:  
 alle Geräte  
 alle Buchungen  
 Buchungen nach Geräte  
 Suchung nach id/Geräte/Typ/  
 update:  
 Geräte (admin)  
 Buchung (only return\_date)  
  
 delete:  
 Geräte (admin)  
 Buchung (admin)

1. **Zukunftserweiterung (optional)**

* **Benachrichtigungssystem**
  + E-Mail-/Push-Benachrichtigungen: Wenn ein Gerät verfügbar ist, das ein Benutzer reserviert hat, oder eine Ausleihe bevorsteht oder abläuft.
  + Erinnerungen für Rückgabe: Nutzer erhalten eine Benachrichtigung kurz vor Ablauf der Ausleihzeit, um eine rechtzeitige Rückgabe sicherzustellen.
  + Admin-Benachrichtigungen: Meldungen an Administratoren bei ausstehenden Rückgaben oder Geräten, die gewartet werden müssen.
* **Gerätehistorie und Zustand**
* Historie der Geräteverwendung: Jeder Benutzer kann sehen, wann und wie oft er ein bestimmtes Gerät ausgeliehen hat.
* Zustandsberichte und Wartung: Möglichkeit für Benutzer, Schäden oder Probleme an Geräten zu melden. Admins könnten regelmäßige Wartungszeiten festlegen.

**Ablauf**

Projektvorbereitung und Technische Konfiguration

Zu Beginn des Projekts stand die technische Konfiguration im Fokus. Es musste entschieden werden, welche Programmiersprachen, Frameworks und Entwicklungsumgebungen für die Umsetzung genutzt werden sollten. Um die Kommunikation und Zusammenarbeit zu erleichtern und ein einheitliches Entwicklungsumfeld zu schaffen, entschieden wir uns für eine gemeinsame Entwicklungsumgebung.

Nach gemeinsamer Abstimmung fiel die Wahl auf:

* Programmiersprache: Java
* Framework: Vaadin
* IDE: Eine als einheitliche Entwicklungsumgebung (z. B. IntelliJ IDEA), um die Kompatibilität zu gewährleisten und die Zusammenarbeit zu erleichtern.

Versionskontrolle und Kollaborationsplattform

Zur effizienten Zusammenarbeit und zur Versionskontrolle wurde GitHub als zentrales Repository verwendet. So konnten Teammitglieder Code-Änderungen einfach teilen und Konflikte in der Codebasis besser verwalten.

Datenbank

Als Datenbankmanagementsystem entschieden wir uns für PostgreSQL, um eine stabile und performante Grundlage für die Speicherung und Verwaltung unserer Daten zu schaffen.

Hauptfunktionen der Anwendung

Die Mini Web App wurde entwickelt, um eine einfache und benutzerfreundliche Oberfläche zur Geräteverwaltung anzubieten. Die zentralen Funktionen sind:

1. Gerätesuche und Anzeige
   * Benutzer haben die Möglichkeit, nach verfügbaren Geräten zu suchen und deren Details anzeigen zu lassen.
2. Geräteverwaltung durch Administratoren
   * Ein Administrator kann Geräte hinzufügen, deren Eigenschaften ändern und den Status eines Geräts anpassen. Die möglichen Status sind:
     + Vermietet: Das Gerät ist ausgeliehen.
     + Frei: Das Gerät ist verfügbar.
     + Fehlend: Das Gerät wurde nicht zurückgegeben und ist momentan nicht verfügbar.
3. Ausleihprozess für Benutzer
   * Ein Benutzer kann ein Gerät für einen bestimmten Zeitraum über einen Kalender auswählen. Dazu gibt er seinen Namen an, und die Ausleihe wird vermerkt.
   * Es ist möglich, Geräte nur für eine begrenzte Zeit zu reservieren, und der Rückgabeprozess kann jederzeit eingeleitet werden.
4. Buchungsverlauf
   * Der Benutzer kann im Nachhinein seine Ausleihen im Buchungsverlauf einsehen, was eine transparente Übersicht der bisherigen Reservierungen und Ausleihzeiten bietet.