Karlsruher Institut für Technologie

Präsentation des Entwurfs: Program for Splitting Expenses

Praxis der Softwareentwicklung Wintersemester 2024/25

PSE Android Gruppe 2 | 16. Januar 2025

Inhaltsverzeichnis

Begleitendes Beispiel

Verbuchen einer Buchung (angelehnt an $\langle F11 \rangle$)

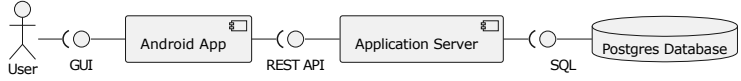
Szenario

- Nutzer will neue Buchung anlegen
- Parameter bereits eingegeben
- Hinzufügen Button gerade betätigt

Systemkomponenten

- **Android App**
  Nutzersicht auf System („Client“)
- **Application Server**
  Verwaltung des Systemzustands („Server“)
- **Datenbank**
  Datenhaltung

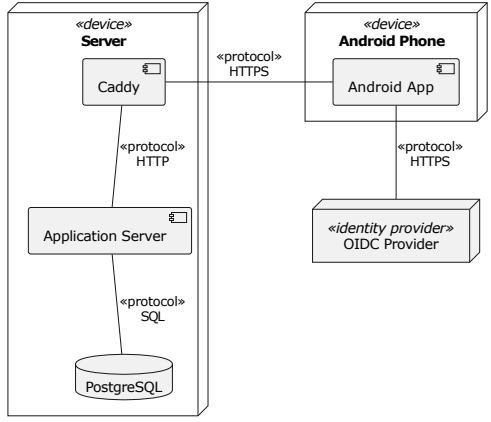
- *GUI*
  Basierend auf Skizzen
- *REST API*
- *SQL*



Extern: **Identity Provider** für Authentifizierung

Deployment

- Datenbank und Server auf gleichem Gerät
- Caddy als Reverse-Proxy für TLS Verschlüsselung
- Server als Debian Paket bereitgestellt.



Authentifizierung - Ausweisung

- Externer Identity Provider (IdP) weist Identität des Nutzer aus
- Nutzer meldet sich in Browser bei IdP an
- App erhält *Authorization Code*
- Mit diesem kann ein *ID Token* erhalten werden

Vorteile

- Kein System zur Verwaltung von Passwörtern benötigt
- Anzeigename kann vom IdP erhalten werden

Authentifizierung - Sessions

- Client weist sich mit *ID Token* bei Server aus
- Server erstellt neue Sitzung in der Datenbank
- Client erhält zur Session zugehörige *Access Token* und *Refresh Token*

Access Token für jede Anfrage
Refresh Token zum Erneuern des *Access Token*

Vorteile

- Unabhängigkeit von IdP Implementierung
- Volle Kontrolle über Authentifizierung

Client: MVVM Architektur

View

Beschreibt UI Aussehen und empfängt UI Zustand vom View Model.

View Model

Bestimmt UI Zustand. Eingaben werden auf Operationen im Model abgebildet.

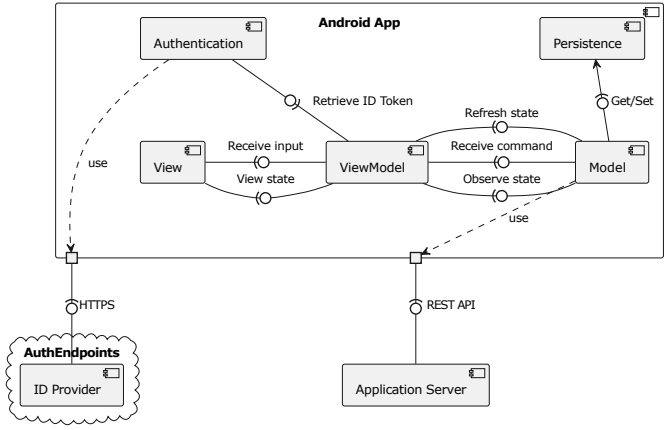
Model

Verwaltet lokale Sicht des Serverzustands (Modellzustand). Enthält restliche App Logik.

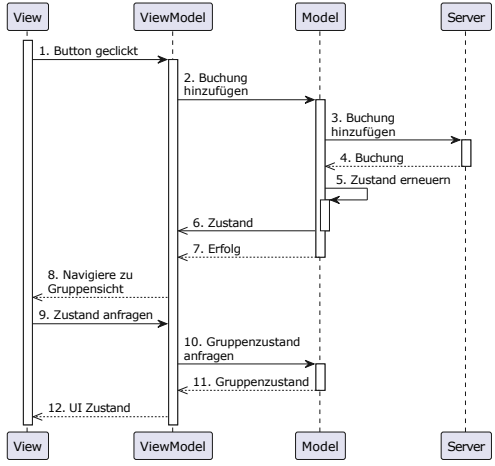
Client: MVVM Architektur - Abwägung

- UI Aussehen leicht anpassbar
- Unabhängigkeit der UI vom Server
- Flexibilität
  - Erlaubt mehrere Datenquellen für Modellzustand
  - Abgefragter Zustand kann persistiert werden
- Größere Komplexität

Client: Komponenten



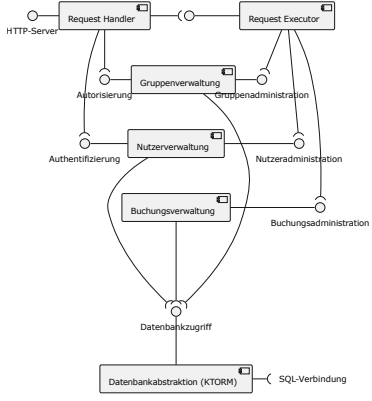
Begleitendes Beispiel - Client Abläufe



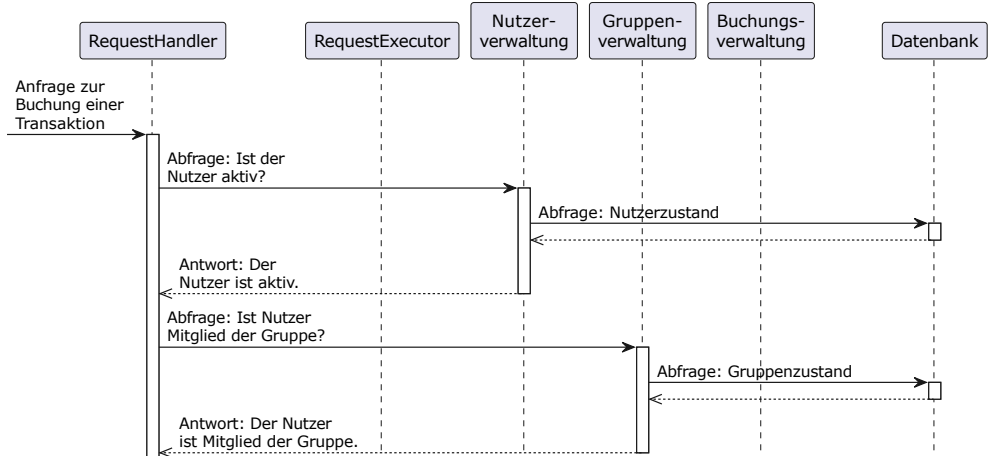
Server: Schichtenarchitektur

- **Request Handler** empfängt und validiert Anfragen
  - Dabei werden auch Berechtigungen geprüft
- **Request Executor** verarbeitet Anfragen
- **Verwaltungskomponenten** implementieren spezifische Teile der Server Logik
- **Datenbankabstraktion** für Datenbankzugriffe

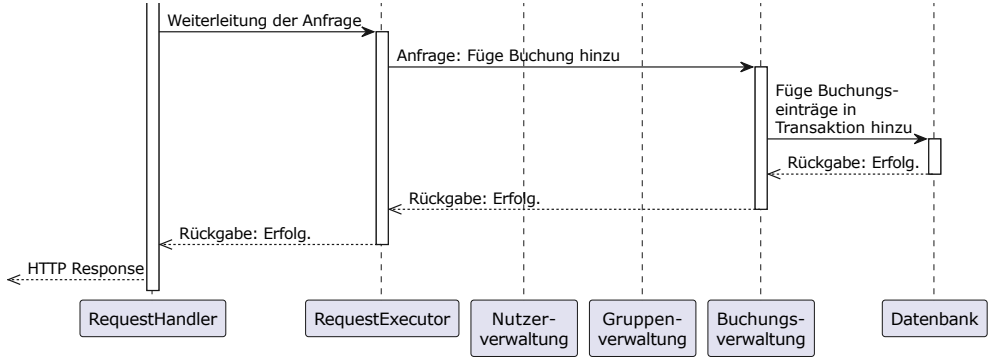
Server Komponenten



Begleitendes Beispiel - Server Abläufe

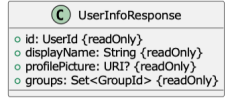


Begleitendes Beispiel - Server Abläufe



REST API

- DTOs für Beschreibung der HTTP Inhalte
- Kein Polling
- Bulk Requests
- Versionierung



`/v1/users` statt `/v1/user`

Begleitendes Beispiel - API Calls

Buchung

PUT `/v1/transactions`
Request Body: `Map<GroupId, TransactionRequest>`

Gruppenzustand

GET `/v1/groups`
Request Body: `Set<GroupId>`
Response Body: `Map<GroupId, GroupInfoResponse>`

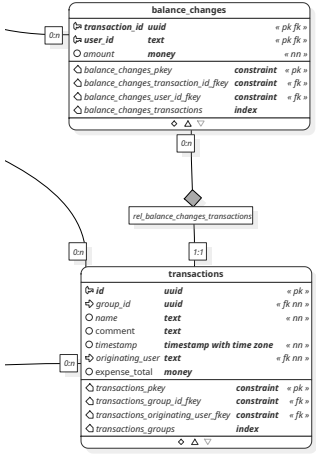
Zusätzlich: Gruppenmitglieder mit **GET** `/v1/users`

Schnittstelle zur Datenbank

Ktorm Library für Datenbankzugriff

- Datenbankschema in DSL
- Typensichere SQL Anfragen

Begleitendes Beispiel - Schema



View Umsetzung mit Compose

- UI Beschreibung mit Jetpack Compose
- „Composables“ als UI Bausteine
- Jeder Screen wird durch eine Composable beschrieben

Vorteile

- UI Beschreibung in Kotlin
- Einfache Datenbindung zu View Model

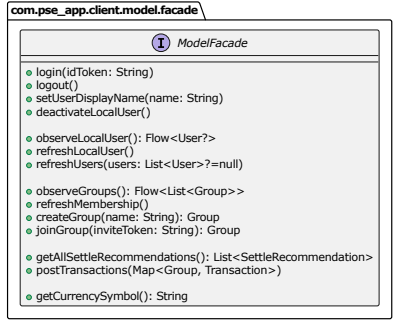
ViewModel Umsetzung

- Eine ViewModel Klasse pro Screen
- Beschreibung des UI Zustands via Datenklasse
- `StateFlow` für Syncronisation

Beispiel:
`PaymentViewModel` für Erstellen der Buchung Besitzt `confirm()` für Bestätigung und `PaymentState`

Model Umsetzung

- Zugriff über **Fassaden** Schnittstellen
  - *ModelFacade*, *User*, *Group*
- Verwaltung des Modellzustands über sog. **Repositories**
  - *UserRepo*, *TransactionRepo*, *GroupRepo*
- Wrapper für Server API
- `Flow` statt Callbacks



Server Umsetzung

- Routing mit *Ktor*
- Dependency Injection für loose Kopplung
- `BigDecimal` für Geldbeträge

Fragen

Danke für die Aufmerksamkeit!

Fragen

Fragen?