Clockwise

Entwurfsphase

20.06.2024

Intuitive Time Tracking

Team

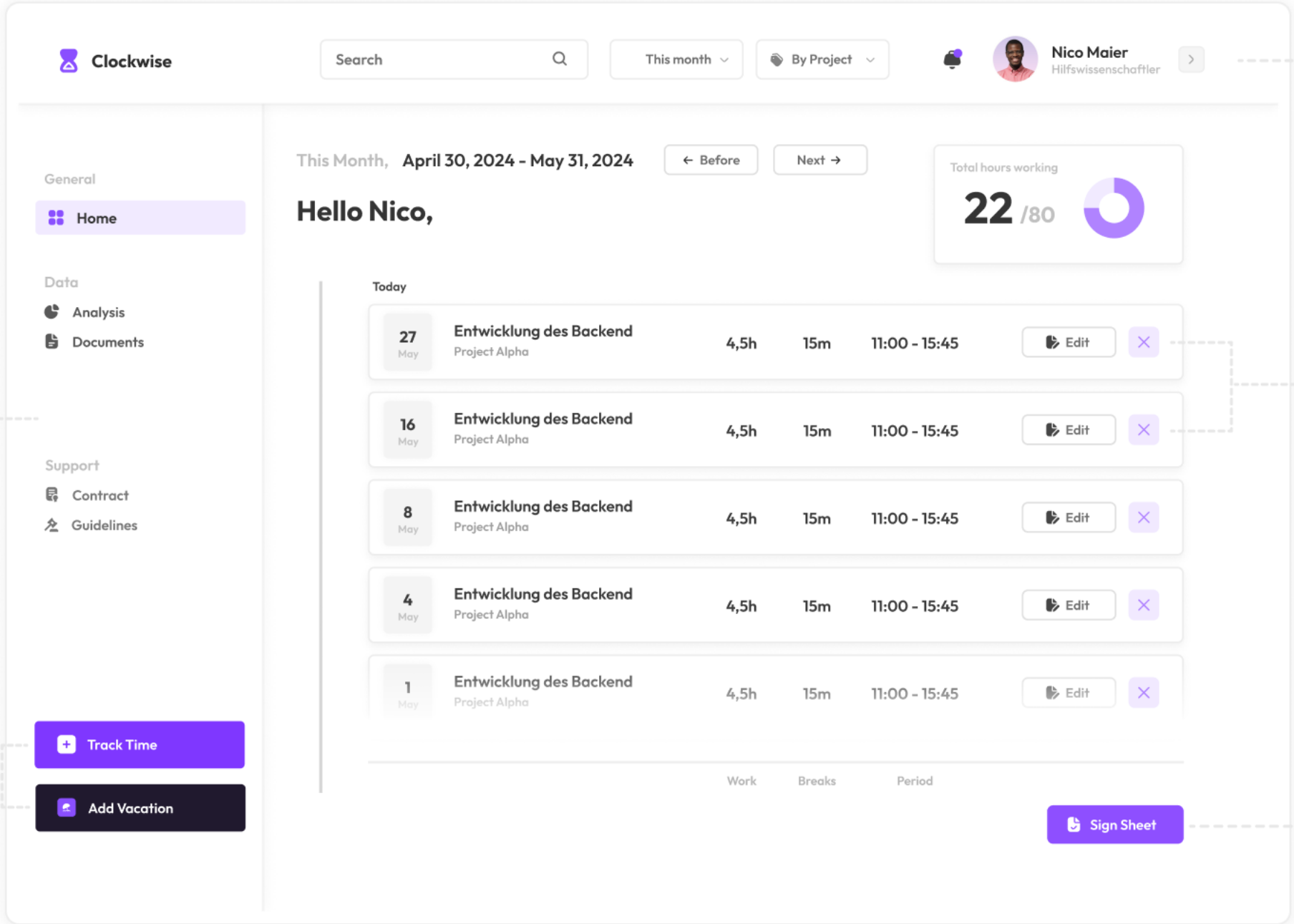
Dominik Pollok
Phil Gengenbach
Alina Petri
José Ayala
Johann Kohl

Betreuer

Maximilian Li

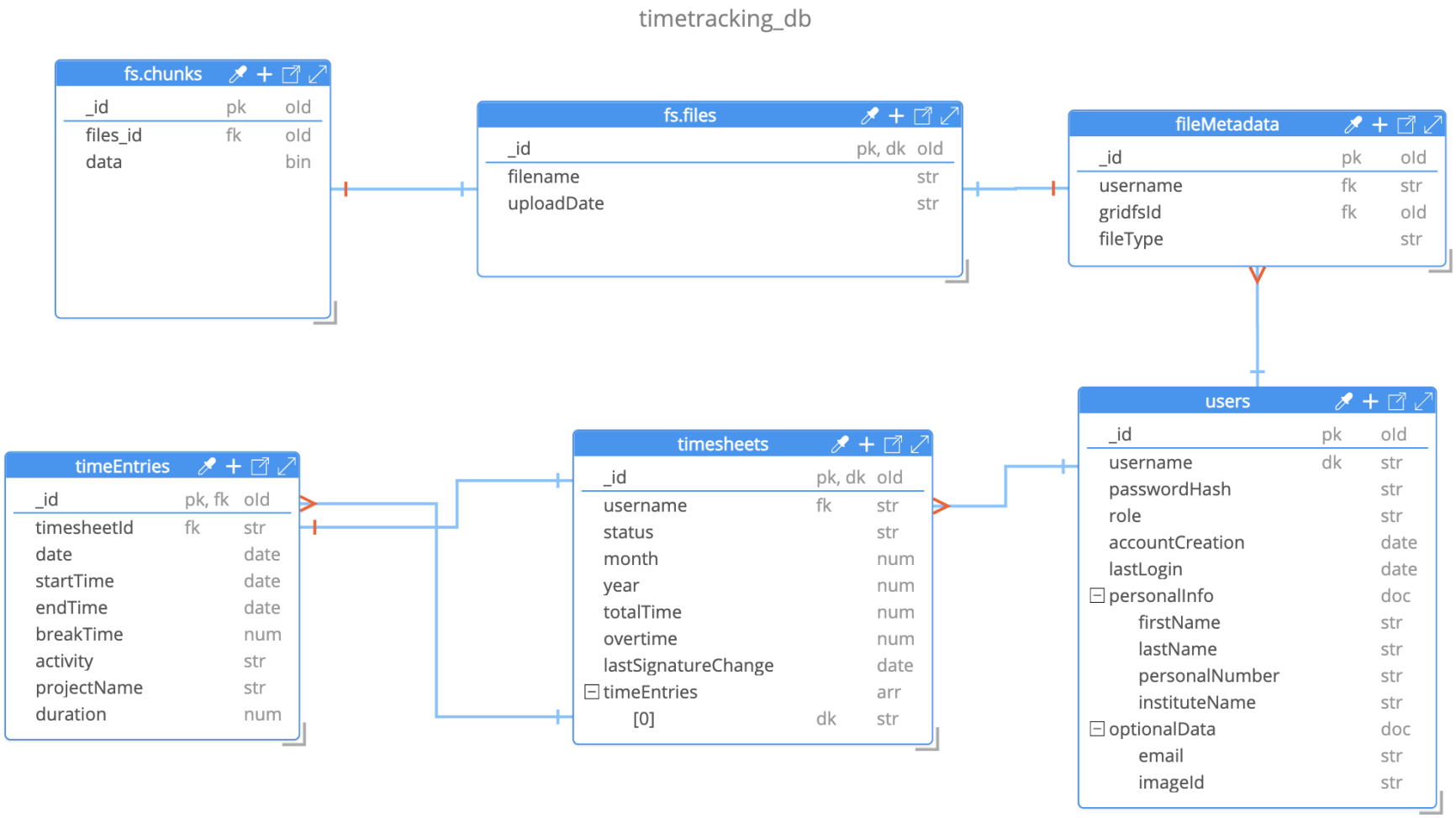
Intuitive Robots Lab

UI / Mockup


Clockwise
Search
This month
By Project
Nico Maier
Hilfswissenschaftler
Profile Bar
General
Home
Data
Analysis
Documents
Support
Contract
Guidelines
Navbar
This Month, April 30, 2024 - May 31, 2024
← Before
Next →
Hello Nico,
Total hours working
22 /80
Today
27 May Entwicklung des Backend Project Alpha 4,5h 15m 11:00 - 15:45 Edit
16 May Entwicklung des Backend Project Alpha 4,5h 15m 11:00 - 15:45 Edit
8 May Entwicklung des Backend Project Alpha 4,5h 15m 11:00 - 15:45 Edit
4 May Entwicklung des Backend Project Alpha 4,5h 15m 11:00 - 15:45 Edit
1 May Entwicklung des Backend Project Alpha 4,5h 15m 11:00 - 15:45 Edit
Work
Breaks
Period
Working Days
Track Time
Add Vacation
Quick Actions
Sign Sheet

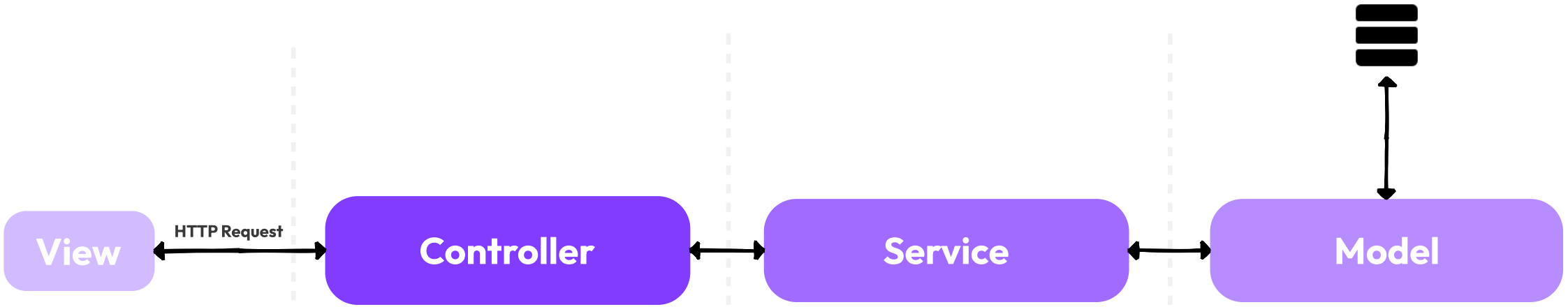
1. Datenbankentwurf
2. MVCS-Architektur
   - 2.1 MVCS am Beispiel User
3. Klassen Entwurf (UML)
   - 3.1 User Factory & Domänenklassen
   - 3.2 Timesheet & Time Entry
   - 3.3 Dateiverwaltung
4. Beispielablauf (Sequenzdiagramm)

Datenbankentwurf



Model-View-Controller-Service

MVCS



View Layer

- Darstellung der Benutzeroberfläche
- Nutzerinteraktion

Controller Layer

- Nimmt GET/POST/PATCH/DELETE Anfragen entgegen
- Weiterleitung an entsprechende Service Klassen

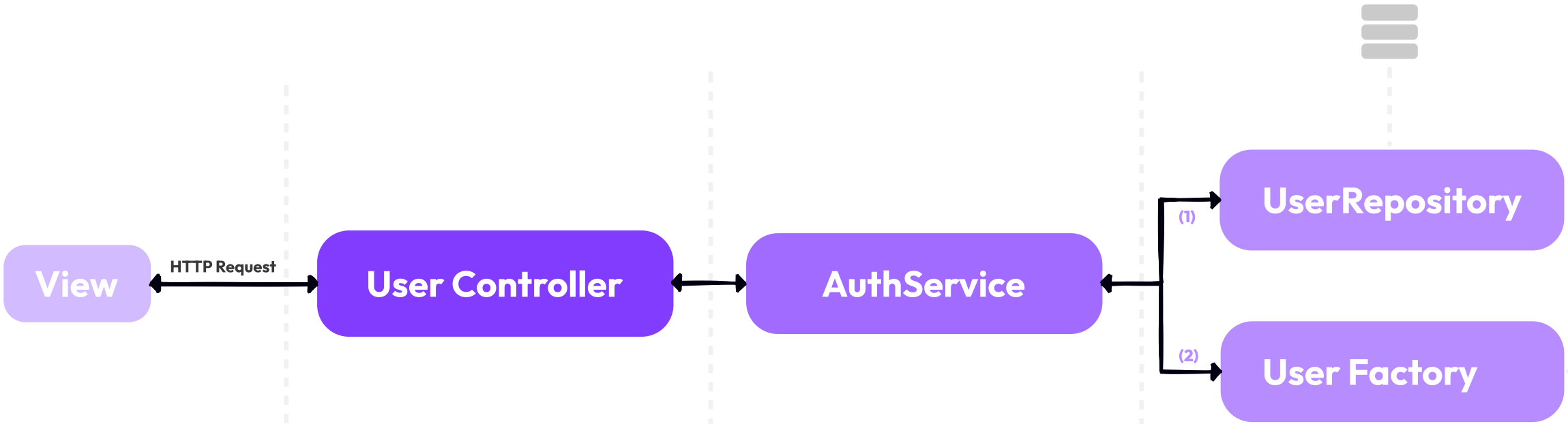
Service Layer

- Beinhaltet die Geschäftslogik
- Zugriff auf relevante Model-Klassen
- Validierung der Eingabedaten

Model Layer

- Repository-Klassen kommunizieren mit Datenbank
- Modellierung der unterschiedlichen Entitäten

Beispiel: Login Vorgang



Frontend

- Benutzeroberfläche
- Eingabe von Login Daten
- Daten in HTTP-Anfrage verpackt -> an Backend

Controller Layer

- Nimmt Login HTTP Anfrage von Frontend entgegen (JSON)
- Weiterleitung an Authentication Service Klasse

Service Layer

- Zuständig für Authentifizierung
- Überprüft Gültigkeit der Benutzerdaten -> Anfrage an UserRepository

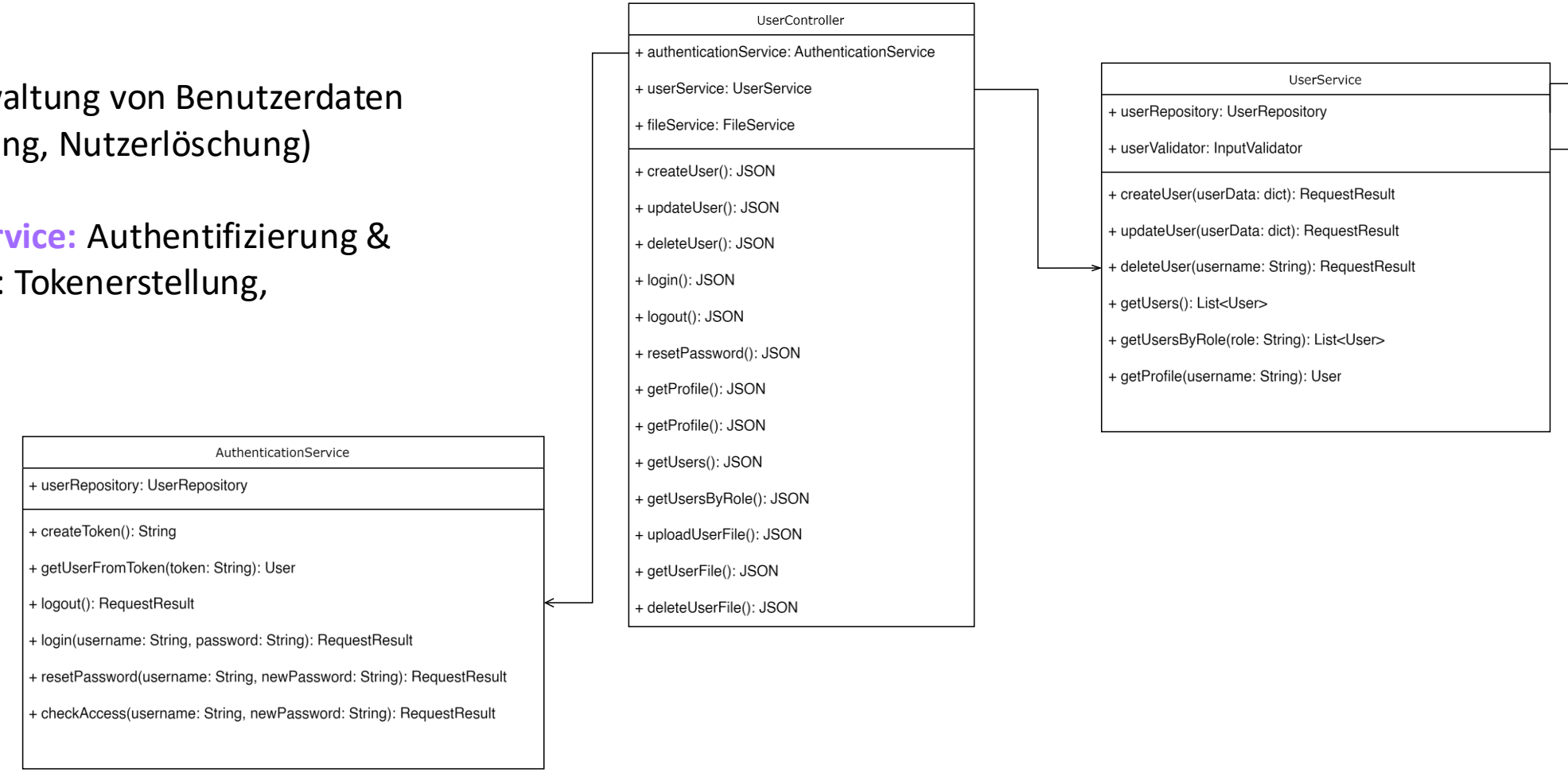
Model Layer

- User-Repo kommuniziert mit Datenbank (Gültigkeitsprüfung)
- User Factory zur Erstellung von User Objekten

User: Controller / Service

UML

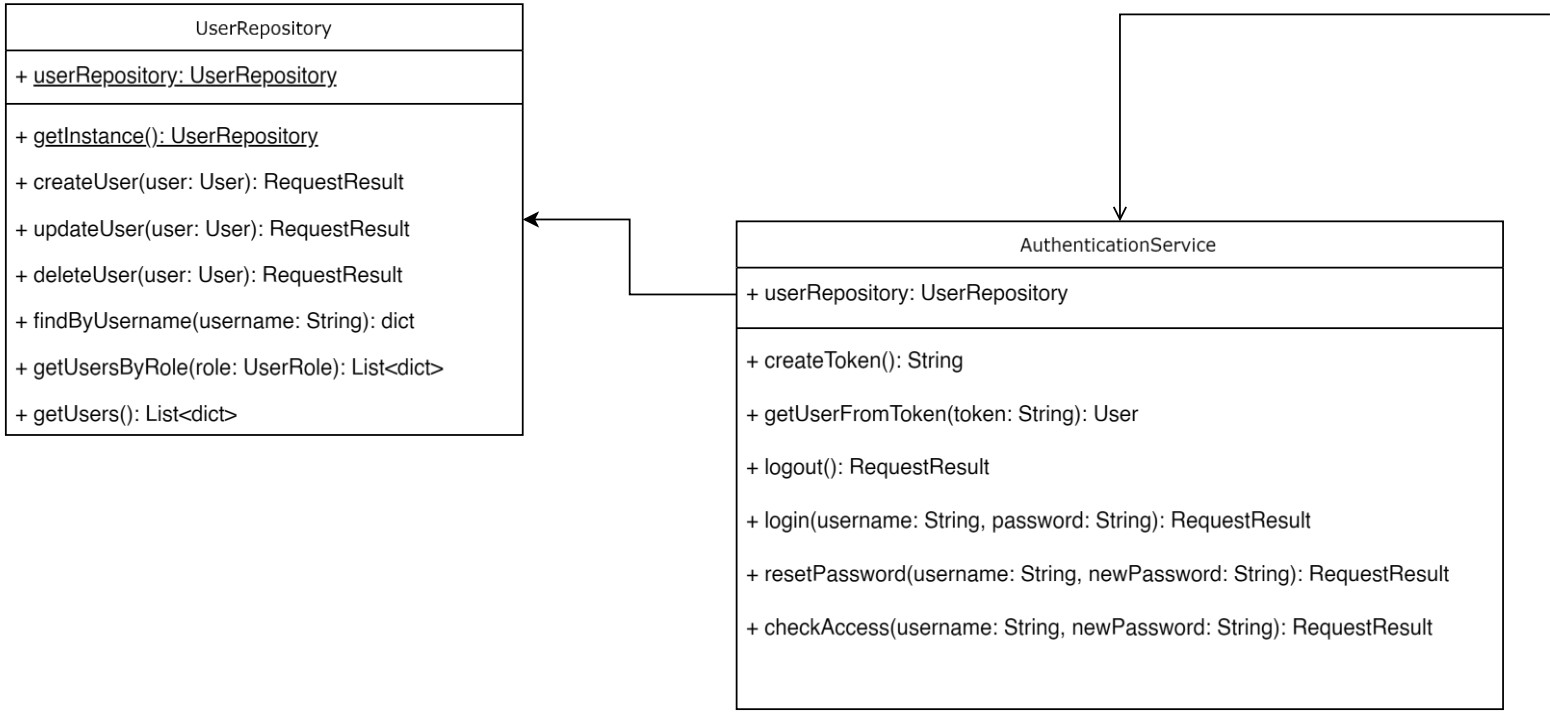
- **User Service:** Verwaltung von Benutzerdaten (z.B.: Kontoerstellung, Nutzerlöschung)
- **Authentication Service:** Authentifizierung & Autorisierung (z.B.: Tokenerstellung, Zugriffkontrolle)



User: Repository

UML

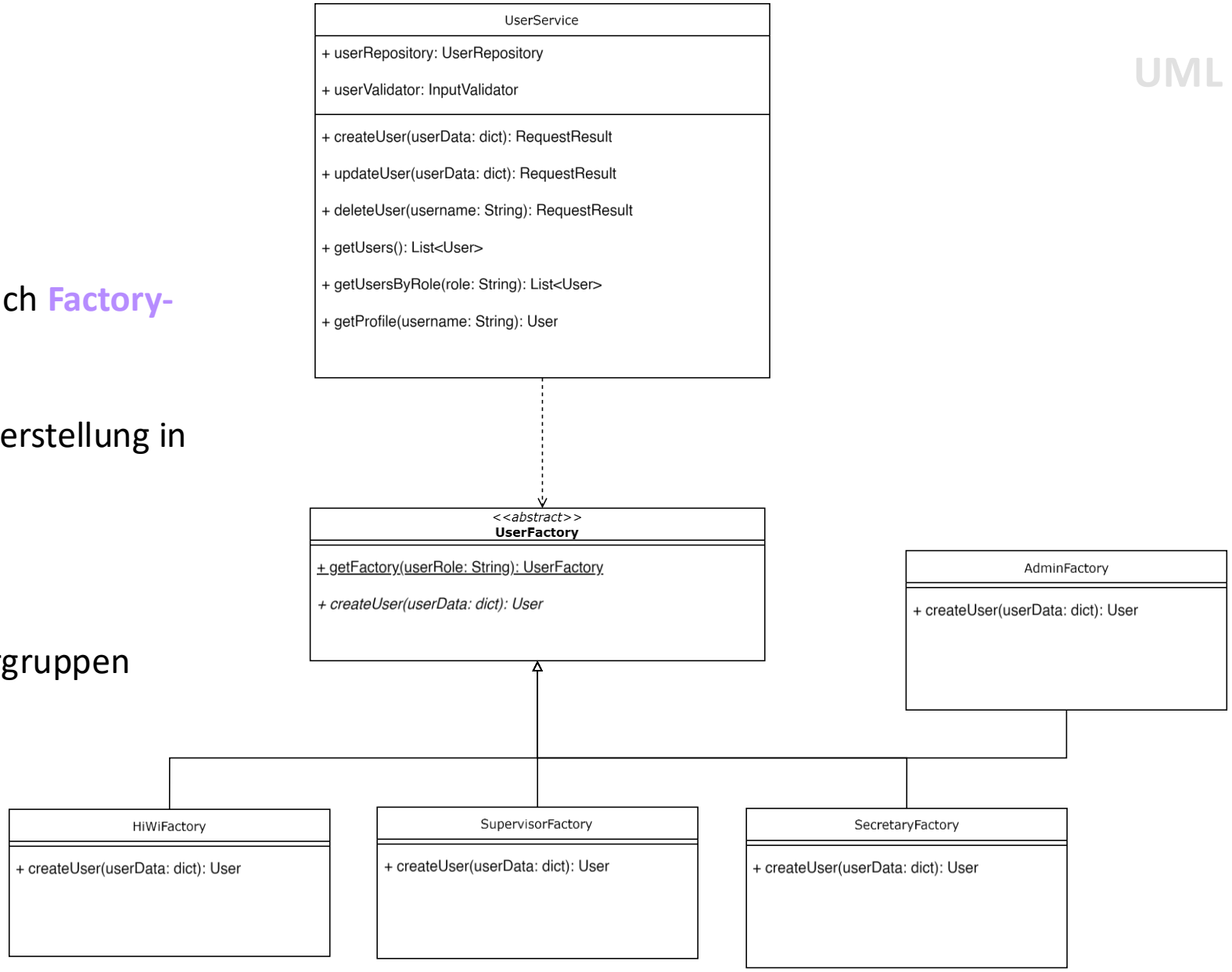
- Zentraler Bestandteil des Modells (für Datenbankzugriff zuständig)
- **Singleton:** nur eine Instanz -> Verwaltung der Datenbankverbindung effizienter & übersichtlicher
- **Repository-Muster:** abstrahiert Komplexität, klare Trennung von Datenbankzugriff und Geschäftslogik



User: Factory

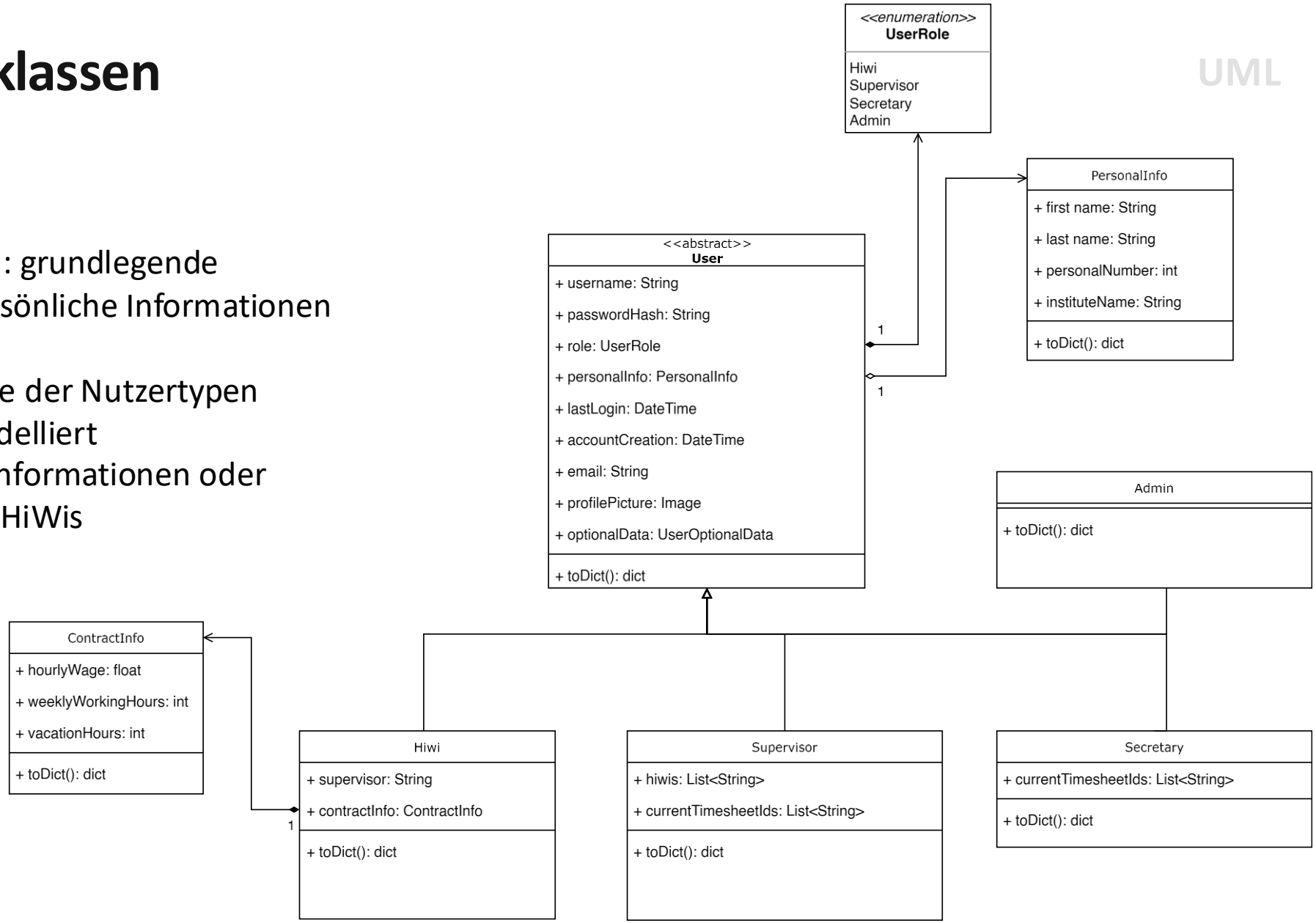
UML

- Erstellung von Nutzerobjekten nach **Factory-Method-Muster**
- Auslagerung der Logik zur Nutzererstellung in Factory Klassen
- Erhöhung der Wartbarkeit und Anpassungsfähigkeit
  z.B.: bei Integration neuer Nutzergruppen



User: Domänenklassen

- **User** Klasse im Zentrum: grundlegende Attribute, Rolle und persönliche Informationen
- Spezifischen Bedürfnisse der Nutzertypen durch Unterklassen modelliert
  z.B.: **HiWi** mit Vertragsinformationen oder **Supervisor** mit Liste an HiWis

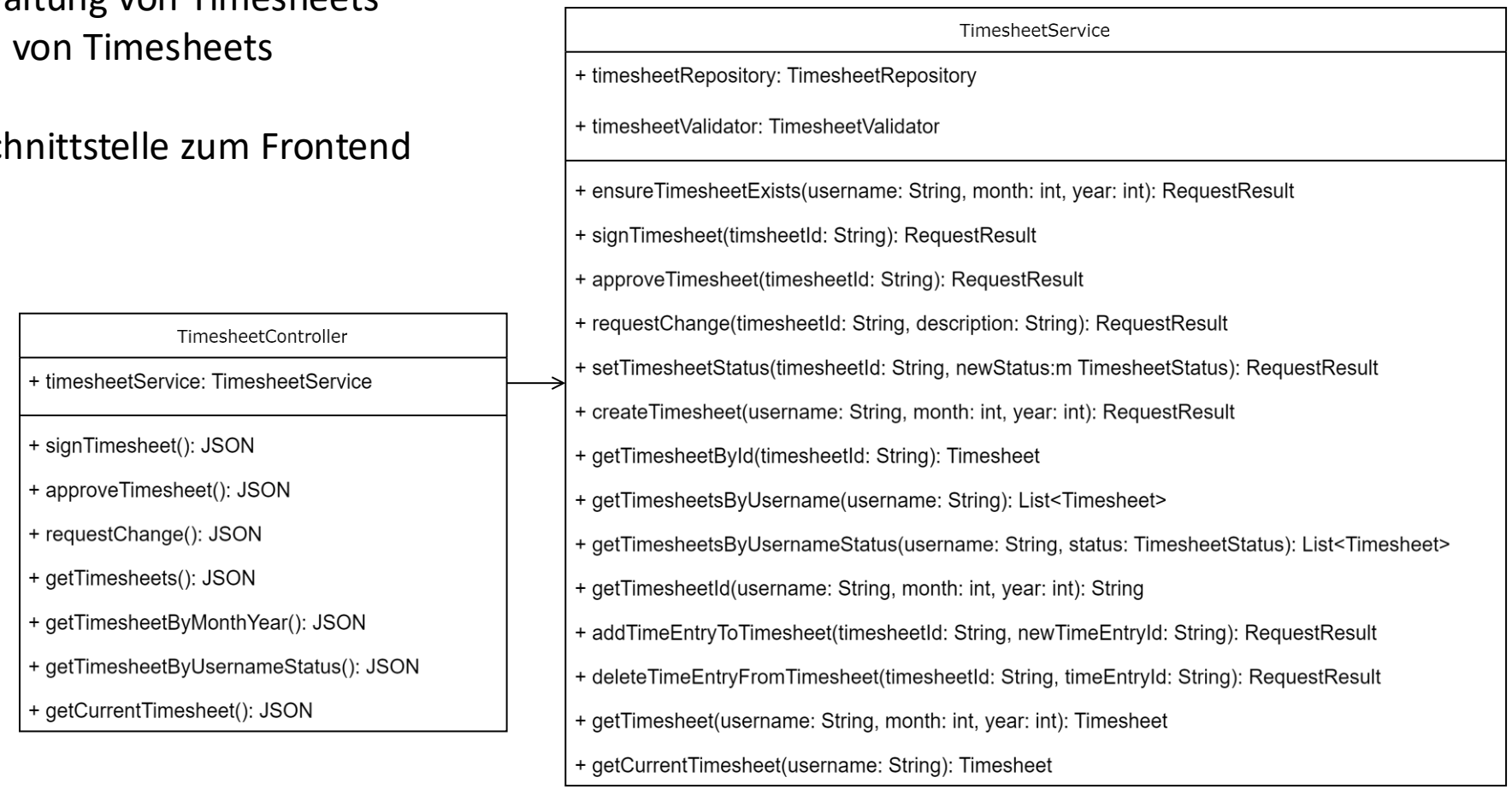


UML

Timesheet: Controller / Service

UML

- **Timesheet Service:** Verwaltung von Timesheets
  z.B. Erstellen & Signieren von Timesheets
- **Timesheet Controller:** Schnittstelle zum Frontend

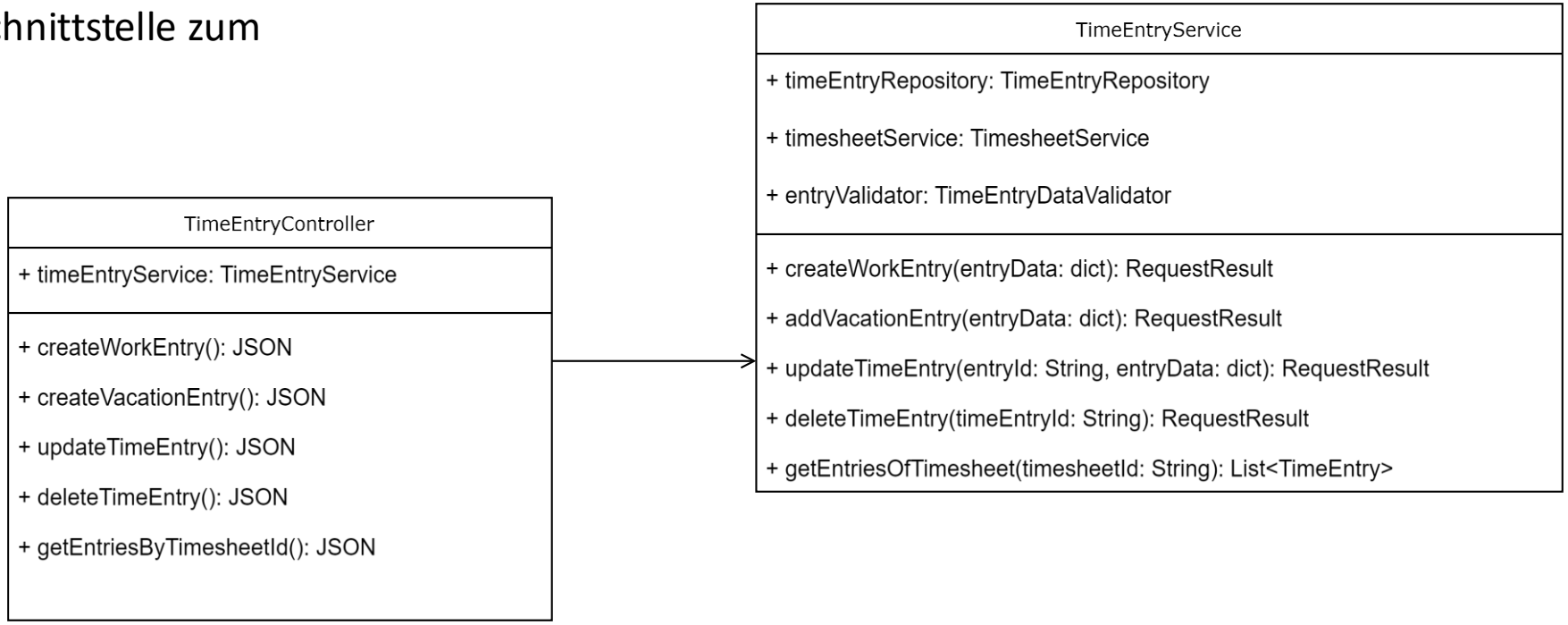


Time Entry: Controller / Service

UML

- **Time Entry Service:** Verwaltung von Time Entries
  z.B.: Erstellen & Bearbeiten von Zeiteinträgen

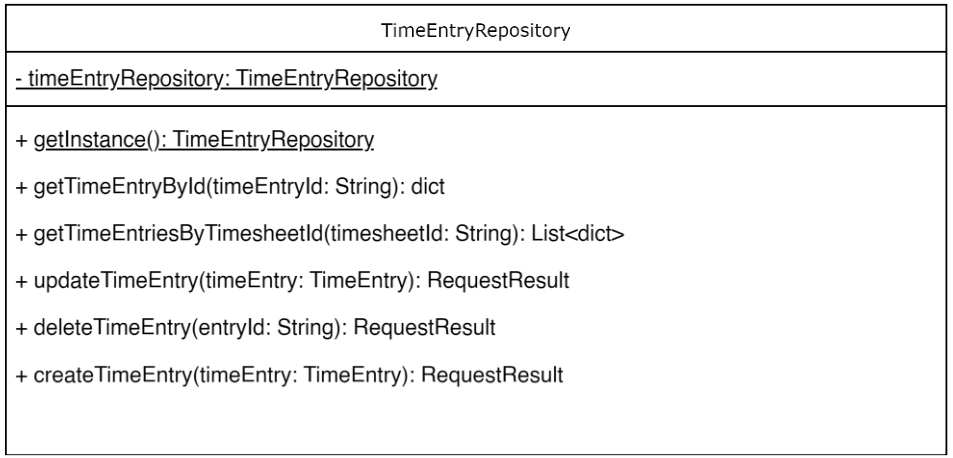
- **Time Entry Controller:** Schnittstelle zum Frontend

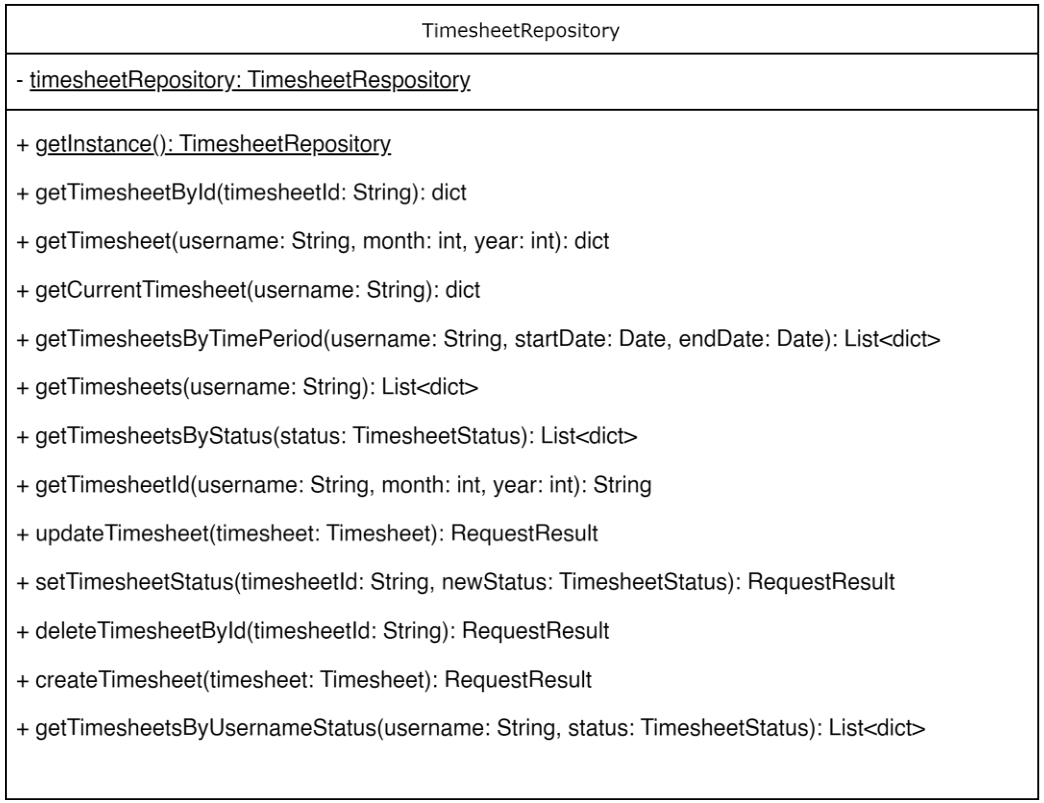


Timesheet / Time Entry: Repository

UML

- Datenbankzugriff zur Persistierung von Timesheets und Time Entries
- **Singleton & Repository-Muster**

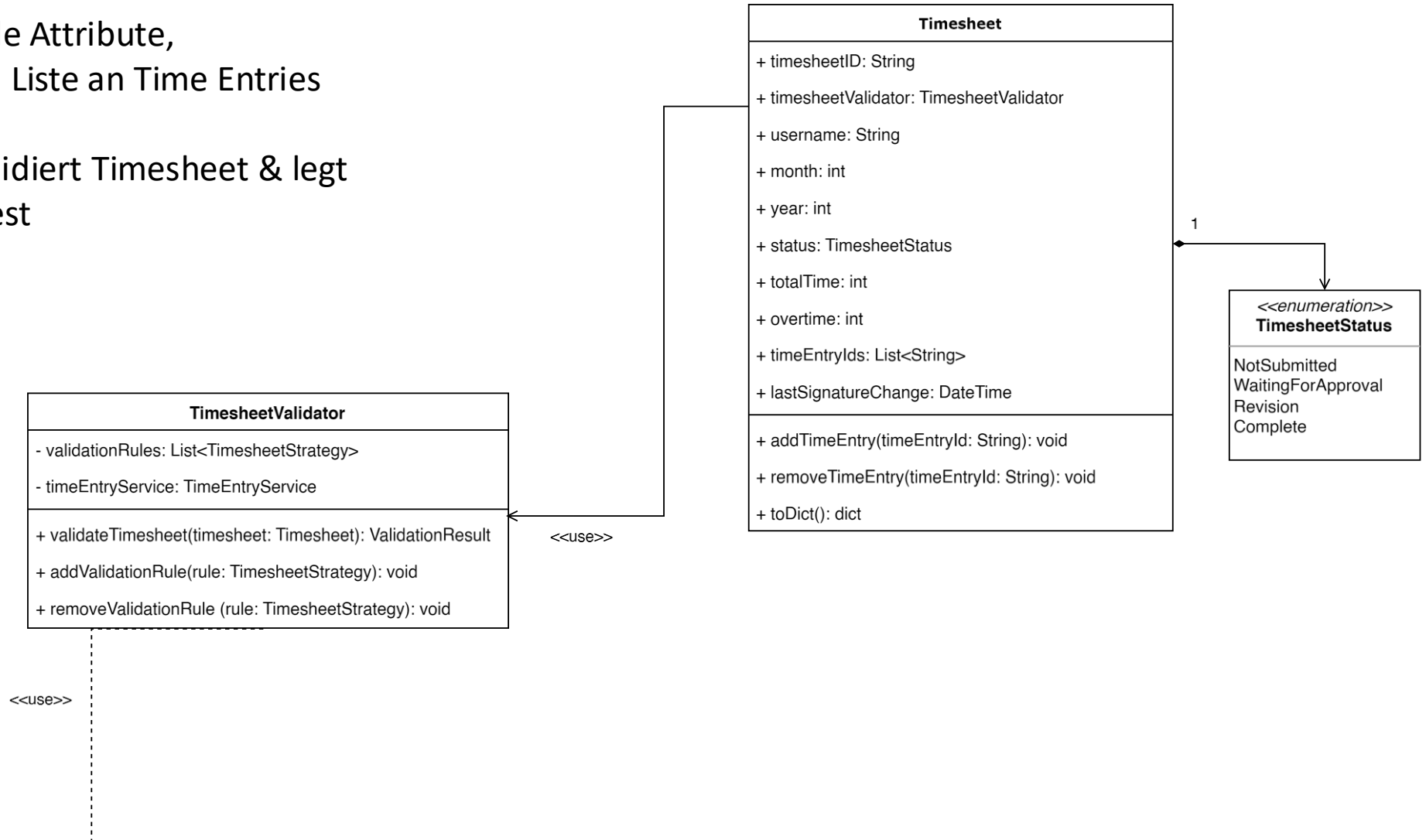




Timesheet: Domänenklassen

UML

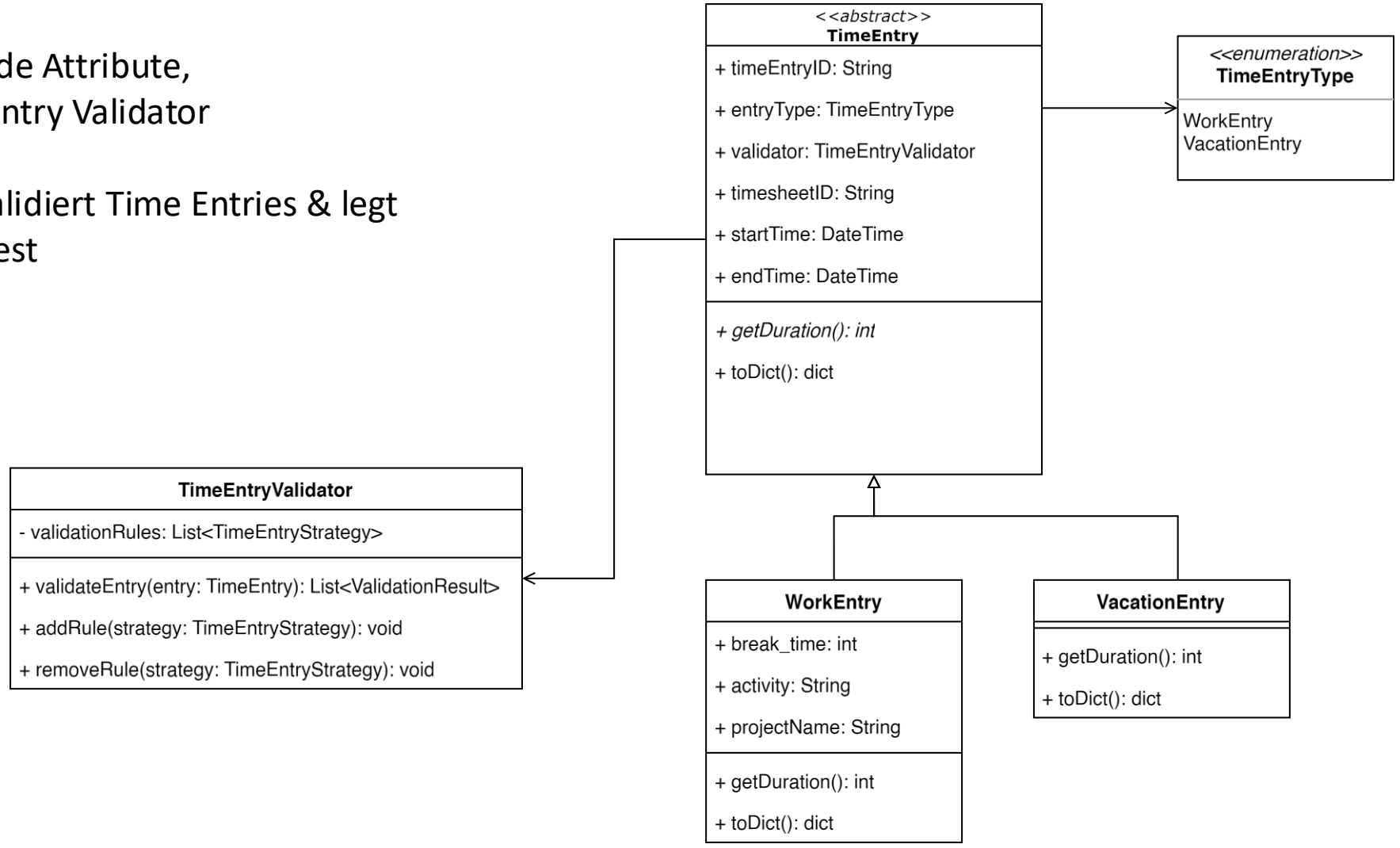
- **Timesheet:** grundlegende Attribute, Timesheet Validator und Liste an Time Entries
- **Timesheet Validator:** validiert Timesheet & legt Validierungsstrategien fest



Time Entry: Domänenklassen

UML

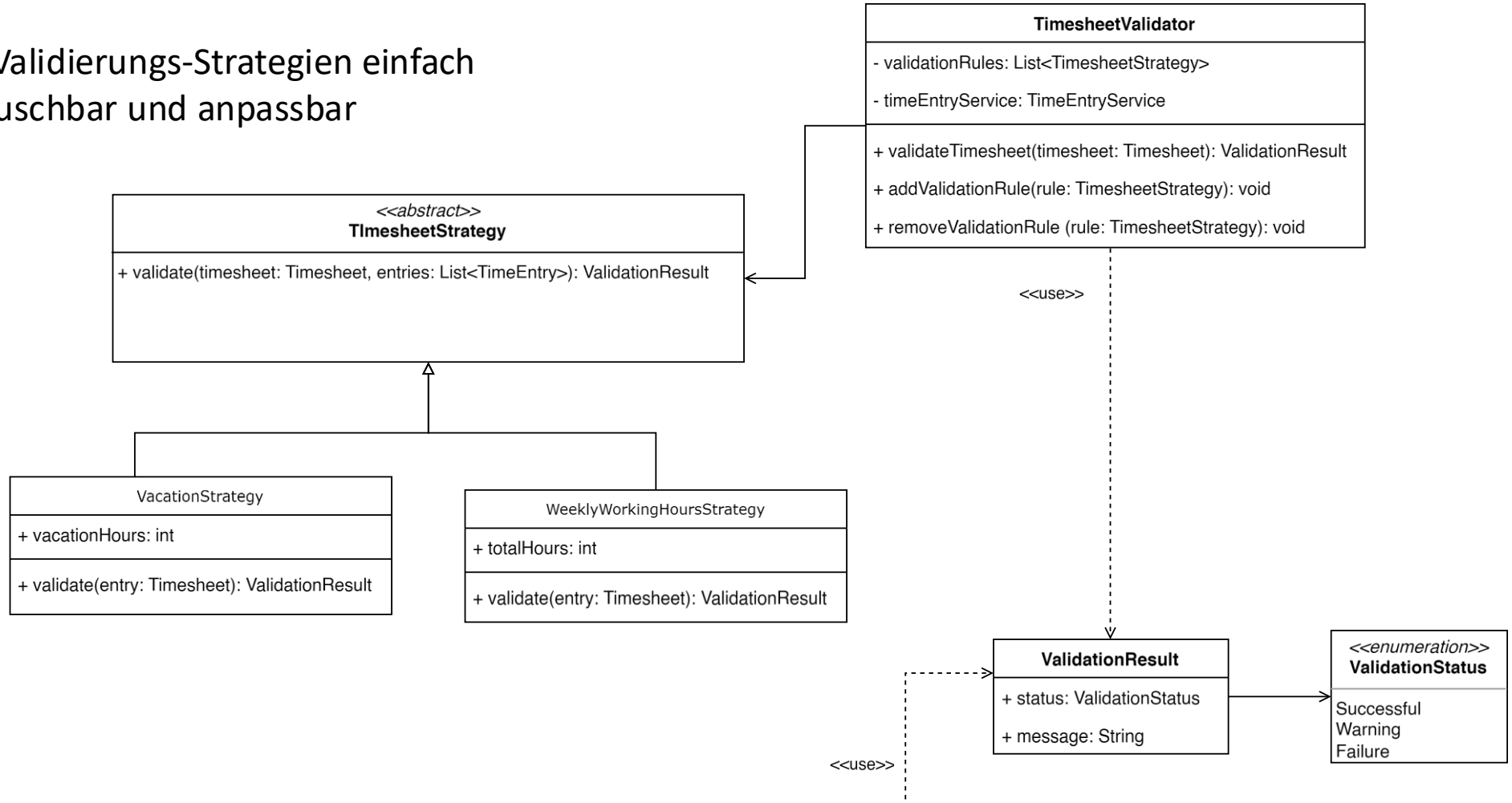
- **Time Entry:** grundlegende Attribute, Time Entry Type, Time Entry Validator
- **Time Entry Validator:** validiert Time Entries & legt Validierungsstrategien fest



Timesheet: Validierung

UML

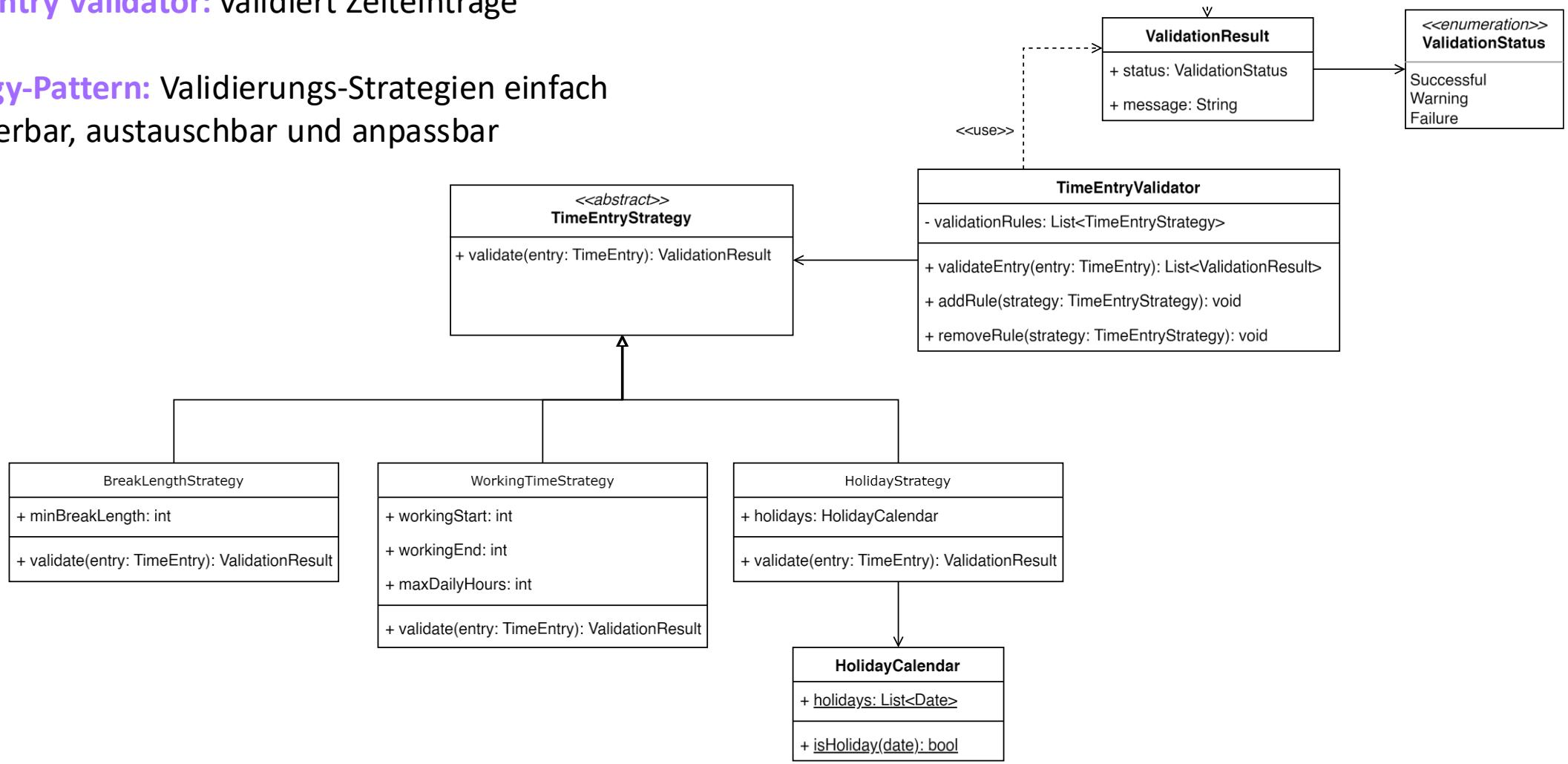
- **Timesheet Validator:** validiert Timesheet bei Signatur
- **Strategy-Pattern:** Validierungs-Strategien einfach erweiterbar, austauschbar und anpassbar



Time Entry: Validierung

UML

- **Time Entry Validator:** validiert Zeiteinträge
- **Strategy-Pattern:** Validierungs-Strategien einfach erweiterbar, austauschbar und anpassbar

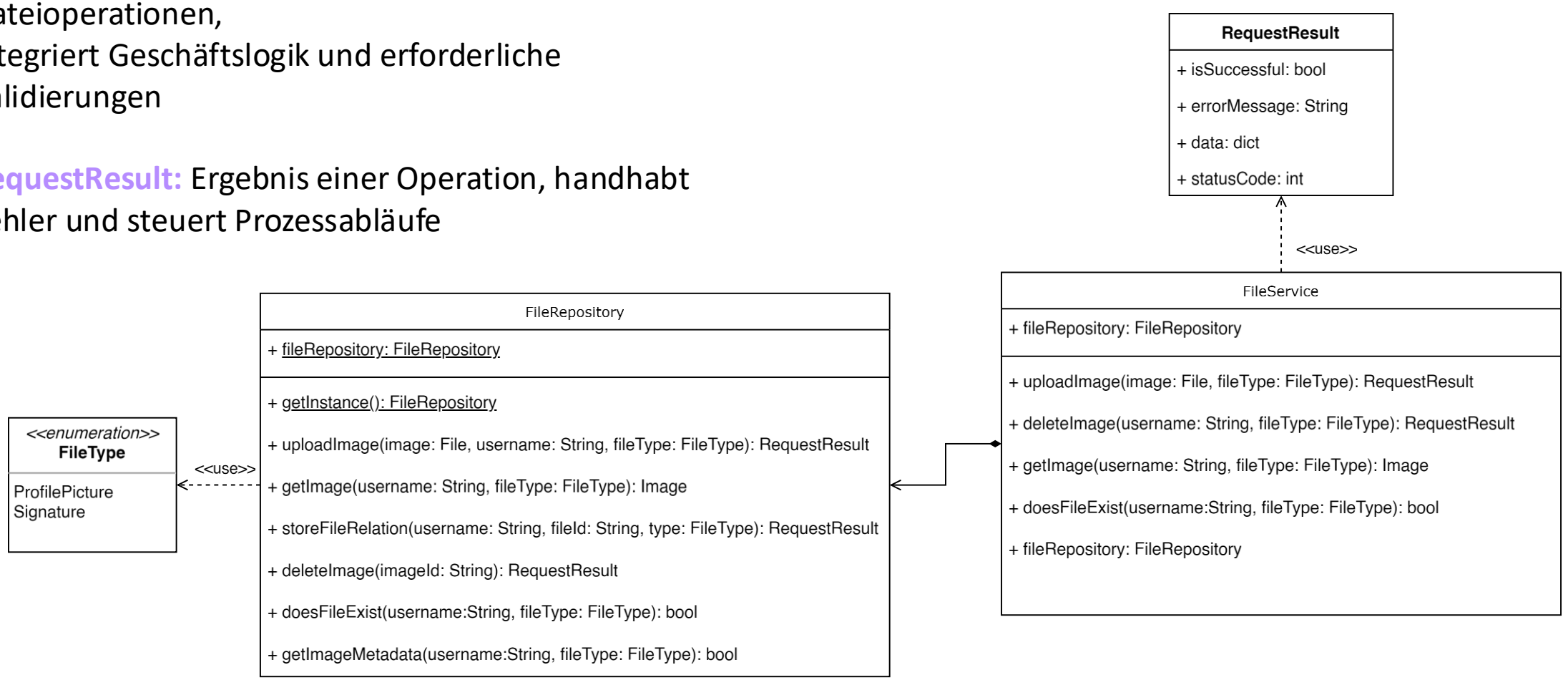


Dateiverwaltung

UML

- **FileService:** Verantwortlich für sämtliche Dateioperationen,
  integriert Geschäftslogik und erforderliche Validierungen

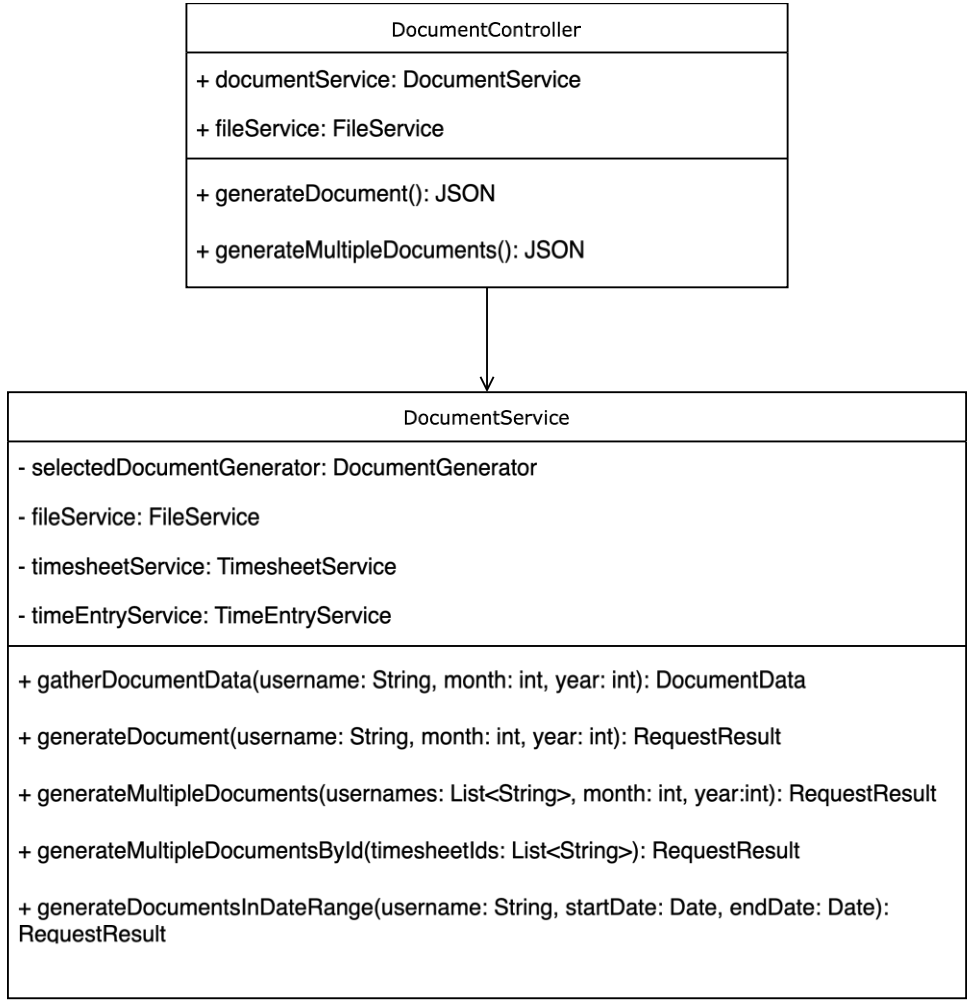
- **RequestResult:** Ergebnis einer Operation, handhabt Fehler und steuert Prozessabläufe



Document: Controller/Service

UML

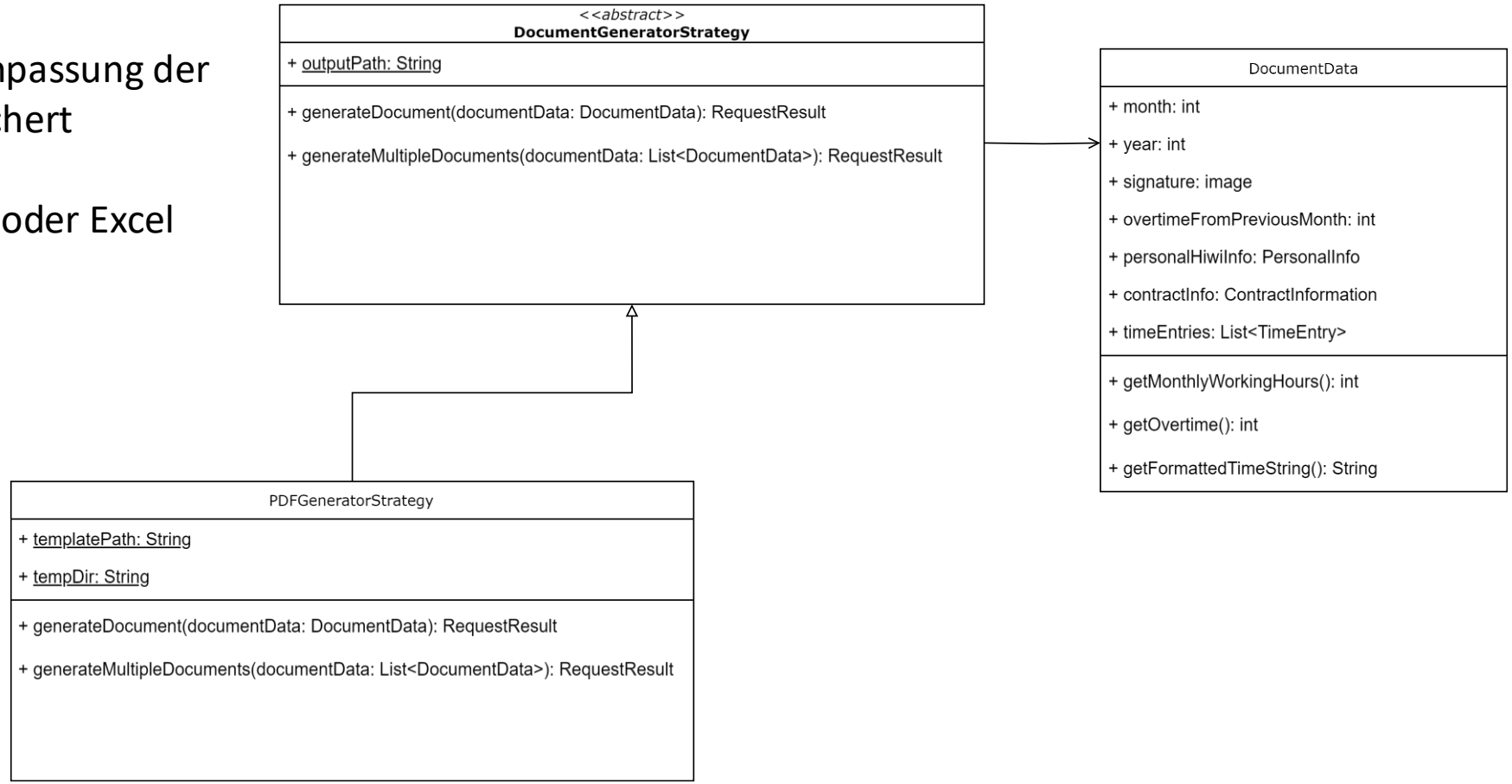
- **Document Controller**: Steuerung und Koordination der Dokumentenerstellung
- **Document Service:** Erzeugung, Verwaltung und Verarbeitung von Dokumenten.
- Anbindung von Document Service an **Strategy-Pattern**



DocumentGeneratorStrategy:

UML

- **Strategy-Pattern:** flexible Anpassung der Algorithmen zur Laufzeit, sichert Erweiterbarkeit
  z.B.: bei Integration von .txt oder Excel Format

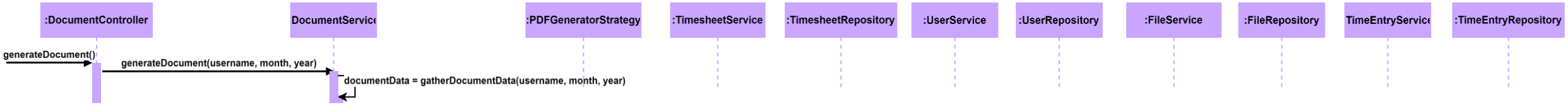


Erstellung eines PDF-Dokuments

Sequenzdiagramm

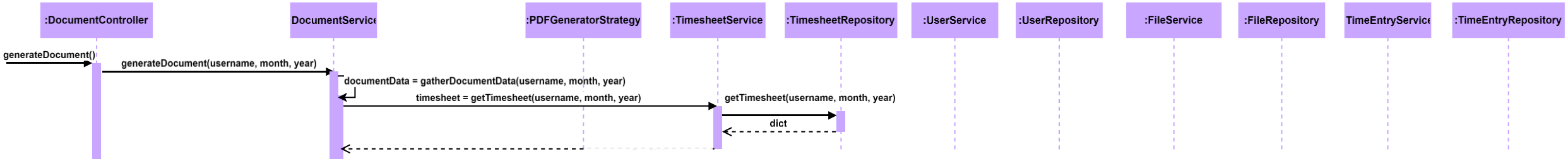
Erstellung eines PDF-Dokuments

Sequenzdiagramm



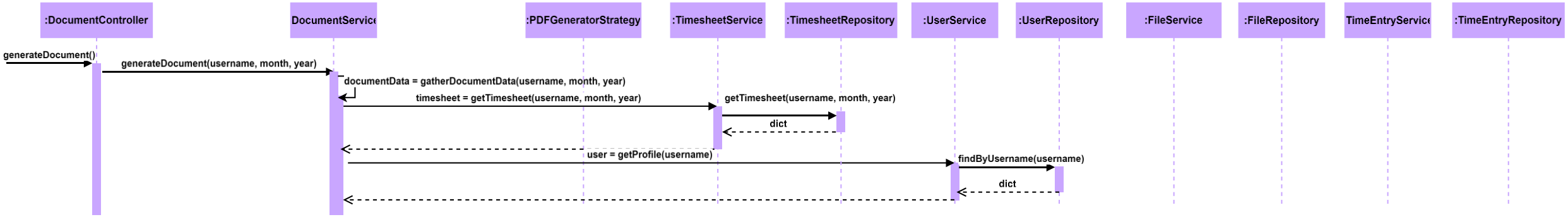
Erstellung eines PDF-Dokuments

Sequenzdiagramm



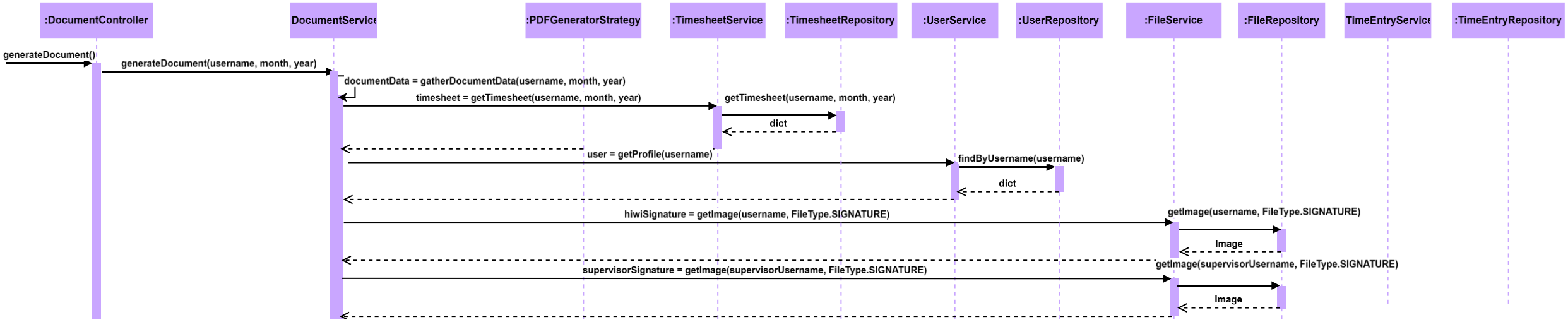
Erstellung eines PDF-Dokuments

Sequenzdiagramm



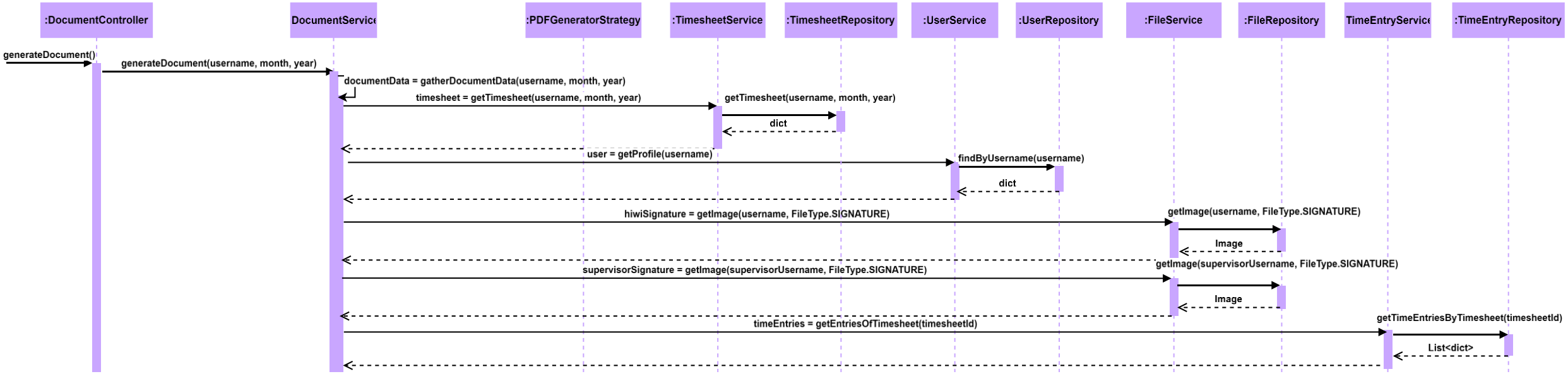
Erstellung eines PDF-Dokuments

Sequenzdiagramm



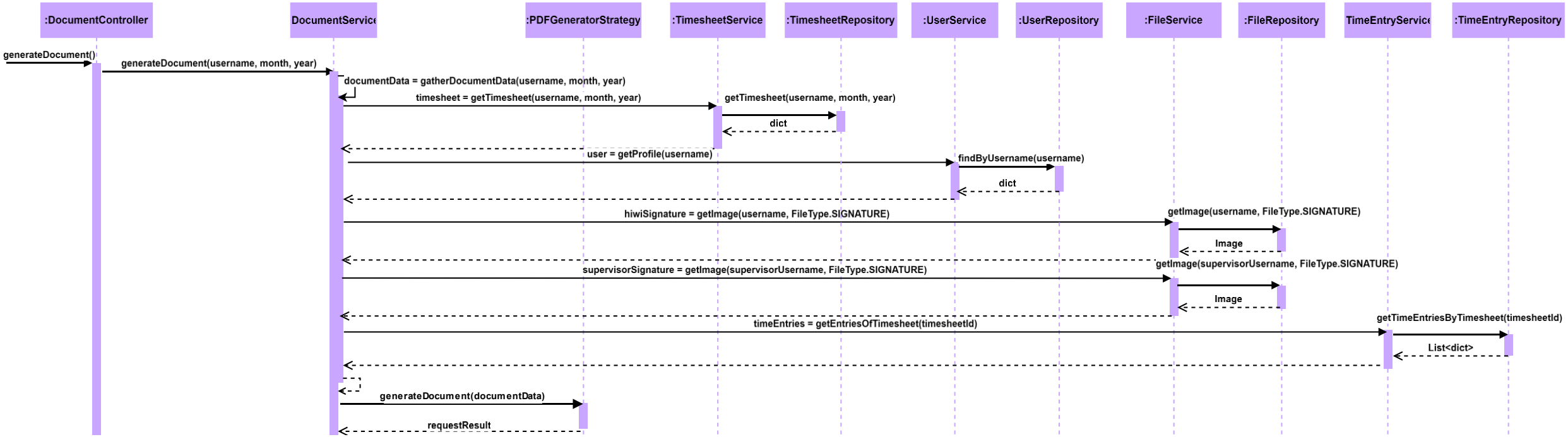
Erstellung eines PDF-Dokuments

Sequenzdiagramm



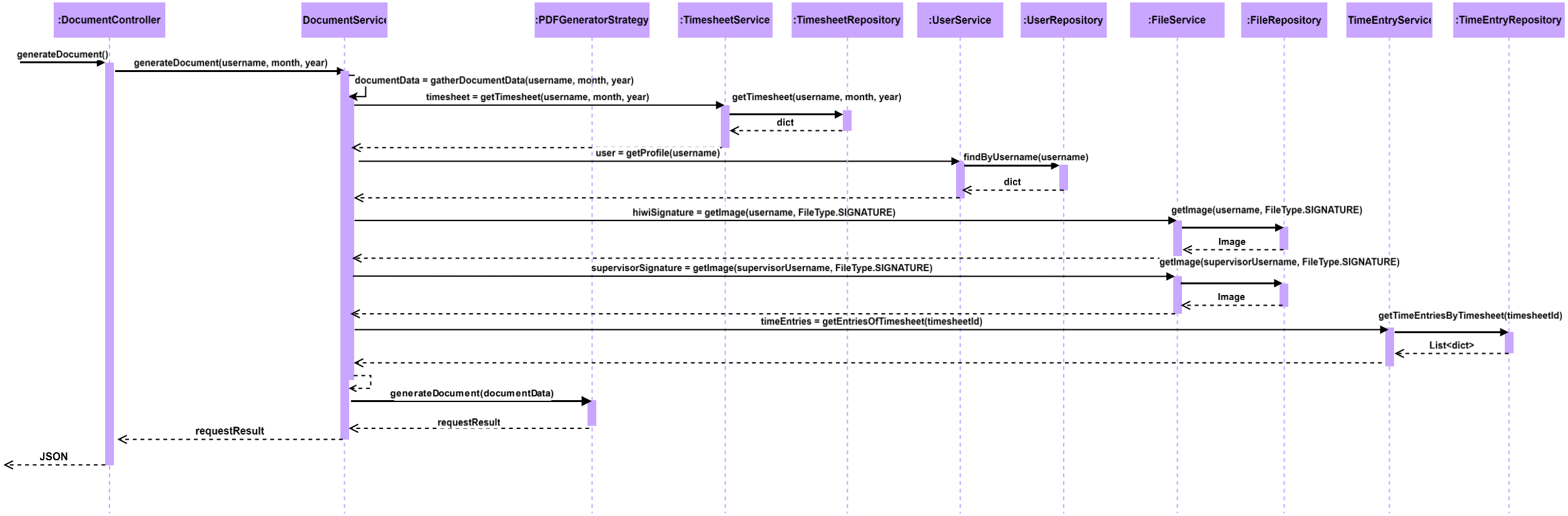
Erstellung eines PDF-Dokuments

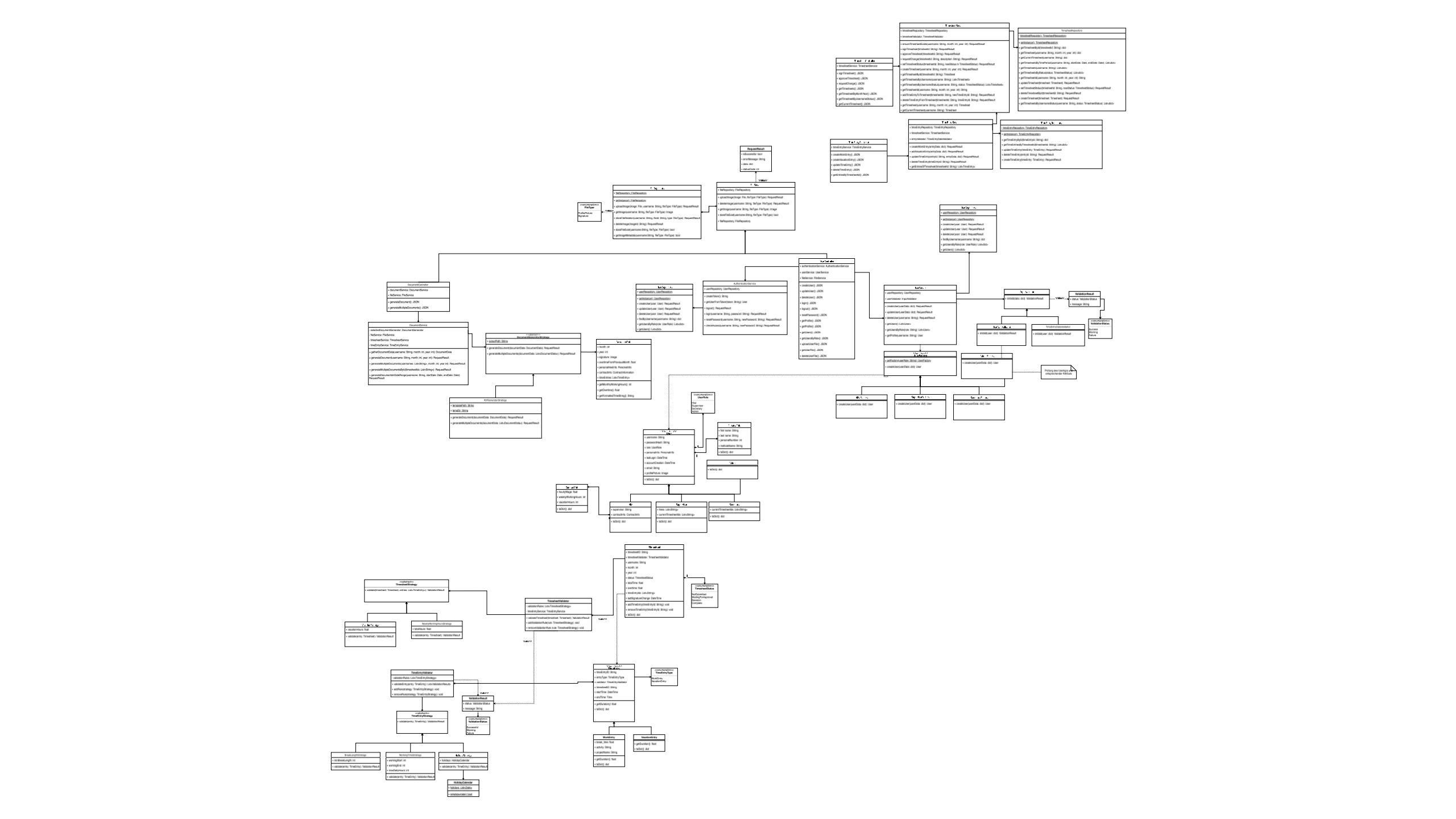
Sequenzdiagramm



Erstellung eines PDF-Dokuments

Sequenzdiagramm





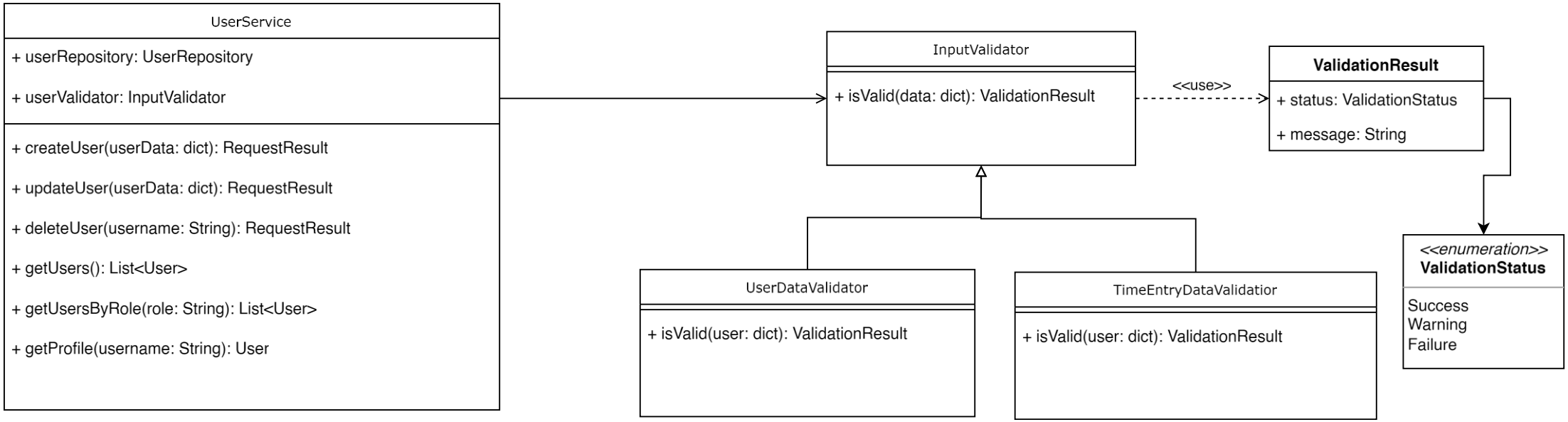
Fragen?

Team

Dominik Pollok
Phil Gengenbach
Alina Petri
José Ayala
Johann Kohl

Input Validierung

Fragen



Timesheet MVCS

Fragen