HOCHSCHULE FURTWANGEN UNIVERSITY | HFU

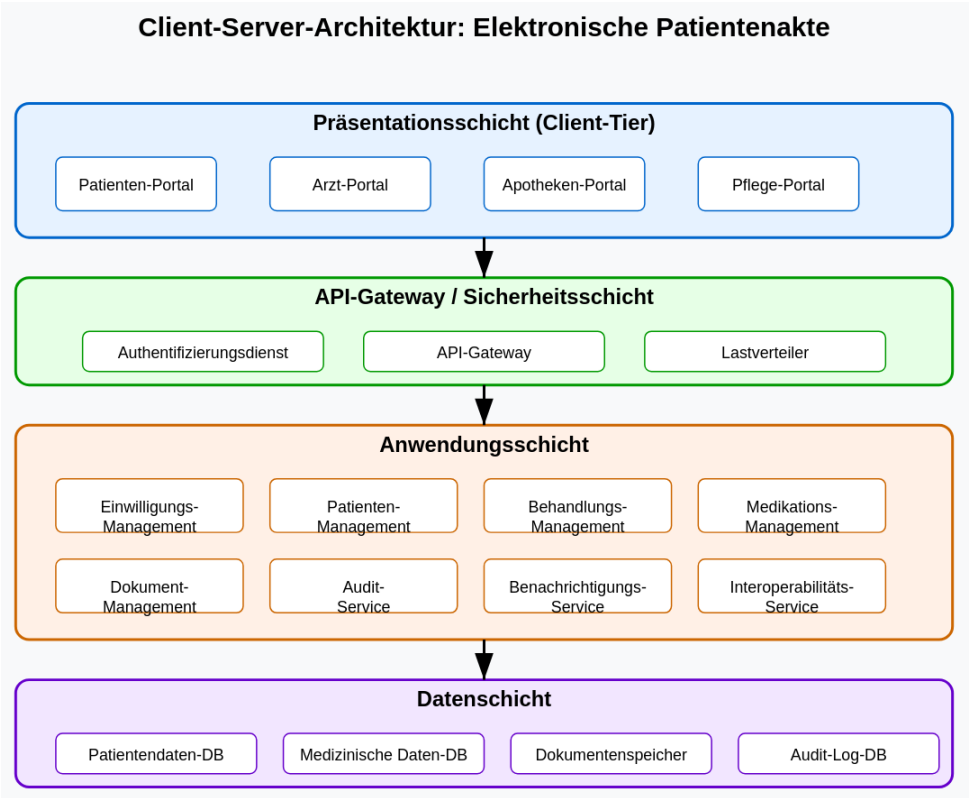
Blatt 7

Software Engineering 2

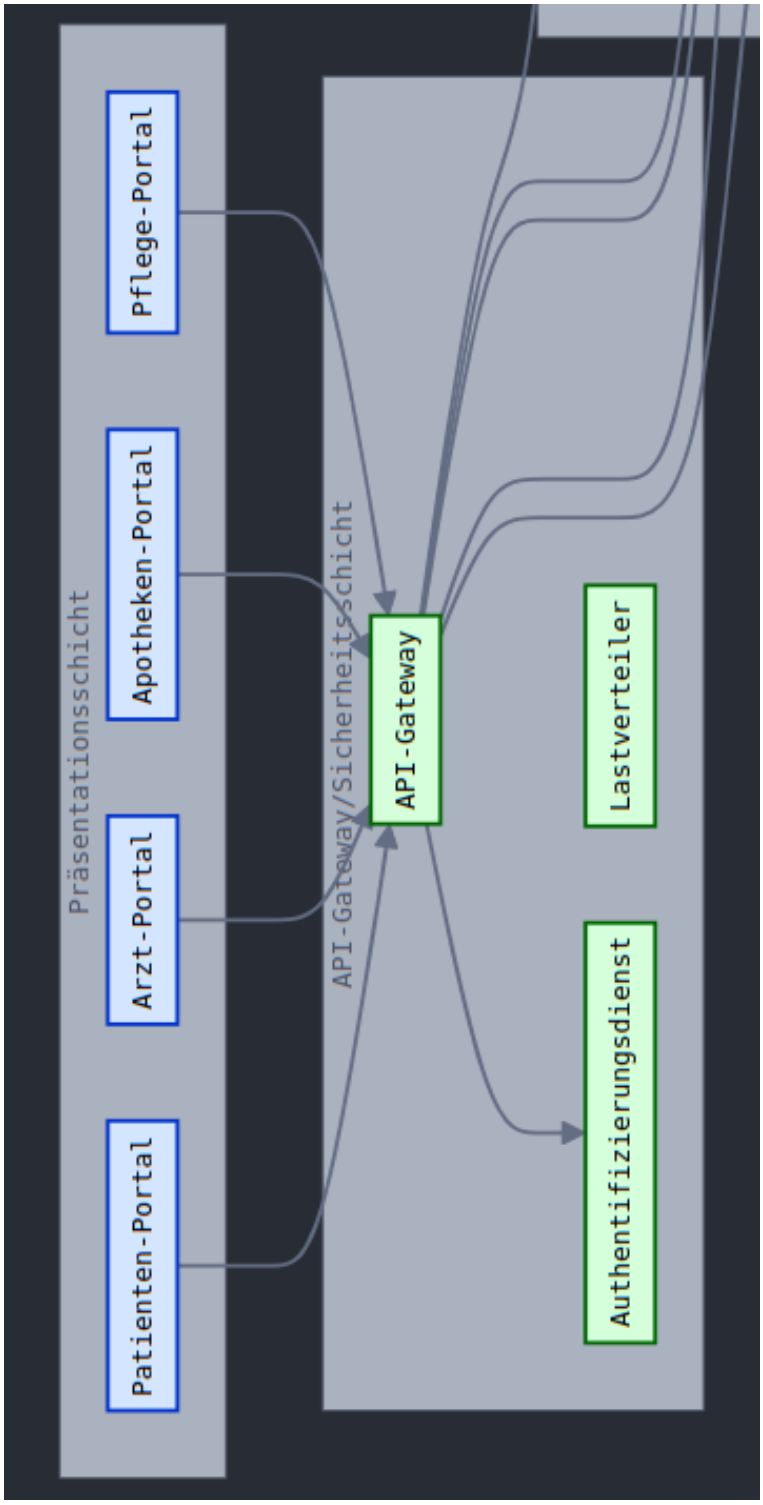
Luis Staudt

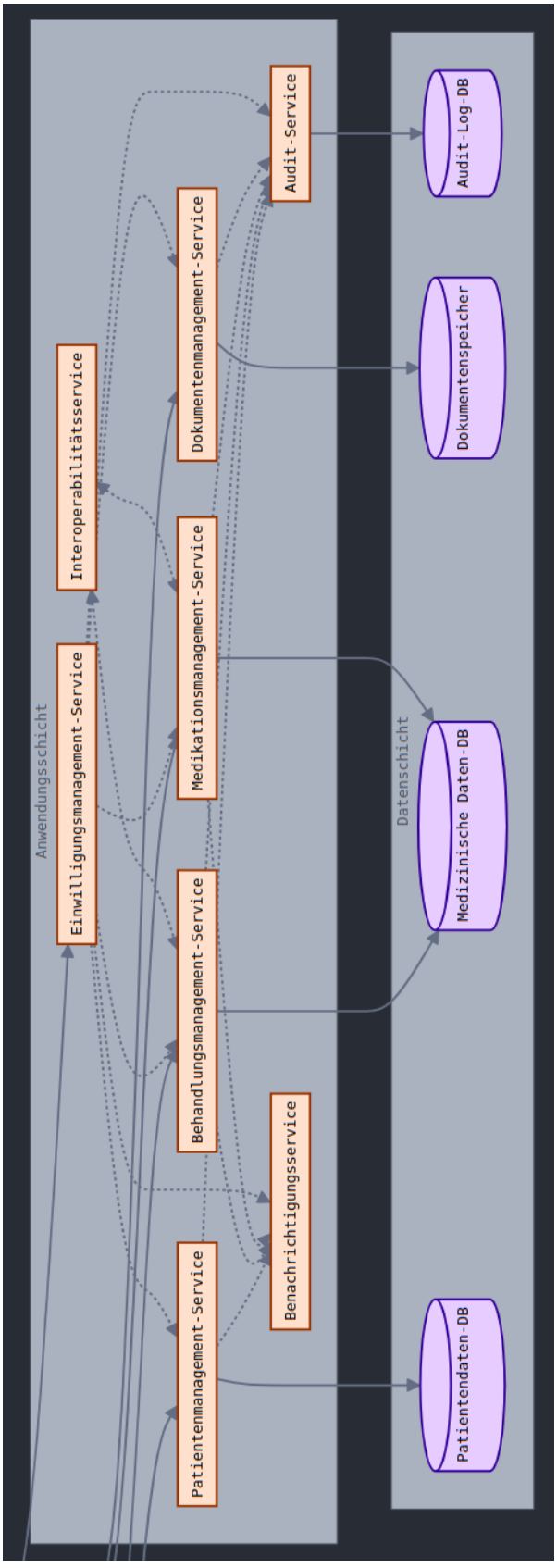
Aufgabe 1

Server-Client Architektur

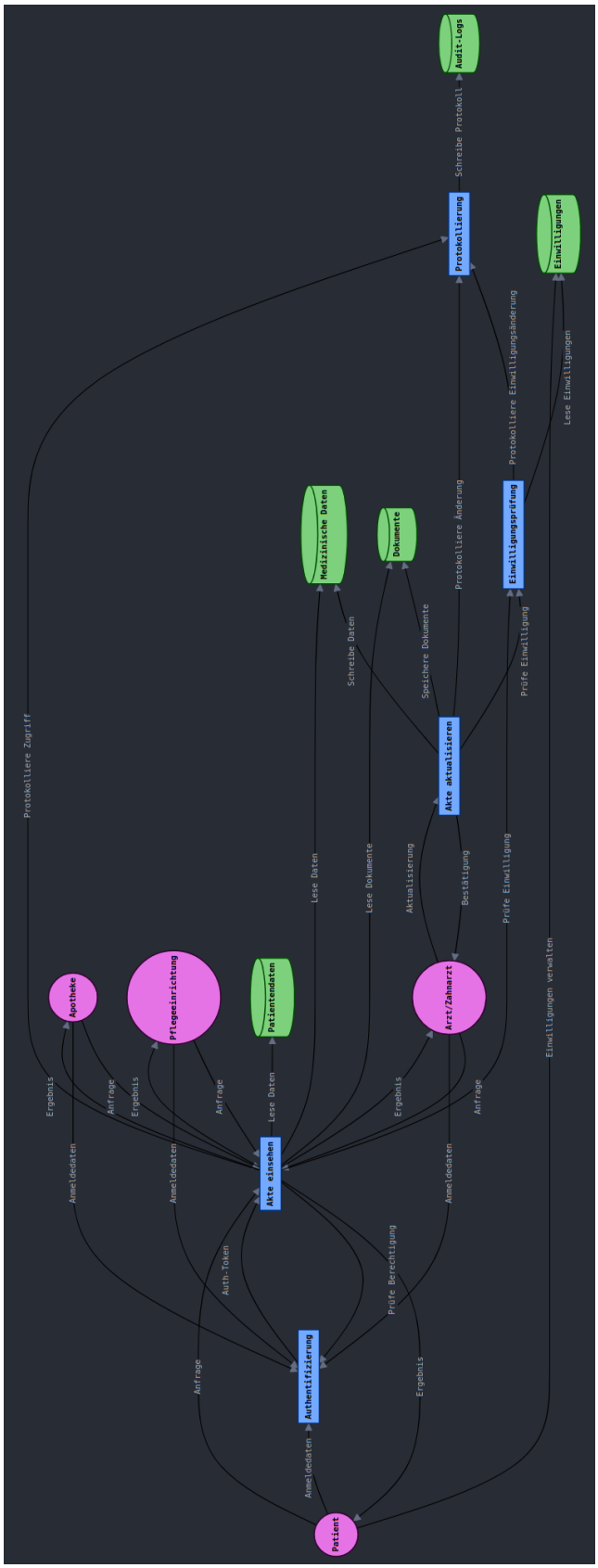


Logische Sicht

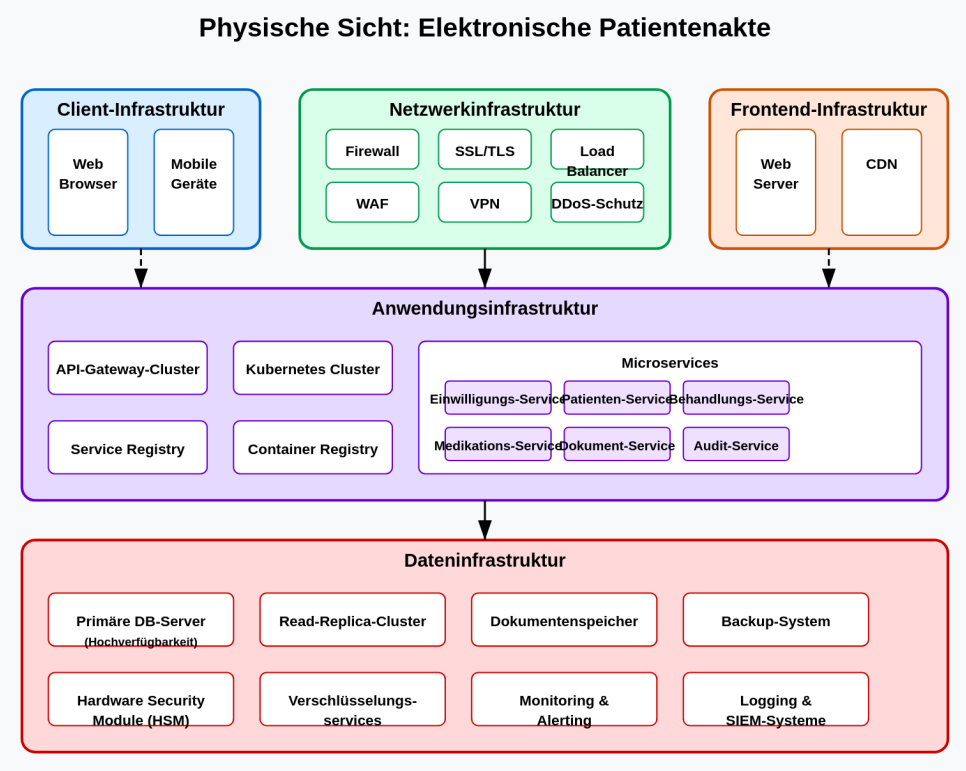



Anwendungsschicht
Datenschicht
Audit-Service
Dokumentenmanagement-Service
Interoperabilitätsservice
Medikationsmanagement-Service
Einwilligungsmanagement-Service
Behandlungsmanagement-Service
Benachrichtigungsservice
Patientenmanagement-Service
Audit-Log-DB
Dokumentenspeicher
Medizinische Daten-DB
Patientendaten-DB

Datenflussdiagramm

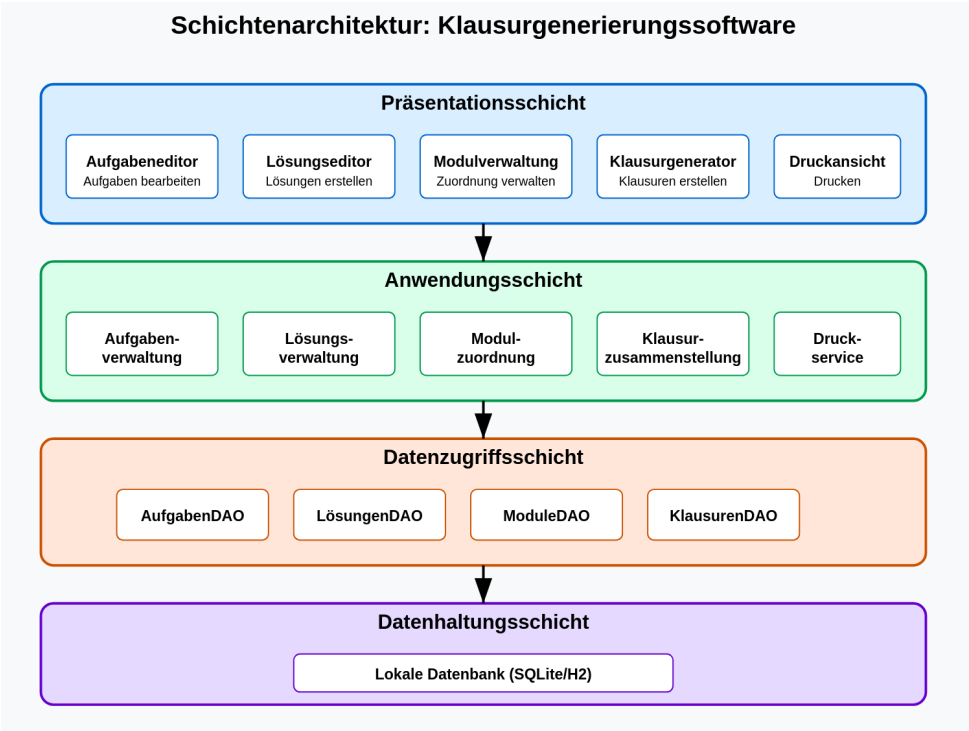


Physische Sicht

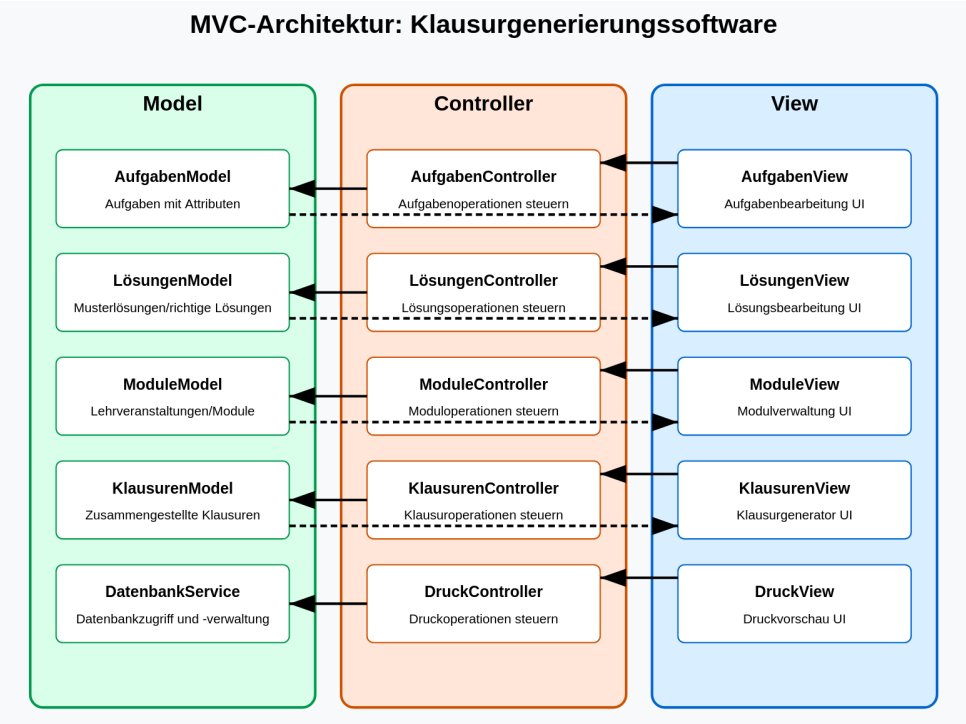


Aufgabe 2

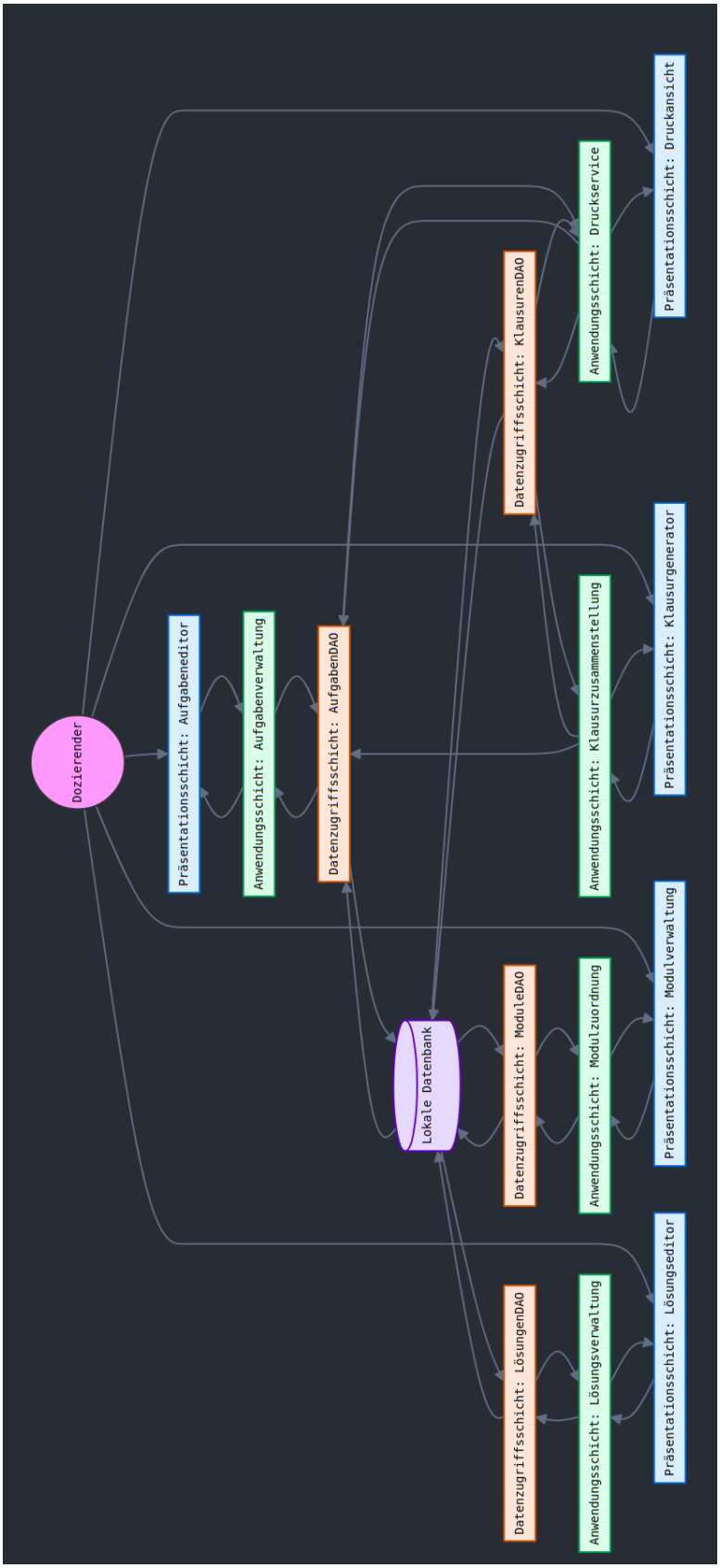
Schichtenarchitektur



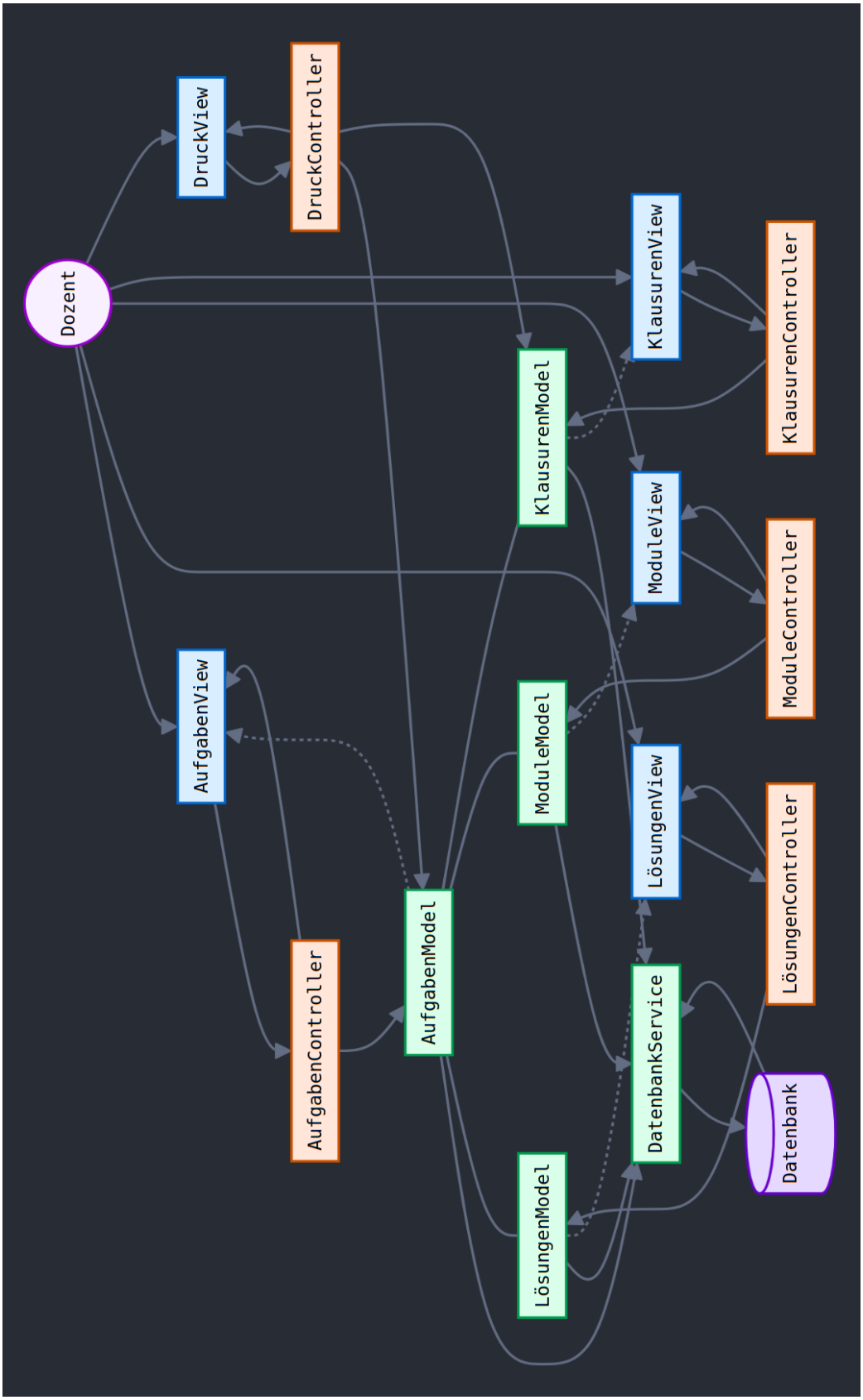
MVC-Architektur



Datenflussdiagramm - Schichtenarchitektur



Datenflussdiagramm - MVC-Architektur



Detaillierter Vergleich der Architekturen

Kriterium	Schichtenarchitektur	MVC-Architektur
Kontrollfluss Wie wird die Steuerung zwischen den Komponenten verteilt?	Hierarchisch, streng gerichtet von oben nach unten Jede Schicht dient der darüberliegenden	Zyklisch, dreieckig Controller koordiniert zwischen View und Model
Abhängigkeiten Wie stark sind die Komponenten voneinander abhängig?	Unidirektional, vorhersehbar Jede Schicht hängt nur von der direkt darunter liegenden Schicht ab	Bidirektional, komplexer View kann Model beobachten Controller hängt von View und Model ab
Modularität Austauschbarkeit, Erweiterbarkeit	Gut für horizontale Modularität Schichten können ausgetauscht werden	Besser für UI-bezogene Modularität Views können leicht geändert werden
Testbarkeit Wie gut lassen sich die einzelnen Komponenten testen?	Gut für Unit-Tests der Geschäftslogik Mocks für untere Schichten nötig UI-Tests schwieriger isolierbar	Sehr gut für isolierte Tests Model kann ohne UI getestet werden Controller separat testbar
Wartbarkeit Wie einfach ist es, das System zu warten und zu ändern?	Gut für große Änderungen in einer Schicht Funktionen gut kategorisierbar Klare Verantwortlichkeiten je Schicht	Sehr gut für UI-Änderungen Komplexere Struktur bei großen Anwendungen Weniger Doppelcode bei Features

Tabelle 1: Vergleich zwischen Schichtenarchitektur und MVC-Architektur

Aufgabe 3

Fahrzeug-Diagnosesystem in einer modernen Autowerkstatt

Beschreibung

Zentrale Datenhaltung

Fahrzeugdaten aus verschiedensten Quellen (Bordsensoren, Fehlerspeicher, historische Reparaturdaten, Herstellerdatenbanken) werden in einer zentralen Struktur gespeichert.

Unabhängige Analysekomponenten

Verschiedene Diagnosemodule können unabhängig voneinander auf dieselben Daten zugreifen.

Ereignis- und zustandsbasierte Verarbeitung

Das System kann sowohl auf neue eingehende Daten (wie bei einer traditionellen Datenbank) als auch auf bestimmte Datenzustände (wie bei einem Blackboard) reagieren.

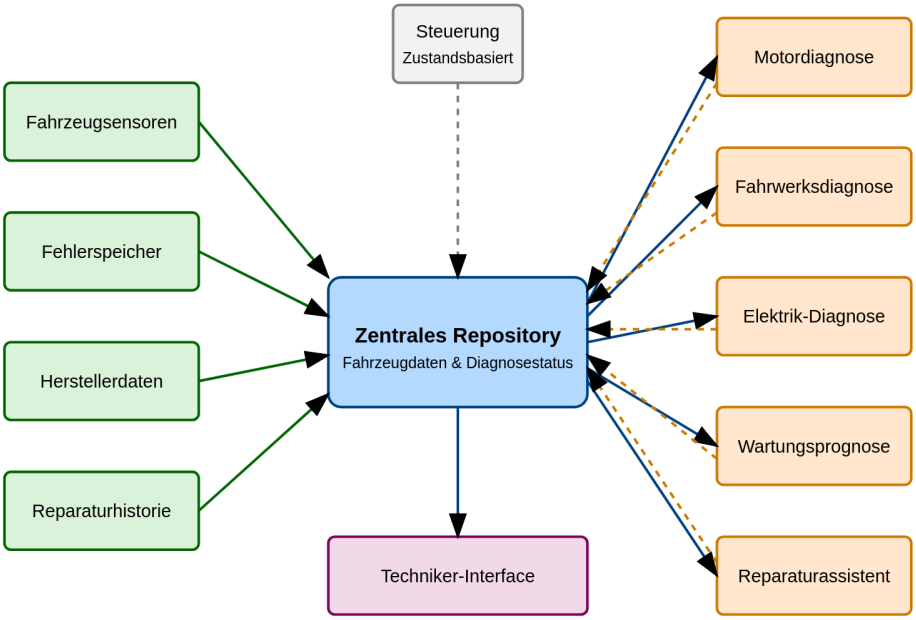
Erweiterbarkeit

Neue Diagnosemodule können ohne Änderung bestehender Komponenten hinzugefügt werden.

Integration verschiedener Expertisen

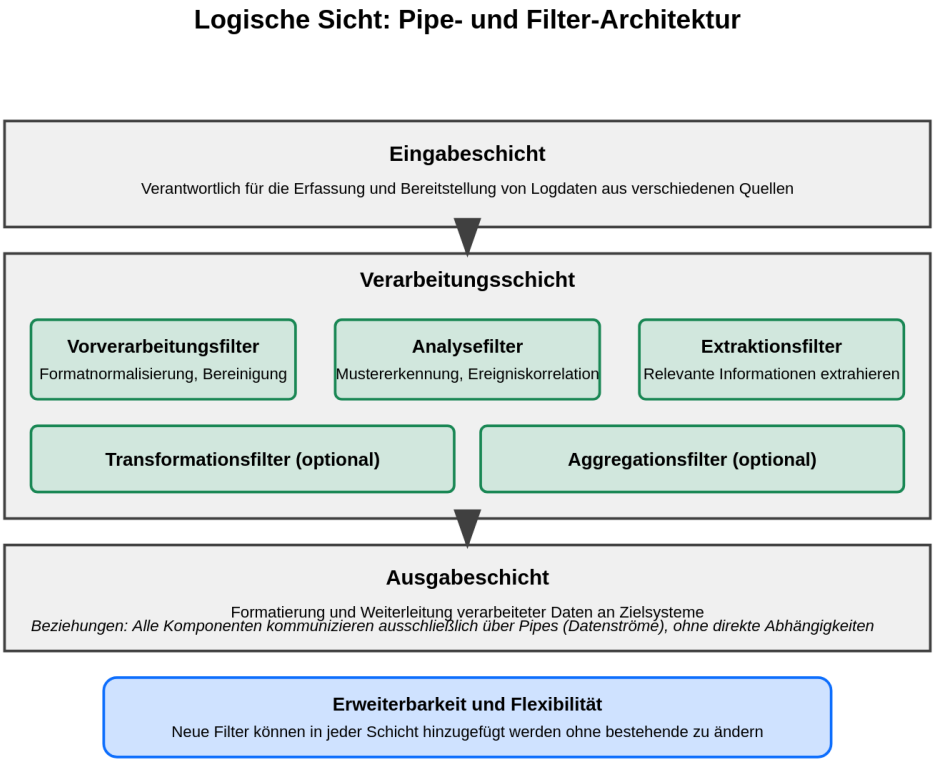
Ähnlich wie bei Blackboard-Systemen können verschiedene “Knowledge Sources“ (Fahrwerksdiagnose, Motordiagnose, elektrische Systeme) unabhängig arbeite

Repository-Architektur



Aufgabe 4

Pipe and Filter Architektur



Datenflussdiagramm - Pipe and Filter Architektur