

Blatt 5

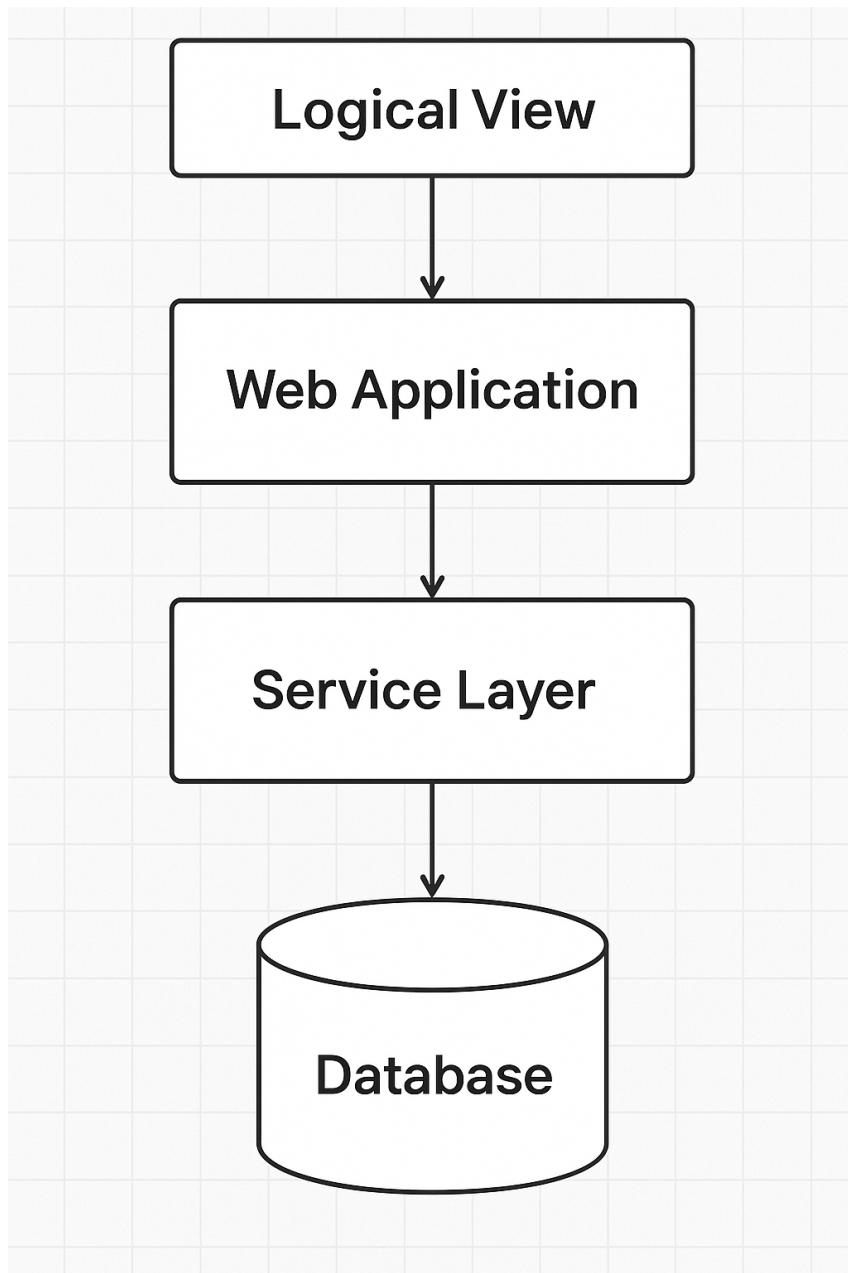
Software Engineering 2

Luis Staudt

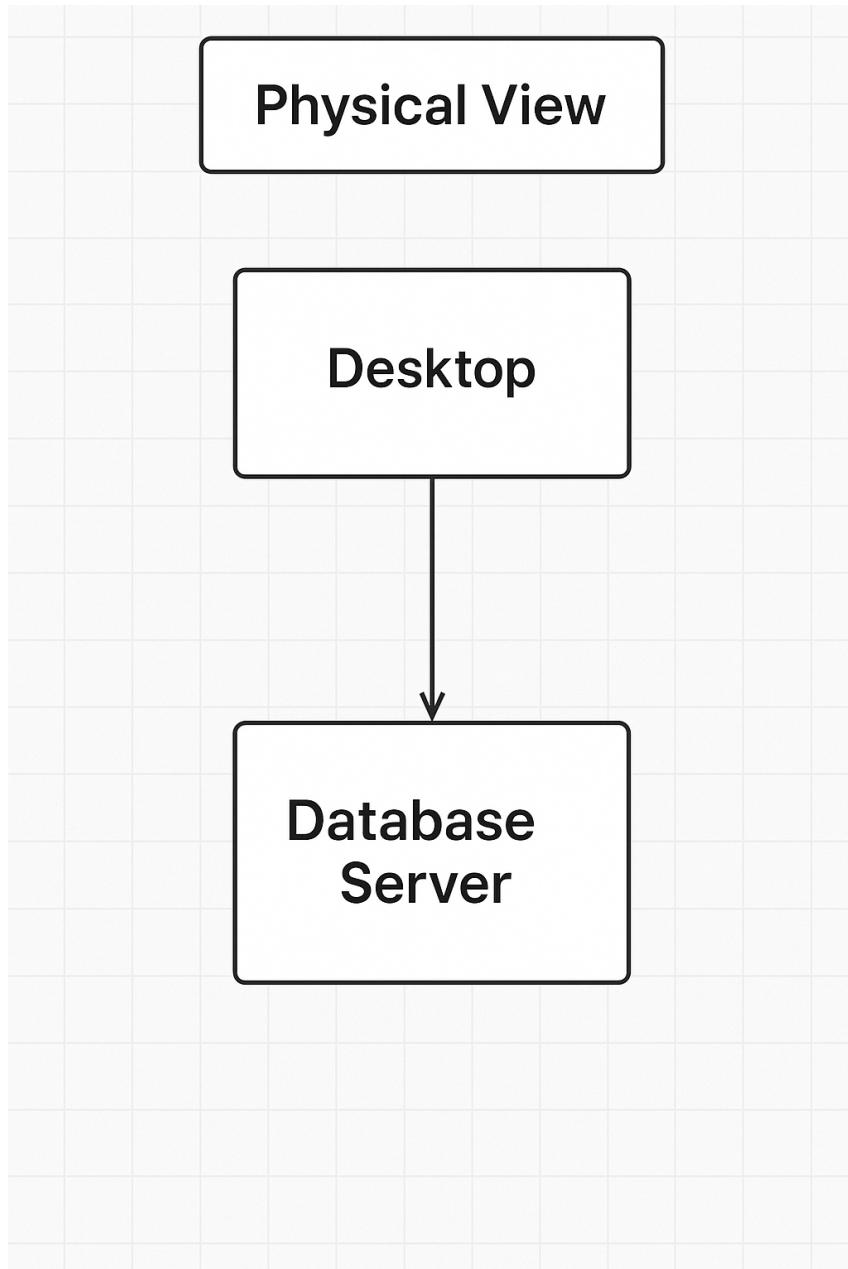
Aufgabe 1

Fallstudie 1: Software Elektronische Patientenakte

Logische Sicht

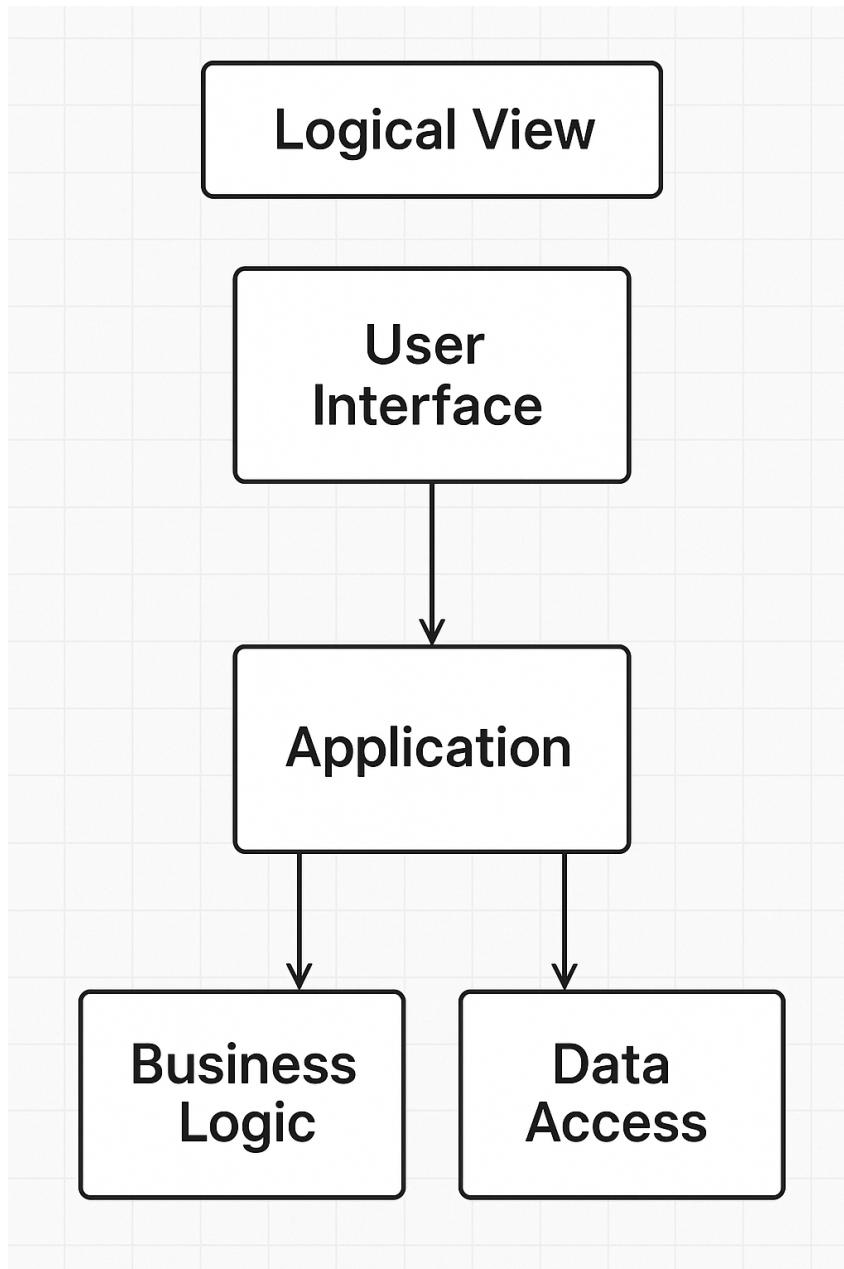


Physische Sicht

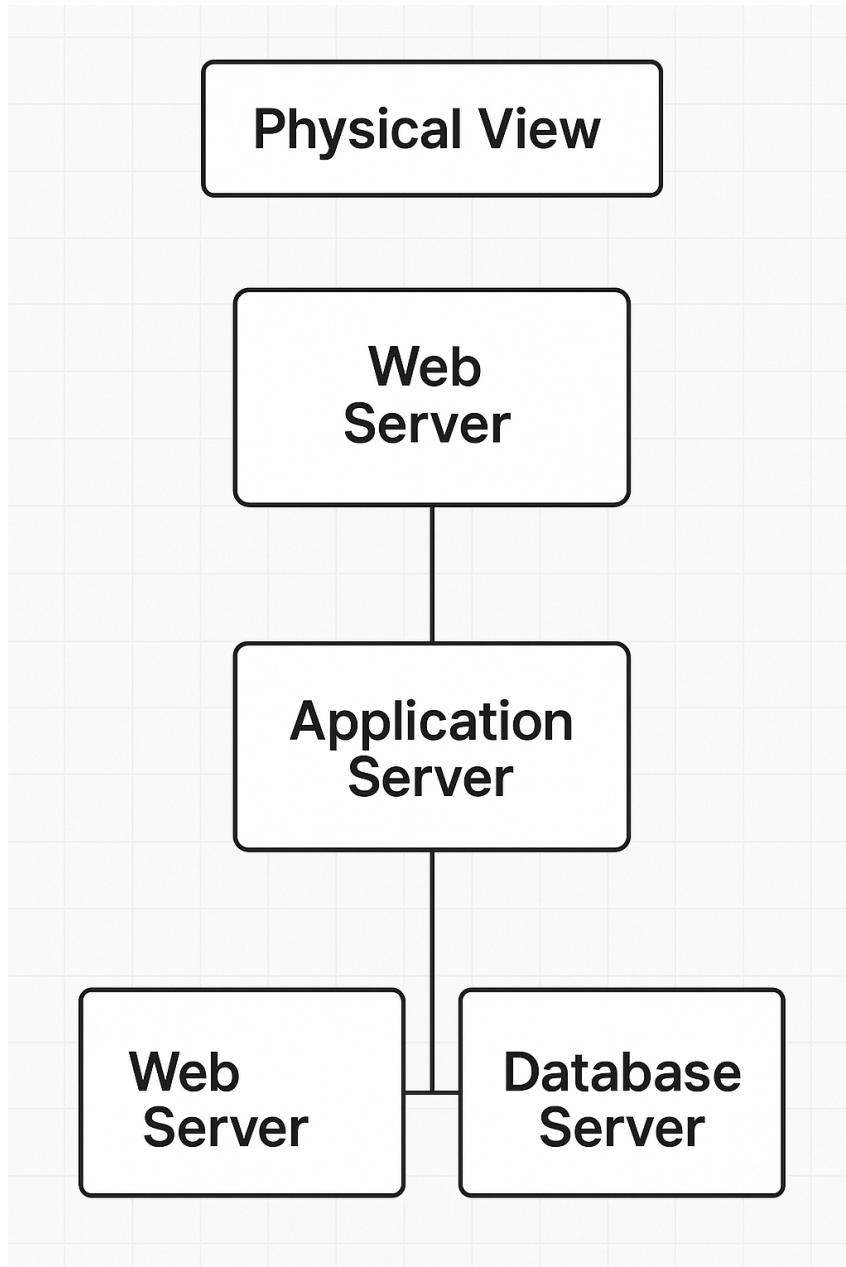


Fallstudie 2: Software zur Generierung von Klausuren

Logische Sicht



Physische Sicht



Aufgabe 2

Die wichtigsten Architekturaspekte für ein Softwareprodukt für Finanzunternehmen wären meiner Meinung nach die Softwarekompatibilität und die nichtfunktionalen Produkteigenschaften. Die Softwarekompatibilität ist entscheidend, da Finanzunternehmen oft komplexe Systemlandschaften mit Legacy-Systemen haben, die nahtlos integriert werden müssen, um Datenflüsse ohne Verluste zu gewährleisten. Die nichtfunktionalen Produkteigenschaften wie Sicherheit, Compliance und Zuverlässigkeit sind im Finanzsektor besonders wichtig, da sie strengen regulatorischen Anforderungen unterliegen und mit sensiblen Kundendaten arbeiten. Ein Produkt, das diese beiden Aspekte priorisiert, bietet Finanzunternehmen nicht nur die technische Integration, sondern auch die Gewissheit, dass es die hohen Branchenstandards für Datensicherheit und Betriebsstabilität erfüllt, was für die Akzeptanz und den Erfolg im Markt ausschlaggebend ist.

Aufgabe 3

Zentralisiertes Sicherheitsmodell

Vorteile

- Einfachere Verwaltung und Überwachung, da alle Sicherheitskontrollen an einem Ort implementiert werden.
- Konsistente Durchsetzung von Sicherheitsrichtlinien und -standards.
- Geringere Kosten für Sicherheitsinfrastruktur und Personal.
- Vereinfachte Compliance-Nachweise und Audits.
- Klare Verantwortlichkeiten und Zuständigkeiten.

Nachteile

- Single Point of Failure – bei Kompromittierung sind alle Daten gefährdet.
- Höheres Schadensausmaß bei erfolgreichen Angriffen.

- Kann zu Leistungsengpässen führen, besonders bei geografisch verteilten Nutzern.
- Begrenzte Skalierbarkeit bei wachsenden Datenmengen.
- Potentiell höhere Latenz bei Datenzugriffen.

Verteiltes Sicherheitsmodell

Vorteile

- Höhere Ausfallsicherheit – kein einzelner Fehlerpunkt.
- Größere Skalierbarkeit für wachsende Datenmengen.
- Bessere Performance durch lokale Datenzugriffe.
- Geringeres Risiko eines vollständigen Datenverlusts.
- Kann besser an lokale rechtliche Anforderungen angepasst werden.

Nachteile

- Komplexere Verwaltung und Überwachung mehrerer Sicherheitssysteme.
- Schwierigere Durchsetzung einheitlicher Sicherheitsrichtlinien.
- Höhere Kosten für verteilte Sicherheitsinfrastruktur.
- Kompliziertere Compliance-Nachweise und Audits.
- Größere Angriffsfläche durch mehrere potenzielle Einstiegspunkte.

Aufgabe 4

Fallstudie 1: Software Elektronische Patientenakte

Technologieaspekte	Architekturentscheidung Fallstudie Elektronische Patientenakte mit Begründung
Datenbank	Relationale SQL-Datenbank. Eine relationale Datenbank gewährleistet durch das ACID-Prinzip die Integrität der medizinischen Daten und bildet komplexe Beziehungen zwischen Patienten, Behandlungen und Zugriffsberechtigungen optimal ab. Die ausgereiften Sicherheitsmechanismen und Auditmöglichkeiten erfüllen die regulatorischen Anforderungen im Gesundheitsbereich.
Plattform	Webplattform mit nativer mobiler App. Die Webplattform ermöglicht medizinischem Personal einen vollumfänglichen Zugriff an stationären Arbeitsplätzen, während die native App durch optimierte Bedienung und biometrische Authentifizierung sicheren mobilen Zugriff bietet. Diese Kombination gewährleistet höchste Sicherheitsstandards bei gleichzeitiger universeller Verfügbarkeit für alle Nutzergruppen.
Server	Hybridlösung mit dedizierten Servern und spezialisierter Healthcare-Cloud. Dedizierte Server in deutschen Rechenzentren sichern sensible Kernfunktionen und gewährleisten die Einhaltung strengster Datenschutzanforderungen. Die ergänzende Healthcare-Cloud (z.B. Azure oder AWS mit Gesundheitszertifizierungen) bietet Skalierbarkeit und Hochverfügbarkeit bei gleichzeitiger Einhaltung von Standards wie HIPAA.
Open Source	Selektiver Einsatz geprüfter Open-Source-Komponenten. Für die Patientenakte sollten nur sicherheitsgeprüfte Open-Source-Komponenten wie OpenEHR oder HAPI FHIR für Standardschnittstellen eingesetzt werden. Sicherheitskritische Kernfunktionen sollten durch proprietäre, speziell entwickelte und zertifizierte Lösungen abgedeckt werden, um höchste Sicherheitsstandards zu garantieren.

Fallstudie 2: Software zur Generierung von Klausuren

Technologieaspekte	Architekturentscheidung Fallstudie Klausurgenerierung mit Begründung
Datenbank	Relationale SQL-Datenbank. Die klar strukturierten Aufgaben mit definierten Beziehungen zu Lehrveranstaltungen, Modulen und Musterlösungen passen optimal zum relationalen Modell. SQLite eignet sich besonders gut, da es als eingebettete Datenbank keine separate Serverinstallation erfordert und die lokale Desktop-Installation erleichtert.
Plattform	Desktop-Anwendung mit optionaler Web-Oberfläche. Eine plattformübergreifende Desktop-Anwendung (z.B. mit Electron) erfüllt die explizite Anforderung der lokalen Installation und unterstützt den typischen Arbeitsablauf von Dozierenden. Eine optionale Web-Oberfläche könnte für einfache Aufgaben wie das Durchsuchen der Aufgabensammlung ergänzend angeboten werden.
Server	Keine Cloud-Lösung, nur lokal. Die Software sollte vollständig offline funktionieren, da eine lokale Desktop-Installation gefordert ist und sensible Prüfungsdaten lokal geschützt werden müssen. Für erweiterte Funktionen wie die Zusammenarbeit zwischen Dozierenden könnte optional ein lokaler Server im Hochschulnetzwerk ausreichen.
Open Source	Umfangreicher Einsatz von Open-Source-Komponenten. Bibliotheken für PDF-Generierung (wie PDFKit), Rich-Text-Editoren (wie TinyMCE) und UI-Frameworks beschleunigen die Entwicklung und verbessern die Wartbarkeit. Da es sich nicht um ein hochsensibles System handelt, ist der umfassende Einsatz von Open-Source-Komponenten mit geringeren Sicherheitsbedenken verbunden.