

Abschlussprüfung Winter 2026 Fachinformatiker für Anwendungsentwicklung Dokumentation zur betrieblichen Projektarbeit

Entwicklung eine „Interaktive UML-Lern-App mit Quiz, Notizen, Lernpfaden (Levels) und Kursverwaltung“

25. September 2025

Marburg, 2025

Eingereicht von:

Josiane Kanouo Maneyo

Umschülerin zur Fachinformatikerin für Anwendungsentwicklung



IAD GmbH Marburg

Neue Kasseler Straße 62e

35039 Marburg

Betreuer: Marcus Brauer

Abgabedatum:

Inhaltsverzeichnis

Abbildungsverzeichnis III

Tabellenverzeichnis IV

Listings V

Abkürzungsverzeichnis VI

Inhalt

[1 Einleitung 1](#_Toc210643342)

[1.1 Projektumfeld 1](#_Toc210643343)

[1.2 Projektziel 1](#_Toc210643344)

[1.3 Projektbegründung 2](#_Toc210643345)

[1.4 Projektschnittstellen 2](#_Toc210643346)

[1.5 Projektabgrenzung 2](#_Toc210643347)

[2 Projektplanung 2](#_Toc210643348)

[2.1 Projektphasen 2](#_Toc210643349)

[2.2 Abweichungen vom Antrag 2](#_Toc210643350)

[2.3 Ressourcenplanung 2](#_Toc210643351)

[2.4 Vorgehensmodell 3](#_Toc210643352)

# Einleitung

Dieses Kapitel beschreibt die Ausgangssituation, die Motivation und die Zielsetzung des Projekts. Es erläutert, welches Problem durch die Anwendung gelöst werden soll, welchen Nutzen sie bietet und wie das Projekt in den Gesamtkontext der Ausbildung eingebettet ist. Darüber hinaus werden die Projektumgebung, die Beteiligten und die Rahmenbedingungen vorgestellt, um einen Überblick über den Hintergrund des Projekts zu geben.

## Projektumfeld

Das Projekt wird im Rahmen der Umschulung zur Fachinformatikerin für Anwendungsentwicklung an der IAD Bildungszentrum Marburg GmbH¹ durchgeführt. Die IAD, Teil der IAD Group mit über 50 Jahren Erfahrung, ist einer der größten privaten Bildungsanbieter in Deutschland für IT und Management. Sie bietet praxisorientierte Ausbildungen und Umschulungen an, darunter den dualen Fachinformatiker für Anwendungsentwicklung mit Fokus auf Softwareentwicklung (z. B. Kotlin, Mobile Apps) und Datenhaltung. Der Marburg-Standort (Wilhelm-Röpke-Straße 11) dient als zentraler Hub mit modernen IT-Labs und Kooperationen mit der Agentur für Arbeit. Jährlich werden ca. 8.000 Teilnehmer bundesweit qualifiziert, mit hohen Erfolgsquoten: 90 % der Absolventen finden direkt einen Job, 95 % Zufriedenheit und 98 % Bestehensrate bei Prüfungen. Die Projektarbeit erfolgt als Einzelprojekt. Die Verantwortung umfasst Analyse, Entwurf, Implementierung, Test und Dokumentation. Fachliche Betreuung erfolgt durch Marcus Brauer, Dozent an der IAD, der die technische Umsetzung unterstützt und die Einhaltung der IHK-Vorgaben überwacht. Zielgruppe der Anwendung sind Lernende und Lehrkräfte, die eine digitale Plattform zur Kursauswahl, Bearbeitung von Quizfragen, Verwaltung von Fehlerlisten und Erstellung persönlicher Notizen nutzen können. Die Anwendung adressiert den Übergang von statischen Papierübungen zu interaktiven Tools, die ortsunabhängiges Training mit direktem Feedback und zentraler Auswertung ermöglichen. Neben der bestehenden Lernumgebung wird mit der App eine moderne, mobile Erweiterung geschaffen, die neuen Lernenden den Einstieg durch ein interaktives Tutorial erleichtert und Rückmeldungen direkt in die Weiterentwicklung einfließen lässt.

## Projektziel

Ziel des Projekts ist die Entwicklung einer interaktiven Android-Anwendung², die Lernenden und Lehrkräften des IAD Bildungszentrums ein Werkzeug zum Üben, Wiederholen und Verwalten von Lerninhalten bietet. Die Anwendung ersetzt traditionelle, statische Methoden und schafft eine strukturierte, motivierende Lernumgebung für UML-Diagramme.

Die App umfasst Funktionen wie Kursauswahl, Quiz mit direktem Feedback, Fehlerlisten, persönliche Notizen, ein Levelsystem und einen Prüfungsmodus mit Timer. Administratoren erhalten einen Bereich zur Verwaltung von Kursen und Fragen. Technisch wird die Anwendung mit Kotlin³ in Android Studio umgesetzt und nutzt Firebase⁴ für Authentifizierung, Datenspeicherung in Firestore und Medienverwaltung in Storage. Sicherheitsregeln und Offline-Funktionen gewährleisten Stabilität auch ohne Internetverbindung, wodurch Lernende von transparenter Fortschrittskontrolle profitieren und Lehrkräfte zentrale Inhaltsverwaltung erhalten. Neben den Kernfunktionen wird die App um zwei wesentliche Erweiterungen ergänzt: ein Onboarding-Tutorial, das neuen Nutzenden in wenigen Schritten die wichtigsten Funktionen erklärt, und eine In-App-Feedback-Funktion, mit der Rückmeldungen direkt an die Entwicklerin gesendet werden können. Dadurch wird der Einstieg erleichtert, die Benutzerfreundlichkeit erhöht und die kontinuierliche Verbesserung der Anwendung unterstützt.

## Projektbegründung

Der Unterricht an der IAD basiert derzeit überwiegend auf Papierübungen und Word-Dokumenten für UML-Diagramme. Diese Methode ist unflexibel, bietet kein direktes Feedback und erlaubt keine zentrale Verwaltung oder Auswertung. Lehrkräfte haben dadurch keinen einfachen Überblick über den Lernstand, und Ergebnisse lassen sich nicht systematisch dokumentieren oder vergleichen. Die geplante Anwendung digitalisiert diese Prozesse, erhöht Transparenz und steigert die Motivation. Lernende können gezielt Fehler wiederholen, ihren Fortschritt verfolgen und individuelle Notizen anlegen.

Eine Eigenentwicklung wurde gewählt, da bestehende Lernplattformen wie Moodle oder Quizlet keine Kombination aus Fehlerwiederholung, Notizen und Levelsystem bieten. Die Lösung ist somit speziell auf die Bedürfnisse der Umschulungsteilnehmenden zugeschnitten und unterstützt die Digitalisierung des Unterrichts, wodurch der Lernprozess effizienter, praxisnäher und motivierender gestaltet wird als mit herkömmlichen Papierübungen. Durch das Onboarding-Tutorial wird der Einstieg in die Anwendung auch für technisch weniger erfahrene Lernende deutlich einfacher. Die integrierte Feedback-Funktion ermöglicht es, Verbesserungsvorschläge und Fehler direkt zu melden, was eine nachhaltige Weiterentwicklung und Qualitätssicherung unterstützten.

## Projektschnittstellen

Die Anwendung kommuniziert ausschließlich mit Firebase-Diensten (Authentication, Cloud Firestore, Storage, Security Rules) über sichere HTTPS-Verbindungen. Offline-Nutzung wird durch lokale Datenspeicherung ermöglicht. Beteiligte Personen sind Marcus Brauer als Auftraggeber, Josiane Kanouo Maneyo als Entwicklerin und Testpersonen aus der IAD. Externe Systeme oder APIs werden nicht integriert. Zusätzlich wird die Feedback-Sammlung in Firebase zur Speicherung eingehender Rückmeldungen genutzt.

## 1.5 Projektabgrenzung

Das Projekt beschränkt sich auf die Entwicklung der Android-Anwendung mit den im Antrag festgelegten Kernfunktionen. Ausgeschlossen sind eine Webversion, KI-Funktionen, komplexe Statistiken oder die Veröffentlichung im Google Play Store. Das Projekt endet mit einem funktionsfähigen Prototyp, der die geforderten Anforderungen erfüllt. Es wird keine UML-Diagramm-Erstellung integriert; der Fokus liegt auf quizbasiertem Lernen und Verwaltung.

# Projektplanung

Im folgenden Kapitel wird der zeitliche und organisatorische Ablauf des Projekts dargestellt. Hierzu werden die einzelnen Projektphasen beschrieben, der Zeitaufwand dokumentiert und die eingesetzten Methoden und Werkzeuge erläutert.

## 2.1 Projektphasen

Das Projekt umfasst einen Zeitrahmen von 80 Stunden und gliedert sich in acht aufeinanderfolgende Phasen: Analyse, Konzept, Entwurf, Implementierung, Test und Dokumentation. Die Analysephase diente der Anforderungsermittlung, gefolgt von Entwurf und Programmierung sowie Tests zur Sicherstellung der Funktionalität. Die Soll-Phase und Implementierung umfassen zusätzlich die Entwicklung eines Onboarding-Tutorials sowie einer In-App-Feedback-Funktion, um die Benutzerfreundlichkeit zu erhöhen. Eine detaillierte Übersicht der Stundenverteilung ist im Anhang A1 enthalten.

## 2.2 Abweichungen vom Antrag

Gegenüber dem ursprünglichen Antrag wurde der Funktionsumfang leicht erweitert.  
Ergänzt wurden ein Onboarding-Tutorial und eine In-App-Feedback-Funktion, um das Nutzererlebnis zu verbessern. Der Zeitrahmen von 80 Stunden blieb unverändert; der Mehraufwand wurde durch optimierte Arbeitsschritte ausgeglichen.

## 2.3 Ressourcenplanung

Zur Umsetzung stehen ein Windows-Laptop, ein Android-Smartphone sowie die Software Android Studio, Kotlin und Firebase zur Verfügung. Versionierung und Datensicherung erfolgen über GitHub. Zusätzliche Tools wie Draw.io und Microsoft Word unterstützen die Erstellung von Diagrammen und der Dokumentation.

## 2.4 Vorgehensmodell

Es wurde ein phasenorientiertes Vorgehensmodell mit agilen Elementen angewandt. Die Struktur orientiert sich am Wasserfallmodell⁵, ergänzt durch kurze Feedbackzyklen während der Implementierung. Dies ermöglichte eine flexible Reaktion auf Designänderungen und Testergebnisse bei gleichzeitiger Einhaltung eines klaren Ablaufs. Während der Implementierung wurden nach jeder abgeschlossenen Funktion (Login, Quiz, Feedback, Tutorial) kurze Test- und Feedback-Zyklen durchgeführt, um Stabilität und Nutzerfreundlichkeit sicherzustellen.

Fußnoten

¹ IAD Bildungszentrum Marburg GmbH: [www.iad.de](http://www.iad.de).

² Android-Anwendung: Basierend auf Kotlin für mobile Entwicklung, siehe developer.android.com.

³ Kotlin: Moderne Programmiersprache für Android, siehe kotlinlang.org.

⁴ Firebase: Cloud-Plattform von Google für Authentifizierung und Daten, siehe firebase.google.com.

⁵ Wasserfallmodell: Traditionelles Modell der Softwareentwicklung.

# 3 Analysephase

In der Analysephase werden die Anforderungen an die zu entwickelnde Anwendung systematisch erfasst und präzisiert. Dabei werden sowohl funktionale als auch nicht-funktionale Anforderungen beschrieben, um sicherzustellen, dass alle relevanten Aspekte berücksichtigt werden. Zusätzlich wird das bestehende Problem analysiert, und es werden die Ziele sowie der Nutzen der geplanten Lösung klar abgegrenzt. Diese Phase bildet die inhaltliche Grundlage für den späteren Entwurf und die technische Umsetzung.

## 3.1 Ist-Analyse

Der Unterricht an der IAD Bildungszentrum Marburg GmbH basiert derzeit auf klassischen Lehrmethoden, bei denen die Lernenden UML-Diagramme in Papier- oder PDF-Form üben. Eine unmittelbare Rückmeldung zu den Ergebnissen erfolgt nicht; Korrekturen werden von Lehrkräften manuell durchgeführt und können nicht zentral dokumentiert werden. Auch der individuelle Lernfortschritt ist nicht transparent einsehbar. Vorhandene Tools wie draw.io oder PlantUML² ermöglichen zwar das Zeichnen von Diagrammen, unterstützen jedoch keinen interaktiven Lernprozess mit Feedback-, Notiz- oder Fortschrittsfunktionen. Das Fehlen einer digitalen, benutzerfreundlichen Lösung führt zu hohem Zeitaufwand und geringerer Motivation bei den Lernenden. Eine moderne, mobile App-Lösung kann diese Probleme beseitigen, indem sie Lernfortschritte automatisch erfasst, Fehler dokumentiert und Lerninhalte flexibel zur Verfügung stellt.

## 3.2 Soll-Analyse

Ziel dieses Projekts ist die Entwicklung einer interaktiven Android-Anwendung³, die den Lernprozess für UML-Diagramme digitalisiert und strukturiert. Die App soll Lernenden ermöglichen, eigenständig zu üben, sofort Feedback zu erhalten und ihre Fehler nachzuvollziehen. Lehrkräfte erhalten gleichzeitig ein einfaches Werkzeug zur Pflege von Inhalten und Fragen. Im Rahmen der Projektweiterentwicklung wurde der Funktionsumfang gezielt erweitert. Neben den Kernfunktionen wie Login, Kurswahl, Quiz, Fehlerliste, Notizen und Levelsystem enthält die App nun ein kurzes Onboarding-Tutorial, das neue Nutzende in wenigen Schritten durch die Hauptbereiche führt, sowie eine In-App-Feedback-Funktion, über die Anregungen oder Fehlermeldungen direkt an die Entwicklerin übermittelt werden können. Beide Erweiterungen erhöhen die Benutzerfreundlichkeit und unterstützen eine kontinuierliche Verbesserung der Anwendung. Die vollständige Funktionsübersicht befindet sich im Anhang A2.

## 3.3 Wirtschaftlichkeitsbetrachtung

Das Projekt wird im Rahmen der Umschulung zur Fachinformatikerin für Anwendungsentwicklung durchgeführt und über Leistungen nach dem Sozialgesetzbuch III (Arbeitslosengeld I) finanziert⁴. Somit entstehen für den Bildungsträger keine zusätzlichen Kosten. Der Hauptnutzen liegt in der Effizienzsteigerung des Lernprozesses. Die App reduziert Papierverbrauch, spart Korrekturzeit und erhöht die Motivation der Lernenden durch direktes Feedback und sichtbare Fortschritte. Gleichzeitig ermöglicht die Feedback-Funktion eine laufende Optimierung der Inhalte und trägt somit zur Qualitätssicherung bei. Der geplante Zeitaufwand für das gesamte Projekt beträgt 80 Stunden, verteilt auf Analyse, Planung, Implementierung, Test und Dokumentation.

## 3.4 Qualitätsanforderungen

Die UML-Lern-App legt besonderen Wert auf eine hohe Benutzerfreundlichkeit und Stabilität. Die Oberfläche ist klar strukturiert und ermöglicht eine intuitive Navigation. Eine zuverlässige Offline-Funktion gewährleistet, dass bereits geladene Inhalte auch ohne Internetverbindung verfügbar sind. Hinsichtlich der Datensicherheit werden die Zugriffsrechte in Firestore so geregelt, dass jeder Nutzer nur seine eigenen Informationen sehen kann. Firebase Security Rules sichern alle Operationen ab. Der Quellcode folgt dem MVVM-Muster, um eine klare Trennung zwischen Logik und Oberfläche sowie eine einfache Erweiterbarkeit zu gewährleisten. Performance und Zuverlässigkeit werden durch lokales Caching, regelmäßige Tests und optimierte Abfragen sichergestellt. Neue Nutzende werden durch das Onboarding-Tutorial schnell eingeführt, während die Feedback-Funktion eine aktive Kommunikation zwischen Entwicklerin und Anwendern ermöglicht.

## 3.5 Anwendungsfälle

Das System unterstützt zwei Akteure: Lernende und Administratoren. Lernende können nach dem Login Kurse auswählen, Fragen beantworten und Ergebnisse direkt einsehen. Fehler werden gespeichert und können wiederholt geübt werden. Administratoren verwenden die App, um Kurse und Fragen zu pflegen und Inhalte laufend zu aktualisieren. Das Onboarding-Tutorial erleichtert den Einstieg für neue Nutzende, während die Feedback-Funktion Rückmeldungen an die Entwicklerin ermöglicht. Eine grafische Darstellung der Anwendungsfälle befindet sich im Anhang A3.

## 3.6 Verweis auf das Lastenheft

Alle funktionalen und nicht-funktionalen Anforderungen sind im Anhang A4 – Lastenheft detailliert beschrieben. Das Lastenheft definiert Muss- und Kann-Kriterien sowie Abgrenzungen des Projekts und bildet die Grundlage für die Abnahme.

¹ IAD Bildungszentrum Marburg GmbH – https://www.iad-bildung.de  
² draw.io / PlantUML – Werkzeuge zur Erstellung von UML-Diagrammen  
³ JetBrains s.r.o.: Kotlin Documentation, 2025  
⁴ Förderung nach SGB III (Arbeitsagentur Marburg), 2025

# Entwurfsphase

Dieses Kapitel beschreibt die technische und gestalterische Planung der UML-Lern-App. Hier werden die grundlegende Architektur, die Struktur der Datenbank, das Design der Benutzeroberfläche sowie der Aufbau der App erläutert. Die Entwurfsphase bildet somit den Übergang zwischen der theoretischen Planung und der praktischen Umsetzung.

## Zielplattform

Für die Umsetzung der UML-Lern-App wurde die Android-Plattform als Zielumgebung gewählt. Ausschlaggebend dafür waren die weite Verbreitung von Android-Geräten, die einfache Verfügbarkeit von Entwicklungswerkzeugen sowie die Möglichkeit, moderne Technologien wie Firebase direkt zu integrieren. Als Programmiersprache kommt Kotlin zum Einsatz, da sie von Google offiziell unterstützt wird, eine klare Syntax besitzt und sich besonders gut für mobile Anwendungen eignet. Die Datenhaltung erfolgt in der Cloud Firestore-Datenbank von Firebase, die eine dokumentenbasierte Struktur bietet und sowohl Online- als auch Offline-Betrieb ermöglicht. Diese Architektur erlaubt eine flexible, sichere und leicht erweiterbare Speicherung aller Benutzerdaten, Kurse und Quiz-Ergebnisse. Die App folgt dem Client-Server-Prinzip: Der Client (die Android-App) kommuniziert mit dem Server (Firebase-Dienste) über eine gesicherte HTTPS-Verbindung. Dadurch können Benutzer weltweit auf ihre persönlichen Daten zugreifen, ohne dass lokale Server notwendig sind. Als Hardwarebasis wurde ein Standard-Android-Gerät mit mindestens 4 GB RAM, 6-Zoll-Display und Android-Version 10 oder höher festgelegt. Damit wird sichergestellt, dass die Anwendung flüssig läuft und gleichzeitig auch auf älteren Geräten nutzbar bleibt. Die Kombination aus Kotlin, Firebase und Android Studio bietet somit eine moderne, performante und zukunftssichere Grundlage für die Umsetzung der Lern-App.

4.2 Systemarchitektur

Die Architektur der UML-Lern-App basiert auf dem Model-View-ViewModel-Muster (MVVM). Dieses Architekturkonzept trennt die Datenhaltung, die Benutzeroberfläche und die Geschäftslogik klar voneinander. Dadurch ist die Anwendung leicht wartbar und flexibel erweiterbar.  
Das Modell (Model) speichert und verwaltet alle Anwendungsdaten, etwa Informationen zu Benutzern, Kursen und Fragen, und stellt die Verbindung zur Firebase-Datenbank her. Das ViewModel bildet die Mittelschicht zwischen Daten und Benutzeroberfläche: Es ruft Daten ab, bereitet sie für die Anzeige auf und überwacht Änderungen, damit die Oberfläche sich automatisch aktualisiert. Die Ansicht (View) umfasst die grafischen Layouts in XML, über die Lernende mit der App interagieren. Diese klare Trennung erleichtert die Entwicklung, das Testen und spätere Anpassungen einzelner Komponenten. Das gewählte Muster entspricht modernen Architekturprinzipien mobiler Anwendungen.

4.3 Datenmodellierung

Für die Datenspeicherung der Anwendung wird Firebase Cloud Firestore eingesetzt – eine dokumentenorientierte NoSQL-Datenbank, die sich durch hohe Skalierbarkeit, einfache Datenstrukturierung und automatische Synchronisation mit mobilen Endgeräten auszeichnet. Firestore speichert die Informationen nicht in Tabellen, sondern in Sammlungen und Dokumenten. Jedes Dokument besteht aus einzelnen Feld-Wert-Paaren und kann wiederum weitere Unterdokumente oder Subcollections enthalten. Dieses Konzept ermöglicht eine flexible und hierarchische Organisation der Daten, die sich besonders für dynamische mobile Lernanwendungen eignet.Die Datenbankstruktur wurde so entworfen, dass sie übersichtlich, logisch und erweiterbar bleibt. Ziel war es, die zentralen Funktionen der App Benutzerverwaltung, Kursorganisation, Quiz-Mechanik, Fortschrittsauswertung, Zeitmessung und Offline-Nutzung – vollständig und klar abbilden zu können. Insgesamt wurden sechs Hauptsammlungen definiert: Benutzer, Kurse, Lerneinheiten, Fragen, Versuche sowie Feedback. Ergänzend existieren Subcollections für Fehler und Notizen innerhalb der Benutzerdaten.In der Sammlung Benutzer werden alle registrierten Personen verwaltet. Sie enthält grundlegende Informationen wie Benutzername, E-Mail-Adresse, Rolle, Punktestand, aktuelles Level und das Datum der Registrierung. Über das Feld role wird die jeweilige Zugriffsberechtigung gesteuert: Lernende dürfen Inhalte lesen, Quizfragen bearbeiten, Notizen hinzufügen und Feedback senden, während Administratorinnen und Administratoren zusätzliche Schreibrechte besitzen, um Kurse, Lerneinheiten und Fragen zu erstellen oder zu ändern. Diese Trennung der Berechtigungen wird durch speziell konfigurierte Firestore-Sicherheitsregeln technisch sichergestellt. Für jede Benutzerin und jeden Benutzer existieren außerdem zwei Unterbereiche: mistakes zur Speicherung falsch beantworteter Aufgaben sowie notes für persönliche Notizen. Diese Struktur unterstützt individuelle Lernprozesse und gezieltes Wiederholen von schwierigen Themen.

Die Sammlung Kurse bildet die verschiedenen Lernmodule ab. Jedes Kursdokument enthält einen Titel, eine Beschreibung, den Schwierigkeitsgrad beziehungsweise die erforderliche Levelstufe, eine Sortierreihenfolge und einen Veröffentlichungsstatus. Unterhalb der Kurse befindet sich die Subcollection units, in der die einzelnen Lerneinheiten verwaltet werden. Eine Lerneinheit enthält die inhaltliche Beschreibung, die Anzahl der enthaltenen Fragen, ein festgelegtes Zeitlimit für den Quiz-Timer und die Levelgrenze, ab der die Einheit freigeschaltet wird. Auf diese Weise lassen sich Lernfortschritte strukturiert aufbauen und der Schwierigkeitsgrad der Inhalte kann stufenweise erhöht werden. Jede Lerneinheit besitzt wiederum eine eigene Unterstruktur für questions. Eine Frage besteht aus dem Fragetext, mehreren Antwortoptionen, der Position der richtigen Lösung, einem Punktwert sowie optionalen Parametern wie Schwierigkeitsgrad oder zufälliger Reihenfolge der Antworten. Durch diese flexible Gestaltung lassen sich verschiedene Fragetypen und Bewertungslogiken problemlos integrieren.

Die Sammlung attempts dokumentiert alle abgeschlossenen Quizdurchläufe. Sie enthält die Zuordnung zu Benutzer, Kurs und Einheit sowie die erzielten Punkte, die Anzahl richtiger Antworten, die Gesamtzahl der Fragen und die detaillierte Liste der gegebenen Antworten. Außerdem werden der Start- und Endzeitpunkt sowie die tatsächliche Bearbeitungsdauer in Sekunden gespeichert. Diese Zeitangaben bilden die Grundlage für den integrierten Timer-Mechanismus, der jedem Quiz-Versuch eine feste Laufzeit zuweist und am Ende automatisch eine Auswertung durchführt. Damit ist eine faire und reproduzierbare Bewertung aller Lernenden gewährleistet.

Für die gezielte Wiederholung falsch beantworteter Fragen wird zusätzlich in jeder Benutzersammlung die Subcollection mistakes genutzt. Dort werden Verweise auf die jeweiligen Fragen gespeichert, zusammen mit dem Zeitstempel der letzten falschen Antwort. Auf diese Weise kann die Anwendung gezielt nur jene Aufgaben erneut abfragen, bei denen zuvor Fehler aufgetreten sind. Die Subcollection notes dient zur individuellen Kommentierung von Fragen oder Einheiten und unterstützt so eigenständiges Lernen und Reflexion.

Neben dem Lern- und Quizsystem ist auch ein Feedback-Bereich integriert. In der Sammlung feedback werden Rückmeldungen der Nutzenden gespeichert, etwa Vorschläge, Fehlermeldungen oder allgemeine Hinweise. Diese Funktion erleichtert die fortlaufende Verbesserung der Anwendung und stellt eine direkte Kommunikationsmöglichkeit zwischen Anwendern und Entwicklungsteam bereit.

Ein wesentliches technisches Merkmal des Datenmodells ist die Unterstützung des Offline-Modus. Firestore stellt durch die integrierte Synchronisationslogik sicher, dass alle Daten lokal auf dem Gerät zwischengespeichert werden. Dadurch bleibt die App vollständig nutzbar, selbst wenn zeitweise keine Internetverbindung besteht. Änderungen an Kursen, Notizen oder Quiz-Versuchen werden lokal gespeichert und automatisch mit der Cloud synchronisiert, sobald eine Verbindung wieder verfügbar ist. Damit ist der kontinuierliche Lernfortschritt der Nutzerinnen und Nutzer jederzeit gewährleistet.

Die gesamte Datenstruktur wurde rollenbasiert abgesichert. Über die Sicherheitsregeln von Firestore wird überprüft, ob der authentifizierte Benutzer einer bestimmten Rolle zugeordnet ist und nur auf die für ihn erlaubten Daten zugreift. Lernende können ausschließlich ihre eigenen Profile, Fehler und Notizen einsehen, während Administratorinnen und Administratoren zusätzlich Schreibrechte für Kurs- und Fragensammlungen besitzen. Diese technische Zugriffskontrolle schützt die Integrität der Daten und gewährleistet den Datenschutz der Benutzer. Das zugehörige ER-Diagramm im Anhang A6 verdeutlicht die Beziehungen zwischen den Sammlungen.

Jede Benutzerin bzw. jeder Benutzer steht über Referenzen mit ihren beziehungsweise seinen Quiz-Versuchen in Verbindung. Diese Versuche wiederum beziehen sich auf konkrete Kurse und Lerneinheiten, die wiederum die zugehörigen Fragen enthalten. Darüber hinaus sind Notizen, Fehler und Feedbacks logisch dem jeweiligen Benutzer zugeordnet. Durch diese strukturierte Modellierung ist die Datenbasis der Anwendung vollständig beschrieben. Das System unterstützt alle im Lasten- und Pflichtenheft definierten Funktionen – einschließlich Benutzer-Login, Kursverwaltung, Quiz-Timer, Fortschrittsberechnung, Fehlerwiederholung und Offline-Nutzung – und ist zugleich flexibel genug, um zukünftige Erweiterungen wie Statistiken oder Ranglisten ohne größere Anpassungen zu ermöglichen

## 4.4 Benutzeroberfläche

Die grafische Gestaltung orientiert sich am Material Design von Google und folgt dem IAD-Farbschema mit einem kräftigen Rot (#C62828) als Akzentfarbe und Weiß als Hintergrund. Das Erscheinungsbild ist bewusst schlicht und klar gehalten, mit großzügigen Abständen, klarer Typografie und einem hohen Kontrast. Als Schriftart kommt Roboto zum Einsatz.Die wichtigsten Ansichten der App sind: die Login- und Registrierungsseite, die Kursübersicht, die Quiz-Ansicht, die Ergebnisanzeige, die Fehlerliste, die Notizverwaltung, die Feedback-Seite sowie die Einstellungsseite mit den Bereichen Allgemein, Konto, Datenschutz und Über die App. Nach der ersten Anmeldung wird zusätzlich ein kurzes Onboarding-Tutorial angezeigt, das die wichtigsten Funktionen vorstellt und danach deaktiviert wird.Für Administratorinnen und Administratoren existiert eine eigene Verwaltungsoberfläche, über die neue Kurse oder Fragen angelegt, geändert und gelöscht werden können. Alle Ansichten wurden zunächst als Mockups entworfen (siehe Anhang A5) und anschließend als XML-Layouts umgesetzt. Symbole stammen aus den „Outlined Material Icons“, und sämtliche Oberflächen sind responsiv.

## 4.5 Technischer Aufbau

Die App besteht aus mehreren Aktivitäten und Fragmenten, die zusammen die einzelnen Module bilden. Beim Start prüft eine Splash-Activity, ob der Benutzer bereits angemeldet ist. Wenn nicht, öffnet sich die Login-Activity; andernfalls erfolgt die Weiterleitung in die Hauptansicht. Die Main-Activity enthält eine Navigationsleiste, über die man zwischen Kursen, Fehlerliste, Notizen und Profil wechseln kann. Die Quiz-Funktion wurde besonders sorgfältig entworfen. Vor Beginn kann die gewünschte Anzahl an Fragen ausgewählt werden (z. B. 5, 10 oder alle). Die Fragen und Antwortoptionen erscheinen in zufälliger Reihenfolge, damit kein Lerneffekt durch Positionserkennung entsteht. Bei Multiple-Choice-Aufgaben werden Teilpunkte vergeben: richtige Häkchen erhöhen, falsche reduzieren die Punktzahl, jedoch nie unter null. Alle Aufgaben sind gleich gewichtet; jede trägt dieselbe Punktzahl. Zusätzlich stehen zwei Modi zur Verfügung: ein Übungsmodus ohne Zeitlimit und ein Prüfungsmodus mit Timer, bei dem die Bearbeitung automatisch beendet wird, sobald die Zeit abläuft. Nach Abschluss des Quiz wird das Ergebnis angezeigt. Dabei berechnet die App die erreichten Punkte und aktualisiert das Lernlevel. Falsch beantwortete Fragen werden automatisch in der Fehlerliste gespeichert, um sie später erneut üben zu können. Die App funktioniert auch offline, da Firestore Daten zwischenspeichert und Änderungen automatisch synchronisiert, sobald eine Internetverbindung besteht. Für die Verwaltung ist ein eigener Administrationsbereich vorgesehen, der nur für Konten mit Administratorrolle sichtbar ist. Änderungen an Kursen oder Fragen werden direkt in der Datenbank gespeichert.

4.6 Zusammenfassung

Die Entwurfsphase bildet das technische und gestalterische Fundament der UML-Lern-App. Durch den Einsatz des MVVM-Musters bleibt die Struktur übersichtlich, während Firebase Firestore eine sichere und flexible Datenhaltung ermöglicht. Das Design nach den Material-Design-Richtlinien sorgt für ein modernes Erscheinungsbild und eine intuitive Benutzerführung. Die vollständige Umsetzung aller Kern- und Zusatzfunktionen – von Login, Kursverwaltung und Quiz bis hin zu Notizen, Fehlerlisten, Feedback, Einstellungen und Onboarding macht die Anwendung zu einer umfassenden, digitalen Lernplattform für UML-Diagramme

Fußnoten:  
1. Microsoft (2025): MVVM-Guidelines – Architectural Patterns.  
2. Google (2025): Material Design Principles.

# 5 Implementierungsphase

In der Implementierungsphase wird der zuvor erstellte Entwurf technisch umgesetzt. Das Kapitel dokumentiert den Entwicklungsprozess, beschreibt die eingesetzten Technologien und erläutert, wie die verschiedenen Komponenten miteinander verbunden wurden. Darüber hinaus werden Herausforderungen, Lösungsansätze und Optimierungen beschrieben, die während der Umsetzung aufgetreten sind. Diese Phase bildet den Kern des Projekts, in dem die Anwendung Schritt für Schritt realisiert wurde.

## 5.1 Entwicklungsumgebung

Die Entwicklung der App erfolgte in Android Studio Giraffe (2025) mit der Programmiersprache Kotlin. Verwendet wurde das Android SDK 34 in Kombination mit einem virtuellen Testgerät vom Typ Pixel 6 (Android 13). Als Versionsverwaltung kam GitHub Desktop zum Einsatz, um alle Änderungen nachvollziehbar zu dokumentieren und regelmäßig Backups anzulegen. Firebase wurde direkt über Android Studio eingebunden. Dazu wurde ein Projekt in der Firebase Console angelegt und die notwendigen Komponenten – Authentication, Cloud Firestore und Storage aktiviert.  
Die Anbindung der App an Firebase erfolgte über die automatisch generierte Konfigurationsdatei *google-services.json*.  
Zusätzlich wurden in der Gradle-Datei die benötigten Bibliotheken eingebunden:

* firebase-auth-ktx – Benutzeranmeldung
* firebase-firestore-ktx – Datenbankabfragen
* firebase-storage-ktx – Speicherung von Bildern oder PDFs
* material-components – moderne Benutzeroberflächen

Diese Kombination ermöglichte eine stabile und einheitliche Entwicklungsumgebung.

## 5.2 Struktur der Anwendung

Die App wurde nach dem MVVM-Muster (Model-View-ViewModel) umgesetzt.  
Dadurch sind Logik, Daten und Darstellung klar voneinander getrennt.

* Model: verwaltet die Datenobjekte wie Benutzer, Kurse, Fragen und Notizen.
* ViewModel: greift auf die Firestore-Daten zu und übergibt diese an die Oberfläche.
* View: stellt die Inhalte über XML-Layouts dar und reagiert auf Benutzerinteraktionen.

Diese Struktur erleichtert die Wiederverwendung und zukünftige Erweiterung der App,  
z. B. für neue Kurstypen oder zusätzliche Sprachen.

## 5.3 Firebase-Integration

Die Firebase-Integration bildet das technische Herzstück der Anwendung. Über die Authentifizierung können sich Nutzer registrieren, anmelden oder abmelden. Die Anmeldedaten werden verschlüsselt in Firebase gespeichert, und die App erkennt beim Start automatisch, ob ein Nutzer bereits eingeloggt ist. Alle kurs- und quizbezogenen Informationen werden aus der Sammlung „courses“ und „questions“ geladen. Dazu wurden Abfragen mit Filtern implementiert, die nur aktive Kurse und Fragen anzeigen. Falsch beantwortete Fragen werden in der Sammlung „errors“ gespeichert, damit Lernende sie später gezielt wiederholen können. Die persönlichen Notizen und Feedback-Einträge werden in den Sammlungen „notes“ und „feedback“ verwaltet. Dank der integrierten OfflineFunktionalität von Firestore können Daten auch ohne Internetverbindung angezeigt werden. Sobald eine Verbindung besteht, synchronisiert Firebase alle Änderungen automatisch.

## 5.4 Benutzeroberfläche

Die Benutzeroberfläche wurde mit XML-Layouts gestaltet und an das IAD-Farbschema (Rot #C62828, Weiß) angepasst. Alle Ansichten basieren auf den Material Design Guidelines von Google. Buttons, Karten und Navigationsleisten wurden mit den *Material Components* erstellt, um eine moderne, konsistente Gestaltung sicherzustellen. Die App enthält folgende Hauptansichten:

* Login- und Registrierungsbildschirm
* Kursübersicht
* Quiz-Bildschirm
* Ergebnisanzeige
* Fehlerliste
* Notizen
* Feedback- und Einstellungsseite

Besonderer Wert wurde auf gute Lesbarkeit, intuitive Navigation und ein ruhiges, übersichtliches Erscheinungsbild gelegt. Das Layout ist vollständig responsiv und passt sich automatisch an verschiedene Bildschirmgrößen an

## 5.5 Umsetzung der Kernfunktionen

Die Implementierung der zentralen Funktionen erfolgte schrittweise und orientierte sich an der Planung aus Kapitel 2 und 4. Im Folgenden sind die wichtigsten Funktionalitäten zusammengefasst:

* **Login & Registrierung:**  
  Umsetzung mit Firebase Authentication. Nach erfolgreicher Anmeldung wird der Nutzer zur Hauptansicht weitergeleitet.
* **Kursverwaltung:**  
  Die Kurse werden aus Firestore geladen und nach Schwierigkeitsgrad angezeigt.  
  Administratorinnen und Administratoren können Kurse über ein separates Menü hinzufügen oder bearbeiten.
* **Quizlogik:**  
  Fragen werden aus Firestore geladen und in zufälliger Reihenfolge dargestellt.  
  Es können Einzel- oder Multiple-Choice-Fragen vorkommen.  
  Bei Multiple-Choice-Fragen werden Teilpunkte vergeben, und am Ende erfolgt eine automatische Auswertung.
* **Ergebnis & Fortschritt:**  
  Nach dem Quiz werden Punkte, Lernfortschritt und Level berechnet und im Benutzerprofil gespeichert.
* **Fehlerliste:**  
  Alle falsch beantworteten Fragen werden automatisch gespeichert, um sie später gezielt wiederholen zu können.
* **Notizen:**  
  Nutzer können zu jeder Frage eigene Notizen anlegen, bearbeiten und löschen.
* **Feedback:**  
  Ein einfaches Formular ermöglicht es, Rückmeldungen oder Verbesserungsvorschläge an die Entwicklerin zu senden.
* **Einstellungen:**  
  Hier können Nutzer Sprache, Designmodus (Hell/Dunkel) und Kontoeinstellungen anpassen.

## 5.6 Administrationsbereich

Für Administratoren wurde ein separater Bereich geschaffen, in dem neue Kurse und Fragen angelegt, bearbeitet oder gelöscht werden können. Die Oberfläche ist bewusst schlicht gestaltet und zeigt eine Liste aller vorhandenen Einträge. Über einfache Eingabefelder lassen sich Titel, Beschreibung und Level anpassen. Alle Änderungen werden sofort in Firebase übernommen und stehen in Echtzeit zur Verfügung

## 5.7 Test und Optimierung während der Entwicklung

Nach der Implementierung einzelner Module wurden regelmäßig Funktionstests durchgeführt. Dabei wurde geprüft, ob alle Ansichten korrekt geladen werden, die Daten synchron bleiben und Benutzeraktionen zuverlässig funktionieren. Besonderes Augenmerk lag auf der Stabilität des Quiz-Ablaufs und der Synchronisation mit der Datenbank. Fehler wurden direkt in Android Studio über den *Logcat* verfolgt und behoben. Durch wiederholtes Testen konnten Abstürze, Darstellungsfehler oder ungewollte Datenüberschreibungen vermieden werden.

## 5.8 Zusammenfassung

In der Implementierungsphase wurde der Entwurf erfolgreich in eine funktionsfähige Anwendung überführt.Alle geplanten Funktionen – von der Anmeldung über das Quizsystem bis hin zu dem Fehler und Notizverwaltung – wurden vollständig umgesetzt. Die Nutzung moderner Technologien wie Kotlin, Firebase und Material Design führte zu einer stabilen und benutzerfreundlichen App. Die Anwendung kann jederzeit erweitert werden, beispielsweise um zusätzliche Kurse oder neue Diagrammtypen. Damit bildet die Implementierung den technischen Kern des Projekts und legt den Grundstein für die abschließende Test- und Dokumentationsphase.

**Fußnoten:**  
¹ Firebase Documentation (2025): Developer Guide for Android.  
² Google (2025): Material Design Guidelines.