

Bericht über die Entwicklung der Zeiterfassungsapp

1. Einleitung	4
1.1 Hintergrund und Ziel der App	4
1.2 Bedeutung der Zeiterfassung in modernen Arbeitsumgebungen	4
1.3 Übersicht über die verwendeten Technologien und Tools	4
2. Entwicklungsumgebung und Technologien	5
2.1 Android Studio	5
2.1.1 Installation und Einrichtung	5
2.1.2 Funktionen und Vorteile	5
2.1.3 Integration mit Firebase	5
2.2 Programmiersprache: Java	5
2.2.1 Wahl der Programmiersprache	5
2.2.2 Vorteile von Java für Android-Entwicklung	6
2.2.3 Beispielcode und seine Funktionsweise	6
3. Architektur und Design	7
3.1 Anwendungsarchitektur	7
3.1.1 Client-Server-Modell	7
3.1.2 Datenfluss und Interaktionen	7
3.2 Benutzeroberfläche (UI)	7
3.2.1 Layout und Designprinzipien	7
3.2.2 Screenshots und Beschreibung der UI-Komponenten	7
3.3 Backend-Design	9
3.3.1 Datenstruktur in Cloud Firestore	9
3.3.2 Sicherheitsregeln und Indexierung	9
3.3.3 Screenshots und Beschreibung	9
4. Funktionen der App	10
4.1 Benutzeranmeldung und -registrierung	10
4.1.1 Ablauf der Registrierung	10
4.1.2 Ablauf der Anmeldung	10
4.1.3 Fehlerbehandlung und Benutzerfeedback	10
4.2 Zeiterfassung	10
4.2.1 Check-In und Check-Out Funktionen	10
4.2.2 Datenvalidierung und -speicherung	10
4.2.3 Synchronisierung und Echtzeit-Updates	10
4.3 Zeitübersicht	10
4.3.1 Anzeige der Zeiteinträge	10
4.3.2 Formatierung und Darstellung der Zeitdaten	11
4.4 Admin-Funktionen	11
4.4.1 Zugriffskontrolle und Rollenmanagement	11
4.4.2 Verwaltung von Benutzerdaten und Zeiteinträgen	11
4.4.3 Berichtserstellung und Analyse	11

Bericht über die Entwicklung der Zeiterfassungsapp

5. Implementierung	12
5.1 Code-Übersicht	12
5.1.1 Hauptaktivität (MainActivity.java)	12
5.1.2 Admin-Aktivität (AdminActivity.java)	12
5.1.3 Login-Aktivität (LoginActivity.java)	12
5.1.4 App-Benachrichtigungen (MyFirebaseMessagingService.java)	12
5.2 Layout-Dateien	12
5.2.1 activity_main.xml	12
5.2.2 activity_login.xml	12
5.2.3 activity_admin.xml	12
5.3 Firebase-Konfiguration	12
5.3.1 Firebase-Projekt erstellen und konfigurieren	12
5.3.2 Firebase Authentication einrichten	13
5.3.3 Cloud Firestore-Datenbankstruktur erstellen	13
6. Herausforderungen und Lösungen	14
6.1 Datenintegration und -strukturierung	14
6.1.1 Herausforderung: Konsistente Datenstruktur	14
6.1.2 Lösung: Nutzung von Firestore-Datenmodellen und Abfragen	14
6.2 Benutzeranmeldung und -authentifizierung	14
6.2.1 Herausforderung: Sichere Authentifizierung	14
6.2.2 Lösung: Implementierung von Firebase Authentication	14
6.3 Performance-Optimierung	14
6.3.1 Herausforderung: Effiziente Datenabfragen	14
6.3.2 Lösung: Nutzung von Indexen und Optimierung der Firestore-Abfragen	14
6.4 Fehlerbehandlung und Benutzerfeedback	14
6.4.1 Herausforderung: Benutzerfreundliche Fehlermeldungen	14
6.4.2 Lösung: Implementierung von Toast-Nachrichten und Dialogen	15
7. Teststrategien und -methoden	16
7.1 Integrationstests	16
7.1.1 Testen der Interaktionen zwischen Komponenten	16
7.1.2 Beispiele für Integrationstests	16
7.2 Benutzerakzeptanztests	16
7.2.1 Testen der Benutzeroberfläche	16
7.2.2 Feedback von Testnutzern und Anpassungen	16
8. Sicherheit und Datenschutz	17
8.1 Sicherheitsüberlegungen	17
8.1.1 Verschlüsselung und sichere Speicherung	17
8.1.2 Zugriffskontrollen und Authentifizierung	17
9. Wartung und zukünftige Erweiterungen	18
9.1 Wartungsplan	18
9.1.1 Regelmäßige Updates und Bugfixes	18

Bericht über die Entwicklung der Zeiterfassungsapp

9.1.2 Überwachung und Performance-Management	18
9.2 Geplante Erweiterungen	18
9.2.1 Funktionen zur Verbesserung der Benutzererfahrung	18
9.2.2 Integration zusätzlicher Firebase-Dienste oder Drittanbieter-APIs	18
10. Fazit	19
10.1 Zusammenfassung der wichtigsten Punkte	19
10.2 Reflektion über den Entwicklungsprozess	19
10.3 Schlussfolgerungen und Empfehlungen	19
11. Anhang	20
11.1 Referenzen und Quellen	20

Bericht über die Entwicklung der Zeiterfassungsapp

1. Einleitung

1.1 Hintergrund und Ziel der App

In der heutigen Arbeitswelt ist präzise Zeiterfassung von großer Bedeutung für die Personalverwaltung und das Projektmanagement. Die Zeiterfassungsapp wurde entwickelt, um diesen Bedarf zu decken, indem sie eine benutzerfreundliche Lösung für die Erfassung und Verwaltung von Arbeitszeiten bietet. Die App ermöglicht es Nutzern, ihre Arbeitszeiten einfach zu protokollieren, und bietet Administratoren die Möglichkeit, die Zeiterfassungen der Benutzer zu überwachen und zu verwalten.

1.2 Bedeutung der Zeiterfassung in modernen Arbeitsumgebungen

Zeiterfassung ist ein entscheidendes Element für das Management von Arbeitsabläufen, insbesondere in flexiblen Arbeitsmodellen und für projektbasierte Arbeiten. Eine präzise und zuverlässige Zeiterfassung hilft Unternehmen dabei, die Produktivität zu überwachen, Überstunden zu berechnen und die Einhaltung gesetzlicher Vorschriften zu gewährleisten.

1.3 Übersicht über die verwendeten Technologien und Tools

Die Zeiterfassungsapp wurde mit Android Studio entwickelt und nutzt Java für die Programmierung der Anwendungslogik. Firebase wird als Backend-Dienst verwendet, um Authentifizierung, Datenmanagement und Echtzeit-Synchronisation zu ermöglichen.

Bericht über die Entwicklung der Zeiterfassungsapp

2. Entwicklungsumgebung und Technologien

2.1 Android Studio

Android Studio ist die bevorzugte Entwicklungsumgebung für Android-Entwickler. Es bietet eine integrierte Entwicklungsumgebung, die umfassende Tools und Funktionen für die App-Entwicklung bereitstellt.

2.1.1 Installation und Einrichtung

Um Android Studio zu installieren, wird die neueste Version von der offiziellen Website heruntergeladen und auf dem Entwickler-PC installiert. Die Einrichtung umfasst die Konfiguration der Android SDKs und die Installation zusätzlicher Plugins, die für die App-Entwicklung erforderlich sind. Für weiteres gucken Sie die Unterlagen vom Prof. Kampmann auf OLAT unter dem Kurs MOBC.

2.1.2 Funktionen und Vorteile

Android Studio bietet Funktionen wie:

- **Code-Editor:** Unterstützung für Syntaxhervorhebung, Code-Vervollständigung und Refactoring.
- **Layout-Editor:** Visuelle Gestaltung der Benutzeroberfläche mit Drag-and-Drop-Funktionalität.
- **Debugger:** Werkzeuge zum Testen und Debuggen von Anwendungen.
- **Emulatoren:** Virtuelle Geräte zur Ausführung und Tests von Apps in verschiedenen Konfigurationen.

2.1.3 Integration mit Firebase

Die Integration von Firebase in Android Studio erfolgt durch das Hinzufügen von Firebase-Bibliotheken zum Projekt und die Verwendung von Google Services JSON-Dateien, die Konfigurationsdetails enthalten. Android Studio bietet spezielle Assistenten, die die Integration erleichtern. Gucken Sie die Links unter dem letzten Kapitel dieses Berichts, nämlich die Referenzen und Quellen.

2.2 Programmiersprache: Java

Java wurde für die Entwicklung der Zeiterfassungsapp verwendet. Es ist eine weit verbreitete Programmiersprache, die besonders für Android-Anwendungen geeignet ist.

2.2.1 Wahl der Programmiersprache

Java wurde aufgrund seiner Stabilität, umfangreichen Bibliotheken und der Unterstützung durch die Android-Plattform gewählt. Es ermöglicht eine effiziente Entwicklung und Wartung von Android-Anwendungen. In diesem Projekt wurde Java sogar vorausgesetzt.

Bericht über die Entwicklung der Zeiterfassungsapp

2.2.2 Vorteile von Java für Android-Entwicklung

- **Plattformunabhängigkeit:** Java-Code kann auf verschiedenen Plattformen ausgeführt werden, was die Portabilität verbessert.
- **Reiche Bibliothek:** Java bietet umfangreiche Bibliotheken und Frameworks, die die Entwicklung beschleunigen.
- **Große Entwicklergemeinschaft:** Eine große Community bietet Unterstützung und Ressourcen für die Problemlösung.

2.2.3 Beispielcode und seine Funktionsweise

Der folgende Code zeigt die Implementierung der Check-In-Funktion:

```
private void checkIn() {
    FirebaseAuth.getInstance().getCurrentUser();
    if (user != null) {
        // Prüfen, ob schon eingestempelt wurde
        FirebaseFirestore.getInstance().collection("timeEntries")
            .whereEqualTo("userId", user.getId())
            .whereEqualTo("checkedIn", true)
            .get()

        .addOnCompleteListener(task -> {
            if (task.isSuccessful() && task.getResult().isEmpty()) {
                // Tragen von check-in Zeit
                Map<String, Object> entry = new HashMap<>();
                entry.put("userId", user.getId());
                entry.put("checkInTime", new Timestamp(new Date()));
                entry.put("checkedIn", true);
                FirebaseFirestore.getInstance().collection("timeEntries")
                    .add(entry)
                    .addOnSuccessListener(documentReference -> {
                        Toast.makeText(this, "Erfolgreich eingestempelt.",
Toast.LENGTH_SHORT).show();
                    })
                    .addOnFailureListener(e -> {
                        Toast.makeText(this, "Fehler checking in", Toast.LENGTH_SHORT).show();
                    });
            } else {
                Toast.makeText(this, "Schon eingestempelt.", Toast.LENGTH_SHORT).show();
            }
        });
    }
}
```

Bericht über die Entwicklung der Zeiterfassungsapp

3. Architektur und Design

3.1 Anwendungsarchitektur

Die Architektur der Zeiterfassungsapp basiert auf dem Client-Server-Modell, wobei der Client (die App) mit dem Server (Firebase) kommuniziert, um Daten zu speichern und abzurufen.

3.1.1 Client-Server-Modell

- **Client:** Die mobile App, die auf dem Android-Gerät des Nutzers installiert ist.
- **Server:** Firebase-Dienste, die als Backend für die Authentifizierung, Datenhaltung und -verarbeitung dienen.

3.1.2 Datenfluss und Interaktionen

- **Benutzeranmeldung:** Die App sendet Anmeldeinformationen an Firebase Authentication und erhält ein Authentifizierungstoken zurück.
- **Zeiterfassung:** Bei Check-In oder Check-Out werden Zeitstempel an Cloud Firestore gesendet und dort gespeichert.
- **Datenabfrage:** Die App ruft Zeiteinträge aus Cloud Firestore ab und zeigt sie dem Benutzer an. Sowie auch die Nutzer-E-mails und ihre IDs werden aufgerufen und nur dem Admin angezeigt.

3.2 Benutzeroberfläche (UI)

Die Benutzeroberfläche der App ist in verschiedenen Aktivitäten und Layouts organisiert.

3.2.1 Layout und Designprinzipien

- **Klarheit und Einfachheit:** Die Benutzeroberfläche ist so gestaltet, dass sie intuitiv und einfach zu bedienen ist.
- **Responsives Design:** Layouts passen sich verschiedenen Bildschirmgrößen an, um eine optimale Darstellung auf unterschiedlichen Geräten zu gewährleisten.

3.2.2 Screenshots und Beschreibung der UI-Komponenten

Screenshots der Hauptaktivitäten wie der Login-Aktivität, der Hauptaktivität und der Admin-Aktivität folgen gleich. Achten Sie bitte auf Bilder-Titeln, um zu verstehen, worum es in Bildern geht.

Bericht über die Entwicklung der Zeiterfassungsapp

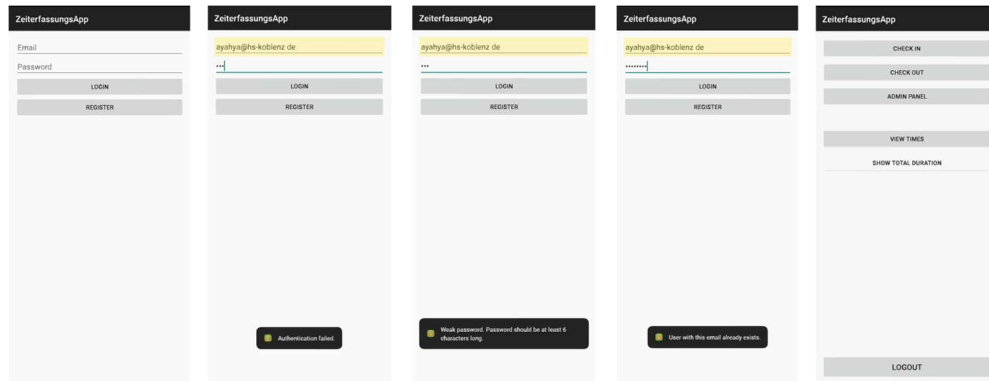


Abb. 1 Login-Seite

Abb. 2 Anmeldung mit falschem Passwort oder ohne Internet.

Abb. 3 Anmeldung mit schwachem Passwort

Abb. 4 Anmeldung mit vorhandener E-Mail

Abb. 5 Hauptseite

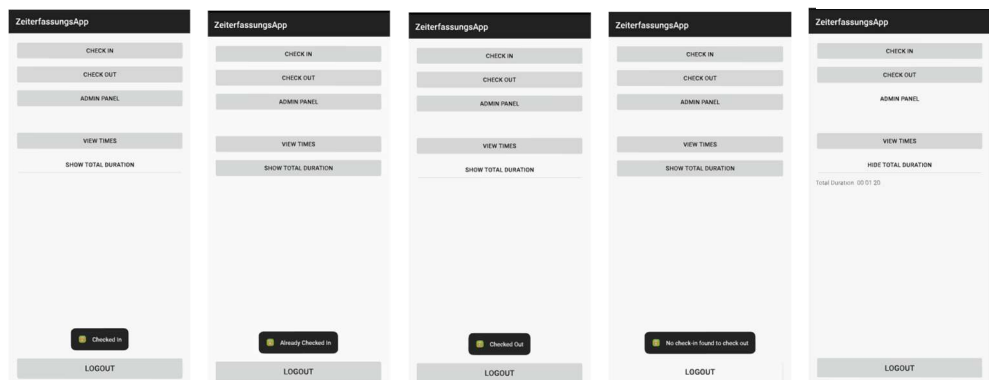


Abb. 6 Einstempeln

Abb. 7 Nochmal einstempeln

Abb. 8 Ausstempeln

Abb. 9 Nochmal ausstempeln

Abb. 10 Anzeigen von Summe der Zeiteinträgen

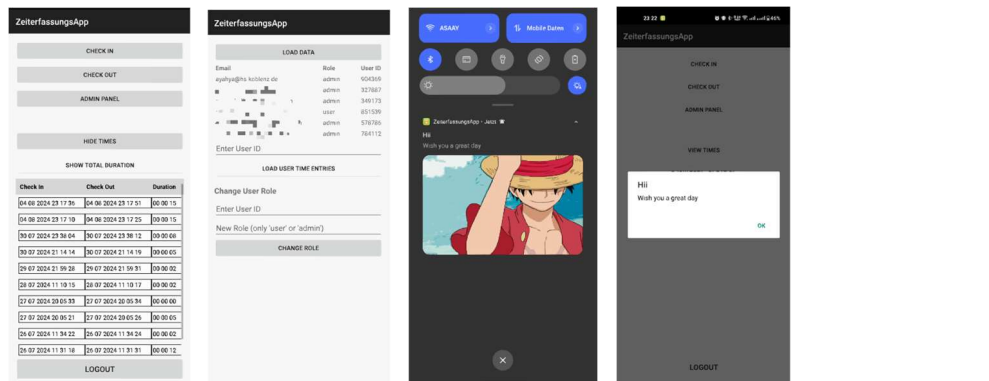


Abb. 11 Anzeigen von Zeiteinträgen

Abb. 12 Admin-Seite

Abb. 13 App-Benachrichtigung

Abb. 24 In-App Messaging

Bericht über die Entwicklung der Zeiterfassungsapp

3.3 Backend-Design

Das Backend-Design umfasst die Struktur und Verwaltung der Daten in Firebase.

3.3.1 Datenstruktur in Cloud Firestore

Die Datenstruktur umfasst Sammlungen und Dokumente für Benutzer- und Zeiteinträge:

- **Benutzerdaten:** Sammlung „users“ mit Dokumenten, die Benutzerdaten wie E-Mail, Benutzer-ID und Rolle enthalten.
- **Zeiteinträge:** Sammlung „timeEntries“ mit Dokumenten, die Check-In- und Check-Out-Zeiten für jeden Benutzer enthalten.

3.3.2 Sicherheitsregeln und Indexierung

- **Sicherheitsregeln:** Definieren den Zugriff auf Daten basierend auf Benutzerrollen und Authentifizierungsstatus.
- **Indexierung:** Optimierte Datenabfragen durch Erstellen von Indizes für häufig verwendete Abfragen wie Zeiteinträge nach Benutzer-ID.

3.3.3 Screenshots und Beschreibung

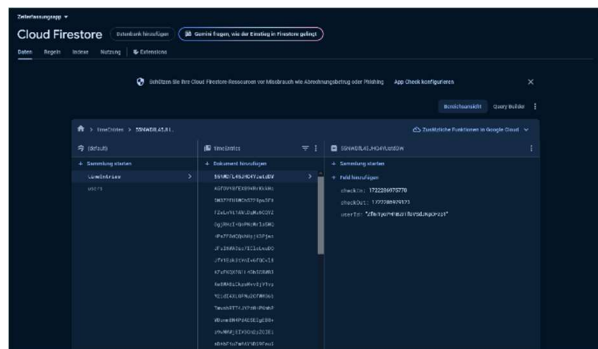


Abb. 1 Aussehen von Zeiteinträge in Firebase

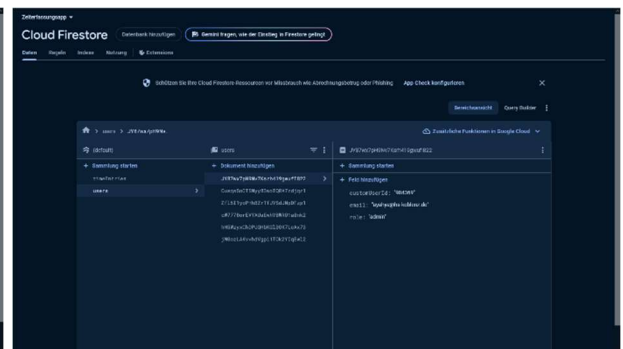


Abb. 2 Liste aller Nutzer auf Firebase

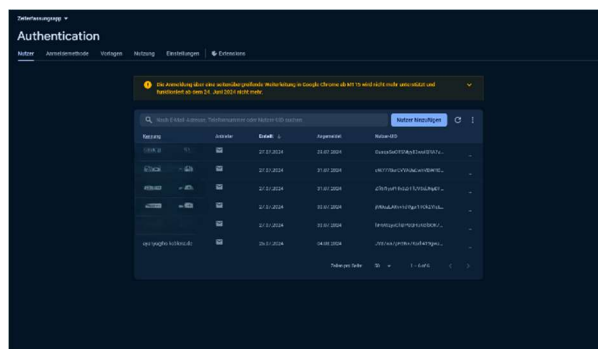


Abb. 3 Authentifizierungsseite auf Firebase

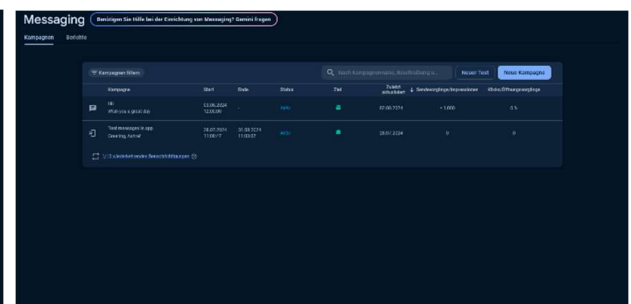


Abb. 4 Benachrichtigungs- bzw. In-App-Messagingseite

Bericht über die Entwicklung der Zeiterfassungsapp

4. Funktionen der App

4.1 Benutzeranmeldung und -registrierung

4.1.1 Ablauf der Registrierung

Nutzer geben ihre E-Mail-Adresse und ein Passwort ein, um ein Konto zu erstellen. Firebase Authentication prüft die Anmeldeinformationen und erstellt ein neues Benutzerkonto. Eingabe von einer vorhandenen bzw. schon registrierter E-Mail, sowie schwachem Passwort wird abgelehnt. Eine klare Rückmeldung kriegt der Nutzer als Toast angezeigt.

4.1.2 Ablauf der Anmeldung

Nutzer melden sich mit ihrer E-Mail-Adresse und ihrem Passwort an. Bei erfolgreicher Authentifizierung wird der Nutzer zur Hauptaktivität weitergeleitet.

4.1.3 Fehlerbehandlung und Benutzerfeedback

Fehler bei der Anmeldung oder Registrierung werden durch Toast-Nachrichten angezeigt, die den Nutzer über Probleme informieren, z.B. ungültige Anmeldeinformationen oder bereits existierende E-Mail-Adressen.

4.2 Zeiterfassung

4.2.1 Check-In und Check-Out Funktionen

Die App ermöglicht das Ein- und Ausstempeln durch Buttondruck. Die aktuellen Zeitstempel werden in Cloud Firestore gespeichert und mit der Benutzer-ID verknüpft.

4.2.2 Datenvalidierung und -speicherung

Vor dem Speichern wird überprüft, ob der Benutzer bereits eingestempelt ist, um doppelte Einträge zu vermeiden. Bei doppelten Einträgen wird der zweite Eintrag abgelehnt und eine passende Rückmeldung wird dem Nutzer als Toast angezeigt. Dies wurde in den obigen Screen Shoots gezeigt.

4.2.3 Synchronisierung und Echtzeit-Updates

Cloud Firestore ermöglicht Echtzeit-Synchronisation, sodass Zeiteinträge sofort angezeigt werden, sobald sie gespeichert sind.

4.3 Zeitübersicht

4.3.1 Anzeige der Zeiteinträge

Die Zeitübersicht zeigt alle Zeiteinträge des Benutzers in einer Tabelle an, die nach Datum und Uhrzeit sortiert sind.

Bericht über die Entwicklung der Zeiterfassungsapp

4.3.2 Formatierung und Darstellung der Zeitdaten

Zeiteinträge werden in einem benutzerfreundlichen Format angezeigt, das den Beginn und das Ende jeder Arbeitsperiode sowie die Dauer berechnet.

4.4 Admin-Funktionen

4.4.1 Zugriffskontrolle und Rollenmanagement

Administratoren haben Zugriff auf spezielle Funktionen zur Verwaltung von Benutzerdaten und Zeitübersichten. Die Zugriffsrechte werden durch Benutzerrollen gesteuert.

4.4.2 Verwaltung von Benutzerdaten und Zeiteinträgen

Administratoren können Zeitdaten aller Benutzer einsehen und bearbeiten sowie Benutzerrollen ändern.

4.4.3 Berichtserstellung und Analyse

Die App ermöglicht die Erstellung von Berichten über die Arbeitszeiten und bietet Funktionen zur Analyse von Zeitdaten (nur Cloudseitig).

Bericht über die Entwicklung der Zeiterfassungsapp

5. Implementierung

5.1 Code-Übersicht

5.1.1 Hauptaktivität (MainActivity.java)

Die Hauptaktivität verwaltet die Benutzeroberfläche für die Zeiterfassung und die Zeitübersicht. Sie enthält Methoden zum Logout, Ein- und Ausstempeln sowie zur Anzeige von Zeiteinträgen, Admin-Seite und Summe der Zeiteinträge.

5.1.2 Admin-Aktivität (AdminActivity.java)

Die Admin-Aktivität bietet Funktionen zur Verwaltung von Benutzerdaten und Zeiteinträgen sowie zur Änderung von Benutzerrollen.

5.1.3 Login-Aktivität (LoginActivity.java)

Die Login-Aktivität verwaltet die Benutzeranmeldung und -registrierung sowie das Error-Handling.

5.1.4 App-Benachrichtigungen (MyFirebaseMessagingService.java)

Die App-Benachrichtigungen kümmert sich um Benachrichtigungen, die aus Serverseite in die App geschickt werden können. Dies umfasst zwei Arte von Benachrichtigungen. Die erste Art der Benachrichtigungen ist die Benachrichtigung, die dem Nutzer an den oberen Rand des Bildschirms angezeigt wird, auch wenn die App nicht genutzt wird. Die zweite Möglichkeit heißt In-App-Messaging. Diese Art von Benachrichtigungen wird nur dann gezeigt, falls der Nutzer die App zur Zeit der Sendung aus Cloud nutzt.

5.2 Layout-Dateien

5.2.1 activity_main.xml

Dieses Layout enthält die UI-Komponenten für die Hauptaktivität der App, einschließlich Buttons und Tabellenlayouts für die Zeiterfassung und -übersicht.

5.2.2 activity_login.xml

Dieses Layout bietet Eingabefelder für E-Mail-Adresse und Passwort sowie Buttons für die Anmeldung und Registrierung.

5.2.3 activity_admin.xml

Das Admin-Layout enthält die UI-Komponenten für die Verwaltung von Benutzerdaten und Zeitübersichten.

5.3 Firebase-Konfiguration

5.3.1 Firebase-Projekt erstellen und konfigurieren

Ein Firebase-Projekt wird erstellt und konfiguriert, um die erforderlichen Dienste wie Authentication und Firestore zu nutzen. Die Konfigurationsdateien werden in das

Bericht über die Entwicklung der Zeiterfassungsapp

Android-Projekt integriert. Für weiteres dazu gucken Sie die Webseite von Firebase. Hilfreiche Links zu diesem Thema finden Sie unter in Referenzen und Quellen beigefügt.

5.3.2 Firebase Authentication einrichten

Firebase Authentication wird konfiguriert, um die Registrierung und Anmeldung der Nutzer zu ermöglichen. Dabei wird eine E-Mail-Adresse und ein Passwort benötigt. Man könnte auch Zwei-Faktor-Authentifizierung nutzen. Dies ist aber kostenpflichtig und nicht nötig in diesem Projekt.

5.3.3 Cloud Firestore-Datenbankstruktur erstellen

Die Datenbankstruktur wird in Firestore definiert, um Benutzerdaten und Zeiteinträge zu speichern. Sie besteht z.b. aus mehreren Sammlungen, die aus mehreren Felder bestehen. Al Beispiel nenne ich die user-Sammlung, die aus drei Felder besteht, nämlich Name, E-Mail und Id.

Bericht über die Entwicklung der Zeiterfassungsapp

6. Herausforderungen und Lösungen

6.1 Datenintegration und -strukturierung

6.1.1 Herausforderung: Konsistente Datenstruktur

Es war notwendig, eine konsistente Datenstruktur zu erstellen, um sicherzustellen, dass alle Zeiteinträge korrekt erfasst und abgerufen werden.

6.1.2 Lösung: Nutzung von Firestore-Datenmodellen und Abfragen

Durch sorgfältige Planung der Datenstruktur und die Nutzung von Firestore-Abfragen konnte die Konsistenz und Effizienz der Datenverwaltung gewährleistet werden.

6.2 Benutzeranmeldung und -authentifizierung

6.2.1 Herausforderung: Sichere Authentifizierung

Die Sicherstellung einer sicheren Authentifizierung und der Schutz von Benutzerdaten waren wesentliche Herausforderungen.

6.2.2 Lösung: Implementierung von Firebase Authentication

Firebase Authentication ermöglichte eine sichere Verwaltung von Anmelde- und Registrierungsprozessen. Dabei werden Dokumenten mit schwierigen IDs genutzt. Die Dokumente der Zeiteinträge werden nur einmal von Firebase benutzt, damit keine Mischung auftritt.

6.3 Performance-Optimierung

6.3.1 Herausforderung: Effiziente Datenabfragen

Die Performance der App konnte durch effiziente Datenabfragen und die Nutzung von Indizes in Firestore optimiert werden.

6.3.2 Lösung: Nutzung von Indexen und Optimierung der Firestore-Abfragen

Durch die Implementierung von Indexen und Optimierungen in den Abfragen konnte die Ladezeit der Daten erheblich verkürzt werden.

6.4 Fehlerbehandlung und Benutzerfeedback

6.4.1 Herausforderung: Benutzerfreundliche Fehlermeldungen

Eine Herausforderung bestand darin, den Benutzern klare und hilfreiche Fehlermeldungen anzuzeigen.

Bericht über die Entwicklung der Zeiterfassungssapp

6.4.2 Lösung: Implementierung von Toast-Nachrichten und Dialogen

Die Verwendung von Toast-Nachrichten und Dialogen ermöglichte eine benutzerfreundliche Fehlermeldung und Feedback.

Bericht über die Entwicklung der Zeiterfassungsapp

7. Teststrategien und -methoden

7.1 Integrationstests

7.1.1 Testen der Interaktionen zwischen Komponenten

Integrationstests wurden durchgeführt, um sicherzustellen, dass verschiedene Komponenten der App korrekt zusammenarbeiten.

7.1.2 Beispiele für Integrationstests

Als Beispiele gucken Sie bitte die Screenshots im Kap 2 und 3. Dies repräsentieren die Zusammenarbeit der Komponenten mit sich selbst und mit dem Server.

7.2 Benutzerakzeptanztests

7.2.1 Testen der Benutzeroberfläche

Benutzerakzeptanztests wurden durchgeführt, um sicherzustellen, dass die Benutzeroberfläche den Anforderungen und Erwartungen der Nutzer entspricht.

7.2.2 Feedback von Testnutzern und Anpassungen

Das Feedback von Testnutzern wurde gesammelt und genutzt, um Verbesserungen und Anpassungen an der App vorzunehmen.

Bericht über die Entwicklung der Zeiterfassungsapp

8. Sicherheit und Datenschutz

8.1 Sicherheitsüberlegungen

8.1.1 Verschlüsselung und sichere Speicherung

Sensible Daten, wie Passwörter, werden verschlüsselt und sicher gespeichert, um die Integrität und Vertraulichkeit zu gewährleisten. Nutzung von Dokumenten nur einmalig auf Server, um Mischung zu vermeiden.

8.1.2 Zugriffskontrollen und Authentifizierung

Zugriffskontrollen werden implementiert, um sicherzustellen, dass nur autorisierte Benutzer auf bestimmte Funktionen und Daten zugreifen können. Dies lässt sich an die Admin-Seite klar darstellen, da nur Admins die Zeiteinträge des jeweiligen Nutzers anzeigen dürfen, sowie die Änderung der Rolle des jeweiligen Nutzers.

Bericht über die Entwicklung der Zeiterfassungsapp

9. Wartung und zukünftige Erweiterungen

9.1 Wartungsplan

9.1.1 Regelmäßige Updates und Bugfixes

Ein Plan für regelmäßige Updates und Bugfixes kann entwickelt werden, um die App aktuell und fehlerfrei zu halten. Dies ist gedacht, falls die App zukünftig benutzt werden soll.

9.1.2 Überwachung und Performance-Management

Die Performance der App wird überwacht, um mögliche Probleme frühzeitig zu erkennen und zu beheben. Dabei hilft das Feedback der Nutzer und Tester.

9.2 Geplante Erweiterungen

9.2.1 Funktionen zur Verbesserung der Benutzererfahrung

Zukünftige Erweiterungen könnten Funktionen zur Verbesserung der Benutzererfahrung, wie z.B.

- Berichterstellung, so dass man dies weiterleiten kann, wenn es den Bedarf gibt.
- Anmeldedaten speichern, damit der Nutzer das Passwort und die E-Mail nicht immer eingeben muss, also Erleichterung der Nutzung meiner App
- Responsivität bei vielen Geräten sowie in Hoch- und Querformat des Bildschirms
- Die automatische Rechnung von Überstunden bzw. Minus-Stunden.
- Die Möglichkeit die Zeiteinträge zu speichern oder löschen sowie der Trennung der Zeitstempel verschiedenen Monaten.
oder zusätzliche Integrationen, umfassen.

9.2.2 Integration zusätzlicher Firebase-Dienste oder Drittanbieter-APIs

Die Integration zusätzlicher Firebase-Dienste oder Drittanbieter-APIs wird in Betracht gezogen, um die Funktionalität der App zu erweitern. Als Beispiel möchte ich die App-Hosting. Dies kann dem Nutzer erheblich helfen und die Möglichkeit geben, die App auf Ihrem Rechner zu nutzen.

Bericht über die Entwicklung der Zeiterfassungsapp

10. Fazit

10.1 Zusammenfassung der wichtigsten Punkte

Die Zeiterfassungsapp bietet eine benutzerfreundliche Lösung für die Zeiterfassung und -verwaltung. Die verwendeten Technologien und Werkzeuge haben die Entwicklung erleichtert und ermöglicht, eine leistungsfähige und sichere Anwendung zu erstellen.

10.2 Reflektion über den Entwicklungsprozess

Der Entwicklungsprozess umfasste die Auswahl geeigneter Technologien, die Implementierung der App-Features und die Bewältigung von Herausforderungen. Dabei wurden verschiedene Methoden und Quellen wie YouTube, Firebase, Stack Overflow und usw. benutzt. Dadurch habe ich gelernt, wie ich recherchieren und wie ich Hilfe bei Debugging bekommen kann. Ohnehin zu sagen ist auch, dass mir dieses Projekt die erste Verbindung zu Android Studio mit Java gegeben hat. Dies ist sehr hilfreich für die spätere Arbeit als Softwareentwickler.

10.3 Schlussfolgerungen und Empfehlungen

Die App erfüllt die Anforderungen und Erwartungen der Benutzer sowie des Projekts und bietet zahlreiche Funktionen zur Zeiterfassung und -verwaltung. Es gibt viele Möglichkeiten für Funktionen und Erweiterungen, die zum Teil oben erwähnt wurden und man der App hinzufügen kann.

Bericht über die Entwicklung der Zeiterfassungsapp

11. Anhang

11.1 Referenzen und Quellen

- <https://firebase.google.com/docs/android/setup?hl=de>
- <https://firebase.google.com/docs/cloud-messaging?hl=de>
- <https://firebase.google.com/docs/cloud-messaging/android/receive?hl=de>
- https://www.youtube.com/watch?v=0pl60ME5WQk&ab_channel=TechFreak
- <https://firebase.google.com/docs/projects/learn-more?hl=de>
- <https://firebase.google.com/docs/rules?hl=de>