Meine App

SalahTime

Version 1.0.0, 4. Juli 2023 | Ensar Korkmaz, Rafaa El Sherkasi

Bbc
Berufs
bildungs
center

Inhalt

1 Abstract (Kurzbeschreibung)

Die SalahTime-App ist eine benutzerfreundliche Android Applikation, die muslimischen Benutzern hilft, ihre Gebetszeiten pünktlich einzuhalten. Die App verwendet die aladhan.com Prayer Times API, um genaue Gebetszeiten für verschiedene Standorte abzurufen. Benutzer können ihren Standort angeben und erhalten die exakten Gebetszeiten für ihren spezifischen Ort. Die App sendet Benachrichtigungen, um Benutzer rechtzeitig an bevorstehende Gebetszeiten zu erinnern. Sie bietet eine einfache Möglichkeit, die individuellen Einstellungen anzupassen, um die Benachrichtigungen nach Bedarf anzupassen. Zusätzlich zur Anzeige der Gebetszeiten enthält die App einen Kompass, der die Qibla-Richtung anzeigt, um den Benutzern bei der Ausrichtung ihrer Gebete in Richtung Mekka zu helfen.

2 Konkurrenzanalyse

Die Konkurrenzanalyse für Apps zur Verfolgung von muslimischen Gebetszeiten und der Ausrichtung der Qibla-Richtung zeigt, dass es mehrere beliebte Apps auf dem Markt gibt. Hier sind einige wichtige Konkurrenten:

1. Muslim Pro: Eine weit verbreitete App, die genaue Gebetszeiten basierend auf dem Standort des Benutzers bietet. Sie verfügt über eine Qibla-Kompassfunktion, die die Ausrichtung zum Gebet in Richtung Mekka anzeigt. Muslim Pro bietet auch Quran-Texte, Audio-Rezitationen und soziale Funktionen. Im Vergleich zu anderen bekannten Gebetszeiten-Apps ist Muslim Pro eine kostenfreie Applikation die eine Vielzahl von Funktionen bietet.

2. Muslim Assistant: Eine umfassende Gebetszeiten-App, die verschiedene Berechnungsmethoden unterstützt. Sie enthält einen Qibla-Kompass und bietet auch Zugang zu Quran-Texten und Audio-Rezitationen. Muslim Assistant ermöglicht Benutzern ausserdem das Teilen von Gebetszeiten und das Verbinden mit anderen Muslimen. Doch kostet einmalig CHF 35.-

3. Salaat First: Eine App, die Gebetszeiten für verschiedene Orte bereitstellt und über einen Qibla-Kompass verfügt. Sie bietet auch eine automatische Standortbestimmung und personalisierbare Benachrichtigungseinstellungen für Gebetszeiten.

Die Muslim Gebetszeiten- und Qibla-Apps zielen darauf ab, eine benutzerfreundliche und zuverlässige Lösung für muslimische Benutzer bereitzustellen. Sie bieten genaue Gebetszeiten basierend auf dem Standort des Benutzers und eine klare Anzeige der Qibla-Richtung. Die Apps konzentrieren sich auf die Kernfunktionen, um eine einfache und effektive Nutzung zu gewährleisten und muslimischen Benutzern bei der Erfüllung ihrer religiösen Pflichten zu unterstützen.

Im Vergleich zu diesen Gebetszeiten Apps bietet unsere benutzerfreundliche App eine leichte und verständliche Bedienung auf dem ersten Blick. Ausserdem ist SalahTime völlig kostenfrei und enthält keine Werbungen. Als Entwickler der SalahTime-App liegt unser Fokus darauf, eine positive Benutzererfahrung zu schaffen.

3 User Stories

User Story 1:

Als Muslimin möchte ich die SalahTime App verwenden, um die genauen Gebetszeiten basierend auf meinem aktuellen Standort zu erhalten. Somit kann ich sicherstellen, dass ich meine Gebete pünktlich und korrekt ausführe. Ausserdem finde ich es toll, dass ich auf der Hauptseite die Zeit bis zur nächsten Gebetszeit sehen kann.

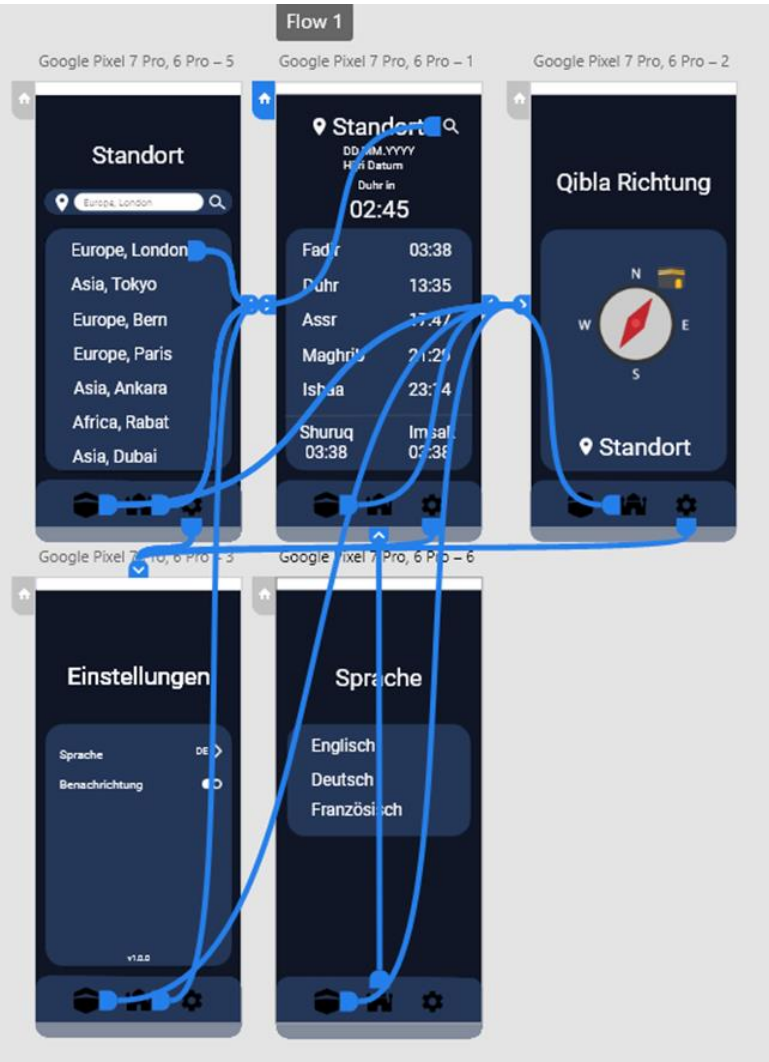
User Story 2:

Als Muslim möchte ich den Qibla-Kompass in der SalahTime App nutzen, um die genaue Ausrichtung zur Kaaba in Mekka zu finden. Somit kann ich mein Gebet in die richtige Richtung auszuführen, egal wo ich mich befinde. Auch wenn ich Zuhause, draussen oder in den Ferien bin kann ich immer zuversichtlich die SalahTime App nutzen um die richtige Richtung zur Kaaba zu finden.

User Story 3:

Als Muslim möchte ich Benachrichtigungen für die Gebetszeiten in der SalahTime App einstellen können, um rechtzeitig benachrichtigt zu werden und keine Gebete zu verpassen. Ausserdem bietet mir die SalahTime App die Auswahl an verschiedene Sprachen, sodass ich dort meine Sprache auswählen kann, die ich gerne möchte. Es bietet sich eine Auswahl zwischen Englisch, Deutsch und Französisch an.

4 Mockups



1. **Startactivity**

Auf unserer Startseite können Benutzer die fünf Gebetszeiten, den Imsak und den Sonnenaufgang sehen. Zusätzlich wird der Standort angezeigt durch Auswahl des Benutzers über das Suchsymbol. In der unteren Navigationsleiste können Benutzer entweder zur Qibla-Richtung oder zu den Einstellungen navigieren. Das Designkonzept betont die einfache Platzierung der Informationen, während das Suchsymbol und die Navigationsoptionen in der unteren Navigationsleiste leicht zugänglich sind.

2. **Standort**

Auf der Standortauswahlseite hast du die Möglichkeit, entweder aus einer Liste von vorgeschlagenen Städten auszuwählen oder deine gewünschte Stadt in der Suchleiste einzugeben. Zusätzlich findest du auch auf dieser Seite eine Navbar am unteren Rand des Bildschirms. Wenn du auf das Moschee-Symbol in der Navbar klickst, gelangst du zurück zur Startseite. Diese Funktion ermöglicht es dir, den Standort bequem zu wählen und zur Hauptansicht zurückzukehren.

3. **Qibla richtung**

Auf der Qibla-Richtungsseite kannst du die genaue Ausrichtung zur Kaaba in Mekka bestimmen. Die Seite ist speziell dafür konzipiert, Muslimen dabei zu helfen, die Qibla-Richtung für das Gebet zu finden.

Die Hauptfunktion der Seite ist ein Kompass, der die Richtung zur Kaaba anzeigt. Du kannst dein Mobilgerät oder Tablet verwenden, um den Kompass auszurichten und die Pfeilrichtung zur Kaaba zu verfolgen. Dadurch kannst du dich in der richtigen Richtung ausrichten, um dein Gebet in Mekka zu verrichten.

4. **Einstellungen**

Um die Sprache anzupassen, kannst du einfach auf die entsprechende Option klicken. Dadurch gelangst du zu einem neuen Fenster, in dem du die gewünschte Sprache auswählen kannst. Durch die Änderung der Sprache wird die gesamte App in der ausgewählten Sprache angezeigt.

Die Benachrichtigungseinstellungen ermöglichen es dir, zu kontrollieren, ob du Benachrichtigungen von der App erhalten möchtest oder nicht. Mit einem praktischen Toggle-Schalter kannst du die Benachrichtigungen aktivieren oder deaktivieren, je nach deinen Vorlieben und Bedürfnissen.

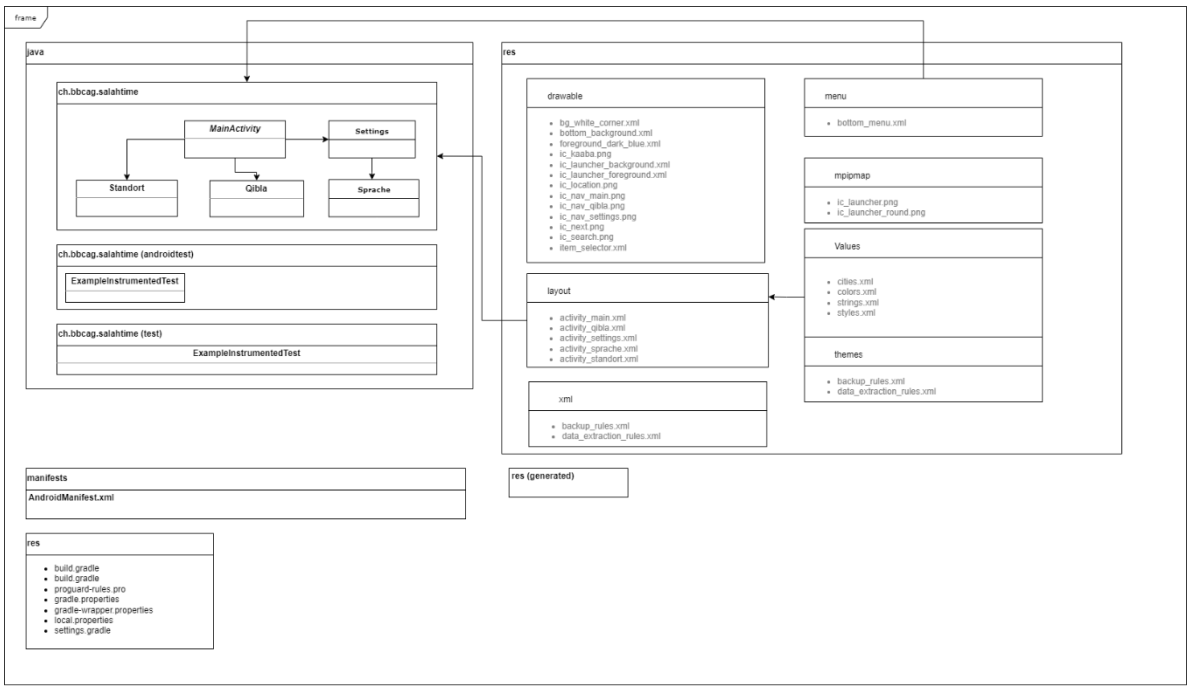
5. **Sprache**

   Auf der Sprachseite hast du die Möglichkeit, zwischen den drei verfügbaren Sprachen zu wählen: Deutsch, Französisch oder Englisch. Du kannst einfach auf die gewünschte Sprache klicken, um sie auszuwählen.

   Wenn du die Sprache änderst, wird die gesamte App in der ausgewählten Sprache angezeigt, einschliessßlich aller Texte und Benutzeroberflächenlemente.

   Um zur vorherigen Seite zurückzukehren, steht dir die Navigationsleiste unten zur Verfügung. Dort findest du eine Option, um zur vorherigen Seite oder zur Startseite zurückzukehren.

5 Technische Realisierung



In diesem Projekt gibt es mehrere Klassen und XML-Dateien, die zusammenarbeiten, um die Funktionalität der Anwendung zu implementieren. Die Beziehungen zwischen den Klassen und den XML-Dateien lassen sich wie folgt beschreiben:

1. MainActivity.java und activity_main.xml:

- Die Klasse MainActivity ist mit der XML-Datei activity_main.xml verknüpft.

- Die XML-Datei activity_main.xml definiert das Layout der Hauptaktivität, einschließlich einer ImageButton, TextViews und einem BottomNavigationView.

- Die Klasse MainActivity enthält den Code, der auf Benutzerinteraktionen in der Hauptaktivität reagiert und die entsprechenden Aktionen ausführt.

2. Qibla.java und activity_qibla.xml:

- Die Klasse Qibla ist mit der XML-Datei activity_qibla.xml verknüpft.

- Die XML-Datei activity_qibla.xml definiert das Layout der Qibla-Aktivität, einschließlich eines TextViews, eines ConstraintLayouts mit einer ImageView und Views sowie eines BottomNavigationView.

- Die Klasse Qibla enthält den Code, der die Qibla-Richtung basierend auf Sensordaten und Standortinformationen berechnet und das Layout entsprechend aktualisiert.

3. Settings.java und activity_settings.xml:

- Die Klasse Settings ist mit der XML-Datei activity_settings.xml verknüpft.

- Die XML-Datei activity_settings.xml definiert das Layout der Einstellungsaktivität, einschließlich eines ImageButtons und eines BottomNavigationView.

- Die Klasse Settings enthält den Code, der auf Benutzerinteraktionen in der Einstellungsaktivität reagiert und die entsprechenden Aktionen ausführt.

4. Sprache.java und activity_sprache.xml:

- Die Klasse Sprache ist mit der XML-Datei activity_sprache.xml verknüpft.

- Die XML-Datei activity_sprache.xml definiert das Layout der Sprache-Aktivität.

- Die Klasse Sprache enthält den Code, der auf Benutzerinteraktionen in der Sprache-Aktivität reagiert und die entsprechenden Aktionen ausführt.

5. Standort.java und activity_standort.xml:

- Die Klasse Standort ist mit der XML-Datei activity_standort.xml verknüpft.

- Die XML-Datei activity_standort.xml definiert das Layout der Standort-Aktivität, einschließlich eines AutoCompleteTextViews und eines Buttons.

- Die Klasse Standort enthält den Code, der auf Benutzerinteraktionen in der Standort-Aktivität reagiert und die entsprechenden Aktionen ausführt.

5.1 Qibla Richtung anzeigen

Wir haben zwei komplexe Komponenten hinzugefügt. Die erste ist ein Kompass, der die Richtung zur Kaaba in Mekka anzeigt, die Gebetsrichtung der Muslime. Dies ist für uns wichtig, da wir oft unterwegs beten oder keine Moschee in der Nähe haben und daher die Gebetsrichtung bestimmen müssen. Der Kompass wurde so implementiert, dass ein Punkt in der Mitte oben definiert ist, der den Pfeil in die richtige Richtung zeigt. Dadurch bleibt das Bild des Kompasses stabil, während sich nur der Pfeil bewegt, um die Drehung anzuzeigen.

Die Berechnung der Qibla-Richtung erfolgt folgendermaßen: Zuerst erhält die App die Berechtigung, den Standort des Benutzers abzurufen. Anschließend werden die Längen- und Breitengrade des Standorts mithilfe der Methode `getDeviceLocation()` festgelegt. Die Koordinaten der Kaaba in Mekka, also der Längengrad (`kaabaLongitude`) und der Breitengrad (`kaabaLatitude`), werden ebenfalls definiert.

Die Qibla-Richtung wird mit Hilfe von trigonometrischen Funktionen berechnet. Dabei werden der Breitengrad des Benutzers, der Breitengrad der Kaaba und der Längengradunterschied verwendet. Die berechnete Richtung wird auf einen Bereich von 0-359 Grad normalisiert, falls sie negativ ist.

In der Methode `onLocationChanged()` wird die Methode `getQiblaDirection()` aufgerufen und die berechnete Qibla-Richtung angezeigt.

Der Kompass benutzt einen speziellen Sensor in deinem Smartphone, um herauszufinden, in welche Richtung das Gerät zeigt. Der Sensor verwendet Informationen von anderen Sensoren im Smartphone, wie zum Beispiel dem Bewegungssensor und dem Magnetfeldsensor, um die genaue Ausrichtung des Geräts im Raum zu bestimmen.

Die App liest kontinuierlich die Daten dieses Sensors aus und berechnet daraus die Kompassrichtung. Diese Richtung wird dann verwendet, um das Kompassbild auf dem Bildschirm entsprechend zu drehen und anzuzeigen. Durch die Nutzung dieses Sensors kann der Kompass genau verfolgen, wie sich das Gerät bewegt, und dir immer die korrekte Richtung anzeigen.

5.2 SalahPrayer API

Die zweite komplexe Komponente war die SalahPrayer API. Mit der API können wir schliesslich die Gebetszeiten der jeweilige Standort und Datum ausgelesen bekommen. Um eine http Get Request machen zu können haben wir allerdings eine vorgefertigte Class von Android Studio namens «volley» benutzt. Diese konnten wir sehr leicht im build.grade file implementieren:

```
implementation 'com.android.volley:volley:1.2.1'
```

Nun konnten wir im MainActivity Class eine Mehode namens OnResponse machen und darin einen JsonObjectRequest über unseren Api url machen. Die API url ist so aufgebaut, dass man das Jahr und Monat beliebig im Url ändern kann. Sobald man diese geändert hat werden alle Tage des Monats als Json array angezeigt. Hier ein Beispiel Api Url →

https://api.aladhan.com/v1/calendarByCity/2023/7?city=Bern &country=&method=2

Doch nun möchten wir das Datum so ändern, wann der Nutzer eine Gebetszeiten Suche nach dem beliebigen Standort macht. Dieses konnten wir dank dem «Calendar» Class von Android Studio lösen. Somit haben wir folgende Code geschrieben.

```
Calendar calendar = Calendar.getInstance();
int year = calendar.get(Calendar.YEAR);
int month = calendar.get(Calendar.MONTH) + 1;
```

→

```
url = "https://api.aladhan.com/v1/calendarByCity/" + year + "/" + month + "?city=" + enteredText + "&country=&method=2";
```

Schlussendlich haben wir im OnResponse Methode dann einen Try Catch versucht. In diesem Code haben wir eine Methode namens "onResponse". Diese Methode wird automatisch aufgerufen, wenn wir eine JSON-Antwort erhalten. Wir haben diese Methode erstellt, um die Gebetszeiten aus der JSON-Antwort zu extrahieren und sie anzuzeigen. Zuerst überprüfen wir, ob die Antwort ein Array namens "data" enthält. Dieses Array enthält Informationen zu den Gebetszeiten für verschiedene Tage.

Sobald wir das JSON-Objekt für den aktuellen Tag haben, greifen wir auf die einzelnen Gebetszeiten zu, indem wir die entsprechenden Schlüssel verwenden, wie "Fajr", "Dhuhr", "Asr", usw. Jede Zeit speichern wir in einer separaten Variablen.

Zum Schluss kombinieren wir die bereinigten Gebetszeiten und ihre Namen in separaten Zeichenketten. Wir fügten Zeilenumbrüche hinzu, um die Lesbarkeit zu verbessern.

```
String allPrayerTimes = fajr + "\n\n" + dhuhr + "\n\n"  + asr + "\n\n" + maghrib + "\n\n" + isha + "\n\n\nImsak\n" + imsak;
String allPrayerNames = "Fajr\n\n" + "Dhuhr\n\n" + "Asr\n\n" + "Maghrib\n\n" + "Isha\n\n\n" + "Shuruq\n" + shuruq;
gebetszeiten.setText(allPrayerTimes);
gebetsnamen.setText(allPrayerNames);
```

Und im catch geben wir einen Stacktrace aus, falls die Api nicht gefetch werden konnte.

Mittels der Volley Class wird die Api Request gemacht und somit kann der vorherige code mit den Zeichenketten im App ausgegeben werden.

```
Volley.newRequestQueue(this).add(request); //
```

6 Testing

6.1 Manuelle UI-Tests

Abschnitt	Inhalt
ID	ST-01
Anforderungen	Gebetsrichtung angezeigt
Vorbedingungen	• Die SalahTime-App ist installiert und geöffnet. • Der Benutzer hat den Zugriff auf den Rotation Vector Sensor gewährt. • Der Benutzer hat die Standortberechtigung erteilt. • Der Benutzer befindet sich an einem Ort mit einer Internetverbindung.
Ablauf	• Der Benutzer öffnet die SalahTime-App und navigiert zum Qibla-Fenster. • Der Benutzer überprüft, ob die Gebetsrichtung angezeigt wird. • Der Benutzer vergleicht die angezeigte Gebetsrichtung mit der tatsächlichen Ausrichtung zur Kaaba in Mekka.
Erwartetes Resultat	Das Kompass sollte sich drehen und immer in die gleiche Richtung zeigen.

Abschnitt	Inhalt
ID	ST-02
Anforderungen	• Überprüfung der Reaktion des Kompasses auf Gerätebewegungen
Vorbedingungen	• Die SalahTime-App ist installiert und geöffnet. • Der Benutzer hat den Zugriff auf den Rotation Vector Sensor gewährt. • Der Benutzer hat die Standortberechtigung erteilt. • Der Benutzer befindet sich an einem Ort mit einer Internetverbindung.
Ablauf	• Der Benutzer öffnet die SalahTime-App und navigiert zum Qibla-Fenster. • Der Benutzer überprüft, ob der Kompass angezeigt wird. • Der Benutzer dreht das Gerät um 180 Grad und überprüft, ob sich der Kompass entsprechend der Ausrichtung zur Kaaba dreht. • Der Benutzer dreht das Gerät um weitere 90 Grad nach links oder rechts und überprüft erneut die Ausrichtung des Kompasses zur Kaaba.

	• Der Benutzer wiederholt Schritt 3 und 4 mehrmals, um sicherzustellen, dass der Kompass korrekt auf Gerätebewegungen reagiert. •
Erwartetes Resultat	• Der Kompass sollte sich entsprechend der Ausrichtung zur Kaaba drehen, wenn das Gerät gedreht wird.

Abschnitt	**Inhalt**
ID	ST-03
Anforderungen	• API fetch von Gebetszeiten nach Standort und Datum
Vorbedingungen	• Die SalahTime-App ist installiert und geöffnet. • Der Benutzer befindet sich an einem Ort mit einer Internetverbindung.
Ablauf	• Der Benutzer öffnet die SalahTime-App. • Der Benutzer navigiert zum Standort-suche Seite. • Der Benutzer gibt einen Standort seiner Wahl aus. • Der Benutzer bestätigt die Suche.
Erwartetes Resultat	• Die SalahTime-App sollte nach der Standort Such bestätigung auf die Hauptseite übergehen und die 5 Gebetszeiten des gesuchten Standortes (Stadt) aufgelistet anzeigen.

6.2 Testauswertung

Zusammenfassung aller durchgeführten Tests. Nur fehlgeschlagene Tests und Tests mit Bemerkungen müssen in der folgenden Tabelle aufgelistet werden.

ID	Erfolgreich	Bemerkungen
ST-01	Ja	Der Testfall war erfolgreich, der Testperson 1 konnte den Kompass umdrehen und es wurde in die Richtung gegen Mekka angezeigt.
ST-02	Ja	Die Testperson 2 konnte erfolgreich den Qibla-Kompass in der SalahTime-App testen. Bei der Drehung des Geräts um 180 Grad hat sich der Kompass entsprechend der Ausrichtung zur Kaaba gedreht. Auch bei weiteren Drehungen um 90 Grad nach links oder rechts hat sich der Kompass korrekt zur Kaaba ausgerichtet. Die Testperson 2 hat den Test mehrmals wiederholt, um sicherzustellen, dass der Kompass zuverlässig auf Gerätebewegungen reagiert. Bei schneller bewegung des Kompass dauerte es aber einwenig bis es sich in die richtige Richtung positioniert hat.
ST-03	Ja	Die Testperson 3 konnte auch hier erfolgreich die Funktionalität der API-Fetch von Gebetszeiten nach Standort und Datum testen. Nachdem die Testperson 3 die Standortsuche bestätigt hat, wurde sie zur Hauptseite weitergeleitet, und dort wurden die 5 Gebetszeiten für den gesuchten Standort angezeigt. Die Testperson 3 hat überprüft, ob die angezeigten Gebetszeiten korrekt und aktuell sind.

7 Fazit

Trotz einiger anfänglicher Schwierigkeiten mit Git lief die Zusammenarbeit gut. Wir konnten die gestellten Aufgaben problemlos erledigen und hatten keine größeren Probleme beim Design und der Arbeit mit Android Studio.

Die Herausforderung bestand vor allem darin, alles innerhalb des gegebenen Zeitrahmens fertigzustellen. Wir hatten uns viele Ziele gesetzt und einige komplexere Aufgaben standen an. Die Integration von Funktionen wie einem Kompass und einer API war nicht einfach und erforderte viel Zeit. Zudem stellte uns die Programmiersprache Java vor einige Schwierigkeiten, da wir sie nicht perfekt beherrschten.

Trotzdem sind wir mit dem Endergebnis ziemlich zufrieden, da wir die schwierigsten Aufgaben gemeistert haben und die meisten Funktionen so funktionieren, wie wir es uns vorgestellt haben. Wir haben unsere Kenntnisse in Java verbessert und sind auch besser geworden, wenn es darum geht, Daten abzurufen (fetchen). Ausserdem haben wir gelernt, wie eine Android-App aufgebaut ist und konnten uns gut in die Entwicklung für mobile Apps einarbeiten. Des Weiteren haben wir gelernt, wie Berechtigungen verwaltet werden. Wir haben auch gelernt, wie wir Sensoren verwenden können. Und wie diese Sensoren Funktionieren.

Es gibt zwar noch einige einfachere Aufgaben, die wir aus Zeitgründen nicht umsetzen konnten, wie zum Beispiel das Anzeigen des Datums im Hijri-Kalender und das Erstellen einer Liste von Städten in der API, die auf der Standortsuche-Seite angezeigt werden sollen. Trotzdem sind wir stolz auf das, was wir erreicht haben, und freuen uns über das Ergebnis unserer Arbeit.

- Was lief gut/schlecht?
- Wie seid ihr mit dem Endergebnis zufrieden?
- Was habt ihr gelernt?
- War alles vorhanden oder was fehlte noch?
- Usw.