Duale Hochschule Baden-Württemberg

Stuttgart Campus Horb

Geogram - Dokumentation

T3INF4310 - Entwicklung mobiler Applikationen

---

eingereicht von:	Benita Dietrich, Paul Finkbeiner, Josua Stricker, Jonas Schwarz, Sven Stoll und Moris Kotsch
Modul:	T3INF4310 - Entwicklung mobiler Applikationen
Dozent:	B.Sc. Torsten Hopf
Kurs:	TINF2018
Studiengang:	Informatik
Hochschule:	DHBW Stuttgart Campus Horb
Bearbeitungszeitraum:	08.03.2021 - 06.06.2021

Horb am Neckar, 21. April 2021

Inhaltsverzeichnis

Abbildungsverzeichnis

Kapitel 1

1 Einführung

Nachfolgend wird eine Einleitung in die hier vorliegende Projektdokumentation gegeben. Neben einer vorgegebenen Aufgabenstellung und selbstdefinierter Kernidee, wird eine Auflistung aller Projektteilnehmer aufgezeigt.

1.1 Aufgabenstellung

Zieles dieses Projektes ist die Konzeption und Implementierung einer mobilen App. Jenes Projekt findet im Rahmen der Vorlesung „Entwicklung mobiler Applikationen“ statt.

- Der Use Case sollte kein Spiel sein, sondern eher eine klassische App abbilden.
- Der Use Case soll mindestens von zwei der zur Verfügung stehenden Funktionen eines Mobilgeräts nutzen. Beispielsweise
  - GPS-Sensor
  - Neigungssensor
  - Kamera
  - Fingerabdruck
  - NFC
- Eine FSK Einstufung der Applikation sollte nicht nötig sein.

1.2 Projektidee

Als Gruppe wurde eine Kernidee für die Erfüllung der Projektanforderungen entworfen. Die mobile Anwendung orientiert sich an der bestehenden mobilen Anwendung „Instagram“.

Ziel ist eine Plattform für das Teilen von Bildern und Entdecken von Content, welche sich in der Nähe des Benutzers befinden. Vergleichbar zu Instagram sollen relevante Beiträge in Feeds dargestellt werden. Der große Unterschied zu Instagram ist der Fokus auf die geografischen Hintergrundinformationen der Beiträge. So werden den Benutzern

lediglich die Beiträge von anderen Benutzern angezeigt, welche sich in einer festgelegten geografischen Entfernung befinden. Durch die Einbindung und Verwendung von geografischen Informationen, soll die Vermarktung von lokalen Angeboten, Aktivitäten und Sehenswürdigkeiten erleichtert werden.

Durch die große Ähnlichkeit mit Instagram, und der Erweiterung um die Nutzung von GPS-Informationen, für die Darstellung von ortsnahen Feed-Beiträgen, wurde sich einheitlich für den Projektnamen „Geogram“ entschieden.

Wie in der Aufgabenstellung (Abschnitt 1.1) gefordert, beinhaltet die mobile Anwendung „Geogram“ zwei Funktionen von Mobilgeräten. Verwendet wird unter anderem die **Kamera** und **GPS-Sensorik** von heutigen Mobilgeräten.

1.3 Projektteilnehmer

Das Projekt wird von sechs Student*innen bearbeitet. Zusätzlich wird das Projekt während der kompletten Laufzeit von einem Stakeholder (Dozent des Moduls) betreut.

Rolle	Name	Kontakt
Mitglied	Benita Dietrich	i18008@hb.dhbw-stuttgart.de
Mitglied	Paul Finkbeiner	i18011@hb.dhbw-stuttgart.de
Mitglied	Josua Stricker	i18039@hb.dhbw-stuttgart.de
Mitglied	Jonas Schwarz	i18037@hb.dhbw-stuttgart.de
Mitglied	Sven Stoll	i18038@hb.dhbw-stuttgart.de
Mitglied	Moris Kotsch	i18021@hb.dhbw-stuttgart.de
Stakeholder	Torsten Hopf	torsten.hopf@mhp.com

2 Kapitel 2
Konzeption

Während der Konzeptionsphase ging es darum einen ersten Entwurf für die spätere Implementierung zu entwerfen. Hierfür wurden verschiedene Mockups erstellt, welche einen ersten Einblick in die Designvorstellungen für die mobile Anwendung „Geogram“ geben sollen.

Mit dem Definieren der MVP-Kriterien, wurde der Funktionsumfang, für die erste minimal funktionsfähige Iteration der mobile Anwendung „Geogram“ aufgelistet. Zusätzlich wurden noch weitere Soll- und Kann-Kriterien festgelegt.

2.1 Mockup’s

In Abbildung 2.1 und Abbildung 2.2 ist die Login-Ansicht abgebildet. Bereits als Mockup realisiert, ist der Wechsel zwischen einem White- und Dark-Mode der mobilen Anwendung.

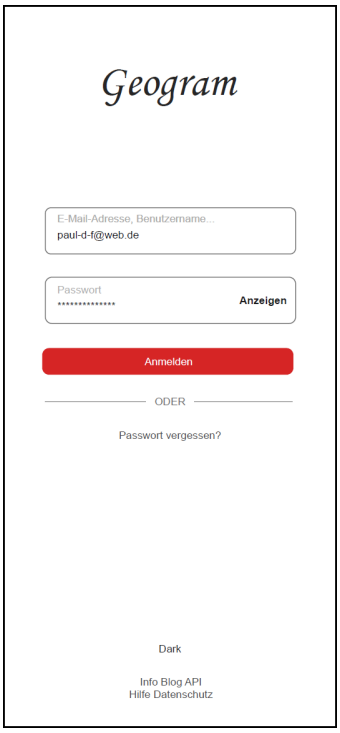

Abb. 2.1: Login-Ansicht in weiß

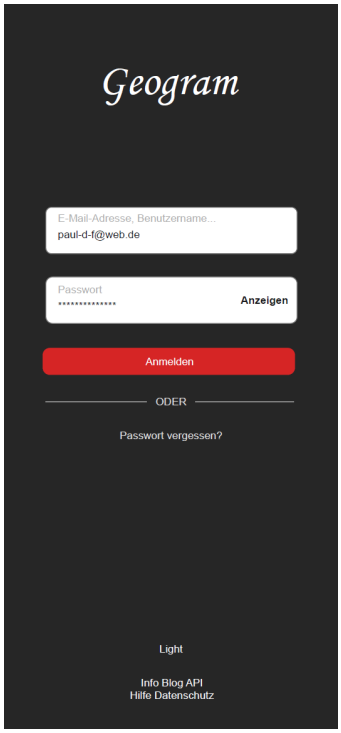

Abb. 2.2: Login-Ansicht in schwarz

In Abbildung 2.3 und Abbildung 2.4 sind die Feed-Ansichten abgebildet. Die Feed-Übersicht ähnelt der Feed-Übersicht von Instagram, jedoch mit dem Unterschied, dass mithilfe einer Standortangabe die Filterung der Feeds vollzogen wird.

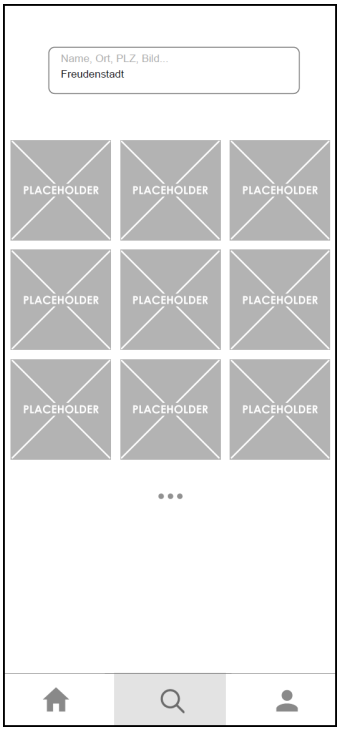

Abb. 2.3: Feed Overview

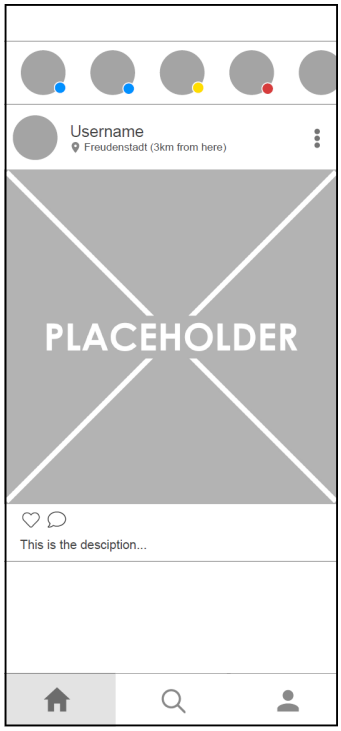

Abb. 2.4: Feed Detail

Die folgenden zwei Abbildungen zeigen die Profilansicht mit Änderungsmöglichkeit.

Abb. 2.5: Eigene Profil-Übersicht

Abb. 2.6: Eigenes Profil bearbeiten

Der Vorgang beim Hochladen eines neuen Feeds wird in Abbildung 2.7 und Abbildung 2.8 visualisiert. Der Benutzer hat die Möglichkeit das Foto entweder aus der eigenen Galerie auszuwählen oder mit der Kamera aufzunehmen. Noch während dem Hochladevorgang kann der Benutzer dem Feed eine Beschreibung und die GPS-Informationen für die Standortangabe hinzufügen.

Abb. 2.7: PopUp-Fenster für neuen Feed

Abb. 2.8: Neuen Feed bearbeiten

2.2 MVP-Kriterien

Folgende Kriterien müssen für eine erste, minimal funktionsfähige Iteration der mobilen Anwendung „Geogram“, während der Implementierung umgesetzt werden:

- Login/Nutzerverwaltung
- Upload-Funktion für die Bilder mit Standortdaten
- Zugriff auf die Kamerafunktion des Mobilgerätes
- Zugriff auf die GPS-Daten (GPS-Modul des Mobilgerätes)
- Datenbank für Benutzerkonten, Standortdaten und Bildern
- Anzeigen der Beiträge

2.3 Soll- und Kann-Kriterien

Neben den relevanten MVP-Kriterien wird die Konzeption noch mit Soll- und Kann-Kriterien ergänzt.

Soll-Kriterien:

- Löschen von Bildern bzw. Beiträgen
- Hinzufügen von Bildbeschreibungen
- Ansicht des eigenen Kontos
- Einbindung von Google Maps für die Standortdaten

Kann-Kriterien:

- Anmelden per Fingerabdruckssensor
- Anderen Benutzer „folgen“ („folgen“-Funktion)
- Benutzerdefinierte Filterfunktion für Beiträge
- Kommentare, Likes und Hastags für die Beiträge
- Dark-Mode abhängig von Systemeinstellungen

Kapitel 3
3 Architektur

Für die Architektur der mobilen Anwendung „Geogram“, wurden die zwei Frameworks „Ionic“ und „Express“ und die Entwicklungs-Plattform „Firebase“ verwendet.

Wie in Abbildung 3.1 veranschaulicht, wird das Web-Framework „Ionic“ für das Frontend verwendet. Das Backend kann in zwei Bereiche unterteilt werden. Zum einen wird die Entwicklungs-Plattform „Firebase“ und das Node.js Web-Framework „Express“ verwendet. Genauere Informationen bezüglich der Aufteilung des Backends wird in Abschnitt 3.2 beschrieben.

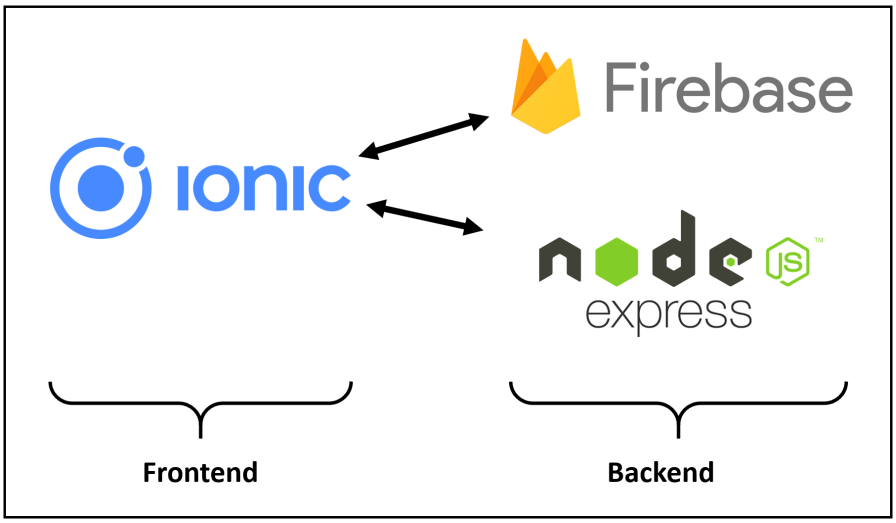

Abb. 3.1: Technologieübersicht

3.1 Frontend

Inhalt

3.2 Backend

Den Großteil der Backend-Aufgaben übernimmt die Entwicklungs-Platform Firebase. Verwendet wird hierfür die kostenlose „Spark Plan“-Version von Firebase.

Die **Authentifizierung** und das **Credentialmanagent** wird hierbei vollständig von Firebase übernommen und verwaltet. Über vordefinierte Programmierschnittstellen kann man anschließend auf die Funktionen von Firebase zugreifen. Ebenso wird im kostenlosen Lizenzmodell „Spark Plan“ eine Cloud-Speicherung namens „**Firestore**“ angeboten. In der darin enthaltenen **Echtzeitdatenbank** stehen jedoch nur 1 GiB Speicherplatz zur Verfügung. Da sich das Kernkonzept von Geogram hauptsächlich um Fotos dreht und die finanziellen Mittel der Gruppe nicht für ein kostenpflichtiges Lizenzmodell ausreichen, wurde zusätzlich ein **Foto-Server** mithilfe des Web-Frameworks „Express“ implementiert. Der Foto-Server wird von der Projekt-Gruppe selbst gehostet und beinhaltet mehr als 1 Gib Speicherplatz.

3.2.1 Credentialmanagent

Die Verwaltung von Anmeldeinformationen wird in Geogram von der Entwicklungs-Plattform Firebase übernommen. Firebase bietet verschiedene Sign-in Methoden an. Neben der klassischen E-Mail/Passwort Kombination stellt Firebase noch die Sign-in Methoden von anderen Anbietern zur Verfügung. So kann man die Benutzerkonten von zum Beispiel Google, Facebook, Twitter oder GitHub für die Verwendung verwenden.

Für die mobile Anwendung Geogram wird die klassische Sign-in Methode „E-Mail und Passwort“ verwendet.

3.2.2 Cloud Firestore

Die verwendete Firestore-Datenbank ist eine in der Cloud gehostete NoSQL-Datenbank. Über native SDKs können die iOS-, Android- und Web-Apps auf Firestore zugreifen.

In dem NoSQL-Datenmodell von Firestore, werden die Daten in Dokumente gespeichert, welche Felder enthalten, denen Werte zugeordnet sind. Diese Dokumente wiederrum werden in Sammlungen (collections) abgespeichert. Diese hierarchische Rangordnung der Datenspeicherung ist nochmals in Abbildung 3.2 abgebildet.

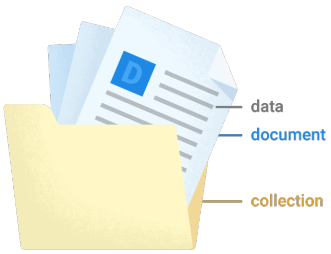

Abb. 3.2: Speicherstruktur Firestore

Für Geogram werden die zwei Sammlungen „images“ und „users“ benötigt. Wie die Bezeichnungen schon vermuten lassen, werden darin die Fotos und die Benutzer von Geogram abgespeichert.

Die Dokumente der Sammlung „users“ bilden die verschiedenen Benutzerprofile von Geogram ab. Ein Benutzerprofil wird durch folgende Felder definiert:

- **biography**: Eine kurze Beschreibung über den Benutzer
- **email**: Die E-Mail Adresse des Benutzers
- **profilepic**: URL zu dem Profilbild, welches im Foto-Server abgespeichert ist
- **userFirstName**: Vorname des Benutzers
- **userLastName**: Nachname des Benutzers
- **username**: Username des Benutzers

In Abbildung 3.3 ist ein beispielhaftes users-Dokument abgebildet.

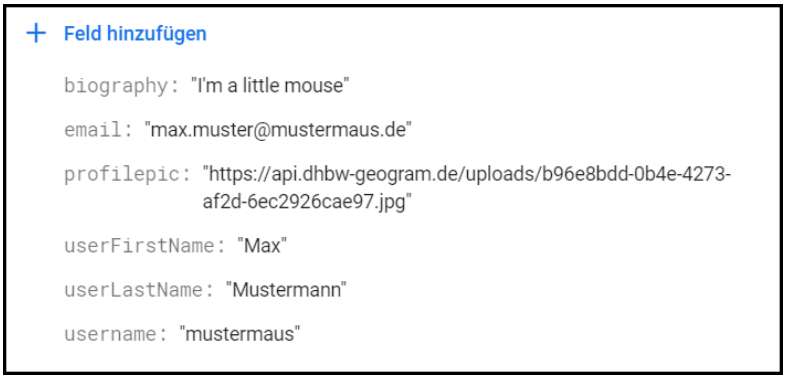

Abb. 3.3: Felder eines „users“-Dokumentes

Die Dokumente der Sammlung „images“ bilden die verschiedenen Bilder von Geogram ab. Ein Bild wird durch folgende Felder definiert:

- **description**: Beschreibung des Bildes
- **id**: Eindeutige Id für das Bild
- **location**: GPS-Informationen für das Bild
- **timestamp**: Zeitpunkt des Hochladens des Bildes
- **title**: Titel des Bildes
- **url**: URL zu dem Bild, welches im Foto-Server abgespeichert ist
- **user**: Username des Bild-Inhabers

In Abbildung 3.4 ist ein beispielhaftes images-Dokument abgebildet.

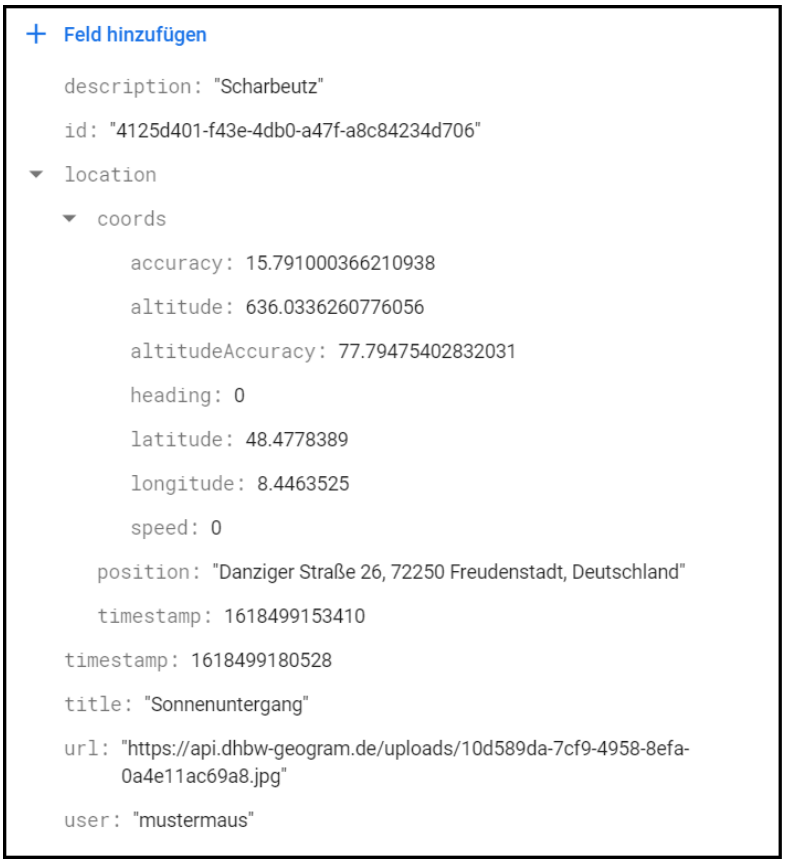

Abb. 3.4: Felder eines „images“-Dokumentes

3.2.3 Foto-Server

Um das kostenpflichtige Lizenzmodell von Firebase zu umgehen, wurde ein eigener Foto-Server bereitsgestellt. Aufgrund des begrenzten Speicherplatzes (max. 1 GiB) im Firestore, werden nur noch die URL-Adressen der jeweiligen Bilder im Firestore abgelegt. Zu sehen ist das Abspeichern der URL-Adressen in den Abbildungen Abbildung 3.3 und Abbildung 3.4. Die URL-Adressen sind alle nach folgendem URL-Schema aufgebaut:

```
https://api.dhbw-geogram.de/uploads/{image_id}.jpg
```

Für die Umsetzung des Foto-Servers wurde sich für das Web-Framework „Express“ entschieden. Es zählt zu den bekanntesten Web-Frameworks, welche in der Node.js Laufzeitumgebung beheimatet ist.

Für die Verarbeitung von eingehenden Web-Anfragen, verwendet Express eine Art Pipeline-Mechanismus. In Listing 3.1 ist ein Ausschnitt aus dem Quellcode des Foto-Servers abgebildet. Im Code wird hierfür zunächst ein Express-Objekt erstellt. Mit der use-Methode können der Pipeline verschiedene Middlewares hinzugefügt werden. So gibt es zum Beispiel die Möglichkeit, bei der Verarbeitung der Web-Anfrage, einen Logger oder JSON-Parser hinzuzufügen. Die Express-Pipeline orientiert sich am REST-Paradigma. Der Foto-Server hat daher die get- und post-Methoden implementiert.

Mit den beiden Methoden besteht die Möglichkeit ein Foto entweder dem Server hinzuzufügen (post) oder abzufragen (get).

```
1   ...
2
3   var app = express();
4
5   app.use(logger("dev"));
6   app.use(express.json({ limit: "50mb" }));
7   app.use(express.urlencoded({ extended: false, limit: "50mb" }));
8   app.use(cookieParser());
9   app.use(express.static(path.join(__dirname, "public")));
10  app.use(cors());
11
12  app.get("/upload1/:data", (req, res) => {
13      ...
14  )};
15
16  app.post("/upload1", (req, res) => {
17      ...
18  )};
19
20  ...
```

Listing 3.1: Codeausschnitt aus dem Foto-Server

Gehostet und verwaltet wird der Foto-Server, wie beschrieben, von Paul Finkbeiner.

Kapitel 4

4 Funktionalitäten