INTERNATIONALE HOCHSCHULE

Projektbericht

Entwicklung einer Board-Gamer App in Android

Eingereicht am 30.07.2022

Verfasser: Manuel Veigel

E-Mail: [redacted]

Matrikelnummer: [redacted]

Studiengang: B.Sc. Wirtschaftsinformatik

Modul: Projekt Mobile Software Engineering (IWMB02)

Tutor: [redacted]

Source Code Repository: https://github.com/Rakantor/iubh-gamer-app

Inhaltsverzeichnis

I. Abbildungsverzeichnis

II. Abkürzungsverzeichnis

API ..... *Application Programming Interface*

App ..... *Applikation*

aws ..... *Amazon Web Services*

bzw. ..... *beziehungsweise*

DB ..... *Datenbank*

DBMS ..... *Datenbankmanagementsystem*

GB ..... *Gigabyte*

JSON ..... *JavaScript Object Notation*

NoSQL ..... *Not only SQL*

RDBMS ..... *Relationales Datenbankmanagementsystem*

RTDB ..... *Realtime Database*

SDK ..... *Software Development Kit*

SQL ..... *Structured Query Language*

UID ..... *Unique Identifier*

1. Einleitung

Ziel dieser Projektarbeit war die Entwicklung einer Android-App, die es einer Gruppe von Board-Gamern ermöglichen soll, ihre Spieleabende effizienter zu organisieren. Konkret sollten die folgenden Anforderungen erfüllt werden:

- Spieler sollen immer wissen, wann und wo der nächste Spieltermin stattfindet
- Die Gruppe möchte sich immer turnusmäßig bei einem anderen Spieler zu Hause treffen
- Spieler sollen an der Gestaltung des Spielabends teilhaben und Spiele vorschlagen können
- Spieler sollen den Spieleabend, den Gastgeber und das Essen im Anschluss bewerten können
- Spieler sollen sich untereinander Nachrichten zukommen lassen können

2. Konzept

2.1 Grundlagen

Dieses Kapitel befasst sich mit den Fragen und Herausforderungen, die während der Planungsphase aufgetreten sind. Zunächst werden die Problemstellungen definiert, anschließend werden mögliche Lösungsansätze erläutert.

2.1.1 Datenspeicherung

Um den Spielern die Nutzung der App zu jeder Zeit und von überall aus zu ermöglichen, wird ein Server benötigt, der die Anfragen der Spieler verarbeitet und Daten bereitstellt. Die Daten werden in einer Datenbank (DB) gespeichert und mithilfe eines Datenbankmanagementsystems (DBMS) verwaltet. Zunächst stellte sich also die Frage nach dem zu verwendenden DBMS. Der Markt wird deutlich von relationalen Datenbanken, welche auf einem tabellenbasierten relationalen Datenbankmodell beruhen, dominiert.[1] Das zugehörige DBMS wird als relationales Datenbankmanagementsystem (RDBMS) bezeichnet. Zum Abfragen und Verändern der Daten wird überwiegend die Datenbanksprache SQL[2] eingesetzt. Gegenspieler traditioneller relationaler Datenbanken sind unter anderem die NoSQL-Datenbanken, die einen nicht-relationalen Ansatz verfolgen und in den letzten Jahren stetig an Popularität gewinnen konnten.

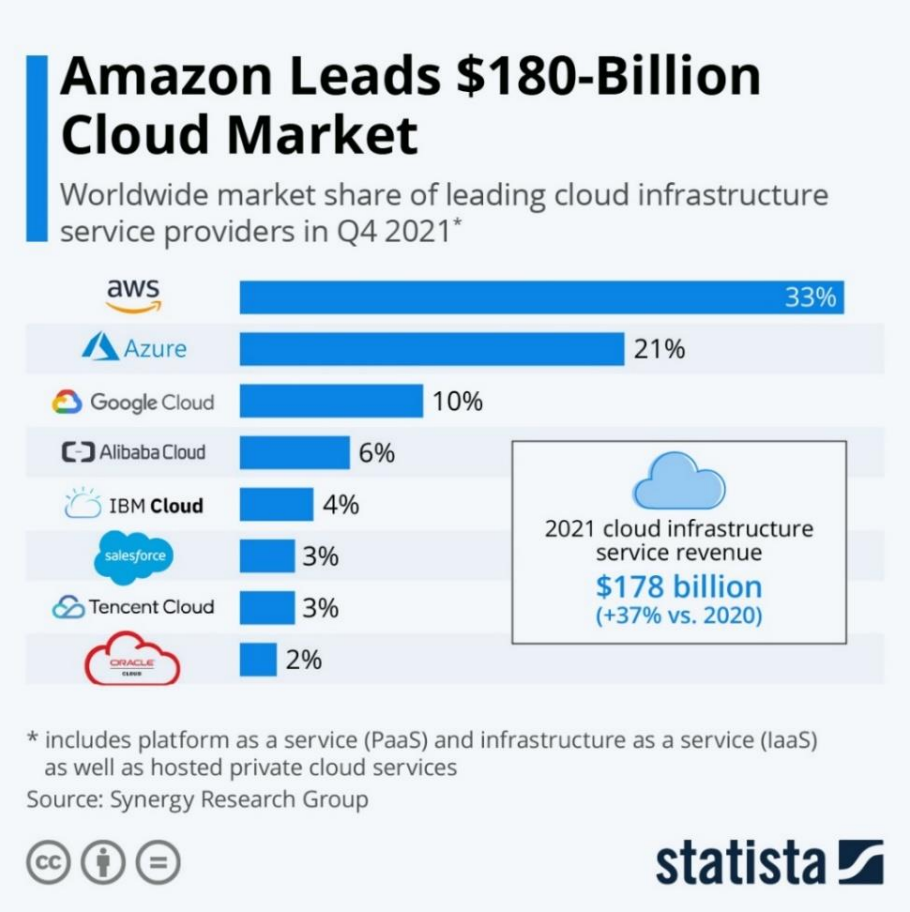
> NoSQL-Datenbanken sind speziell für bestimmte Datenmodelle entwickelt worden und verfügen über flexible Schemata zur Erstellung moderner Anwendungen. NoSQL-Datenbanken sind weithin bekannt für ihre einfache Entwicklung, Funktionalität und Skalierbarkeit. … Diese Typen von Datenbanken sind speziell für Anwendungen optimiert, die große Datenmengen, geringe Latenz sowie flexible Datenmodelle erfordern. (Amazon Web Services, Inc., 2022)

[1] https://db-engines.com/de/ranking_categories
[2] https://de.wikipedia.org/wiki/SQL

Da sich NoSQL-Datenbanken gut für Echtzeit-Anwendungen wie Instant-Messaging Apps eignen und die Implementierung eines Chats Teil dieser Projektarbeit war, erschien es sinnvoll, den NoSQL-Ansatz gegenüber relationalen Datenbanken zu bevorzugen.

Als nächstes galt es die Frage zu beantworten, auf welchem bzw. welcher Art von Server die Datenbank aufgesetzt werden sollte. Der Betrieb eigener Server ist mit enormem organisatorischem Aufwand sowie hohen Anschaffungs- und laufenden Kosten verbunden und erfordert darüber hinaus umfassendes technisches Know-how. Unternehmen aller Größenordnungen sowie Privatpersonen greifen daher zumeist auf Cloud Computing[3] Lösungen zurück. Die Fülle an Cloud-Serviceanbietern und -produkten macht die Auswahl des passenden Anbieters bzw. der passenden Lösung zu einer komplexen Aufgabe. Es ist hilfreich, sich zunächst einen Überblick über die vorherrschende Marktsituation zu verschaffen. Zu den weltweit führenden Cloud-Serviceanbietern zählen, wie in der folgenden Abbildung ersichtlich, Amazon (aws), Microsoft (Azure) und Google (Google Cloud).

Abb. 1 Marktanteile führender Cloud-Serviceanbieter weltweit, Q4 2021



Quelle: Synergy Research Group, 2021. CC BY 3.0.

[3] “Simply put, cloud computing is the delivery of computing services—including servers, storage, databases, networking, software, analytics, and intelligence—over the Internet (“the cloud”) to offer faster innovation, flexible resources, and economies of scale” (Microsoft Corporation, 2022).

Die Services & Produkte dieser Anbieter ähneln sich grundsätzlich hinsichtlich der wesentlichen Funktionen, dem angebotenen Kundensupport und der Sicherheit. Unterschiede sind besonders bei der Preisgestaltung zu finden. Für diese Projektarbeit wurden ausschließlich kostenfreie Angebote in Erwägung gezogen.

Nach umfassenden Recherchearbeiten fiel die Wahl auf Googles App-Entwicklungsplattform *Firebase*, in dessen Produktangebot eine Echtzeitdatenbank zu finden ist: Die *Firebase Realtime Database* (RTDB). Die Firebase RTDB ist eine in der Cloud gehostete NoSQL-Datenbank (Google LLC, 2022). Daten werden als JSON[4] gespeichert und in Echtzeit mit jedem verbundenen Client synchronisiert. Alle Clients teilen sich eine RTDB-Instanz und erhalten automatisch Updates mit den neuesten Daten.

Google stellt Softwareentwicklern, die die Firebase-Plattform verwenden möchten, eine Reihe von Software Development Kits (SDKs) zur Verfügung. Die Integration der *Firebase Android SDK* in ein bestehendes App-Projekt ermöglicht die direkte Kommunikation zwischen einem Client-Gerät und der Firebase RTDB. Dadurch entfällt die Notwendigkeit der Einrichtung eines Anwendungsservers sowie die Entwicklung eines Back-Ends bzw. APIs[5].

Der kostenlose Spark-Plan von Firebase beinhaltet 1 GB Speicherplatz für die RTDB und erlaubt den gleichzeitigen Datenbankzugriff von bis zu 100 Nutzern (Stand: Q2, 2022).

2.1.2 Benutzerverwaltung

Die Erstellung, Verwaltung und Authentifizierung von Benutzern läuft innerhalb der Firebase-Plattform über den Dienst *Firebase Authentication*, der sich eng in die anderen Firebase-Dienste integrieren lässt. Grundsätzlich registriert sich ein neuer Benutzer für gewöhnlich selbst, direkt in der App. Der Einfachheit wegen und auch weil bei den in der Einleitung angesprochenen Board-Gamern von einer festen, überschaubaren Gruppe von Freunden ausgegangen werden konnte, wurde die Erstellung von Benutzern zu Testzwecken über den Webbrowser direkt in der Firebase-Konsole vorgenommen.

2.1.3 Automatisierung

Da sich die Gruppe von Board-Gamern immer turnusmäßig bei einem anderen Spieler zu Hause treffen möchte, musste ein Weg gefunden werden, um programmatisch den Gastgeber des nächsten Spieleabends zu bestimmen. Aus Gründen der Fairness sollte diese Auswahl nicht dem Zufallsprinzip überlassen werden. Stattdessen soll in der Datenbank jener Spieler gefunden und ausgewählt werden, dessen letzter Spieleabend als Gastgeber am längsten in der Vergangenheit liegt. Es wurde angenommen, dass jede Woche genau ein Spieleabend stattfindet. Das Stück

[4] JSON ist ein kompaktes Datenformat in einer einfach lesbaren Textform für den Datenaustausch zwischen Anwendungen (Wikimedia Foundation Inc., 2022a).
[5] Über eine Anwendungsschnittstelle, kurz API (von englisch application programming interface), können zwei oder mehrere Programme miteinander kommunizieren.

Programmcode, das den Gastgeber bestimmt, muss demnach in regelmäßigen Abständen von einer Woche wiederholt ausgeführt werden. Im Kontext von RDBMS wird in diesem Zusammenhang häufig von Events bzw. *Scheduled Events* gesprochen. In der MySQL[6] Dokumentation findet sich dazu folgende Definition:

> MySQL Events are tasks that run according to a schedule. Therefore, we sometimes refer to them as *scheduled* events. When you create an event, you are creating a named database object containing one or more SQL statements to be executed at one or more regular intervals, beginning and ending at a specific date and time. Conceptually, this is similar to the idea of the Unix `crontab` (also known as a "cron job") or the Windows Task Scheduler. (Oracle Corporation, 2022)

Das Äquivalent zu Scheduled Events stellt innerhalb der Firebase-Plattform der Dienst *Cloud Functions for Firebase* dar. Hier wird eine in JavaScript oder TypeScript geschriebene Funktion in der Google Cloud gespeichert, die dann in sich regelmäßig wiederholenden Abständen die entsprechenden Datenbankoperationen ausführen kann. Da die Verwendung von *Cloud Functions for Firebase* jedoch nur mit dem kostenpflichtigen Blaze-Plan von Firebase möglich ist, wurde für diese Projektarbeit eine andere Vorgehensweise gewählt. Der Codeabschnitt, der den Gastgeber bestimmt, wurde in den Programmcode der App implementiert. Die genaue Vorgehensweise wird im Kapitel *3.2.2 Main Activity* erläutert.

2.2 Aufbau

Die App wurde so konzipiert, dass sich der Spieler beim erstmaligen Start auf einer Login-Maske wiederfindet, wo er zur Eingabe seiner Anmeldeinformationen aufgefordert wird. Nach erfolgreicher Authentifizierung durch den Firebase-Server wird der Spieler zum Hauptbereich weitergeleitet. Dieser besteht wiederum aus drei Fragmenten[7] und einer unteren Navigationsleiste (*BottomNavigationBar*[8]). Die Navigationsleiste ermöglicht den Wechsel zwischen den Fragmenten. Alle soeben genannten Komponenten werden in den folgenden Kapiteln detailliert beschrieben.

3. Implementierung

3.1 Datenbank

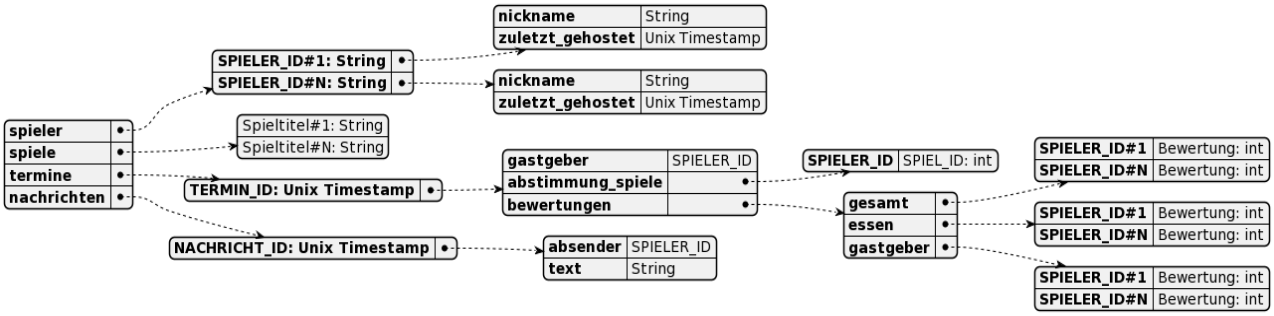
Die Datenbank wurde direkt in der Firebase-Konsole erstellt. Anschließend wurden über den *Firebase Authentication* Dienst fünf Test-Benutzer angelegt und die UIDs der Benutzerkonten manuell in die Datenbank übertragen.

---

[6] MySQL ist eines der weltweit verbreitetsten RDBMS.
[7] Ein Fragment stellt einen wiederverwendbaren Teil der Benutzeroberfläche der App dar (Google LLC, 2021).
[8] https://developer.android.com/reference/com/google/android/material/bottomnavigation/BottomNavigationView

Die folgende Abbildung zeigt das Schema der NoSQL-Datenbank. Das Diagramm wurde mit dem Open-Source-Tool[9] PlantUML (plantuml.com) erstellt.

Abb. 2 Datenbankstruktur



Quelle: Eigene Darstellung.

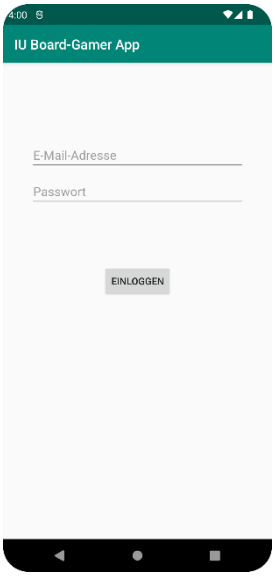
Beim *Unix timestamp* (zu Deutsch *Unixzeit*) handelt es sich um die Anzahl der vergangenen Sekunden seit dem 01. Jänner 1970 (UTC[10]). Er wird in der Softwareentwicklung häufig zur Speicherung von Zeitpunkten verwendet.

3.2 Applikation

In diesem Kapitel werden die Hauptkomponenten der App erläutert. Es wird auf die verschiedenen Oberflächen sowie die Kommunikation mit der Firebase-Plattform eingegangen.

3.2.1 Login Activity

Abb. 3 Login Activity



Quelle: Eigene Darstellung.

Diese Activity[11] ist im Android App Manifest[12] als *Launcher Activity* deklariert, wird also direkt nach dem Start der App ausgeführt. Auf der Oberfläche befinden sich zwei Texteingabefelder zur Angabe des Benutzernamens und -passworts. Der Benutzername entspricht der E-Mail-Adresse des Spielers. Unter dem Passwortfeld befindet sich ein Button, der den Log-in-Vorgang einleitet. Die Anmeldeinformationen werden zur Prüfung an den *Firebase Authentication* Dienst übermittelt. Schlägt die Authentifizierung durch den Firebase-Server fehl oder wurde eines der beiden Eingabefelder freigelassen, so wird dem Benutzer eine entsprechende Fehlermeldung angezeigt.

---

[9] „Als **Open Source** (aus englisch *open source*, wörtlich *offene Quelle*) wird Software bezeichnet, deren Quelltext öffentlich und von Dritten eingesehen, geändert und genutzt werden kann. Open-Source-Software kann meistens kostenlos genutzt werden“ (Wikimedia Foundation Inc., 2022b).

[10] “Die **koordinierte Weltzeit** (englisch *Coordinated Universal Time*), kurz **UTC**, ist die heute gültige Weltzeit“ (Wikimedia Foundation Inc., 2022c).

[11] Eine Activity ist eine Hauptkomponente einer Android-App, die dem Benutzer eine Oberfläche zur Interaktion mit der App bereitstellt.

[12] Das App Manifest ist zwingender Bestandteil jeder Android-App. Es enthält essenzielle Information über die App.

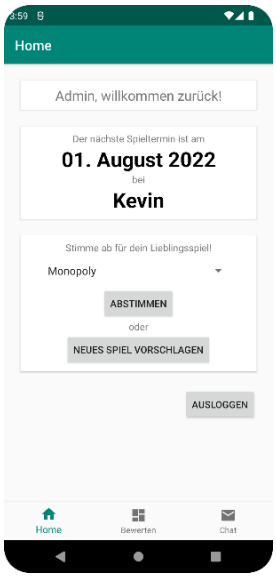
Der *Firebase Authentication* Dienst wird zudem bei der Initialisierung der Login Activity kontaktiert, um zu prüfen, ob der Benutzer bereits eingeloggt ist. Ist dies der Fall, so wird unmittelbar zur Main Activity gewechselt.

3.2.2 Main Activity

Die Main Activity dient als Basis für eines von drei Fragmenten, die die verschiedenen Benutzeroberflächen darstellen. Eine untere Navigationsleiste ermöglicht den Wechsel zwischen den Fragmenten.

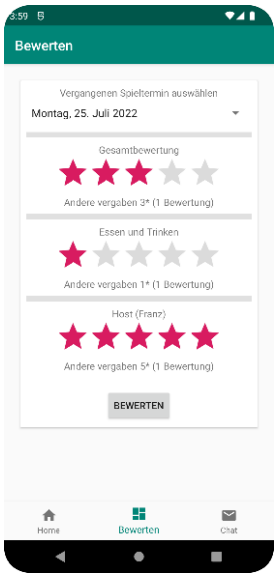
Im Kapitel *2.1.3 Automatisierung* wurde erwähnt, dass die wöchentliche Bestimmung des Gastgebers des nächsten Spieleabends in der App stattfindet. Dies geschieht beim Aufruf der Main Activity, das heißt, nachdem der Spieler vom Firebase-Server authentifiziert wurde. Zunächst wird eine Verbindung zur Datenbank hergestellt und die Daten des jüngsten Spieleabends abgerufen. Sollte das Datum (der Unix timestamp) des Termins in der Vergangenheit liegen, wird ein neues, zukünftiges Event angelegt. Um den Gastgeber zu bestimmen, wird durch die Liste der registrierten Spieler iteriert und jener Spieler ausgewählt, dessen letzter Spieleabend als Gastgeber am längsten in der Vergangenheit liegt.

Abb. 4 Home Fragment



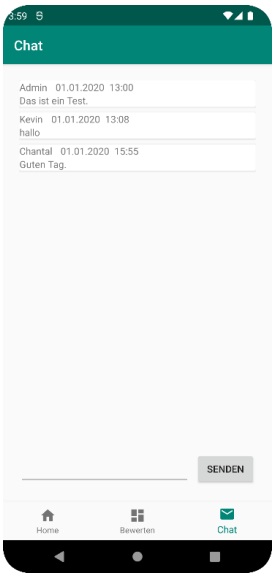
Quelle: Eigene Darstellung.

Abb. 5 Rating Fragment



Quelle: Eigene Darstellung.

Abb. 6 Chat Fragment



Quelle: Eigene Darstellung.

Home Fragment

Hierbei handelt es sich um jene Oberfläche, die dem bereits angemeldeten Benutzer beim Start der App, oder einem nicht angemeldeten Benutzer nach erfolgtem Log-in-Vorgang angezeigt wird. Die Oberfläche ist in drei *CardView*[13] Elemente unterteilt. Die erste *CardView* beinhaltet eine persönliche Begrüßung des Spielers. In der zweiten *CardView* werden das Datum des nächsten Spieltermins

[13] CardView ist ein Layout mit Hintergrund, abgerundeten Ecken und Schattierung.

und der Gastgeber angeführt. Die letzte *CardView* enthält eine Drop-Down-Liste der für den Abend zur Auswahl stehenden Spiele. Dem Spieler steht es frei für sein Lieblingsspiel abzustimmen oder ein neues Spiel vorzuschlagen. Dazu muss er lediglich auf die entsprechenden Buttons tippen. Das letzte Element dieser Oberfläche stellt ein Log-out-Button dar. Zur Durchführung des Log-outs muss wieder der *Firebase Authentication* Dienst kontaktiert werden. Nach dem erfolgten Log-out wird zur Login Activity gewechselt.

Rating Fragment

Über diese Oberfläche kann der Benutzer die letzten Spieleabende bewerten und die Bewertungen anderer Spieler einsehen. Aus Gründen der Aktualität können nur die fünf letzten Termine angesehen und bewertet werden. Dies kann durch Veränderung einer Variablen im Programmcode nach Belieben angepasst werden. Die Bewertung erfolgt anonym. Bewertet werden kann der Abend als Ganzes, die Verpflegung sowie der Gastgeber. Es können jeweils ein bis fünf Sterne vergeben werden[14]. Die Bewertung wird nur dann in der Datenbank gespeichert, wenn in jeder der drei Kategorien eine Bewertung abgegeben wurde. Andernfalls wird dem Spieler eine entsprechende Fehlermeldung angezeigt.

Chat Fragment

Diese Oberfläche zeigt die zwischen den Spielern ausgetauschten Sofortnachrichten an. Über ein Texteingabefeld kann eine neue Nachricht verfasst werden. Durch Tippen auf den Button mit der Aufschrift „Senden“ wird die Nachricht zur Speicherung an die Datenbank übermittelt. Demnach handelt es sich um eine einfache, klassische Instant-Messaging Anwendung ohne die heutzutage übliche Funktionalität zum Versenden von Medien, Sprachaufzeichnungen und Emojis[15].

Jede Nachricht enthält den Namen des Absenders, das Sendedatum, die Uhrzeit und natürlich den Text. Um die einzelnen Nachrichten ressourceneffizient darzustellen, wurde die *RecyclerView* Bibliothek verwendet. In der Android-Dokumentation wird diese wie folgt beschrieben:

> RecyclerView makes it easy to efficiently display large sets of data. You supply the data and define how each item looks, and the RecyclerView library dynamically creates the elements when they're needed.
>
> As the name implies, RecyclerView recycles those individual elements. When an item scrolls off the screen, RecyclerView doesn't destroy its view. Instead, RecyclerView reuses the view for new items that have scrolled onscreen. This reuse vastly improves performance, improving your app's responsiveness and reducing power consumption. (Android Open Source Project, 2022)

---

[14] Ein Stern ist gleichbedeutend mit der Schulnote „ungenügend“, fünf Sterne mit „sehr gut“.
[15] Emojis sind Bildschriftzeichen 🤓.

4. Fazit und Ausblick

Es ist im Rahmen dieser Projektarbeit gelungen, eine Android Board-Gamer App zu entwickeln, die die in der Einleitung aufgezählten Anforderungen erfüllt.

Da es sich lediglich um ein Übungsprojekt handelte, wurden einige Aspekte der App sehr einfach gehalten. So kann das Datum eines Spieltermins beispielsweise nicht verändert werden und fällt stets auf einen Montag. Es fehlt die Funktionalität für die Benutzerregistrierung in der App und die Möglichkeit zur Wiederherstellung bzw. Änderung eines vergessenen Passworts. Die Benutzeroberfläche ist optisch wenig ansprechend - Einstellungen zur Anpassung der Darstellung oder des Benutzerprofils fehlen zur Gänze. Es könnten Push-Benachrichtigungen implementiert werden, die den Spieler z.B. über den Termin & den Gastgeber des nächsten Spieleabends oder über neue Nachrichten im Chat informieren. Denkbar wäre es auch, den Chat um den Versand von Bildern, Sprachaufzeichnungen usw. zu erweitern. Ob dies in Zeiten von Telegram-, WhatsApp-Gruppen und Co. aber tatsächlich zielführend wäre, sei dahingestellt.

Grundsätzlich ist die App funktionstüchtig und könnte, theoretisch, für den nächsten Brettspiel-, Videospiel- oder Leseabend mit den Freunden verwendet werden.

III. Literaturverzeichnis

Amazon Web Services, Inc. (2022). *Was ist NoSQL?* https://aws.amazon.com/de/nosql

Android Open Source Project. (2022). *Create dynamic lists with RecyclerView*. https://developer.android.com/guide/topics/ui/layout/recyclerview

Google LLC. (2021). *Fragments*. https://developer.android.com/guide/fragments

Google LLC. (2022). *Firebase Realtime Database*. https://firebase.google.com/docs/database

Microsoft Corporation. (2022). What is cloud computing? https://azure.microsoft.com/de-de/resources/cloud-computing-dictionary/what-is-cloud-computing

Oracle Corporation. (2022). *25.4.1 Event Scheduler Overview*. https://dev.mysql.com/doc/refman/8.0/en/events-overview.html

Synergy Research Group. (2021). *Worldwide market share of leading cloud infrastructure service providers in Q4 2021*. Statista. https://www.statista.com/chart/18819/worldwide-market-share-of-leading-cloud-infrastructure-service-providers

Wikimedia Foundation Inc. (2022a). *JavaScript Object Notation*. https://de.wikipedia.org/wiki/JavaScript_Object_Notation

Wikimedia Foundation Inc. (2022b). *Open Source*. https://de.wikipedia.org/wiki/Open_Source

Wikimedia Foundation Inc. (2022c). *Koordinierte Weltzeit*. https://de.wikipedia.org/wiki/Koordinierte_Weltzeit