VILNIUS UNIVERSITY
FACULTY OF MATHEMATICS AND INFORMATICS
INSTITUTE OF COMPUTER SCIENCE
INFORMATION TECHNOLOGIES STUDY PROGRAM

Problem-Based Project

Words Learning App: Report

Done by:
Nikita Liakhov
Fiona Pruvost
Dominyka Rimova
Valeriia Rudenko

Supervisor:
Linas Būtėnas

Vilnius
2023

Preface

This project was done during the second semester of the study programme *Information Technologies* in the specialization of Innovative studies. We chose the topic *Words Learning app* suggested during the project market on the first lecture of the subject ‘‘Problem-Based Project’’.

March 22, 2023

Nikita Liakhov

Fiona Pruvost

Dominyka Rimova

Valeriia Rudenko

Contents

Abstract

Since the 21st century, with the development of modern technologies, such as the Internet, tablets, and smartphones, there are more innovative ways to learn languages. For example, people are able to use electronic devices to send messages, present ideas or share information anytime, anywhere. However, most of the mobile apps for learning languages or words are designed and presented only in English interface. In the recent decade, the popularity of mobile phones has skyrocketed and there has been an explosive growth in apps designed for children. The purpose of our project is to create an Android application where 5-10 years old children can learn vocabulary of a selected language. The application is available in four different languages: Lithuanian, English, Ukrainian, French. There are two different options to learn words: "Quiz" and "Flashcard" modes. Application includes three difficulty levels. The app is be able to display user's quiz score. "Flashcard" and "Quiz" modes will have three different ways to learn: either by text, pictures, or audio. To save and transfer learning progress a parent must register an account. A user with an account will be able to add new sets of custom flashcards. The app is implemented using the MVVM architecture and integrates a web server and PostgreSQL database to enable scalability and modularity. This app is going to help parents to teach their children different languages and words using smartphones.

Keywords: Word Learning App, Android, Vocabulary, Quiz, Flashcard, Children

Santrauka

Darbo pavadinimas kita kalba: "Žodžių mokymosi aplikacija"

Nuo XXI amžiaus, tobulėjant šiuolaikiškoms technologijoms, tokioms kaip internetas, planšetiniai kompiuteriai ir išmanieji telefonai, atsiranda naujoviškesnių kalbų mokymosi būdų. Pavyzdžiui, žmonės gali naudoti išmaniuosius įrenginius norėdami siųsti žinutes, išreikšti idėjas ar dalytis informacija bet kur ir bet kada. Tačiau dauguma mobiliųjų programėlių, skirtų kalbų ar žodžių mokymuisi, yra sukurtos ir pateiktos anglų kalba. Per pastarąjį dešimtmetį mobiliųjų telefonų populiarumas smarkiai išaugo, ir taip pat sparčiai išaugo vaikams skirtų programėlių skaičius. Mūsų projekto tikslas – sukurti Android aplikaciją, kurioje 5-10 metų vaikai galėtų išmokti pasirinktos kalbos žodyno. Aplikacija yra pateikiama keturi skirtingomis kalbomis: lietuvių, anglų, ukrainiečių, prancūzų. Aplikacijoje yra du skirtingi žodžių mokymosi būdai : „Quiz“(lt. testas) ir „Flashcard“(lt. (dvipusė) kortelė) režimai. Programoje yra trys skirtingų sunkumų lygiai. Programa gali rodyti vartotojo testo rezultatą. „Flashcard“ ir „Quiz“ režimai turi trys skirtingus mokymosi būdus: pagal tekstą, paveikslėlius arba garsą. Norėdami išsaugoti ir perkelti mokymosi progresą, tėvai turi užregistruoti paskyrą. Registruotas vartotojas galės pridėti savo sukurtų kortelių rinkinius ("Flashcard" režime). Programėlė sukurta naudojant MVVM architektūrą ir integruoja žiniatinklio serverį bei PostgreSQL duomenų bazę, kad būtų galima keisti mastelį ir moduliškumą. Ši programa padės tėvams išmokyti savo vaikus skirtingų kalbų ir žodžių naudojant išmaniuosius telefonus.

Raktiniai žodžiai: žodžių mokymosi aplikacija, „Android“, žodynas, viktorina/testas, dvipusė kortelė, vaikai.

1 Introduction

The goal of this project was to develop a words learning app that would help young learners improve their vocabulary and language skills in a fun and interactive way. The app includes three quiz modes, in which children can learn new words by looking at pictures and selecting the corresponding word, listening to the pronunciation of a word and selecting the correct spelling, or writing the word that corresponds to a given image.

In addition to the quiz modes, the app also includes flashcards to help kids study and review the words they've learned. By flipping between a word and its translation, picture, or pronunciation, children can get a better understanding and remember each word.

Throughout the development process, we kept the needs and interests of our target audience in mind, designing the app to be engaging and educational for children between the ages of 5 and 10. In this report, we will detail the design and development process of the app, as well as the results of user testing. We hope that this app will be a valuable resource for children as they learn and improve their language skills.

2 Purpose of the system

2.1 Git link

The source code for this project is available on the Vilnius University Gitlab at https://git.mif.vu.lt/fipr8888/lifruk.

2.2 Target audience

The target audience for this quiz app is children between the ages of 5 and 10. This age group was chosen as they are typically at a stage where they are starting to learn new languages and expand their vocabulary. The app is designed to be engaging and educational for this age group, with features and functions that are appropriate for their cognitive and developmental level.

In order to ensure that the app is suitable for this age group, it was important to consider factors such as the level of difficulty of the quiz modes, the clarity and simplicity of the user interface, and the appeal of the visual elements. By taking these factors into account, we aim to create an app that is both enjoyable and effective for young learners as they learn and improve their language skills.

2.3 Users

- Children: The primary users of the app are children between the ages of 5 and 10. These users will interact with the app by participating in the quiz and flashcard activities and learning new words in the selected language.
- Parents: Parents or caregivers may also use the app to help their children learn languages and track their progress. They may have the ability to customize the app's settings, such as the language and difficulty level, and view their children's quiz scores.
- Administrator: The administrator is responsible for managing the app and its content. This includes adding new languages or quiz modes, updating the flashcard sets, and managing user accounts. The administrator may also have the ability to view and analyze usage data to understand how the app is being used and identify areas for improvement.

3 Analysis

3.1 Related work

There are several existing quiz apps that aim to help children (and people in general) learn new languages. These apps use a variety of techniques, such as games or other interactive activities, to make the learning process more fun and engaging.

One example of a related app is Duolingo, which uses a combination of multiple choice questions and fill-in-the-blank exercises to teach new words. Duolingo has the advantage of providing immediate feedback to users, allowing them to see their progress and identify areas where they need to improve. It also offers a wide range of languages to choose from and has a user-friendly interface. However, it may not be as interactive or visually appealing as some other apps on the market.

Another related app is Babbel, which utilizes flashcards and picture matching games to teach new words. Babbel has the advantage of providing a variety of activities for users to choose from, making it more engaging and potentially more effective at promoting retention of new words. It also offers professional voice acting and a speech recognition feature to help users practice pronunciation. However, it may not offer as much immediate feedback as Duolingo and may be more expensive for users who want to access all of its features.

Overall, there are a number of existing apps that use different approaches to teaching languages and vocabulary to children. While each of these apps has its own advantages and disadvantages, they all share the goal of making the learning process enjoyable and effective for young learners.

3.1.1 Systems

Duolingo and Babbel are available on iOS and Android devices. They also have a website version of their apps.

3.1.2 Existing algorithms

- Multiple choice algorithm: Users are presented with a list of options (e.g. four images) and asked to select the correct picture that corresponds to the answer. Users get notified if they choose the wrong answer. This is used to test users' knowledge of new words and provide immediate feedback on their progress.

- Flashcard algorithm: This displays a word on one side of the card and its translation, pronunciation or corresponding image on the other side. Users can flip back and forth between the two sides to study and review the words they have learned.

- Spelling algorithm: This algorithm plays the pronunciation of a displayed word and asks users to spell it correctly in the given text field. This helps users learn the correct spelling of given words, improve their hearing skills and by pronouncing the word out loud improve their pronunciation skills.

3.1.3 Existing solutions

- Gamification: Gamification refers to the use of game-like elements, such as points, badges, and leaderboards, to make learning more engaging and enjoyable. By incorporating gamification elements into the app, users can feel a sense of accomplishment as they progress through and complete quizes, learn new words, get the high score.

- Personalization: Personalization involves tailoring the learning experience to the individual needs and preferences of each user. This includes adapting the difficulty level of the quiz modes based on a user's progress, allowing users to select specific words or categories they want to learn, or allow users to create their own quiz categories and words for personal use.

- Adaptive learning: Adaptive learning refers to the use of algorithms that adjust the learning experience based on a user's performance. For example, if a user is struggling with a particular word or concept, the app provides additional explanations or examples to help them understand it better, or makes the user redo a quiz or task with the word they struggle with.
- Multisensory learning: Multisensory learning involves using a variety of methods to engage different senses and facilitate language learning. This includes providing audio pronunciations of words, displaying images to illustrate concepts and words, or using virtual flashcards or matching games.

4 Functional requirements

4.1 Use case diagram

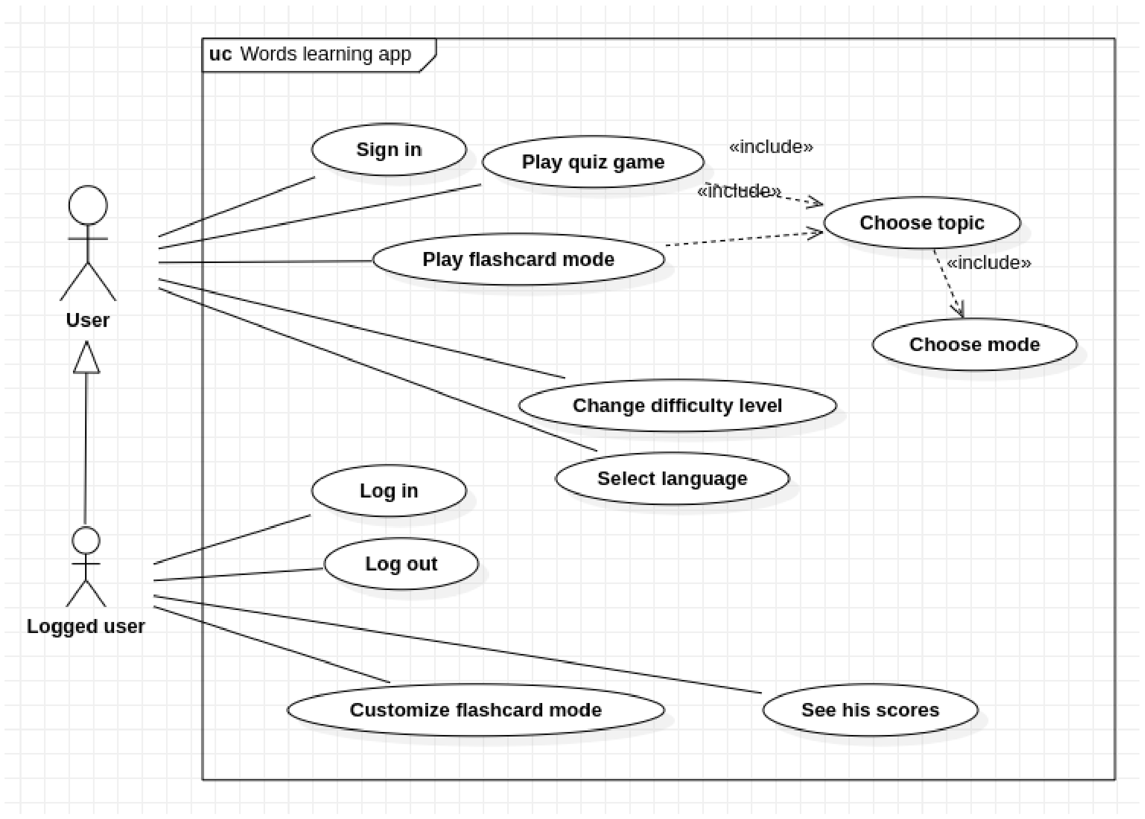

Figure 1. *Use case diagram*

Some of the key functions of the app include:

- Login and sign-up: These functions allow users to create an account and access their personalized learning progress and settings.
- View score: This function allows users to see their progress within the app.
- Change language and difficulty mode: These functions allow users to customize their learning experience by selecting the language they want to learn and the level of difficulty they want.

- Play game: This function allows users to choose a topic and play a quiz game to test their knowledge of new words.
- Play flashcard: This function allows users to choose a topic and to use a digital flashcard to learn new words.
- Customize flashcards: This function allows users to create their own flashcards.

4.2 Requirements list

4.2.1 High priority

- Users should be able to create an account and log in to access their personalized learning progress and settings.
- Users should be able to choose from a variety of languages (English, Lithuanian, Ukrainian, French) to learn and select their preferred level of difficulty (easy, medium, hard).
- Users should be able to play a quiz game after selecting a specific topic (i.e. category of words to learn).
- Users should be able to access flashcards to study and review the words they have learned.

4.2.2 Medium priority

- Users should be able to view their score within the app.
- Users should be able to customize their flashcards by adding their own words.

4.2.3 Low priority

- Ability to flip the card with two sides repeatedly by tapping it.

5 Non-functional requirements

5.1 Compatibility

- The mobile app must be able to run on mobile phones running Android 7 or higher.

5.2 Reliability

- The app should be easy to maintain and update, with a clear and well-documented codebase.
- The app should be scalable, with the ability to handle an increasing number of users and data without degrading performance.

5.3 Security

- The app should be secure, with measures in place to protect user data and prevent unauthorized access.
- Avoid collecting any unnecessary personal information.

5.4 Performance

- The app should be responsive and perform well on a variety of smartphones.
- User can use this application quickly with a 3G internet connection.

5.5 Design

- It is a children (5 - 10 years old) application so the application must have a lot of colors, pictures and the words to learn must be simple. The app should be visually appealing and engaging, with clear and appealing graphics and layout.
- The app should have a user-friendly and intuitive interface that is easy for children to navigate.

6 Preliminary design

6.1 Prototype

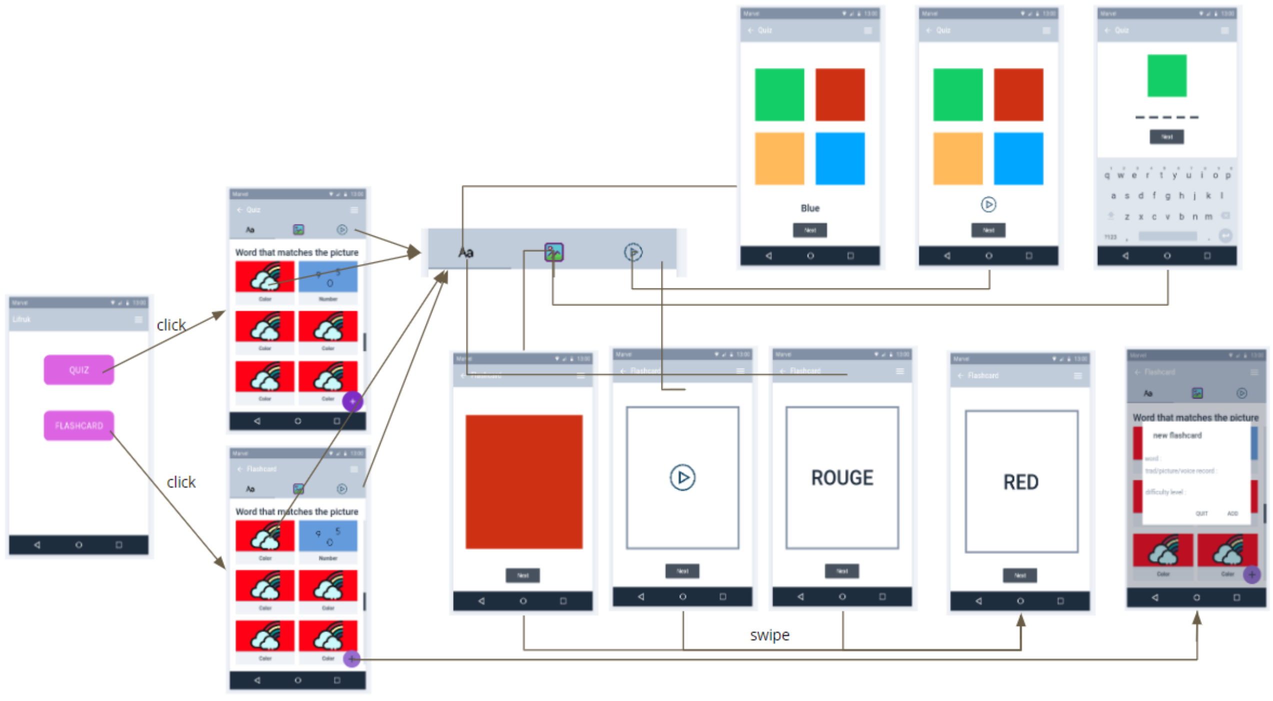

Figure 2. *Prototype*

6.2 Sign up

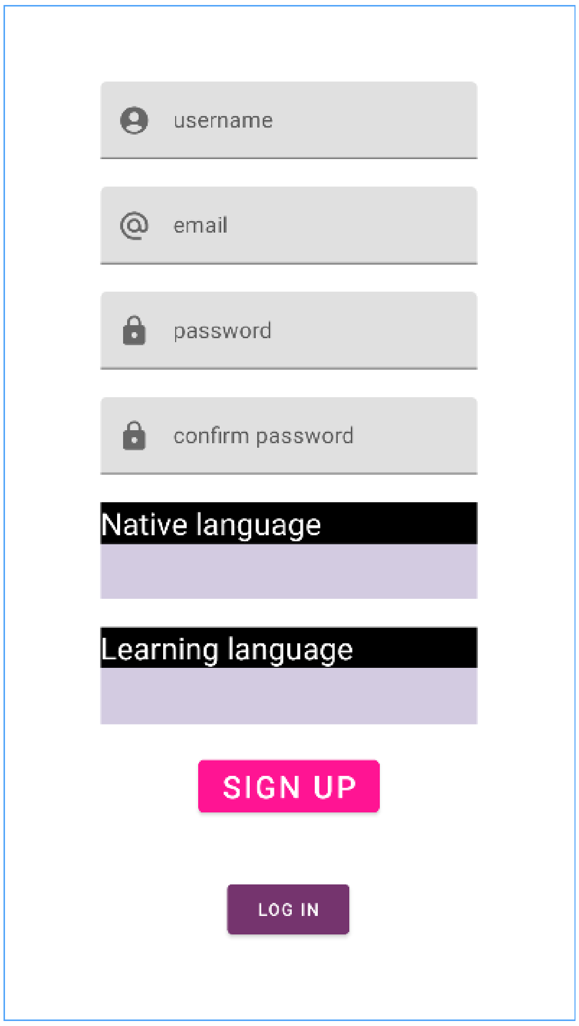

Figure 3. *Sign up interface*

In this screen, users can enter their username, email address, password and confirm the password by reentering it in order to register an account. Moreover, they choose their native and learning language.

They can submit this form by pressing the "Sign Up" button or they can choose the "Log in" button if they already have an account.

6.3 Log in

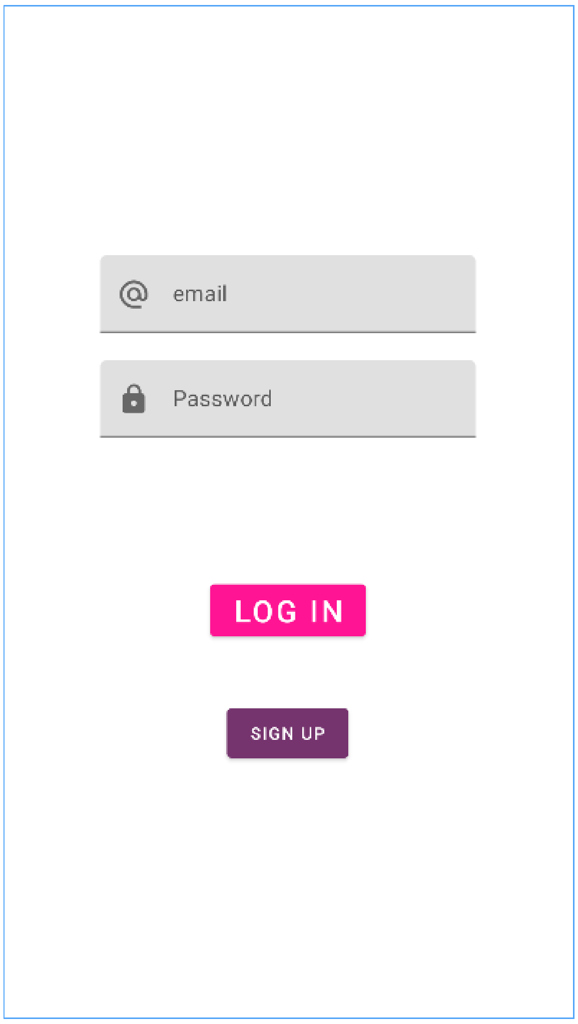

Figure 4. *Log in interface*

In this screen, users can enter their username, email address, password in order to log in into their account. If login is successful, the user will see the main menu page. If it's not successful, error message will be displayed.

They can submit this form by pressing the "Log in" button or they can choose the "Sign Up" button if they want to create an account.

6.4 Game menu

The main screen will have 2 buttons, "Quiz" and "Flashcard", with each redirecting the user to the different screen.

After selecting a desired mode, the user will see a new screen with different topics and at the top of the screen there will be a navigation panel with 3 tabs: Text tab, Picture tab, Audio tab. Text tab will be opened by default.

6.5 Quiz mode

After submitting the answer in each quiz screen, the user will be shown a new quiz screen. There will be 10 questions in the quiz and at the of the quiz the player will be able to see their score.

6.5.1 Word

After selecting Word tab, the user will be shown a set of 4 pictures and a word. The user then will have to choose the picture that corresponds to the word given.

6.5.2 Picture

If the user chooses Picture tab, they will be shown a picture and will have to type in the answer (i.e. what they think the picture depicts).

6.5.3 Audio

Finally, if the user selects Audio tab, they will be shown a set of pictures and an audio button that has the word pronunciation audio. The user then will have to choose the picture that corresponds to the audio given.

6.6 Flashcard mode

After the user is done using a flashcard, they will have the option to move onto the next one in the set. Next flashcard is picked randomly. A floating button will allow the user to create a new flashcard: it opens a dialog box where they can enter flashcard information and submit it.

6.6.1 Word

After selecting Text tab, one side of the flashcard will be a word, the other side will contain a translation of that word.

6.6.2 Picture

If the user chooses Picture tab, one side of the flashcard will be a picture, the other side will have a word that corresponds to the picture.

6.6.3 Audio

Finally, if the user selects Audio tab, one side of the flashcard will be pronunciation audio of a word, the other side will have the word written out.

6.7 Settings

Settings allow to change the language you want to learn and the difficulty level.

7 System architecture

7.1 High level overview

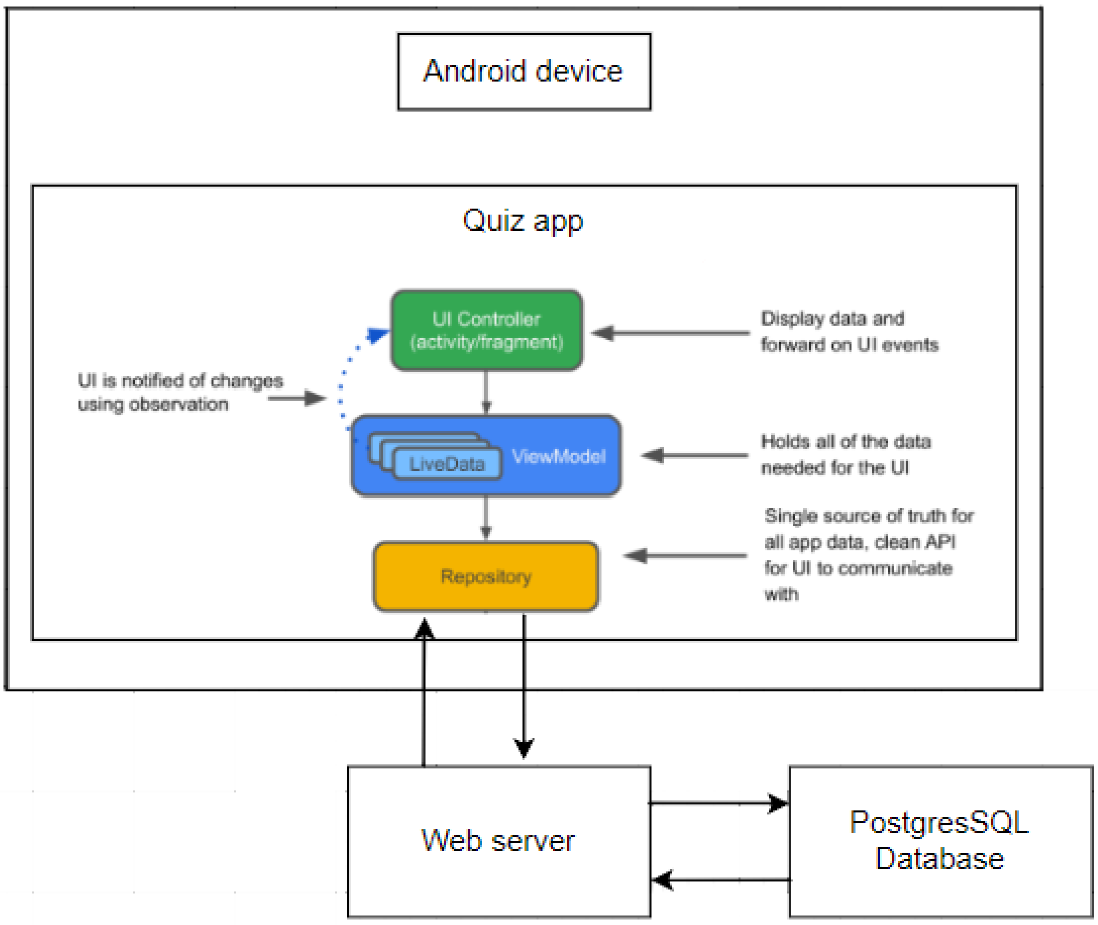
Android device
Quiz app
UI Controller (activity/fragment)
Display data and forward on UI events
UI is notified of changes using observation
LiveData
ViewModel
Holds all of the data needed for the UI
Repository
Single source of truth for all app data, clean API for UI to communicate with
Web server
PostgresSQL Database

Figure 5. *System architecture diagram*

- Android client: The Android client is the front-end of the app, which users interact with on their Android phones. It is developed using Kotlin programming language in Android Studio, and is responsible for rendering the user interface and handling user input.
- Web server: The web server is responsible for handling requests from the Android client and interacting with the remote database. It is implemented using the Spring framework and serves as a connection between the client and the database.
- Database: The database stores certain app data, such as user accounts, quiz scores, and flashcard information. It is implemented using PostgreSQL and is accessed via the web server.
- Network: The network connects Android client, web server, database and enables communication between them. It can be a local network (e.g. within an organization) or a public network (e.g. the Internet).

7.2 Deployment

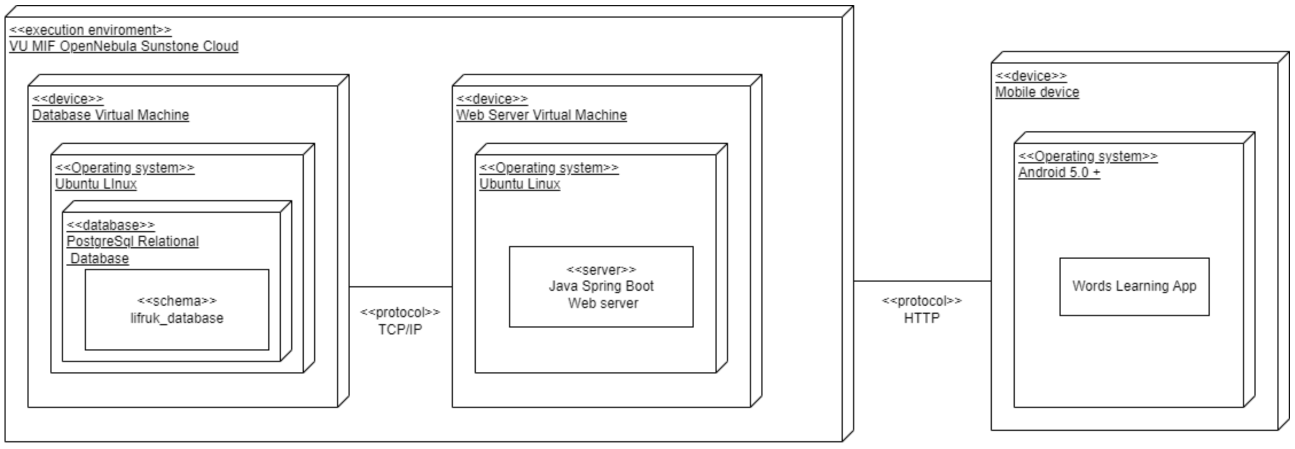

Figure 6. *UML deployment Diagram*

UML deployment diagram: overview of the whole system showing the deployed environment and interactions between the main components.

7.3 Model-View-ViewModel (MVVM)

The application is made using the MVVM architecture. The Model-View-ViewModel (MVVM) architecture is a design pattern that separates the user interface (UI) of the application into three distinct layers: the model, the view, and the view model.

- Model: The model represents the data and business logic of the application. It is responsible for managing and manipulating the data, as well as performing any necessary calculations or processing.
- View: The view represents the UI of the application. It is responsible for rendering the interface to the user and handling user input.
- View Model: The view model acts as a bridge between the model and the view. It is responsible for exposing the data and functionality of the model to the view, as well as handling any UI-specific logic or tasks.

The MVVM architecture helps to decouple the UI from the underlying data and business logic, making it easier to maintain and test the application. It also allows for a more modular and scalable design, as the model, view, and view model can be developed and tested independently of each other.

7.4 Technologies and tools

- Android Studio: Android Studio is the official integrated development environment (IDE) for android app development. It is based on the IntelliJ IDEA platform and provides a needed tools and features for building, testing, and debugging android apps.

- Kotlin: Kotlin is a programming language that runs on the Java virtual machine (JVM) and is fully interoperable with Java. It is used to implement the model, view model, and other components of the app.
- Spring framework: The Spring framework is a Java-based application development framework that provides needed tools and features for building web and mobile applications. It is used to implement the web server and handle communication between the android client and the database.
- PostgreSQL: PostgreSQL is an object-relational database management system (ORDBMS). It is used to store and manage the data for the app, such as user accounts, quiz scores, and flashcard information.
- Git: Git is a version control system that allows us to track and manage changes to the codebase of our application. It is used to collaborate on the development of the app and maintain a history of changes.

8 Database

8.1 Description

- Each player has a username, an email and a password and each player can choose their native language and the language they want to learn.
- Each word has text (i.e. the word), a pronunciation, a topic and a picture.
- A translation has a word and word's translation.
- A picture has a name and a URL.
- Answer choices have 4 words and the index of the correct word (i.e. answer).
- Each question has text (i.e. the word), a word to guess and a type.
- Each quiz has 10 questions.
- An answer indicates if the player answered correctly at a given question.
- Game links a player and a word.
- A flashcard has a topic and a word.

8.2 Data

There are several different entities: player, flashcard, word, topic, game, answer, translation. The data that is stored in each entity:

- player (id, username, email, password, native language, language the player wants to learn);
- flashcard (id, wordID, playerID);
- game (id, playerID, topicID, type);

- word (id, language, content, topicID, picture);
- topic (id, name, difficulty);
- answer (id, wordID, gameID, correct, type);
- translation (id, wordID, wordTranslatedID);

The game type can be audio, word or picture. The difficulty level can be easy, medium or hard. Languages are English, Lithuanian, Ukrainian, French.

8.3 Entity Relationship Diagram

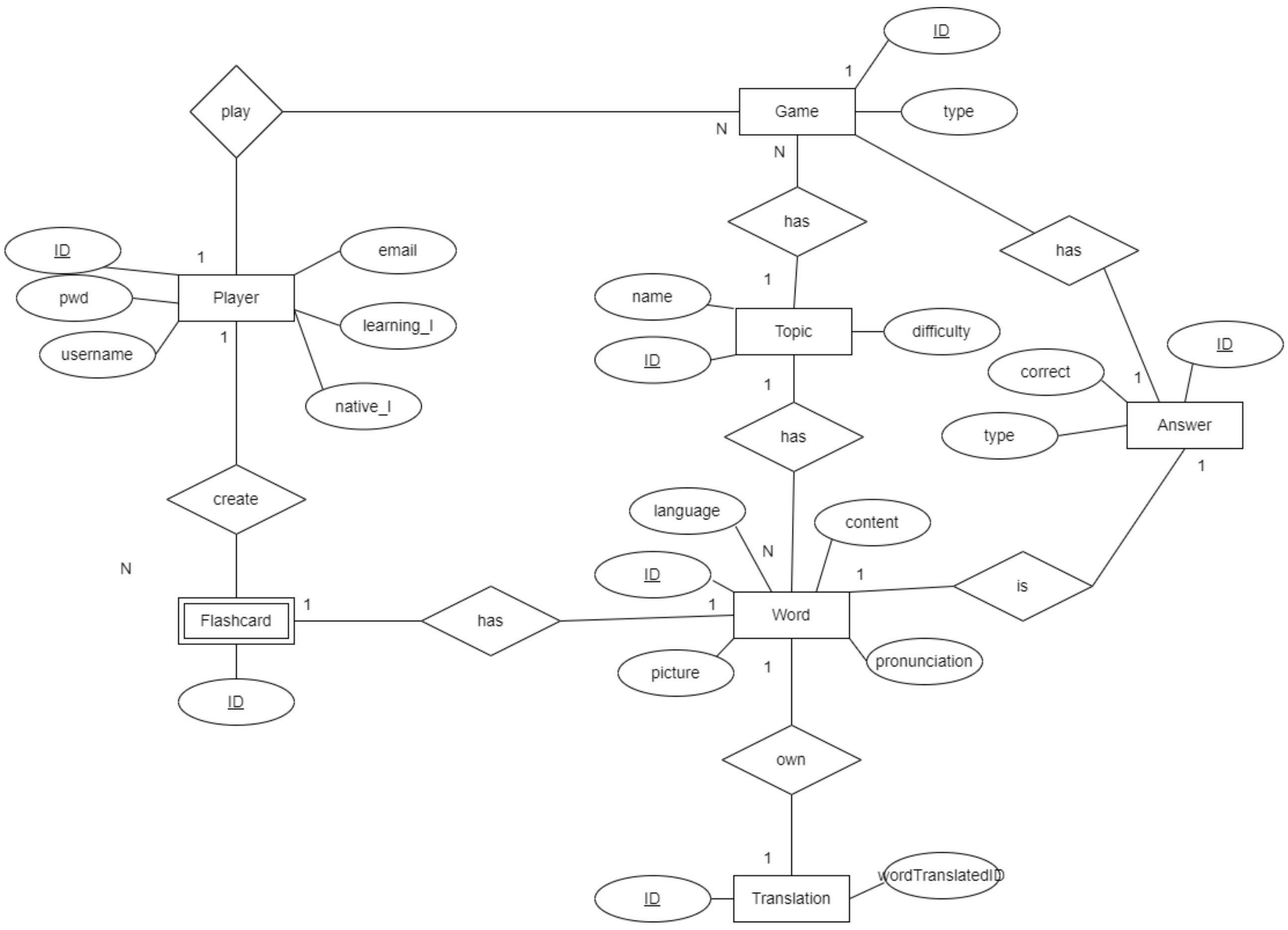

Figure 7. *Entity-Relationship Diagram*

8.4 Webserver

- The Android client sends a request to the web server. This request could be triggered by a user action, such as tapping a button in the app, or by a background task, such as synchronizing data with the server.
- The web server receives the request and processes it. Depending on the type of request, this might involve interacting with the database to retrieve or update data, performing calculations or processing, or performing other tasks.

- The web server generates a response based on the request and sends it back to the android client. This response could be a simple acknowledgement that the request was received, or it could be data or other information that the android client can use to update the UI or perform other tasks.
- The Android client receives the response from the web server and processes it. This might involve updating the UI to reflect any changes in the data, or it could involve performing other tasks based on the information received in the response.

9 Developed Algorithm

9.1 Generating next quiz question algorithm

In this section, the algorithm which generates the next quiz question is described.

9.1.1 Database select request

In this algorithm we have 2 selection requests.
The first one is a call in function getErrorQuestions(idPlayer: Int) that allows to get questions, to which the player (find with ID: playerId and get int) answered incorrectly in a selected topic.

Function ***getErrorQuestions(player: Int, topicName: String)***

```
SELECT Question.*
FROM Question, Player, Answer, Game, Word
WHERE  Player.id = Game.player
    AND Game.id = Answer.game
    AND Question.id = Answer.game
    AND Question.to\_guess = Word.id
    AND Game.topicName = Word.topicName
    AND Word.topicName = :topicName
    AND Answer.correct = 0
    AND Player.id = :player
ORDER BY RANDOM()
LIMIT 5
```

The second one allows to get the random questions, at most 10, from all questions in the database.

Function ***getRandomQuestions(nb: Int)***

```
SELECT *
FROM Question
ORDER BY RANDOM()
LIMIT :nb
```

9.1.2 Pseudo-code

At the beginning of a game, before selecting the next question, algorithm gets a set of 10 questions, with some (at most 5) answered incorrectly by the player (if they exist) and random questions from the database.

Function *generateQuestions(player: Player)*
INPUT:
player - the player's ID who play the game - Player
OUTPUT:
list10Questions - the list of 10 questions of the game - List<Question>
OUTPUT:

Algorithm 1. Function *generateQuestions(player: Player)* - Select set of 10 questions

```
Require: $player! = null \wedge player \geq 0$
1: if $player = null || player < 0$ then
2:     RETURN
3: else
4:     $errorQuestion \leftarrow getErrorQuestions(player)$
5:     $list10Questions.addAll(errorQuestions)$
6:     $numberQuestionLeft \leftarrow 10 - errorQuestions.size$
7:     $list10Questions.addAll(questionViewModel.getRandomQuestions(numberQuestionLeft))$

8:     return list10Questions
9: end if
```

Then this function is used to find the next question:
Function *nextQuestion(index: Int)*

INPUT:
index - the index of the next question - int
list10Questions - the list of 10 questions of the game - List<Question>
OUTPUT:
nextQuestion - the next question for the player - Question

Algorithm 2. Select next question

```
Require: $index \geq 0 \wedge index \leq 10$
1: if $index < 0 || index > 10$ then
2:     RETURN
3: else
4:     $nextQuestion \leftarrow list10Questions|index|$
5:     $index = index + 1$
6:     RETURN nextQuestion
7: end if
```

9.1.3 Real World Example

Using first example of data (Subsection 12.1)

Algorithm 1

- On line 1, the IF statement isn't validated since the id of the player is positive and different from null. Consequently, THEN section is skipped.
- On line 4, errorQuestion stores 2 questions (id 0 and 1).
- On line 5, the 2 questions in errorQuestion are added to the list list10Questions.
- On line 6, 8 random questions are added.
- On line 7, they are added to the list list10Questions.
- On line 8, this line returns list10Questions list with 2 questions which the player answered wrong and 8 random questions.

Algorithm 2

- On line 1, the IF statement isn't validated since the index is equal to 1 and is between 0 and 10. Consequently, THEN section is skipped.
- On line 4, the next question is the question at the index 1 which is the question with the id equal to 1.
- On line 5, the index is incremented.
- On line 6, RETURN returns this question (nextQuestion) with all data about it.

Using second example of data (Subsection 12.1)

Algorithm 1

- On line 1, the IF statement is validated since the id of the player is negative. The algorithm ends.

Using third example of data (Subsection 12.1)

Algorithm 1

- On line 1, the IF statement isn't validated since the id of the player is positive and different from null. Consequently, THEN section is skipped.
- On line 4, errorQuestion stores 2 questions (id 0 and 1).
- On line 5, the 2 questions in errorQuestion are added to the list list10Questions.
- On line 6, 8 random questions are added.
- On line 7, they are added to the list list10Questions.
- On line 8, this line returns list10Questions list with 2 questions which the player answered wrong and 8 random questions.

Algorithm 2

- On line 1, the IF statement is validated since the index is equal to -1 and is less than 0. The algorithm ends.

9.1.4 Data storage

Consequently the result will be this question, presented like this in the question table:

id	content	to_guess	type
1	Which color it is ?	2	AUDIO

in an Object Question(id, content, to_guess, type) :

```
Question(id = 1, content = "Which color it is ?", to\_guess = 2,
type = AUDIO")
```

10 Functional Testing

10.1 Test case: unit testing

10.1.1 UT-1

- **Title:** Incorrect email confirmation.
- **Test of:** testIsFormValidWithInCorrectData().
- **Description:** Unit test returns true if input email is incorrect.
- **Precondition:** Entered email is not valid.
- **Assumption:** Email and password input is of incorrect format, and is not a duplicate.
- **Test Data:**
  email: “123”
  password: “12345678”
- **Test Steps:**
  1. Navigate to com.example.lifruk (test) directory.
  2. Select LoginFragmentTest.kt
  3. Run ‘testIsFormValidWithInCorrectData’ method (or all associated unit tests by running the class).
- **Expected Result:** Test returns true.

10.1.2 UT-2

- **Title:** Check if entered login data is correct.
- **Test of**: testIsFormValidWithCorrectData().
- **Description:** Unit test returns true if email and password inputs are correct.
- **Precondition:** Login details (i.e. email and password) have been entered into their respective fields.

- **Assumption:** Email input is of correct format and is not a duplicate, password input is of correct format and is not a duplicate.
- **Test Data:**
email: “test@mail.com”
password: “12345678”
- **Test Steps:**
  1. Navigate to com.example.lifruk (test) directory.
  2. Select LogInFragmentTest class file.
  3. Run ‘testIsFormValidWithCorrectData’ method (or all associated unit tests by running the class).
- **Expected Result:** Test returns true.

10.1.3 UT-3

- **Title:** Signing up with correct data.
- **Description:** Unit test returns true if sign-up information is correct.
- **Precondition:** Database VM is running, sign-up input is correct.
- **Assumption:** Sign-up input is of correct format and is not a duplicate.
- **Test Data:**
username: "qwerty"
email: "test@mail.com"
password: "12345678"
password confirmation: "12345678"
native language: "English"
learning language: "English"
- **Test Steps:**
  1. Navigate to com.example.lifruk (test) directory.
  2. Select SignUpFragmentTest class file.
  3. Run ‘testIsFormValidWithCorrectData’ method (or all associated unit tests by running the class).
- **Expected Result:** Test returns true

10.1.4 UT-4

- **Title:** Signing up with incorrect data.
- **Description:** Unit test returns true if sign-up information is incorrect.
- **Precondition:** Database VM is running, sign-up input is incorrect.

- **Assumption:** none.
- **Test Data:**
  username: "qwerty"
  email: "test@mail.com"
  password: "12345678"
  password confirmation: "12345678"
  native language: "English"
  learning language: "Lithuanian".
- **Test Steps:**
  1. Navigate to com.example.lifruk (test) directory.
  2. Select SignUpFragmentTest class file.
  3. Run ‘testIsFormValidWithInCorrectData’ method (or all associated unit tests by running the class).
- **Expected Result:** Test returns true

10.2 Test case: Integration testing

10.2.1 IT-1

- **Test Case Objective/Title:** Check the interface link between the Login and Main Screen module.
- **Test Case Description:** A registered user should be able to enter login credentials and click on the Login button.
- **Precondition:** The user must already be registered with an email address and password.
- **Assumption:**
- **Test Steps:**
  1. Launch the application
  2. Navigate to ’Connection’ screen.
  3. In the ’Email’ field, enter the email address of the registered user.
  4. In the ’Password’ field, enter the password of the registered user.
  5. Click ’Login’
- **Expected Result:** To be directed to the Main Screen and shown "Successful login" message.

10.3 Test case: System testing

10.3.1 ST-1

- **Test Case Objective/Title:** System test for registration of new users

- **Test Case Description:** User should be able to enter new account data and create an account in the Android app.
- **Precondition:** Username used for registration should be unique (should not be taken)
- **Assumption:** The user, webserver and database are all connected to the Internet.
- **Test Steps:**
  1. Launch the application
  2. Click Connection button.
  3. Click on "Sign up" button
  4. Enter username and password for a new account.
- **Expected Result:** To be directed to the Main Screen and shown "Successful account creation and login" message.

10.4 Test case: Acceptance testing

10.4.1 AT-1

- **Test Case Objective/Title:** Retrieving word topic list.
- **Test Case Description:** The user should be presented with a list of available word topics.
- **Precondition:** User has chosen preferred mode (either 'Flashcard' or 'Quiz')
- **Assumption:** : A user's device is connected to the internet.
- **Test Steps:**
  1. Launch application.
  2. Click either 'Quiz' or 'Flashcard' button on the main screen.
- **Expected Result:** A screen showing a list of available topics should be shown.

10.4.2 AT-2

- **Test Case Objective/Title:** Retrieving quiz questions
- **Test Case Description:** The user should be presented with a question along with pictures depicting possible answer for said question.
- **Precondition:** The user must have selected a topic.
- **Assumption:** A user's device is connected to the internet.
- **Test Steps:**
  1. Launch application.
  2. Click 'Quiz' button on the main screen.
  3. Select desired topic.
- **Expected Result:** The first question for the selected topic is shown with the pictures.

11 Non-Functional Testing

11.1 Test case: Usability testing

- **Test Case Objective/Title:** Check how easy and user-friendly the application is.
- **Test Case Description:** Application should be easy and understandable for user.
- **Precondition:** Users must have mobile phones which they are going to use to launch the app.
- **Test Steps:**
  1. Launch the application.
  2. Check the convenience of colors, test buttons.
  3. Analyze app usability.
- **Expected Result:** Application is understandable and easy to use for the user.

11.2 Test case: compatibility testing

- **Test Case Objective/Title:** Check if information on the screen is displayed correctly.
- **Test Case Description:** User should be able to see the app displayed correctly. App must be able to optimize screen layout for different sizes of screens.
- **Precondition:** The device must be on.
- **Test Steps:**
  1. Using mobile emulator, create 3 devices with different display sizes: 4.95", 5.5", 6" (inches, diagonally).
  2. Run app on the first device and test.
  3. Run app on the second device and test.
  4. Run app on the third device and test.
- **Expected result:** App is optimized on different devices and screen layouts are displayed correctly.

11.3 Test case: GUI

- **Main screen:**

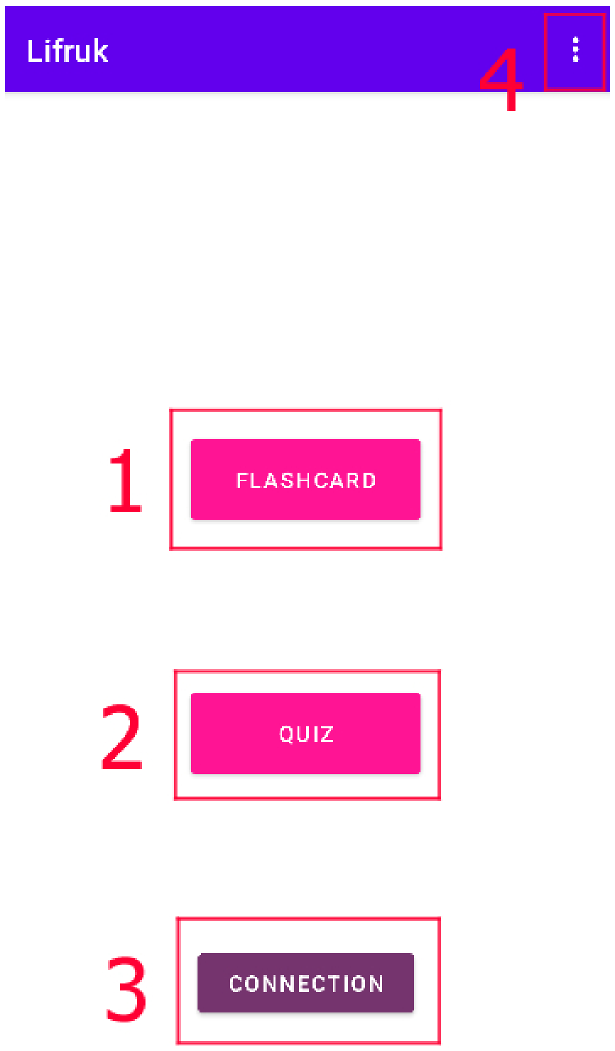

Figure 8. *Main Screen*

1. Flashcard button
   - Check if the button can be clicked.
   - Test redirection to the list of topics.
   - Check the readability.
2. Quiz button
   - Check if the button can be clicked.
   - Test redirection to the list of topics.
   - Check the readability.
3. Connection button
   - Check if the button can be clicked.
   - Check the readability.
4. Settings button

– Check if the button can be clicked.
– Check the readability.

- **Login screen:**

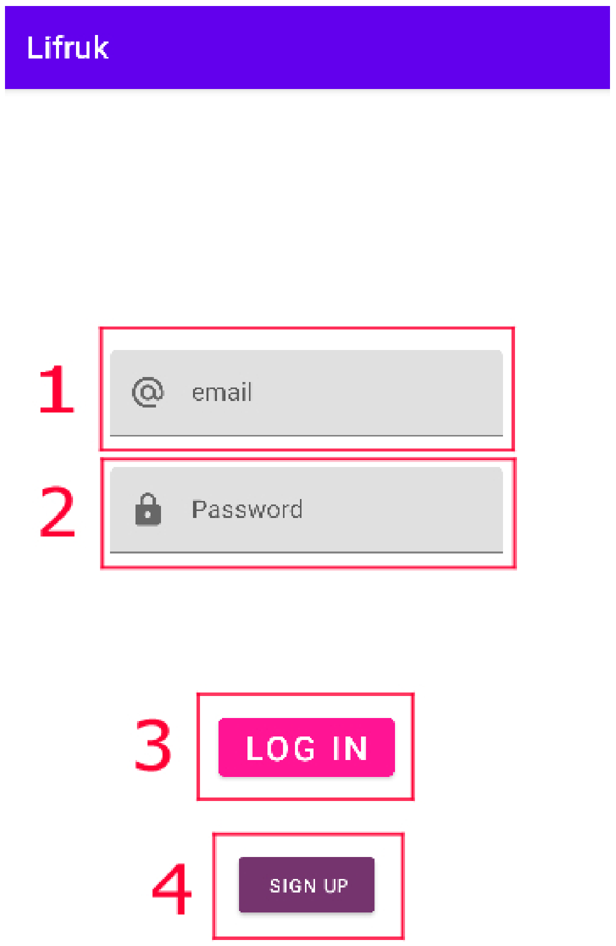

Figure 9. *Login Screen*

1. Email field
   – Validate the acceptance of both valid and invalid characters
   – Check the readability.
2. Password field
   – Validate the acceptance of both valid and invalid characters
   – Check the readability.
3. Log in button
   – Check if the button can be clicked.

- Test form submission
- Check the readability.

4. Sign up button
   - Check if the button can be clicked.
   - Check the readability.

Conclusion

In conclusion, the android quiz app is a useful and engaging tool for helping children learn new languages and improve their vocabulary. It offers a variety of quiz modes and customization options to cater to different learning styles and preferences. The app is also easy to use and has a visually appealing and intuitive interface, making it accessible and enjoyable for children.

Overall, the development of this words learning app has been a success. The use of the MVVM architecture and the integration of a web server and PostgreSQL database has allowed for a scalable and modular design, making it easy to maintain and update the app over time. The app still has several bugs that were not fixed but the core mechanics still work. Group communication has definitely been a challenge as there were several language barriers we had to overcome.

Moving forward, it will be important to continue improving and expanding the app's features and functionality to meet the evolving needs of its users. This could include adding new languages and quiz modes, integrating additional learning resources, or adding new features to support user engagement and retention.

12 Appendices

12.1 Example Data

- **Example 1**
  - *player*: 1
  - *index*: 1
  - *topicName*: "Color"
- **Example 2**
  - *player*: -1
  - *index*: 1
  - *topicName*: "Number"
- **Example 3**
  - *player*: 1
  - *index*: -1
  - *topicName*: "Color"
- **Both**
  Player

id	username	email	pwd	native_l	learning_l
0	Fio	fiona@gmail.com	ApzindzZ	FRENCH	ENGLISH

Game

id	player	topicName
0	0	Color
1	0	Number
2	0	Color

Word

id	language	content	pronunciation	topicName	picture
0	ENGLISH	Red		Color	
1	ENGLISH	Blue		Color	
2	ENGLISH	Yellow		Color	
3	ENGLISH	Gray		Color	
4	ENGLISH	One		Number	
5	ENGLISH	Two		Number	

Question

id	content	to_guess	type
0	Which color it is ?	1	AUDIO
1	Which color it is ?	2	AUDIO
2	Which color it is ?	3	AUDIO
3	Which color it is ?	4	AUDIO
4	Which number it is ?	5	AUDIO
5	Which number it is ?	6	AUDIO

Answer

id	question	game	correct
0	0	0	0
1	1	0	0
2	2	0	1
3	3	0	1
4	4	1	0
5	5	1	1