Word Master 3000

Technical Manual

Submitted by

Syed Yusuf Ahmed (BSSE **1016**)

Radowan Mahmud Redoy (BSSE **1001**)

IIT
University of Dhaka

Date: August 28,2021

Project:	Word Master 3000
Developers:	Syed Yusuf Ahmed (Exam roll: 1016) Radowan Mahmud Redoy (Exam roll: 1001)
Submitted:	26/8/2021
Supervised by:	Dr. Kazi Muheymin-Us Sakib Professor, Institute of Information Technology, University of Dhaka.
Supervisor's Signature:	[illegible]

Acknowledgement

We are highly indebted for getting such a wonderful opportunity to prepare the User Manual report of the project ‘Word Master 3000’. We would like to thank whole-heartedly to our supervisor, Dr. Kazi Muheymin-Us-Sakib, Professor, Institute of Information Technology, University of Dhaka for giving us guidelines on preparing the project and this report.

Abstract

Word Master 3000, or WM3K for short, is a mobile based application which aims to help/guide the user achieve mastery over English words. The app will help users learn spelling, word meaning, multiple meanings of words, use of words in sentences, spelling, synonyms, etc., using standard memorization techniques, short training session, games and quizzes. This app will be particularly helpful for students who find memorizing word meanings difficult, and it can help students prepare their vocabulary for competitive exams like GRE, TOEFL, BCS, university admission tests, etc. In addition, this will boost the preparation of Spelling Bee contestants so that they might make it to the international level.

To build this app, we started with extracting requirements from a sample of intended end users. We analyzed the requirements and formulated an SRS documentation which we then implemented in code. The android application has been coded with Flutter framework and Java. The dictionary database has been coded in Java, and the server-side programming has been coded in Typescript. The output is a full-fledged application powered by a cloud server. We aim to release our app in the play store and hope that it will gain popularity among students and contestants.

Contents

Introduction

Many students, especially the ones good at analytical tasks, do not like memorization. As a result, they find it very problematic to memorize hundreds of word meanings as preparation for competitive exams like GRE, university admission test, BCS, etc. Besides, there are many international level achievements in fields of science and technology from Bangladesh like achievements in international physics Olympiad, math Olympiad, robotics Olympiad, etc. However, there is none in English Language based competitions like Spelling Bee. These are the problems we aim to solve with WM3K. By bringing different learning tricks together and with tons of features like memorization sessions, quizzes, floating bubble, games, creating and sharing courses, etc., WM3K hopes to make vocabulary fun and easy to improve. With AI generated training sessions, we ensure that our users get the best long-term training possible. WM3K is the perfect companion app for any learner.

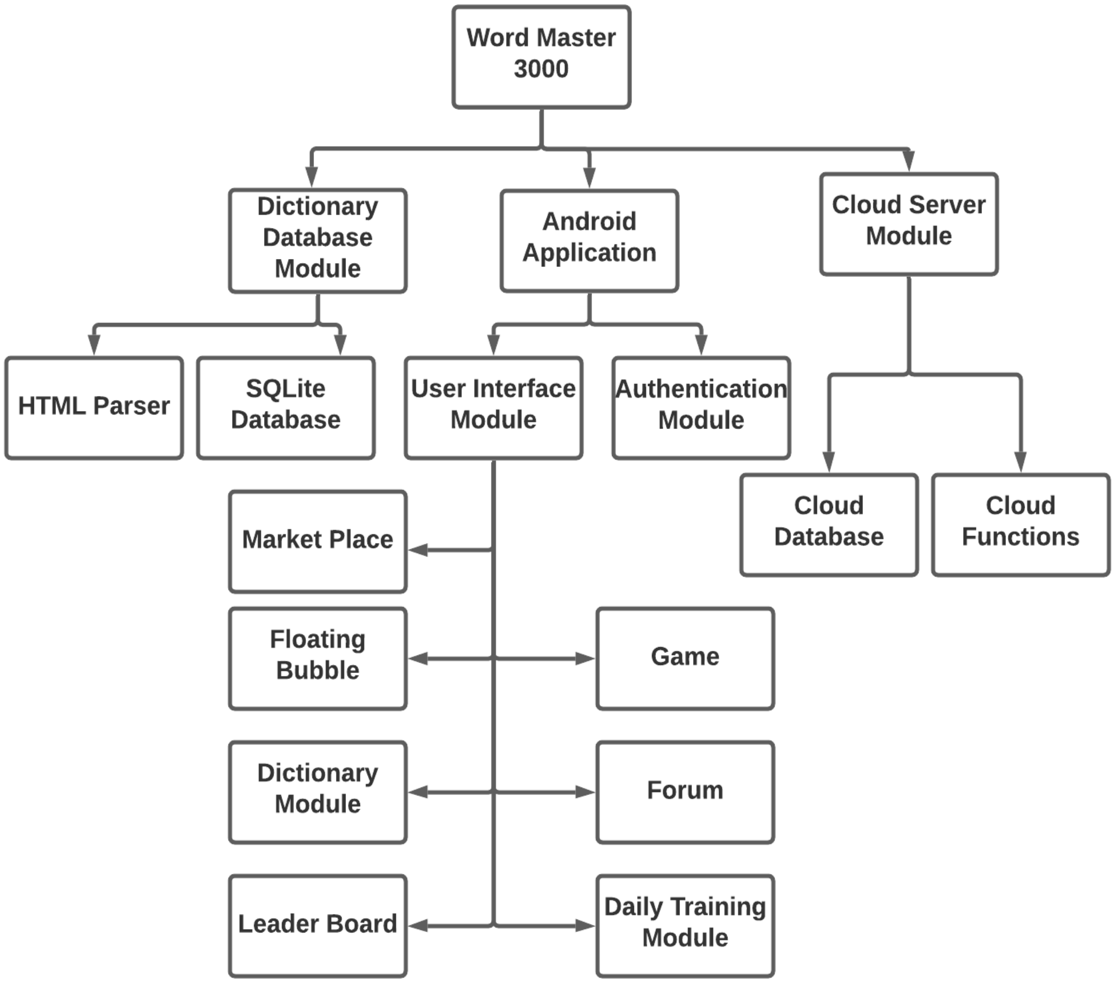
Features

- ✔ Offline dictionary of over 77,500 words with parts of speech, meaning, examples, synonyms and related words, idioms and phrases.
- ✔ Search on copy
- ✔ View meaning on floating bubble for quick and easy access
- ✔ Search and store words in word-lists
- ✔ Create and share courses
- ✔ Enroll and unenroll in courses
- ✔ Memorize word meanings by repetition
- ✔ Practice word's spellings
- ✔ Take quiz
- ✔ Fun game like spelling master
- ✔ Leaderboard
- ✔ Daily training
- ✔ Training summary of daily training
- ✔ Forum
- ✔ Create post, preview post, delete post
- ✔ Showing appreciation on a post
- ✔ Write comment and delete comment on a post
- ✔ Check activity and notification log
- ✔ Search for a user

Explanation Of Implementation

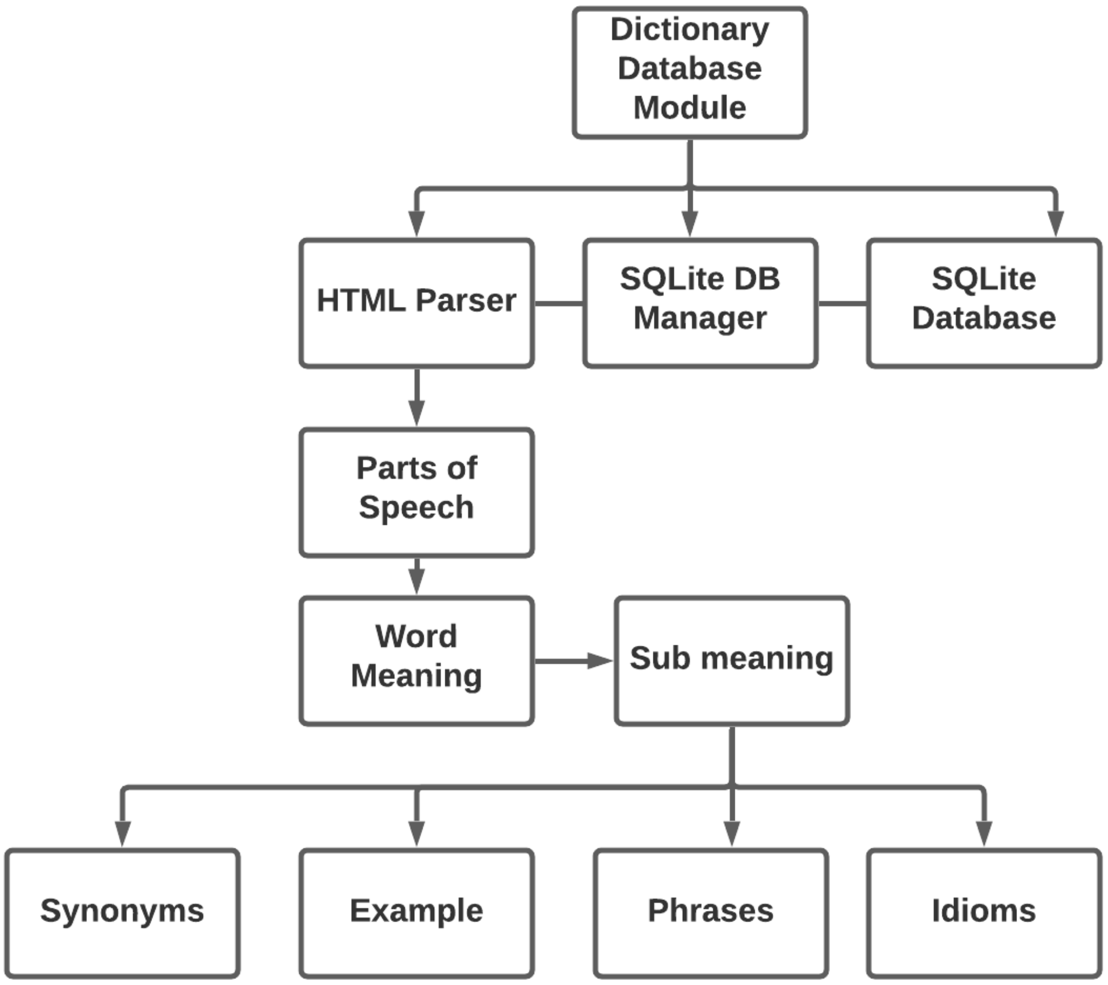
A high-level overview of our project implementation is given below. In the following diagrams, each box stands for a module or functionality. The lines denote "contains" relationships. For example, Word Master 3000 contains a Dictionary database module, Android application, and Cloud server module.

Top Level Overview of Word Master 3000



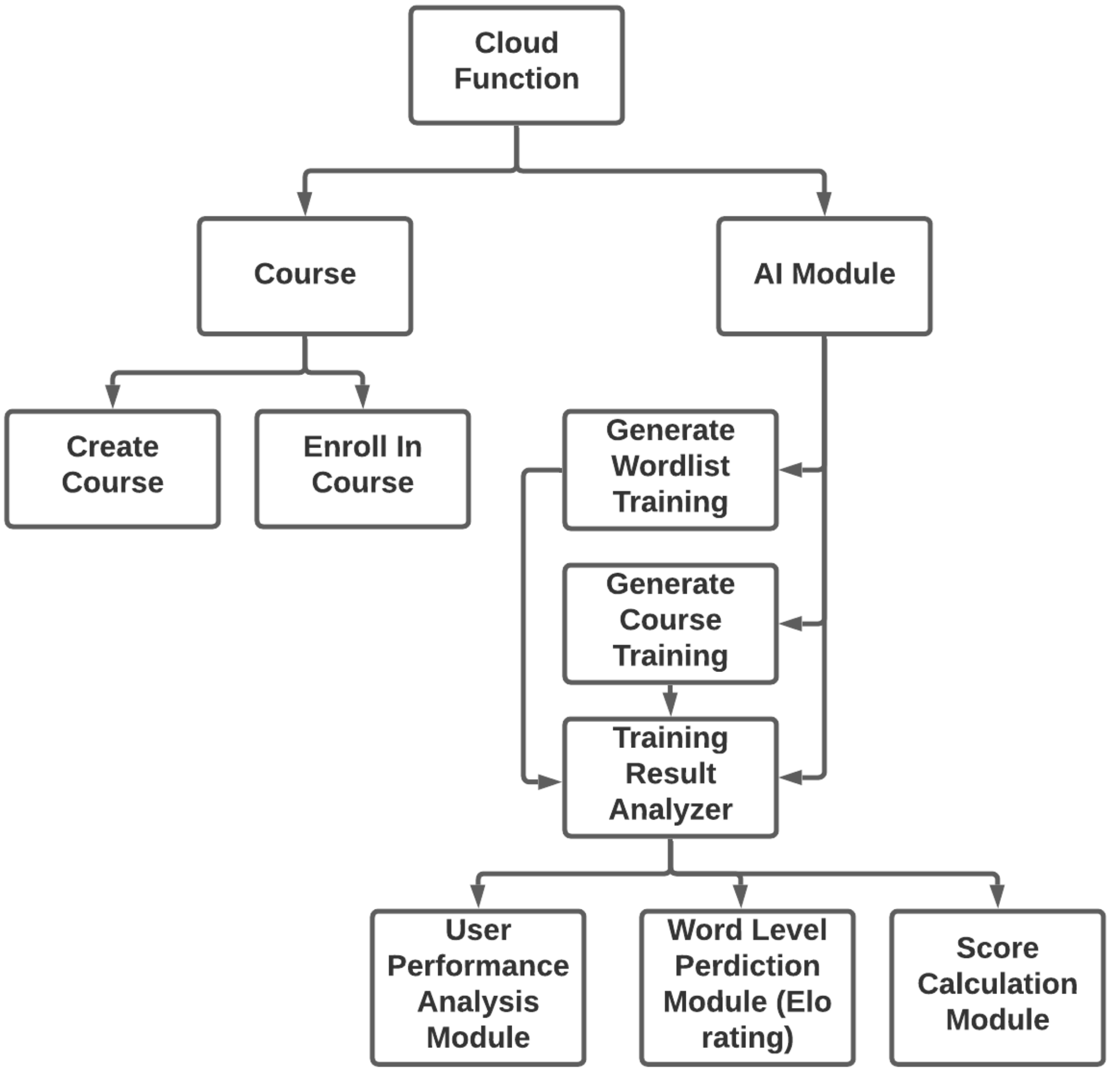
The diagram gives a glimpse of all the submodules in this project.

Dictionary Database Module



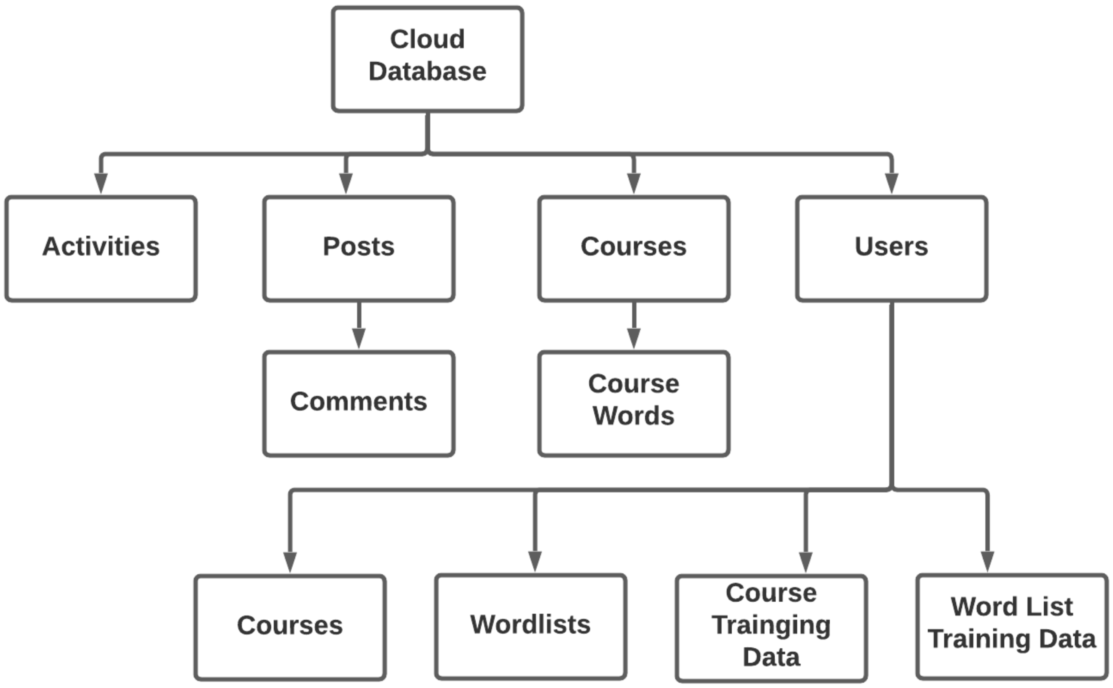
The above diagram further demonstrates the dictionary database module which has 3 sub modules: HTML parser, SQLite DB Manager, SQLite Database. This code is mainly responsible for querying the online Cambridge Dictionary database, parsing the HTML, and storing the result in the database.

Cloud Functions



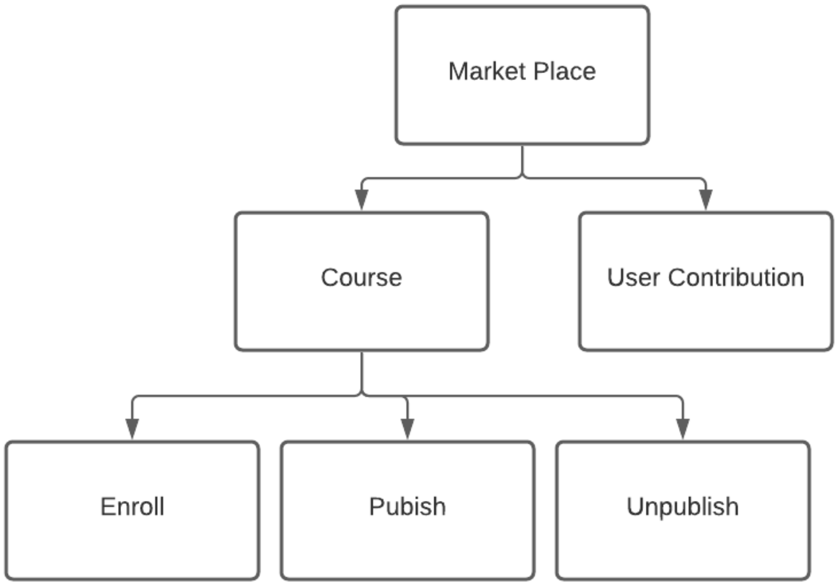
The diagram demonstrates the cloud functions. Mainly it is divided into two parts - course related functions and AI Module. The AI module further divides into 3 interconnecting parts which uses various functionalities to generate results.

Cloud Database



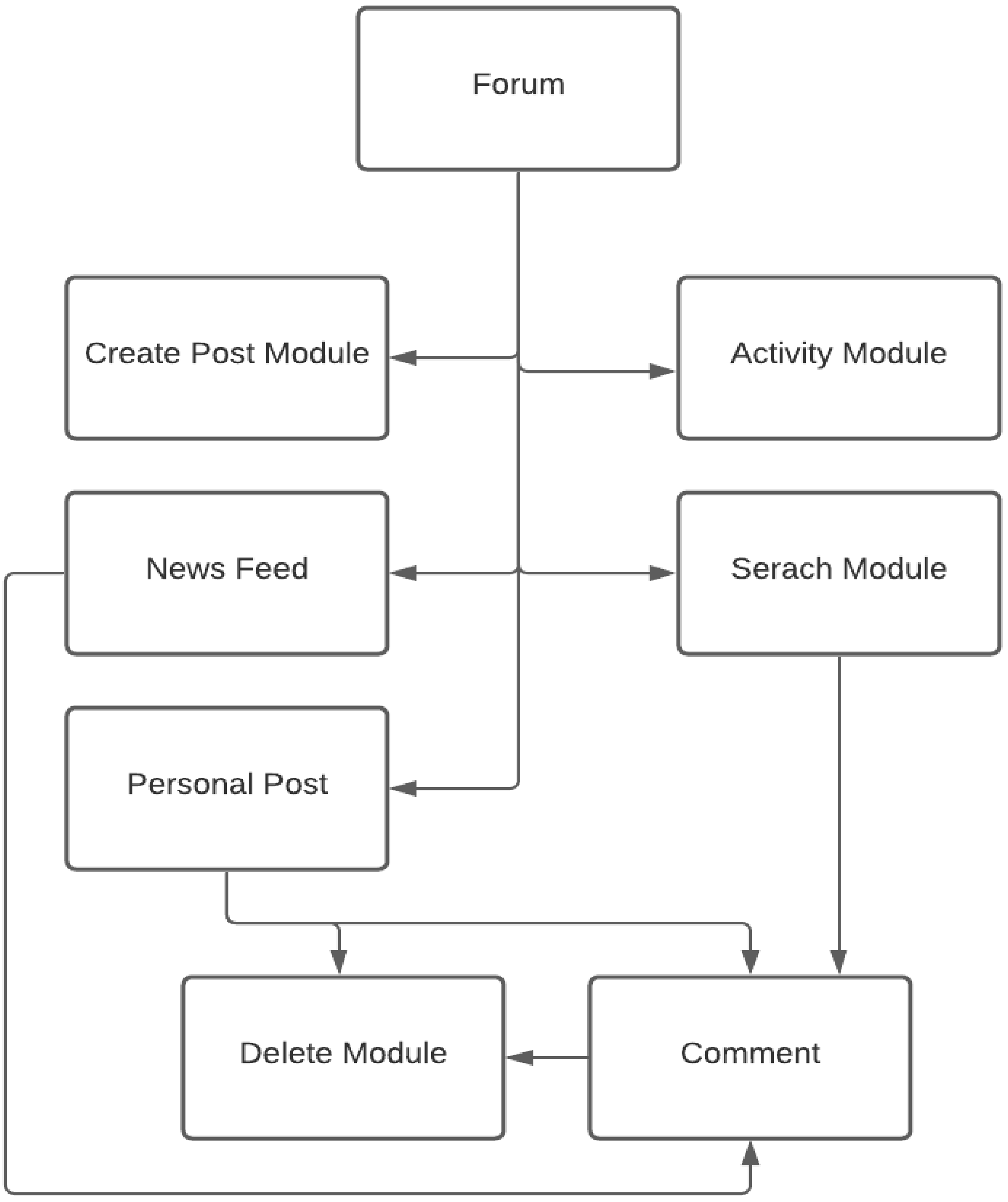
The diagram shows the collections of cloud databases. There are 4 main collections Activities, Posts, Courses, Users and each of these further carries sub collections.

Market Place



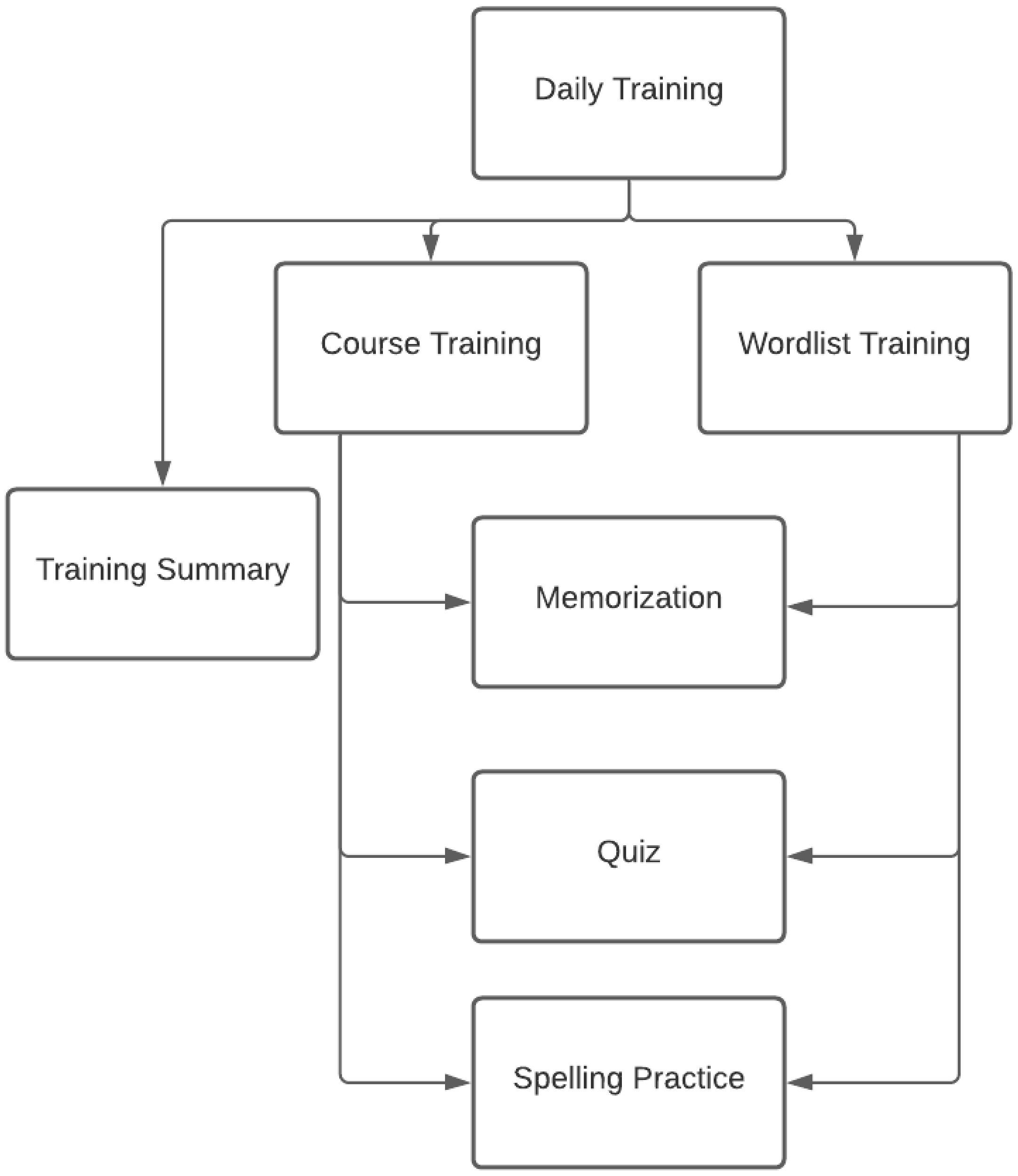
The marketplace has two parts, a new course and a User contribution. Users can publish, enroll, and unpublish a course. The "User contribution" module shows what courses they have published in the marketplace. It also gives the users the option to unpublish the course if they wish to.

Forum



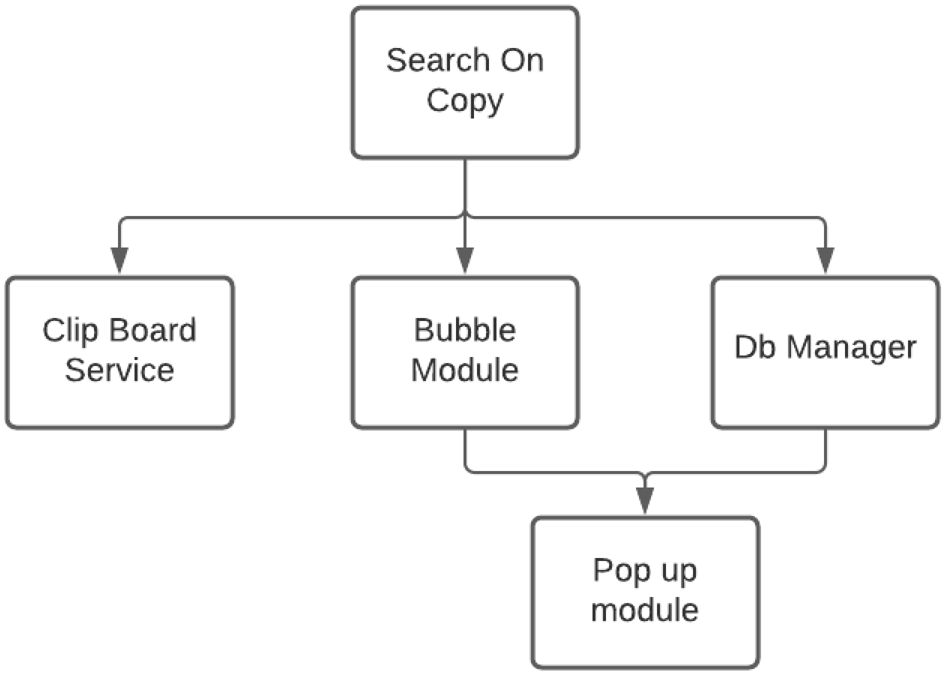
The diagram demonstrates the forum module which has 5 main parts - create post module, news feed, personal post, search module and activity module. News feed, personal post and search module share two submodules which are delete module and comment.

Daily Training



Daily training has 3 main parts. Course training, Wordlist Training and both of them have 3 common submodules Memorization, Quiz, Spelling practice. Daily training sessions consist of course training and word-list training, each of which involve the three steps – memorization, quiz, and spelling practice. At the end of daily training, a training summary is shown.

Search On Copy



The module is divided into three parts, Clipboard Service, Bubble Module and Db Manager where Bubble Module and Db Manager share a common submodule Popup module. The clipboard service is responsible for listening for changes in the clipboard. As soon as the user copies something, the clipboard service notifies the bubble module. The bubble module queries the database through the popup module and displays the result.

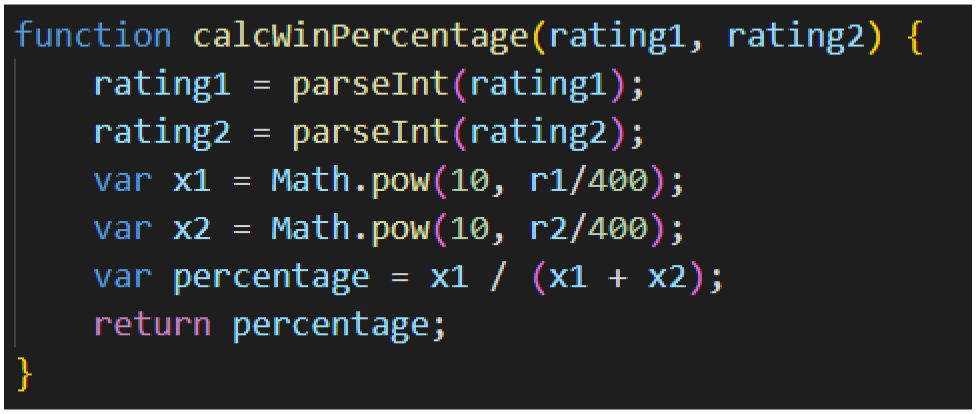
Explanation of code

Cloud functions

Cloud functions are functions that have been deployed to the server as server-side scripting. The code for this module has been written in structured language (structured side of typescript).

AI Module

The AI's task is to make the best possible training routine for users. For this, it firstly analyzes how well the user is doing in his daily training **(User performance analysis module)**. Based on user's performance it predicts how likely the user is to be able to recall a word correctly **(Word level prediction module) –**



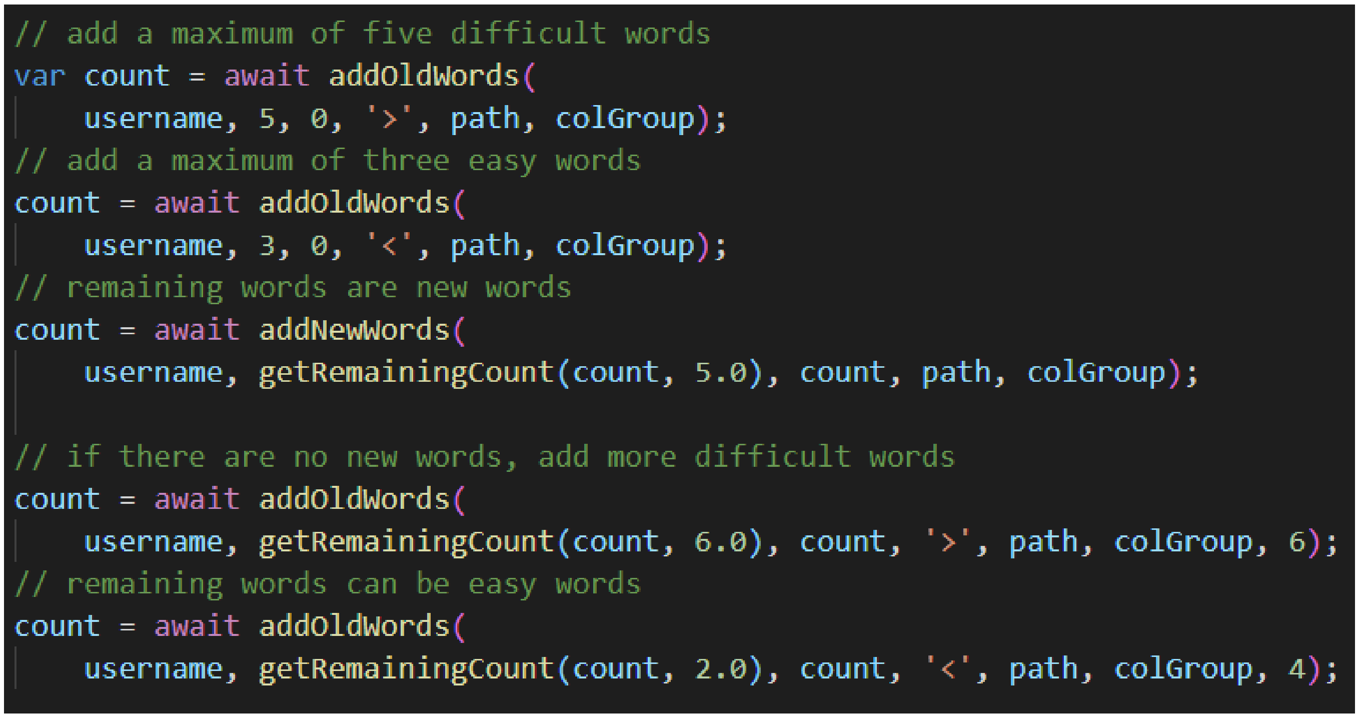
```
function calcWinPercentage(rating1, rating2) {
    rating1 = parseInt(rating1);
    rating2 = parseInt(rating2);
    var x1 = Math.pow(10, r1/400);
    var x2 = Math.pow(10, r2/400);
    var percentage = x1 / (x1 + x2);
    return percentage;
}
```

Here rating1 is the rating of the user and rating2 is the rating of the word. The higher the rating, the more difficult the word is.

If the predicted success rate is 50%, it means the word is new. The user may or may not know the word. Such words are called **new words**.

The closer the rate is to 100%, the more familiar the user is with the word. Such words are called **easy words**.

If the predicted rate is less than 50%, it means the user has associated a wrong meaning with the word and is more likely to answer incorrectly. Such words are called **difficult words**.

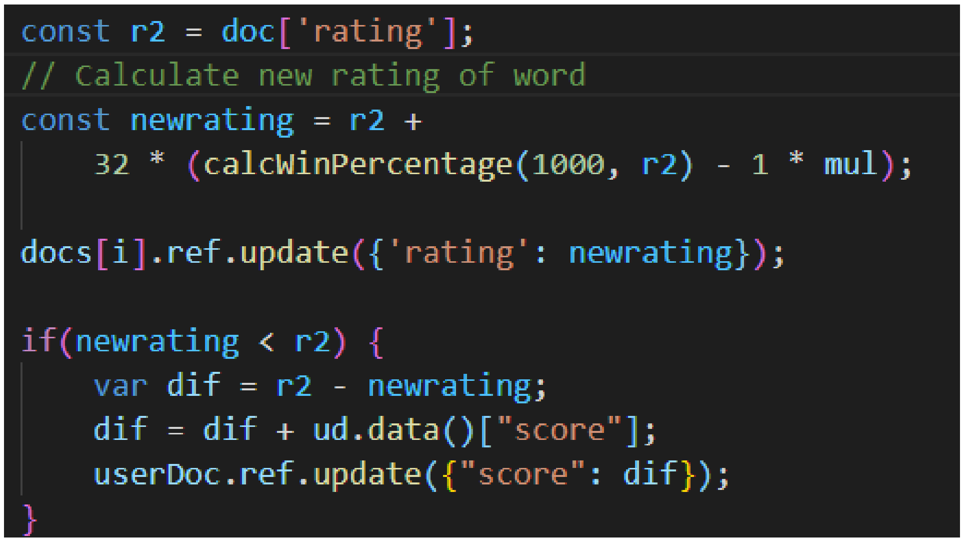


```
// add a maximum of five difficult words
var count = await addOldWords(
    username, 5, 0, '>', path, colGroup);
// add a maximum of three easy words
count = await addOldWords(
    username, 3, 0, '<', path, colGroup);
// remaining words are new words
count = await addNewWords(
    username, getRemainingCount(count, 5.0), count, path, colGroup);

// if there are no new words, add more difficult words
count = await addOldWords(
    username, getRemainingCount(count, 6.0), count, '>', path, colGroup, 6);
// remaining words can be easy words
count = await addOldWords(
    username, getRemainingCount(count, 2.0), count, '<', path, colGroup, 4);
```

The snapshot above shows how a training session is generated. A training session can have a maximum of 26 words. Each easy word is repeated two times, hard words three times, and new words five times. The repetition may help the user better remember the word.

Once the user completes a training session, his results are matched against the predictions made by the algorithm. Then the difficulty level of the words is updated (More difficult words have higher rating). The user's score is updated based on how much he performed above the predictions of the algorithm as shown below –



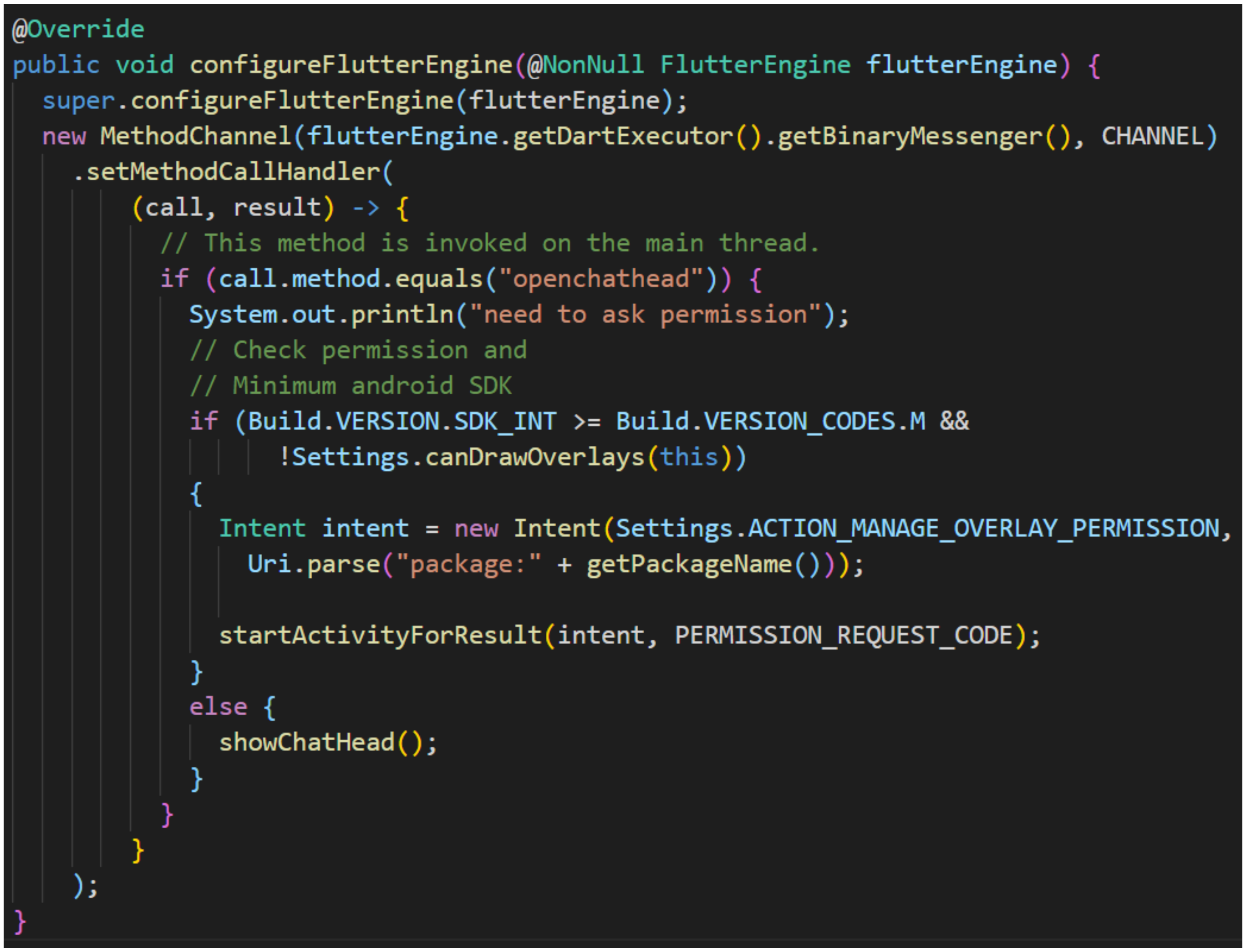
```
const r2 = doc['rating'];
// Calculate new rating of word
const newrating = r2 +
    32 * (calcWinPercentage(1000, r2) - 1 * mul);

docs[i].ref.update({'rating': newrating});

if(newrating < r2) {
    var dif = r2 - newrating;
    dif = dif + ud.data()["score"];
    userDoc.ref.update({"score": dif});
}
```

Search-on-copy module

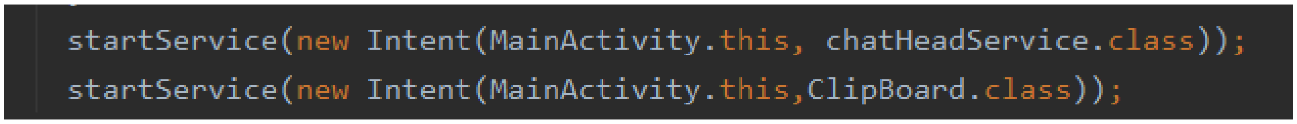
Search on copy allows users to see the meaning of words from a floating bubble upon copying the word to clipboard. For this, our project uses android native to run background services. A channel is created between the app and native android service. The app passes a string to the service that activates it.



```
@Override
public void configureFlutterEngine(@NonNull FlutterEngine flutterEngine) {
  super.configureFlutterEngine(flutterEngine);
  new MethodChannel(flutterEngine.getDartExecutor().getBinaryMessenger(), CHANNEL)
    .setMethodCallHandler(
        (call, result) -> {
          // This method is invoked on the main thread.
          if (call.method.equals("openchathead")) {
            System.out.println("need to ask permission");
            // Check permission and
            // Minimum android SDK
            if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M &&
                  !Settings.canDrawOverlays(this))
            {
              Intent intent = new Intent(Settings.ACTION_MANAGE_OVERLAY_PERMISSION,
                Uri.parse("package:" + getPackageName()));

              startActivityForResult(intent, PERMISSION_REQUEST_CODE);
            }
            else {
              showChatHead();
            }
          }
        }
    );
}
```

The app displays meaning using a pop up. In order to show popup over other services 'Display over other services permission is needed'. The app first checks whether the permission is given. If required permission is given then it starts all necessary background services.

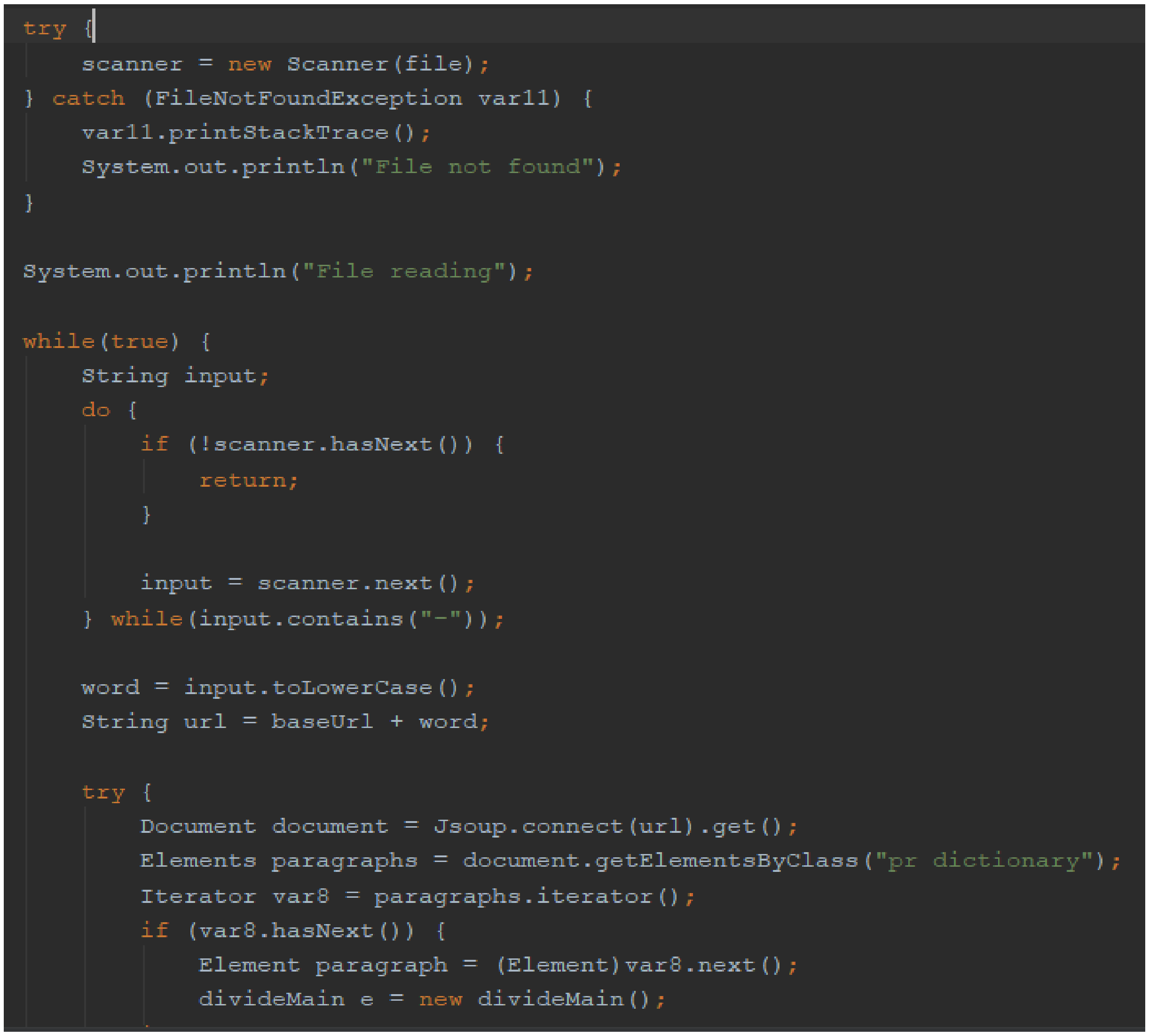


```
startService(new Intent(MainActivity.this, chatHeadService.class));
startService(new Intent(MainActivity.this,ClipBoard.class));
```

From the "showChatHead ()" method, all necessary background services are called which are needed for search on copy functionality.

Dictionary Database Module

We have created a separate java project in order to create our dictionary database. The java project mainly acted as a html parser which helped us to collect word meanings, examples, idioms, phrases, synonyms from Cambridge dictionary.



```
try {
    scanner = new Scanner(file);
} catch (FileNotFoundException var11) {
    var11.printStackTrace();
    System.out.println("File not found");
}

System.out.println("File reading");

while(true) {
    String input;
    do {
        if (!scanner.hasNext()) {
            return;
        }

        input = scanner.next();
    } while(input.contains("-"));

    word = input.toLowerCase();
    String url = baseUrl + word;

    try {
        Document document = Jsoup.connect(url).get();
        Elements paragraphs = document.getElementsByClass("pr dictionary");
        Iterator var8 = paragraphs.iterator();
        if (var8.hasNext()) {
            Element paragraph = (Element)var8.next();
            divideMain e = new divideMain();
```

We managed to collect more than 77500 words for our dictionary database along with their parts of speech, sub meanings, examples, idioms, phrases and synonyms.

PROJECT SET UP

To run the current version of our application, pull source code from the current folder. Open the folder with android studio. The host must have flutter 2.0 installed in his/her machine. Then, get all dependencies running 'flutter pub get' and then run the application.

Achievements

Compiling an independent, offline dictionary database of Cambridge Dictionary containing word meanings, parts of speech, examples, synonyms and related words, idioms and phrases, etc.

Learning the mathematics of ELO rating system and adapting it to the context of this project for AI generated training sessions.

Exploring UI/UX design.

Gaining practical experience of extracting, analyzing, and refining requirements from stakeholders.

Planning, implementing, and maintaining a large project.

Understanding project management and teamwork.

Learning android development.

Challenges

Designing professional grade UX and lucrative UI with vibrant colors, original design and fluid animations.

Preventing background services from being killed by Android OS.

Connecting flutter project with native android channel for native functionalities.

Lack of a dictionary database with satisfactory quality; No comprehensive list of all words in English Language.

Running dictionary database compiler code continuously for days and managing unexpected stops.

Future plan

Releasing the app to the play store.

Analyzing all user's daily training data to create a neural network AI for best predictions.
Analyzing all user's daily training data to identify what factors affect memorization ability and how.
Implementing more games with 2D and 3D graphics, multiplayer options, etc.

Conclusion

This project was our first attempt to develop a full scale mobile based application. In this journey, we faced many obstacles from generating a dictionary database to proper implementation of training mechanisms. We have learnt how to deal with unseen problems and come up with proper solutions. The end result is that we have a complete android application that fulfills the requirements in the SRS. Now our future aim is to make this app reachable for all users so that they can use it in their day-to-day life and gain the intended benefits from our application.

GitHub links

https://github.com/yusf1013/wordmaster3000