CHAPTER-1

INTRODUCTION

1.1 DEVELOPING ANDROID APPLICATION FOR SIMPLE DICTIONARY ANDROID APP:

The Simple Dictionary Android App is an innovative mini project developed using Android Studio, a powerful integrated development environment (IDE) specifically designed for Android app development. This report provides a comprehensive overview of the project, highlighting its objectives, features, and implementation details.

In today's fast-paced world, effective communication and a strong vocabulary are crucial skills. The Simple Dictionary Android App aims to address this need by offering a convenient and user-friendly tool for looking up word definitions directly from Android devices. With the increasing popularity of mobile devices, having a reliable dictionary app readily available on smart phones can greatly enhance language learning and improve communication skills.

The primary objective of the Simple Dictionary Android App is to provide users with quick and easy access to word definitions. By offering a simple and intuitive interface, the app ensures that users can effortlessly search for words and retrieve accurate definitions in a matter of seconds. This feature makes the app suitable for users of all age groups and proficiency levels, from students looking to expand their vocabulary to professionals seeking precise word meanings.

PROCEDURE TO RUN THE APPLICATION:

Insertion of a data to the database:

1. Input any word in place of enter a word.

For example: apple, bat, cat and so on.

1. Button: Enter the search button.
2. Result:
3. If any word is present in the dictionary. Then, the meaning of the word is displayed.
4. If any number is entered. Then, the output displays- please enter a word.
5. If any word is entered. Then, that word is not present in the dictionary(DB) then the output displays- The word is not present in the dictionary. We will try to include it to the dictionary.

1.2 INTRODUCTION TO ANDROID:

Android is an open source and Linux-based Operating System for mobile devices such as smart phones and tablet computers. Android was developed by the *Open Handset Alliance*, led by Google, and other companies.

Android offers a unified approach to application development for mobile devices which means developers need only develop for Android, and their applications should be able to run on different devices powered by Android.

The first beta version of the Android Software Development Kit (SDK) was released by Google in 2007 where as the first commercial version, Android 1.0, was released in September 2008.

On June 27, 2012, at the Google I/O conference, Google announced the next Android version, 4.1 Jelly Bean. Jelly Bean is an incremental update, with the primary aim of improving the user interface, both in terms of functionality and performance. The source code for Android is available under free and open source software licenses.

Google publishes most of the code under the Apache License version 2.0 and the rest, Linux kernel changes, under the GNU General Public License version.

### 1.3 FEATURES OF ANDROID:

Android is a powerful open-source operating system that open-source provides immense features and some of these are listed below;

1. Beautiful UI: Android OS basic screen provides a beautiful and intuitive user interface.
2. Connectivity: GSM/EDGE, IDEN, CDMA, EV-DO, UMTS, Bluetooth, Wi-Fi, LTE, NFC and WiMAX.
3. Storage: SQLite, a lightweight relational database, is used for data storage purposes.
4. Media support: H.263, H.264, MPEG-4 SP, AMR, AMR-WB, AAC, HE-AAC, AAC 5.1, MP3, MIDI, OggVorbis, WAV, JPEG, PNG, GIF, and BMP.
5. Messaging: SMS and MMS.
6. Web browser: Based on the open-source Web Kit layout engine, coupled with Chrome's V8 JavaScript engine supporting HTML5 and CSS3.
7. Multi-tasking: User can jump from one task to another and same time various application can run simultaneously.
8. Resizable widgets: Widgets are resizable, so users can expand them to show more content or shrink them to save space.
9. Multi-Language: Supports single direction and bi-directional text.
10. GCM: Google Cloud Messaging (GCM) is a service that lets developers send short message data to their users on Android devices, without needing a proprietary sync solution.
11. Android development toolkit: Android is a development tool that lets you perform these tasks: Manage Android Virtual Devices (AVD) Create and update Android projects. Update your sdk with new platform add-ons and documentation. A Software Development Kit, or an SDK, is a collection of tools that you need to develop an application for a specific software framework. For example, to develop applications in Java, you need a Java SDK (JDK). SDKs contain binaries, source code for the binaries, and documentation for the source code.
12. SDK tools: SDK tools are generally platform independent and are required no matter which android platform you are working on. When you install the Android SDK into your system, these tools get automatically installed.

CHAPTER-2

INTRODUCTION TO FRONT END SOFTWARE

INTRODUCING ANDROID STUDIO:

Android Studio is the official Integrated Development Environment (IDE) for Android app development. Based on the powerful code editor and developer tools from [IntelliJ IDEA](https://www.jetbrains.com/idea/), Android Studio offers even more features that enhance your productivity when building Android apps.

INSTALLING AND RUNNING APPLICATION ON ANDROID STUDIO:

### **Installation Guide:**

**Step 1:**Head over to [this link](https://developer.android.com/studio/#downloads) to get the Android Studio executable or zip file.

**Step 2:**Click on the **Download Android Studio** Button.

Click on the “I have read and agree with the above terms and conditions” checkbox followed by the download button.

Click on the Save file button in the appeared prompt box and the file will start downloading.

**Step 3:**After the downloading has finished, open the file from downloads and run it. It will prompt the following dialog box.

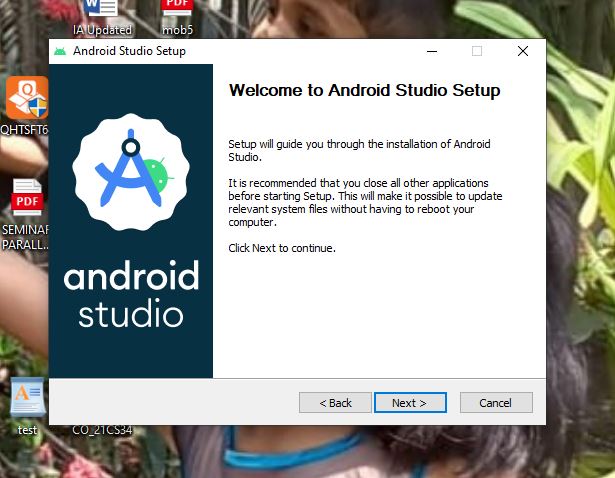


Fig. 2.1: Setup Android Studio.

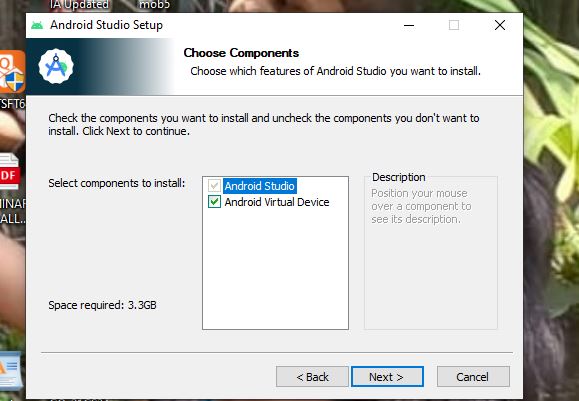
Click on next. In the next prompt, it’ll ask for a path for installation. Choose a path and hit next.

Fig. 2.2: Selecting the Necessary Components to be Installed.

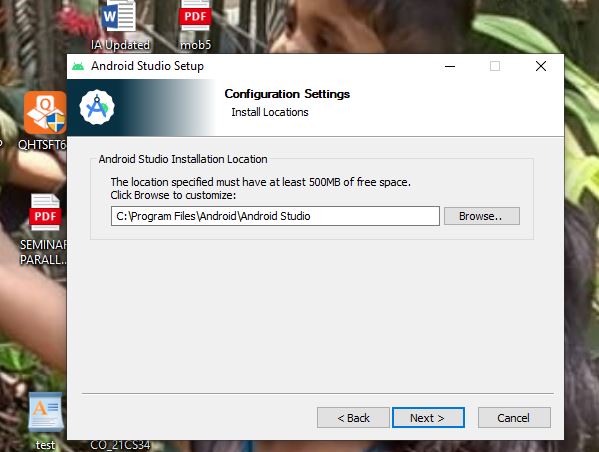


Fig. 2.3: Selecting where the Android Studio could be Installed.

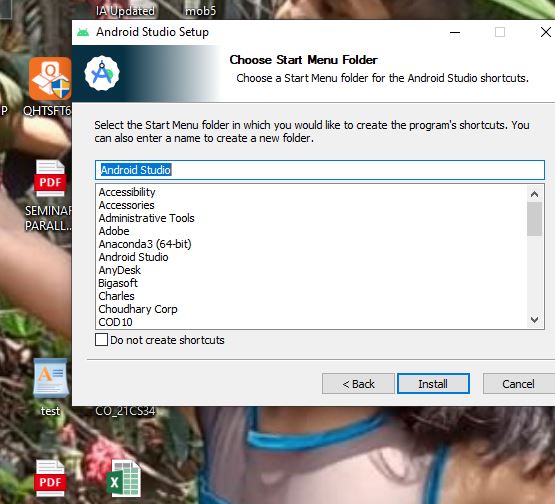


Fig. 2.4: Choosing a Start Menu Folder.

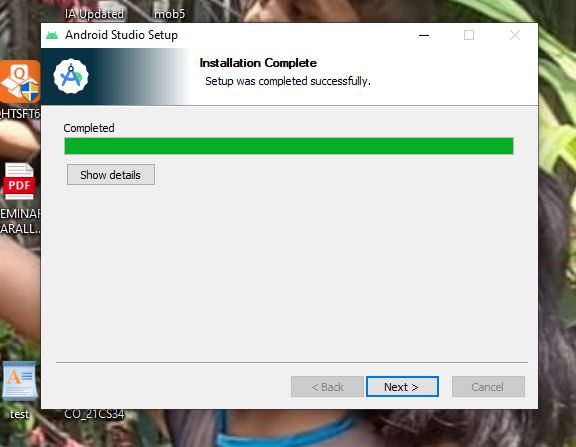
Step 4: It will start the installation, and once it is completed, it will be like the image shown below.

Fig. 2.5: Installation Completed Successfully.

**Step 5:**Once “**Finish**” is clicked, it will ask whether the previous settings need to be imported [if the android studio had been installed earlier], or not. It is better to choose the ‘Don’t import Settings option’. Click the **OK**button.

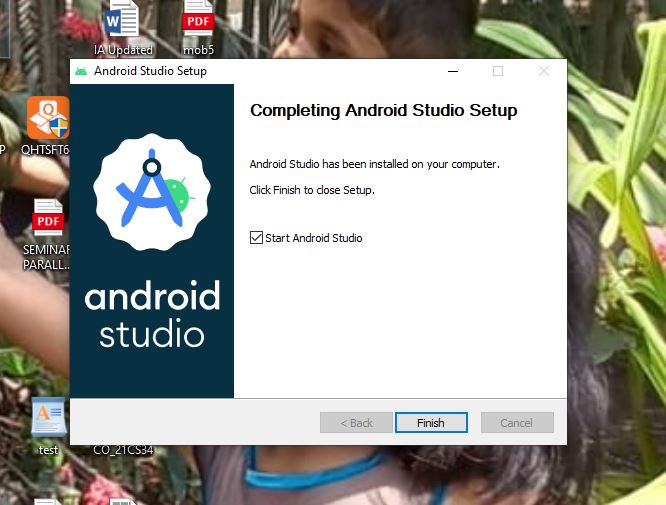


Fig. 2.6: Android Studio Setup Completed.

**Step 6:**This will start the Android Studio.

Meanwhile, it will be finding the available SDK components.

Fig. 2.7: Starting the Android Studio.

**Step 7:**After it has found the SDK components, it will redirect to the Welcome dialog box.

Click on **Next**.

Choose Standard and click on Next. Now choose the theme, whether the **Light**theme or the **Dark**one. The light one is called the **IntelliJ**theme whereas the dark theme is called **Dracula**. Choose as required.

Click on the **Next**button.

**Step 8:**Now it is time to download the SDK components.

Click on Finish. Components begin to download let it complete.

The Android Studio has been successfully configured. Now it’s time to launch and build apps. Click on the Finish button to launch it.

**Step 9:**Click on **Start a new Android Studio project** to build a new app.

RUNNING ANDROID APP:

After successfully [Setting up an Android project](https://www.geeksforgeeks.org/android-starting-with-first-app-android-project/), all of the default files are created with default code in them. Let us look at this default code and files and try to run the default app created.

The panel on the left side of the android studio window has all the files that the app includes. Under the java folder, observe the first folder containing the java file of your project.

For every activity, a “.java” file and a “.xml” file is created. In this case for MainActivity, “MainActivity.java” and “activity\_main.xml” are created.

The above java file shows us the default code that is present when an app is created. An activity is created that extends AppCompactActivity class.

The “res” folder contains “layout” subfolder, which includes the xml files of the projects.

CHAPTER-3

INTRODUCTION TO XML

XML stands for Extensible Markup Language. XML is a markup language much like HTML used to describe data. It is derived from Standard Generalized Markup Language(SGML). Basically, the XML tags are not predefined in XML. We need to implement and define the tags in XML. XML tags define the data and used to store and organize data. It’s easily scalable and simple to develop. In Android, the XML is used to implement UI-related data, and it’s a lightweight markup language that doesn’t make layout heavy. XML only contains tags, while implementing they need to be just invoked.

Basically in Android XML is used to implement the UI-related data. So understanding the core part of the UI interface with respect to XML is important. The User Interface for an Android App is built as the hierarchy of main **layouts, widgets**. The layouts are **ViewGroup**objects or containers that control how the child view should be positioned on the screen. **Widgets** here are view objects, such as Buttons and text boxes.

XML stands for Extensible Markup Language. XML is a markup language much like HTML used to describe data.  XML tags are not predefined in XML. We must define our own Tags. Xml as itself is well readable both by human and machine. Also, it is scalable and simple to develop. In Android we use xml for designing our layouts because xml is lightweight language so it doesn’t make our layout heavy.

CHAPTER-4

INTRODUCTION TO JAVA

Android App are mostly developed in JAVA language using Android SDK (Software Development Kit). Other languages like C, C++, Scala etc. can also be used for developing Android App, but JAVA is most preferred and mostly used programming language for Android App Development. So if you are a beginner in Android then JAVA language and complete knowledge of OOPS concepts is the first thing you need to learn before beginning Android Development.

JAVA is a programming language which is used in Android App Development. It is class based and object oriented programming whose syntax is influenced by C++. The primary goals of JAVA is to be simple, object-oriented, robust, secure and high level.JAVA application runs on JVM (JAVA Virtual Machine) but Android has it’s own virtual machine called **Dalvik Virtual Machine (DVM)** optimized for mobile devices.

JAVA is the technology of choice for building applications using managed code that can execute on mobile devices. Android is an open source software platform and Linux-based operating system for mobile devices. The Android platform allows developers to write managed code using Java to manage and control the Android device. Android applications can be developed by using the Java programming language and the Android SDK. So, familiarity with the basics of the Java programming language is a prerequisite for programming on the Android platform. This article discusses where Java fits in mobile application development and how we can use Java and Android SDK to write applications that can work on Android devices.

Java even made inroads into embedded processors technology as well; the Java Mobile Edition was built for creating applications that can run on mobile devices. All these, added to Java’s meteoric rise, were the prime factors that attributed to the decision of adopting Java as the primary development language for building applications that run on Android. Java programs are secure because they run within a sandbox environment. Programs written in Java are compiled into intermediate code known as bytecode. This bytecode is then executed inside the context of the Java Virtual Machine. You can [learn more about Java](https://www.developer.com/java/) from this android.

CHAPTER-5

REQUIREMENT SPECIFICATION

5.1 SOFTWARE REQUIREMENTS:

1. Any Operating System with JDK installed.
2. To develop this project the various Software resources used are.
3. Front End: JVA ,XML.
4. Tool: Android SDK with ADT Plugin.
5. Technology: Java validation.
6. IDE: Android studio.

5.2 HARDWARE REQUIREMENTS:

1. Quad Core Processor and Above.
2. RAM 4GB and Above.
3. 20 GB Free Hard Disk Space.
4. Android device.

CHAPTER-6

OBJECTIVES AND FEATURES

6.1 OBJECTIVES OF THE PROJECT:

1. Convenience: The objective of providing convenience through the Simple Dictionary Android App is to make word lookup a hassle-free experience for users. By having a dictionary app readily available on their Android devices, users can access word definitions at any time and from anywhere without the need to carry physical dictionaries or rely on an internet connection. The app aims to simplify the process of finding word meanings, ensuring that users have a convenient tool at their fingertips.
2. User-Friendly Interface: The app's objective of offering a user-friendly interface is to create an intuitive and easy-to-use experience for users. The interface is designed to be visually appealing, with clear and organized elements that guide users through the app's functionality. The objective is to ensure that users, regardless of their technological expertise, can navigate the app effortlessly and find the information they need without confusion or frustration.
3. Quick Word Search: The objective of enabling quick word search functionality is to save users' time and provide them with instant access to word definitions. The app aims to optimize the search process, allowing users to enter a word or phrase and retrieve its definition promptly. By minimizing search time and delivering rapid results, the app enhances user satisfaction and encourages frequent use.
4. Vocabulary Enhancement: The primary objective of the Simple Dictionary Android App is to facilitate vocabulary enhancement. By providing accurate and concise definitions for searched words, the app aims to expand users' knowledge and understanding of the language. The objective is to offer users a reliable resource that helps them discover new words, comprehend their meanings, and incorporate them into their vocabulary. By actively engaging users in vocabulary development, the app supports their language growth and communication skills.
5. Language Learning Support: The objective of providing language learning support through the app is to assist users in their language acquisition journey. By including audio pronunciations for words, the app helps users improve their pronunciation skills, develop an ear for correct language usage, and enhance their overall language proficiency. The app aims to provide a comprehensive learning experience by combining written definitions with auditory reinforcement, fostering a deeper understanding and connection to the language.
6. Accessibility: The objective of ensuring accessibility is to make the Simple Dictionary Android App usable by a diverse range of users. The app aims to accommodate users with different needs, including those with visual impairments or users who prefer alternative input methods. Accessibility features, such as support for screen readers or customizable font sizes, are incorporated to ensure that all users can effectively utilize the app's features and benefit from its functionality.
7. Offline Functionality: The objective of including offline functionality is to address the limitations of internet connectivity. By storing a local database of word definitions on the user's device, the app enables users to access word meanings even in offline mode or when internet access is limited. This objective ensures that users can rely on the app's resources regardless of their internet connection status, promoting uninterrupted learning and reference.
8. By focusing on convenience, user-friendliness, quick search, vocabulary enhancement, language learning support, accessibility, and offline functionality, the Simple Dictionary Android App strives to create a comprehensive and valuable tool for users seeking word definitions. The app's objectives align with the goal of providing a reliable and user-centric dictionary app that enhances language skills and supports users' language learning and communication efforts.

6.2 FEATURES OF THE PROJECT:

The Simple Dictionary Android App incorporates the following features:

1. Word Search: Users can enter a word or phrase into the search bar and retrieve its definition from an extensive dictionary database. The app supports both single-word searches and multi-word phrase searches.
2. Definition Display: The app displays the definitions of searched words in a clean and readable format. It provides accurate and concise explanations of word meanings, ensuring users can easily comprehend and learn new words.
3. Offline Mode: The app offers offline functionality, allowing users to search for word definitions even when they don't have an internet connection. The dictionary database is stored locally on the device, ensuring uninterrupted access to word meanings.

CHAPTER-7

DESIGN AND IMPLEMENTATION

XML CODE:

activity\_main.xml

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:paddingLeft="16dp"

android:paddingTop="16dp"

android:paddingRight="16dp"

android:paddingBottom="16dp"

tools:context=".MainActivity">

<EditText

android:id="@+id/editText1"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:hint="Enter a word"

android:imeOptions="actionDone"

android:inputType="text"

android:minHeight="48dp" />

<Button

android:id="@+id/buttonSearch1"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_below="@id/editText1"

android:layout\_centerHorizontal="true"

android:layout\_marginTop="16dp"

android:text="Search" />

<TextView

android:id="@+id/textViewResult"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_below="@id/buttonSearch1"

android:layout\_marginTop="16dp"

android:textSize="18sp" />

</RelativeLayout>

JAVA CODE:

MainActivity.java

package com.example.dictionary;

import android.os.Bundle;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStream;

import java.io.InputStreamReader;

import java.util.HashMap;

public class MainActivity extends AppCompatActivity {

private HashMap<String, String> dictionary;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

final EditText editText = findViewById(R.id.editText1);

Button buttonSearch = findViewById(R.id.buttonSearch1);

final TextView textViewResult = findViewById(R.id.textViewResult);

buttonSearch.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

String word = editText.getText().toString().trim().toLowerCase();

if (!word.matches("[a-zA-Z]+")) {

textViewResult.setText("please enter a word");

} else {

String definition = dictionary.get(word);

if (definition != null) {

textViewResult.setText(definition);

} else {

textViewResult.setText("The word is not present in the dictionary. We will try to include it in the dictionary.");

// Code to include the word in the dictionary goes here

}

}

}

});

initializeDictionary();

}

private void initializeDictionary() {

dictionary = new HashMap<>();

try {

InputStream inputStream = getAssets().open("dictionary.txt");

BufferedReader reader = new BufferedReader(new InputStreamReader(inputStream));

String line;

while ((line = reader.readLine()) != null) {

String[] parts = line.split(","); // Modify the delimiter if necessary

if (parts.length == 2) {

String word = parts[0].trim().toLowerCase();

String meaning = parts[1].trim();

dictionary.put(word, meaning);

}

}

reader.close();

inputStream.close();

} catch (IOException e) {

e.printStackTrace();

}

}

}

CHAPTER-8

RESULTS

SNAPSHOTS:



Fig. 8.1: Main Page Display.



Fig. 8.2: When a Word is Entered, it Retrieves the Meaning from the Dictionary Successfully.

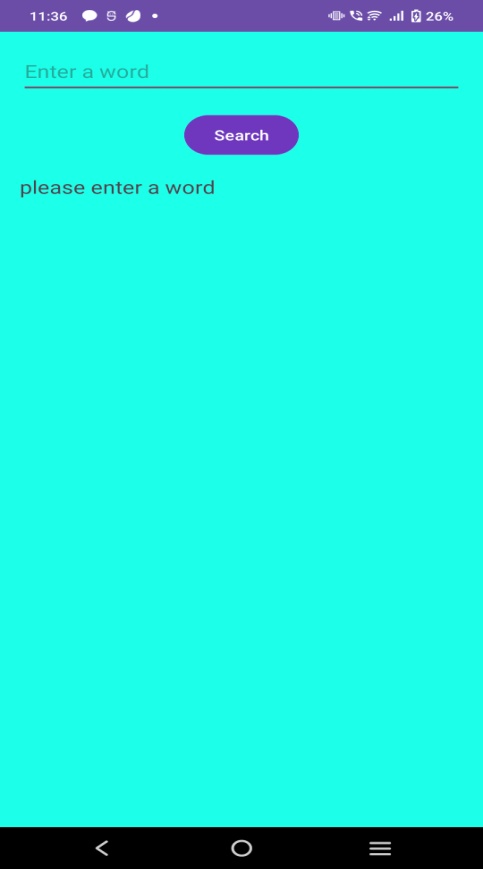


Fig. 8.3: When Space is Given as Input.



Fig. 8.4: When Numbers are Given as Input.

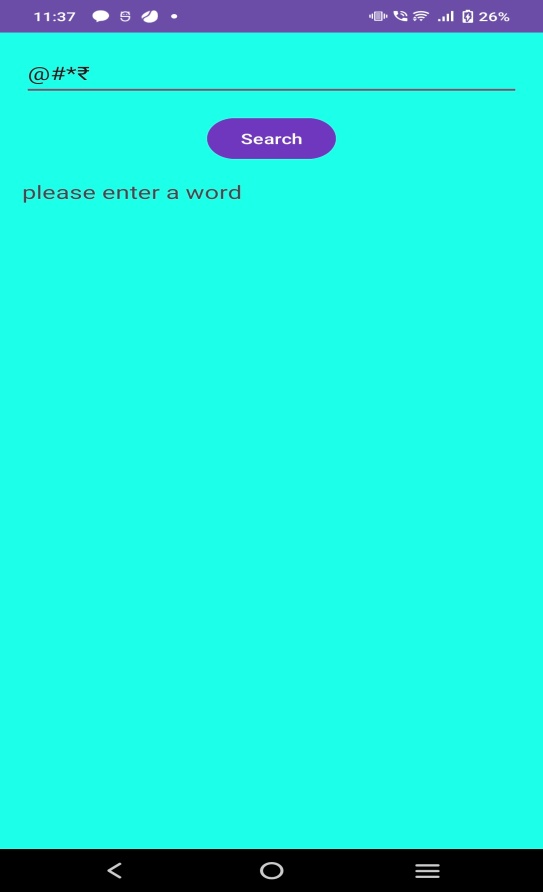


Fig. 8.5: When Symbols are Given as Input.

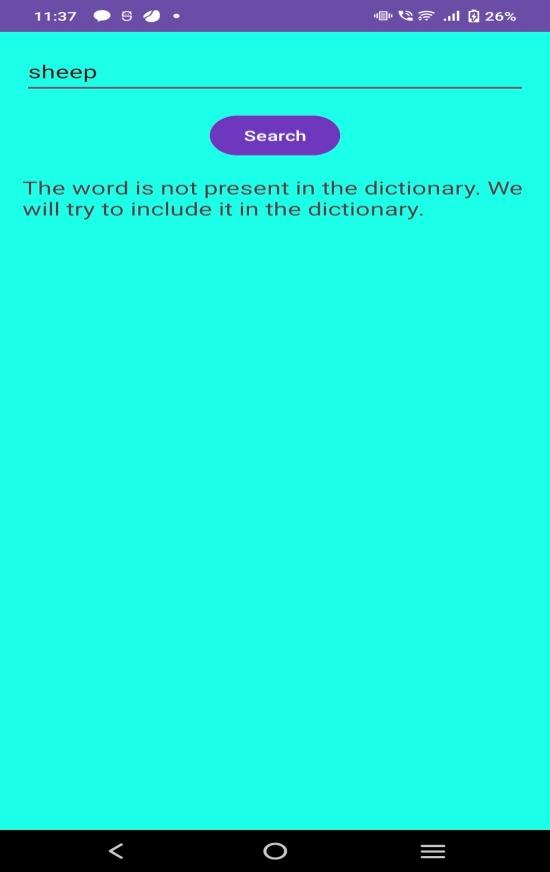


Fig. 8.6: When a Word is Given as Input & if it’s Not Present in the Dictionary,

CONCLUSION

The Simple Dictionary Android App project has successfully achieved its objectives of providing users with a convenient and feature-rich platform for word exploration. Throughout the development process, the app has been designed to offer an intuitive user interface, quick and accurate word searches, comprehensive definitions, daily word updates, bookmarking functionality, and interactive quizzes to enhance users' language skills and vocabulary.

The app demonstrates the effective utilization of Android Studio and Java programming to create a functional and user-friendly dictionary application. By leveraging the Android SDK and its associated tools, the project team has been able to develop an app that meets the needs of users who seek a reliable and accessible tool for word lookup on their Android devices.

To further improve the app in the future, additional features could be considered, such as word pronunciation, example sentences, offline mode, and integration with language learning platforms. These enhancements would provide users with even more comprehensive language resources and a seamless learning experience.

Regular updates and maintenance are essential to ensure the app remains compatible with the latest Android versions and incorporates user feedback for further improvements. By actively addressing user needs and incorporating new technologies and features, the Simple Dictionary Android App can continue to evolve and provide an increasingly valuable tool for language learners and individuals seeking to enhance their vocabulary.

In conclusion, the Simple Dictionary Android App mini project exemplifies the successful implementation of Android Studio and Java programming to develop a functional, user-friendly, and continually improving dictionary application.

FUTURE ENHANCEMENT

Enhancements for a simple dictionary Android app can include a modern and visually appealing UI, predictive search, auto-complete, voice input, word suggestions, random word feature, word pronunciation with different accents, word history and favorites, offline mode with local data storage, word examples and synonyms, word of the day with notifications, cross-app integration, user feedback system, and ratings. Prioritize these enhancements based on project scope and user needs, while ensuring compatibility with different screen sizes. Collect feedback during development to improve the app and provide a comprehensive language learning experience.

REFERENCES

BOOKS:

[1] "Android Programming: The Big Nerd Ranch Guide" by Bill Phillips and Brian Hardy.

[2] "Android Studio Development Essentials" by Neil Smyth.

[3] "Head First Android Development" by Dawn Griffiths and David Griffiths.

RESEARCH PAPERS AND ARTICLES:

[1] "Android Application Development and its Security Issues" by Abhishek Sharma and Anand Mohan.

[2] "Design and Development of Mobile Application for Android Platform" by S. R. Thorat and V. V. Wani.

[3] "Performance Analysis of Android Applications" by Haiping Xu, Xusheng Xiao, and Xuehai Zhou.