VISVESVARAYA TECHNOLOGICAL UNIVERSITY

JnanaSangama, Belgaum-590014

A Mobile Application Development Project Report On

"FUSE – Minigames"

Submitted in Partial fulfilment of the Requirements for the VI Semester of the Degree of

Bachelor of Engineering

In

Computer Science & Engineering

By

ANANYA BHOMBORE

(1CE19CS008)

FOZAIL AHMED

(1CE19CS031)

Under the Guidance of

Mrs. SHASHIKALA H C

Asst. Prof. Dept. of CSE

CITY ENGINEERING COLLEGE

Doddakallasandra, Kanakapura Road, Bengaluru-560061

CITY ENGINEERING COLLEGE

Doddakallasandra, Kanakapura Road, Bengaluru-560061

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certified that the **MOBILE APPLICATION DEVELOPMENT** Mini Project work entitled **“FUSE – Minigames”** has been carried out by **ANANYA BHOMBORE(1CE19CS008)** and **FOZAIL AHMED(1CE19CS031)**, bonafide students of **City Engineering College**, in partial fulfilment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the **Visvesvaraya Technological University**, **Belgaum**, during the year **2021-2022**. It is certified that all corrections/suggestions indicated for internal assessment have been incorporated in the report deposited in the departmental library. The MOBILE APPLICATION DEVELOPMENT Mini Project Report has been approved as it satisfies the academic requirements in the respect of project work prescribed for the said Degree.

Mrs. Shashikala H C	**Dr. Sowmya Naik**	**Dr. Thippeswamy .H.N**
Asst. Prof, Dept. of CSE	HOD, Dept. of CSE	Principal of CEC

External Viva

Name of the examiners	Signature with date
1.	
2.	

ABSTRACT

The objective of this Mini Project is to illustrate concepts and usage of inbuilt functions in XML and Java for the development of an Android Application. The FUSE – Minigames is a modular minigames app that contains a number of activities. The application is started with splash activity which has an activity of its own. Key point for splash activity is to ensure that in the manifest, the launching activity is set to Splash activity as our main launcher. Some of the activities are Tic Tap Toe, Tap man, Tap That Song, Simon Tap, Reaction Tap and Leader board. Keyboard and mouse can be used to interact with the project.

Using inbuilt functions of Java, the application can be navigated. The functions of the Buttons, Spinner, Images are set using Java. XML functions are used to design the Animation, Media player and other applications. Edit text and Font style is selected using XML functions. Database implementation is done using SQLite.

ACKNOWLEDGEMENT

While presenting this MOBILE APPLICATION DEVELOPMENT project on “FUSE – Mini games”, we feel that it is our duty to acknowledge the help rendered to us by various persons. Firstly we thank God for showering his blessings on us. We are grateful to our institution City Engineering College for providing us a congenial atmosphere to carry out the project successfully.

We would like to express our heartfelt gratitude to **Dr.Thippeswamy H.N**, CEC, Bangalore, for extending his support.

We would also like to express our heartfelt gratitude to **Dr. Sowmya Naik**, HOD, Computer Science and Engineering whose guidance and support was truly invaluable.

We are very grateful to our guide **Mrs. Shashikala H C**, Asst. Prof., Department of Computer Science, for her able guidance and valuable advice at every stage of our project which helped me in the successful completion of our project.

We would also have indebted to our Parent and Friends for their continued moral and material support throughout the course of project and helping me in finalize the presentation.

Our heartful thanks to all those have contributed bits, bytes and words to accomplish this project.

ANANYA BHOMBORE(1CE19CS008)

FOZAIL AHMED(1CE19CS031)

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1

INTRODUCTION

1.1 Android

Android is a mobile operating system based on a modified version of the Linux kernel and other open-source software, designed primarily for touchscreen mobile devices such as smartphones and tablets. Android is developed by a consortium of developers known as the Open Handset Alliance and commercially sponsored by Google.

1.1.1 **Features of Android**

Messaging

SMS and MMS are available forms of messaging, including threaded text messaging and Android Cloud To Device Messaging (C2DM) and now enhanced version of C2DM, Android Google Cloud Messaging (GCM) is also a part of Android Push Messaging services. Android phones also have the ability to send and receive RCS via the messages app (if supported by the carrier).

Auto Correction and Dictionary

Android Operating System has an interesting feature called Auto Correction. When any word is misspelled, then Android recommends the meaningful and correct words matching the words that are available in Dictionary. Users can add, edit and remove words from Dictionary as per their wish.

Web browser

The web browser available in Android is based on the open-source Blink (previously WebKit) layout engine, coupled with Chromium's V8 JavaScript engine. Then the WebKitusing Android Browser scored 100/100 on the Acid3 test on Android 4.0 ICS; the Blinkbased browser currently has better standards support. The old web browser is variably known as 'Android Browser', 'AOSP browser', 'stock browser', 'native browser', and 'default browser'
(from the time it was always the default). Starting with Android 4.4 KitKat, Google has begun licensing Google Chrome (a proprietary software) separately from Android, but usually bundled with (what most device vendors did). Since Android 5.0 Lollipop, the

WebView browser that apps can use to display web content without leaving the app has been separated from the rest of the Android firmware in order to facilitate separate security updates by Google.

Voice-based features

Google search through voice has been available since initial release. Voice actions for calling, texting, navigation, etc. are supported on Android 2.2 onwards. As of Android 4.1, Google has expanded Voice Actions with ability to talk back and read answers from Google's Knowledge Graph when queried with specific commands. The ability to control hardware has not yet been implemented.

Multi-touch

Android has native support for multi-touch which was initially made available in handsets such as the HTC Hero. The feature was originally disabled at the kernel level (possibly to avoid infringing Apple's patents on touch-screen technology at the time). Google has since released an update for the Nexus One and the Motorola Droid which enables multi-touch natively.

Multitasking

Multitasking of applications, with unique handling of memory allocation, is available.

Screen capture

Android supports capturing a screenshot by pressing the power and home-screen buttons at the same time. Prior to Android 4.0, the only methods of capturing a screenshot were through manufacturer and third-party customizations (apps), or otherwise by using a PC connection (DDMS developer's tool). These alternative methods are still available with the latest Android.

TV recording

Android TV supports capturing and replaying it.

Video calling

Android does not support native video calling, but some handsets have a customized version of the operating system that supports it, either via the UMTS network (like the Samsung

Galaxy S) or over IP. Video calling through Google Talk is available in Android 2.3.4 (Gingerbread) and later. Gingerbread allows Nexus S to place Internet calls with a SIP account. This allows for enhanced VoIP dialing to other SIP accounts and even phone numbers. Skype 2.1 offers video calling in Android 2.3, including front camera support. Users with the Google+ Android app can perform video chat with other Google+ users through Hangouts.

Multiple language support

Android supports multiple languages.

Accessibility

Built-in text-to-speech is provided by TalkBack for people with low or no vision. Enhancements for people with hearing difficulties are available, as are other aids.

1.1.2 Android Software Development

Android software development is the process by which applications are created for devices running the Android operating system. Google states that Android apps can be written using Kotlin, Java, and C++ languages using the Android software development kit (SDK), while using other languages is also possible. All non-JVM languages, such as Go, JavaScript, C, C++ or assembly, need the help of JVM language code, that may be supplied by tools, likely with restricted API support.

The Android software development kit (SDK) includes a comprehensive set of development tools. The Android SDK Platform Tools are a separately downloadable subset of the full SDK, consisting of command-line tools such as ADB and fastboot. The Android Debug Bridge (ADB) is a tool to run commands on a connected Android device. Fastboot is a protocol used for flashing filesystems. Code written in C/C++ can be compiled to ARM, or x86 native code (or their 64-bit variants) using the Android Native Development Kit (NDK).

1.2 Java in Android Studio

The Java language is a key pillar in Android, an open-source mobile operating system. Although Android, built on the Linux kernel, is written largely in C, the Android SDK uses the Java language as the basis for Android applications but does not use any of its standard GUI, SE, ME or other established Java standards. The bytecode language supported by the

Android SDK is incompatible with Java bytecode and runs on its own virtual machine, optimized for low-memory devices such as smartphones and tablet computers. Depending on the Android version, the bytecode is either interpreted by the Dalvik virtual machine or compiled into native code by the Android Runtime.

1.3 XML in Android

Extensible Markup Language (XML) is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. Simple API for XML (SAX) is a lexical, event-driven API in which a document is read serially and its contents are reported as callbacks to various methods on a handler object of the user's design. SAX is fast and efficient to implement, but difficult to use for extracting information at random from the XML, since it tends to burden the application author with keeping track of what part of the document is being processed. It is better suited to situations in which certain types of information are always handled the same way, no matter where they occur in the document.

We can create XML layouts in Android, and handle them using Java. Android XML layouts are also part of a larger umbrella of Android files and components called resources. Resources are the additional files and static content an application needs, such as animations, color schemes, layouts, menu layouts.

1.3.1 Anatomy of Android XML Layouts

Each layout file must contain one (and only one) root element. Linear Layouts, Relative Layouts, and Frame Layouts (see Root Views section below) may all be root elements. Other layouts may not be. All other XML elements will reside within this root object.

A View is simply an object from Android's built-in View class. It represents a rectangular area of the screen, and is responsible for displaying information or content, and event handling. Text, images, and buttons are all Views in Android.

A ViewGroup is a subclass of View, and is essentially an 'invisible container' that holds multiple Views or ViewGroups together, and defines their layout properties.

1.4 Project Goal

The aim of this project is to develop an Android application which supports gaming application comprising of numerous minigames. We aim to deliver marvellous and thrilling gaming experience through this minigames application. The application should be user friendly.

1.5 Scope

It is developed using Android Studio. It has been implemented on Windows. The application developed here provides an interface to interact with different activities at a time. The keyboard is the main input devices. The application splash screen and many other main activities which the user can interact with.

FUSE – Minigames:

FUSE, a 'fuse'-ion of minigames , is a minigames app which consists of six minigames .The six mini games are TapTorial , Tic Tap Toe , Tapman , Tap That Song , Simon Tap , Reaction Tap . We have developed an interactive user interface which users can play during boredom hours.

Once the Splash Screen loads up with Fuse Logo , we enter the main page of the application. We have enter to enter a username /gamertag (or leave it as blank and you will registered as anonymous player) , then we can click on the play icon.

On clicking the play icon, we enter the game catalogue , where the gamer can choose which game he wishes to play . Highscores of the users in the respectively games will be maintained along with the username/gamertag .We can go to the home screen of the application , on clicking highscore logo , it will take you to the Scoreboard of various gamers and their highscores in different games .

Chapter 2

Literature Survey

Recent years have seen the rise of mobile users with PDAs, laptops and particularly mobile phones. Worldwide there are over 2 billion mobile phone users [1] – roughly one third of the world’s population. In South Africa alone there are 30 million mobile phone users [1] out of a population of 47 million. This pervasiveness of mobile devices has created a substantial market for mobile applications and games that continues to grow extremely rapidly.

With so many mobile devices, it is crucial for a developer to target the right mobile platform and development environment, or a combination thereof. Portability across platforms will ensure the maximum target audience is reached. This is specifically important for mobile game development.

The mobile game industry has firmly asserted itself as a significant industry with major development houses such as EA Mobile driving it forward. As such, it is possible to make some observations about the mobile game industry. Primarily, we are developing a android mobile game application. In addition to the nature of the game, user input and hardware limitations also affect the success of a mobile game.

2.1 Java Programming on Android Studio

Java offers a collection of packages filled with in-built function, and widgets to develop an Android application. The Java programming language requires the presence of a software platform in order for compiled programs to be executed.

Oracle supplies the Java platform for use with Java. The Android SDK is an alternative software platform, used primarily for developing Android applications with its own GUI system.

2.2 XML Programming on Android Studio

Using Android's XML vocabulary, you can quickly design UI layouts and the screen elements they contain. Each layout file must contain exactly one root element, which must be a View or ViewGroup object. Once you've defined the root element, you can add additional layout objects or widgets as child elements to gradually build a View hierarchy that defines your layout. Android provides a straightforward XML vocabulary that corresponds to the View classes and subclasses, such as those for widgets and layouts. You can also use Android Studio's Layout Editor to build your XML layout using a drag-and-drop interface.

2.3 Basic Working of Fuse: Minigames

We have developed an interactive user interface of the minigames application which users can play during boredom hours. Once the Splash Screen loads up with Fuse Logo , we enter the main page of the application. We have enter to enter a username /gamertag (or leave it as blank and you will registered as anonymous player) , then we can click on the play icon.

On clicking the play icon, we enter the game catalogue , where the gamer can choose which game he wishes to play . Highscores of the users in the respectively games will be maintained Along with the username/gamertag .We can go to the home screen of the application , on clicking highscore logo , it will take you to the Scoreboard of various gamers and their highscores in different games .

Chapter 3

Requirements

3.1 User Requirements

- Easy and simple to use
- Android Device with Lollipop and above version.

3.2 Hardware Requirements

- 64bit
- Windows 7 and above
- Intel i5 or above processor
- 8GB and above RAM

3.3 Software Requirements

This application has been designed on Windows Platform and uses Android Studio Chipmunk with JDK and SDK tools.

Development Platform

- Windows 10 Home

Development Tool

- Android Studio Chipmunk 2021.2.1

Language Used in Coding

- Java
- XML
- SQLite database

Chapter 4

SOFTWARE DESIGN

4.1 Proposed System

To achieve the required layout and design, the in-built function is used on XML. It is the section which provides the graphical interface. It is used to set the coloring, and backgrounds. Java functions are used to handle the application.

- Gradle-based build support.
- Android-specific refactoring and quick fixes.
- Lint tools to catch performance, usability, version compatibility and other problems.
- ProGuard integration and app-signing capabilities.
- Template-based wizards to create common Android designs and components.
- A rich layout editor that allows users to drag-and-drop UI components, option to preview layouts on multiple screen configurations.
- Support for building Android Wear apps.
- Built-in support for Google Cloud Platform, enabling integration with Firebase Cloud Messaging (Earlier 'Google Cloud Messaging') and Google App Engine.
- Android Virtual Device (Emulator) to run and debug apps in the Android studio.

4.2 Packages

import android.os.Bundle: A mapping from String keys to values of various types. In most cases, you should work directly with either the Bundle or Persistable Bundle subclass.

import android.widget.ImageButton: ImageButton is used to display a normal button with a custom image in button. In simple words we can say, ImageButton is a button with an image that can be pressed or clicked by the users. By default it looks like a normal button with the standard button background that changes the color during different button states.

import android.widget.EditText: A user interface element for entering and modifying text. When you define an edit text widget, you must specify the R.styleable.TextView_inputType attribute.

import android.util.Timer: Timer is a class in android which is used to perform some task according to time period. Like countdown clock. Or we can call any function after some time like after 2 minutes using timer class.

import android.util.log: Mock Log implementation for testing on non android host.

import android.view.View: This class represents the basic building block for user interface components. A View occupies a rectangular area on the screen and is responsible for drawing and event handling. View is the base class for widgets, which are used to create interactive UI components (buttons, text fields, etc.). The ViewGroup subclass is the base class for layouts, which are invisible containers that hold other Views (or other ViewGroups) and define their layout properties.

import android.widget.Toast: A toast is a view containing a quick little message for the user. The toast class helps you create and show those.

When the view is shown to the user, appears as a floating view over the application. It will never receive focus. The user will probably be in the middle of typing something else. The idea is to be as unobtrusive as possible, while still showing the user the information you want them to see. Two examples are the volume control, and the brief message saying that your settings have been saved. The easiest way to use this class is to call one of the static methods that constructs everything you need and returns a new Toast object.

4.3 Interaction

A user interface element the user can tap or click to perform an action. The above snippet creates an instance of `View.OnClickListener` and wires the listener to the button using `setOn`~~`ClickListener(View.OnClickListener)`. As a~~ result, the system executes the code you write in `onClick(View)` after the user presses the button.

4.4 Main Class

```
public class MainActivity extends AppCompatActivity
{
   ImageView img;
   ImageButton buttonToScore;
```

```
Animation rotateAnim;
EditText editText;
String username;
String currentUser;

@Override
protected void onCreate(Bundle savedInstanceState)
{
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    buttonToScore = findViewById(R.id.buttonScoreList);
    editText=findViewById(R.id.userName);
    img=findViewById(R.id.playButton);
    img.setClickable(true);
    rotateAnim = AnimationUtils.loadAnimation(this, R.anim.rotate);

    createDB();
    createTables();

    final Intent intent=new Intent(MainActivity.this,LevelGrid.class);

    buttonToScore.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            startActivity(new Intent(MainActivity.this, ViewScores.class));
        }
    });

    img.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            username = editText.getText().toString();
            if (username.equals("")) {
                username = "Anonymous";
            }
            saveUser(username);
            img.startAnimation(rotateAnim);

        }
    });

    rotateAnim.setAnimationListener(new Animation.AnimationListener() {
        @Override
        public void onAnimationStart(Animation animation) {
        }

        @Override
        public void onAnimationEnd(Animation animation) {
            startActivity(intent);
            }

            @Override
            public void onAnimationRepeat(Animation animation) {
            }
        });

    }

    public void saveUser(String username)
    {
        SharedPreferences sharedPref =
PreferenceManager.getDefaultSharedPreferences(MainActivity.this);
        SharedPreferences.Editor editor = sharedPref.edit(); //editor needed to put content in

        editor.putString(Constants.USERNAME_CURRENT, username);
        editor.commit();

        Toast.makeText(this, username + " is now a gamer ", Toast.LENGTH_SHORT).show();

    }

    public void loadData()
    {
        SharedPreferences sharedPref = getSharedPreferences(Constants.SHARED_PREFS,
MODE_PRIVATE);
        currentUser = sharedPref.getString(Constants.USERNAME_CURRENT, "Anonymous"); //if user
not found, make username "Anonymous"
    }

    private void createDB()
    {
        try {
            DBHelper.tappyDB = openOrCreateDatabase("tappy.db", MODE_PRIVATE, null);

            Log.d("DB Tappy", "DB created");
        }
        catch (Exception ex)
        {
            Log.e("DB Tappy", ex.getMessage());
        }
    }


    private void createTables()
    {
        try
        {
            String setPRAGMAForeignKeysOn = "PRAGMA foreign_keys = ON;";
            String createScoresTableCmd = "CREATE TABLE scores "
```

```
                + "(username TEXT, game TEXT, score REAL, info TEXT);";


            DBHelper.tappyDB.execSQL(setPRAGMAForeignKeysOn);
            DBHelper.tappyDB.execSQL(createScoresTableCmd);

            Log.d("DB Tappy", "Tables created");
        }
        catch (Exception ex)
        {
            Log.e("DB Tappy", "Error creating table" + ex.getMessage());
        }

    }

    @Override
    public void onBackPressed(){
        Intent a = new Intent(Intent.ACTION_MAIN);
        a.addCategory(Intent.CATEGORY_HOME);
        a.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
        startActivity(a);
    }
}
```

Chapter 5
Implementation

5.1 Source Code

5.1.1 XML Code:

```
 <?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity"
    android:background="@color/splashColor">

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:fontFamily="sans-serif"
        android:text="Enter Your GamerTag !"
        android:textColor="@color/black"
        android:textSize="24sp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.113" />

    <ImageView
        android:id="@+id/splashImg"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.497"
        app:srcCompat="@drawable/fuse1" />

    <EditText
        android:id="@+id/userName"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="36dp"
        android:ems="10"
        android:fontFamily="casual"
        android:hint="  Eg GamerTag : deathstroke-x10"
        android:inputType="textPersonName"
        android:textColor="@color/darkOrange"
        android:textColorHint="@color/buttonHangMan"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.496"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/textView" />

    <ImageButton
        android:id="@+id/buttonScoreList"
        android:layout_width="189dp"
        android:layout_height="126dp"
        android:background="@android:color/transparent"
        android:scaleType="centerInside"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="1.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.861"
        app:srcCompat="@drawable/topthree" />

    <ImageView
        android:id="@+id/playButton"
        android:layout_width="247dp"
        android:layout_height="144dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/textView"
        app:layout_constraintVertical_bias="0.854"
        app:srcCompat="@drawable/videoplayer" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

5.1.2 Java Code:

```
package com.example.sampleproject;

import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.preference.PreferenceManager;
import android.util.Log;
import android.view.View;
import android.view.animation.Animation;
import android.view.animation.AnimationUtils;
import android.widget.EditText;
import android.widget.ImageButton;
import android.widget.ImageView;
import android.widget.Toast;

import com.example.sampleproject.SupportClasses.Constants;
import com.example.sampleproject.SupportClasses.DBHelper;

public class MainActivity extends AppCompatActivity
{
   ImageView img;
   ImageButton buttonToScore;
   Animation rotateAnim;
   EditText editText;
   String username;
   String currentUser;

   @Override
   protected void onCreate(Bundle savedInstanceState)
   {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);

      buttonToScore = findViewById(R.id.buttonScoreList);
      editText=findViewById(R.id.userName);
      img=findViewById(R.id.playButton);
      img.setClickable(true);
      rotateAnim = AnimationUtils.loadAnimation(this, R.anim.rotate);


      createDB();
      createTables();

      final Intent intent=new Intent(MainActivity.this,LevelGrid.class);

      buttonToScore.setOnClickListener(new View.OnClickListener() {
         @Override
```

```
            public void onClick(View view) {
                startActivity(new Intent(MainActivity.this, ViewScores.class));
            }
        });

        img.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                username = editText.getText().toString();
                if (username.equals("")) {
                    username = "Anonymous";
                }
                saveUser(username);
                img.startAnimation(rotateAnim);

            }
        });

        rotateAnim.setAnimationListener(new Animation.AnimationListener() {
            @Override
            public void onAnimationStart(Animation animation) {
            }

            @Override
            public void onAnimationEnd(Animation animation) {
                startActivity(intent);
            }

            @Override
            public void onAnimationRepeat(Animation animation) {
            }
        });

    }
    public void saveUser(String username)
    {
        SharedPreferences sharedPref =
PreferenceManager.getDefaultSharedPreferences(MainActivity.this);
        SharedPreferences.Editor editor = sharedPref.edit(); //editor needed to put content in

        editor.putString(Constants.USERNAME_CURRENT, username);
        editor.commit();

        Toast.makeText(this, username + " is now a gamer ", Toast.LENGTH_SHORT).show();

    }
    public void loadData()
    {
        SharedPreferences sharedPref = getSharedPreferences(Constants.SHARED_PREFS,
MODE_PRIVATE);
        currentUser = sharedPref.getString(Constants.USERNAME_CURRENT,
"Anonymous"); //if user not found, make username "Anonymous"
    }

    /*create main DB*/
    private void createDB()
    {
        try {
            DBHelper.tappyDB = openOrCreateDatabase("tappy.db", MODE_PRIVATE, null);

            Log.d("DB Tappy", "DB created");
        }
        catch (Exception ex)
        {
            Log.e("DB Tappy", ex.getMessage());
        }
    }


    private void createTables()
    {
        try
        {
            String setPRAGMAForeignKeysOn = "PRAGMA foreign_keys = ON;";

            String createScoresTableCmd = "CREATE TABLE scores "
                    + "(username TEXT, game TEXT, score REAL, info TEXT);";


            DBHelper.tappyDB.execSQL(setPRAGMAForeignKeysOn);
            DBHelper.tappyDB.execSQL(createScoresTableCmd);

            Log.d("DB Tappy", "Tables created");
        }
        catch (Exception ex)
        {
            Log.e("DB Tappy", "Error creating table" + ex.getMessage());
        }

    }
    @Override
    public void onBackPressed(){
        Intent a = new Intent(Intent.ACTION_MAIN);
        a.addCategory(Intent.CATEGORY_HOME);
        a.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
        startActivity(a);
    }

}
```

Chapter 6
Snapshots

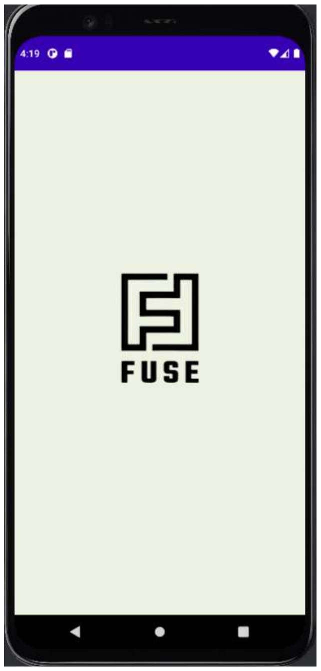


Fig 6.1: Splash Activity Screen

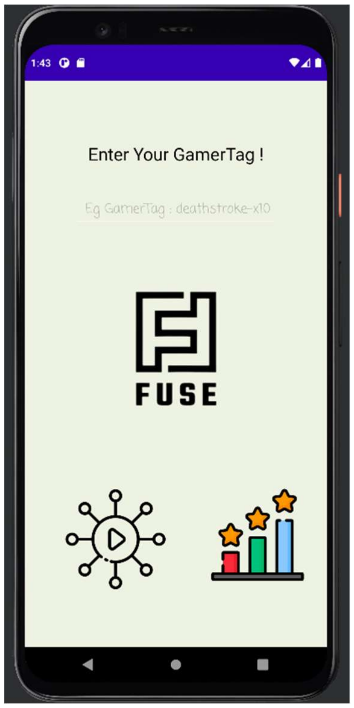


Fig 6.2: Main Activity Screen

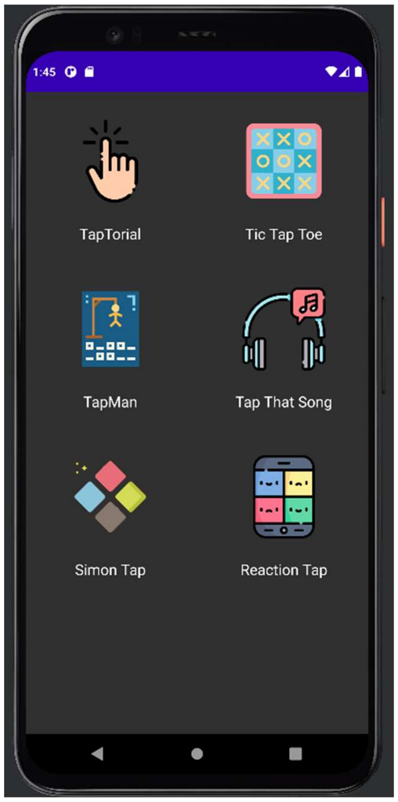


Fig 6.3: Level Selector Screen

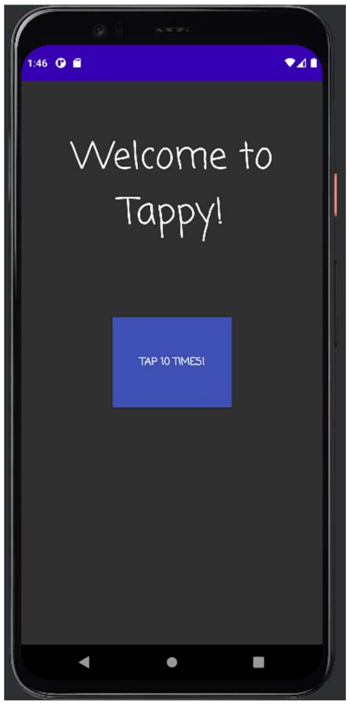


Fig 6.4: TapTorial Minigame

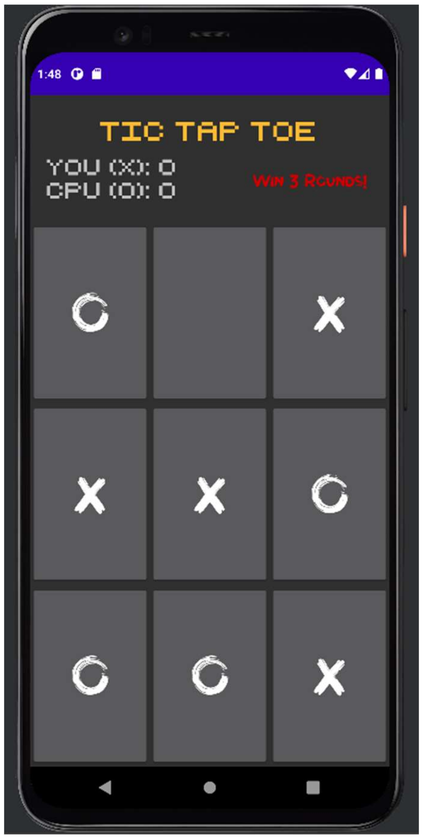


Fig 6.5: Tic Tap Toe Minigame

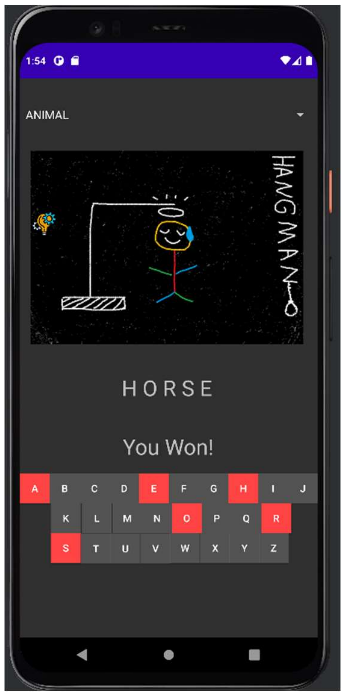


Fig 6.6: TapMan Minigame

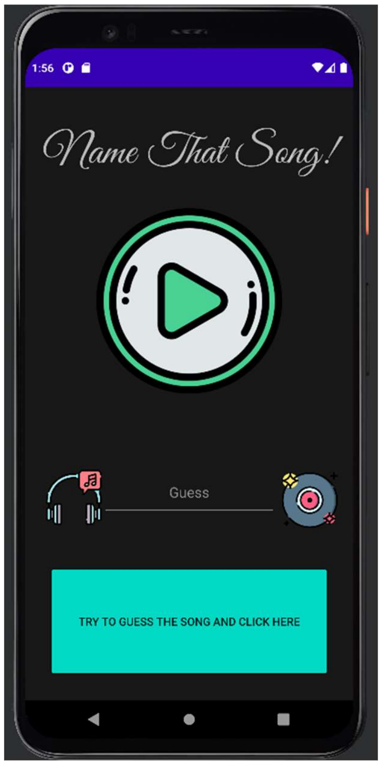


Fig 6.7: Name That Song Minigame

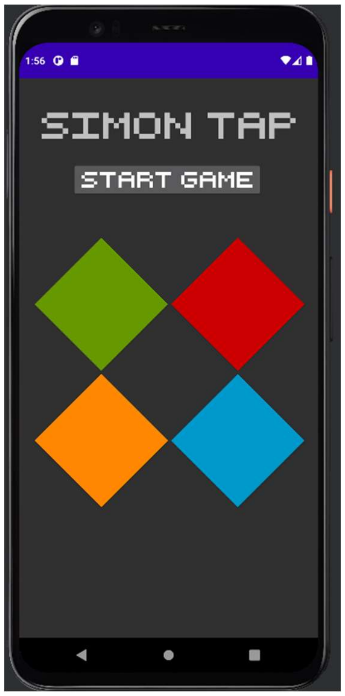


Fig 6.8: Simon Tap Minigame

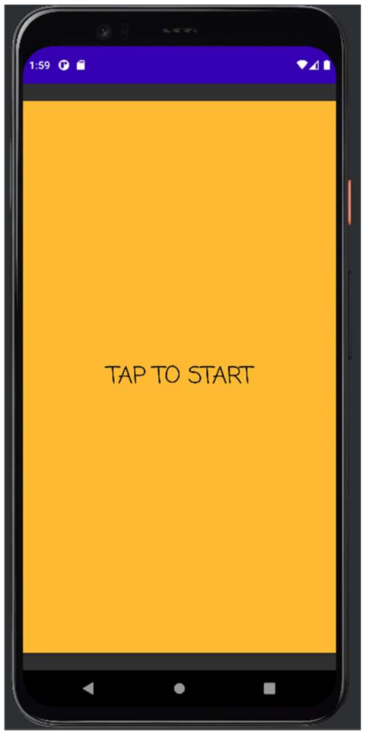
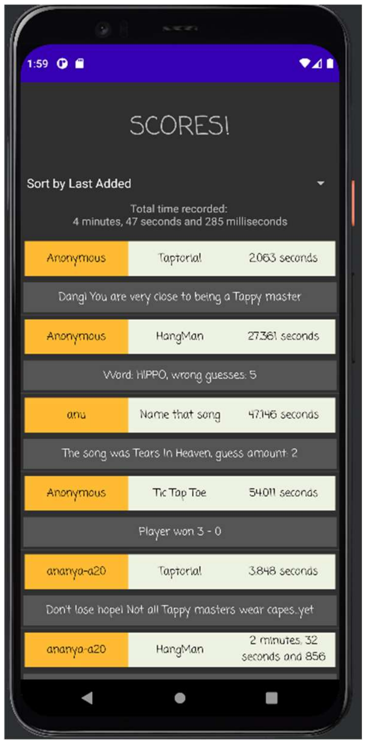


Fig 6.9: Reaction Tap Minigame



1:59

SCORES!

Sort by Last Added

Total time recorded:
4 minutes, 47 seconds and 285 milliseconds

Anonymous	Taptorial	2.063 seconds
Dang! You are very close to being a Tappy master		
Anonymous	HangMan	27.361 seconds
Word: HIPPO, wrong guesses: 5		
anu	Name that song	47.146 seconds
The song was Tears In Heaven, guess amount: 2		
Anonymous	Tic Tap Toe	54.011 seconds
Player won 3 - 0		
ananya-a20	Taptorial	3.848 seconds
Don't lose hope! Not all Tappy masters wear capes..yet		
ananya-a20	HangMan	2 minutes, 32 seconds and 856

Fig 6.10: HighScore Board Screen

Chapter 7

CONCLUSION AND FUTURE ENHANCEMENTS

We have put in a lot of time and thought in this project to make it efficient, user-friendly and elegant and have succeeded in building a better app. All the required functions run smoothly and are hassle-free creating a pleasant experience for the user. This app has been tested multiple times and is now primed for use.

In the future the following enhancements can be done:

- The application design can be upgraded.
- Inclusion of more games to give more diverse gaming options .
- Providing more number of levels and stages, new dynamic background for better visualization.
- Implementing multi-player gaming system to enjoy the games with friends.
- Introducing a difficulty level for playing against the CPU for a more challenging experience.

REFERENCES

[1] Android Boot Camp for Developers Using Java®: A Guide to Creating Your First Android Apps, 3rd Edition, by Corinne Hoisington

[2] Android official documentation: https://developer.android.com/docs

[3] Tic Tac Toe tutorial: https://www.youtube.com/watch?v=apDL78MFR3o

[4] Font: https://www.dafont.com/brushot.font

[5] Font: https://www.dafont.com/retro-computer.font

[6] Timer and TimerTask tutorial: https://www.youtube.com/watch?v=7QVr5SgpVog

[7] Graph tutorials: https://www.youtube.com/watch?v=VriiDn676PQ

[8] Hatchful logo creator: https://hatchful.shopify.com/

[9] All icons and images used are from are free to use

DECLARATION

We students of 6th semester BE, Computer Science and Engineering College hereby declare that project work entitled "**FUSE – mini games**" has been carried out by us at City Engineering College, Bangalore and submitted in partial fulfilment of the course requirement for the award of the degree of **Bachelor of Engineering in Computer Science and Engineering of Vishvesvaraya Technological University, Belgaum**, during the academic year 2021-2022.

We also declare that, to the best of our knowledge and belief, the work reported here does not form the part of dissertation on the basis of which degree or award was conferred on a earlier occasion on this by any other student.

Date: 13/07/2022

Place: Bangalore

ANANYA BHOMBORE	**FOZAIL AHMED**
(1CE19CS008)	**(1CE19CS031)**