Visvesvaraya Technological University

Belagavi, Karnataka – 590018

Project report on

2D CAR GAME APPLICATION

Submitted in partial fulfillment of the requirements for the course

MOBILE APPLICATION DEVELOPMENT
(18CSMP68)

Submitted by

VARSHINI SANJEEVA JAKATI	**1JS20CS179**
VARSHITA B M	**1JS20CS180**

Under the guidance of

Mrs. Namitha S.J **B. E, M. Tech**

Assistant professor, Dept. of CSE,

JSS Academy of Technical Education, Bengaluru

JSS Academy of Technical Education, Bengaluru – 560060,

Department of Computer Science and Engineering

2022-2023

JSS Mahavidyapeetha, Mysuru

JSS Academy of Technical Education

JSS Campus, Uttarahalli – Kengeri Main Road, Bengaluru – 560060

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the project work entitled "**2D CAR GAME APPLICATION**" is a bona fide work carried out by **Ms. VARSHINI SANJEEVA JAKATI (1JS20CS179)** and **Ms. VARSHITA BM (1JS20CS180)** in partial fulfillment of the requirements for the course Mobile Application Development of 6th semester, Bachelor of engineering in Computer Science and engineering of the Visvesvaraya Technological University, Belagavi, during the academic year 2022 – 2023. It is certified that all corrections and suggestions indicated for internal assessment have been incorporated in the report. The project report has been approved as it satisfies the academic requirements in respect of the project work prescribed for the said degree.

Mrs. Namitha S.J. B.E, MTech	Mrs. Bhavani B.H.B.E, MTECH	Dr. Mallikarjuna P.B. B.E, MTech, Ph.D.
Assistant Professor,	Assistant Professor,	Associate Professor and Head,
Dept. of CSE,	Dept.of CSE,	Dept.of CSE,
JSSATE, Bengaluru	JSSATE,Bengaluru	JSSATE, Bengaluru

Name of the Examiners	Signature with date
i)……..	…….
ii)…….	…….

ABSTRACT

In today's world gaming application plays an important role as it offers entertainment, social interaction, technological advancements. They have become an important aspect of modern world, shaping how people interact with technology.

It is essential to create games that are not only fun and engaging but also optimized for the platforms they are intended to run on.

A 2-dimensional car game where the player drives a car on a three-lane road that never ends. Through the course the player encounters obstacles in the form of other cars that are moving in the opposite direction as the player. The main objective of this game is to drive the car as much as possible for a high score while avoiding being hit by other cars. Once the player is hit by another car, the game ends and the final score achieved by the player is displayed.
It also has different backgrounds where the game can be played thus making the game more interesting.IT also includes login and registration facility for the user. This game features simple graphics and user-friendly interface through which any new player can get an understanding of the game objective and mechanics in the first go.

This game will provide an engaging and an enjoyable gaming experience that challenges player's skills, provides excitement, encourages competition, and promotes cognitive development.

ACKNOWLEDGEMENTS

I express my humble greetings to his holiness **Jagadguru Sri Shivarathri Deshikendra Mahaswamijigalavaru** who has showered their blessings on us for framing our career successfully.

The completion of any project involves the efforts of many people. I've been lucky to have received a lot of help and support from Indian Railway aspirants and all other quarters, during the making of this project. I take this opportunity to acknowledge all those, whose guidance and encouragement helped me emerge successful.

I express our sincere thanks to our beloved principal, **Dr. Bhimasen Soragaon** for having supported us in our academic endeavors.
I am also indebted to **Dr. Mallikarjuna P.B.**, Associate Professor and head of department of Computer Science and engineering for the facilities and support extended towards us.

I am thankful to the resourceful guidance, timely assistance and graceful gesture of my guide **Mrs. Namitha S.J.**, Assistant professor, Department of Computer Science and engineering, who helped me in every aspect of my project work.
I am thankful to the resourceful guidance, timely assistance and graceful gesture of my guide **Mrs. Bhavani B.H.**, Assistant professor, Department of Computer Science and engineering, who helped me in every aspect of my project work.

Last but not the least, I am pleased to express our heart full thanks to all the teaching and non-teaching staff of department of Computer and Science engineering and my friends whohave rendered their help, motivation and support.

VARSHINI SANJEEVA JAKATI (1JS20CS179)

VARSHITA BM(1JS20CS180)

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Overview

In this project we have developed a mobile game using Android Studio. The programming language used here is Kotlin. The mobile game in question is a 2D car game which involves a player controlling the movement of a car. The 2D car game developed is simple and a fun game to play. A user of any age can find this game relaxing and entertaining as it incorporates many beautiful backgrounds that can be changed by the player.

The car can be controlled by mere mouse clicks. To move the car to the right, click anywhere on the screen on the right side of the car. Similarly, to move the car to the left side, click anywhere on the screen on the left side of the car. In this game, the car moves forward continuously till time infinite and only its right or left movements are controllable by the player. To make this game more interesting, there are many challenging features throughout the game. Challenges in a game sparks the competitive spirit of a user. The main challenge or obstacle in this game are cars that move in the opposite direction as that of the player's car. Once the player's car crashes with another car, the game ends and the player's score are displayed.

Building this application requires a solid understanding of Kotlin language, Android Studio, Android development and bunch of ideas. Various relevant libraries are incorporated in the program with suitable drawables and resources. How much distance can you cover while dodging all the obstacles?

1.2 Problem Statement

The aim of this mini project is to develop an Android application that provides entertainment and joy to the user while also igniting their competitive spirits by posing various challenges and difficulty levels to the user.

The 2D car game developed is simple and a fun game to play. A user of any age can find this game relaxing and entertaining as it incorporates many beautiful backgrounds that can be changed by the player. The main objective of the game is for the player to control their car such a way that they are able to dodge all the obstacles which come in the way of the player's car. The player here drives a red car and is met with many other cars of the color yellow which move in the direction opposite to the player's car. These yellow cars pose a threat to the player's car because the moment the two cars crash, the game ends. The speed of the car also increases automatically as the game runs its course which makes the game even more difficult as the player aims to cover more distance.

It is essential to create games that are not only fun and engaging but also optimized for the platforms they are intended to run on.

1.3 Motivation

The motivation behind this mini project is to get a hang of Android Studio and its various functionalities and to develop an android application on this IDE and put to test all our knowledge regarding android development and learn new program features along the way.

Our aim was to create a game that was appealing to the user's eyes and also relaxing and entertaining to play. This game is made for users of all ages and anyone can enjoy this simple game. We were motivated to use the kotlin functions that we learnt in our mobile application development laboratory such as Intent, setOnClickListener, findViewByID, makeToast, and so on and many attributes such as text, visibility, and so on. We also learnt to create our own functions to perform a set of desired instructions and execute an action. This game incorporates Firebase which is a set of backend cloud computing services and application development platforms provided by Google. It hosts databases, services, authentication, and integration for a variety of applications, including Android, iOS, JavaScript, Node.js, Java, Unity, PHP, and C++. This game was developed solely for entertainment purposes and to spark competitiveness among players. In these busy times where people spend most of their

days' time in doing work, they seek ways to entertain themselves so that they don't get burned out by their day-to-day work. Few minutes of playing a game can relax the users mind and they can feel fresh after playing a game and can function better in their daily work.

The 2D car game developed is simple and a fun game to play. The main objective of the game is for the player to control their car such a way that they are able to dodge all the obstacles which come in the way of the player's car.

1.4 MAD Technologies

In the mini project, the following mobile application development technologies are used:

- Kotlin: Kotlin is the primary programming language used for Android application development. It is a modern, expressive, and interoperable language that is fully supported by the Android platform. Kotlin offers concise syntax, null safety, and improved code readability, making it a popular choice for developing Android applications.
- Android Studio: Android Studio is the official Integrated Development Environment
- (IDE) for Android app development. It provides a comprehensiveset of tools and features specifically designed for building Android applications. Android Studio offers features like code editing, debugging, testing, and performance profiling, making it the preferred choice for Android developers.
- Internet Connectivity: The application requires internet connectivity to perform Google searches and potentially communicate with external OCR services. Standard Android technologies, such as the Network API or libraries like Retrofit or Volley, may be utilized to establish internet connections and handle network requests.

- Google Firebase: Firebase is a set of backend cloud computing services and application development platforms provided by Google. It hosts databases, services, authentication, and integration for a variety of applications, including Android, iOS, JavaScript, Node.js, Java, Unity, PHP, and C++.

1.5 Applications of MAD Technologies

Mobile application development technologies have a wide range of applications across various industries and use cases.

- E-commerce and Retail: Mobile apps are extensively used in the e-commerce and retail sector for online shopping, product browsing, and personalized shopping experiences. Technologies such as mobile payment gateways, push notifications, and location-based services enhance the user experience and facilitate seamless transactions.
- Social Networking: Social media platforms heavily rely on mobile application technologies for user engagement, content sharing, messaging, and real-time interactions. Features like news feeds, photo/video sharing, and social authentication are implemented using mobile app development technologies.
- Travel and Tourism: Mobile apps play a crucial role in the travel and tourism industry, offering features like flight bookings, hotel reservations, itinerary management, navigation, and reviews. Integration with external APIs for travel services and geolocation technologies are utilized to provide a seamless travel experience.
- Health and Fitness: Mobile health apps have become increasingly popular, allowing users to track their fitness goals, monitor vital signs, access medical information, and receive personalized health recommendations. Integration with wearables, fitness tracking sensors, and cloud-based data storage are common in this domain.
- Banking and Finance: Mobile banking apps provide users with the ability to

perform financial transactions, check account balances, view transaction histories, and manage investments. Security measures such as biometric authentication and encryption are implemented using mobile app development technologies.

- Education and E-learning: Mobile apps are used extensively in the education sector for e-learning, online courses, and remote education. They provide features like video lectures, interactive quizzes, progress tracking, and communication with instructors. Augmented reality (AR) and virtual reality (VR) technologies are also integrated into educational apps.
- Entertainment and Media: Mobile apps in the entertainment industry include video streaming platforms, music apps, gaming apps, and news aggregators. These apps leverage technologies like multimedia streaming, content recommendations, in-app purchases, and social sharing.
- Productivity and Business: Mobile apps improve productivity by offering tools for project management, task tracking, collaboration, document editing, and communication. Cloud storage integration, synchronization across devices, and integration with enterprise systems are common in business-oriented apps.

The versatility and widespread adoption of mobile apps across industries continue to drive innovation and provide solutions to diverse user needs.

CHAPTER 2

SYSTEM REQUIREMENTS

1.6 Hardware and Software Requirements

Hardware Requirements:

- Android Device: The application will be developed for Android devices. Any Android smartphone or tablet with a camera and sufficient processing power should be capable of running the application smoothly. The specific Android version supported by the application should be determined during the development process. Minimum android version is 7

Software Requirements:

- Android SDK: The Android Software Development Kit (SDK) is necessary for developing Android applications. It includes tools, libraries, and resources required to build, test, and debug Android apps. The SDK provides APIs for interacting with device features like the camera and internet connectivity.
- Kotlin Programming Language: The application will be developed using the Kotlin programming language. Kotlin is fully compatible with the Android platform and can be seamlessly integrated with existing Java code. It offers concise syntax, null safety, and various other features that enhance code readability and maintainability.
- Android Studio: Android Studio provides a comprehensive set of tools and features, including code editor, emulator, debugging tools, and project management capabilities. Android Studio simplifies the development process by offering a user-friendly interface and seamless integration with the Android SDK.

- Google Firebase: Firebase is a set of backend cloud computing services and application development platforms provided by Google. It hosts databases, services, authentication, and integration for a variety of applications, including Android, iOS, JavaScript, Node.js, Java, Unity, PHP, and C++.

CHAPTER 3

SYSTEM DESIGN

1.7 Proposed System

The proposed system is an Android mobile game developed in Android Studio using Kotlin language. The mobile game is 2D car game where the player or user is driving a car and controlling its movements on road. The player is required to move their right or left to dodge all obstacles in the game that come in their way. The moment player's car hits any of the obstacle, the game ends and the player's score is displayed on the screen. The player can then choose to either continue playing or exit the application.

The system focuses on providing a user-friendly and intuitive interface for seamless navigation throughout the game.

Key Features of the Proposed System:

- User-Friendly Interface: The system will feature a visually appealing and easy-to- navigate user interface. The user is provided with many options like start the game, view instructions to know how the game works, change backgrounds to some beautiful scenes and roads, login, register and logout from the application.
- Google Firebase: Firebase is a set of backend cloud computing services and application development platforms provided by Google. It hosts databases, services, authentication, and integration for a variety of applications, including Android, iOS, JavaScript, Node.js, Java, Unity, PHP, and C++.
- Internet Connectivity: The system will require internet connectivity to perform Google searches. It will utilize standard Android networking technologies or libraries like Retrofit or Volley to establish a connection with the internet and handle search requests. Internet access will be crucial for

retrieving search results and providing an enhanced user experience.

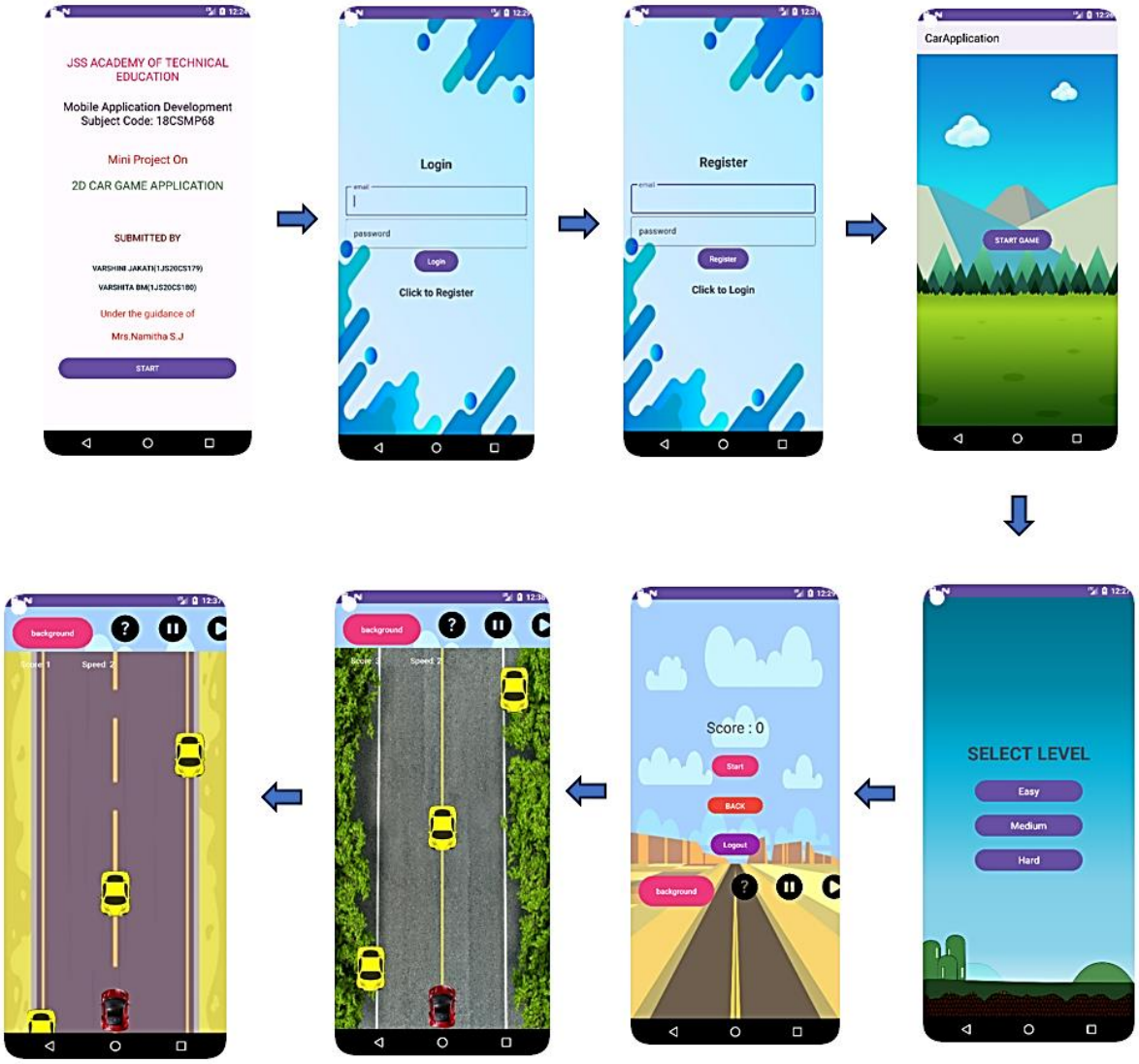
- Change background: The user is provided with a button to change the background of the game. They can choose from many different beautiful scenes to change the look and feel of the game.
- Challenges: The game poses many challenges to the user to make a game more interesting and intense. These challenges also spark the competitive spirits among users. The challenges are the yellow cars that approach the player's car in order to block the player car's path – these cars also increase their speed with time which makes it even more difficult to dodge, the player car's speed also increases in the course of the game.

The proposed system aims to provide entertainment to the user. The various sections of this game are appealing. The user is given an option to change the background of the game which changes the look and feel of the game. This game is developed to provide a simple user interface so that the user of any age can understand and play the game without any problems or confusion.

1.8 Flow of Activity

After launching the activity in an emulator, the first page that opens gives the front page of the mini project. After clicking the start button, the user is taken to a login page where he is supposed to provide his login credentials. If the user is new to the application then he cannot directly log in to the game. The user must register and create a new account by providing legit username and password in the email id and password plain text boxes. If the user has entered the right credentials then a prompt is displayed that reads "Account created" else another prompt is displayed that reads "Authentication failed" when either or both email id and password provided are incorrect. The application is connected to Google Firebase which authenticates the login and registration of users. Once a user has either logged in or registered as a new user, he can directly enter the game.

The main menu of the game provides the user with three options: start button to start the game, change background button to change the backgrounds of the game, and instructions button to view the instructions on how to play the game. On clicking the "Start" button, the game starts where the player's car starts moving in the forward direction. The player must avoid getting hit by yellow cars in order to survive in the game. If the player car gets hit by nay of the yellow cars, the game ends and their score is revealed. Then the user is provided with the same options again in the menu. He can choose to start the game again. If the user clicks on the "Change background" button, the current background of the game is changed to a new one which the user can enjoy. If the user clicks on the "Instructions" button then he is showed a bunch of instructions on how to play the game if the user is a new player or is confused about how the game works. The "logout" button is clicked to logout the user from the game and the user can log in again or any other user can login. The player faces many challenges in the game which were put in the game to make it more interesting and intense and to spark the competitiveness among users. The speed of the yellow cars as well as the player's car increases gradually which increases the difficulty level of the game and avoiding the yellow cars becomes difficult. The game has a very simple and appealing user interface so a user of any age can get an understanding of the mechanics of the game and enjoy it.


JSS ACADEMY OF TECHNICAL EDUCATION
Mobile Application Development
Subject Code: 18CSMP68
Mini Project On
2D CAR GAME APPLICATION
SUBMITTED BY
VARSHINI JAKATI(1JS20CS179)
VARSHITA BM(1JS20CS180)
Under the guidance of
Mrs.Namitha S.J
START
Login
email
password
Login
Click to Register
Register
email
password
Register
Click to Login
CarApplication
START GAME
SELECT LEVEL
Easy
Medium
Hard
Score : 0
Start
BACK
Logout
background
background
background

CHAPTER 4

IMPLEMENTATION

4.1 Module Description

Login Module

The "Login" module is a key component of an Android application that facilitates user authentication and login functionality. From a technical standpoint, the module imports necessary dependencies and extends the "AppCompatActivity" class. It declares variables to reference user interface elements like EditText, Button, and TextView.

In terms of functionality, the module allows users to enter their username and password. When the login button is clicked, through Firebase connectivity the login credentials are authenticated. If the usename and password typed by the user are incorrect, the authentication fails and a prompt is displayed reading “Authentication failed.” If the username and password entered by the user are legit, the login is successful and “Login successful” message is displayed on the screen. Once the authentication is successful, this module creates an intent to launch the MainActivity module to start the game.

From a functional perspective, the Login module ensures that only authorized users can access the application and its features. It enhances security by validating the provided credentials and prevents unauthorized access. The module also provides a smooth user experience by displaying error messages for invalid credentials and clearing input fields for retry.

Register Module

The "Register" module is an essential part of the Android application and is responsible for handling user signup or registration functionality. The module consists of variable declarations for UI elements like EditText and Button. It allows users to enter their desired username and password for creating a new account.

From a technical standpoint, the Register module imports necessary dependencies and extends the "AppCompatActivity" class. It initializes the UI elements in the onCreate() method and sets up click listeners for the register button.

When the register button is clicked, the module retrieves the entered username and password from the EditText fields and the Firebase connectivity authenticates those values. It checks whether the entered amil ID is legit. If the entered new user credentials are correct then a message is displayed that reads “Account created”. Then the Register module creates an intent to launch the MainActivity module and opens the main menu of the game.

Functionally, the Register module enables users to create a new account by entering a username and password. It enhances user experience by providing a smooth transition to the MainActivity after successful registration. From a security perspective, the module facilitates the creation of new user accounts, allowing for personalized access tothe application's features and resources.

The Register module combines technical implementation, such as variable initialization and intent creation, with functional aspects like user account creation and navigation, to deliver a seamless and intuitive signup process within the Android application.

MainActivity Module

The "MainActivity" module is a key component of an Android application that consists of the main game. The main menu of the game provides the user with three options: start button to start the game, change background button to change the backgrounds of the game, and instructions button to view the instructions on how to play the game. On clicking the “Start” button, the game starts where the player’s car starts moving in the forward direction. The player must avoid getting hit by yellow

cars in order to survive in the game. If the player car gets hit by nay of the yellow cars, the game ends and their score is revealed. Then the user is provided with the same options again in the menu. He can choose to start the game again. If the user clicks on the "Change background" button, the current background of the game is changed to a new one which the user can enjoy. If the user clicks on the "Instructions" button then he is showed a bunch of instructions on how to play the game if the user is a new player or is confused about how the game works. The "logout" button is clicked to logout the user from the game and the user can log in again or any other user can login. The player faces many challenges in the game which were put in the game to make it more interesting and intense and to spark the competitiveness among users. The speed of the yellow cars as well as the player's car increases gradually which increases the difficulty level of the game and avoiding the yellow cars becomes difficult. The game has a very simple and appealing user interface so a user of any age can get an understanding of the mechanics of the game and enjoy it.

4.2 Source code

MainActivity.kt

```
package com.example.carapplication

import android.content.Intent
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.view.View
import android.widget.Button
import android.widget.LinearLayout
import android.widget.TextView
import com.google.android.material.color.utilities.Score
import com.google.firebase.auth.FirebaseAuth
```

```
class MainActivity : AppCompatActivity(), GameTask {
    private lateinit var rootLayout: LinearLayout
    private lateinit var startBtn: Button
    private lateinit var mGameView: GameView
    private lateinit var score: TextView
    private lateinit var pauseBtn: Button
    private lateinit var resumeBtn: Button
    private lateinit var logoutBtn: Button


    private val backgrounds = Backgrounds()
    private lateinit var backgroundButton: Button

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)

        // Check if the user is logged in
        val currentUser = FirebaseAuth.getInstance().currentUser
        if (currentUser == null) {
            // User is not logged in, start the Login activity
            val intent = Intent(this, Login::class.java)
            startActivity(intent)
            finish() // Finish the MainActivity to prevent going back to it after login
        } else {
            // User is logged in, proceed to the game page
            setContentView(R.layout.activity_main)
            setupGamePage()
            backgroundButton = findViewById(R.id.backgroundButton)
            backgroundButton.setOnClickListener {
                updateBackground()
            }
            val instructionsButton: Button = findViewById(R.id.instructionsButton)
            instructionsButton.setOnClickListener {
                val intent = Intent(this, instructions::class.java)
```

```
            startActivity(intent)
        }

        pauseBtn = findViewById(R.id.pauseBtn)
        resumeBtn = findViewById(R.id.resumeBtn)
        logoutBtn = findViewById(R.id.logoutBtn)

        pauseBtn.setOnClickListener {
            pauseGame()
        }

        resumeBtn.setOnClickListener {
            resumeGame()
        }

        logoutBtn.setOnClickListener {
            logoutUser()
        }

    }
}

private fun setupGamePage() {
    startBtn = findViewById(R.id.startBtn)
    rootLayout = findViewById(R.id.rootLayout)
    score = findViewById(R.id.score)
    mGameView = GameView(this, this)

    startBtn.setOnClickListener {
        startGame()
    }
}

private fun startGame() {
```

```
    updateBackground()
    rootLayout.addView(mGameView)
    startBtn.visibility = View.GONE
    score.visibility = View.GONE
    backgroundButton.visibility = View.GONE
    pauseBtn.visibility = View.VISIBLE
    resumeBtn.visibility = View.VISIBLE
    logoutBtn.visibility = View.GONE
    backgroundButton.setOnClickListener(){
        updateBackground()
    }
}

private fun updateBackground() {
    val background = backgrounds.getBackground()
    mGameView.setBackgroundResource(background)
}

override fun closeGame(mScore: Int) {
    score.text = "Score: $mScore"
    rootLayout.removeView(mGameView)
    startBtn.visibility = View.VISIBLE
    score.visibility = View.VISIBLE
    backgroundButton.visibility = View.VISIBLE
    pauseBtn.visibility = View.GONE
    resumeBtn.visibility = View.GONE
    logoutBtn.visibility = View.VISIBLE
    setupGamePage()
}

private fun pauseGame() {
    mGameView.pauseGame()
}
```

```
    private fun resumeGame() {
        mGameView.resumeGame()
    }

    private fun logoutUser() {
        FirebaseAuth.getInstance().signOut()
        val intent = Intent(this, Login::class.java)
        startActivity(intent)
        finish()
    }

}
```

Register.kt

```
package com.example.carapplication;

import android.content.Intent;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.View;
import android.widget.Button;
import android.widget.ProgressBar;
import android.widget.TextView;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.android.material.textfield.TextInputEditText;
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;
```

```
import com.google.firebase.auth.FirebaseUser;

public class Register extends AppCompatActivity {

    TextInputEditText editTextEmail, editTextPassword;
    Button buttonReg;
    FirebaseAuth mAuth;
    ProgressBar progressBar;
    TextView textView;

    @Override
    public void onStart() {
        super.onStart();
        // Check if user is signed in (non-null) and update UI accordingly.
        FirebaseUser currentUser = mAuth.getCurrentUser();
        if(currentUser != null){
            Intent intent = new Intent(getApplicationContext(), MainActivity.class);
            startActivity(intent);
            finish();

        }
    }
    @Override
    protected void onCreate(Bundle savedInstanceState){
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_register);
        mAuth = FirebaseAuth.getInstance();
        editTextEmail = findViewById(R.id.email);
        editTextPassword = findViewById(R.id.password);
        buttonReg = findViewById(R.id.btn_register);
progressBar = findViewById(R.id.progressBar);
textView = findViewById(R.id.loginNow);
textView.setOnClickListener(new View.OnClickListener() {
    @Override
```

```
    public void onClick(View view) {
        Intent intent = new Intent(getApplicationContext(), Login.class);
        startActivity(intent);
        finish();
    }
});
        buttonReg.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                progressBar.setVisibility(View.VISIBLE);
                String email, password;
                email = String.valueOf(editTextEmail.getText());
                password = String.valueOf(editTextPassword.getText().toString());

                if (TextUtils.isEmpty(email)){
                    Toast.makeText(Register.this, "Enter email", Toast.LENGTH_SHORT).show();
                    return;
                }

                if (TextUtils.isEmpty(password)){

                    Toast.makeText(Register.this, "Enter password", Toast.LENGTH_SHORT).show();
                    return;

                }
                mAuth.createUserWithEmailAndPassword(email, password)
                        .addOnCompleteListener(new OnCompleteListener<AuthResult>() {
                            @Override
                            public void onComplete(@NonNull Task<AuthResult> task) {
                                progressBar.setVisibility(View.GONE);
                                if (task.isSuccessful()) {

                                    Toast.makeText(Register.this, "Accout created.",
                                            Toast.LENGTH_SHORT).show();
```

```
                    } else {
                        // If sign in fails, display a message to the user.

                        Toast.makeText(Register.this, "Authentication failed.",
                                Toast.LENGTH_SHORT).show();

                    }
                }
            });
        }
    });

  }
}
```

Login.java

```
package com.example.carapplication;

import android.content.Intent;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.View;
import android.widget.Button;
import android.widget.ProgressBar;
import android.widget.TextView;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import com.google.android.gms.tasks.OnCompleteListener;
```

```
import com.google.android.gms.tasks.Task;
import com.google.android.material.textfield.TextInputEditText;
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;

public class Login extends AppCompatActivity {

    TextInputEditText editTextEmail, editTextPassword;
    Button buttonLogin;
    FirebaseAuth mAuth;
    ProgressBar progressBar;
    TextView textView;

    @Override
    public void onStart() {
        super.onStart();
        // Check if user is signed in (non-null) and update UI accordingly.
        FirebaseUser currentUser = mAuth.getCurrentUser();
        if(currentUser != null){
            Intent intent = new Intent(getApplicationContext(), MainActivity.class);
            startActivity(intent);
            finish();

        }
    }

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

        mAuth = FirebaseAuth.getInstance();
        editTextEmail = findViewById(R.id.email);
        editTextPassword = findViewById(R.id.password);
```

```
buttonLogin = findViewById(R.id.btn_login);
progressBar = findViewById(R.id.progressBar);
textView = findViewById(R.id.registerNow);
textView.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        Intent intent = new Intent(getApplicationContext(), Register.class);
        startActivity(intent);
        finish();
    }
});

buttonLogin.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view){
        progressBar.setVisibility(View.VISIBLE);
        String email, password;
        email = String.valueOf(editTextEmail.getText());
        password = String.valueOf(editTextPassword.getText().toString());

        if (TextUtils.isEmpty(email)){
            Toast.makeText(Login.this, "Enter email", Toast.LENGTH_SHORT).show();
            return;
        }

        if (TextUtils.isEmpty(password)){

            Toast.makeText(Login.this, "Enter password", Toast.LENGTH_SHORT).show();
            return;

        }

        mAuth.signInWithEmailAndPassword(email, password)
                .addOnCompleteListener(new OnCompleteListener<AuthResult>() {
                    @Override
```

```
                    public void onComplete(@NonNull Task<AuthResult> task) {
                        progressBar.setVisibility(View.GONE);
                        if (task.isSuccessful()) {
                            Toast.makeText(getApplicationContext(),"Login Successful",
Toast.LENGTH_SHORT).show();
                            Intent intent = new Intent(getApplicationContext(), MainActivity.class);
                            startActivity(intent);
                            finish();
                           } else {
                            Toast.makeText(Login.this, "Authentication failed.",
                                    Toast.LENGTH_SHORT).show();

                        }
                    }
                });
        }
    });
}
}
```

GameView.kt

```
package com.example.carapplication

import android.content.Context
import android.graphics.Canvas
import android.graphics.Color
import android.graphics.Paint
import android.view.MotionEvent
import android.view.View

class GameView(var c: Context, var gameTask: GameTask) : View(c) {
    private var myPaint: Paint? = null
    private var speed = 1
```

```
private var time = 0
private var score = 0
private var myCarPosition = 0
private val otherCars = ArrayList<HashMap<String, Any>>()
private var isPaused = false

var viewWidth = 0
var viewHeight = 0

init {
    myPaint = Paint()
}

override fun onDraw(canvas: Canvas?) {
    super.onDraw(canvas)
    viewWidth = this.measuredWidth
    viewHeight = this.measuredHeight

    if (!isPaused) {
        if (time % 700 < 10 + speed) {
            val map = HashMap<String, Any>()
            map["lane"] = (0..2).random()
            map["startTime"] = time
            otherCars.add(map)
        }
        time = time + 10 + speed
    }

    val carWidth = viewWidth / 5
    val carHeight = carWidth + 10
    myPaint!!.style = Paint.Style.FILL

    // Draw my car
    val myCarDrawable = resources.getDrawable(R.drawable.cars_reds, null)
    myCarDrawable.setBounds(
```

```
        myCarPosition * viewWidth / 3 + viewWidth / 15 + 25,
        viewHeight - 2 - carHeight,
        myCarPosition * viewWidth / 3 + viewWidth / 15 + carWidth - 25,
        viewHeight - 2
    )
    myCarDrawable.draw(canvas!!)

    // Draw other cars
    val carType1Drawable = resources.getDrawable(R.drawable.cars_yellows, null)
    for (i in otherCars.indices) {
        try {
            val carX = otherCars[i]["lane"] as Int * viewWidth / 3 + viewWidth / 15
            var carY = time - otherCars[i]["startTime"] as Int

            val drawable = carType1Drawable

            drawable.setBounds(
                carX + 25, carY - carHeight, carX + carWidth - 25, carY
            )
            drawable.draw(canvas)

            if (otherCars[i]["lane"] as Int == myCarPosition) {
                if (carY > viewHeight - 2 - carHeight && carY < viewHeight - 2) {
                    gameTask.closeGame(score)
                }
            }

            if (carY > viewHeight + carHeight) {
                otherCars.removeAt(i)
                score++
                speed = 1 + Math.abs(score / 8)
            }
        } catch (e: Exception) {
            e.printStackTrace()
        }
```

```
    }

    myPaint!!.color = Color.WHITE
    myPaint!!.textSize = 40f
    canvas.drawText("Score: $score", 80f, 80f, myPaint!!)
    canvas.drawText("Speed: $speed", 380f, 80f, myPaint!!)
    invalidate()
}

override fun onTouchEvent(event: MotionEvent?): Boolean {
    if (!isPaused) {
        when (event?.action) {
            MotionEvent.ACTION_DOWN -> {
                val x1 = event.x
                if (x1 < viewWidth / 2) {
                    if (myCarPosition > 0) {
                        myCarPosition--
                    }
                }
                if (x1 > viewWidth / 2) {
                    if (myCarPosition < 2) {
                        myCarPosition++
                    }
                }
                invalidate()
            }
            MotionEvent.ACTION_UP -> {
            }
        }
    }
    return true
}

fun pauseGame() {
    isPaused = true
```

```
    }

    fun resumeGame() {
        isPaused = false
    }
}
```

loginpage.java

```
package com.example.carapplication;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;

import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;

public class loginpage extends AppCompatActivity {


    FirebaseAuth auth;

    Button button;
    TextView textView;
    FirebaseUser user;
@Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.loginpage);
```

```
    auth = FirebaseAuth.getInstance();
    button = findViewById(R.id.logout);

    textView = findViewById(R.id.user_details);

    user =auth.getCurrentUser();
          if(user ==null)

    {
       Intent intent = new Intent(getApplicationContext(), Login.class);
       startActivity(intent);
       finish();

    }
          else {
             textView.setText(user.getEmail());
          }

          button.setOnClickListener(new View.OnClickListener() {
             @Override
             public void onClick(View view) {
                FirebaseAuth.getInstance().signOut();
                Intent intent = new Intent(getApplicationContext(), Login.class);
                startActivity(intent);
                finish();
             }
          });

    }
}
```

Instructions.kt

```
package com.example.carapplication

import android.os.Bundle
import android.view.View
import android.widget.Button
import androidx.appcompat.app.AppCompatActivity
import com.example.carapplication.R

class instructions : AppCompatActivity() {

    private lateinit var backBtn: Button

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.instructions)

        backBtn = findViewById(R.id.backBtn)
        backBtn.setOnClickListener {
            // Handle back button click
            finish() // Close the instructions activity and go back to the previous activity (main activity)
        }
    }
}
```

GameTask.kt

```
package com.example.carapplication

interface GameTask {
    fun closeGame(mScore:Int)
}
```

Firstpage.kt

```
package com.example.carapplication

import android.content.Intent
import android.os.Bundle
import androidx.appcompat.app.AppCompatActivity
import com.example.carapplication.databinding.FirstpageBinding

class firstpage : AppCompatActivity() {
    private lateinit var binding: FirstpageBinding

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        binding = FirstpageBinding.inflate(layoutInflater)
        val view = binding.root
        setContentView(view)

        binding.STARTButton.setOnClickListener {
            val intent = Intent(this, Register::class.java)
            startActivity(intent)
            finish()
        }
    }
}
```

Backgrounds.kt

```
package com.example.carapplication

import kotlin.random.Random

class Backgrounds {
    private val backgrounds = arrayOf(
```

```
        R.drawable.bg1,
        R.drawable.bg2,
        R.drawable.bg3,
        R.drawable.road
    )

    fun getBackground(): Int {
        val randomIndex = (0 until backgrounds.size).random()
        return backgrounds[Random.nextInt(backgrounds.size)]
        }
}
```

CHAPTER 5:

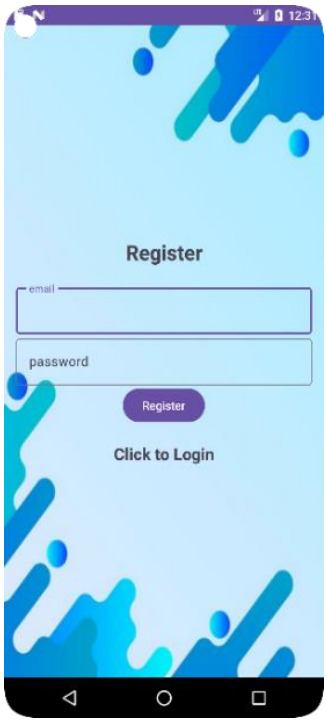
RESULTS

Front Page

Fig 5.1 Front Page

This is the front page of the application which is shown as soon as the application is launched. This page displays the mini project description It consists of many TextView components which contain text as shown in the above figure. Linear Layout is used to better organize and distinguish the TextView components.. On clicking the “Start” button the user will be directed to the registration page of the application to register his account.

Register



The registration page offers a simple user interface as seen in the fig 5.1 It has a TextView that displays title of the page “Register” and a TextView at the bottom “Click to Login”. TextInputLayout box for entering username and a text input layout with password toggle enabled for entering password.

The user has to fill in the username and password fields with valid strings and click on register button to register as a new user. If the entered credentials are authenticated by Firebase then the user is logged in and the game’s main menu is launched.

Loginpage

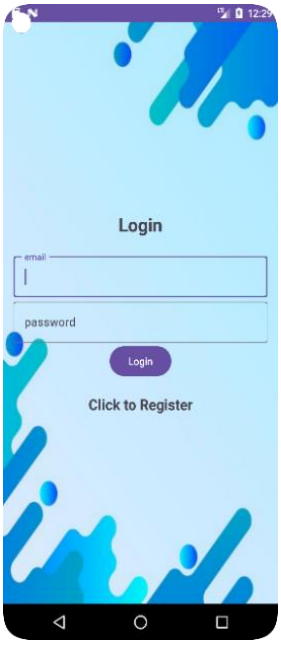


Fig 5.3 Login Page

The login page offers a simple user interface as seen in the fig 5.2 and has a TextView for displaying the title of the page “Login” and another TextView at the bottom “Click to Login”. A TextInputLayout for entering username and a text input layout with password toggle enabled for entering password.

The user has to fill in the username and password fields with valid strings and click on register button to login as an existing user. If the entered credentials are authenticated by Firebase then the user is logged in and the game’s main menu is launched. If the entered username and password are incorrect then a toast message is displayed reading “Authentication failed”. If the user is a new user then he must click on the Click To Register to register as a new user.

MainActivity

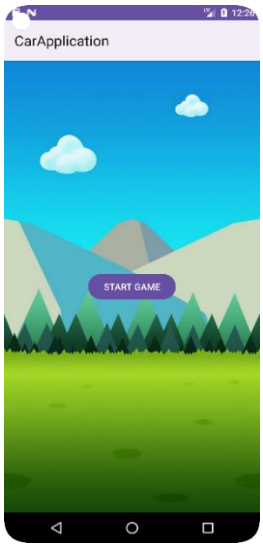


Fig 5.4 Start Screen

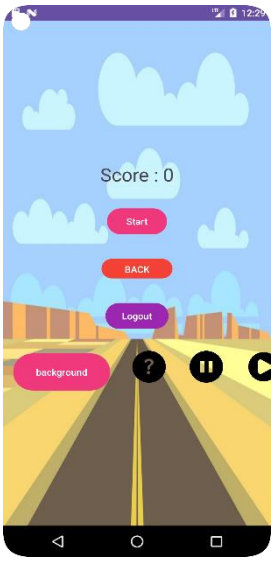


Fig 5.5 Game Screen

This is the main menu of the game and it contains many different options from which the user can choose from.At the very top there is a TextView that displays the Score of the player after he has lost the game. It resets to 0 whenever the player starts a new game. On clicking he 'Start' button the application launches an intent to start the game, on clicking the 'change background' button the user is allowed to change the background to any of the beautiful scenens, the 'pause' button lets the user pause the game, the 'resume' game lets user to resume the paused game and play again, on clicking the 'instructions' button the user can view the istructions in case the user is new to the game or want to have a better understanding of the game mechansics of the game.

The 'logout' button to log the user out of the game and then the user can wish to login again or a new user can register a new account.

ActivitySelectMenu.kt

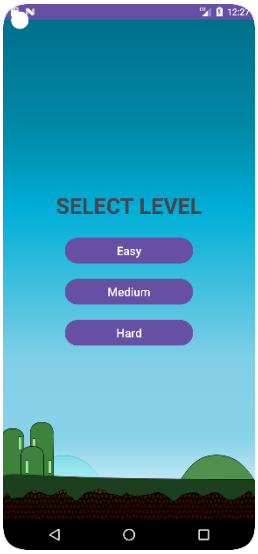


Fig 5.6 Level Select menu

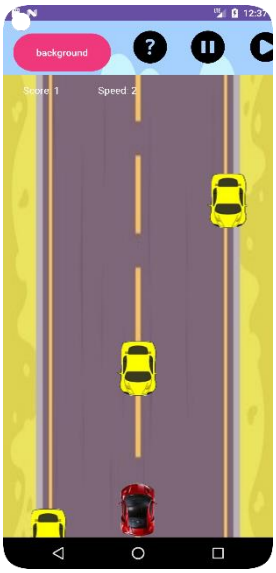


Fig 5.7 Start game

Figure 5.6 shows the level select menu. The menu shows easy, medium, hard levels. The user can select from any of the above levels based on level difficulty.

Figure 5.7 Start Game screen shows the screen when the user selects the level they want to play. The easy level is where the user starts with speed 1 and hard level starts with speed 4.

Instructions

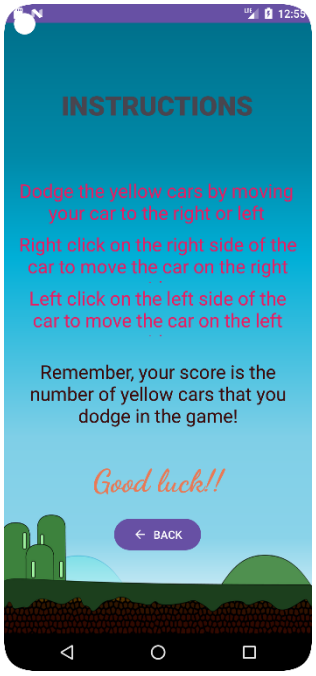


Fig 5.7 Instructions

This is the instructions page which displays the instructions on how to play the game for the users who are new to the game and want to get a better understading of the working of the game. It consists of TextView components which read the instructions of the game and a button ‘Back’ at the bottom of the screen to go back to the main menu of the game and resume playing the game.

CONCLUSIONS

The proposed system is an Android mobile game developed in Android Studio using Kotlin language. The mobile game in question is a 2D car game which involves a player controlling the movement of a car. The 2D car game developed is simple and a fun game to play. A user of any age can find this game relaxing and entertaining. In today's world gaming application plays an important role as it offers entertainment, social interaction, technological advancements. They have become an important aspect of modern world, shaping how people interact with technology. This application incorporates usage of cloud services such as Google Firebase for authentication of accounts created by the users to play this game.

This application can be easily run on any android device as it requires minimum android version 7 and it is essential to create games that are not only fun and engaging but also optimized for the platforms they are intended to run on.

This game will provide an engaging and an enjoyable gaming experience that challenges player's skills, provides excitement, encourages competition, and promotes cognitive development.

FUTURE ENHANCEMENTS

There are several potential future enhancements you can consider for your 2D car game app using Kotlin in Android Studio. Here are a few ideas to get you started:

Multiplayer functionality: Implement multiplayer support to allow users to compete with their friends or other players online. You can integrate real-time multiplayer features using technologies like Google Play Games Services or Firebase Realtime Database. Additional game modes: Introduce new game modes to provide variety and keep users engaged. For example, you could add a time trial mode, where players compete to complete laps within a given time limit, or a challenge mode with specific objectives to accomplish. Power-ups and bonuses: Incorporate power-ups or bonuses that players can collect during gameplay. These power-ups can give temporary speed boosts, shields for protection, or other special abilities to enhance the gameplay experience. Customizable cars: Allow players to customize their cars with different colors, body styles, or even unlockable parts. This customization feature adds a personal touch and can increase user engagement and satisfaction. Leaderboards and achievements: Implement a leaderboard system to showcase the highest scores or fastest lap times achieved by players. Additionally, integrate achievements to reward players for completing specific challenges or milestones. Improved graphics and effects: Enhance the visual experience by improving the game's graphics, animations, and special effects. This can include better car models, realistic physics simulations, dynamic lighting, particle effects, and more. Level editor: Create a level editor tool within the app that allows users to design their own tracks or courses. This feature empowers players to unleash their creativity and share their custom levels with others. Social media integration: Enable users to share their achievements, high scores, or gameplay screenshots directly from the app to social media platforms. Integrating social media APIs can help increase user engagement and promote your game. In-app purchases: Implement a system for in-app purchases, such as allowing players to buy additional cars, power-ups, or cosmetic items. Ensure that the purchases are well-balanced and do not negatively impact the game's fairness.

REFERENCES

- https://developer.android.com
- https://stackoverflow.com
- https://www.youtube.com
- https://www.udemy.com