VISVESVARAYA TECHNOLOGICAL UNIVERSITY

JNANA SANGAMA, BELAGAVI – 590 018

An Internship Project Report
on

Flutter Quiz App

Submitted in partial fulfillment of the requirements for the VII Semester of degree of **Bachelor of Engineering in Information Science and Engineering** of Visvesvaraya Technological University, Belagavi

Submitted By

Divyanshu Kumar
1RN18IS042

Under the Guidance of

Mrs. Shreedevi Suresh
Assistant Professor
Department of ISE

ESTD:2001
An Institute with a Difference

Department of Information Science and Engineering

RNS Institute of Technology

Dr. Vishnuvaradhan Road, Rajarajeshwari Nagar post,
Channasandra, Bengaluru-560098

2021-2022

RNS INSTITUTE OF TECHNOLOGY

Dr. Vishnuvaradhan Road, Rajarajeshwari Nagar post,
Channasandra, Bengaluru - 560098

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

CERTIFICATE

Certified that the Internship work entitled ***Flutter Quiz App*** has been successfully completed by **Divyanshu Kumar (1RN18IS42)** a Bonafede student of **RNS Institute of Technology, Bengaluru** in partial fulfillment of the requirements of 8th semester for the award of degree in **Bachelor of Engineering in Information Science and Engineering** of **Visvesvaraya Technological University, Belagavi** during academic year **2021-2022.** The internship report has been approved as it satisfies the academic requirements in respect of internship work for the said degree.

Mrs. Shreedevi Suresh	**Dr. Suresh L**	**Dr. M K Venkatesha**
Internship Guide	Professor and HoD	Principal
Assistant Professor	Department of ISE	RNSIT
Department of ISE	RNSIT	

External Viva

Name of the Examiners	**Signature with Date**
1. _____	**1.** _____
2. _____	**2.** _____

DECLARATION

I, **Divyanshu Kumar [USN: 1RN18IS042]** student of VII Semester BE, in Information Science and Engineering, RNS Institute of Technology hereby declare that the Internship work entitled ***Flutter Quiz App*** has been carried out by us and submitted in partial fulfillment of the requirements for the *VII Semester degree of* ***Bachelor of Engineering in Information Science and Engineering*** *of Visvesvaraya Technological University, Belagavi* during academic year 2021-2022.

Place: Bengaluru
Date:

Divyanshu Kumar (1RN18IS042)

ABSTRACT

The project: "Flutter Quiz App" is a collection of number of different types of questions on technical games, sports, etc. A user can access/play all of the question and can attempt any of the one. There will be limited number of questions and for each correct answer user will get a credit score. User can see answers as well as can ask a query related to it.

There are many quiz applications available currently on internet. But there are few Which provide better understanding between users and the application like, providing proper answers, user query solving, uploading user questions as well as answer to it, etc. To develop a User friendly quiz application which will contain: Numbers of quiz. Answers to every question, Query solving regarding any question, Uploading of user question and answer, and to improve the knowledge level of users.

To develop a application which will contain solution to the above problems. By this application the user will come to know about his/her level and can leam additional knowledge. Also by this application a user can expand his/her knowledge among the world.

ACKNOWLEDGMENT

At the very onset we would like to place our gratefulness to all those people who helped me in making the Internship a successful one.

Coming up, this internship to be a success was not easy. Apart from the sheer effort, the enlightenment of the very experienced teachers also plays a paramount role because it is, they who guided me in the right direction.

First of all, I would like to thank the **Management of RNS Institute of Technology** for providing such a healthy environment for the successful completion of internship work.

In this regard, I express sincere gratitude to our beloved Principal **Dr. M K Venkatesha,** for providing us all the facilities.

We are extremely grateful to our own and beloved Professor and Head of Department of Information science and Engineering, **Dr. Suresh L**, for having accepted to patronize me in the right direction with all her wisdom.

We place our heartfelt thanks to **Mrs. Shreedevi Suresh** Assistant Professor and HOD, Department of Information Science and Engineering for having guided internship and all the staff members of the department of Information Science and Engineering for helping at all times.

I thank **Mr. Akshay D R, ENMAZ** , for providing the opportunity to be a part of the Internship program and having guided me to complete the same successfully.

I also thank our internship coordinator **Dr. R Rajkumar,** Associate Professor, Department of Information Science and Engineering. I would thank my friends for having supported me with all their strength and might. Last but not the least, I thank my parents for supporting and encouraging me throughout. I have made an honest effort in this assignment.

TABLE OF CONTENTS

LIST OF FIGURES

ABBREVIATIONS

Acronym	Description
ADO	Active X Data Object
SQL	Structed Query Language
MSSQL	Microsoft SQL Server
HTML	Hypertext Markup Language
CSS	Cascading Style Sheets
CLR	Common Language runtime
IE	Internet Explorer
VB	Visual Basics
ISO	International Organization of Standardization
ANSI	American National Standard Institutes

1.INTRODUCTION

1.1 Introduction to Flutter

Flutter is Google's Mobile SDK to build native iOS and Android, Desktop (Windows, Linux, macOS), Web apps from a single codebase. When building applications with Flutter everything towards Widgets – the blocks with which the flutter apps are built. They are structural elements that ship with a bunch of material design-specific functionalities and new widgets can be composed out of existing ones too. The process of composing widgets together is called composition. The User Interface of the app is composed of many simple widgets, each of them handling one particular job. That is the reason why Flutter developers tend to think of their flutter app as a tree of widgets.

1.2 History

Flutter launched as a project called Sky which at the beginning worked only on Android. Flutter's goal is enabling developers to compile for every platform using its own graphic layer rendered by the Skia engine. Here's a brief presentation of Flutter's relatively short history.

Flutter is a free and open-source mobile UI framework created by Google and released in May 2017. In a few words, this allows you to create a native mobile application with only one code. It means that you can use one programming language and one codebase to create two different apps (IOS and Android).

The first version of Flutter was known by the codename "Sky" and ran on the Android operating system. It was unveiled at the 2015 Dart developer summit[6] with the stated intent of being able to render consistently at 120 frames per second.[7] During the keynote of Google Developer Days in Shanghai in September 2018, Google announced Flutter Release Preview 2, which is the last big release before Flutter 1.0. On December 4th of that year, Flutter 1.0 was released at the Flutter Live event, denoting the first "stable" version of the Framework. On December 11, 2019, Flutter 1.12 was released at the Flutter Interactive event.[8]

On May 6, 2020, the Dart software development kit (SDK) in version 2.8 and the Flutter in version 1.17.0 were released, where support was added to the Metal API, improving performance on iOS devices (approximately 50%), new Material widgets, and new network tracking.

On March 3, 2021, Google released Flutter 2 during an online Flutter Engage event. This major update brought official support for web-based applications with new CanvasKit renderer and web specific widgets, early-access desktop application support for Windows, macOS, and Linux and improved Add-to-App APIs.[9] This release included sound null-safety, which caused many breaking changes and issues with many external packages, but the Flutter team included instructions to mitigate these changes as well.

On September 8th, 2021, the Dart SDK in version 2.14 and Flutter version 2.5 were released by Google. The update brought improvements to the Android Full-Screen mode and the latest version of Google's Material Design called Material You. Dart received two new updates, the newest lint conditions have been standardized and preset as the default conditions as well Dart for Apple Silicon is now stable.

1.1.1 Framework-Architecture

The major components of Flutter include:

- Dart platform
- Flutter engine
- Foundation library
- Design-specific widgets
- Flutter Development Tools (DevTools)

Dart platform

Flutter apps are written in the Dart language and make use of many of the language's more advanced features.

On Windows, macOS, and Linux[11] Flutter runs in the Dart virtual machine, which features a just-in-time execution engine. While writing and debugging an app, Flutter uses Just In Time compilation, allowing for "hot reload", with which modifications to source files can be injected into a running application. Flutter extends this with support for stateful hot reload, where in most cases changes to source code are reflected immediately in the running app without requiring a restart or any loss of state.

For better performance, release versions of Flutter apps targeting Android and iOS are compiled with ahead-of-time (AOT) compilation.

Flutter engine

Flutter's engine, written primarily in C++, provides low-level rendering support using Google's Skia graphics library. Additionally, it interfaces with platform-specific SDKs such as those provided by Android and iOS.[10] The Flutter Engine is a portable runtime for hosting Flutter applications. It implements Flutter's core libraries, including animation and graphics, file and network I/O, accessibility support, plugin architecture, and a Dart runtime and compile toolchain. Most developers interact with Flutter via the Flutter Framework, which provides a reactive framework and a set of platform, layout, and foundation widgets.

Foundation library

The Foundation library, written in Dart, provides basic classes andfunctions that are used to construct applications using Flutter, such as APIs to communicate with the engine.

Design-specific widgets

The Flutter framework contains two sets of widgets that conform to specific design languages: Material Design widgets implement Google's design language of the same name, and Cupertino widgets implement Apple's iOS Human interface guidelines.

2.LITERATURE SURVEY

2.1. FLUTTER QUIZ APP

A quiz application can be designed as client server style architecture, multi tier architecture or as an stand alone application. In case of application design with a client server architecture which uses a server to send data continuously to the mobile application system needs bandwidth to run application and load on the server will be depending on the number of applicants using application. One of disadvantage is that we are dependent on network connectivity and in case of network failure complete system will be fail. To overcome some of these issues this application have been designed with slight change in design. In this application SQLLite data base is used and all questions and answers are stored in the database. Each time a teacher wants to conduct a quiz. All subject / topic wise questions are stores in the application apk file and distributed to the participants. Distribution can be done using blue tooth / google play /by en drive also. So dependency of network is reduced. One application apk is usable for once and for one subject / topic only. After the exam is finished user will have to install apk so there is no harm. For the next topic the concern Instructor will set new questions and their answer and will distribute it again. So a student can appear in the test at any time from remote also.The roles of different users in this application are as under.

2.1.1 Introduction

A learning environment is any environment in which students become totally involved in the learning process. Since the mobile devices support the anytime, anywhere learning, m-learning can faster the growth of the learning. Andro Quiz application enables the learner to access the learning object and interact with the instructor and other learner seamlessly from the mobile / tablet / aakash tablet while in class, from his android mobile phone during travelling or at home. Mobile learning provides the freedom from learning environments, learning devices and learning content format and rather emphasize on the constructivist learning process and cognitive development among learners. With the use of mart mobile devices with wireless networks enables mobility and mobile learning, allowing teaching and learning to extend to spaces beyond the

traditional classroom. The evolution of today's mobile devices increases the number of mobile applications developed, and among them the mobile learning applications.

3.ANALYSIS

3.1 Hardware and Software Requirements

The Hardware requirements are very minimal and the program can be run on most of the machines.

Processor	:	Pentium 4 Processor
Processor Speed	:	2.4 GHz
RAM	:	2 GB
Storage Space	:	40 GB

The software requirements are very minimal and the program can be run on the machines with these requirements satisfied:

Editor	:	Visual Studio Code
Operating System	:	Windows/Mac OS
IDE	:	VS Code
Backend Tool	:	SQLite

3.2 Tools/ Languages/ Platform

Various tool used in making this project is given below:

Editor/IDE	:	Visual Studio Code
Operating System	:	Windows/Mac OS
Languages	:	Dart, Swift, SQLite
Backend Tool	:	SQLite

3.3 Functional Requirements

Flutter

Flutter is Google's Mobile SDK to build native iOS and Android apps from a single codebase. When building applications with Flutter everything towards Widgets – the blocks with which the flutter apps are built. The User Interface of the app is composed of many simple widgets, each of them handling one particular job. That is the reason why Flutter developers tend to think of their flutter app as a tree of widgets.

Compared to its contemporary technologies like React Native, Kotlin, and Java, Flutter is much better in regard to having a Single Codebase for Android and iOS, Reusable UI and Business Logic, high compatibility, performance, and productivity.

Dart

Dart is an open-source general-purpose programming language developed by Google. It supports application development in both client and server-side. But it is widely used for the development of android apps, iOS apps, IoT(Internet of Things), and web applications using the Flutter Framework.

Syntactically, Dart bears a strong resemblance to Java, C, and JavaScript. It is a dynamic object-oriented language with closure and lexical scope. The Dart language was released in 2011 but came into popularity after 2015 with Dart 2.0.

SQLite

SQLite is a self-contained, high-reliability, embedded, full-featured, public-domain, SQL database engine. It is the most used database engine in the world. It is an in-process library and its code is publicly available. It is free for use for any purpose, commercial or private. It is basically an embedded SQL database engine. The SQLite database file format is cross-platform s

4.SYSTEM DESIGN

4.1 QUIZ PAGE WIDGET TREE

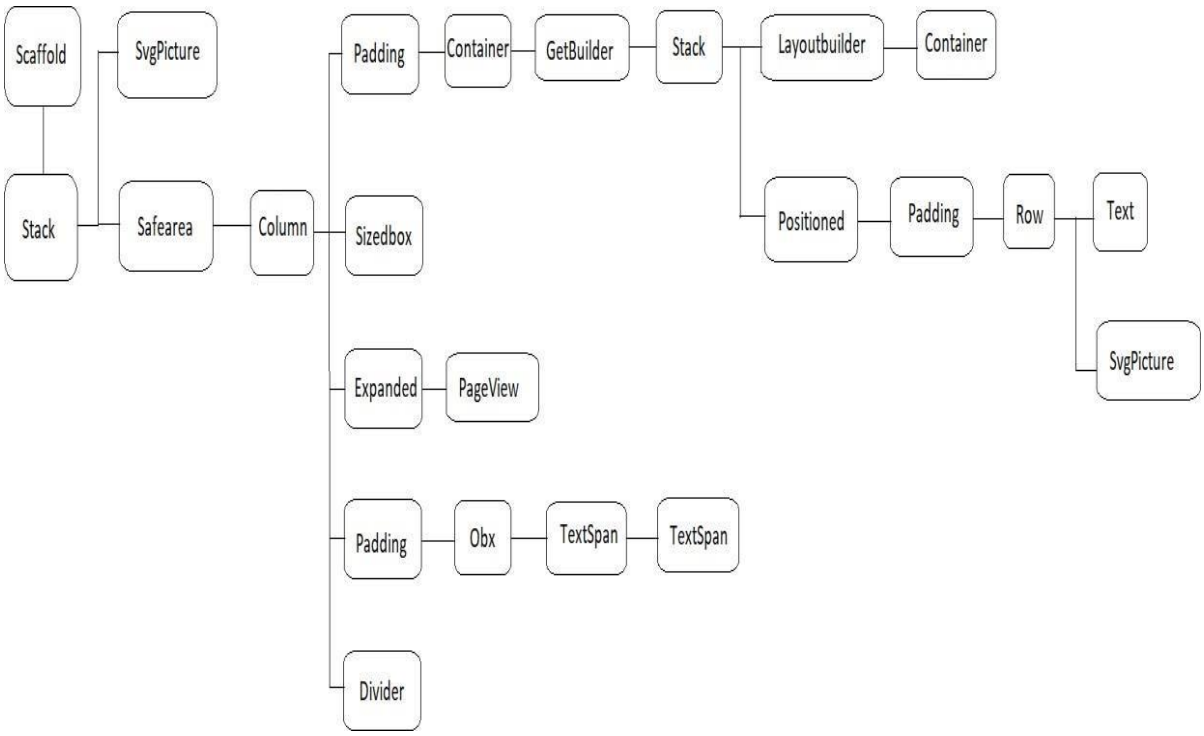


Figure 4.1:

4.2 SCORE PAGE WIDGET PAGE

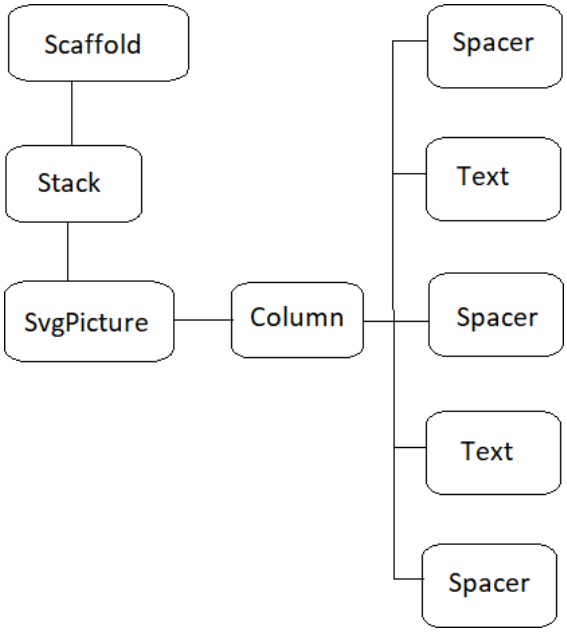
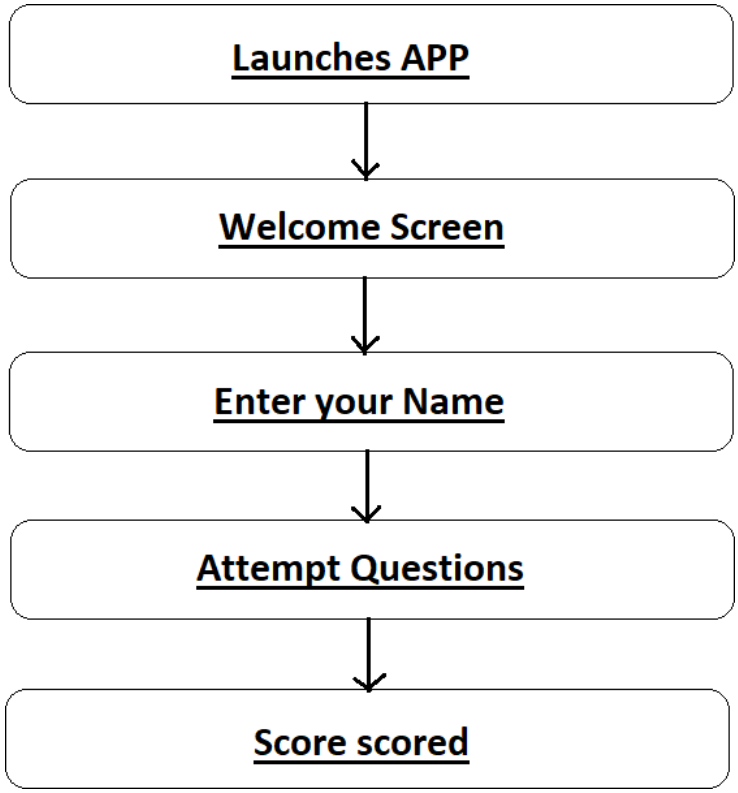


Figure 4.2:

4.3 SCHEMA DESIGN



5.IMPLEMENTATION

5.1 main.dart

```dart
import 'package:flutter/material.dart';
import 'package:get/get_navigation/src/root/get_material_app.dart';
import 'package:quiz_app/screens/welcome/welcome_screen.dart';

void main() {
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  // This widget is the root of your application.
  @override
  Widget build(BuildContext context) {
    return GetMaterialApp(
      title: 'Quiz App',
      debugShowCheckedModeBanner: false,
      theme: ThemeData.dark(),
      home: WelcomeScreen(),
    );
  }
}
```

5.2 Welcome.dart

```dart
import 'package:flutter/material.dart';
import 'package:flutter_svg/svg.dart';
import 'package:get/get.dart';
import 'package:quiz_app/constants.dart';
import 'package:quiz_app/screens/quiz/quiz_screen.dart';

class WelcomeScreen extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: Stack(
        children: [
          SvgPicture.asset("assets/icons/bg.svg", fit: BoxFit.fill),
          SafeArea(
            child: Padding(
```

```
            padding: const EdgeInsets.symmetric(horizontal:
kDefaultPadding),
            child: Column(
              crossAxisAlignment: CrossAxisAlignment.start,
              children: [
                Spacer(flex: 2), //2/6
                Text(
                  "Let's Play Quiz,",
                  style: Theme.of(context).textTheme.headline4.copyWith(
                      color: Colors.white, fontWeight: FontWeight.bold),
                ),
                Text("Enter your informations below"),
                Spacer(), // 1/6
                TextField(
                  decoration: InputDecoration(
                    filled: true,
                    fillColor: Color(0xFF1C2341),
                    hintText: "Full Name",
                    border: OutlineInputBorder(
                      borderRadius:
BorderRadius.all(Radius.circular(12)),
                    ),
                  ),
                ),
                Spacer(), // 1/6
                InkWell(
                  onTap: () => Get.to(QuizScreen()),
                  child: Container(
                    width: double.infinity,
                    alignment: Alignment.center,
                    padding: EdgeInsets.all(kDefaultPadding * 0.75), //
15
                    decoration: BoxDecoration(
                      gradient: kPrimaryGradient,
                      borderRadius:
BorderRadius.all(Radius.circular(12)),
                    ),
                    child: Text(
                      "Lets Start Quiz",
                      style: Theme.of(context)
                          .textTheme
                          .button
                          .copyWith(color: Colors.black),
                    ),
                  ),
                ),
                Spacer(flex: 2), // it will take 2/6 spaces
              ],
```

```
            ),
          ),
        ),
      ],
    ),
  );
 }
}
```

5.3 Quiz_screeen.dart

```
import 'package:flutter/material.dart';
import 'package:get/get.dart';
import 'package:quiz_app/controllers/question_controller.dart';

import 'components/body.dart';

class QuizScreen extends StatelessWidget {
 @override
 Widget build(BuildContext context) {
   QuestionController _controller = Get.put(QuestionController());
   return Scaffold(
     extendBodyBehindAppBar: true,
     appBar: AppBar(
       // Futtter show the back button automatically
       backgroundColor: Colors.transparent,
       elevation: 0,
       actions: [
         FlatButton(onPressed: _controller.nextQuestion, child:
Text("Skip")),
       ],
     ),
     body: Body(),
   );
 }
}
```

5.4 body.dart

```
import 'package:flutter/material.dart';
import 'package:get/get.dart';
import 'package:quiz_app/constants.dart';
import 'package:quiz_app/controllers/question_controller.dart';
import 'package:quiz_app/models/Questions.dart';
import 'package:flutter_svg/svg.dart';

import 'progress_bar.dart';
import 'question_card.dart';

class Body extends StatelessWidget {
  const Body({
    Key key,
  }) : super(key: key);

  @override
  Widget build(BuildContext context) {
    // So that we have acccess our controller
    QuestionController _questionController =
Get.put(QuestionController());
    return Stack(
      children: [
        SvgPicture.asset("assets/icons/bg.svg", fit: BoxFit.fill),
        SafeArea(
          child: Column(
            crossAxisAlignment: CrossAxisAlignment.start,
            children: [
              Padding(
                padding:
                    const EdgeInsets.symmetric(horizontal:
kDefaultPadding),
                child: ProgressBar(),
              ),
              SizedBox(height: kDefaultPadding),
              Padding(
                padding:
                    const EdgeInsets.symmetric(horizontal:
kDefaultPadding),
                child: Obx(
                  () => Text.rich(
                    TextSpan(
                      text:
                          "Question
${_questionController.questionNumber.value}",
                      style: Theme.of(context)
```

```
                            .textTheme
                            .headline4
                            .copyWith(color: kSecondaryColor),
                        children: [
                          TextSpan(
                            text:
"/${_questionController.questions.length}",
                            style: Theme.of(context)
                                .textTheme
                                .headline5
                                .copyWith(color: kSecondaryColor),
                          ),
                        ],
                      ),
                    ),
                  ),
                ),
                Divider(thickness: 1.5),
                SizedBox(height: kDefaultPadding),
                Expanded(
                  child: PageView.builder(
                    // Block swipe to next qn
                    physics: NeverScrollableScrollPhysics(),
                    controller: _questionController.pageController,
                    onPageChanged: _questionController.updateTheQnNum,
                    itemCount: _questionController.questions.length,
                    itemBuilder: (context, index) => QuestionCard(
                        question: _questionController.questions[index]),
                  ),
                ),
              ],
            ),
          )
        ],
      );
  }
}
```

5.5 option.dart

```
import 'package:flutter/material.dart';
import 'package:get/get_state_manager/get_state_manager.dart';
import 'package:quiz_app/controllers/question_controller.dart';

import '../../../constants.dart';
```

```
class Option extends StatelessWidget {
  const Option({
    Key key,
    this.text,
    this.index,
    this.press,
  }) : super(key: key);
  final String text;
  final int index;
  final VoidCallback press;

  @override
  Widget build(BuildContext context) {
    return GetBuilder<QuestionController>(
        init: QuestionController(),
        builder: (qnController) {
          Color getTheRightColor() {
            if (qnController.isAnswered) {
              if (index == qnController.correctAns) {
                return kGreenColor;
              } else if (index == qnController.selectedAns &&
                  qnController.selectedAns != qnController.correctAns) {
                return kRedColor;
              }
            }
            return kGrayColor;
          }

           IconData getTheRightIcon() {
             return getTheRightColor() == kRedColor ? Icons.close :
Icons.done;
          }

          return InkWell(
            onTap: press,
            child: Container(
              margin: EdgeInsets.only(top: kDefaultPadding),
              padding: EdgeInsets.all(kDefaultPadding),
              decoration: BoxDecoration(
                border: Border.all(color: getTheRightColor()),
                borderRadius: BorderRadius.circular(15),
              ),
              child: Row(
                mainAxisAlignment: MainAxisAlignment.spaceBetween,
                children: [
                  Text(
                    "${index + 1}. $text",
```

```
                    style: TextStyle(color: getTheRightColor(), fontSize:
16),
                  ),
                  Container(
                    height: 26,
                    width: 26,
                    decoration: BoxDecoration(
                      color: getTheRightColor() == kGrayColor
                          ? Colors.transparent
                          : getTheRightColor(),
                      borderRadius: BorderRadius.circular(50),
                      border: Border.all(color: getTheRightColor()),
                    ),
                    child: getTheRightColor() == kGrayColor
                        ? null
                        : Icon(getTheRightIcon(), size: 16),
                  )
                ],
              ),
            ),
          );
        });
  }
}
```

5.6 progress_bar.dart

```
import 'package:flutter/material.dart';
import 'package:get/get_state_manager/get_state_manager.dart';
import 'package:quiz_app/controllers/question_controller.dart';
import 'package:flutter_svg/svg.dart';

import '../../../constants.dart';

class ProgressBar extends StatelessWidget {
  const ProgressBar({
    Key key,
  }) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return Container(
      width: double.infinity,
```

```
      height: 35,
      decoration: BoxDecoration(
        border: Border.all(color: Color(0xFF3F4768), width: 3),
        borderRadius: BorderRadius.circular(50),
      ),
      child:GetBuilder<QuestionController>(
        init: QuestionController(),
        builder: (controller) {
          return Stack(
            children: [
              // LayoutBuilder provide us the available space for the
conatiner
              // constraints.maxWidth needed for our animation
              LayoutBuilder(
                builder: (context, constraints) => Container(
                  // from 0 to 1 it takes 60s
                  width: constraints.maxWidth *
controller.animation.value,
                  decoration: BoxDecoration(
                    gradient: kPrimaryGradient,
                    borderRadius: BorderRadius.circular(50),
                  ),
                ),
              ),
              Positioned.fill(
                child: Padding(
                  padding: const EdgeInsets.symmetric(
                      horizontal: kDefaultPadding / 2),
                  child: Row(
                    mainAxisAlignment: MainAxisAlignment.spaceBetween,
                    children: [
                      Text("${(controller.animation.value * 60).round()}
sec"),
                      SvgPicture.asset("assets/icons/clock.svg"),
                    ],
                  ),
                ),
              ),
            ],
          );
        },
      ),
    );
  }
}
```

5.7 question_card.dart

```dart
import 'package:flutter/material.dart';
import 'package:get/get.dart';
import 'package:quiz_app/controllers/question_controller.dart';
import 'package:quiz_app/models/Questions.dart';

import '../../../constants.dart';
import 'option.dart';

class QuestionCard extends StatelessWidget {
  const QuestionCard({
    Key key,
    // it means we have to pass this
    @required this.question,
  }) : super(key: key);

  final Question question;

  @override
  Widget build(BuildContext context) {
    QuestionController _controller = Get.put(QuestionController());
    return Container(
      margin: EdgeInsets.symmetric(horizontal: kDefaultPadding),
      padding: EdgeInsets.all(kDefaultPadding),
      decoration: BoxDecoration(
        color: Colors.white,
        borderRadius: BorderRadius.circular(25),
      ),
      child: Column(
        children: [
          Text(
            question.question,
            style: Theme.of(context)
                .textTheme
                .headline6
                .copyWith(color: kBlackColor),
          ),
          SizedBox(height: kDefaultPadding / 2),
          ...List.generate(
            question.options.length,
            (index) => Option(
              index: index,
              text: question.options[index],
              press: () => _controller.checkAns(question, index),
            ),
```

```
          ),
        ],
      ),
    );
  }
}
```

5.8 score_screeen.dart

```
import 'package:flutter/material.dart';
import 'package:get/get.dart';
import 'package:quiz_app/constants.dart';
import 'package:quiz_app/controllers/question_controller.dart';
import 'package:flutter_svg/svg.dart';

class ScoreScreen extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    QuestionController _qnController = Get.put(QuestionController());
    return Scaffold(
      body: Stack(
        fit: StackFit.expand,
        children: [
          SvgPicture.asset("assets/icons/bg.svg", fit: BoxFit.fill),
          Column(
            children: [
              Spacer(flex: 3),
              Text(
                "Score",
                style: Theme.of(context)
                    .textTheme
                    .headline3
                    .copyWith(color: kSecondaryColor),
              ),
              Spacer(),
              Text(
                "${_qnController.correctAns *
10}/${_qnController.questions.length * 10}",
                style: Theme.of(context)
                    .textTheme
                    .headline4
                    .copyWith(color: kSecondaryColor),
              ),
              Spacer(flex: 3),
            ],
```

```
          )
        ],
      ),
    );
  }
}
```

6.TESTING

6.1 Introduction

Testing is a process of executing a program with the interest of finding an error. A good test is one that has high probability of finding the yet undiscovered error. Testing should systematically uncover different classes of errors in a minimum amount of time with a minimum number of efforts. Two classes of inputs are provided to test the process

A software configuration that includes a software requirement specification, a design specification and source code.

A software configuration that includes a test plan and procedure, any testing tool and test cases and their expected results.

6.2 Levels of Testing

6.2.1 Unit Testing

Unit testing is a level of software testing where individual units/ components of a software are tested. The purpose is to validate that each unit of the software performs as designed. A unit is the smallest testable part of any software. It usually has one or a few inputs and usually a single output. Unit testing is commonly automated, but may still be performed manually. The objective in unit testing is to isolate a unit and validate its correctness. A manual approachto unit testing may employ a step-by-step instructional document. The unit testing is the process of testing the part of the program to verify whether the program is working corrector not. In this part the main intention is to check the each and every input which we are inserting to our file. Here the validation concepts are used to check whether the program istaking the inputs in the correct format ornot.

Unit testing may reduce uncertainty in the units themselves and can be used in a bottom-up testing style approach. By testing the parts of a program first and then testing the sum of its parts, integration testing becomes much easier. Unit test cases embody characteristics that are critical to the success of the unit.

6.2.2 Integration Testing

Integration testing is also taken as integration and testing this is the major testing process where the units are combined and tested. Its main objective is to verify whether the major parts of the program is working fine or not. This testing can be done by choosing the options in the program and by giving suitableinputs.

6.2.3 System Testing

System testing is defined as testing of a complete and fully integrated software product. This testing falls in black-box testing wherein knowledge of the inner design of the code isnot a pre-requisite and is done by the testing team. System testing is done after integrationtesting is complete. System testing should test functional and non-functional requirementsof thesoftware.

6.2.4 Validation Testing

In this, requirements established as part of software requirements analysis are validated against the software that has been constructed. Validation testing provides final assurance that software meets all functional, behavioral and performance requirements. Validation can be defined in many ways but a simple definition is that validation succeeds when software Function in a manner that can be reasonably by the customer.

1. Validation test criteria
2. Configuration review
3. Alpha and Beta testing (conducted by end user)

6.1.1 Output Testing

After preparing test data, the system under study is tested using the test data. While testing the system using test data, errors are again uncovered and corrected by using above testing and corrections are also noted for futureuse.

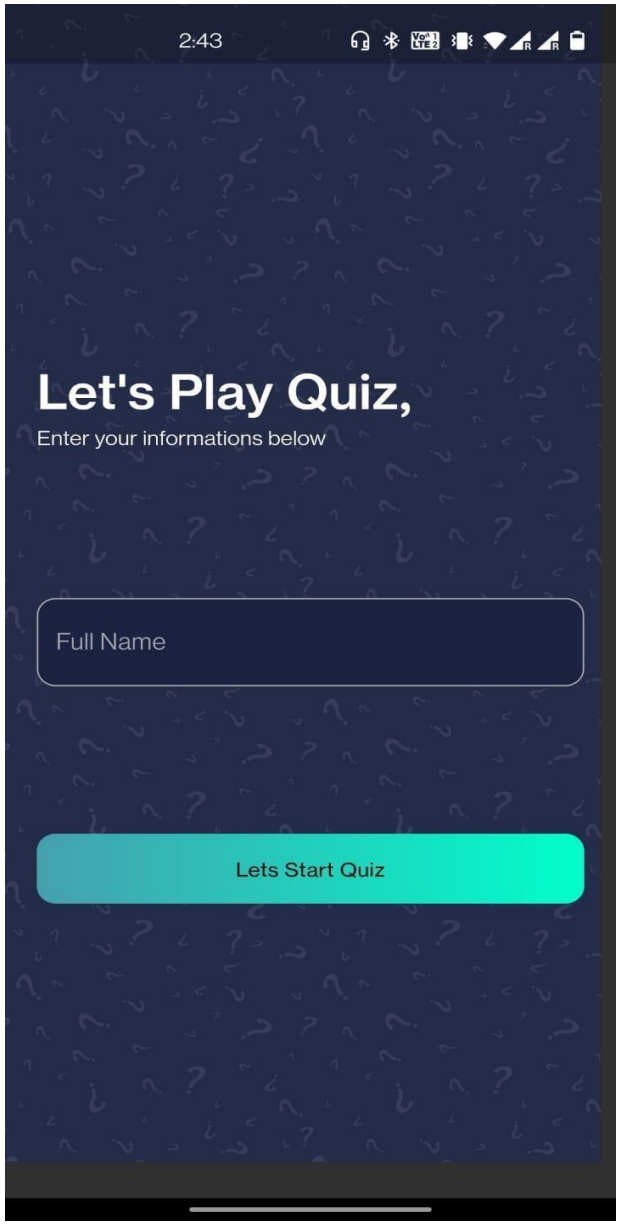
6.1.1 User Acceptance Testing

User acceptance testing is a type of testing performed by the end user or the client to verify/accept the software application to the production environment.

User Acceptance Testing is done in the final phase of testing.

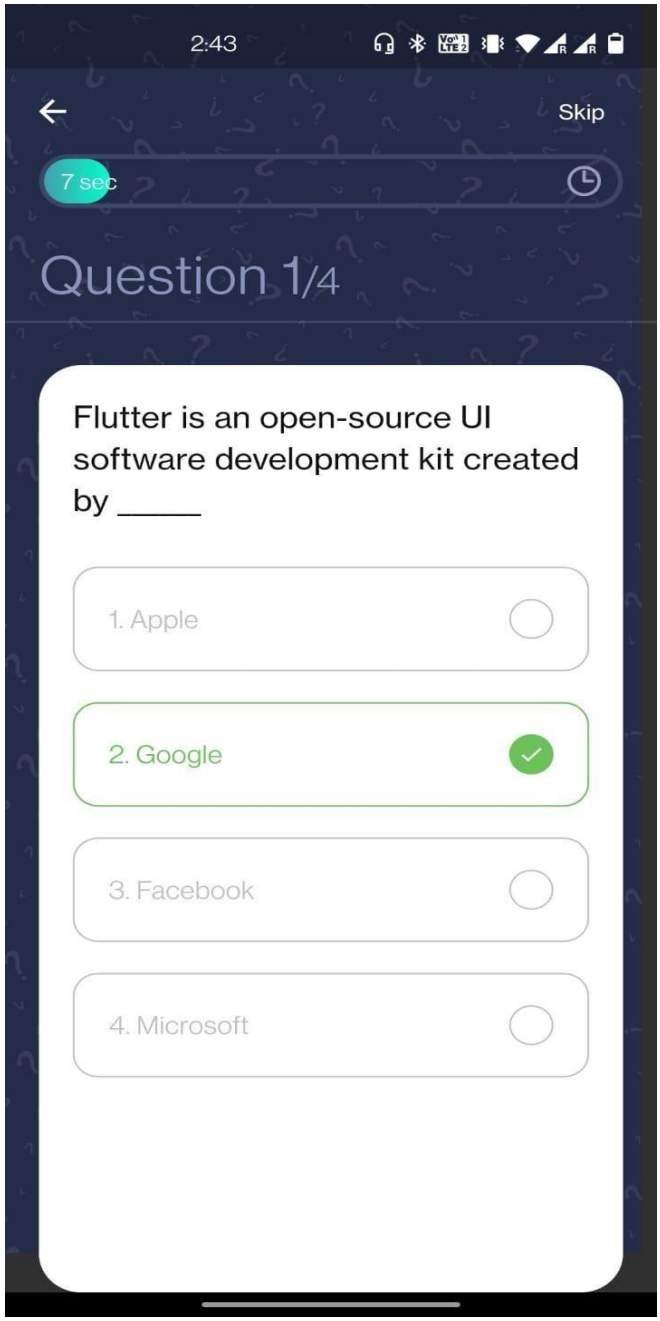
7.DISCUSSION OF RESULTS

7.1 Home page



This is the landing page of the application where we can enter the name of the user or the player and there is a button that will take us to the quiz questions.

7.2 Quiz Questions



This is the quiz questions section page. There will be number of different questions as per the requirement. User will be given with the question statement and the options. User will have to select an option. If the selected option will be true then a green tick will be shown otherwise a red cross will be shown.

7.2 Score Page

This is the Score section page. In this score will be awarded to the user as per their performance. Each question will fetch 10 marks to the user and wrong answer won't affect the marks.

8. CONCLUSION AND FUTURE ENHANCEMENT

8.1 Conclusion

This quiz application provides facility to play quiz anywhere and anytime. It save time since user does need to wait for result. So student/user cannot wait for the result. All Student/ user get extra knowledge and skills. Administrator has a privilege to put as much as question in any category given in application. User can register, put their name and give the test with his/her specific id, and can see the results as well.

8.2 Future Enhancement

At the same time there is some scope for improvement in the future. It can be possible to make it more users friendly by adding more variety of functions to it. We can try and deploy it on the internet so that people can access it online also. We can try to add different sections of the quiz as per user requirement. We can add some admin page where an admin can add questions. We can also let different user can access the app simultaneously with their unique id.

9.REFERNCES

- Microsoft Visual C# Step by Step (Developer Reference Edition)
- RamezElmasri and Shamkant B. Navathe, Fundamentals of Database Systems, Pearson, 7th Edition.
- Learn SQL Tutorial - javatpoint
- MS SQL Server Tutorial (tutorialspoint.com)
- C# Tutorial - GeeksforGeeks
- What is .NET Framework? Explain Architecture & Components (guru99.com)