

PROFESSIONAL TRAINING REPORT
at
Sathyabama Institute of Science and Technology
(Deemed to be University)

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Engineering Degree in Computer Science and Engineering

By
MUGUNTH.D
REG NO: 40733020



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING
SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600119, TAMILNADU

NOVEMBER 2022



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by NAAC

(Established under Section 3 of UGC Act, 1956)

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI- 600119

www.sathyabamauniversity.ac.in



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **MUGUNTH.D(40733020)** who carried out the project entitled "**QUIZ MASTER APP USING FLUTTER**" under my supervision from Aug 2022 to Oct 2022.

Internal Guide Name

Ms.Nancy Kirupanithi.D

Head of the Department

Dr.S.Vigneshwari

Submitted for Viva voce Examination held on 17-04-2023

Internal Examiner

External Examiner

DECLARATION

I, **MUGUNTH.D** hereby declare that the Project Report entitled **QUIZ MASTER APP USING FLUTTER**

done by me under the guidance of Dr./Prof./Mr./Ms. _____ (Internal) and _____ (External) at **ALGOXFUSION** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE: 17-04-2023

PLACE: Chennai



SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala M.E., Ph.D., Dean**, School of Computing , **Dr.S.Vigneshwari M.E., Ph.D., and Dr.L.Lakshmanan M.E., Ph.D.**, Heads of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide ----
-----for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

TRAINING CERTIFICATE



ABSTRACT

The project: "Quiz Application" is a collection of different types of quizzes based on fundamentals of computers . Basically it is a fundamentals of Computer science questions all interested peoples can play this .The main goal of the application is to enable users to practice for subjective tests conducted for admissions and recruitment, with focus on Computer science field. A user can access /play all of the quiz 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.

TABLE OF CONTENTS

CHAPTER No.	TITLE	PAGE No
	ABSTRACT	I
	LIST OF ABBREVIATIONS	iv
	LIST OF FIGURES	1
1	INTRODUCTION	2
	1.1 Features of flutter	3
	1.1.1 Advantages of flutter	3
	1.1.2 Disadvantage of flutter	3
	1.2 Dart programming	3
	1.2.1 Advantages of dart	4
	1.2.2 Disadvantages of dart	5
	1.3 Introduction to widgets	6
	1.3.1 Gestures	7
	1.3.2 Widget bulid visualization	7
	1.4 Installation in windows	9
	1.5 Installation in macos	12
2	AIM AND SCOPE OF THE PRESENT INVESTIGATION	14
	2.1 AIM	14
	2.2 OBJECTIVE	14
	2.2 SCOPE	14

3	METHODOLOGY AND ALGORITHMS USED	15
3.1	System requirements	15
3.1.1	windows	15
3.1.2	Mac	15
3.1.3	Linux	15
3.1.4	Chrome os	16
3.2	METHODS AND ALGORITHMS USED	16
3.3	Flow chart	17
3.4	Components and widgets used	18
3.4.1	Components	18
3.4.2	Widgets	18
3.4.2.1	main.dartt	19
4	RESULTS AND DISCUSSION, PERFORMANCE ANALYSIS	21
4.1	Results	21
4.2	Performance analysis	23

5	SUMMARY AND CONCLUSION	24
	5.1 Summary	24
	5.2 Conclusion	24
	APPENDIX	26
	REFERENCES	33

LIST OF FIGURES

FIGURE No	FIGURE NAME	PAGE No
1.1	Environment variable	10
1.2	Edit environment variable	11
1.3	Command prompt	11
1.4	Export path	12
1.5	source	12
3.1	Flow chart	17
4.1	Home screen	21
4.2	correct answer screen	21
4.3	Wrong Answer Screen	22
4.4	Result Screen	22

LIST OF ABBREVIATIONS

ABBREVIATION	EXPANSION
HTML	Hyper Text Markup language
UI	Userface Interface
OS	Operating System
IDE	Integrated Development Environment
AOT	Ahead Of Time
JIT	Just In Time
RAM	Random Access Memory

CHAPTER – 1

INTRODUCTION

In general, developing a mobile application is a complex and challenging task. There are many frameworks available to develop a mobile application. Android provides a native framework based on Java language and iOS provides a native framework based on Objective-C / Swift language.

However, to develop an application supporting both the OSs, we need to code in two different languages using two different frameworks. To help overcome this complexity, there exists mobile frameworks supporting both OS. These frameworks range from simple HTML based hybrid mobile application framework (which uses HTML for User Interface and JavaScript for application logic) to complex language specific framework (which do the heavy lifting of converting code to native code). Irrespective of their simplicity or complexity, these frameworks always have many disadvantages, one of the main drawback being their slow performance.

In this scenario, Flutter – a simple and high performance framework based on Dart language, provides high performance by rendering the UI directly in the operating system's canvas rather than through native framework.

Flutter also offers many ready to use widgets (UI) to create a modern application. These widgets are optimized for mobile environment and designing the application using widgets is as simple as designing HTML.

To be specific, Flutter application is itself a widget. Flutter widgets also supports animations and gestures. The application logic is based on reactive programming. Widget may optionally have a state. By changing the state of the widget, Flutter will automatically (reactive programming) compare the widget's state (old and new) and render the widget with only the necessary changes instead of re-rendering the whole widget.

1.1 FEATURES OF FLUTTER

Flutter framework offers the following features to developers –

- Modern and reactive framework.
- Uses Dart programming language and it is very easy to learn.
- Fast development.
- Beautiful and fluid user interfaces.
- Huge widget catalog.
- Runs same UI for multiple platforms.
- High performance application

1.1.1 Advantages of flutter

Flutter comes with beautiful and customizable widgets for high performance and outstanding mobile application. It fulfils all the custom needs and requirements. Besides these, Flutter offers many more advantages as mentioned below –

- Dart has a large repository of software packages which lets you to extend the capabilities of your application.
- Developers need to write just a single code base for both applications (both Android and iOS platforms). Flutter may to be extended to other platform as well in the future.
- Flutter needs lesser testing. Because of its single code base, it is sufficient if we write automated tests once for both the platforms.
- Flutter's simplicity makes it a good candidate for fast development. Its customization capability and extendibility makes it even more powerful.
- With Flutter, developers has full control over the widgets and its layout.
- Flutter offers great developer tools, with amazing hot reload.

1.1.2 Disadvantages of flutter

Despite its many advantages, flutter has the following drawbacks in it –

- Since it is coded in Dart language, a developer needs to learn new language (though it is easy to learn).
- Modern framework tries to separate logic and UI as much as possible but, in Flutter, user interface and logic is intermixed. We can overcome this using smart coding and using high level module to separate user interface and logic.
- Flutter is yet another framework to create mobile application. Developers are having a hard time in choosing the right development tools in hugely populated segment.

1.2 DART PROGRAMMING

Dart is an open-source general-purpose programming language. It is originally developed by Google. Dart is an object-oriented language with C-style syntax. It supports programming concepts like interfaces, classes, unlike other programming languages Dart doesn't support arrays. Dart collections can be used to replicate data structures such as arrays, generics, and optional typing.

Dart language supports the following data types –

- Numbers – It is used to represent numeric literals – Integer and Double.
- Strings – It represents a sequence of characters. String values are specified in either single or double quotes.
- Booleans – Dart uses the bool keyword to represent Boolean values – true and false.
- Lists and Maps – It is used to represent a collection of objects.

1.2.1 Advantages of dart

- The first advantage is that it is easy to learn. Any JavaScript programmer can quickly relearn how to write code in Dart. To do this, they only need to familiarize themselves with the basic principles of this programming language.

- The second thing is about its availability of documentation. Since Google is developing the interpreter for Dart, all the features of the language are described in detail. This allows you to quickly get answers to almost any questions that may arise during the training process, or directly while writing code.
- The third advantage is its high performance. Programs written in Dart tend to run faster than programs created in JavaScript.
- Dart is very stable and it can be used to build production quality real-time applications. It is an object-oriented programming language with support for inheritance, interfaces and optional typing features.
- It uses AOT and JIT compilation – Dart has the unique capability to handle both Ahead of time and Just in time compiling. In AOT, the Dart code can be directly converted into native machine code. While in the mode of JIT, it can be compiled for exceptionally fast development cycles and game-changing workflow.
- If you want to start writing your first Dart program without any installation or configuration, there is DartPad for you. The DartPad interface is plain and simple. Write your code and click ‘Run’ to execute the code. DartPad introduces users to the world of Dart programming. The support for libraries in DartPad is restricted to the basic level.

1.2.2 Disadvantages of dart

- The first drawback is that Dart has very limited resources online and it’s hard to find solutions to problems. The reason for it is lack of a larger and cohesive developer communities, which will support you.
- Dart is fairly new to the programmers and rarely used in the market. So if you will look for a job on this position, it will be harder to find a company with a vacancy of Dart programmer, because it hasn’t yet spread widely.
- Dart is currently in development. That’s exciting, and it’s not something “wrong” with Dart, but if you start developing in Dart now, there’s a certain

chance that the API will change, or that things won't be documented fully or correctly, and the amount of information on the web is less than, for example, the amount you can find about jQuery.

- The absence of native support. For a beautiful demonstration of Dart's capabilities, developers must bring it to users' browsers. It turns out that to achieve at least 80% success, Dart developers must provide native support for it in all popular browsers.
- To sum up, Dart is an interesting programming language with features to facilitate Web, mobile and command line apps. Its major advantages are its stability and ease of learning. In the case of Web applications, the combination with AngularDart makes it a very powerful tool. Generally, we think that despite its novelty, this programming language is worth your attention.

1.3 INTRODUCTION TO WIDGETS

The core concept of the Flutter framework is In Flutter, Everything is a widget. Widgets are basically user interface components used to create the user interface of the application.

In Flutter, the application is itself a widget. The application is the top-level widget and its UI is build using one or more children (widgets), which again build using its children widgets. This composability feature helps us to create a user interface of any complexity

Here the following points are worth notable –

- MyApp is the user created widget and it is build using the Flutter native widget, MaterialApp.
- MaterialApp has a home property to specify the user interface of the home page, which is again a user created widget, MyHomePage.
- MyHomePage is build using another flutter native widget, Scaffold
- Scaffold has two properties – body and appBar

- body is used to specify its main user interface and appBar is used to specify its header user interface
- Header UI is build using flutter native widget, AppBar and Body UI is build using Center widget.
- The Center widget has a property, Child, which refers the actual content and it is build using Text widget.

1.3.1 Gestures

Flutter widgets support interaction through a special widget, GestureDetector. GestureDetector is an invisible widget having the ability to capture user interactions such as tapping, dragging, etc., of its child widget. Many native widgets of Flutter support interaction through the use of GestureDetector. We can also incorporate interactive feature into the existing widget by composing it with the GestureDetector widget. We will learn the gestures separately in the upcoming chapters.

1.3.2 Widget build visualization

In Flutter, widgets can be grouped into multiple categories based on their features, as listed below –

- Platform specific widgets
- Layout widgets
- State maintenance widgets
- Platform independent / basic widgets

Some of the most used material widgets are as follows –

- Scaffold
- AppBar
- BottomNavigationBar
- TabBar
- TabBarView
- ListTile

- RaisedButton
- FloatingActionButton
- FlatButton
- IconButton
- DropdownButton
- PopupMenuButton
- ButtonBar
- TextField
- Checkbox
- Radio
- Switch
- Slider
- Date & Time Pickers
- SimpleDialog
- AlertDialog

Some of the most used Cupertino widgets are as follows

- CupertinoButton
- CupertinoPicker
- CupertinoDatePicker
- CupertinoTimerPicker
- CupertinoNavigationBar
- CupertinoTabBar
- CupertinoTabScaffold
- CupertinoTabView
- CupertinoTextField
- CupertinoDialog
- CupertinoDialogAction
- CupertinoFullscreenDialogTransition
- CupertinoPageScaffold

- CupertinoPageTransition
- CupertinoActionSheet
- CupertinoActivityIndicator
- CupertinoAlertDialog
- CupertinoPopupSurface
- CupertinoSlider

The most important properties of the Text widget are as follows

- maxLines, int – Maximum number of lines to show
- overflow, TextOverflow – Specify how visual overflow is handled using TextOverflow class
- style, TextStyle – Specify the style of the string using TextStyle class
- textAlign, TextAlign – Alignment of the text like right, left, justify, etc., using TextAlign class
- textDirection, TextDirection – Direction of text to flow, either left-to-right or right-to-left

1.4 INSTALLATION IN WINDOWS

In this section, let us see how to install Flutter SDK and its requirement in a windows system.

Step 1 – Go to URL, <https://flutter.dev/docs/get-started/install/windows> and download the latest Flutter SDK.

Step 2 – Unzip the zip archive in a folder, like C:\flutter\

Step 3 – Go to URL, <https://developer.android.com/studio#downloads> and download the latest version.

Step 4 – Make sure your PC have jdk.

Step 5 – Update the system path to include flutter bin directory

- From the Start search bar, enter 'env' and select Edit environment variables for your account.
- Under User variables check if there is an entry called Path:

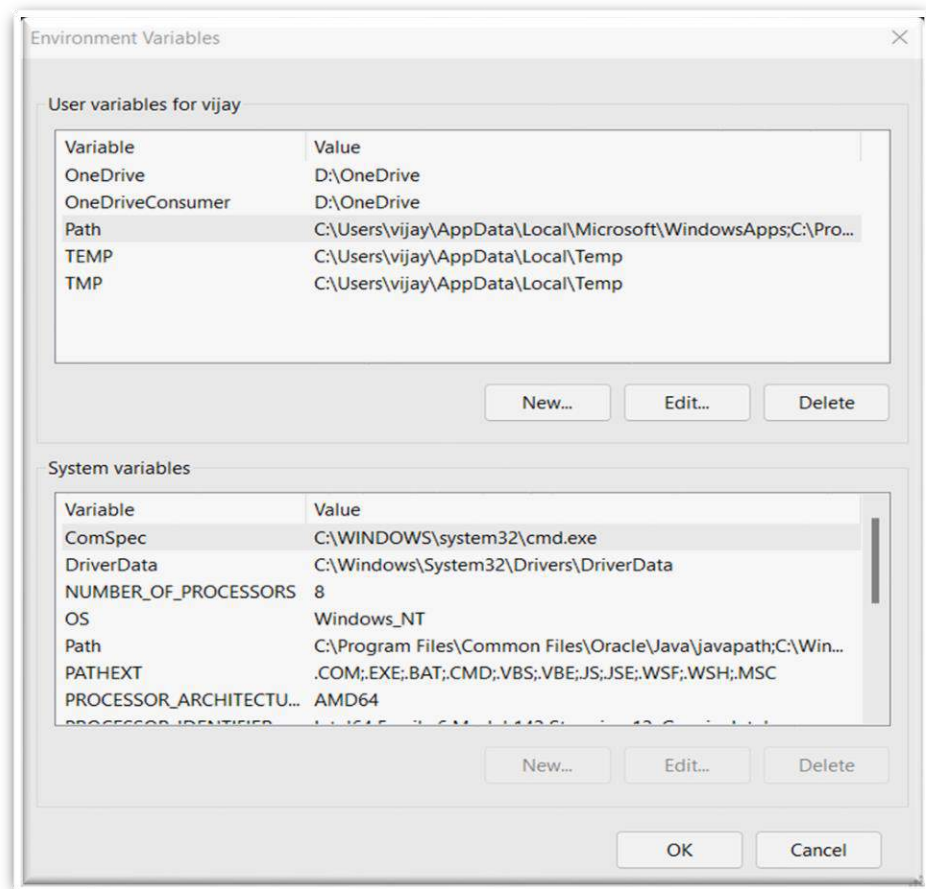


Fig 1.1 Environment variable

Then edit the path:

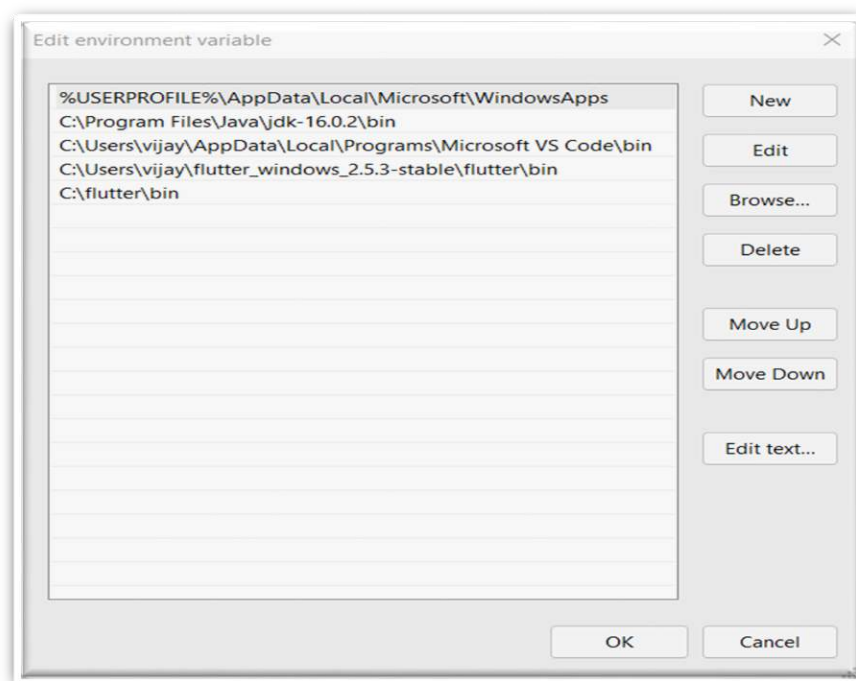
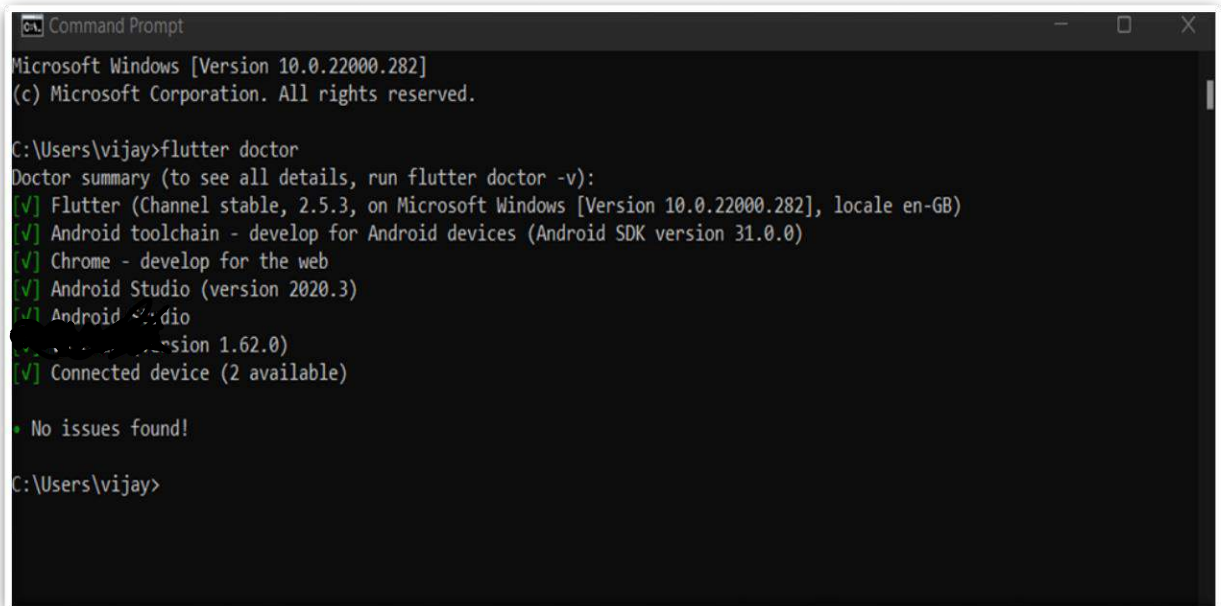


Fig 1.2 Edit environment variable

Step 4 – Flutter provides a tool, “flutter doctor” to check that all the requirement of flutter development is met, running the above command will analyze the system and show its report as shown below



```
Microsoft Windows [Version 10.0.22000.282]
(c) Microsoft Corporation. All rights reserved.

C:\Users\vijay>flutter doctor
Doctor summary (to see all details, run flutter doctor -v):
[✓] Flutter (Channel stable, 2.5.3, on Microsoft Windows [Version 10.0.22000.282], locale en-GB)
[✓] Android toolchain - develop for Android devices (Android SDK version 31.0.0)
[✓] Chrome - develop for the web
[✓] Android Studio (version 2020.3)
[✓] Android Studio
[✓] Android Studio
[✓] Android Studio (version 1.62.0)
[✓] Connected device (2 available)

• No issues found!

C:\Users\vijay>
```

Fig 1.3 Command prompt

Step 5 – Install the latest Android SDK, if reported by flutter doctor

Step 6 – Install the latest Android Studio, if reported by flutter doctor

Step 7 – Start an android emulator or connect a real android device to the system.

Step 8 – Install Flutter and Dart plugin for Visual Studio or Android Studio. It provides startup template to create new Flutter application, an option to run and debug Flutter application, etc.,

- Open Visual Studio / Android Studio.
- Click File → Settings → Plugins.
- Select the Flutter plugin and click Install.
- Click Yes when prompted to install the Dart plugin.
- Restart Visual Studio

1.5 INSTALLATION IN MACOS

To install Flutter on MacOS, you will have to follow the following steps –

Step 1 – Go to URL, <https://flutter.dev/docs/get-started/install/macos> and download latest Flutter SDK. As of April 2019, the version is 1.2.1 and the file is flutter_macos_v1.2.1- stable.zip.

Step 2 – Unzip the zip archive in a folder, say /path/to/flutter

Step 3 – Update the system path to include flutter bin directory (in ~/.bashrc file).

```
> export PATH = "$PATH:/path/to/flutter/bin"
```

Fig 1.4 Export path

Step 4 – Enable the updated path in the current session using below command and then verify it as well.

```
source ~/.bashrc  
source $HOME/.bash_profile  
echo $PATH
```

Fig 1.5 Source

Flutter provides a tool, flutter doctor to check that all the requirement of flutter development is met. It is similar to the Windows counterpart.

Step 5 – Install latest XCode, if reported by flutter doctor

Step 6 – Install latest Android SDK, if reported by flutter doctor

Step 7 – Install latest Android Studio, if reported by flutter doctor

Step 8 – Start an android emulator or connect a real android device to the system to develop android application.

Step 9 – Open iOS simulator or connect a real iPhone device to the system to develop iOS application.

Step 10 – Install Flutter and Dart plugin for Android Studio. It provides the startup template to create a new Flutter application, option to run and debug Flutter application in the Android studio itself, etc.,

- Open Android Studio
- Click Preferences → Plugins
- Select the Flutter plugin and click Install
- Click Yes when prompted to install the Dart plugin.
- Restart Android studio.
- Android studio

CHAPTER – 2

AIM AND SCOPE OF TODO LIST

2.1 AIM

To build a Quiz Master application using Flutter in Android Studio IDE for users in which a user can attempt any number of quizzes related to his/her choice.

2.2 OBJECTIVE

The main objective of “Quiz Application” is to facilitate a user friendly environment for all users and reduces the manual effort. In past days quiz is conducted manually but in further resolution of the technology we are able to generate the score and pose the queries automatically. The functional requirements include creating users that are going to participate in the quiz, automatic score and report generation and administrative tasks like add, delete, update for admin privilege users. In this application, all the permissions lies with the administrator i.e., specifying the details of the quiz with checking result will show to interviewee or not, addition of question and answers, marks for each question, Set timer for each quiz and generate report with score for each quiz.

2.3 SCOPE

The Scope of this project is very broad in terms of gaining knowledge and sharing knowledge among world. Few points are:-

- Can be used anywhere any time as it is a web based application.
- This application will be used in educational institutions as well as in corporate world.

CHAPTER – 3

METHODOLOGY AND ALGORITHMS USED

3.1. SYSTEM REQUIREMENTS

The cool thing about using Flutter for creating cross-platform native mobile apps is the fact that you can build those on almost any OS.

Here are some System Requirements for Android Studio which is needed for running an Android simulator.

3.1.1 Windows

- Microsoft® Windows® 7/8/10 64-bit
- 4 GB RAM minimum, 8 GB RAM recommended
- 2 GB of available disk space minimum,
- 4 GB Recommended 500 MB for IDE + 1.5 GB for Android SDK and emulator system image
- 1280 x 800 minimum screen resolution

3.1.2 Mac

- Mac® OS X® 10.10 Yosemite or higher, up to 10.14 macOS Mojave
- 4 GB RAM minimum, 8 GB RAM recommended
- 2 GB of available disk space minimum,
- 4 GB Recommended 500 MB for IDE + 1.5 GB for Android SDK and emulator system image
- 1280 x 800 minimum screen resolution

3.1.3 Linux

- GNOME or KDE desktop - *Tested on gLinux based on Debian.*
- 64-bit distribution capable of running 32-bit applications
- GNU C Library glibc 2.19 or later
- 4 GB RAM minimum, 8 GB RAM recommended

- 2 GB of available disk space minimum,
- 4 GB Recommended 500 MB for IDE + 1.5 GB for Android SDK and emulator system image
- 1280 x 800 minimum screen resolution

3.1.4 Chrome OS

- 8 GB RAM or more recommended
- 4 GB of available disk space minimum
- 1280 x 800 minimum screen resolution
- Intel i5 or higher U series or higher recommended

3.2 METHODOLOGY AND ALGORITHMS USED

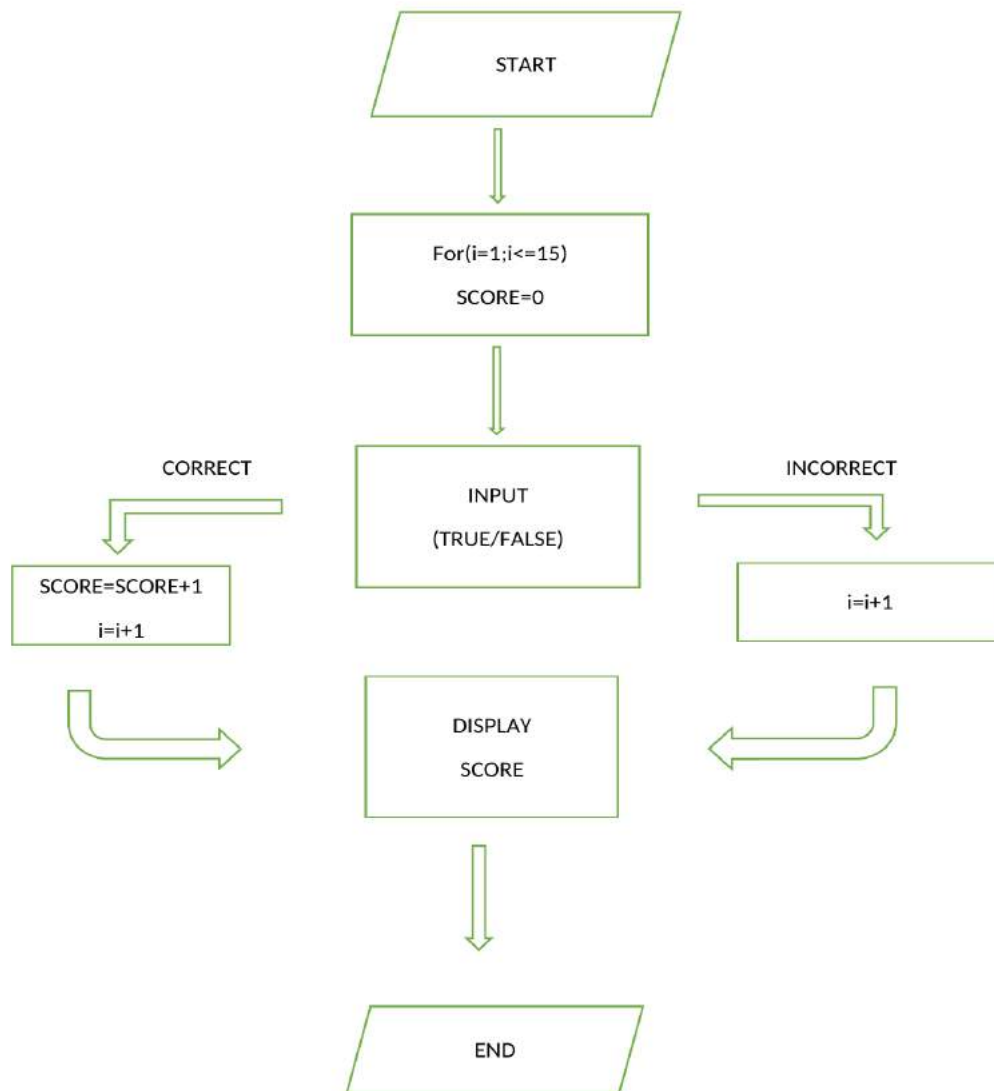
Using Flutter a free and open-source mobile UI framework created by google. Quiz Master application project is build using a Dart language.

3.2.1. Starting up

- Open and setup Visual Studio Code or Android Studio Code according to your convince and install the Flutter and Dart plugins/extensions in order to create a new flutter project
- A default basic flutter application will created and the necessary files and package are included by Android Studio as default.

3.2.2. Import all necessary package and create necessary pages, widgets and providers for our app.

3.3 FLOWCHART



3.4 COMPONENTS AND WIDGETS USED

3.4.1 Components

- android - Auto generated source code to create android application
- assets – contains image folder for the splash screen
- build – When you run a flutter project, it builds depending on which emulator or device it is running on, doing a Gradle or XCode build using the folders inside it. In short, those folders are entire apps which set the stage for the flutter code to run
- ios - Auto generated source code to create ios application
- lib - Main folder containing Dart code written using flutter framework
- lib/main.dart - Entry point of the Flutter application
- lib/pages – contains dart code for Splash Screen Home Page and Edit to-do Page
- lib/widgets – contains dart files
- lib/provider contains todo.dart file
- lib/api – contains dart file for firebase
- gitignore - Git version control file
- metadata - auto generated by the flutter tools
- packages - auto generated to track the flutter packages
- pubspec.yaml - Used by Pub, Flutter package manager
- pubspec.lock - Auto generated by the Flutter package manager, Pub
- README.md - Project description file written in Markdown format.

3.4.2 Widgets

Stateless and statefull widgets:

To make our app look more like a Quiz Master app, let's show some tasks. You may have noticed that our simple app above is a `StatelessWidget`. This means that it cannot be changed dynamically. For our Quiz Master app, this is no good, because items get added and removed all the time. However, a `StatelessWidget` can

have children which are dynamic, which are `StatefulWidgets`. Let's break out our stateful functionality the list container from the overall app container.

To make a stateful widget, we need two classes — one for the widget itself, and a second which creates the state. This setup allows the state to be easily saved, and enables the use of features such as hot reloading.

- Widgets are high level objects used to describe any part of an application. It can be but is not limited to UI elements such as buttons ; layout alignment, padding;data a theme, configurations
- Widgets can be everything. In flutter, almost everything you'll code will be inside a Widget. Even things like "redux reducers" are widgets in flutter.
- The core concept of the Flutter framework is In Flutter, Everything is a widget. Widgets
- are basically user interface components used to create the user interface of the application.
- In Flutter, the application is itself a widget. The application is the top- level widget and its UI is build using one or more children widgets, which again build using its children
- widgets. This composability feature helps us to create a user interface of any complexity.

3.4.2.1. main.dart

- `runApp()` Function takes the given Widget and makes it to the widget tree in main.drat the widget tree consists of a `MaterialApp` widget which describes theme ,primarySwatch and scaffoldBackgroundColor.
- `Scaffold()` – implements the baic material design visual layout structure. This class provides APIs for showing deawers and bottom sheets.
- `AppBar()` – `AppBar` and `Scaffold` widgets, and from `material.dart`, the app

is starting to look a bit more Material. For example, the app bar has a shadow and the title text inherits the correct styling automatically. AppBar widget let you pass in widgets for the leading widget, and the actions of the title widget.

- BottomNavigationBar class – A material widget that's displayed at the bottom of an app for selecting among a small number of views, typically between three and five.

The bottom navigation bar consists of multiple items in the form of text labels, icons, or both, laid out on top of a piece of material. It provides quick navigation may be a better fit.

A bottom navigation bar is usually used in conjunction with a scaffold, where it is provided as the Scaffold.bottomNavigationBar argument.

- FloatingActionButton class – A material design floating action button.

A floating action button is a circular icon button that hovers over content to promote a primary action in the application. Floating action buttons are most commonly used in the Scaffold.

onPressed callbacks to AddToDoDialogWidget()

CHAPTER – 4

RESULTS AND DISCUSSION, PERFORMANCE ANALYSIS

4.1 RESULTS

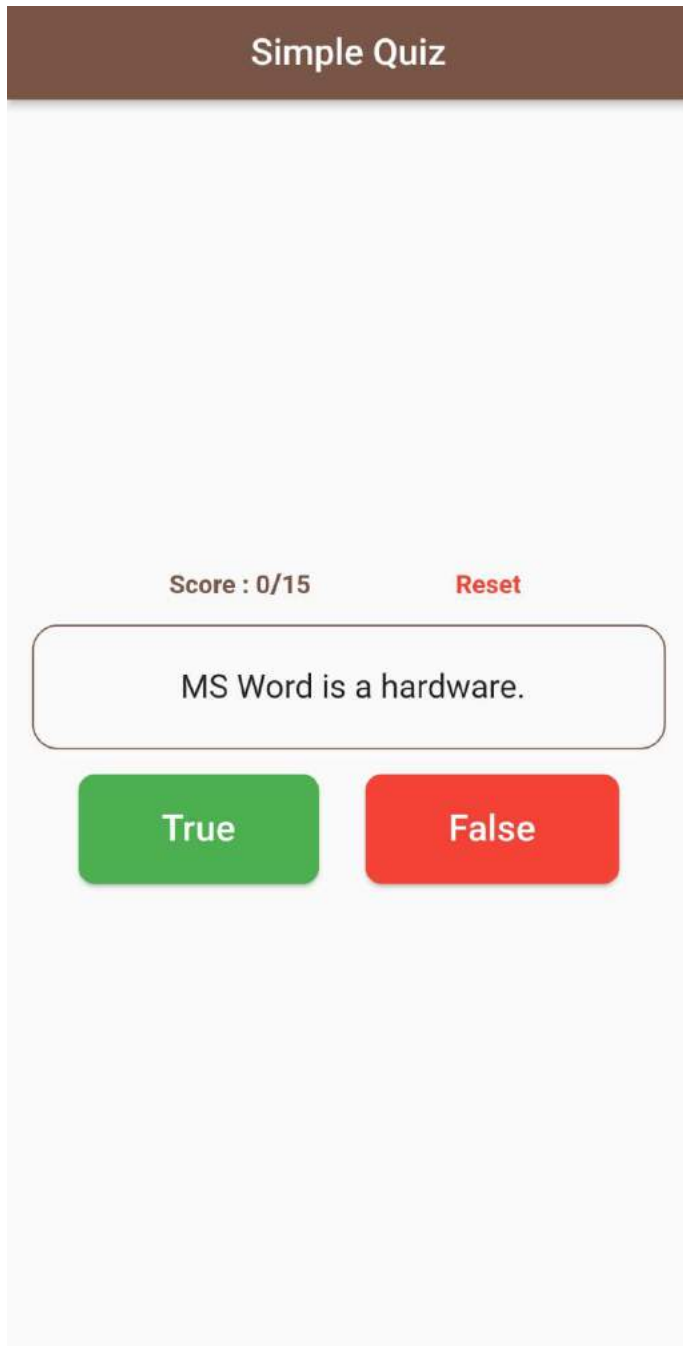


Fig 4.1 Home Screen

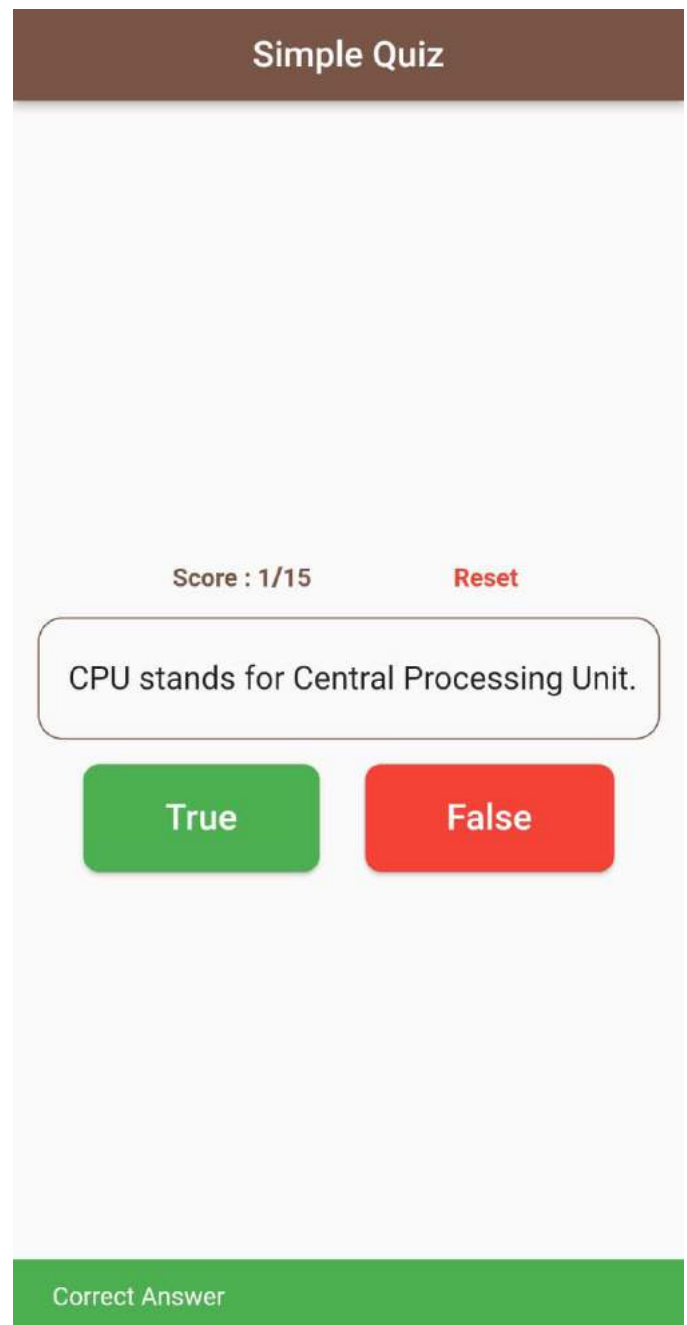


Fig 4.2 Correct Answer Screen

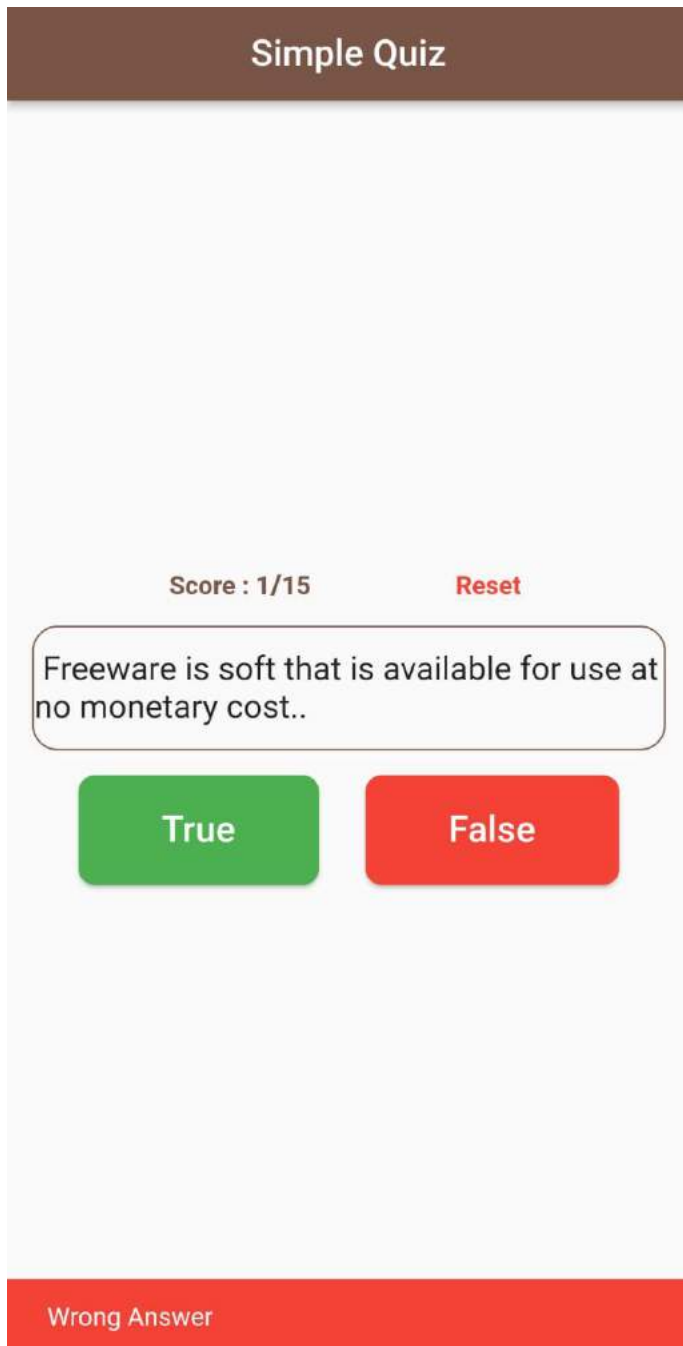


Fig 4.3 Wrong Answer Screen

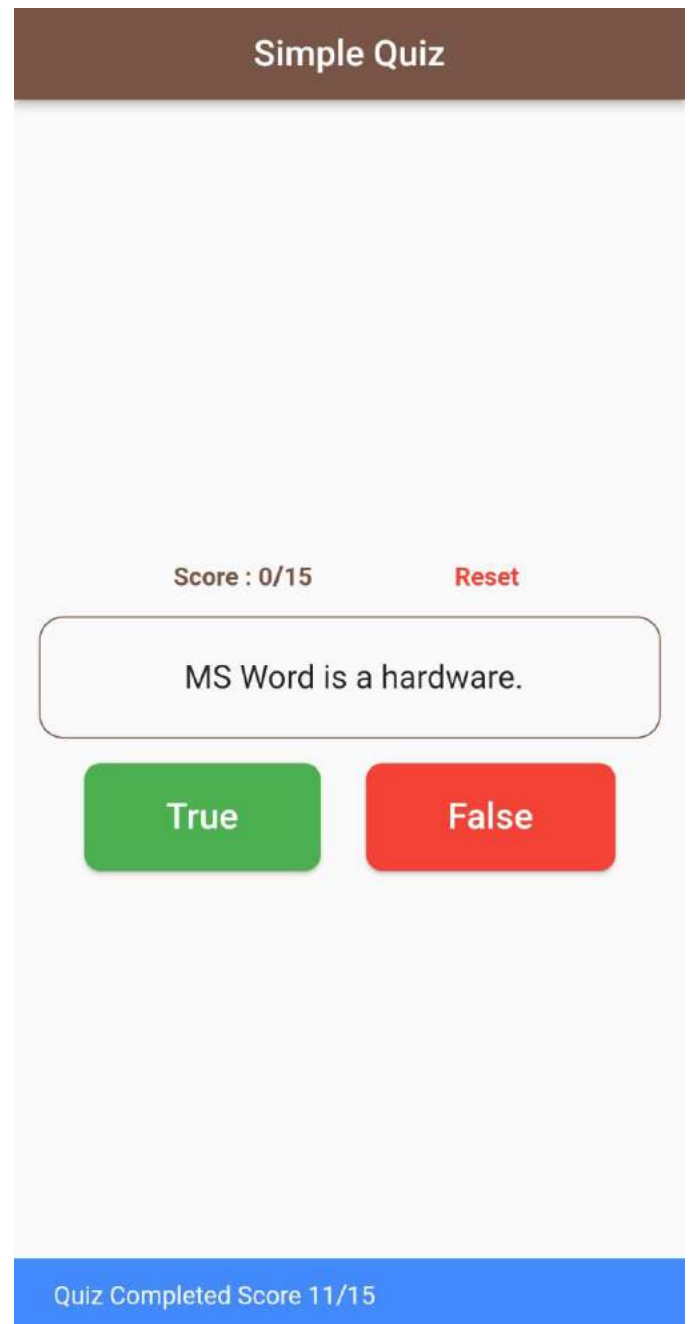


Fig 4.4 Result Screen

4.2. PERFORMANCE ANALYSIS

- The performance of the application is very efficient on latest android Os version 12
- The performance of the application is very efficient on latest mac version macos 12 monterey
- Internet connection is recommended
- Minimum RAM required is 2gb
- Due to efficient build strategy the app size is comparatively lower so the performance will be better
- Completely responsive

CHAPTER – 5

SUMMARY AND CONCLUSION

5.1. SUMMARY

- This project is done based on flutter frame work using dart programming language with the help of Android Studio and mobile emulator / real time device .
- Quiz application provides facility to attempt quiz anywhere and anytime.
- It save time since user does need to wait for result.

5.2. CONCLUSION

As we are leaving in a Fast growing and competitive world we need things to get completed at an instant. This Quiz Master application provides facility to attempt quiz anywhere and anytime. And it also enables student/user save time since user does need to wait for result. All Student/ user get extra knowledge and skills. Administrator has a privilege to put as much as question in given Quiz in application. User can register, log-in to attempt Quiz. If user forget their password then he/ she can reset password with the help of Forgot Password option. There is instruction page for students o get information about quiz like number of question, given time etc. After that student get the result of quiz. so, student cannot wait for result. also admin is responsible to add, delete and update question in the system. also he can view result of all students.

It has been a great pleasure for me to work on this exciting and challenging project. This project proved good for me as it provided practical knowledge of programming in Flutter. It also provides knowledge about the latest technology used in developing application and server technology the will be great demand in future. This will provide better opportunities and guidance in future in developing projects independently. This Quiz Application can be efficiently used by any students and Educational platforms. During the course of this assignment I have gone through many obstacle which made me to research and also increase my knowledge.

At last it can be concluded that the “Quiz Application” was the real learning experience. The project has been made as per as the given specification. The system has been made as user-friendly as possible

FUTURE WORK

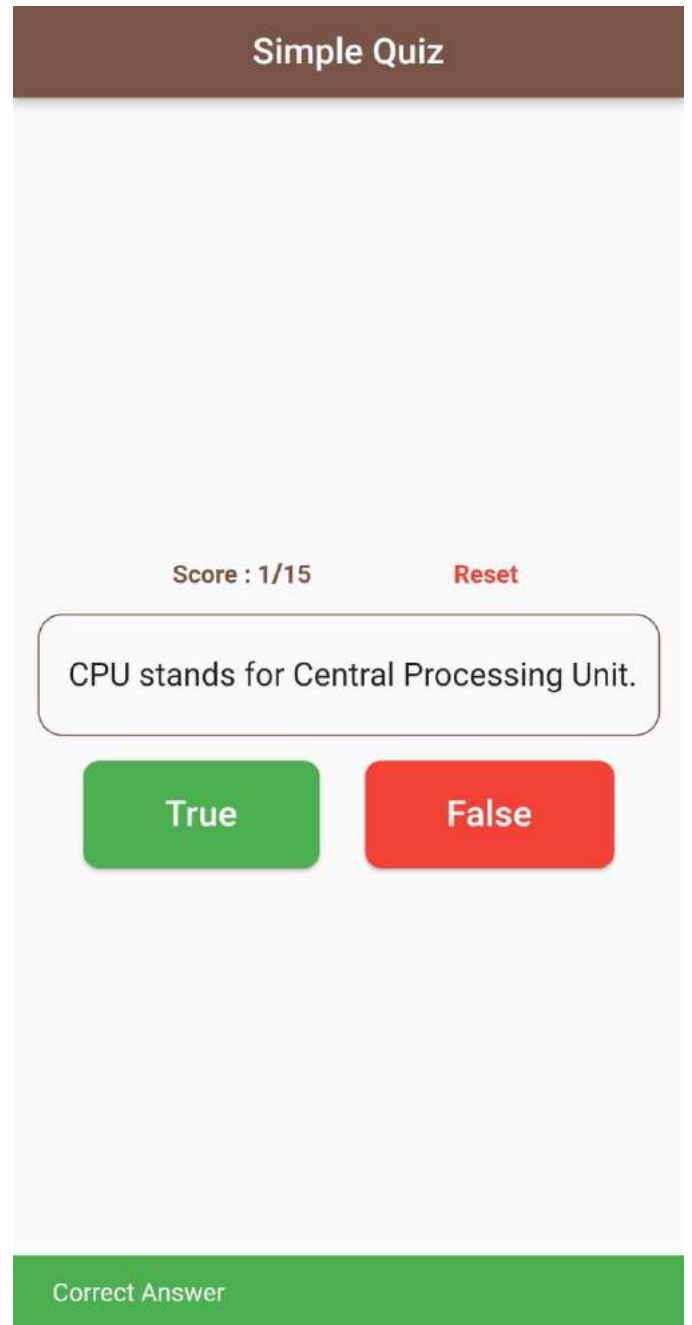
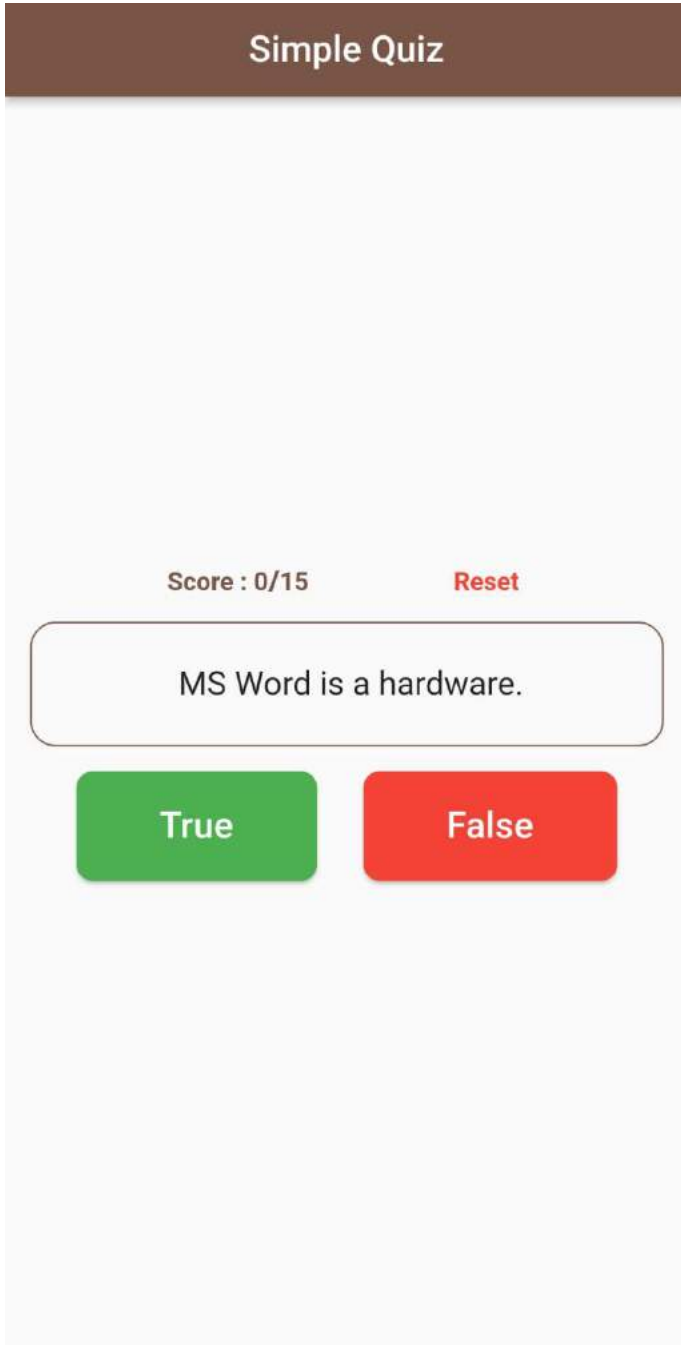
When it comes to mobile app development there are plenty of frameworks and programming languages to choose from. But, in recent years Flutter is gaining popularity because of its numerous benefits. Developers are enjoying working on this platform and the subsequent updates provide them with the flexibility and innovation they crave.

In fact, there has been a gradual growth in the popularity of Flutter SDK (software development kit) as about 42% of Statista survey respondents chose it as their preferred platform in 2021. While it has surely become the most used programming language for mobile app development, the release of Flutter 3 on May 11, 2022, is expected to further solidify this effect.

Flutter is easy to learn and implement as it is dependent on Dart programming language. Dart is simple and easy to learn by all programmers compared to other programming languages. Due to various Flutter 3 features like a single code base, support for multiple platforms, etc., Flutter stands out.

APPENDIX

A. SCREENSHOTS



Simple Quiz

Score : 1/15

[Reset](#)

Freeware is soft that is available for use at no monetary cost..

True

False

Wrong Answer

Simple Quiz

Score : 0/15

[Reset](#)

MS Word is a hardware.

True

False

Quiz Completed Score 11/15

B. SOURCE CODE

Main.dart

```
import 'package:flutter/material.dart';
import 'package:quiz_master/Questions.dart';

void main() {
  runApp( const MaterialApp(
    home: MyApp(),
    debugShowCheckedModeBanner: false,
  ));
}

class MyApp extends StatefulWidget {
  const MyApp({Key? key}) : super(key: key);

  @override
  _MyAppState createState() => _MyAppState();
}

class _MyAppState extends State<MyApp> {
  var score=0;
  var n=0;
  List que_list=[
    Questions(" MS Word is a hardware.", false),
    Questions(" CPU controls only input data of computer." , false),
    Questions(" CPU stands for Central Processing Unit.", true),
    Questions(" Freeware is soft that is available for use at no
monetary cost.. ", true),
    Questions(" Digital camera is input device used to take
photographs", true),
    Questions(" FAX stands for First Away Xerox", false),
    Questions(" Octal number system contains digits from 0-7", true),
    Questions(" GNU/Linux is not a open source operating system. ",
true),
    Questions(" The hexadecimal number system contains digits from 1-15.
", false),
    Questions(" Pen drive is a storage device. ", true),
    Questions(" CD stands for collective disk", false),
    Questions(" A Scanner is an output device", false),
    Questions(" Bluetooth allows your device to connect with another
device", true),
    Questions(" A terabyte is equal to 1 million gigabytes. ", false),
    Questions(" The CPU is referred as the stomach of the computer. ",
false)
  ];
  void checkAnswer(bool choice, BuildContext ctx){
    if(choice==que_list[n].ans)
    {
      //debugPrint("Correct");
      score=score+1;
      final snackbar=SnackBar(content: Text("Correct Answer"),
duration: Duration(milliseconds: 500),
      backgroundColor: Colors.green,
    );
      Scaffold.of(ctx).showSnackBar(snackbar);
    }
    else
    {

```

```

        final snackbar=SnackBar(content: Text("Wrong Answer"),
            duration: Duration(milliseconds: 500),
            backgroundColor: Colors.red,
        );
        Scaffold.of(ctx).showSnackBar(snackbar);
    }
    setState(() {
        if(n<que_list.length-1)
        {
            n=n+1;
        }
        else{
            final snackbar=SnackBar(content: Text("Quiz Completed Score
$score/15"),
                duration: Duration(seconds: 5),
                backgroundColor: Colors.blueAccent,
            );
            Scaffold.of(ctx).showSnackBar(snackbar);
            reset();
        }
    });
}

void reset(){
    setState(() {
        n=0;
        score=0;
    });
}

@override
Widget build(BuildContext context) {
    return Scaffold(
        appBar: AppBar(backgroundColor: Colors.brown,title: Text("Simple
Quiz"),centerTitle:true,),
        body: Builder(
            builder: (ctx)=>
                Container(
                    padding: EdgeInsets.symmetric(vertical: 30,horizontal:
15),

                    child: Column(
                        mainAxisAlignment: MainAxisAlignment.center,
                        children: <Widget>[
                            Row(
                                mainAxisAlignment: MainAxisAlignment.spaceEvenly,
                                children: <Widget>[
                                    Text("Score : $score/15 ",style: TextStyle(
                                        color: Colors.brown,
                                        fontWeight: FontWeight.bold
                                    )),
                                    InkWell(
                                        onTap: ()=>reset(),
                                        child: Text("Reset ",style: TextStyle(
                                            color: Colors.red,
                                            fontWeight: FontWeight.bold
                                        )),
                                    ),
                                ],
                            ),
                            SizedBox(height: 15,),
                            Container(

```

```

        height: 75,
        width: double.infinity,
        decoration: BoxDecoration(
          borderRadius: BorderRadius.circular(15.0),
          border: Border.all(color: Colors.brown),
        ),
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: <Widget>[
            Text(que_list[n].que, style: TextStyle(fontSize:
18.0))
          ],
        ),
      ),
      SizedBox(height: 15,),
      Row(
        mainAxisAlignment: MainAxisAlignment.spaceEvenly,
        children: <Widget>[
          RaisedButton(
            child: Text("True", style: TextStyle(color:
Colors.white, fontSize: 20.0)),
            padding: EdgeInsets.fromLTRB(50, 20, 50, 20),
            onPressed: ()=>checkAnswer(true, ctx),
            color: Colors.green,
            shape: RoundedRectangleBorder(
              borderRadius: BorderRadius.circular(10.0)
            ),
          ),
          RaisedButton(
            child: Text("False", style: TextStyle(color:
Colors.white, fontSize: 20.0)),
            padding: EdgeInsets.fromLTRB(50, 20, 50, 20),
            onPressed: ()=>checkAnswer(false, ctx),
            color: Colors.red,
            shape: RoundedRectangleBorder(
              borderRadius: BorderRadius.circular(10.0))
            ),
        ],
      ),
    ),
  ),
),
);
}
}

```

Pubspec.yml

```

name: quiz_master
description: A new Flutter project.

# The following line prevents the package from being accidentally
published to
# pub.dev using `flutter pub publish`. This is preferred for private
packages.
publish_to: 'none' # Remove this line if you wish to publish to pub.dev

# The following defines the version and build number for your

```

```

application.
# A version number is three numbers separated by dots, like 1.2.43
# followed by an optional build number separated by a +.
# Both the version and the builder number may be overridden in flutter
# build by specifying --build-name and --build-number, respectively.
# In Android, build-name is used as versionName while build-number used
as versionCode.
# Read more about Android versioning at
https://developer.android.com/studio/publish/versioning
# In iOS, build-name is used as CFBundleShortVersionString while build-
number used as CFBundleVersion.
# Read more about iOS versioning at
#
https://developer.apple.com/library/archive/documentation/General/Reference/InfoPlistKeyReference/Articles/CoreFoundationKeys.html
version: 1.0.0+1

environment:
  sdk: ">=2.17.6 <3.0.0"

# Dependencies specify other packages that your package needs in order
to work.
# To automatically upgrade your package dependencies to the latest
versions
# consider running `flutter pub upgrade --major-versions`.
Alternatively,
# dependencies can be manually updated by changing the version numbers
below to
# the latest version available on pub.dev. To see which dependencies
have newer
# versions available, run `flutter pub outdated`.
dependencies:
  flutter:
    sdk: flutter

  # The following adds the Cupertino Icons font to your application.
  # Use with the CupertinoIcons class for iOS style icons.
  cupertino_icons: ^1.0.2

dev_dependencies:
  flutter_test:
    sdk: flutter

  # The "flutter_lints" package below contains a set of recommended
  lints to
  # encourage good coding practices. The lint set provided by the
  package is
  # activated in the `analysis_options.yaml` file located at the root of
  your
  # package. See that file for information about deactivating specific
  lint
  # rules and activating additional ones.
  flutter_lints: ^2.0.0

# For information on the generic Dart part of this file, see the
# following page: https://dart.dev/tools/pub/pubspec

# The following section is specific to Flutter packages.
flutter:

  # The following line ensures that the Material Icons font is

```

```
# included with your application, so that you can use the icons in
# the material Icons class.
uses-material-design: true
```

Questions.dart

```
class Questions{
  String que;
  bool ans;
  Questions(this.que,this.ans);
}
```

Widget_test.dart

```
// This is a basic Flutter widget test.
//
// To perform an interaction with a widget in your test, use the
WidgetTester
// utility in the flutter_test package. For example, you can send tap
and scroll
// gestures. You can also use WidgetTester to find child widgets in the
widget
// tree, read text, and verify that the values of widget properties are
correct.

import 'package:flutter/material.dart';
import 'package:flutter_test/flutter_test.dart';

import 'package:quiz_master/main.dart';

void main() {
  testWidgets('Counter increments smoke test', (WidgetTester tester)
  async {
    // Build our app and trigger a frame.
    await tester.pumpWidget(const MyApp());

    // Verify that our counter starts at 0.
    expect(find.text('0'), findsOneWidget);
    expect(find.text('1'), findsNothing);

    // Tap the '+' icon and trigger a frame.
    await tester.tap(find.byIcon(Icons.add));
    await tester.pump();

    // Verify that our counter has incremented.
    expect(find.text('0'), findsNothing);
    expect(find.text('1'), findsOneWidget);
  });
}
```

Reference:

- <https://en.wikipedia.org/>
- <https://www.javatpoint.com/>
- <https://www.w3schools.com/>
- <https://www.tutorialspoint.com/>