

**A
PROJECT
ON**

“Whiz: UPSC Quiz App”

Submitted to

Shiksha Mandal's
**G. S. COLLEGE OF COMMERCE & ECONOMICS, NAGPUR
(AUTONOMOUS)**
In the Partial Fulfillment of

B.Com. (Computer Application) Final Year

Submitted by

Syed Arshad Hussain
Mustafa Hussaini

Under the Guidance of

Pravin J. Yadao



Shiksha Mandal's
**G. S. COLLEGE OF COMMERCE & ECONOMICS, NAGPUR
(AUTONOMOUS)**
2021-2022

Shiksha Mandal's
**G. S. COLLEGE OF COMMERCE & ECONOMICS,
NAGPUR
(AUTONOMOUS)
CERTIFICATE**

(2021 - 2022)

This is to certify that Mr. Syed Arshad Hussain and Mustafa Hussaini has completed their project on the topic of Whiz Quiz App prescribed by G. S. College of Commerce & Economics, Nagpur (Autonomous) for B.Com. (Computer Application) – Semester-VI.

Date: 07/04/2022

Place: Nagpur

Pravin J. Yadao

Project Guide

External Examiner

Internal Examiner

ACKNOWLEDGEMENT

We take this opportunity to express our deep gratitude and whole hearted thanks to project guide Prof. Pravin Yadao, Coordinator for his guidance throughout this work. We are very much thankful to him for his constant encouragement, support and kindness.

We are also grateful to our teachers Prof. Rahul Tiwari, Prof. Sushma Gawande, Prof. Preeti Rangari, Prof. Prajkta Deshpande and Prof. Haresh Naringe for their encouragement, help and support from time to time. We also wish to express our sincere thanks to Principal Dr. N. Y. Khandait for providing us wide range of opportunities, facilities and inspiration to gather professional knowledge and material without which this project could not have been completed.

Syed Arshad Hussain

Mustufa Hussaini

Student Names & Signature

Date: 07/04/2022

Place: Nagpur

DECLARATION

We (**Syed Arshad Hussain and Mustafa Hussaini**) hereby honestly declare that the work entitled “**Whiz Quiz App**” submitted by us at G. S. College of Commerce & Economics, Nagpur (Autonomous) in partial fulfillment of requirement for the award of B.Com. (Computer Application) degree by Rashtrasant Tukadoji Maharaj, Nagpur University, Nagpur has not been submitted elsewhere for the award of any degree, during the academic session 2021-2022. The project has been developed and completed by us independently under the supervision of the subject teacher and project guide.

Syed Arshad Hussain

Mustafa Hussaini

Student Name & Signature

Date: 07/04/2022

Place: Nagpur

INDEX

| SR. NO. | PARTICULARS | PAGE NO. | REMARK | SIGNATURE |
|---------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------|--------|-----------|
| 1 | Introduction | 1-2 | | |
| 2 | Objectives | 3-5 | | |
| 3 | Preliminary System Analysis <ul style="list-style-type: none">• Identification of Need• Preliminary Investigation• Present System in Use• Flaws in Present System• Need of New System• Feasibility Study• Project Category | 6-17 | | |
| 4 | Software & Hardware Requirement Specification | 18-19 | | |
| 5 | Detailed System Analysis <ul style="list-style-type: none">• Data Flow Diagram• Numbers of Modules and Process Logic• Data Structures and Tables• Entity-Relationship Diagram | 19-24 | | |
| 6 | System Design <ul style="list-style-type: none">• Form Design• Source Code• Input Screen & Output Screen | 25-53 | | |
| 7 | Testing & Validation Checks | 54-55 | | |
| 8 | System Security Measures | 56-57 | | |
| 9 | Implementation, Evaluation and Maintenance | 58-59 | | |
| 10 | Future Scope of the project | 60-61 | | |
| 11 | Conclusion | 62-63 | | |
| 12 | Bibliography & References | 64-65 | | |

INTRODUCTION

INTRODUCTION

In today's world, Smart phones have changed our lives and have become an indispensable part of our lives because of its speciality to specify our routine work and thereby saving our time and educational competitions are also high. So, the 'Whiz Quiz App' is developed to help students to judge and practicing their educational knowledge about the UPSC Exam through MCQ based pattern. Whiz Quiz App provides quiz on general knowledge topics where students can give MCQ's test and also get to know the scores. This Whiz Quiz App includes quiz on UPSC which cover various topics such as Sociology, History, Indian Economy, Geography, Current Affair, Indian Polity, Science & Technology and Environment etc.

Preparing for the Civil Services is an enriching and empowering experience. During IAS Preparation, candidates are exposed to a variety of topics and viewpoints adding to their knowledge and experience. An aspirant preparing for the UPSC Exam is expected to develop a large knowledge base on traditional subjects such as history, geography, languages and economics as well as dynamic subjects such as polity, social issues, management etc. The nature of the job of a Government Officer, such as the variety of work, the ability to effect lasting positive changes and the perks of service make the Civil Services is one of the most coveted careers in the country.

Development of this app is mainly required by students and learners to prepare themselves for UPSC examinations directly through smartphones and tablets in hands. One of the major goal of our project is to facilitate best quiz app to judge their knowledge and educational improvements about UPSC Examination. The main goal is to enable user to practice for subjective tests.

The students must register his/her name along with all information required and enter the email id and password for the login process. Using this email id and password to log in to the Whiz Quiz App. As soon as the student chooses the quiz topic, each topic contain 10 questions each, the questions with four choices will be shown. The students must choose any option as the answer. After submitting the answer if it is turn into Green color it means your answer is right and if it is turn into red you give wrong answer after completing the quiz you know the score .

OBJECTIVE

OBJECTIVE

The Whiz UPSC App offer UPSC aspirants the flexibility to study whenever they have free time and access to a smartphone. Generally, it is advisable to keep only a single app for IAS preparation as having multiple apps becomes distracting and slows down the preparation. With this in mind, IAS aspirants should choose only the best from a plethora of options available. The objective of this app are:

- This app is designed to help a student prepare for competitive examinations like UPSC
- The main objective of this Quiz System Project is to facilitate a user friendly environment to reduce the manual effort. The purpose of the system is to develop Quiz System., used to test the Domain knowledge of the students.
- To provide an interface through which student can solve quiz.
- The app should be easy to use and have a clean interface so that topics and study material can be found easily.
- The questions will be display after choosing topic.
- The UPSC app should also feature regular quizzes from various topics and also options for full-scale tests so that aspirants can test their knowledge on a regular basis and identify their strong and weak points.

- The main objective of this Quiz System Project is to facilitate a user friendly environment to reduce the manual effort. The purpose of the system is to develop Quiz System for aspirant, used to test the Domain knowledge of the students.
- Included Login and Sign Up pages, User can login with email address and password.
- The timer is the most exciting feature where the user can solve questions in a given time.

PRELIMINARY SYSTEM ANALYSIS

PRELIMINARY SYSTEM ANALYSIS

The main objectives of preliminary analysis is to identify the customer's needs, evaluate system concept for feasibility, perform economic and technical analysis, perform cost benefit analysis and create system definition that forms the foundation for all subsequent IT works. There should be enough expertise available for hardware and software for doing analysis.

While performing analysis, the following questions arise.

How much time should be spent on it? As such, there are no rules or formulas available to decide on this. However, size, complexity, application field, end-use, contractual obligation are few parameters on which it should be decided.

- Other major question that arises is who should do it. Well an experienced well-trained analyst should do it. For large project, there can be an analysis team.

After the preliminary analysis, the analyst should report the findings to management, with recommendations outlining the acceptance or rejection of the proposal.

Preliminary Investigation:

Preliminary Investigation basically refers to the collection of information that guides the management of an organization to evaluate the merits and demerits of the project request and make an informed judgment about the feasibility of the proposed system. This sort of investigation provides us with a through picture of the kind of software and hardware requirements which are most feasible for the system, plus the environment in which the entire project has to be installed and made operational. The preliminary-investigation phase sets the stage for gathering information about the current problem and the existing information system. This information is then used in studying the feasibility of possible information systems solutions.

Present system in use:

Traditional exams will die out within a decade. In a world that seems increasingly dominated by technology and computers, Tests in India continue to be primarily paper - and pencil - based. Online testing accounts for only around 20 percent of the total testing done in India.

We want to carefully examine the difference between online and pen and - paper quiz test. Further since the future seems to be strongly swaying towards online testing.

If we go to see the earlier standards of having an exam in a pen and paper format, there was no choice but to have it that way since the technology was not as advanced as to carry an online test. But this is not the scene today.

Preparing for the Civil Services is an enriching and empowering experience. During UPSC Preparation, candidates are exposed to a variety of topics and viewpoints adding to their knowledge and experience. An aspirant preparing for the UPSC Exam is expected to develop a large knowledge base on traditional subjects such as history, geography, languages and economics as well as dynamic subjects such as polity, social issues, management etc. The nature of the job of a Government Officer, such as the variety of work, the ability to effect lasting positive changes and the perks of service make the Civil Services is one of the most coveted careers in the country.

Preparation for the exam should be started early due to the high competition and vastness of the UPSC Syllabus. Technology has made it easier to master any topic through online platforms such as websites and smartphone apps. We focus on UPSC Exam Preparation through smartphone apps.

Flaws in present system:

As looking to the present system due to outdated version flaws is occurred in present system.

- 1) **Lack of security:** As security facility is not available so unsecure app can pose a problem there will be chances of misuse and also a user will be hesitate to visit system.
- 2) **User Interface:** As present system is not too good visitors will judge how your app looks so new user interface is required.
- 3) **Difficulty:** To finding the questions to reviewed.
- 4) **No changes allowed:** Once an answer is marked. Thus, higher chances of losing important marks.

- 5) **Slow loading Time:** Due to database slow loading time can absolutely kill the app experience of visitors.
- 6) **Basic Questions:** The quality of content in an UPSC preparation is low level, the needs of aspirants and the structure of the UPSC Exam is to be professional.

Needs of new system:

This app is designed in such ways which help in overcome all the flaws which is in current system .The present system is outdated and also security in not provided by current system so there will be need of new system.

- **Techniques and coding:** If app is developed several year ago it probably has a lot of unnecessary java code may slowing down app speed so modern techniques such as XML is help for app.
- **Content:** The first impression of our app will be over all layout but reader is visiting our app because they looking for useful and updated information.
- **Reliable Platform:** All quiz material provided was checked. What we need? Should provide you exactly.
- **All in one:** Now we need single platform which will provide all types of quizzes as well as user friendly platform.
- **Flextime:** You don't need a classroom environment for testing. Learners can solve quiz wherever they are, on their own schedule.

Feasibility Study

A feasibility study is an analysis that takes all of a project's relevant factors into account—including economic, technical, legal, and scheduling considerations—to ascertain the likelihood of completing the project successfully. A feasibility study is part of the initial design stage of any project/plan. It is conducted in order to objectively uncover the strength and weaknesses of a proposed project or an existing business. It can help to identify and assess the opportunities and threats present in the natural

environment, the resources required for the project, and the prospects for success. A feasibility study is an evaluation and analysis of a project or system that somebody has proposed. We also call it a feasibility analysis.

Feasibility Study in Software Engineering is a study to evaluate feasibility of proposed project or system. Feasibility study is one of stage among important four stages of Software Project Management Process. As name suggests feasibility study is the feasibility analysis or it is a measure of the software product in terms of how much beneficial product development will be for the organization in a practical point of view. Feasibility study is carried out based on many purposes to analyze whether software product will be right in terms of development, implantation, contribution of project to the organization etc.

A feasibility study is a high-level capsule version of the entire System analysis and Design Process. The study begins by classifying the problem definition. Feasibility is to determine if it's worth doing. Once an acceptance problem definition has been generated, the analyst develops a logical model of the system. A search for alternatives is analysed carefully.

What are the user's demonstrable needs?

The user wants a web-based system, which will reduce the bulk of paperwork, provide ease of work, flexibility, fast result.

How can the problem be redefined?

We proposed our perception of the system, in accordance with the problems of existing system by making a full layout of the system on paper. We tallied the problems and needs by existing system and requirements. We were further updating in the layout in the basis of redefined the problems. In feasibility study phase we had undergone through various steps, which are described as under: How feasible is the system proposed? This was analysed by comparing the following factors with both the existing system and proposed system.

Cost: The cost required in the proposed system is comparatively less to the existing system.

Effort: Compared to the existing system the proposed system will provide a better working environment in which there will be ease of work and the effort required will be comparatively less than the existing system.

Time: The time required generating a report or for doing any other work will be comparatively very less than in the existing system. Record finding and updating will take less time than the existing system.

Types of Feasibility Study:

The feasibility study mainly concentrates on below five mentioned areas. Among these Economic Feasibility Study is most important part of the feasibility analysis and Legal Feasibility Study is less considered feasibility analysis.

Technical Feasibility –

In Technical Feasibility current resources both hardware software along with required technology are analyzed/assessed to develop project. This technical feasibility study gives report whether there exists correct required resources and technologies which will be used for project development. Along with this, feasibility study also analyses technical skills and capabilities of technical team, existing technology can be used or not, maintenance and up-gradation is easy or not for chosen technology etc.

Operational Feasibility –

In Operational Feasibility degree of providing service to requirements is analyzed along with how much easy product will be to operate and maintenance after deployment. Along with this other operational scopes are determining usability of product, Determining suggested solution by software development team is acceptable or not etc.

Economic Feasibility –

In Economic Feasibility study cost and benefit of the project is analyzed. Means under this feasibility study a detail analysis is carried out what will be cost of the project for development which includes all required cost for final development like hardware and software resource required, design and development cost and operational cost and so on. After that it is analyzed whether project will be beneficial in terms of finance for organization or not.

Legal Feasibility –

In Legal Feasibility study project is analyzed in legality point of view. This includes analyzing barriers of legal implementation o project, data protection acts or social media laws, project certificate, license, copyright etc. Overall it can be said that Legal Feasibility Study is study to know if proposed project conform legal and ethical requirements.

Social feasibility -

Social feasibility is a detailed study on how one interacts with others within a system or an organization. Social impact analysis is an exercise aimed at identifying and analyzing such impacts in order to understand the scale and reach of the project's social impacts.

Project Category:

In this project “Whiz Quiz App” we use Android Java programming language and Firebase as database and for styling of Application we use XML.

Java:

25 years on, Java still remains the most popular programming language among developers, despite all the new entrants that made their mark. In a world where new technology quickly replaces old ones, none has been able to replace Java.

One of the biggest reasons why Java is the first choice of all app developers is because it is very easy to learn and get started with, and also offers wide-reaching community support which is an added help to new developers.

Don't let the ease of learning, Java is a power-packed programming language for mobile apps. Some of the best Android apps have been developed using Java, including Spotify, Twitter and of course, the Android Operating System.

With the Android OS itself being developed in Java, you will be able to easily develop all android apps once you master Java. Some of the best features of Java include:

- The simple, easy to understand syntax of Java is much more readable than Python and other coding languages used for mobile app development.
- Being an Object-Oriented programming language, it handles the complexity of real-world applications better.
- With its rich API, Java allows you to integrate much more into your app.
- Java's rich community support helps new developers truly hone their skills and never be stuck without any form of help. You can expect to find answers quickly and in adequate detail.

- Another thing you'll love about Java is that it is a strongly typed language. This means that it will catch many mistakes you make as a newbie.
- To sum up, Java is one of the best programming languages for app development, especially if you're starting out in the field of app development and need an android programming language that delivers on all counts.

Firestore:

Firestore is basically a Google-backed app development platform which was initially developed by James Tamplin and Andrew Lee in 2011. It was officially launched in 2012, and right after the two years of launch, Google acquired this platform. In the beginning, Firestore was only designed as a Realtime database but after its acquisition by Google, it started giving more services.

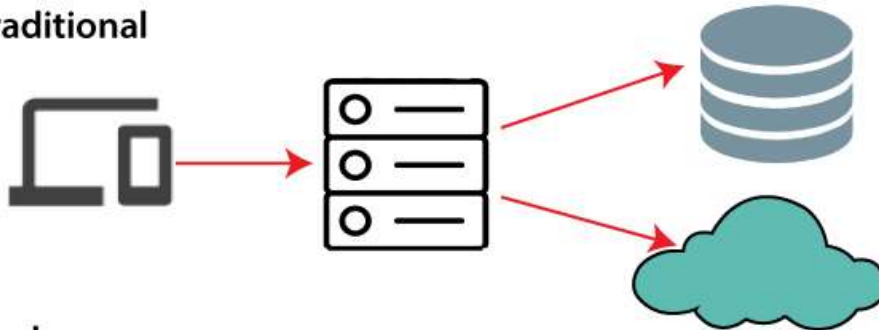
In simple words, Firestore is a software development platform that helps in building web and mobile applications with its 18 services. These 18 services of this BaaS solution also include purposeful APIs and four beta products. In addition, it is compatible to integrate with Android, web, iOS, and Unity setups.

Features:

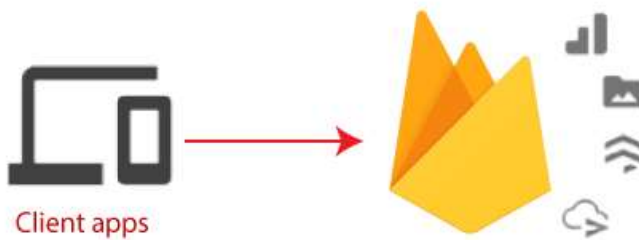
- Firestore manages real-time data in the database. So, it easily and quickly exchanges the data to and from the database. Hence, for developing mobile apps such as live streaming, chat messaging, etc., we can use Firestore.
- Firestore allows syncing real-time data across all devices - iOS, Android, and Web - without refreshing the screen.
- Firestore provides integration to Google Advertising, AdMob, Data Studio, BigQuery DoubleClick, Play Store, and Slack to develop our apps with efficient and accurate management and maintenance.
- Everything from databases, analytics to crash reports are included in Firestore. So, the app development team can stay focused on improving the user experience.

- Firebase applications can be deployed over a secured connection to the firebase server.
- Firebase offers a simple control dashboard.
- It offers a number of useful services to choose from.

Traditional



Firebase



Google Firebase is Google-backed application development software which allows developers to develop **Android, IOS, and Web apps**. For reporting and fixing app crashes, tracking analytics, creating marketing and product experiments, firebase provides several tools.



Firebase has three main services, i.e., a real-time database, user authentication, and hosting. We can use these services with the help of the Firebase iOS SDK to create apps without writing any server code.

Realtime Database:

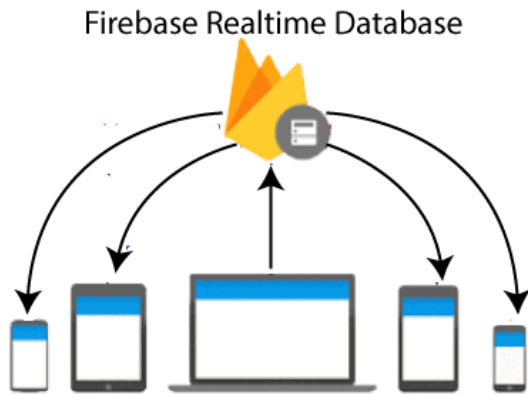
The Firebase Realtime Database is a cloud-hosted database in which data is stored as JSON. The data is synchronized in real-time to every connected client. All of our clients share one Realtime Database instances and automatically receive updates with the newest data, when we build cross-platform applications with our iOS, and JavaScript SDKs.

The Firebase Realtime Database is a NoSQL database from which we can store and sync the data between our users in real-time. It is a big JSON object which the developers can manage in real-time. By using a single API, the Firebase database provides the application with the current value of the data and updates to that data. Real-time syncing makes it easy for our users to access their data from any device, be it web or mobile.

Firestore Database:

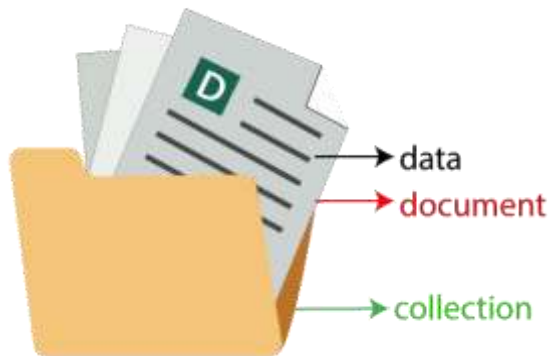
We have two options with Firebase, i.e., Firebase Real-time Database, and Cloud Firestore. Cloud Firestore is newer, but it is not replacing the Firebase Real-time Database. Cloud Firestore is a flexible as well as scalable NoSQL cloud database. It is used to store and sync data for client and server-side development. It is used for mobile, web, and server development from Google Cloud Platform and Firebase. Like the Firebase Real-time Database, it keeps syncing our data via real-time listeners to the client app. It provides offline support for mobile and web so we can create responsive apps that work regardless of network latency or Internet connectivity.

Cloud Firestore also provides seamless integration with Google Cloud Platform products and other Firebase, including cloud functions.



How does it work?

Cloud Firestore, a cloud-hosted, NoSQL database, is accessed directly through the native SDK by our iOS, Android, and web apps. In addition to REST and RPC APIs, Cloud Firestore is also available in native Node.js, Java, Python, and Go SDKs.



After Cloud Firestore's NoSQL data model, we can store data in documents that have field mappings for values. The documents are stored in a container called collections. These containers are used to organize our data and create queries. There are different data types, from simple string and numbers to complex nested objects, supported by documents. We can also create sub-collection within a document and create a hierarchical data structure that scales to the growth of our database. The Firestore data model supports whatever data structure works best for our app.

Additionally, the query in Cloud Firestore is expressive, efficient, and flexible. The shallow queries are created to retrieve data at the document level without the need to retrieve the entire collection or any nested subdivision. Add sorting, filtering, and limits for our queries or cursors to

index the result. Add a real-time listener to our app for keeping the data running. Every time it is updated without recovering our entire database.

Adding real-time listeners to our app informs us with a data snapshot whenever our customer apps are changing data, only getting new changes.

For protecting our data access in Cloud Firestore, Firebase authentication, and Cloud Firestore security rules are used for Identity and Access Management (IAM).

XML:

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

SOFTWARE AND HARDWARE REQUIREMENTS SPECIFICATIONS

SOFTWARE AND HARDWARE REQUIREMENTS SPECIFICATIONS

Every application needs the software in which it has to be executed and a hardware the application is going to perform its function. Some application cannot run on every platforms and some applications needs some specific requirement in the software or in hardware to get operated. Lets take an example of the applications which cannot be run on every platforms like windows, android, linux, etc. Applications made in Android Studio is only supported for the windows, one cannot access this applications from the mobile phones, etc. So, here are some hardware and software specifications which are mandatory for the application to get operated.

HARDWARE:

Hardware is a term that refers to all the physical parts that make up a computer. The internal hardware devices that make up the computer. Various devices which are essentials to form a hardware is called as components. Following are the hardware specifications that is required to develop this project is as follows:

64-bit Microsoft® Windows® 8/10.

x86/64 CPU architecture; 2nd generation Intel Core or newer, or AMD CPU with support for a Windows Hypervisor.

8 GB RAM or more.

8 GB of available disk space minimum (IDE + Android SDK + Android Emulator)

1280 x 800 minimum screen resolution.

SOFTWARE:

The system software layer interfaces with the operating system, which in turn communicates with the hardware.

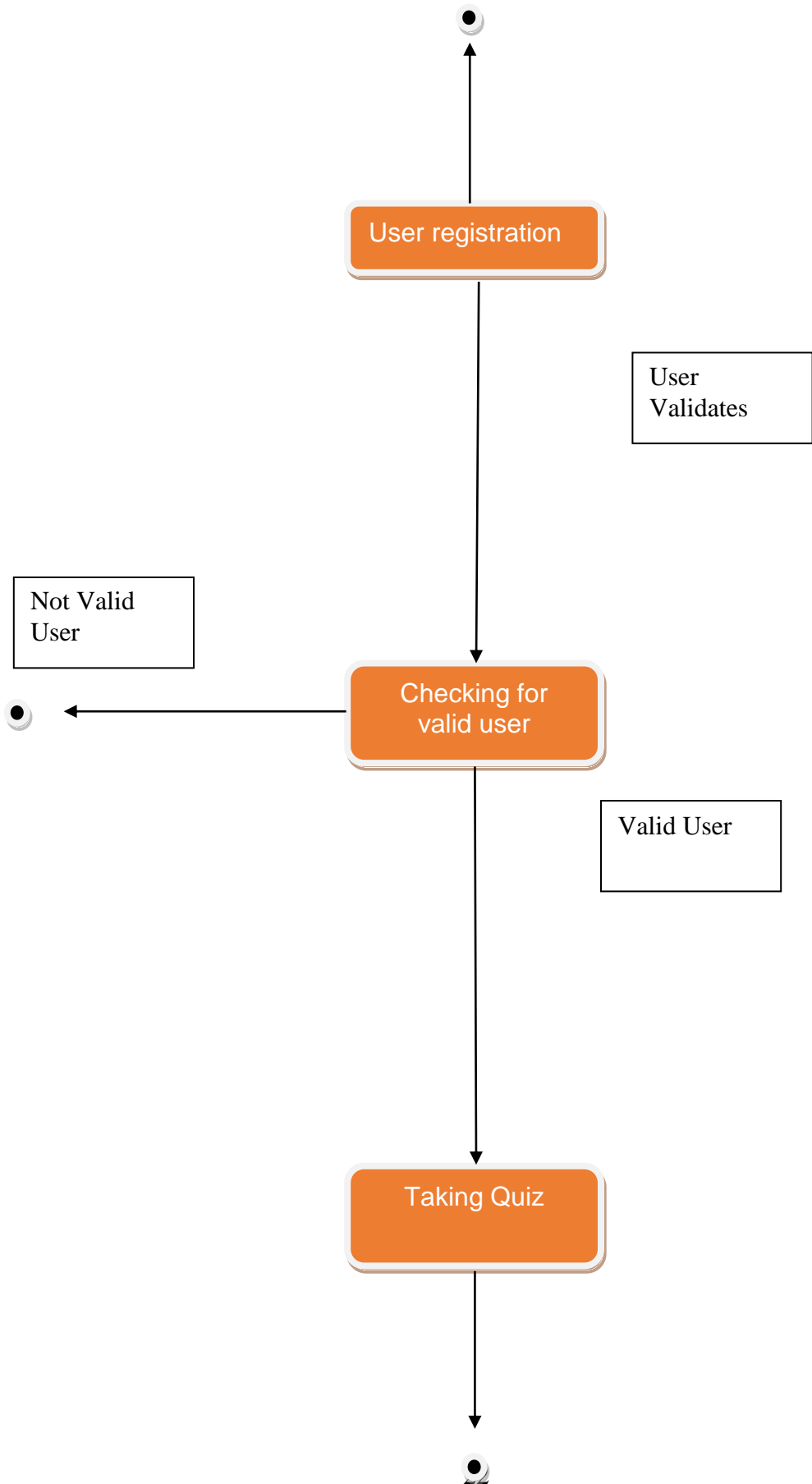
- Internet Explorer.
- Mozilla Firefox.
- Google Chrome.
- Android Studio

Operating System: Windows 7, Windows 8, Windows 10 or above versions.

Server Required: Firebase

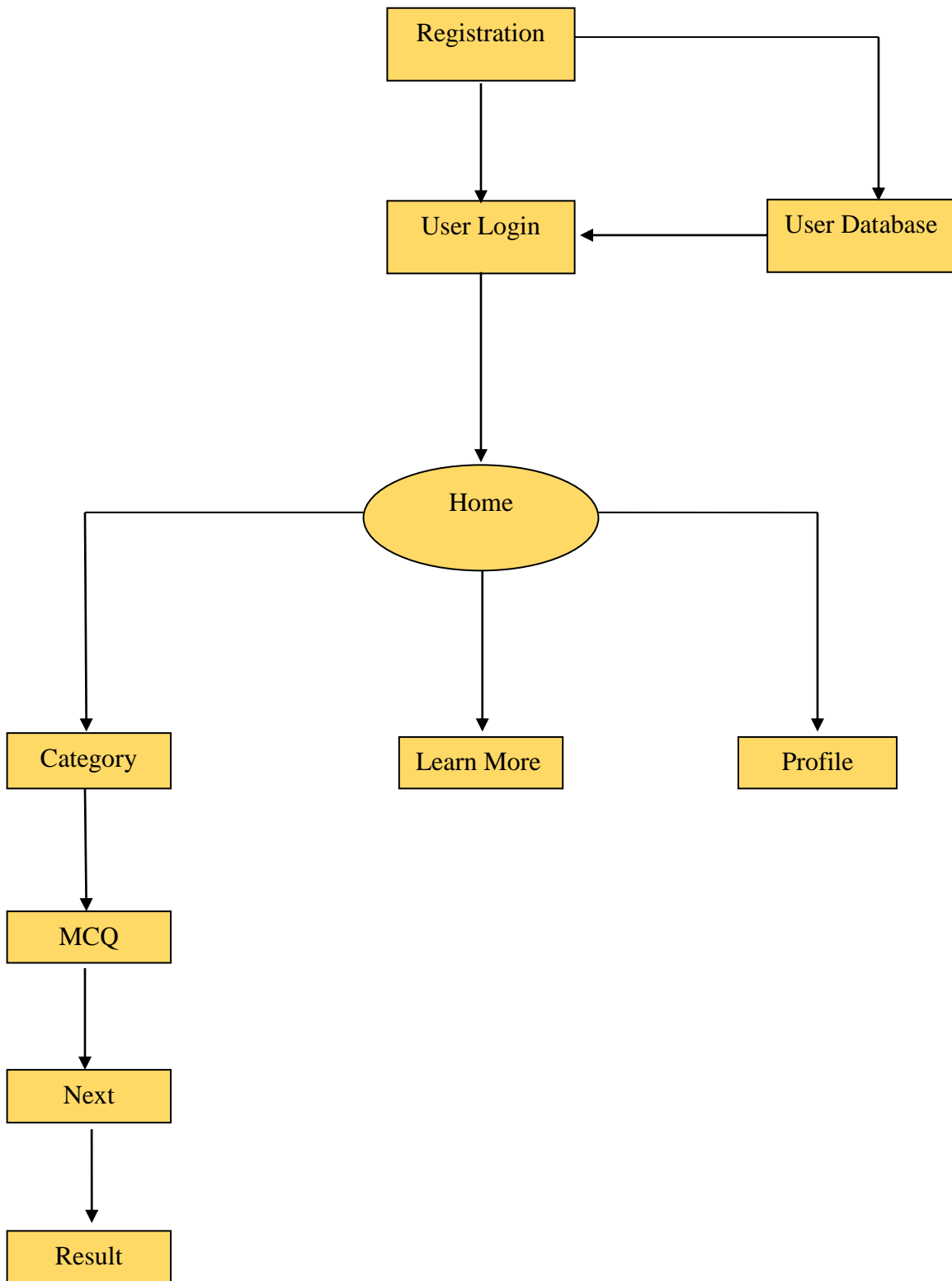
DETAILED SYSTEM ANALYSIS

Data Flow Diagram:



Data Structure and Tables:

Data Structure:



Data Table:

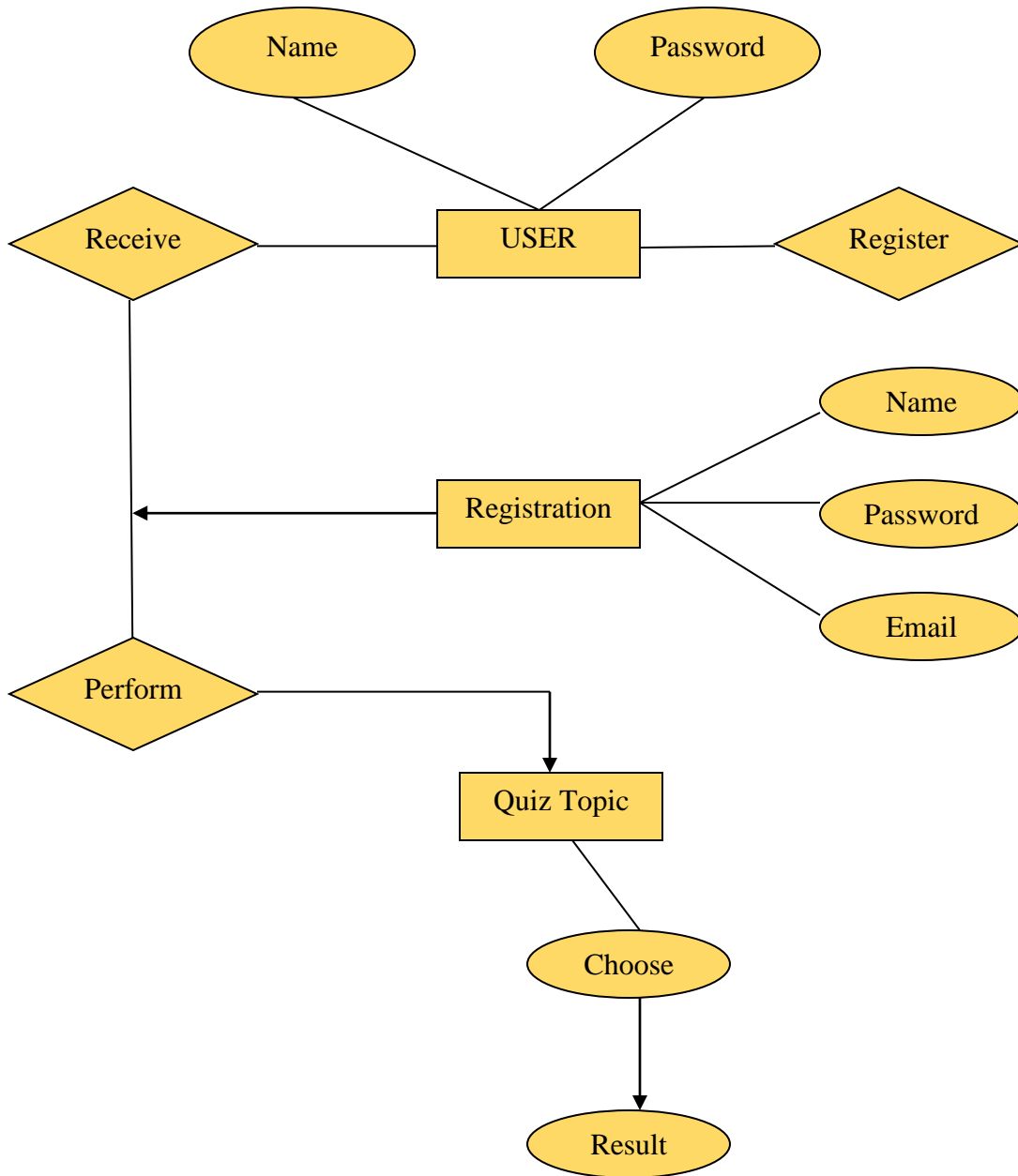
Cloud Firestore

Data Rules Indexes Usage

The screenshot shows the Cloud Firestore Data console. The breadcrumb path is `users > ZwCsXtsUYcWD...`. The left sidebar shows the project `whiz-quiz-app-7d0ee` and a collection `users` selected. The main area shows a document with ID `ZwCsXtsUYcWDEvQG09w9VbjzxsI3`. The document content is:

```
{
  "email": "arshadsabi31@gmail.com",
  "name": "Syed Arshad Hussain",
  "pass": "hussain25"
}
```

Entity-Relationship Diagram:



SYSTEM DESIGN

FORM DESIGN:

Sign Up Form –

7:52 67%

Whiz
UPSC Quiz App

CREATE AN ACCOUNT

SUBMIT

ALREADY HAVE AN ACCOUNT?

Login Form-

7:52 67%

Whiz
UPSC Quiz App

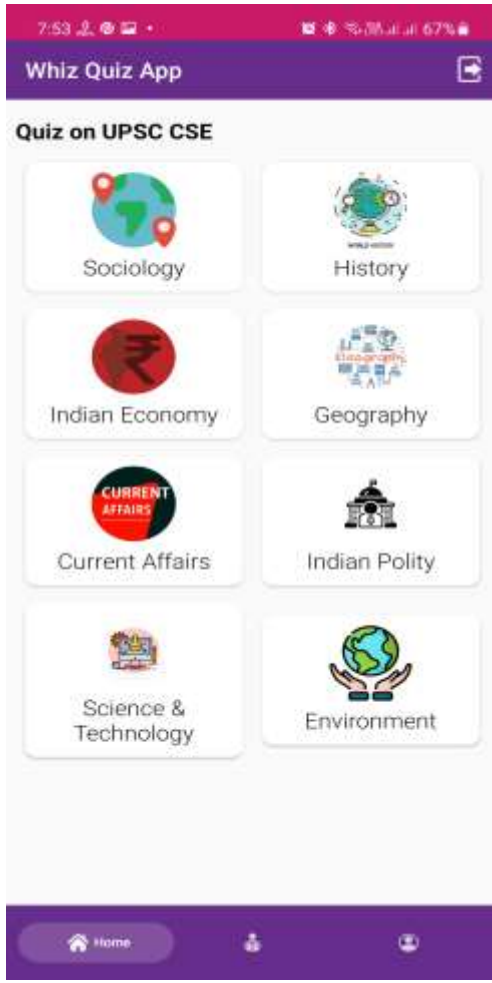
LOGIN WITH ACCOUNT

[Forget you password?](#)

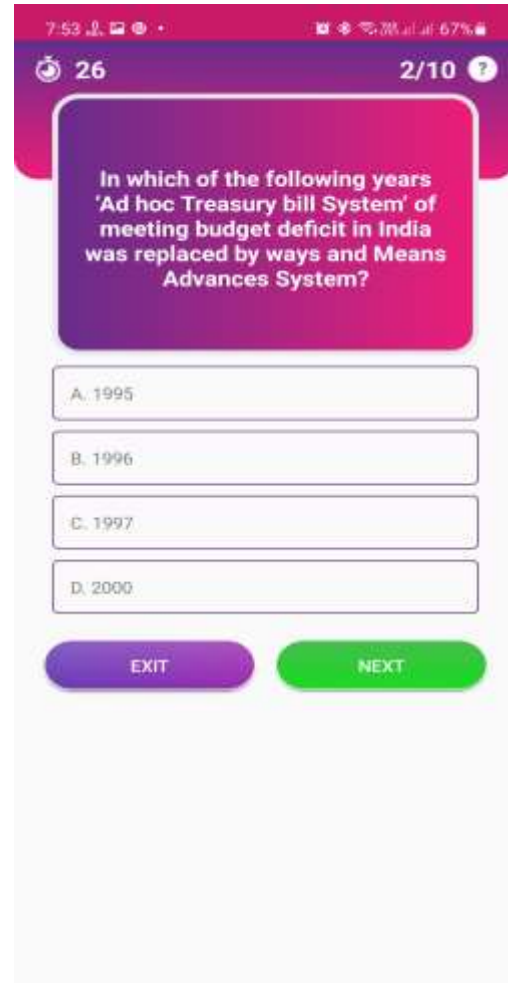
LOGIN

CREATE AN ACCOUNT

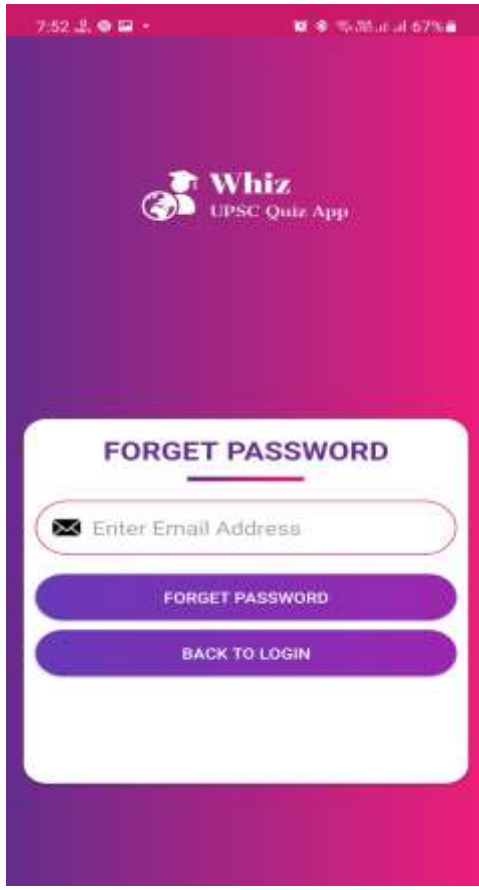
Home Page-



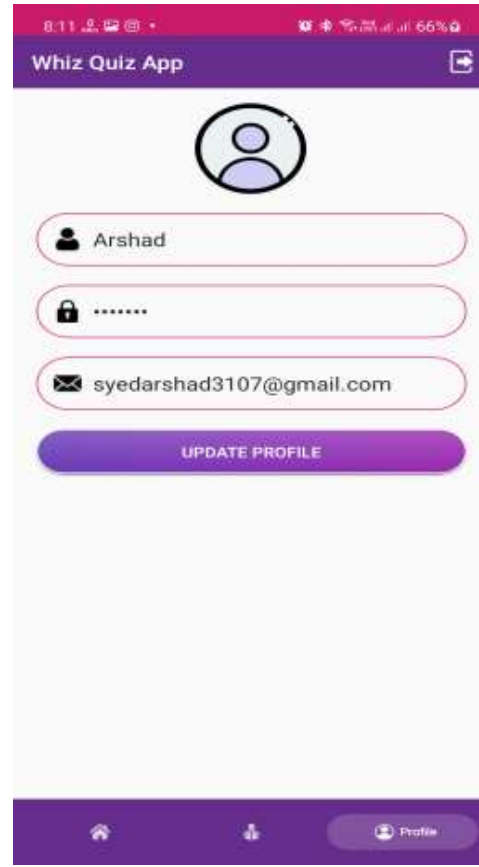
Quiz From –



Forget Password –



Profile –



SOURCE CODE:

LoginActivity.java

```
package com.example.myapplication;

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

import android.app.Activity;
import android.app.AlertDialog;
import android.content.Intent;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.View;
import android.widget.TextView;
import android.widget.Toast;

import com.example.myapplication.databinding.ActivityLoginBinding;
import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;

public class LoginActivity extends AppCompatActivity {

    ActivityLoginBinding binding;
    FirebaseAuth auth;
    ProgressDialog dialog;
    TextView openForgetPass;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        binding = ActivityLoginBinding.inflate(getLayoutInflater());
        setContentView(binding.getRoot());

        auth = FirebaseAuth.getInstance();

        dialog = new ProgressDialog(this);
        dialog.setMessage("Logging in...");
```

```

if (auth.getCurrentUser() !=null) {
    startActivity(new Intent(LoginActivity.this, MainActivity.class));
    finish();
}

openForgetPass = findViewById(R.id.ForgetPassword);

binding.submitBtn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        String email, pass;
        email = binding.emailBox.getText().toString();
        pass = binding.passwordBox.getText().toString();

        // Validations for input email and password
        if (TextUtils.isEmpty(email)) {
            Toast.makeText(getApplicationContext(), "Please enter your
email", Toast.LENGTH_LONG).show();
            return;
        }
        if (TextUtils.isEmpty(pass)) {
            Toast.makeText(getApplicationContext(), "Please enter your
password", Toast.LENGTH_LONG).show();
            return;
        }

        dialog.show();

        auth.signInWithEmailAndPassword(email,
pass).addOnCompleteListener(new OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {
                dialog.dismiss();
                if (task.isSuccessful()) {
                    startActivity(new Intent(LoginActivity.this,
MainActivity.class));
                    finish();
                } else {

```


SignupActivity.java

```
package com.example.myapplication;

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

import android.app.ProgressDialog;
import android.content.Intent;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.View;
import android.widget.ProgressBar;
import android.widget.Toast;

import com.example.myapplication.databinding.ActivitySignupBinding;
import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.firestore.FirebaseFirestore;

public class SignupActivity extends AppCompatActivity {

    ActivitySignupBinding binding;
    FirebaseAuth auth;
    FirebaseFirestore database;
    ProgressDialog dialog;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        binding = ActivitySignupBinding.inflate(getLayoutInflater());
        setContentView(binding.getRoot());

        auth = FirebaseAuth.getInstance();
        database = FirebaseFirestore.getInstance();

        dialog = new ProgressDialog(this);
        dialog.setMessage("Wait a moment creating a new account...");
    }
}
```

```

binding.createNewBtn.setOnClickListener(new
View.OnClickListener() {
    @Override
    public void onClick(View view) {
        String email, pass, name;

        email = binding.emailBox.getText().toString();
        pass = binding.passwordBox.getText().toString();
        name = binding.nameBox.getText().toString();

        final User user = new User(name, email, pass);

        // Validations for input email and password
        if (TextUtils.isEmpty(name)) {
            Toast.makeText(getApplicationContext(),
                "Please enter your details",
                Toast.LENGTH_LONG)
                .show();
            return;
        }
        if (TextUtils.isEmpty(email)) {
            Toast.makeText(getApplicationContext(),
                "Please enter your email",
                Toast.LENGTH_LONG)
                .show();
            return;
        }
        if (TextUtils.isEmpty(pass)) {
            Toast.makeText(getApplicationContext(),
                "Please enter your password",
                Toast.LENGTH_LONG)
                .show();
            return;
        }
        dialog.show();

        auth.createUserWithEmailAndPassword(email,
pass).addOnCompleteListener(new OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {
                if (task.isSuccessful()) {

```


MainActivity.java

```
package com.example.myapplication;

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

import android.content.Intent;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.Toast;

import com.example.myapplication.databinding.ActivityMainBinding;
import com.google.firebase.auth.FirebaseAuth;

import me.ibrahimsn.lib.OnItemSelectedListener;

public class MainActivity extends AppCompatActivity {

    ActivityMainBinding binding;
    FirebaseAuth mAuth;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        binding = ActivityMainBinding.inflate(getLayoutInflater());
        setContentView(binding.getRoot());

        mAuth = FirebaseAuth.getInstance();

        setSupportActionBar(binding.toolbar);

        FragmentTransaction transaction =
        getSupportFragmentManager().beginTransaction();
        transaction.replace(R.id.content, new HomeFragment());
        transaction.commit();

        binding.bottomBar.setOnItemSelectedListener(new
        OnItemSelectedListener() {
```

```

@Override
public boolean onOptionsItemSelected(int i) {
    FragmentTransaction transaction =
getSupportFragmentManager().beginTransaction();
    switch (i) {
        case 0:
            transaction.replace(R.id.content, new HomeFragment());
            transaction.commit();
            break;
        case 1:
            transaction.replace(R.id.content, new LearnMoreFragment());
            transaction.commit();
            break;
        case 2:
            transaction.replace(R.id.content, new ProfileFragment());
            transaction.commit();
            break;
    }
    return false;
}

});
}
}

```

```

@Override
public boolean onCreateOptionsMenu(Menu menu) {
    getMenuInflater().inflate(R.menu.home_menu, menu);
    return true;
}

```

```

@Override
public boolean onOptionsItemSelected(@NonNull MenuItem item) {
    switch (item.getItemId()) {
        case R.id.logout:
            mAuth.signOut();
            Intent intent = new Intent(MainActivity.this, LoginActivity.class);
            startActivity(intent);
            Toast.makeText(this, "Logging out...",
Toast.LENGTH_SHORT).show();
            break;
    }
}

```

```

    }
    return super.onOptionsItemSelected(item);
}

/**public void logout()
{
    Toast.makeText(this, "Logging out, please wait...",
Toast.LENGTH_SHORT).show();

}**/
}

```

QuizActivity.java

```

package com.example.myapplication;

import androidx.appcompat.app.AppCompatActivity;

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

import com.example.myapplication.databinding.ActivityQuizBinding;
import com.google.android.gms.tasks.OnSuccessListener;
import com.google.firebase.firestore.DocumentSnapshot;
import com.google.firebase.firestore.FirebaseFirestore;
import com.google.firebase.firestore.QuerySnapshot;

import java.util.ArrayList;
import java.util.Random;

public class QuizActivity extends AppCompatActivity {

    ActivityQuizBinding binding;

    ArrayList<Question> questions;

```

```

int index = 0;
Question question;
CountDownTimer timer;
FirebaseFirestore database;
int correctAnswers = 0;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    binding = ActivityQuizBinding.inflate(getLayoutInflater());
    setContentView(binding.getRoot());

    questions = new ArrayList<>();
    database = FirebaseFirestore.getInstance();

    final String catId = getIntent().getStringExtra("catId");

    //Random random = new Random();
    //final int rand = random.nextInt(10);

    database.collection("categories")
        .document(catId)
        .collection("questions")
        //.whereGreaterThanOrEqualTo("index", rand)
        //.orderBy("index")
        /***.limit(5)*/.get().addOnSuccessListener(new
OnSuccessListener<QuerySnapshot>() {
    @Override
    public void onSuccess(QuerySnapshot queryDocumentSnapshots) {
        if (queryDocumentSnapshots.getDocuments().size() < 5) {
            database.collection("categories")
                .document(catId)
                .collection("questions")
                ///.whereLessThanOrEqualTo("index", rand)
                ///.orderBy("index")
                /***.limit(5)*/.get().addOnSuccessListener(new
OnSuccessListener<QuerySnapshot>() {
                @Override
                public void onSuccess(QuerySnapshot
queryDocumentSnapshots) {

```

```

        for (DocumentSnapshot snapshot :
queryDocumentSnapshots) {
            Question question = snapshot.toObject(Question.class);
            questions.add(question);
        }
        setNextQuestion();
    }
});
} else {
    for (DocumentSnapshot snapshot : queryDocumentSnapshots) {
        Question question = snapshot.toObject(Question.class);
        questions.add(question);
    }
    setNextQuestion();
}
}
});

```

```

resetTimer();

```

```

/////ExitButton/////
binding.btnExit.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        startActivity(new Intent(QuizActivity.this,MainActivity.class));
    }
});
}

```

```

void resetTimer() {
    timer = new CountdownTimer(30000, 1000) {
        @Override
        public void onTick(long millisUntilFinished) {
            binding.timer.setText(String.valueOf(millisUntilFinished / 1000));
        }

        @Override
        public void onFinish() {

```

```

        }
    };
}

void showAnswer() {
    if (question.getAnswer().equals(binding.option1.getText().toString()))

binding.option1.setBackground(getResources().getDrawable(R.drawable.option_right));
    else if
(question.getAnswer().equals(binding.option2.getText().toString()))

binding.option2.setBackground(getResources().getDrawable(R.drawable.option_right));
    else if
(question.getAnswer().equals(binding.option3.getText().toString()))

binding.option3.setBackground(getResources().getDrawable(R.drawable.option_right));
    else if
(question.getAnswer().equals(binding.option4.getText().toString()))

binding.option4.setBackground(getResources().getDrawable(R.drawable.option_right));

}

void setNextQuestion() {
    if (timer != null)
        timer.cancel();

    timer.start();
    if (index < questions.size()) {
        binding.questionCounter.setText(String.format("%d/%d", (index +
1), questions.size()));
        question = questions.get(index);
        binding.question.setText(question.getQuestion());
        binding.option1.setText(question.getOption1());
        binding.option2.setText(question.getOption2());
    }
}

```

```

        binding.option3.setText(question.getOption3());
        binding.option4.setText(question.getOption4());
    }

    binding.nextBtn.setClickable(false);

}

void checkAnswer(Textview textView) {
    String selectedAnswer = textView.getText().toString();
    if (selectedAnswer.equals(question.getAnswer())) {
        correctAnswers++;

textView.setBackground(getResources().getDrawable(R.drawable.option_ri
ght));
    } else {
        showAnswer();

textView.setBackground(getResources().getDrawable(R.drawable.option_w
rong));
    }

    binding.nextBtn.setClickable(true);
}

void reset() {

binding.option1.setBackground(getResources().getDrawable(R.drawable.opt
ion_unselected));

binding.option2.setBackground(getResources().getDrawable(R.drawable.opt
ion_unselected));

binding.option3.setBackground(getResources().getDrawable(R.drawable.opt
ion_unselected));

binding.option4.setBackground(getResources().getDrawable(R.drawable.opt
ion_unselected));
}

```

```

public void onClick(View view) {
    switch (view.getId()) {
        case R.id.option_1:
        case R.id.option_2:
        case R.id.option_3:
        case R.id.option_4:
            if (timer != null)
                timer.cancel();
            TextView selected = (TextView) view;
            checkAnswer(selected);
            disableButton();

            break;
        case R.id.next_Btn:
            reset();
            if (index <= questions.size()) {
                index++;
                enableButton();
                setNextQuestion();

            } else {
                Intent intent = new Intent(QuizActivity.this,
ResultActivity.class);
                intent.putExtra("correct", correctAnswers);
                intent.putExtra("total", questions.size());
                startActivity(intent);
                Toast.makeText(this, "Quiz Finished",
Toast.LENGTH_SHORT).show();
            }
            break;
    }
}

public void enableButton(){
    binding.option1.setClickable(true);
    binding.option2.setClickable(true);
    binding.option3.setClickable(true);
    binding.option4.setClickable(true);
}

```



```

    public void disableButton(){
        binding.option1.setClickable(false);
        binding.option2.setClickable(false);
        binding.option3.setClickable(false);
        binding.option4.setClickable(false);
    }
}

```

ResultActivity.java

```

package com.example.myapplication;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;

import com.example.myapplication.databinding.ActivityResultBinding;

public class ResultActivity extends AppCompatActivity {

    ActivityResultBinding binding;
    int POINTS = 10;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        binding = ActivityResultBinding.inflate(getLayoutInflater());
        setContentView(binding.getRoot());

        int correctAnswers = getIntent().getIntExtra("correct", 0);
        int totalQuestions = getIntent().getIntExtra("total", 0);

        int points = correctAnswers * POINTS;

        binding.score.setText(String.format("%d/%d", correctAnswers,
totalQuestions));

```

```

binding.shareBtn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {

        Intent sharingIntent = new Intent(Intent.ACTION_SEND);
        sharingIntent.setType("text/plain");
        String shareBody = "Hey! I just played Whiz Quiz App and it's
Fantastic!";
        String shareSub = "Your subject here";
        sharingIntent.putExtra(Intent.EXTRA_SUBJECT, shareSub);
        sharingIntent.putExtra(Intent.EXTRA_TEXT, shareBody);
        startActivity(Intent.createChooser(sharingIntent, "Share using"));
    }
});

///RestartButton///
binding.restartBtn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        startActivity(new Intent(ResultActivity.this, MainActivity.class));
    }

});
}
}

```

ForgetPassActivity.java

```
package com.example.myapplication;

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

import android.app.ProgressDialog;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.FirebaseAuth;

public class ForgetPassActivity extends AppCompatActivity {

    ProgressDialog dialog;

    private Button forgetBtn, forgetLoginBtn;
    private EditText txtEmail;
    private String email;
    private FirebaseAuth auth;

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

        dialog = new ProgressDialog(this);
        dialog.setMessage("Wait a moment");

        txtEmail = findViewById(R.id.ForgetemailBox);
        forgetBtn = findViewById(R.id.ForgetBtn);
        forgetLoginBtn = findViewById(R.id.ForgetLoginBtn);

        forgetBtn.setOnClickListener(new View.OnClickListener() {
```

```

@Override
public void onClick(View view) {
    email = txtEmail.getText().toString();

    if (email.isEmpty()){
        Toast.makeText(ForgetPassActivity.this, "Please provide you
email", Toast.LENGTH_SHORT).show();
    }else {
        forgetPassword();
    }

    dialog.show();

}
});

forgetLoginBtn.setOnClickListener(new View.OnClickListener() {
@Override
public void onClick(View view) {
startActivity(new
Intent(ForgetPassActivity.this, LoginActivity.class));
}
});

}

private void forgetPassword() {

    FirebaseAuth auth = FirebaseAuth.getInstance();

    auth.sendPasswordResetEmail(email)
        .addOnCompleteListener(new OnCompleteListener<Void>() {
            @Override
            public void onComplete(@NonNull Task<Void> task) {
                if (task.isSuccessful()){

                    dialog.dismiss();

                    Toast.makeText(ForgetPassActivity.this, "We have sent
you instructions to reset your password", Toast.LENGTH_SHORT).show();
                }
            }
        });
}
}

```

```

        startActivity(new Intent(ForgetPassActivity.this,
LoginActivity.class));
        finish();
    } else {
        dialog.dismiss();
        Toast.makeText(ForgetPassActivity.this, "Error :
"+task.getException().getMessage(), Toast.LENGTH_SHORT).show();
    }
}
});
}
}

```

HomeFragment.java

```

package com.example.myapplication;

import android.os.Bundle;

import androidx.annotation.Nullable;
import androidx.fragment.app.Fragment;
import androidx.recyclerview.widget.GridLayoutManager;

import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

import com.example.myapplication.databinding.FragmentHomeBinding;
import com.google.firebase.firestore.DocumentSnapshot;
import com.google.firebase.firestore.EventListener;
import com.google.firebase.firestore.FirebaseFirestore;
import com.google.firebase.firestore.FirebaseFirestoreException;
import com.google.firebase.firestore.QuerySnapshot;

import java.util.ArrayList;

public class HomeFragment extends Fragment {

    public HomeFragment() {
        // Required empty public constructor
    }
}

```

```

}

@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

}

FragmentHomeBinding binding;
FirebaseFirestore database;

@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
    Bundle savedInstanceState) {

    binding = FragmentHomeBinding.inflate(inflater, container, false);

    database = FirebaseFirestore.getInstance();

    ////Category list////
    final ArrayList<CategoryModel> categories = new ArrayList<>();

    final CategoryAdapter adapter = new CategoryAdapter(
    getContext(),categories);

    database.collection("categories")
        .addSnapshotListener(new EventListener<QuerySnapshot>() {
            @Override
            public void onEvent(@Nullable QuerySnapshot value,
            @Nullable FirebaseFirestoreException error) {
                categories.clear();
                for (DocumentSnapshot snapshot: value.getDocuments()) {
                    CategoryModel model =
                    snapshot.toObject(CategoryModel.class);
                    model.setCategoryId(snapshot.getId());
                    categories.add(model);
                }
                adapter.notifyDataSetChanged();
            }
        });
}

```

```

        }
    });

    binding.categoryList.setLayoutManager(new
    GridLayoutManager(getContext(), 2));
    binding.categoryList.setAdapter(adapter);

    // Inflate the layout for this fragment
    return binding.getRoot();
}
}

```

ProfileFragment.java

```

package com.example.myapplication;

import android.app.ProgressDialog;
import android.os.Bundle;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.fragment.app.Fragment;

import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import com.google.android.gms.tasks.OnFailureListener;
import com.google.android.gms.tasks.OnSuccessListener;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.firestore.DocumentReference;
import com.google.firebase.firestore.DocumentSnapshot;
import com.google.firebase.firestore.EventListener;
import com.google.firebase.firestore.FirebaseFirestore;
import com.google.firebase.firestore.FirebaseFirestoreException;

```

```

import java.util.HashMap;
import java.util.Map;

public class ProfileFragment extends Fragment {

    EditText profile_name, profile_email, profile_password;
    FirebaseAuth fAuth;
    FirebaseFirestore fStore;
    String userID;
    Button updateProfileBtn;
    FirebaseUser user;
    ProgressDialog dialog;

    public ProfileFragment() {
        // Required empty public constructor
    }

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        // Inflate the layout for this fragment
        View view = inflater.inflate(R.layout.fragment_profile, container,
false);

        profile_name = view.findViewById(R.id.nameBox2);
        profile_email = view.findViewById(R.id.emailBox2);
        profile_password = view.findViewById(R.id.passBox);
        updateProfileBtn = view.findViewById(R.id.updateProfile_Btn);

        fAuth = FirebaseAuth.getInstance();
        fStore = FirebaseFirestore.getInstance();

```



```

user = FirebaseAuth.getCurrentUser();

userID = FirebaseAuth.getCurrentUser().getUid();

dialog = new ProgressDialog(getActivity());
dialog.setMessage("Updating...");

DocumentReference documentReference =
fStore.collection("users").document(userID);
documentReference.addSnapshotListener( new
EventListener<DocumentSnapshot>() {
    @Override
    public void onEvent(@Nullable DocumentSnapshot
documentSnapshot, @Nullable FirebaseFirestoreException e) {
        profile_name.setText(documentSnapshot.getString("name"));
        profile_email.setText(documentSnapshot.getString("email"));
        profile_password.setText(documentSnapshot.getString("pass"));
    }
});

updateProfileBtn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        if (profile_name.getText().toString().isEmpty() ||
profile_email.getText().toString().isEmpty()){
            Toast.makeText(getActivity(), "Please fill the fields",
Toast.LENGTH_SHORT).show();
            return;
        }

        String email = profile_email.getText().toString();
        user.updateEmail(email).addOnSuccessListener(new
OnSuccessListener<Void>() {
            @Override
            public void onSuccess(Void unused) {
                DocumentReference docRef =
fStore.collection("users").document(user.getUid());
                Map<String, Object> edited = new HashMap<>();
                edited.put("email", email);
            }
        });
    }
});

```

```

        edited.put("name", profile_name.getText().toString());
        edited.put("pass", profile_password.getText().toString());
        dialog.show();
        docRef.update(edited).addOnSuccessListener(new
OnSuccessListener<Void>() {
            @Override
            public void onSuccess(Void unused) {
                dialog.dismiss();
                Toast.makeText(getActivity(), "Profile updated",
Toast.LENGTH_SHORT).show();
            }
        });
        //Toast.makeText(getActivity(), "Email is changed",
Toast.LENGTH_SHORT).show();

    }
    }).addOnFailureListener(new OnFailureListener() {
        @Override
        public void onFailure(@NonNull Exception e) {
            Toast.makeText(getActivity(), e.getMessage(),
Toast.LENGTH_SHORT).show();

        }
    });

    }
});

return view;
}
}

```

INPUT AND OUTPUT SCREEN:

Registration –



8:10 66%

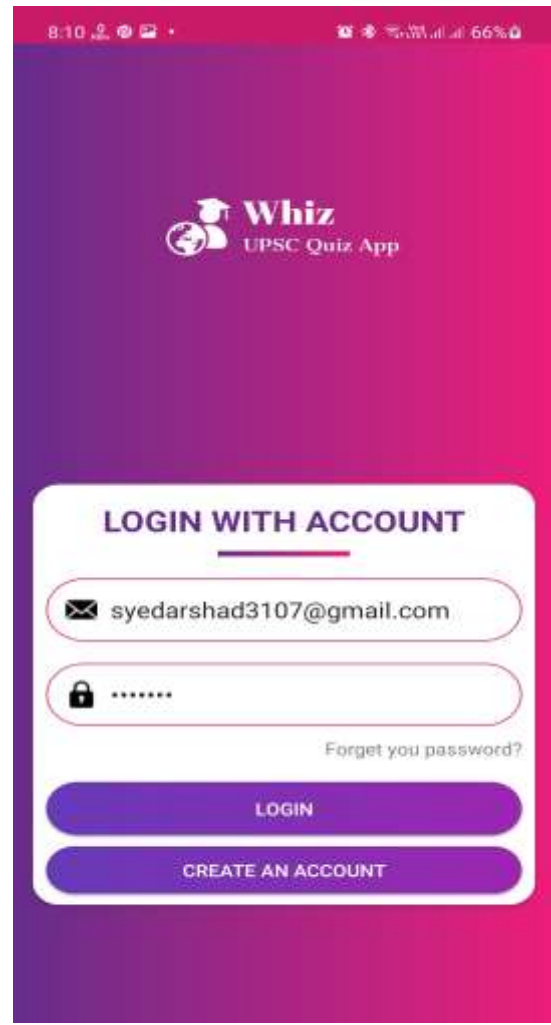
Whiz
UPSC Quiz App

CREATE AN ACCOUNT

SUBMIT

ALREADY HAVE AN ACCOUNT?

Login –



8:10 66%

Whiz
UPSC Quiz App

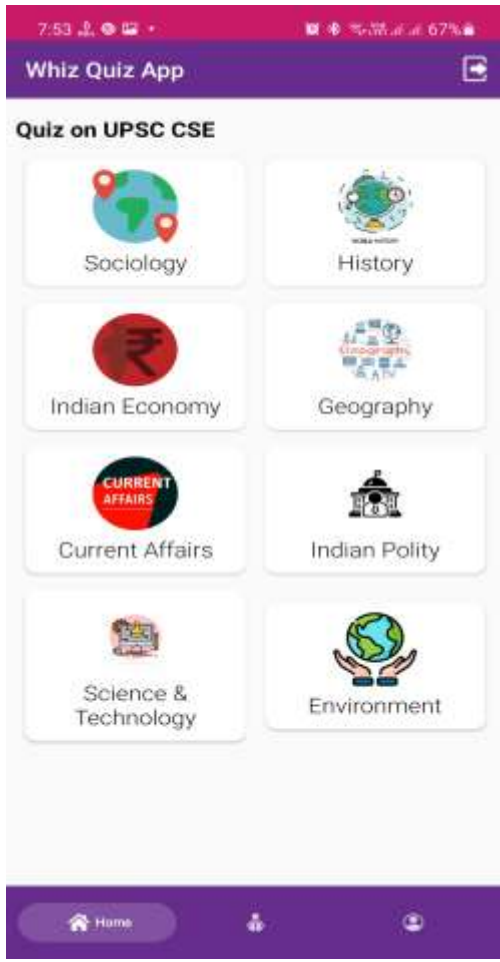
LOGIN WITH ACCOUNT

[Forget you password?](#)

LOGIN

CREATE AN ACCOUNT

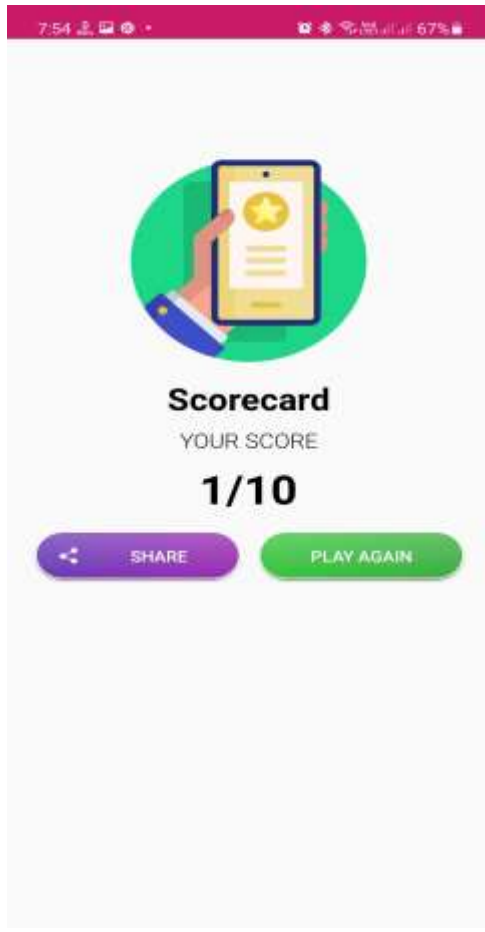
Home –



Quiz –



Result –



Profile –



TESTING & VALIDATION CHECKS

TESTING AND VALIDATION CHECK:

Validation testing in software engineering is in place to determine if the existing system complies with the system requirements and performs the dedicated functions for which it is designed along with meeting the goals and needs of the organization.

The process of evaluating software during the development process or at the end of the development process to determine whether it satisfies specified business requirements.

Validation Testing ensures that the product actually meets the client's needs. It can also be defined as to demonstrate that the product fulfills its intended use when deployed on appropriate environment.

Whenever any particular software is tested then the main motive is to check the quality against the defects being found.

The developers fix the bugs and the software is rechecked to make sure that absolutely no bugs are left out in that. This not only shoots the product's quality but also its user acceptance.

- To ensure customer satisfaction
- To be confident about the product
- To fulfil the client's requirement until the optimum capacity
- Software acceptance from the end-user

Client-side validation is an initial check and an important feature of good user experience; by catching invalid data on the client-side, the user can fix it straight away. If it gets to the server and is then rejected, a noticeable delay is caused by a round trip to the server and then back to the client-side to tell the user to fix their data.

SYSTEM SECURITY MEASURES

SYSTEM SECURITY MEASURES:

Security of a computer system is a crucial task. It is a process of ensuring confidentiality A system is said to be secure if its resources are used and accessed as intended under all the circumstances, but no system can guarantee absolute security from several of the various threats and unauthorized access.

Security measures will be taken:

- **Strong passwords:**
This first measure is taken that users may used proper format in their passwords and password length must be 6 characters.
- **Confidentiality:**
If any users is sharing their personal details in login form it will be secure safely as only users can access such information.

**IMPLEMENTATION, EVALUATION
AND MAINTAINANCE**

IMPLEMENTATION, EVALUATION & MAINTAINANCE:

Implementation is a process of ensuring that the information system is operational. It involves –

Constructing a new system from scratch.

Constructing a new system from the existing one.

Implementation allows the users to take over its operation for use and evaluation. It involves training the users to handle the system and plan for a smooth conversion.

System Maintenance:

Maintenance means restoring something to its original conditions. Enhancement means adding, modifying the code to support the changes in the user specification. System maintenance conforms the system to its original requirements and enhancement adds to system capability by incorporating new requirements. Thus, maintenance changes the existing system, enhancement adds features to the existing system, and development replaces the existing system. It is an important part of system development that includes the activities which corrects errors in system design and implementation, updates the documents, and tests the data.

Maintenance Types:

System maintenance can be classified into three types – Corrective Maintenance – Enables user to carry out the repairing and correcting leftover problems.

Adaptive Maintenance – Enables user to replace the functions of the programs. Perfective Maintenance – Enables user to modify or enhance the programs according to the users' requirements and changing needs.

Post-Implementation Evaluation Review (PIER):

PIER is a tool or standard approach for evaluating the outcome of the project and determine whether the project is producing the expected benefits to the processes, products or services. It enables the user to verify that the project or system has achieved its desired outcome within specified time period and planned cost. PIER ensures that the project has met its goals by evaluating the development and management processes of the project.

FUTURE SCOPE OF THE PROJECT

FUTURE SCOPE OF THE PROJECT:

More Content

At present the content of this app is too less, So we can add more features like videos, documents, test series, discussion section and more courses. This features will help student to gain more knowledge.

Monetize

Monetization via the Paid model is very simple. We have to just upload our app to Google, set the price, select the regions and publish the app.

Providing more security

In future this app can be update with more secured as any users personal information cannot be hacked or also any users cannot be hesitate to access or visit the app as more security features will be developed in updated version.

Global Reach

In future this app can be made globally access we can upload on Google play store so user can download from there.

Additional features

Efforts can be made in future for adding and developing app with additional features such as video facilities related to good deeds or work made by public

CONCLUSION

CONCLUSION:

Whiz Quiz App is Android based platform created for educational purpose and development of this app is mainly required by students and learners to prepare themselves for UPSC examinations directly through smartphones and tablets in hands. One of the major goal of our project is to facilitate best quiz app to judge their knowledge and educational improvements about UPSC Examination. The main goal is to enable user to practice for subjective tests. It provides facility to solve quiz anywhere and every time. User can take the test his/her choice regarding the topic. User can register, log-in, and solve the quiz with his/her specific I'd, and can see the scores as well

The students must register his/her name along with all information required and enter the email id and password for the login process. Using this email id and password to log in to the Whiz Quiz App. As soon as the student chooses the quiz topic, each topic contain 10 questions each, the questions with four choices will be shown. The students must choose any option as the answer. After submitting the answer if it is turn into Green color it means your answer is right and if it is turn into red you give wrong answer after completing the quiz you know the score .

BIBLIOGRAPHY AND REFERENCE

BIBLIOGRAPHY AND REFERENCE:

While developing this project internet was the eternal support.
Following are the websites referred by us which helped us in developing our project:

- ❖ www.developer.android.com
- ❖ www.stackoverflow.com
- ❖ www.github.com
- ❖ www.getbootstrap.com

A
PROJECT SYNOPSIS
ON

“Whiz: UPSC Quiz App”

Submitted to

**G. S. COLLEGE OF COMMERCE & ECONOMICS, NAGPUR
AUTONOMOUS
In the Partial Fulfillment of**

B.Com. (Computer Application) Final Year

Synopsis Submitted by
Syed Arshad Hussain
Mustafa Hussaini

Under the Guidance of
Pravin J. Yadao



**G. S. COLLEGE OF COMMERCE & ECONOMICS, NAGPUR
AUTONOMOUS
2021-2022**

1. Introduction: (Write 4 to 5 lines)

Whiz: UPSC Quiz App is an android application that will be developed by Java programming language, Android Studio and Google Firebase this app will cover competitive examinations like Union Public Service Commission (UPSC) it will help students who are preparing for this examination. In this app, there will be questions with four answer options and questions will be on Indian Polity, Geography, History, Indian Economy, Science and Technology, Environment and Ecology, International Relations and many mores.

2. Objectives of the project: (Write only 5 points)

1. This app is designed to help a student prepare for competitive examinations like UPSC
2. There will be questions like MCQs and users will have to select correct answers from the given options.
3. It will design in such a way that it covers all topic questions including previous year questions from the examination.
4. Included Login and Sign Up pages, User can login with email address and password.
5. The timer is the most exciting feature where the user can solve questions in a given time.

3. Project Category: Android Application

4. Tools/ Platform/ Languages to be used:

Programming Language : Java
Software used : Android Studio
Operating System : Windows 11
Data Base : Google Firebase

5. Scope of future application: (Write 4 to 5 points)

1. We can upgrade to all in one exam Quiz app, this app will design to help a student prepare for competitive examinations like UPSC, MPSC, SSC and BANK in a single app.
2. The scoreboard will help students to compare scores with other users of the app.
3. Invite friends and get rewarded.
4. Can be published on the play store.

Submitted by,

Syed Arshad Hussain

Mustafa Hussaini

Approved by,

**Prof. Pravin Yadao
Project Guide**