**SSN of Engineering, Kalavakkam – 603 110**

**Department of Computer Science and Engineering**

**Practical End Semester Examinations**

**Report generation completed**

**Name :** Rahul Ram M

**Reg No:** 185001121

**Class:** CSE -B

**Semester :** VI

**Date:** 06/05/2021

**Student Mark Analysis System**

**1) Problem Statement:**

For analyzing the marks obtained by the student in an educational institution, an app is built which will perform the mark analysis of each student.

Marks that are entered and processed manually are prone to errors and miscalculations. Changing errors in a record will lead to further changes and the process can be more tedious. Storing these data(records) is very hard and prone to damage. Transferring data from one record to another and performing further calculations may result in loss of data and errors. Retrieving data from these records is difficult.

This errors can be minimized with the help of this application. This application uses firebase as its backend. The student details along with mark obtained are stored in firebase in the form collection relations. In order to use this application we must register first. This application registration allows only authorized emails to ensure no other except the faculty can change the details of the student. After signing in, the faculty can edit the add student details to the database with the help of GUI integrated within the application in the form of form fields. After entering the marks of the student, his grade will automatically be displayed and stored in the database. Since this data is stored in google firebase server it can be accessed from anywhere merely with the help of the applications. Further this data can be easily retrieved, transferred, converted and performing calculations will also be easy.

**2) classes:**

|  |  |
| --- | --- |
| **Conceptual Class Category** | **Class** |
| Transactions | addStudent, deleteStudent, updateStudent |
| roles of people | ModifyDetails |
| organizations | universityMails |
| Events | Login, Register, Modify |
| records | studentDetails |
| Documents | Report, markSheet |
| places | University |

**Association Category List:**

New student is added to Database.

Student is removed from database.

Faculty modify student record.

Faculty add new student.

Many students in one class

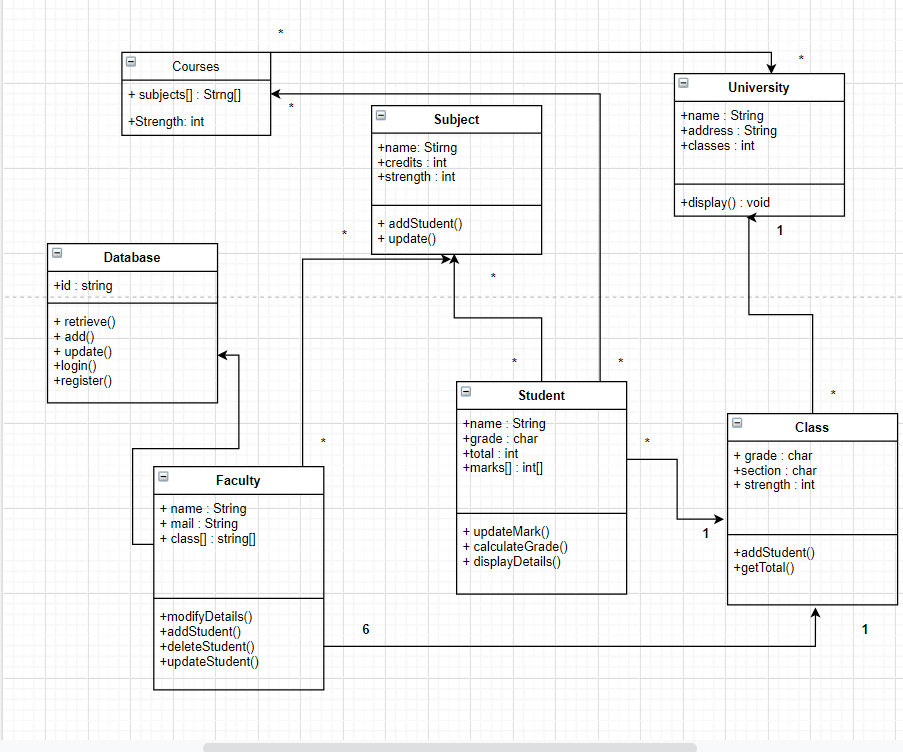
Many classes in a university

One teacher is assigned to many class

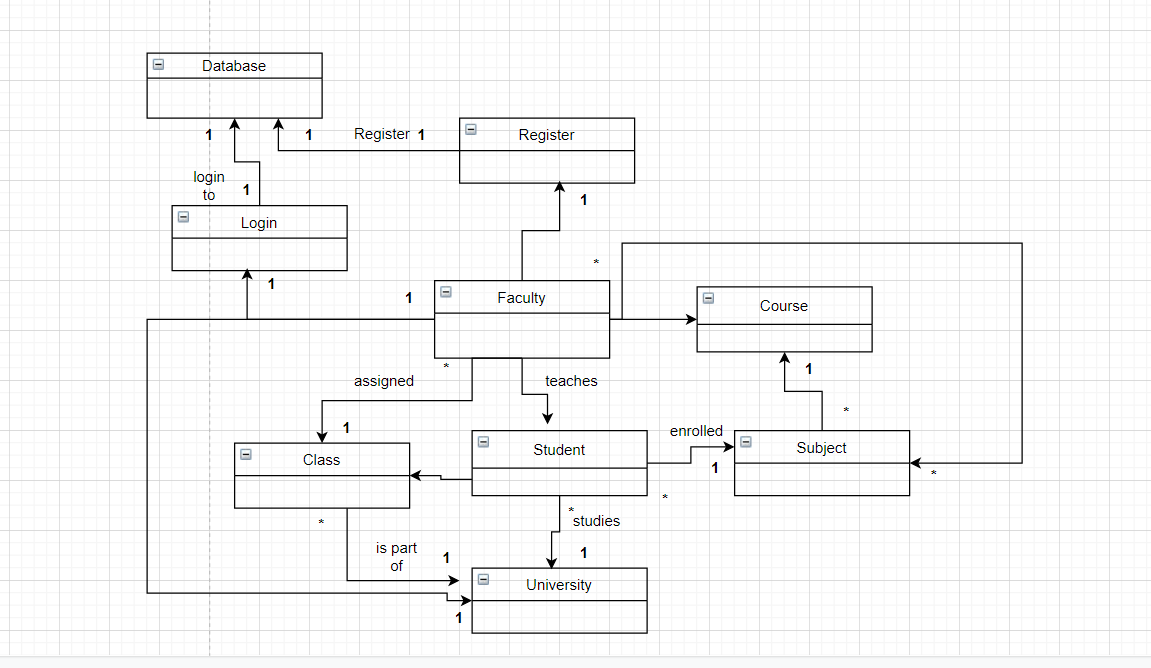
Six teachers is assigned to a single class

**3) Class diagram:**

**Class diagram with association:**



**Domain Model Diagram:**



**Test Cases:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case Id** | **Test Scenario** | **Testing steps** | **Test Cases** | **Expected Outcomes** | **Actual Outcome** | **Pass/Fail** |
| T101 | Login user | 1. Select Signin  2.Enter email and password  2. email and password validity | email: rahul123@gmail.com  pasword: rahul121  Database:  email: rahul123@gmail.com  Password: rahul121 | Login successful. Redirecting to Home page | Login successful.  Redirected to Home page. | Pass |
| T102 | Login user | 1. Select Signin  2.Enter email and password  2. UserId and password validity | email: rahul567@gmail.com  pasword: rahul121  Database:  Username: no entry | Login failed. Stay in login page with error message. | Login failed. Stay in login page with error message. | pass |
| T103 | Login user | 1. Select Signin  2.Enter email and password  2. email and password validity | email: rahul123@gmail.com  pasword: rahul121  Database:  email: rahul123@gmail.com  Password: rahul121 | Login successful. Redirecting to Home page | Login successful.  Redirected failed. | Fail |
| T104 | Register user | 1. Select Register  2. Enter email and password  3. Validate | email: rahul123@gmail.com  pass: raghu120  Database:  email: noentry | Register successful.  Redirect to login page. | Register successful.  Redirect to login page. | Pass |
| T105 | Register user | 1. Select Register  2. Enter email and password  3. Validate | email:  rahul123@gmail.com  pass: raghu120  email:  username: Raghu120 | Register unsuccessful.  Redirect to register page with error message. | Register unsuccessful.  Redirect to register page with error message | pass |
| T106 | Register User | 1. Select Register  2. Enter email and password  3. Validate | email: rahul567@gmail.com  pass: raghu120  Database:  email: Raghu120 | Register unsuccessful.  Redirect to register page with error message. | Register successful.  Redirect to login page. | Fail |
| T107 | AddStudent | 1. enter details  2. press add | regno : 101  database:  regno : 101 | Add student: unsuccessful  Failed. | Add student: unsuccessful  Failed. | pass |
| T107 | AddStudent | 1. enter details  2. press add | regno : 101  database:  regno : no entry | Add student: successful  Db updated. | Add student: successful  Db updated. | pass |

CODE:

Register.dart

import 'package:model\_app/services/auth.dart';  
import 'package:model\_app/shared/constants.dart';  
import 'package:flutter/material.dart';  
import 'package:model\_app/shared/loading.dart';  
  
class Register extends StatefulWidget {  
  
 final Function toggleView;  
 Register({ this.toggleView });  
  
 @override  
 \_RegisterState createState() => \_RegisterState();  
}  
  
class \_RegisterState extends State<Register> {  
  
 final AuthService \_auth = AuthService();  
 final \_formKey = GlobalKey<FormState>();  
 bool loading = false;  
  
 // text field state  
 String email = '';  
 String password = '';  
 String error = '';  
  
 @override  
 Widget build(BuildContext context) {  
 return loading ? Loading() : Scaffold(  
 backgroundColor: Colors.*brown*[100],  
 appBar: AppBar(  
 backgroundColor: Colors.*brown*[400],  
 elevation: 0.0,  
 title: Text('Sign up to Mark Analysis App'),  
 actions: [  
 FlatButton.icon(  
 onPressed: () {  
 widget.toggleView();  
 },  
 icon: Icon(Icons.*person*),  
 label: Text('Sign In'),  
 )  
 ],  
 ),  
 body: Container(  
 padding: EdgeInsets.symmetric(vertical: 20.0, horizontal: 50.0),  
 child: Form(  
 key: \_formKey,  
 child: SingleChildScrollView(  
 child: Column(  
 children: [  
 SizedBox(height: 20.0),  
 TextFormField(  
 decoration: textInputDecoration.copyWith(hintText: 'Email'),  
 validator: (val) => val.isEmpty ? 'Enter an email' : null,  
 onChanged: (val) {  
 setState(() {  
 email = val;  
 });  
 },  
 ),  
 SizedBox(height: 20.0),  
 TextFormField(  
 decoration: textInputDecoration.copyWith(hintText: 'Password'),  
 obscureText: true,  
 validator: (val) => val.length < 6 ? 'Enter a password 6+ characters long' : null,  
 onChanged: (val) {  
 setState(() {  
 password = val;  
 });  
 },  
 ),  
 SizedBox(height: 20.0),  
 RaisedButton(  
 color: Colors.*brown*[400],  
 child: Text(  
 'Register',  
 style: TextStyle(color: Colors.*white*),  
 ),  
 onPressed: () async {  
 if (\_formKey.currentState.validate()) {  
 List<String> mails = ['rahul123@gmail.com', 'rahul@ssn.edu.in'];  
 if (!mails.contains(email)) {  
 setState(() {  
 error = 'Unauthorized email';  
 });  
 }  
 else {  
 setState(() {  
 loading = true;  
 });  
 dynamic result = await \_auth.reigisterWithEmailAndPassword(email, password);  
 if (result == null) {  
 setState(() {  
 error = 'please supply valid email';  
 loading = false;  
 });  
 }  
 }  
 }  
 },  
 ),  
 SizedBox(height: 20.0,),  
 Text(  
 error,  
 style: TextStyle(color: Colors.*red*, fontSize: 14.0),  
 )  
 ],  
 ),  
 ),  
 )  
 ),  
 );  
 }  
}

Signin.dart:

import 'package:model\_app/services/auth.dart';  
import 'package:model\_app/shared/constants.dart';  
import 'package:model\_app/shared/loading.dart';  
import 'package:flutter/material.dart';  
  
class SignIn extends StatefulWidget {  
  
 final Function toggleView;  
 SignIn({ this.toggleView });  
  
 @override  
 \_SignInState createState() => \_SignInState();  
}  
  
class \_SignInState extends State<SignIn> {  
  
 final AuthService \_auth = AuthService();  
 final \_formKey = GlobalKey<FormState>();  
 bool loading = false;  
  
 // text field state  
 String email = '';  
 String password = '';  
 String error = '';  
  
 @override  
 Widget build(BuildContext context) {  
 return loading ? Loading() : Scaffold(  
 backgroundColor: Colors.*brown*[100],  
 appBar: AppBar(  
 backgroundColor: Colors.*brown*[400],  
 elevation: 0.0,  
 title: Text('Sign in to Mark Analysis App'),  
 actions: [  
 FlatButton.icon(  
 onPressed: () {  
 widget.toggleView();  
 },  
 icon: Icon(Icons.*person*),  
 label: Text('Register'),  
 )  
 ],  
 ),  
 body: Container(  
 padding: EdgeInsets.symmetric(vertical: 20.0, horizontal: 50.0),  
 child: Form(  
 key: \_formKey,  
 child: SingleChildScrollView(  
 child: Column(  
 children: [  
 SizedBox(height: 20.0),  
 TextFormField(  
 decoration: textInputDecoration.copyWith(hintText: 'Email'),  
 validator: (val) => val.isEmpty ? 'Enter an email' : null,  
 onChanged: (val) {  
 setState(() {  
 email = val;  
 });  
 },  
 ),  
 SizedBox(height: 20.0),  
 TextFormField(  
 decoration: textInputDecoration.copyWith(hintText: 'Password'),  
 obscureText: true,  
 validator: (val) => val.length < 6 ? 'Enter a password 6+ characters long' : null,  
 onChanged: (val) {  
 setState(() {  
 password = val;  
 });  
 },  
 ),  
 SizedBox(height: 20.0),  
 RaisedButton(  
 color: Colors.*brown*[400],  
 child: Text(  
 'Sign in',  
 style: TextStyle(color: Colors.*white*),  
 ),  
 onPressed: () async {  
 if (\_formKey.currentState.validate()) {  
 setState(() {  
 loading = true;  
 });  
 dynamic result = await \_auth.signInWithEmailAndPassword(email, password);  
 if(result == null) {  
 setState(() {  
 error = 'Could not sign with those credentials';  
 loading = false;  
 });  
 }  
 }  
 },  
 ),  
 SizedBox(height: 20.0,),  
 Text(  
 error,  
 style: TextStyle(color: Colors.*red*, fontSize: 14.0),  
 )  
 ],  
 ),  
 ),  
 )  
 ),  
 );  
 }  
}

Home.dart:

import 'package:model\_app/input/add\_student.dart';  
import 'package:model\_app/models.dart';  
import 'package:model\_app/services/auth.dart';  
import 'package:flutter/material.dart';  
import 'package:model\_app/services/database.dart';  
import 'package:provider/provider.dart';  
  
  
class Home extends StatelessWidget {  
  
 final AuthService \_auth = AuthService();  
  
 @override  
 Widget build(BuildContext context) {  
  
 //final user = Provider.of<FUser>(context);  
  
 return Scaffold(  
 backgroundColor: Colors.*brown*[50],  
 appBar: AppBar(  
 title: Text('Mark Analysis App'),  
 backgroundColor: Colors.*brown*[400],  
 elevation: 0.0,  
 actions: [  
 FlatButton.icon(  
 onPressed: () async {  
 await \_auth.signOut();  
 },  
 icon: Icon(Icons.*person*),  
 label: Text('logout'),  
 ),  
 ],  
 ),  
 body: Container(  
 padding: EdgeInsets.symmetric(vertical: 20.0, horizontal: 150.0),  
 child: Column(  
 children: [  
 ElevatedButton(  
 style: ButtonStyle(  
 backgroundColor: MaterialStateProperty.*all*<Color>(Color(0xFF211F16)),  
 ),  
 onPressed: () async {  
 Navigator.*push*(context,  
 new MaterialPageRoute(  
 builder: (context) {  
 return AddStudent();  
 }  
 )  
 );  
 },  
 child: Text(  
 'Add Student',  
 style: TextStyle(color: Color(0xFFE8CE46)),  
 )  
 ),  
 SizedBox(height: 30.0,),  
 ElevatedButton(  
 style: ButtonStyle(  
 backgroundColor: MaterialStateProperty.*all*<Color>(Color(0xFF211F16)),  
 ),  
 onPressed: () async {  
 Navigator.*push*(context,  
 new MaterialPageRoute(  
 builder: (context) {  
 return StreamProvider<List<StudentData>>.value(  
 value: DatabaseService().studentList,  
 child: Report(),  
 );  
 }  
 )  
 );  
 },  
 child: Text(  
 'View Student',  
 style: TextStyle(color: Color(0xFFE8CE46)),  
 )  
 ),  
 ],  
 ),  
 )  
 );  
 }  
}

AddStudent.dart:

import 'package:flutter/material.dart';  
import 'package:model\_app/services/database.dart';  
import 'package:model\_app/shared/constants.dart';  
import 'package:toast/toast.dart';  
  
class AddStudent extends StatefulWidget {  
 @override  
 \_AddStudentState createState() => \_AddStudentState();  
}  
  
class \_AddStudentState extends State<AddStudent> {  
  
 final \_formKey = GlobalKey<FormState>();  
  
 // input details  
 String name;  
 int regNo;  
 int ip;  
 int ml;  
 int ooad;  
 int fds;  
 int ipr;  
 int cd;  
  
  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 backgroundColor: Color(0xFFB9B5B5),  
 appBar: AppBar(  
 //automaticallyImplyLeading: !widget.isNew,  
 backgroundColor: Color(0xFF211F16),  
 elevation: 0.0,  
 title: Text(  
 'Add Student', style: TextStyle(color: Color(0xFFE8CE46))  
 ),  
 ),  
 body: Container(  
 padding: EdgeInsets.symmetric(vertical: 20.0, horizontal: 50.0),  
 child: Form(  
 key: \_formKey,  
 child: SingleChildScrollView(  
 child: Column(  
 children: [  
 SizedBox(height: 20.0),  
 Align(  
 alignment: Alignment.*centerLeft*,  
 child: Container(  
 child: Text(  
 "Full Name",  
 ),  
 ),  
 ),  
 SizedBox(height: 10.0),  
 TextFormField(  
 decoration: textInputDecoration.copyWith(hintText: 'Name'),  
 validator: (val) => val.length == 0 ? 'Enter valid Name' : null,  
 onChanged: (val) {  
 setState(() {  
 name = val;  
 });  
 },  
 ),  
 SizedBox(height: 20.0),  
  
 TextFormField(  
 decoration: textInputDecoration.copyWith(hintText: 'Register Number'),  
 keyboardType: TextInputType.*number*,  
 validator: (val) => val.length != 3 ? 'Enter valid register number' : null,  
 onChanged: (val) {  
 setState(() {  
 regNo = int.*parse*(val);  
 });  
 },  
 ),  
 SizedBox(height: 20.0),  
 Align(  
 alignment: Alignment.*centerLeft*,  
 child: Container(  
 child: Text(  
 "Enter Marks : ",  
 ),  
 ),  
 ),  
 SizedBox(height: 20.0),  
 TextFormField(  
 decoration: textInputDecoration.copyWith(hintText: 'Internet programming'),  
 keyboardType: TextInputType.*number*,  
 validator: (val) => int.*parse*(val) > 100 || int.*parse*(val) < 0 ? 'Enter valid mark' : null,  
 onChanged: (val) {  
 setState(() {  
 ip = int.*parse*(val);  
 });  
 },  
 ),  
 SizedBox(height: 20.0),  
  
 TextFormField(  
 decoration: textInputDecoration.copyWith(hintText: 'Compiler Design'),  
 keyboardType: TextInputType.*number*,  
 validator: (val) => int.*parse*(val) > 100 || int.*parse*(val) < 0 ? 'Enter valid mark' : null,  
 onChanged: (val) {  
 setState(() {  
 cd = int.*parse*(val);  
 });  
 },  
 ),  
 SizedBox(height: 20.0),  
  
 TextFormField(  
 decoration: textInputDecoration.copyWith(hintText: 'Machine Learning'),  
 keyboardType: TextInputType.*number*,  
 validator: (val) => int.*parse*(val) > 100 || int.*parse*(val) < 0 ? 'Enter valid mark' : null,  
 onChanged: (val) {  
 setState(() {  
 ml = int.*parse*(val);  
 });  
 },  
 ),  
 SizedBox(height: 20.0),  
  
 TextFormField(  
 decoration: textInputDecoration.copyWith(hintText: 'OO Analysis and Design'),  
 keyboardType: TextInputType.*number*,  
 validator: (val) => int.*parse*(val) > 100 || int.*parse*(val) < 0 ? 'Enter valid mark' : null,  
 onChanged: (val) {  
 setState(() {  
 ooad = int.*parse*(val);  
 });  
 },  
 ),  
 SizedBox(height: 20.0),  
  
 TextFormField(  
 decoration: textInputDecoration.copyWith(hintText: 'Data Science'),  
 keyboardType: TextInputType.*number*,  
 validator: (val) => int.*parse*(val) > 100 || int.*parse*(val) < 0 ? 'Enter valid mark' : null,  
 onChanged: (val) {  
 setState(() {  
 fds = int.*parse*(val);  
 });  
 },  
 ),  
 SizedBox(height: 20.0),  
  
 TextFormField(  
 decoration: textInputDecoration.copyWith(hintText: 'IPR'),  
 keyboardType: TextInputType.*number*,  
 validator: (val) => int.*parse*(val) > 100 || int.*parse*(val) < 0 ? 'Enter valid mark' : null,  
 onChanged: (val) {  
 setState(() {  
 ipr = int.*parse*(val);  
 });  
 },  
 ),  
 SizedBox(height: 20.0),  
 ElevatedButton(  
 style: ButtonStyle(  
 backgroundColor: MaterialStateProperty.*all*<Color>(Color(0xFF211F16)),  
 ),  
 onPressed: () async {  
 print(name);  
 if(\_formKey.currentState.validate()) {  
 await DatabaseService().updateStudentData(name, regNo, ip, cd, ml, ooad, fds, ipr);  
 }  
 Toast.*show*(  
 "Student Details Added",  
 context,  
 duration: 3,  
 gravity: Toast.*BOTTOM*  
);  
 Navigator.*pop*(context);  
 },  
 child: Text(  
 'Add',  
 style: TextStyle(color: Color(0xFFE8CE46)),  
 )  
 ),  
 ],  
 ),  
 ),  
 )  
 ),  
 );  
 }  
}

database.dart:

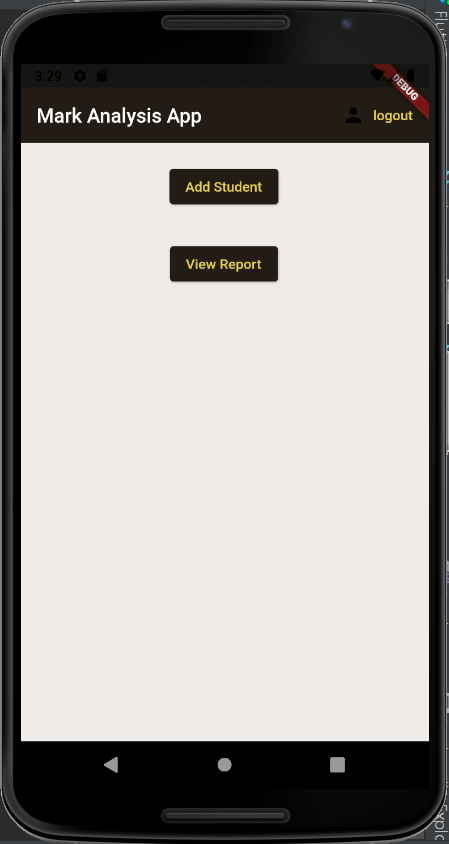
import 'package:model\_app/models.dart';  
import 'package:cloud\_firestore/cloud\_firestore.dart';  
import 'package:firebase\_auth/firebase\_auth.dart';  
  
class DatabaseService {  
  
 // collection reference  
 // once created - won't create another just give the reference  
 final CollectionReference studentCollection = FirebaseFirestore.*instance*.collection('students');  
  
 Future updateStudentData(String name, int regNo, int ip, int cd, int ml, int ooad, int fds, int ipr) async {  
 return await studentCollection.doc(regNo.toString()).set({  
 'name' : name,  
 'reg\_no' : regNo,  
 'ip' : ip,  
 'cd' : cd,  
 'ml' : ml,  
 'ooad' : ooad,  
 'fds' : fds,  
 'ipr' : ipr,  
 });  
 }  
  
 List<StudentData> \_studentListFromSnapshot(QuerySnapshot snapshots) {  
 return snapshots.docs.map((doc) {  
 return \_studentDataFromSnapshot(doc);  
 }).toList();  
 }  
  
 // userData from snapshots  
 StudentData \_studentDataFromSnapshot(DocumentSnapshot snapshot) {  
 return StudentData(  
 name: snapshot.data()['name'],  
 regNo : snapshot.data()['regNo'],  
 ip : snapshot.data()['ip'],  
 cd : snapshot.data()['cd'],  
 ml : snapshot.data()['ml'],  
 ooad : snapshot.data()['ooad'],  
 fds : snapshot.data()['fds'],  
 ipr: snapshot.data()['ipr'],  
 );  
 }  
  
 // get user doc stream  
 Stream<List<StudentData>> get studentList {  
 return studentCollection.snapshots().map( \_studentListFromSnapshot);  
 }  
  
}

auth.dart:

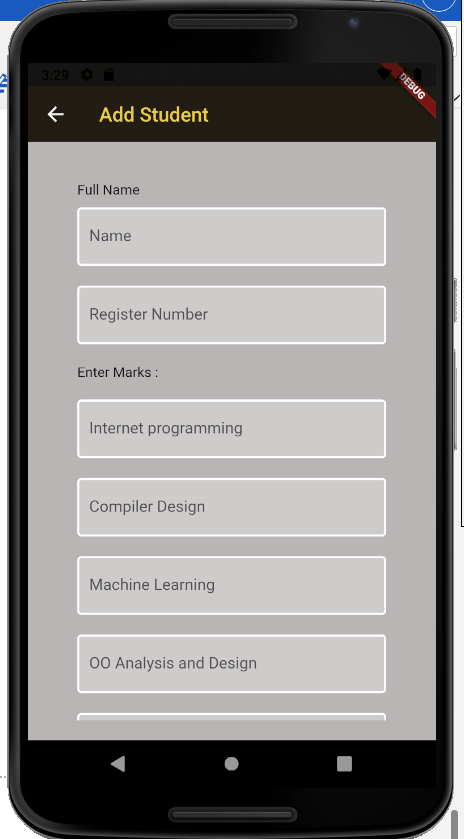
import 'package:model\_app/models.dart';  
import 'package:model\_app/services/database.dart';  
import 'package:firebase\_auth/firebase\_auth.dart';  
  
class AuthService {  
  
 final FirebaseAuth \_auth = FirebaseAuth.*instance*;  
  
 // create FUser obj based on User  
 FUser \_userFromUser(User user ) {  
 return user != null ? FUser(uid: user.uid) : null;  
 }  
  
 // auth change user stream  
 Stream<FUser> get user {  
 //return \_auth.authStateChanges().map(\_userFromUser);  
 return \_auth.authStateChanges().map((User user) => \_userFromUser(user));  
 }  
  
 // sign in anon  
 Future signInAnon() async {  
 try {  
 UserCredential userCredential = await \_auth.signInAnonymously();  
 User user = userCredential.user;  
 return \_userFromUser(user);  
 } catch(e) {  
 print(e.toString());  
 return null;  
 }  
 }  
  
 // sign in with email and password  
 Future signInWithEmailAndPassword(String email, String password) async {  
 try {  
 UserCredential userCredential = await \_auth.signInWithEmailAndPassword(email: email, password: password);  
 User user = userCredential.user;  
 return \_userFromUser(user);  
 } catch(e) {  
 print(e.toString());  
 return null;  
 }  
 }  
  
 // register with email and password  
 Future reigisterWithEmailAndPassword(String email, String password) async {  
 try {  
 UserCredential userCredential = await \_auth.createUserWithEmailAndPassword(email: email, password: password);  
 User user = userCredential.user;  
  
 // create a new document for the user with the uid  
 //await DatabaseService(uid: user.uid).updateUserData('0', 'new crew member', 100);  
 return \_userFromUser(user);  
 } catch(e) {  
 print(e.toString());  
 return null;  
 }  
 }  
  
 // sign out  
 Future signOut() async {  
 try{  
 return await \_auth.signOut();  
 } catch(e) {  
 print(e.toString());  
 return null;  
 }  
 }  
  
}

**Screenshots:**

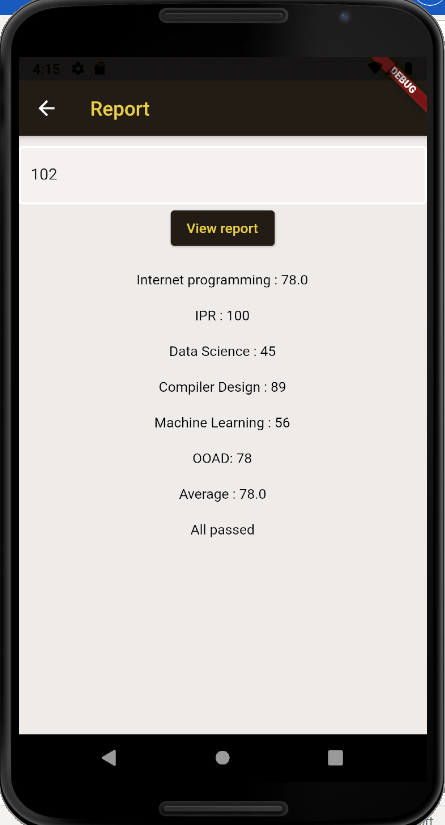
**Home page:**



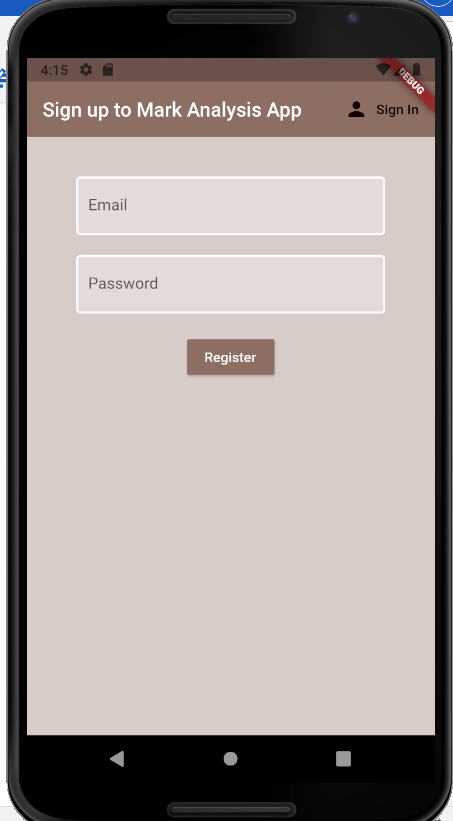
**Add Student:**



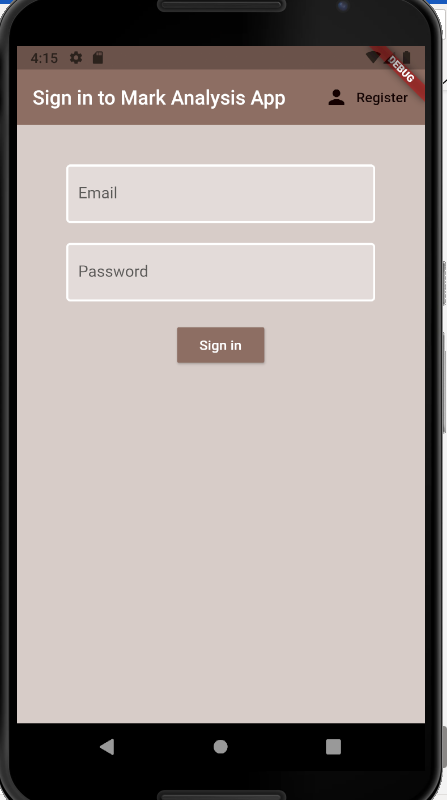
**report:**



Login in:



Signin:



**DataBase:**

