Group Project Description Title:

Quizzie

Course Number:	CSE-5324
Group Name:	Texas Stars
Group Number:	08

Names of group members:

Shrutikirti Saxena (C)

Sai Charan Thannir

Sai Venkata Bhanu Shasank Bonthala

Bhuvan Poola

Sayali Kamble

Change Log:

Change Log.xlsx

	Project Change Log		
Release #	**Change Description**	**Date**	**Affected Page(s)**
Iteration 1.2	Added line numbers to the functionalities.	3/1/2020	5-6, Functionalities
	Added function names instead of "page" or "tab" in functionalities.		5-6, Functionalities
	Changed the requirements as problem oriented instead of solution oriented.		8,Requirements
	Edited and added Non-Functional requirements		8,Requirements
	Added the source for some requirements from requirements feedback document.		8,Requirements
	Changed High Level Use Cases and TUCBW and TUCEW statements for all of them.		12,High Level Use Cases
	Corrected the names of some High Level Use Cases (UC 2, 2.1.1, 2.1.2, 2.1.3, 3.1 etc).		12,High Level Use Cases
	Changed the subsystems in Use Case Diagrams.		9-11, Use Case Diagrams
	Changed the tree structure of some of the Use Case Diagrams.		9-11, Use Case Diagrams
	Changed ambiguous mapping caused by non-atomic use cases being mapped in RUTM.		14, Requirements Use Case Traceability Matrix
	Made sure that each requirement is mapped to an Use Case and vice versa.		14, Requirements Use Case Traceability Matrix
	Edited multiplicity, missing arrows, associations, classes, attributes and changed Domain Model diagram according to that.		16, Domain Model
Iteration 1.3	Changed the TUCBW statement of UC 2 in High Level Use Cases.	4/5/2020	12,High Level Use Cases
	Changed the name of UC 9 to : *View Number of Tests* in High Level Use Cases to avoid ambiguity between UC 6 and UC 9.		12, High Level Use Cases
	Changed the TUCEW statement of UC 12 in HLUC.		12, High Level Use Cases
	Corrected the Use Case Diagram to maintain uniformity in the names of *View About App* Use Cases.		9-11, Use Case Diagrams
	Mapped the requirements R1.1, R1.2, R1.3 against Use Cases in RUTM.		14, Requirements Use Case Traceability Matrix
	Added student_id and quiz_id as attributes in Performance association class in the Domain model.		16, Domain Model
	Removed aggregate Grade and Progress Status as attributes of performance class in the Domain model.		16, Domain Model
	Added feedback class in the domain Model.		16,Domain Model
	Added UC development tasks in the Increment Matrix.		15, Increment Matrix
Iteration 2	Changed the TUCBW statement of UC 2 ; made it less complex by removing attributes of requirement 1.3.	4/23/2020	12,High Level Use Cases
	Added user_id and feedback_id in the Domain Model.		16, Domain Model
	Added develeopment related tasks for UC 7,10,11,12 in the Task Assigned list.		17,Task Assigned list
	Changed the name of "System" as the name of the app in Expanded Use Cases.		18-30, Expanded Use Cases
	Marked all the non-trivial steps in Expanded Use Cases.		18-30,Expanded Use Cases
	Changed the layout of the UI design of UC 4,5,6.		21-23, Expanded Use Cases
	Added reference UI prototypes in UC 8,9,13.		25-26,30,Expanded Use Cases
	Changed Sequence diagrams for UC 4, UC 5.		31-32, Sequence Diagrams
	Updated the Design Class Diagram.		40,Design Class diagram

Table of Contents

DESCRIPTION:

We are trying to develop a mobile quiz application. It will help students to enhance their knowledge of a subject. It will help them trace their progress by keeping a track of the tests they take, along with immediate feedback regarding their quiz grades. The main goal of this application is to facilitate students in learning, gaining and improving their skills through constant practice and evaluation.

By using datasets, we can get multiple choice questions get displayed to the user and the response given by the user is recorded and calculated to provide a grade. The user is free to choose from “easy”, “medium” and “hard” difficulty levels of questions. Each difficulty level will have a set of quiz. Every quiz will have timer.

This app has its own grading system. User can easily get to know the grades of the previous quiz. We are also trying to implement a functionality where the student can retake the quiz (any number of times) if he/she has scored 50% or less than 50% grade in a particular quiz.

User mostly interacts with the application with swipe or click modules. This application even has a timer, so that the user can know how much time the user spends in each question.

FUNCTIONALITIES:

Function 1: Login

The first page whenever user opens the application. If the user already has an account, can login or register if he/she is new to the application. Login page also has a *forgot password* option, where the user can regain access to his/her account.

Function 2: Register

It enables the user to create his/her account in the quizzie application using his/her student ID, First Name, Last Name, Email ID, Phone number, Gender, Age and Password.

Function 3: Home Screen

It displays after login and has main functions of the applications such as i) Quiz details ii) Profile iii) Progress status iv) Settings

Function 4: Quiz details

This option would take us to another page from where user can choose options as: new quiz, retake quiz and view quiz.

Function 5: New Quiz

This option will direct us to a page where the user can take a new quiz and choose the difficulty levels of the questions from easy, medium and hard.

Function 6: Retake Quiz

This option would allow us to retake a quiz depending on the grades of the user (If user has less than or equal to 50% grades in a particular quiz, he/she is allowed to retake that quiz)

Function 7: View Answers

This option would allow the user to view all the past quizzes and their details.

Function 8: Profile

This option would allow the user to view his/her information, specifically: name, email ID.

Function 9: Settings

This option would allow us to get access to options such as Contact Us, Report, Version, About App and would also allow the user to logout.

RESOURCES:

Data sets.

Wi-Fi

****We have changed functionality for profile details which included progress status. We have appended the progress status from profile details into the home page details ****

EXPERIENCE:

Sai Charan Thannir: I'm a certified HADOOP Developer (HDPCD). I worked as an Associate Engineer in Big Data domain for one year. During that period, I worked on various tasks which include data ingestion, skipping validation, merging data from different sources and on enhancements that use different Big Data technologies and tools like Hadoop MapReduce, Hive, Pig, Sqoop, Flume, Kafka and Spark with Java. I'm proficient in Java and has substantial knowledge in various scripting languages and database languages. I have also worked on python project during my bachelors. I am new to Android Development.

Sai Venkata Bhanu Shasank Bonthala: I worked on android application in my internship. I used the iconic framework to build the application. The other resources I used at that moment were, node js, nosql- firebase, HTML5, ts- transcript. The application was called ICRF. It's used to drop petitions regarding any problems that occurs in the society, and when anyone posts the petition, the other users support them by supporting them, they send a message to the respective municipal members regarding the issue. The application even got a national award. During my mini project I worked on a website which was called the Online Local Train Booking System. In that website we provide the customers the ease to book tickets online. The resources I utilized during the project was HTML, PHP, XAMPP server.

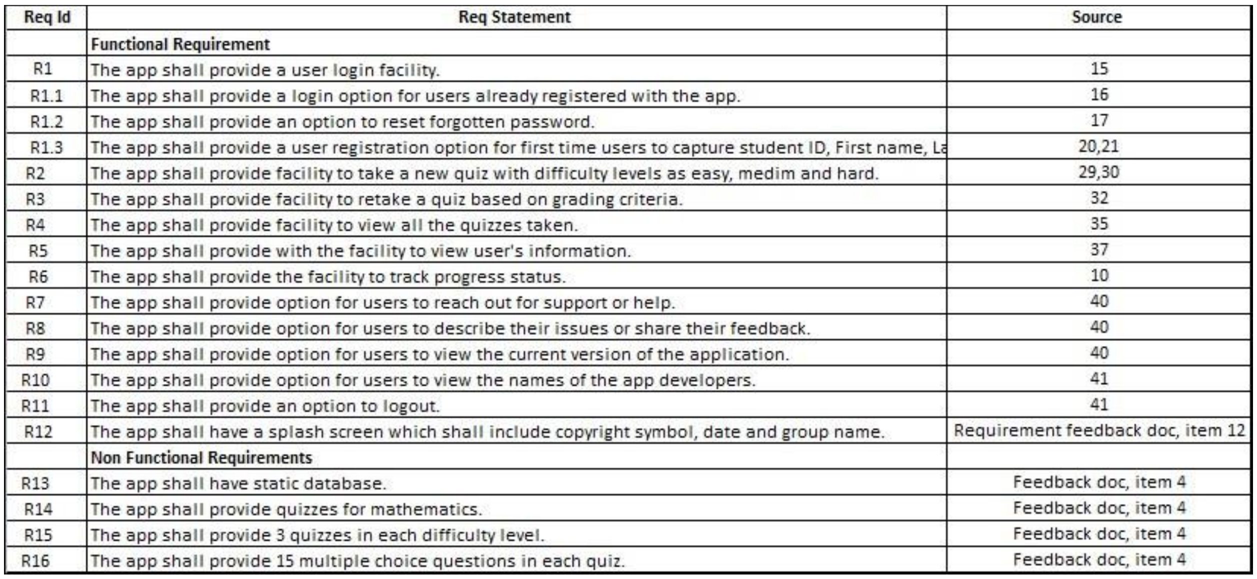
Bhuvan Poola: I have done my major project on Hadoop database management system. Have good knowledge in Hadoop libraries and R programming. Can cope up with Android studio and designing of the application. Made a search engine based on Hadoop file system which is textbased search, implies the result in Jason format.

Sayali Kamble: I am novice towards Android development. Although, I have previous work experience of almost 2 years. During this period, I have had exposure towards Oracle products such as FMW and Oracle Data Integrator. Also, I have worked on scripting language such as Linux.

Shrutikirti Saxena: I have a background in Electronics and Communications Engineering. I have a hands-on experience of programming microcontrollers using embedded C, worked on Internet of Things, implementing basic concepts of Json, MongoDB, REST web services, GPRS and HTTP. During my tenure as a bachelor's student, I worked as a project trainee in Indian Space Research Organization, ISRO, Ahmedabad, India where I got the opportunity to gain hands on working experience of MATLAB prototyping and VLSI Digital System Design. I was also able to gain knowledge of implementation of java in android studio and developed a very basic mobile application, demonstrating the concepts of activities, services, broadcast receivers, content providers, etc. as an intern in L&T Infotech, Mumbai, India during my 3rd year of bachelor's course.

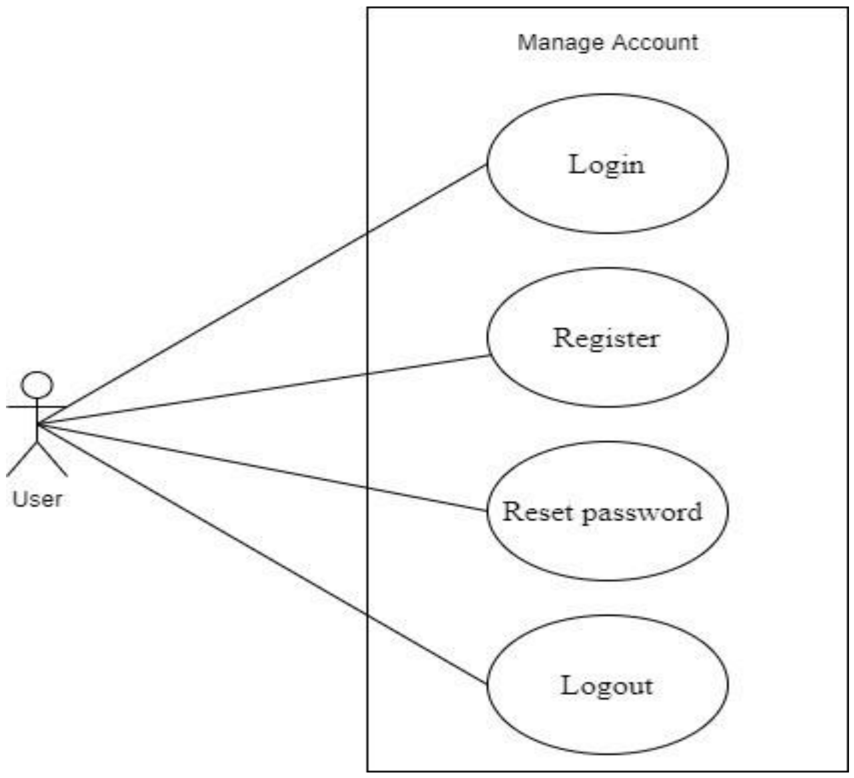
REQUIREMENTS:

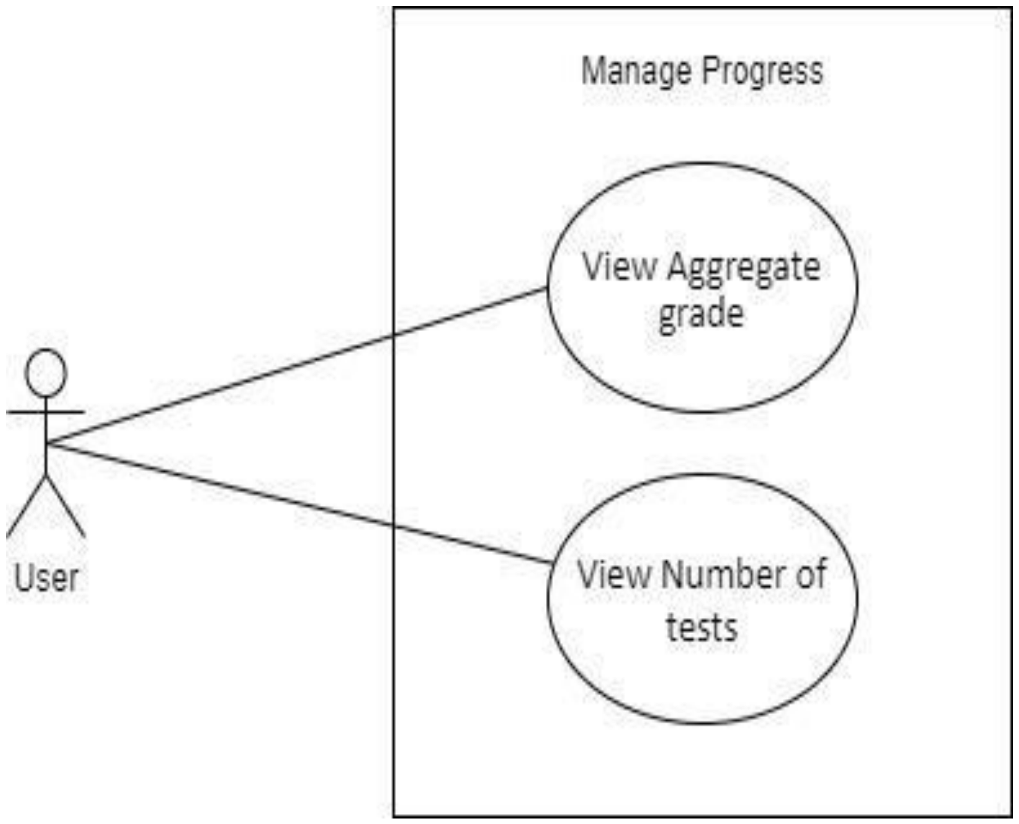
Requirements sheet.xlsx

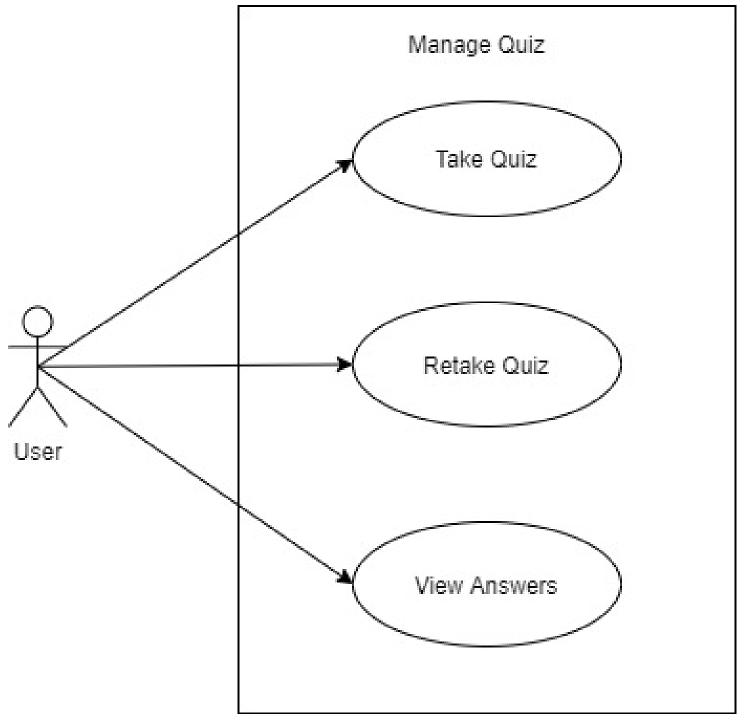


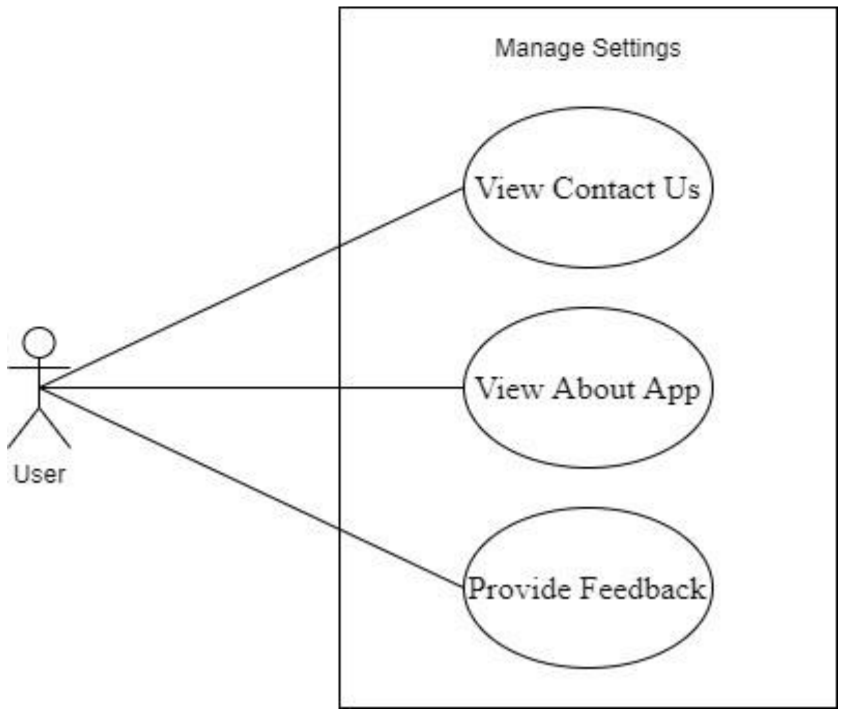
Req Id	Req Statement	Source
	Functional Requirement	
R1	The app shall provide a user login facility.	15
R1.1	The app shall provide a login option for users already registered with the app.	16
R1.2	The app shall provide an option to reset forgotten password.	17
R1.3	The app shall provide a user registration option for first time users to capture student ID, First name, La	20,21
R2	The app shall provide facility to take a new quiz with difficulty levels as easy, medim and hard.	29,30
R3	The app shall provide facility to retake a quiz based on grading criteria.	32
R4	The app shall provide facility to view all the quizzes taken.	35
R5	The app shall provide with the facility to view user's information.	37
R6	The app shall provide the facility to track progress status.	10
R7	The app shall provide option for users to reach out for support or help.	40
R8	The app shall provide option for users to describe their issues or share their feedback.	40
R9	The app shall provide option for users to view the current version of the application.	40
R10	The app shall provide option for users to view the names of the app developers.	41
R11	The app shall provide an option to logout.	41
R12	The app shall have a splash screen which shall include copyright symbol, date and group name.	Requirement feedback doc, item 12
	Non Functional Requirements	
R13	The app shall have static database.	Feedback doc, item 4
R14	The app shall provide quizzes for mathematics.	Feedback doc, item 4
R15	The app shall provide 3 quizzes in each difficulty level.	Feedback doc, item 4
R16	The app shall provide 15 multiple choice questions in each quiz.	Feedback doc, item 4

USE CASE DIAGRAMS:




Manage Progress
View Aggregate grade
View Number of tests
User


Manage Quiz
Take Quiz
Retake Quiz
View Answers
User


Manage Settings
View Contact Us
View About App
Provide Feedback
User

HIGH-LEVEL USE CASES:

UC 1: Login

- TUCBW the user entering the username and password and clicking on ‘login’ option on the login page.
- TUCEW the user viewing the home screen.

UC 2: Register

- TUCBW the user selecting the *Register* option.
- TUCEW the user getting a *Registered successfully* message and the login page being displayed.

UC 3: Reset password

- TUCBW the user selecting ‘Forgot Password’ option on the login page.
- TUCEW the user viewing the *Password sent successfully* message on the screen.

UC 4: Take Quiz

- TUCBW the user selecting ‘New Quiz’ option on the Quiz Details page.
- TUCEW the user viewing his grade.

UC 5: Retake Quiz

- TUCBW the user selecting ‘Retake Quiz’ option on the Quiz Details page.
- TUCEW the user viewing his grade for the completed quiz.

UC 6: View Answers

- TUCBW the user selecting ‘View Answers’ option on the Quiz Details page.
- TUCEW the user viewing the details of the selected quiz**.**

UC 7: View Profile

- TUCBW the user clicking on ‘Profile’.

- TUCEW the user viewing his/her personal information

UC 8: View aggregate grade.

- TUCBW the user selecting ‘View Grade’ option on the Progress Status page.

- TUCEW the user viewing his/her aggregate grade.

UC 9: View Number of tests

- TUCBW the user selecting ‘Number of tests’ option on the Progress Status page.

- TUCEW the user viewing the number of attempted quizzes.

UC 10: View Contact Details

- TUCBW the user selecting the Contact Us option.

- TUCEW the user viewing the contact information of the developer.

UC 11: View About App

- TUCBW the user selecting the ‘About app’ option.

- TUCEW the user viewing the names of the developers and the application description.

UC 12: Provide Feedback

- TUCBW the user entering the feedback and clicking on ‘Submit Feedback’.

- TUCEW the user viewing a confirmation message.

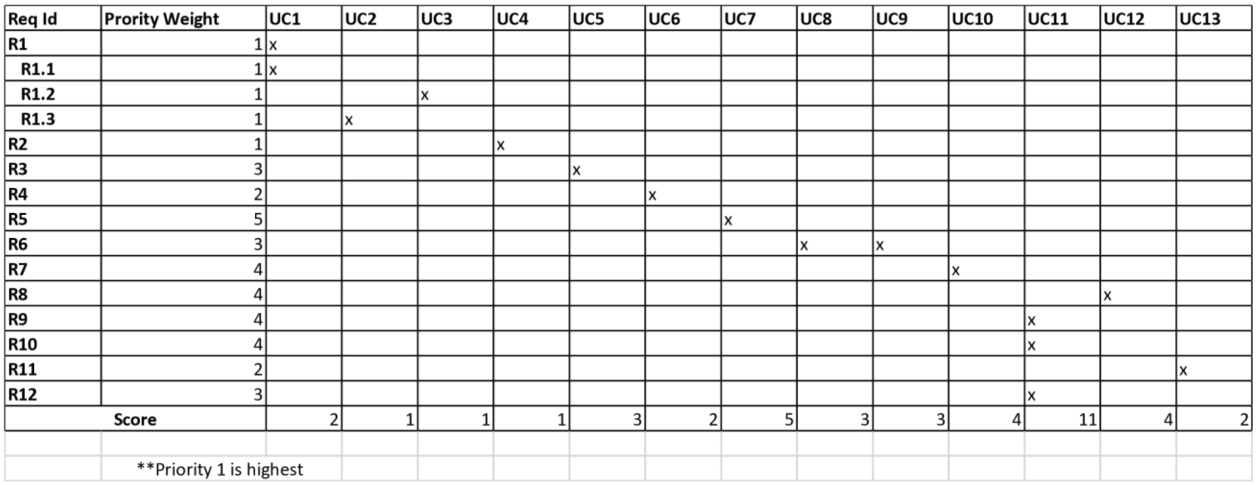
UC 13: Logout

- TUCBW the user selecting ‘Logout’ option on the settings page.

- TUCEW the user viewing the login page.

TRACEABILITY MATRIX:

Final Traceability Matrix.xlsx



Req Id	Prority Weight	UC1	UC2	UC3	UC4	UC5	UC6	UC7	UC8	UC9	UC10	UC11	UC12	UC13
R1	1	x												
R1.1	1	x												
R1.2	1			x										
R1.3	1		x											
R2	1				x									
R3	3					x								
R4	2						x							
R5	5							x						
R6	3								x	x				
R7	4										x			
R8	4												x	
R9	4											x		
R10	4											x		
R11	2													x
R12	3											x		
	Score	2	1	1	1	3	2	5	3	3	4	11	4	2

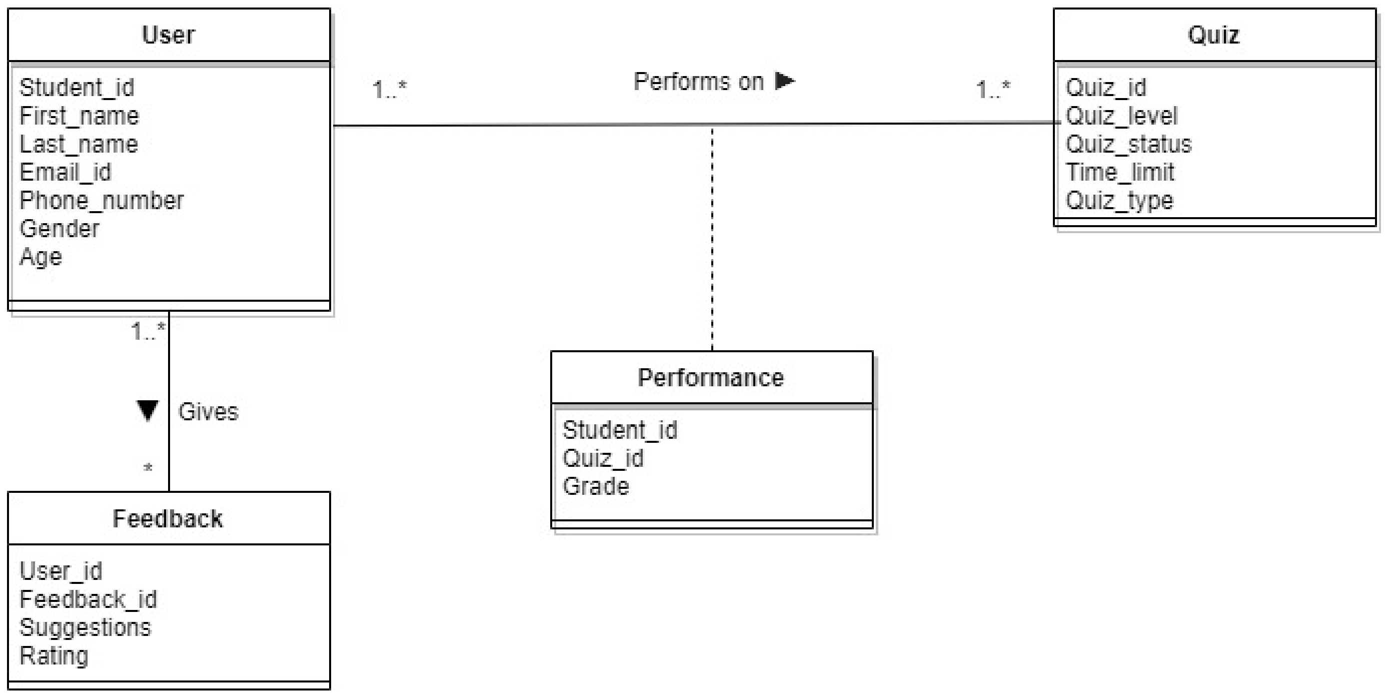
**Priority 1 is highest

INCREMENT MATRIX:

increment matrix.xlsx

Usecase	Priority	Effort (Person - Weeks)	Depends On	Iteration 01 (Due Date) 03/20/2020	Iteration 02 (Due Date) 04/10/2020	Iteration 03 (Due Date) 05/01/2020
UC01	2	1	None	1		
UC02	1	1	UC01	1		
UC03	1	1	UC01	1		
UC04	1	3	None	3		
UC05	3	2	UC04		2	
UC06	2	3	UC04	2	1	
UC07	5	1	UC02			1
UC08	3	1	UC06		1	
UC09	3	1	UC06		1	
UC10	4	1	None			1
UC11	11	1	None			1
UC12	4	1	None			1
UC13	2	1	None	1		
Total Effort		18		9	5	4

DOMAIN MODEL:



TASK ASSIGNED LIST:

task assigned list.xlsx

Task - List	Assigned to
Project description	Everyone
Requirements	Shrutikirti Saxena
Use Case diagrams (update)	Sai Charan Thannir and Shrutikirti Saxena
High level Use Cases (update)	Shrutikirti Saxena
RUTM (update)	Shrutikirti Saxena
Increment Matrix (update)	Shrutikirti Saxena
Domain Model (update)	Sayali Kamble and Sai Charan Thannir
Expanded Use cases (update)	Shrutikirti Saxena, Sayali Kamble, Sai Charan Thannir
Expanded Use Case UI Prototypes (update)	Bhuvan Poola and Sai Venkata Bhanu Shasank Bonthala
Sequence Diagram (update)	Sai Charan Thannir, Shrutikirti Saxena, Sayali Kamble
Design Class Diagram (update)	Sayali Kamble, Sai Charan Thannir
Android Screenshots (update)	Bhuvan Poola and Sai Venkata Bhanu Shasank Bonthala
Task Assigned List (update)	Shrutikirti Saxena
Presentation (update)	Shrutikirti Saxena
Change Logs (update)	Shrutikirti Saxena
UC1: Login (development)	Bhuvan Poola and Sai Venkata Bhanu Shasank Bonthala
UC2: Register (development)	Bhuvan Poola and Sai Venkata Bhanu Shasank Bonthala
UC3: Reset Password (development)	Bhuvan Poola and Sai Venkata Bhanu Shasank Bonthala
UC4: Take Quiz (development)	Bhuvan Poola and Sai Venkata Bhanu Shasank Bonthala
UC5: Retake Quiz (development)	Bhuvan Poola and Sayali Kamble
UC6: View Answers (development)	Bhuvan Poola and Sayali Kamble
UC7: View Profile (development)	Bhuvan Poola and Sai Venkata Bhanu Shasank Bonthala
UC8: View Aggregate grade(development)	Shrutikirti Saxena and Sai Charan Thannir
UC9: View tests (development)	Shrutikirti Saxena and Sai Charan Thannir
UC10: View Contact Details (development)	Bhuvan Poola and Sai Venkata Bhanu Shasank Bonthala
UC11: View About App (development)	Bhuvan Poola and Sai Venkata Bhanu Shasank Bonthala
UC12: Provide Feedback (development)	Bhuvan Poola and Sai Venkata Bhanu Shasank Bonthala
UC13: Logout (development)	Bhuvan Poola and Sai Venkata Bhanu Shasank Bonthala

EXPANDED USE CASES (With Expanded UI prototypes):

****Non-Trivial steps are marked in bold.**

UC 1: Login

Pre-condition: The user must possess valid credentials.	
Actor: User	System: Quizzie
	0. System displays the *login page.* (Fig. a)
1. TUCBW the user entering the username and password and clicking on *login* option on the *login* page.	**2. System displays the *home screen.* (Fig. b)**
3. TUCEW the user views the *home screen.*	
Post-condition: The user being logged into Quizze application.	

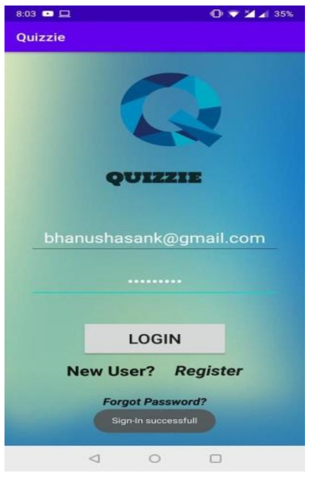


Fig a.

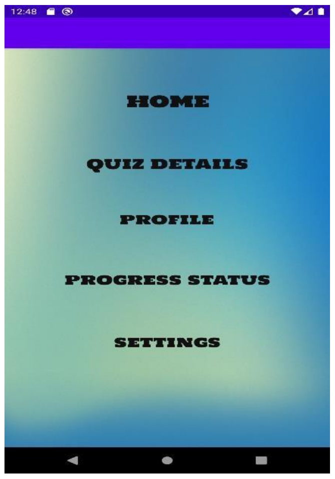


Fig. b

UC 02: Register

Precondition : The user must not be registered previously	
Actor : User	System : Quizzie
	0. System displays *Register* page. (Fig. c)
1. TUCBW the user registering using First Name , Last Name , Email Id , Password , Phone number , Age , Gender on the *Register* page by selecting *Register* option.	**2. System displays a *Registered Successfully* message. (Fig. d)**
3.TUCEW the user getting *registered successfully message* and the *Login* page being displayed .	
Post condition : NONE	

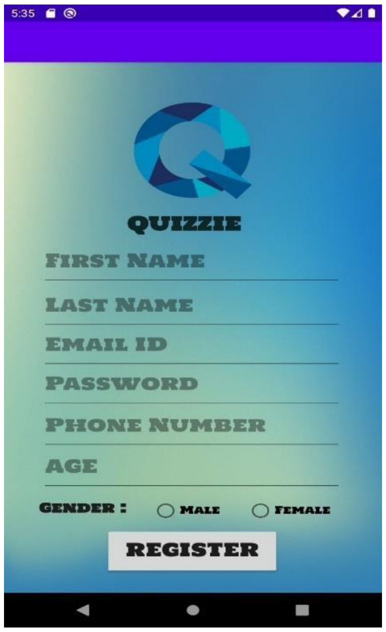
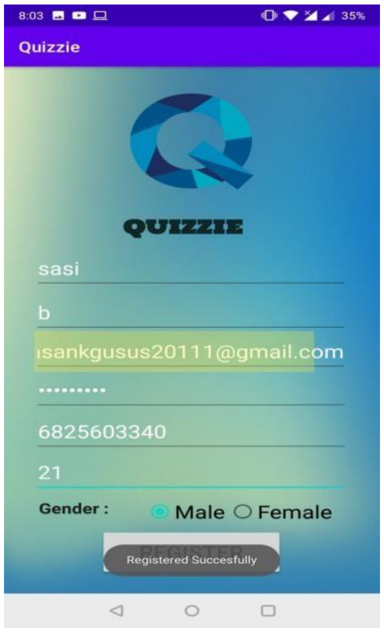


Fig c.



sasi
b
ısankgusus20111@gmail.com
•••••••••
6825603340
21
Registered Succesfully

Fig. d

UC 3: Reset Password

Precondition: The user must have lost access to his/her account.	
Actor: User	System: Quizzie
	0. System displays the *login* page. (Fig. e)
1. TUCBW the user selecting *Forgot Password* option on the *login* page.	2. System displays the *Forgot password* page requesting for the user's registered email address. (Fig. f)
3. The user enters the email address and clicks on *Send Password* button.	**4. System displays a *Password sent successfully* message on the screen. (Fig. g)**
5. TUCEW the user views the *Password sent successfully* message on the screen.	
Postcondition: None	

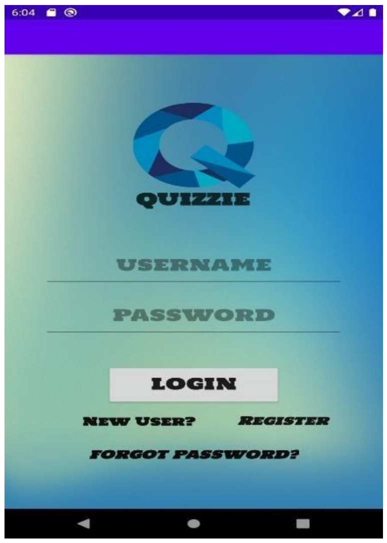


Fig e.

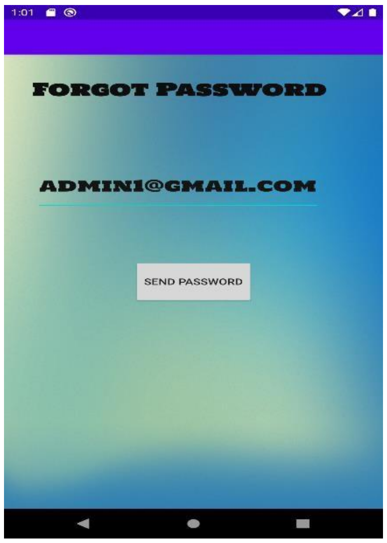


Fig. f

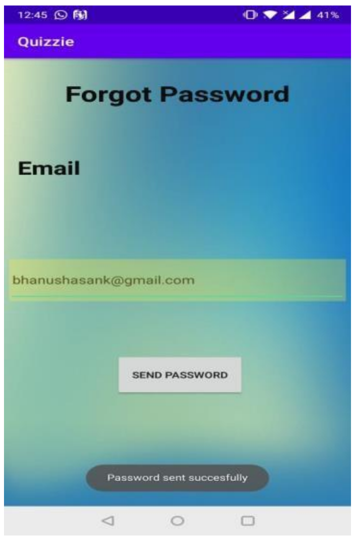


Fig. g

UC 4: Take Quiz

Pre-condition: The user must be logged into the system.	
Actor: User	System: Quizzie
	0. System displays *Quiz Details* page. (Fig. h)
1. TUCBW the user selecting *New* Quiz option on the *Quiz Details* page.	2. System displays the levels *easy, medium and hard.* (Fig. i)
3. The user selects one of the level.	**4. System displays the quizzes in the selected level. (Fig. j)**
5. The user selects one of the quizzes.	**6. System displays the selected quiz. (Fig. k)**
7. The user takes the quiz and clicks on submit.	**8. System displays the grade. (Fig. l)**
9. TUCEW the user viewing his grade.	
Post-condition: The progress status and view quiz getting updated.	

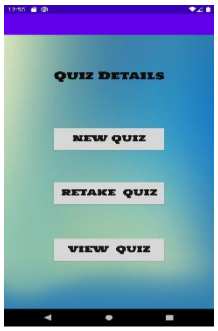


Fig. h

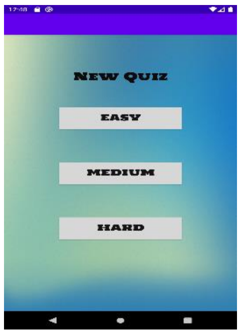


Fig. i

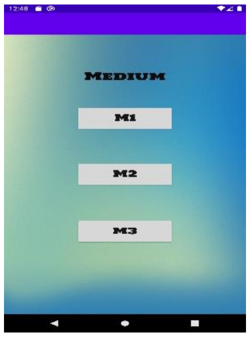
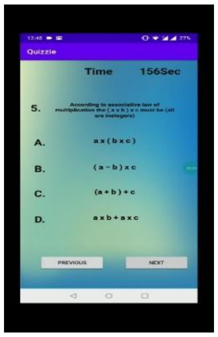


Fig. j



5. According to associative law of multiplication the (a x b) x c must be (all are integers)

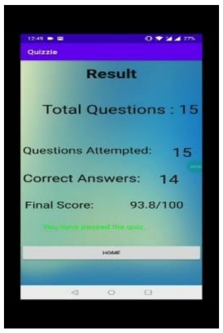
A. a x (b x c)

B. (a - b) x c

C. (a + b) + c

D. a x b + a x c

Fig. k



Total Questions : 15

Questions Attempted: 15

Correct Answers: 14

Final Score: 93.8/100

You have passed the quiz.

Fig. l

UC 5: Retake Quiz

Precondition : The user must have taken at least one test previously	
Actor: User	System: Quizzie
	0. System displays *Quiz Details* page. (Fig. m)
1. TUCBW the user selecting *Retake Quiz* option on the *Quiz details* page.	**2. System displays all the quiz in which the user has scored less than 50% (Fig. n)**
3.The user selects one quiz.	**4. System displays the selected quiz. (Fig. o)**
5. TUCEW user viewing his grades for the quiz completed.	
Post condition : NONE	

Fig. m

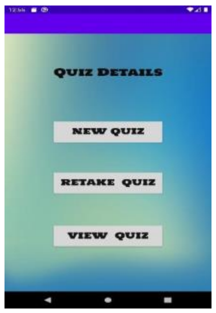
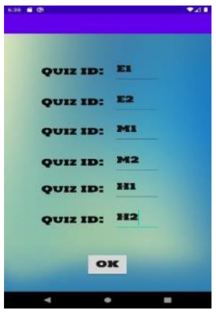
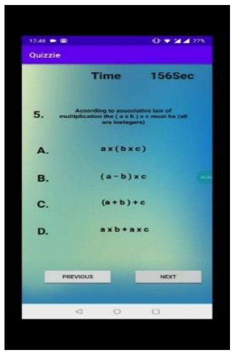


Fig. n





5. According to associative law of multiplication the (a x b) x c must be (all are inetegers)

A. a x (b x c)

B. (a − b) x c

C. (a + b) + c

D. a x b + a x c

Fig. o

UC 6: View Answers

Precondition: The user must have taken at-least one quiz in the past.	
Actor: User	System : Quizzie
	0. System displays *Quiz Details* page. (Fig. p)
1. TUCBW the user selecting *View Answers* option on the *Quiz Details* page.	**2. System displays all quizzes taken by the user in the past. (Fig. q)**
3. The user selects one Quiz and clicks on *OK* button.	**4. System displays the details of the selected quiz. (Fig. r)**
5. TUCEW the user viewing the details of the selected quiz.	
Postcondition: None	

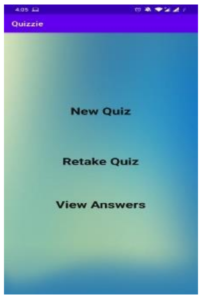


Fig. p

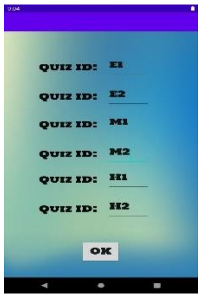
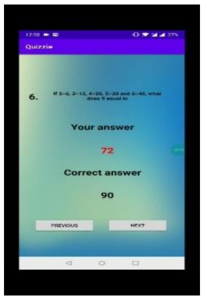

QUIZ ID: E1
QUIZ ID: E2
QUIZ ID: M1
QUIZ ID: M2
QUIZ ID: H1
QUIZ ID: H2

Fig. q



6. If 2=6, 3=12, 4=20, 5=30 and 6=42, what does 9 equal to

Your answer

72

Correct answer

90

Fig. r

UC 7: View Profile

Actor: User	System: Quizzie
	0. System displays the *Home screen.* (Fig i)
1. TUCBW the user selecting *Profile* option on the *Home screen.*	**2. System displays *First Name, Last Name, Email, Phone Number, Age.* (Fig ii)**
3. TUCEW the user viewing his/her profile information.	
Post-condition: None	

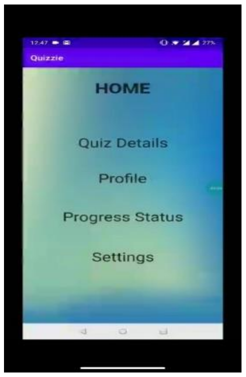
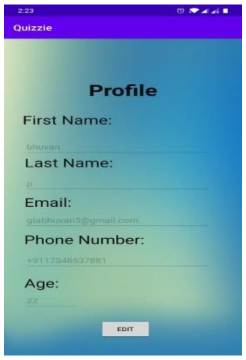


Fig. i



First Name:
bhuvan
Last Name:
p
Email:
gtabhuvan3@gmail.com
Phone Number:
+911734853788l
Age:
22

Fig. ii

UC 8: View Aggregate Grade

Precondition: The user must have taken at least one test	
Actor : User	System : Quizzie
	0. System displays *Progress Status* page. (Fig. s)
1. TUCBW the user selecting *View Grade* option on the *Progress Status* page	**2. System displaying aggregate grade. (Fig. t)**
3. TUCEW the user viewing his/her aggregate grades	
Post condition: NONE	

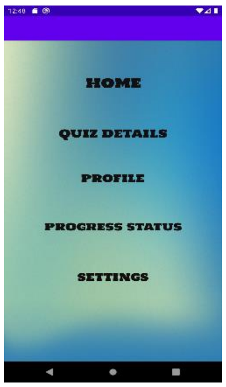


Fig. b

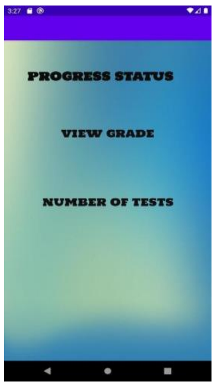


Fig. s

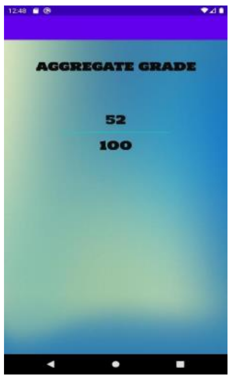


Fig. t

UC 9: View Number of tests

Precondition: The user must have taken at-least one quiz in the past.	
Actor: User	System: Quizzie
	0. System displays *Progress Status* page. (Fig. u)
1. TUCBW the user selects *Number of tests* option on the *Progress Status* page.	**2. System displays the number of the tests (quizzes) attempted. (Fig.v)**
3. TUCEW the user views the number of attempted tests (quizzes).	
Postcondition: None	

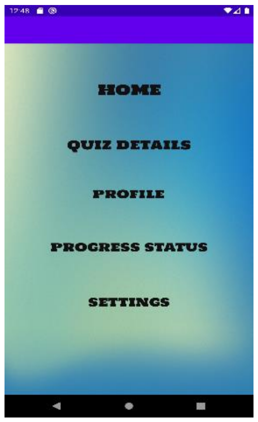


Fig. b

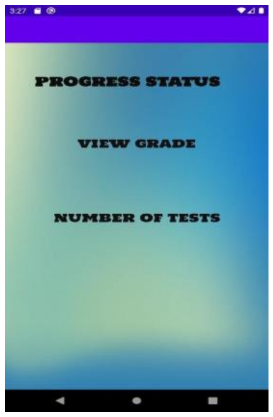


Fig. u

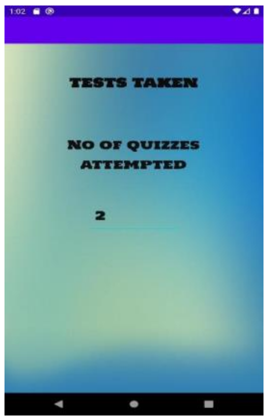


Fig. v

UC 10: View Contact Details

Pre-condition: None	
Actor: User	System: Quizzie
	0. System displays the *Settings* page. (Fig. iii)
1. TUCBW the user selecting *Contact Us* option on the *Settings* page.	**2. System displays contact information of the developer. (Fig. iv)**
3. TUCEW the user viewing *contact* information of the developer	
Post-condition: None	

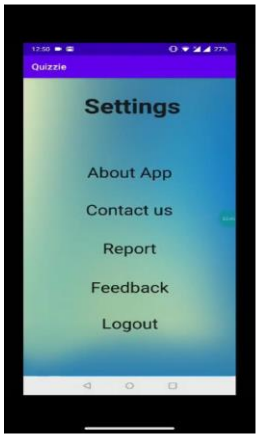
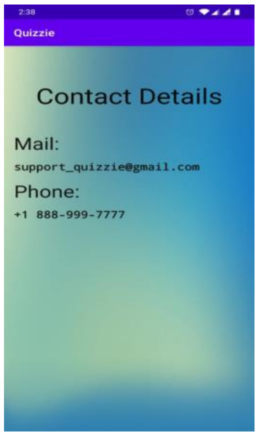


Fig. iii



Contact Details

Mail:

support_quizzie@gmail.com

Phone:

+1 888-999-7777

Fig. iv

UC 11: View About App

Pre-condition: None	
Actor: User	System: Quizzie
	0. System displays the *Settings* page. (Fig. v)
1. TUCBW the user selecting *About App* option on the *Settings* page.	**2. System displays the relevant details. (Fig. vi)**
3. TUCEW the user viewing details about the application.	
Post-condition: None	

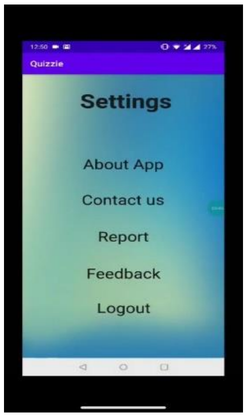
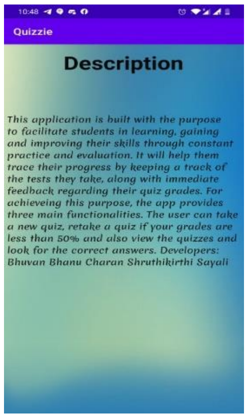


Fig. v



Description

This application is built with the purpose to facilitate students in learning, gaining and improving their skills through constant practice and evaluation. It will help them trace their progress by keeping a track of the tests they take, along with immediate feedback regarding their quiz grades. For achieveing this purpose, the app provides three main functionalities. The user can take a new quiz, retake a quiz if your grades are less than 50% and also view the quizzes and look for the correct answers. Developers: Bhuvan Bhanu Charan Shruthikirthi Sayali

Fig. vi

UC 12: Provide Feedback

Pre-condition: None	
Actor: User	System: Quizzie
	0. System displays the *Settings* page. (Fig. vii)
1. TUCBW the user selecting *Report* option on the *Settings* page.	2. System displays the feedback form and requests for user's feedback. (Fig. viii)
3. The user enters the feedback in the feedback form and clicks on *Submit* button.	**4. System displays a confirmation message of *Thank you for your feedback* on the screen. (ix)**
5. TUCEW the user viewing the confirmation message of *Thank you for your feedback* on the screen.	
Post-condition: None	

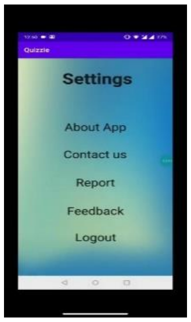
Fig. vii: 

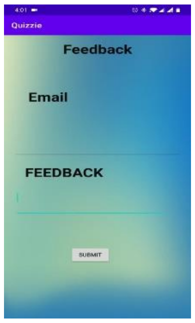
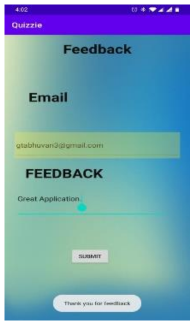
Fig. viii: 

Fig. ix: 

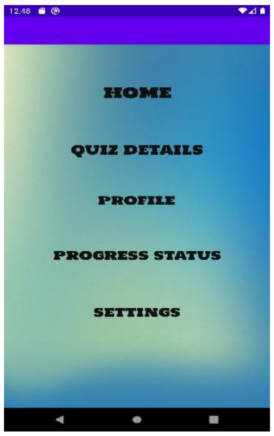
gtabhuvan3@gmail.com

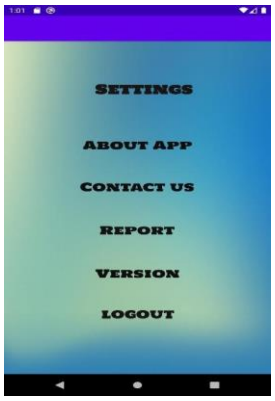
Great Application

Thank you for feedback

UC 13: Logout

Pre-condition: The user must be logged into the system.	
Actor: User	System: Quizzie
	0. System displays the *Settings* page. (Fig. w)
1. TUCBW the user selecting *Logout* option on the *Settings* page.	**2. System displays the *login* page. (Fig x.)**
3. TUCEW the user viewing the *login* page.	
Post-condition: The user being logged out of Quizze application.	





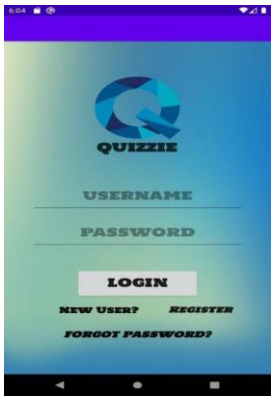
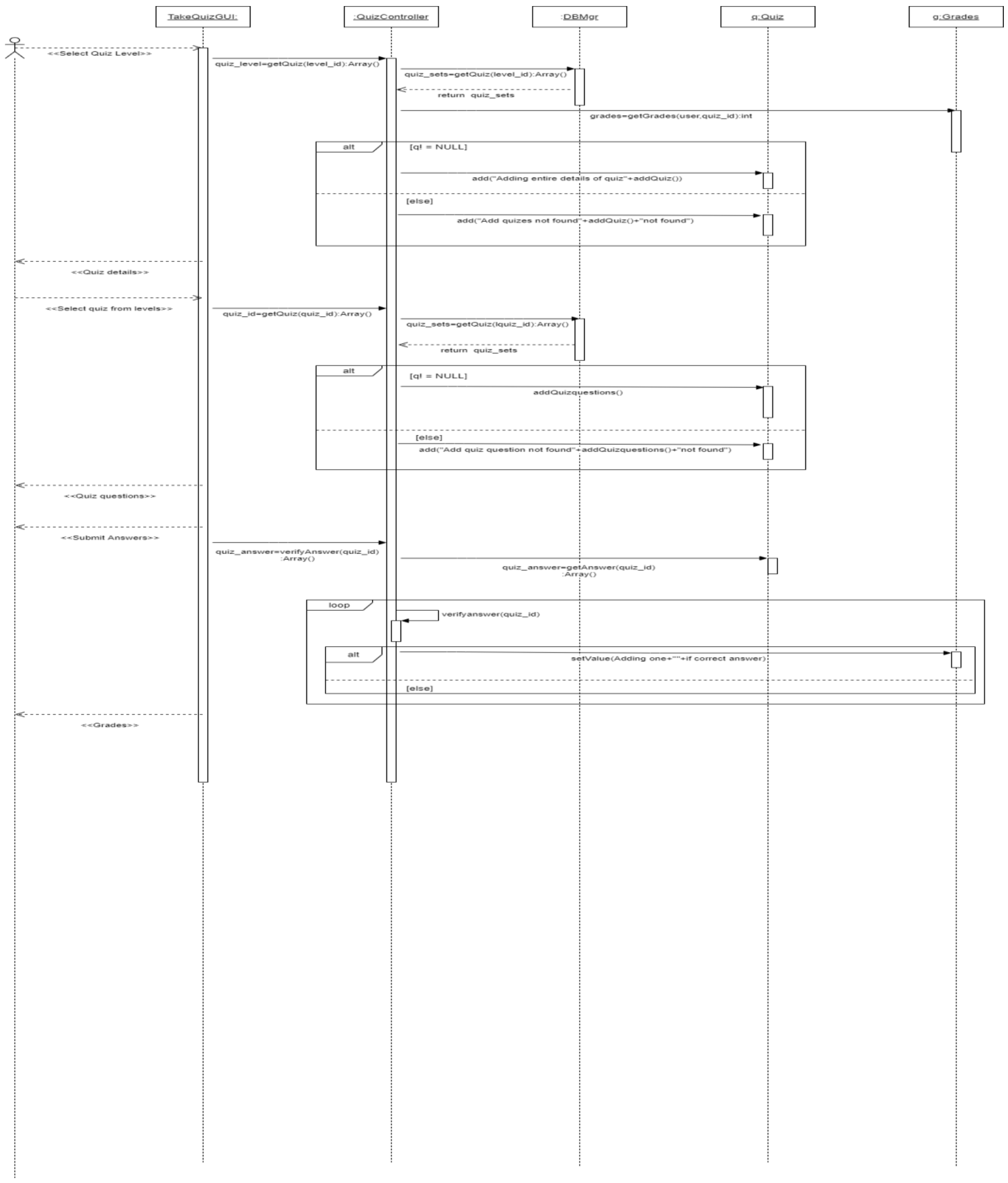


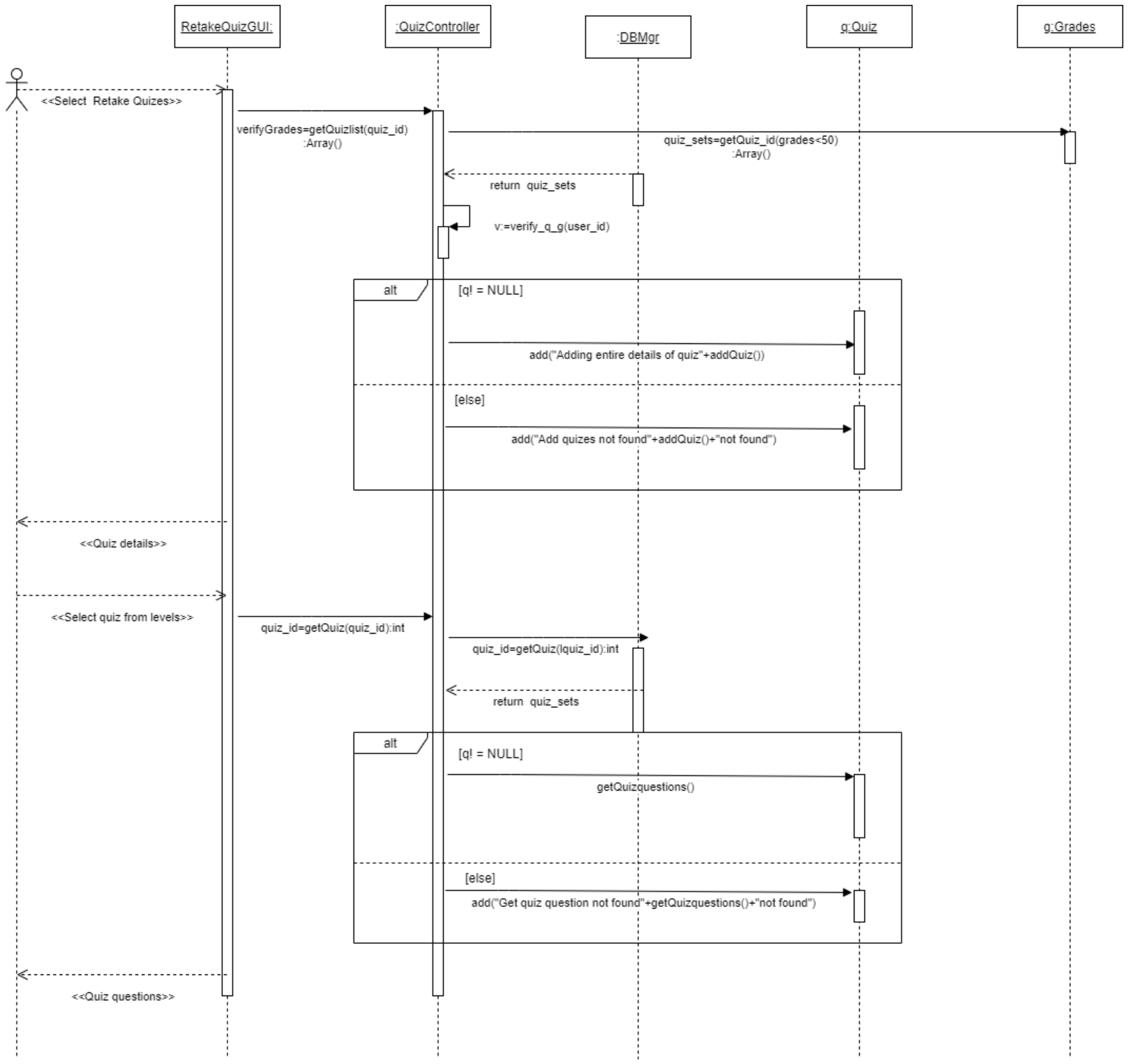
Fig. b Fig. w Fig. x

SEQUENCE DIAGRAMS:

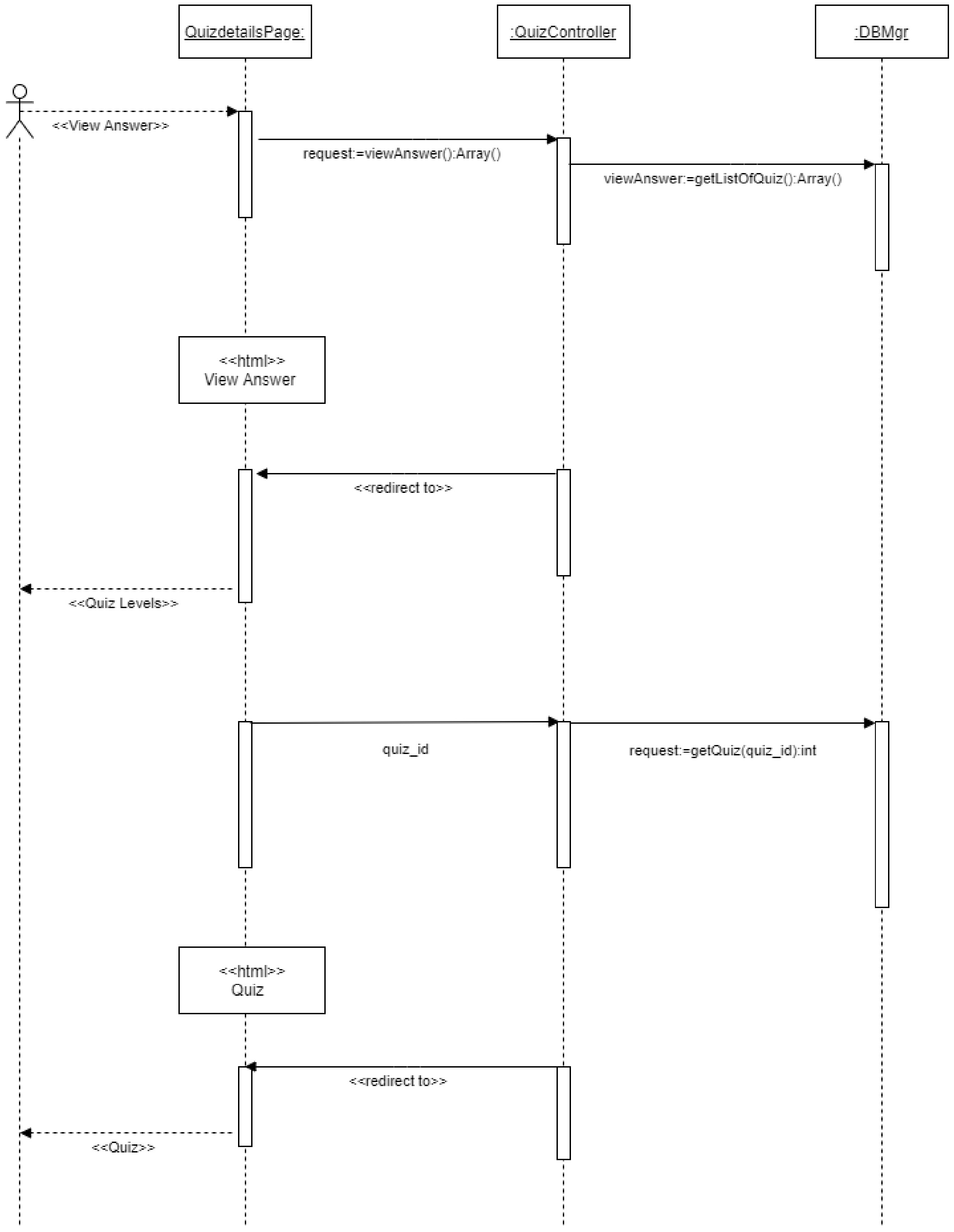
1. UC 4: Take Quiz


TakeQuizGUI:
:QuizController
:DBMgr
q:Quiz
g:Grades
<<Select Quiz Level>>
quiz_level=getQuiz(level_id):Array()
quiz_sets=getQuiz(level_id):Array()
return quiz_sets
grades=getGrades(user,quiz_id):int
alt
[q! = NULL]
add("Adding entire details of quiz"+addQuiz())
[else]
add("Add quizes not found"+addQuiz()+"not found")
<<Quiz details>>
<<Select quiz from levels>>
quiz_id=getQuiz(quiz_id):Array()
quiz_sets=getQuiz(lquiz_id):Array()
return quiz_sets
alt
[q! = NULL]
addQuizquestions()
[else]
add("Add quiz question not found"+addQuizquestions()+"not found")
<<Quiz questions>>
<<Submit Answers>>
quiz_answer=verifyAnswer(quiz_id) :Array()
quiz_answer=getAnswer(quiz_id) :Array()
loop
verifyanswer(quiz_id)
alt
setValue(Adding one+""+if correct answer)
[else]
<<Grades>>

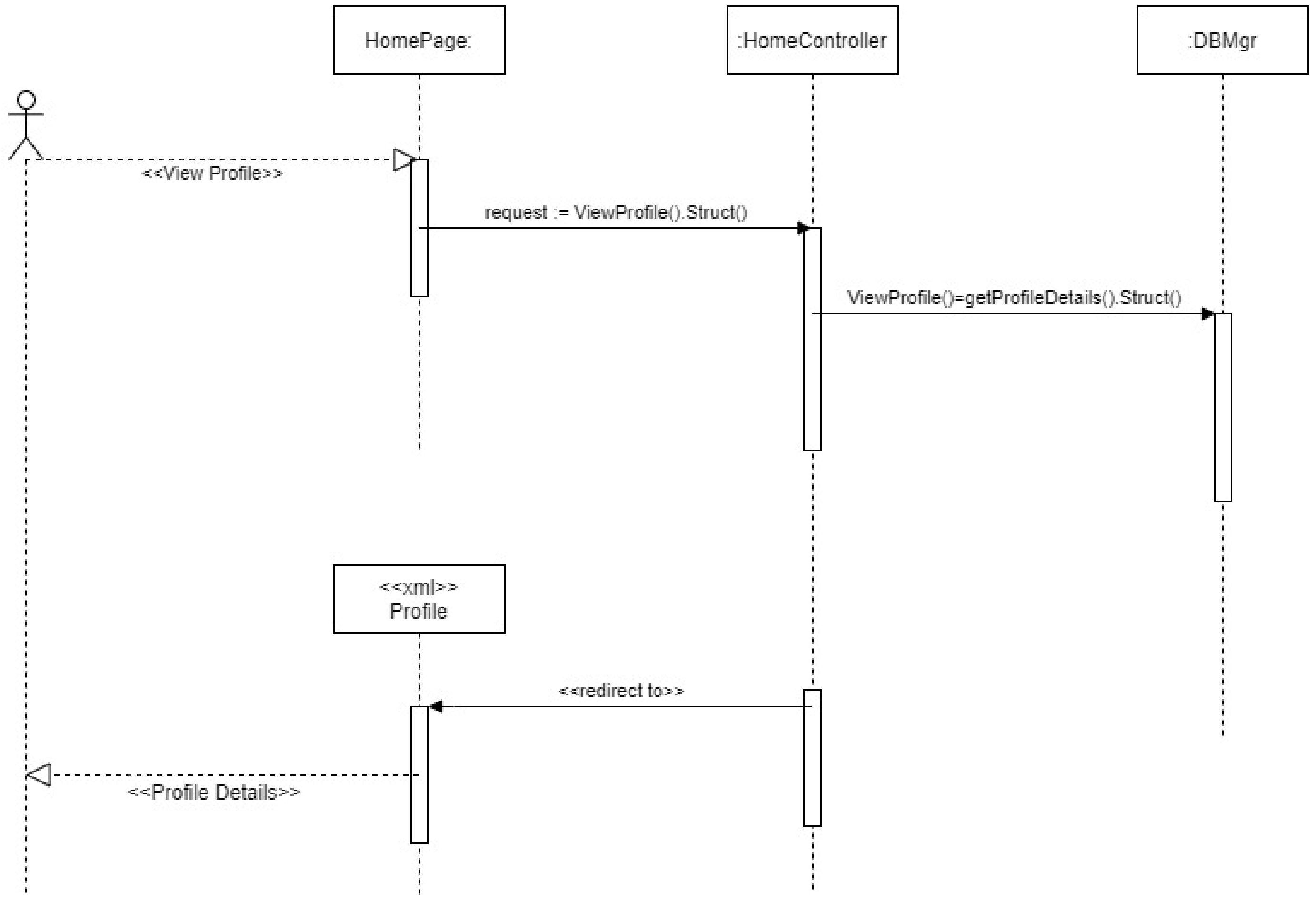
2. UC 5: Retake Quiz


RetakeQuizGUI:
:QuizController
:DBMgr
q:Quiz
g:Grades
<<Select Retake Quizes>>
verifyGrades=getQuizlist(quiz_id) :Array()
quiz_sets=getQuiz_id(grades<50) :Array()
return quiz_sets
v:=verify_q_g(user_id)
alt
[q! = NULL]
add("Adding entire details of quiz"+addQuiz())
[else]
add("Add quizes not found"+addQuiz()+"not found")
<<Quiz details>>
<<Select quiz from levels>>
quiz_id=getQuiz(quiz_id):int
quiz_id=getQuiz(lquiz_id):int
return quiz_sets
alt
[q! = NULL]
getQuizquestions()
[else]
add("Get quiz question not found"+getQuizquestions()+"not found")
<<Quiz questions>>

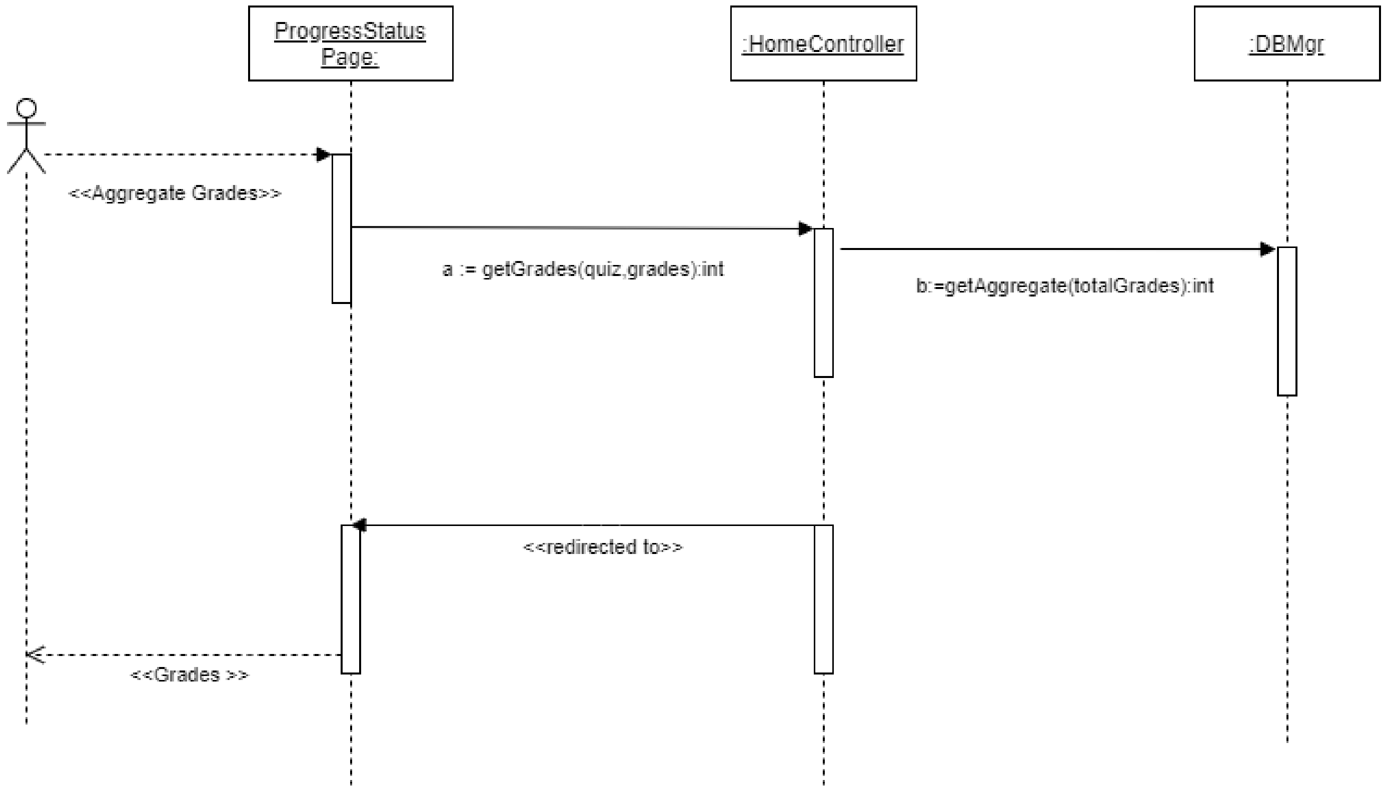
3. UC6: View Answers



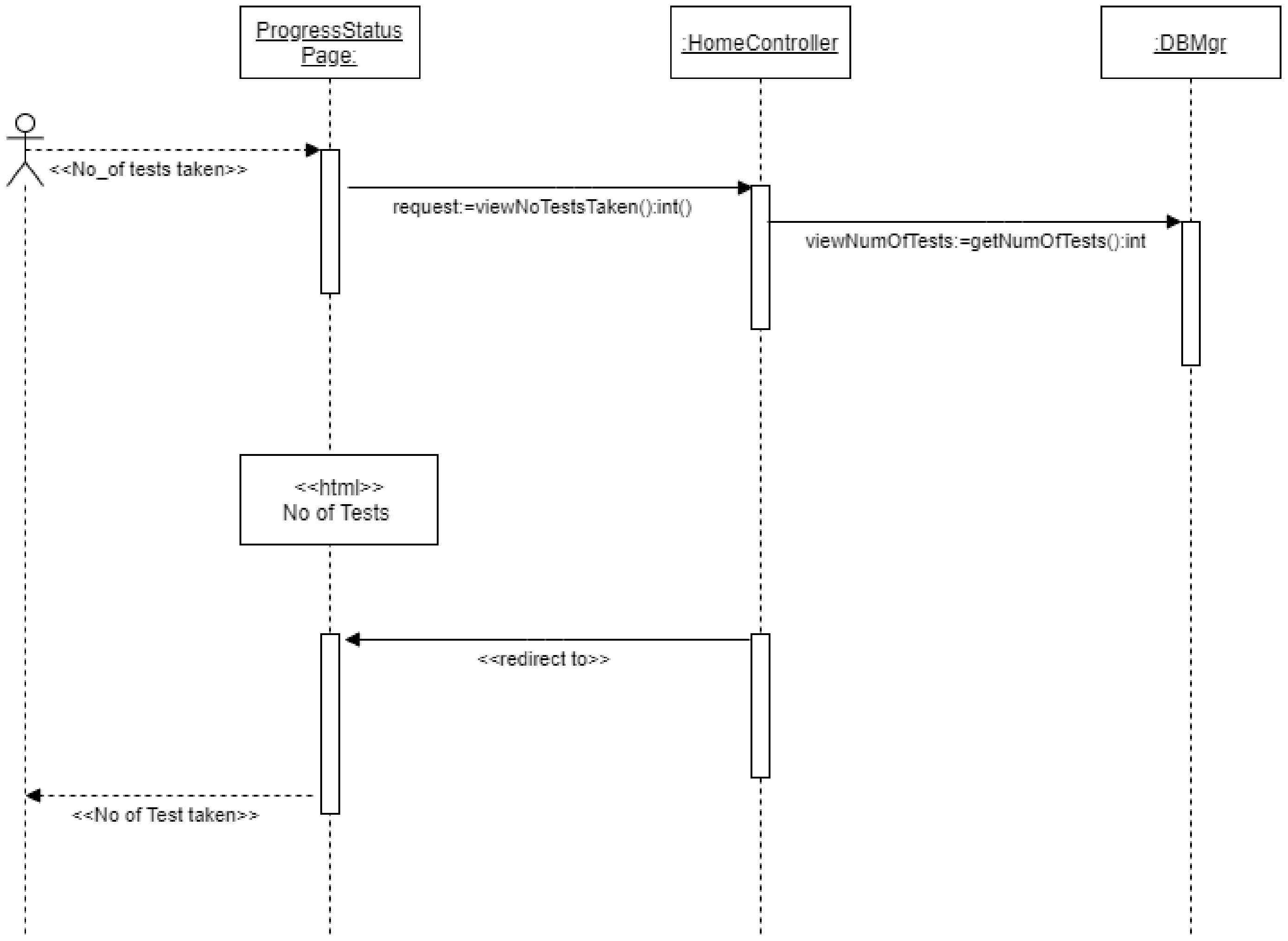
4. UC 7: View Profile



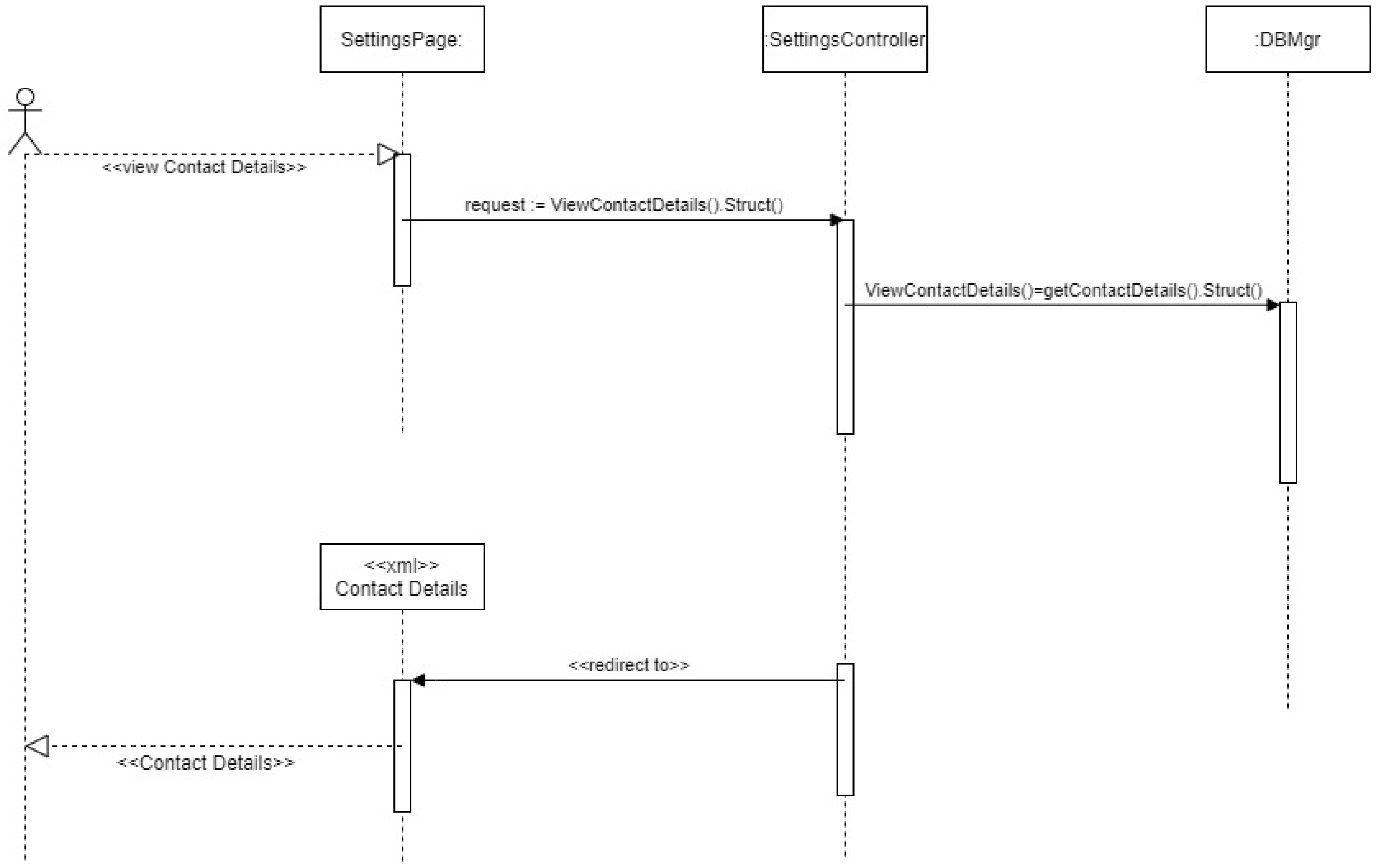
5. UC 8: View Aggregate Grade



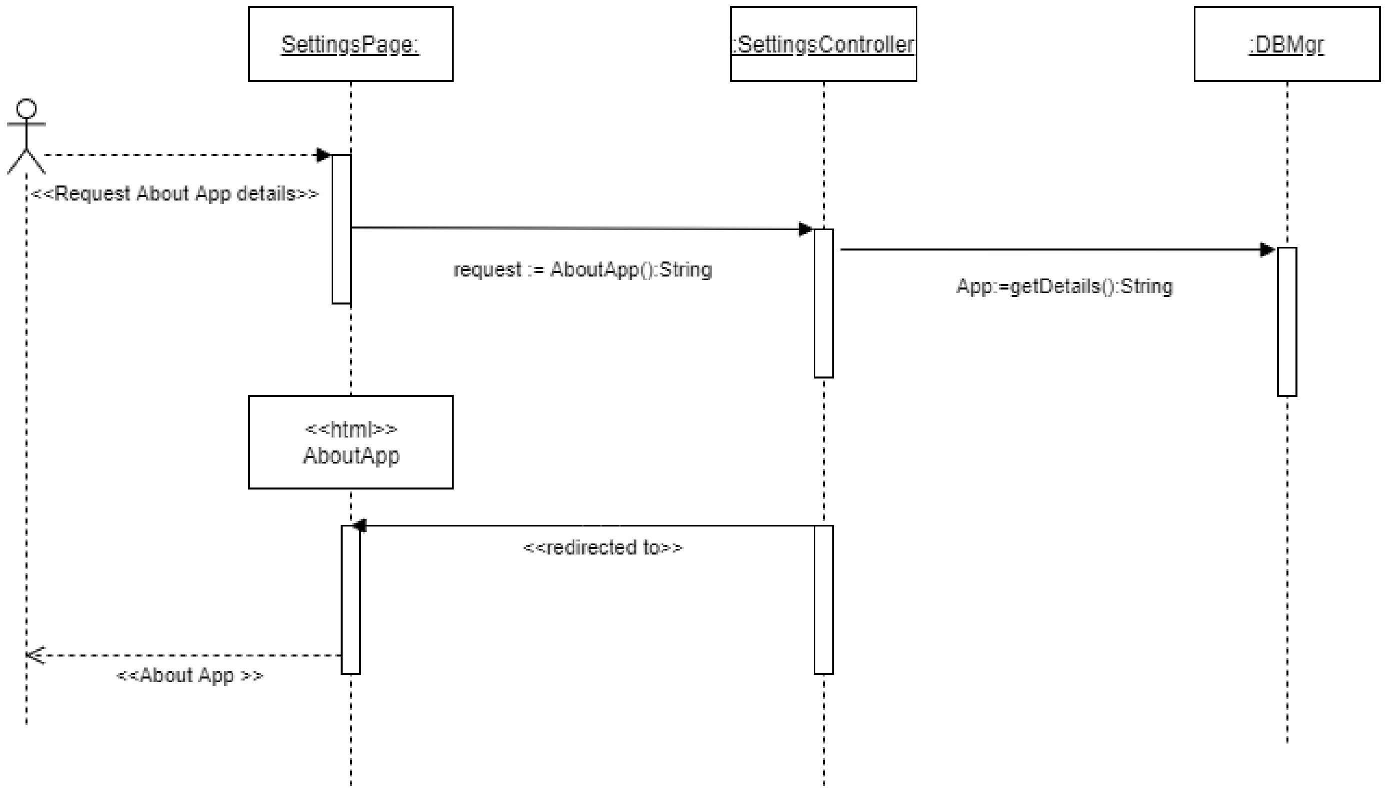
6. UC 9: View Number of Tests



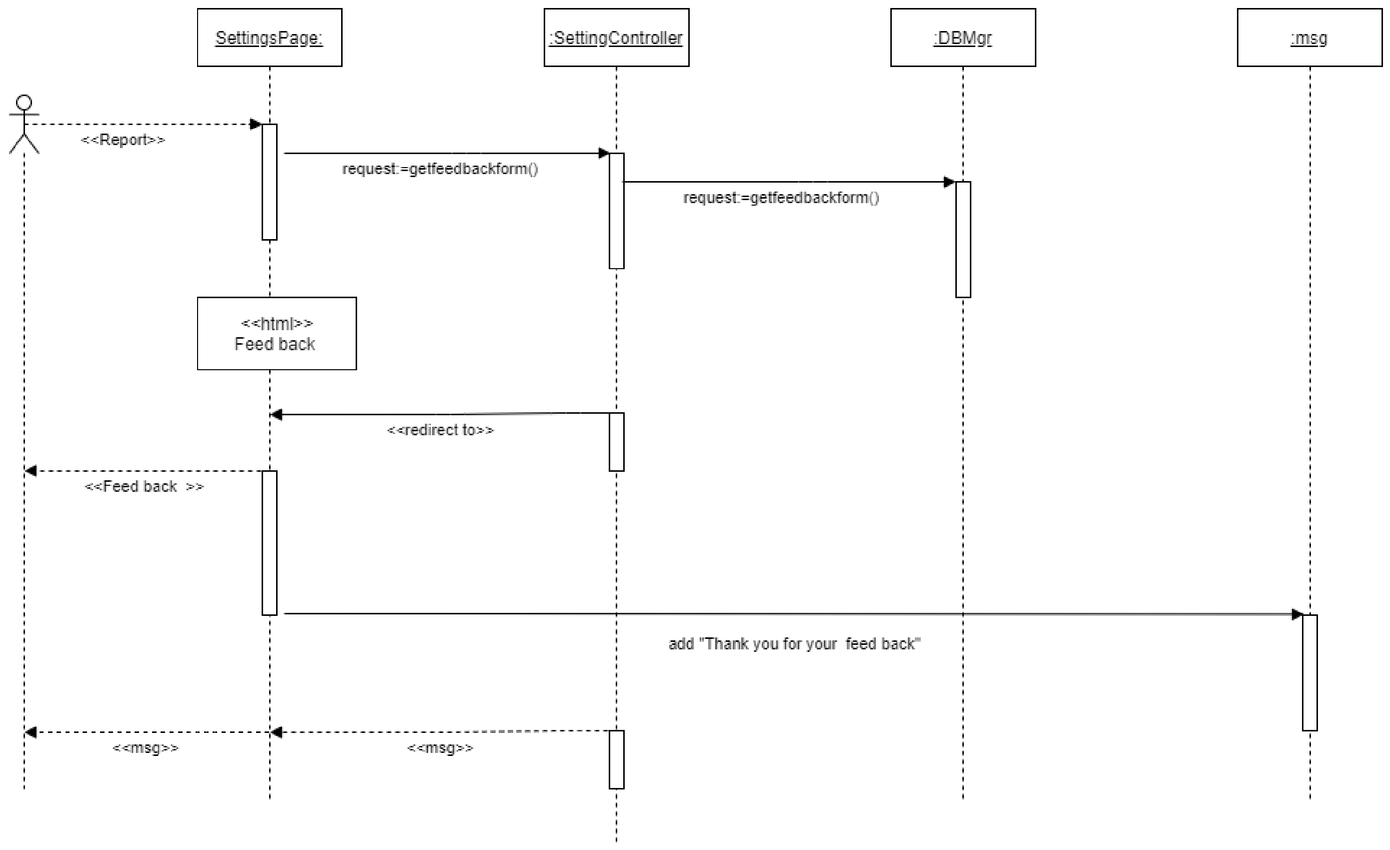
7. UC 10: View Contact Details



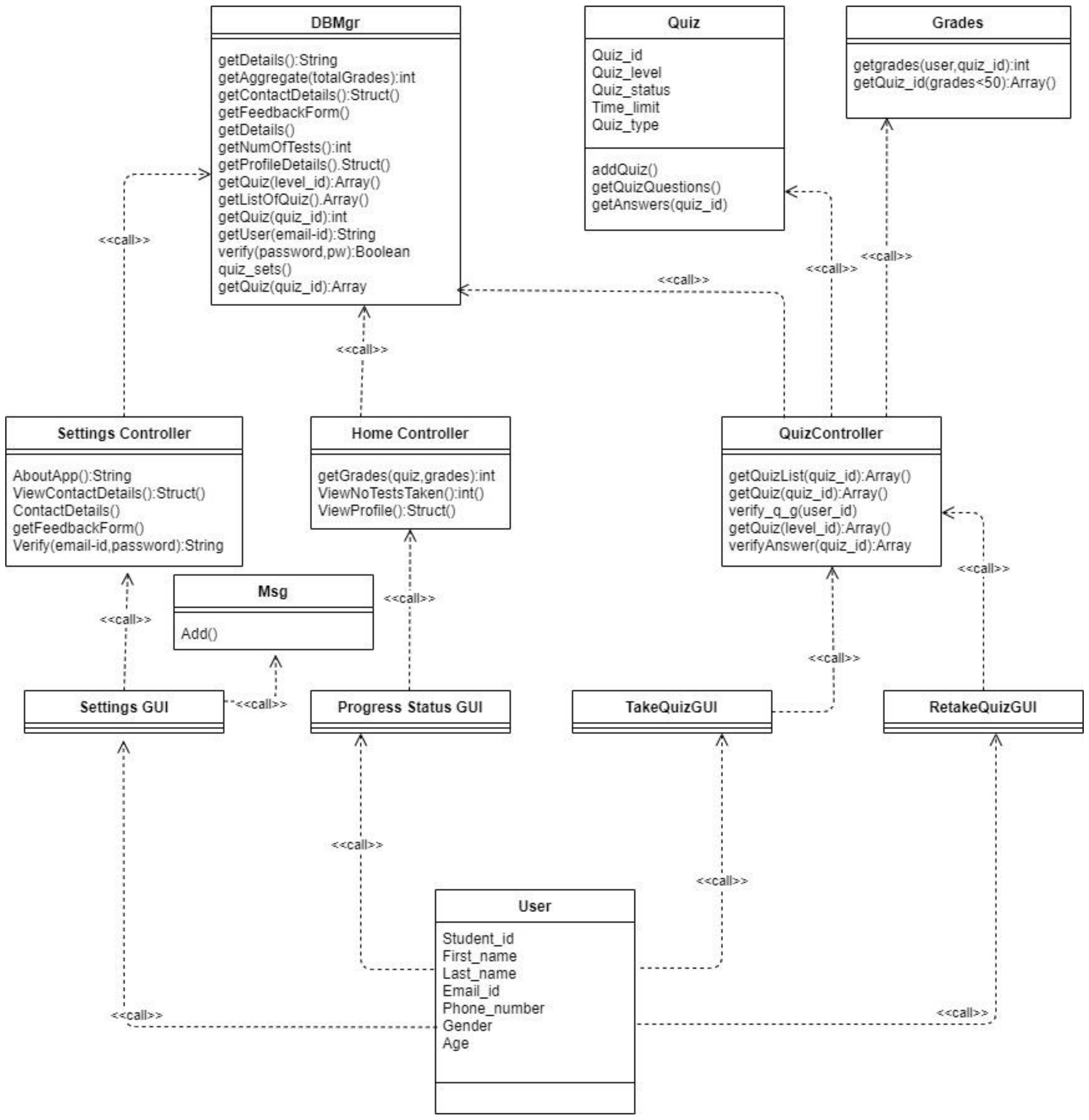
8. UC 11: View About App



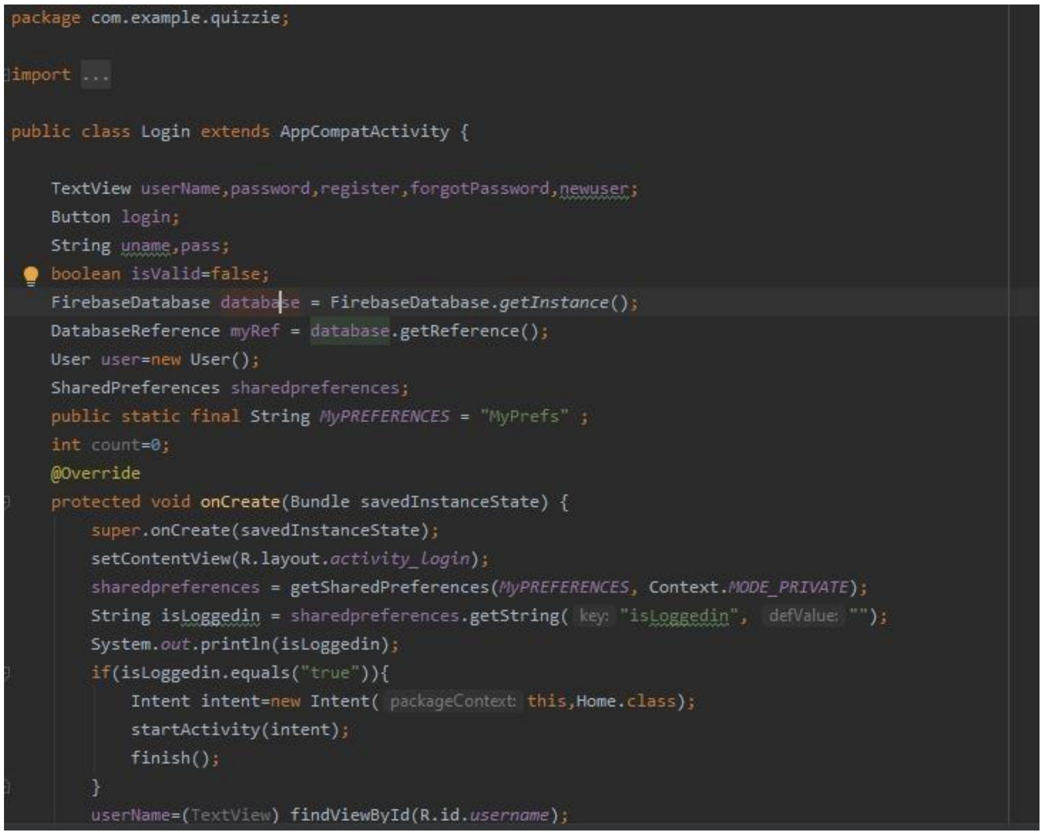
9. UC 12: Provide Feedback



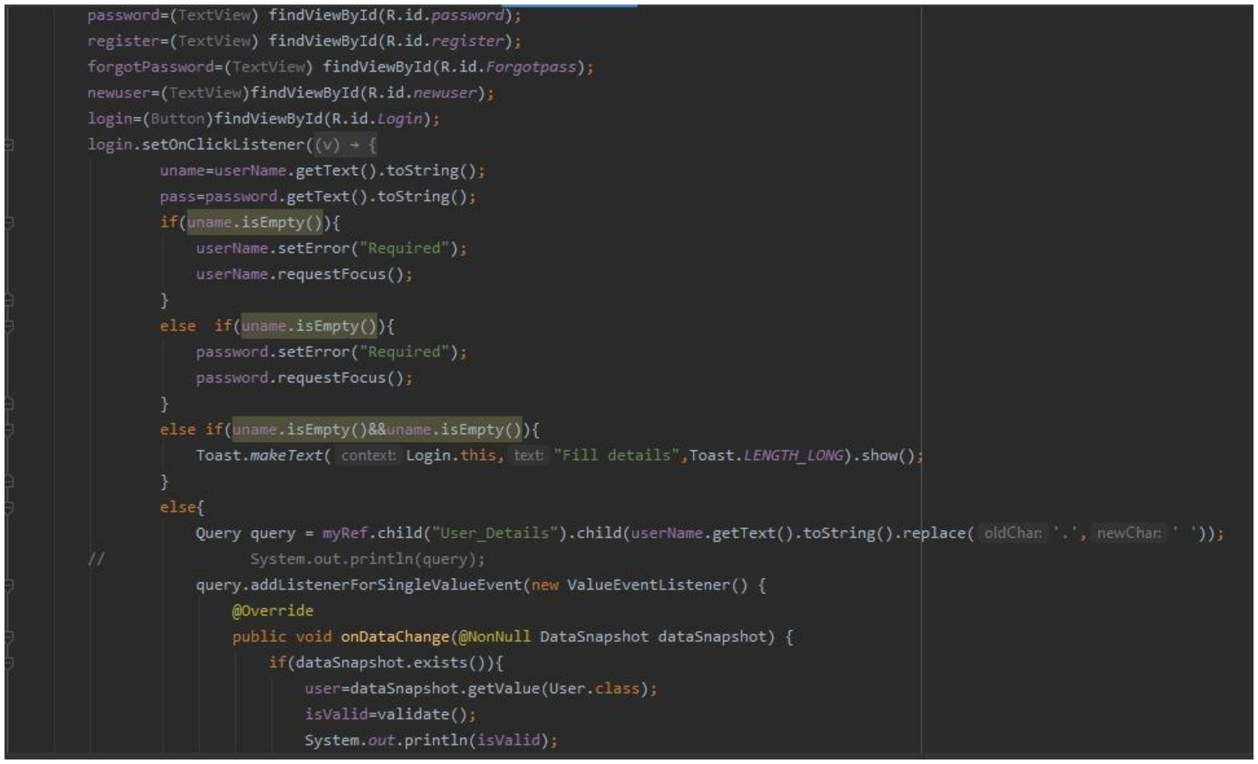
DESIGN CLASS DIAGRAM:



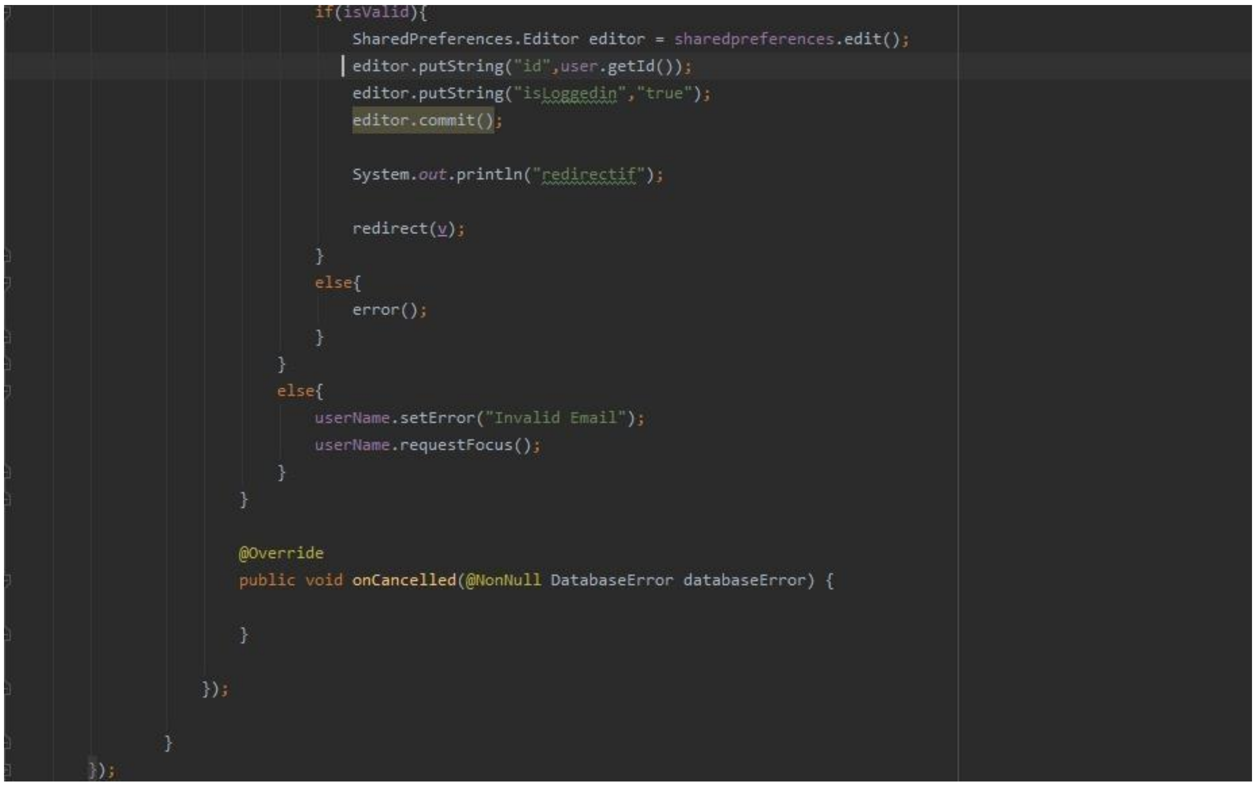
ANDROID SCREENSHOTS:



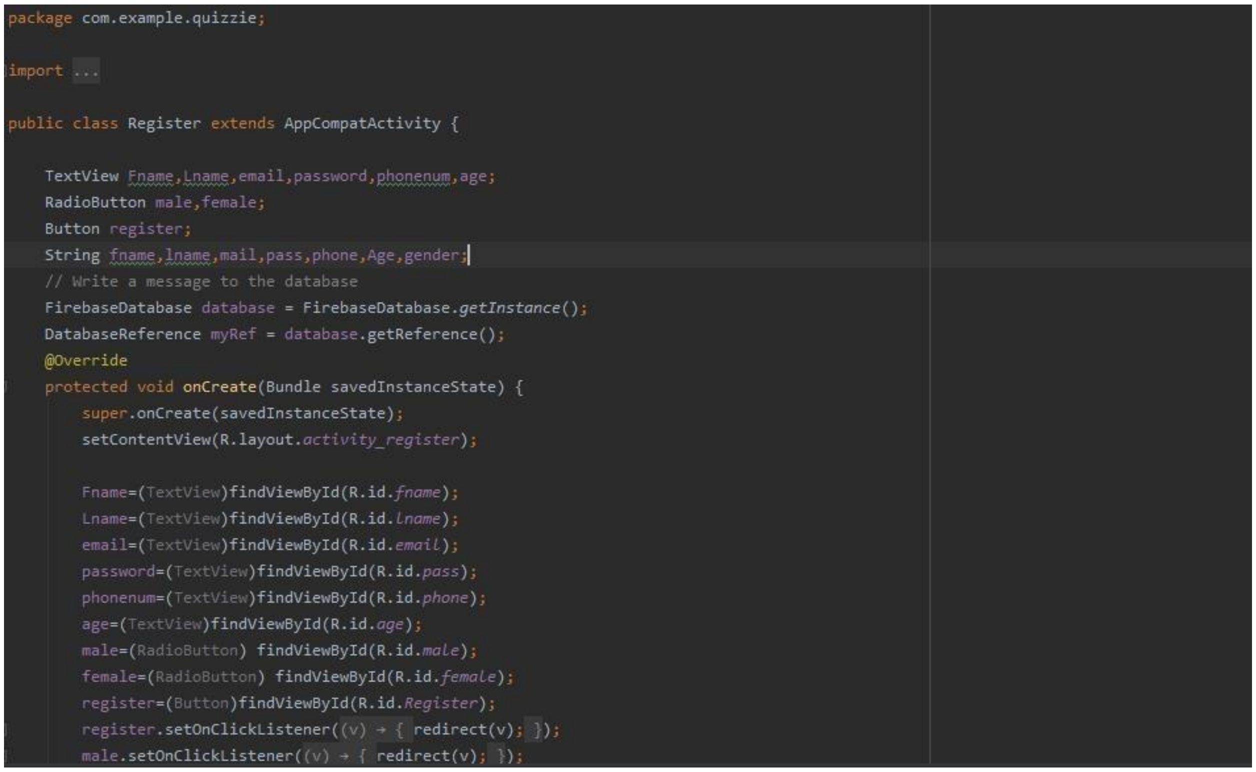
```java
package com.example.quizzie;

import ...

public class Login extends AppCompatActivity {

    TextView userName,password,register,forgotPassword,newuser;
    Button login;
    String uname,pass;
    boolean isValid=false;
    FirebaseDatabase database = FirebaseDatabase.getInstance();
    DatabaseReference myRef = database.getReference();
    User user=new User();
    SharedPreferences sharedpreferences;
    public static final String MyPREFERENCES = "MyPrefs" ;
    int count=0;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);
        sharedpreferences = getSharedPreferences(MyPREFERENCES, Context.MODE_PRIVATE);
        String isLoggedin = sharedpreferences.getString( key: "isLoggedin",  defValue: "");
        System.out.println(isLoggedin);
        if(isLoggedin.equals("true")){
            Intent intent=new Intent( packageContext: this,Home.class);
            startActivity(intent);
            finish();
        }
        userName=(TextView) findViewById(R.id.username);
```



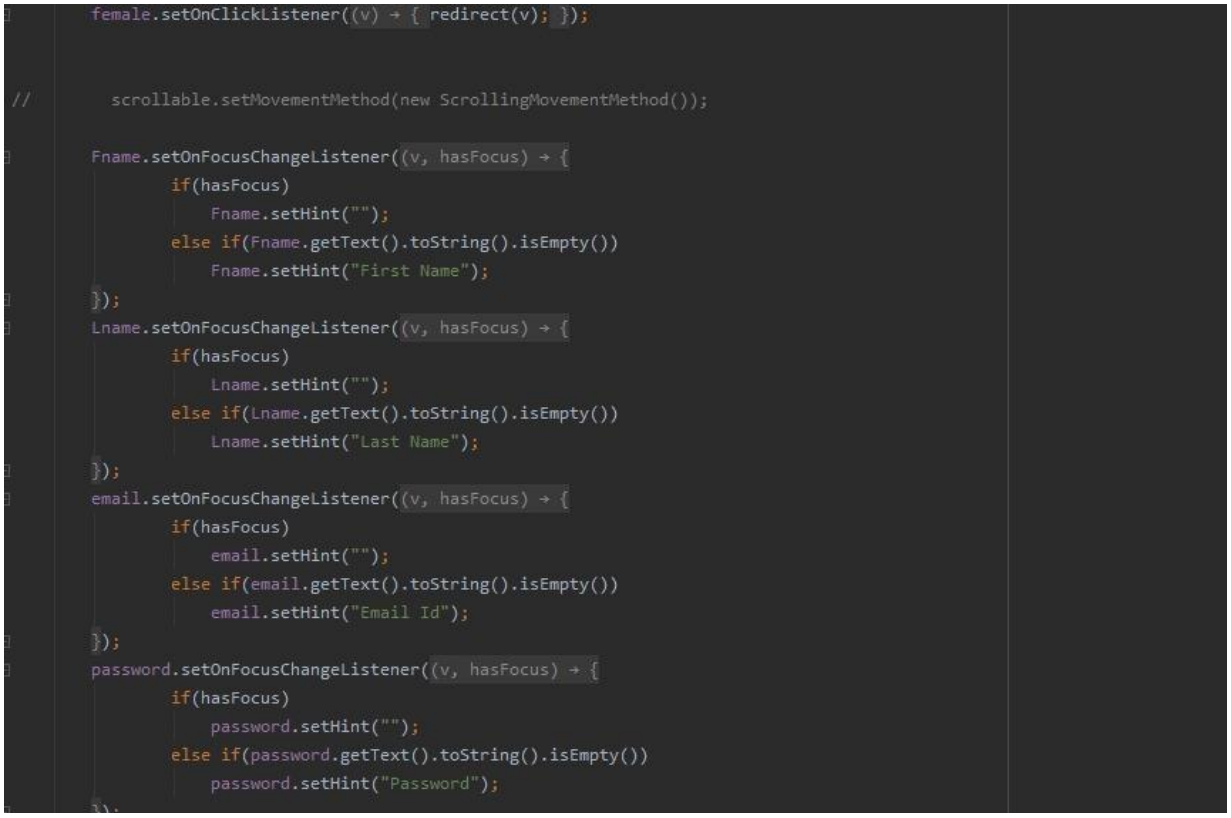
```java
        password=(TextView) findViewById(R.id.password);
        register=(TextView) findViewById(R.id.register);
        forgotPassword=(TextView) findViewById(R.id.Forgotpass);
        newuser=(TextView)findViewById(R.id.newuser);
        login=(Button)findViewById(R.id.Login);
        login.setOnClickListener((v) → {
                uname=userName.getText().toString();
                pass=password.getText().toString();
                if(uname.isEmpty()){
                    userName.setError("Required");
                    userName.requestFocus();
                }
                else  if(uname.isEmpty()){
                    password.setError("Required");
                    password.requestFocus();
                }
                else if(uname.isEmpty()&&uname.isEmpty()){
                    Toast.makeText( context: Login.this,  text: "Fill details",Toast.LENGTH_LONG).show();
                }
                else{
                    Query query = myRef.child("User_Details").child(userName.getText().toString().replace( oldChar: '.',  newChar: ' '));
        //                System.out.println(query);
                    query.addListenerForSingleValueEvent(new ValueEventListener() {
                        @Override
                        public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                            if(dataSnapshot.exists()){
                                user=dataSnapshot.getValue(User.class);
                                isValid=validate();
                                System.out.println(isValid);
```



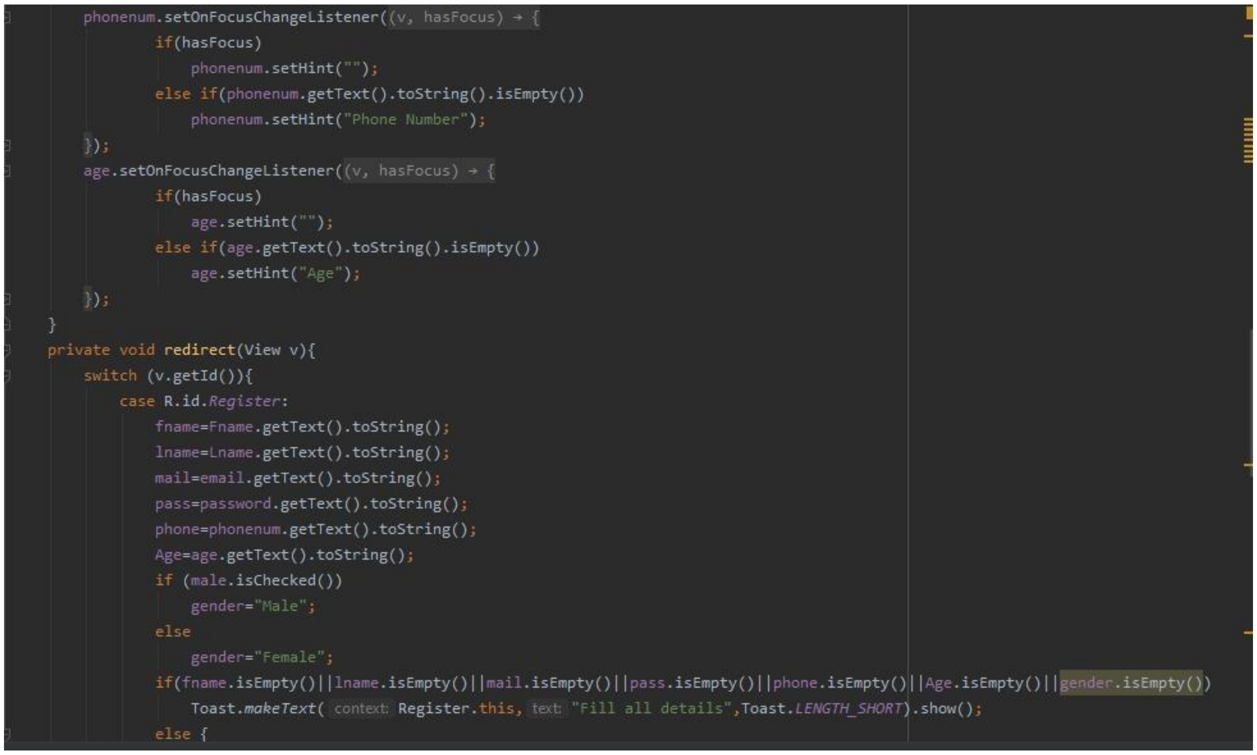
```
                        if(isValid){
                            SharedPreferences.Editor editor = sharedpreferences.edit();
                            editor.putString("id",user.getId());
                            editor.putString("isLoggedin","true");
                            editor.commit();

                            System.out.println("redirectif");

                            redirect(v);
                        }
                        else{
                            error();
                        }
                    }
                    else{
                        userName.setError("Invalid Email");
                        userName.requestFocus();
                    }
                }

                @Override
                public void onCancelled(@NonNull DatabaseError databaseError) {

                }

            });

        }
    });
```



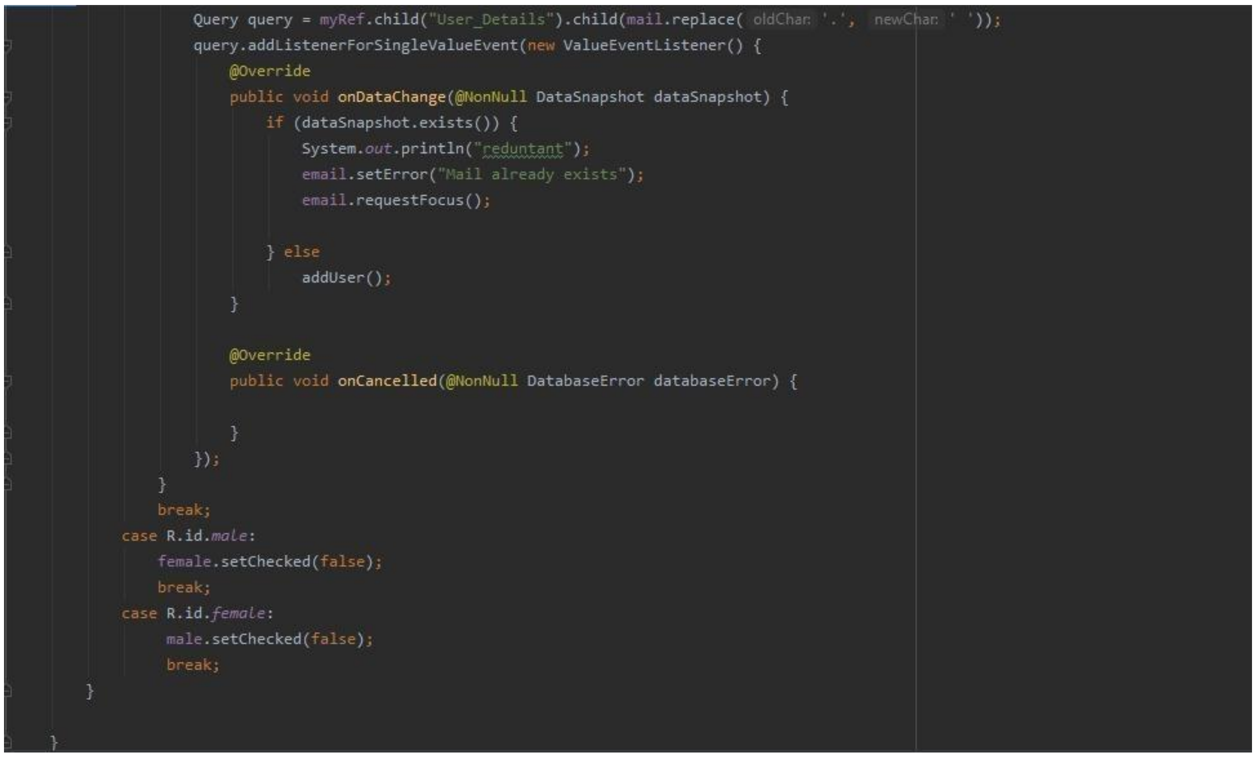
```
package com.example.quizzie;

import ...

public class Register extends AppCompatActivity {

    TextView Fname,Lname,email,password,phonenum,age;
    RadioButton male,female;
    Button register;
    String fname,lname,mail,pass,phone,Age,gender;
    // Write a message to the database
    FirebaseDatabase database = FirebaseDatabase.getInstance();
    DatabaseReference myRef = database.getReference();
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_register);

        Fname=(TextView)findViewById(R.id.fname);
        Lname=(TextView)findViewById(R.id.lname);
        email=(TextView)findViewById(R.id.email);
        password=(TextView)findViewById(R.id.pass);
        phonenum=(TextView)findViewById(R.id.phone);
        age=(TextView)findViewById(R.id.age);
        male=(RadioButton) findViewById(R.id.male);
        female=(RadioButton) findViewById(R.id.female);
        register=(Button)findViewById(R.id.Register);
        register.setOnClickListener((v) → { redirect(v); });
        male.setOnClickListener((v) → { redirect(v); });
```



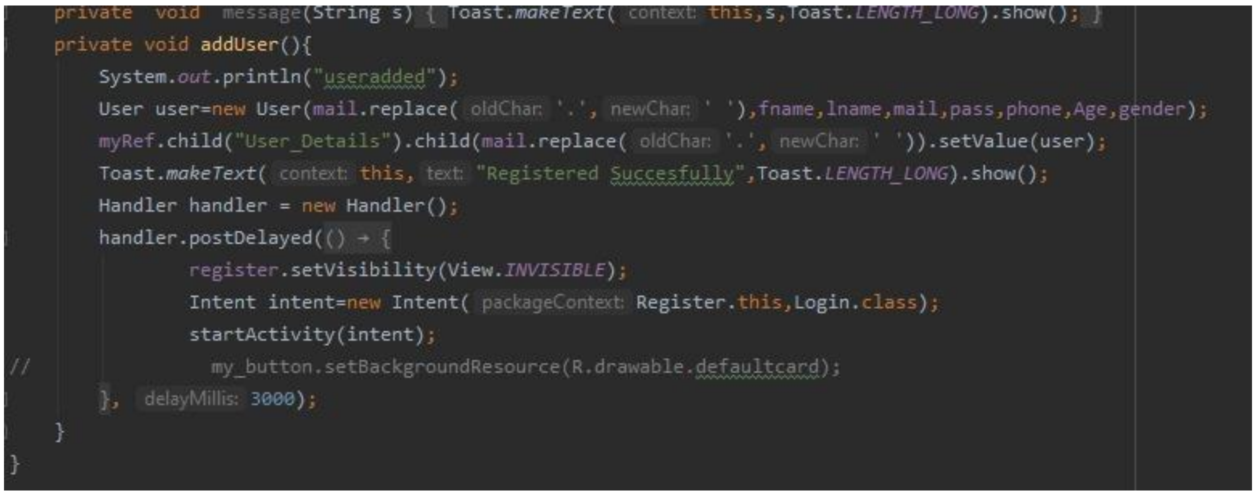
```
        female.setOnClickListener((v) → { redirect(v); });


//        scrollable.setMovementMethod(new ScrollingMovementMethod());

        Fname.setOnFocusChangeListener((v, hasFocus) → {
                if(hasFocus)
                    Fname.setHint("");
                else if(Fname.getText().toString().isEmpty())
                    Fname.setHint("First Name");
        });
        Lname.setOnFocusChangeListener((v, hasFocus) → {
                if(hasFocus)
                    Lname.setHint("");
                else if(Lname.getText().toString().isEmpty())
                    Lname.setHint("Last Name");
        });
        email.setOnFocusChangeListener((v, hasFocus) → {
                if(hasFocus)
                    email.setHint("");
                else if(email.getText().toString().isEmpty())
                    email.setHint("Email Id");
        });
        password.setOnFocusChangeListener((v, hasFocus) → {
                if(hasFocus)
                    password.setHint("");
                else if(password.getText().toString().isEmpty())
                    password.setHint("Password");
```



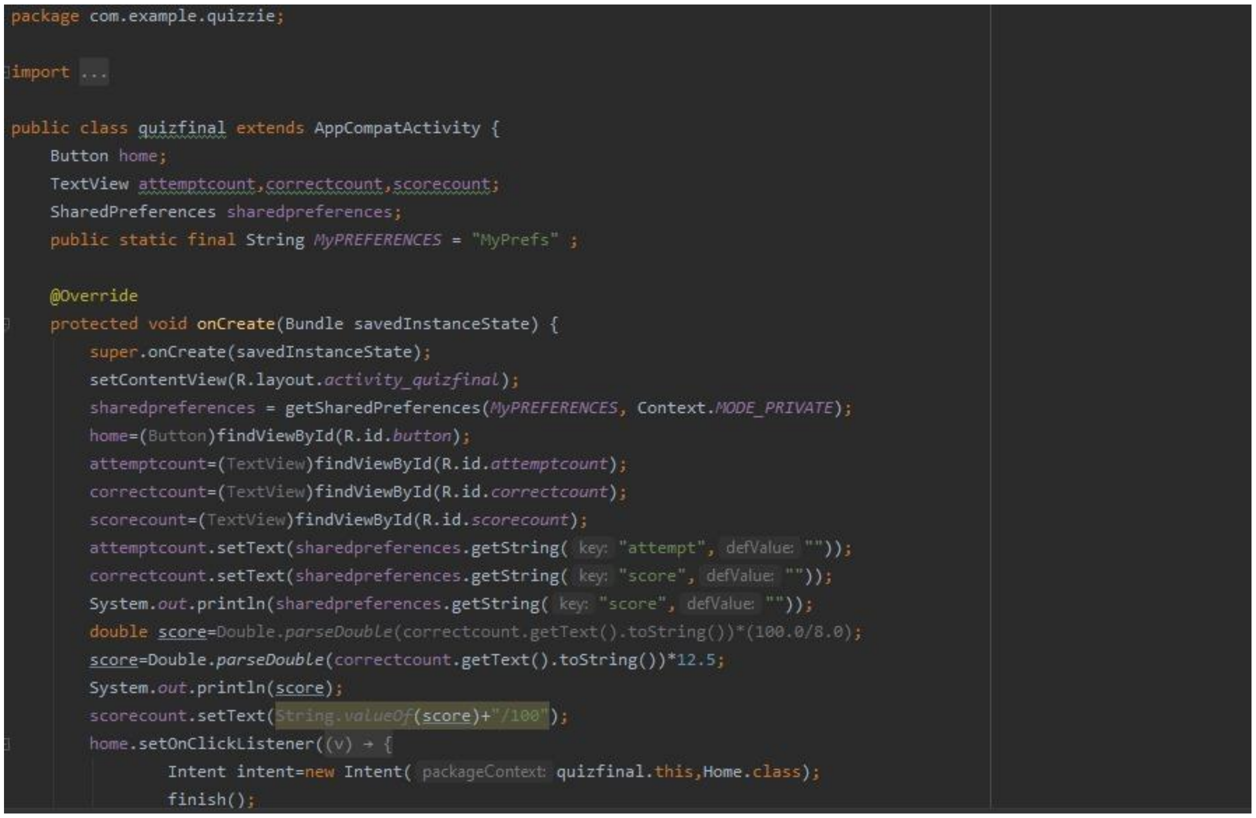
```
        phonenum.setOnFocusChangeListener((v, hasFocus) → {
                if(hasFocus)
                    phonenum.setHint("");
                else if(phonenum.getText().toString().isEmpty())
                    phonenum.setHint("Phone Number");
        });
        age.setOnFocusChangeListener((v, hasFocus) → {
                if(hasFocus)
                    age.setHint("");
                else if(age.getText().toString().isEmpty())
                    age.setHint("Age");
        });
    }
    private void redirect(View v){
        switch (v.getId()){
            case R.id.Register:
                fname=Fname.getText().toString();
                lname=Lname.getText().toString();
                mail=email.getText().toString();
                pass=password.getText().toString();
                phone=phonenum.getText().toString();
                Age=age.getText().toString();
                if (male.isChecked())
                    gender="Male";
                else
                    gender="Female";
                if(fname.isEmpty()||lname.isEmpty()||mail.isEmpty()||pass.isEmpty()||phone.isEmpty()||Age.isEmpty()||gender.isEmpty())
                    Toast.makeText( context: Register.this, text: "Fill all details",Toast.LENGTH_SHORT).show();
                else {
```



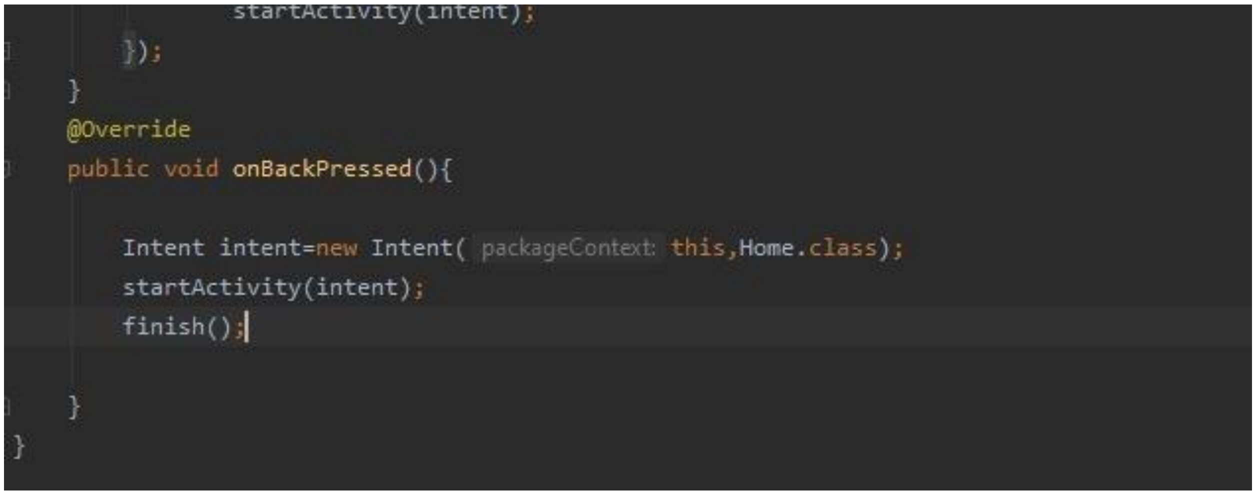
```
                Query query = myRef.child("User_Details").child(mail.replace( oldChar: '.', newChar: ' '));
                query.addListenerForSingleValueEvent(new ValueEventListener() {
                    @Override
                    public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                        if (dataSnapshot.exists()) {
                            System.out.println("reduntant");
                            email.setError("Mail already exists");
                            email.requestFocus();

                        } else
                            addUser();
                    }

                    @Override
                    public void onCancelled(@NonNull DatabaseError databaseError) {

                    }
                });
            }
            break;
        case R.id.male:
            female.setChecked(false);
            break;
        case R.id.female:
             male.setChecked(false);
             break;
    }

}
```



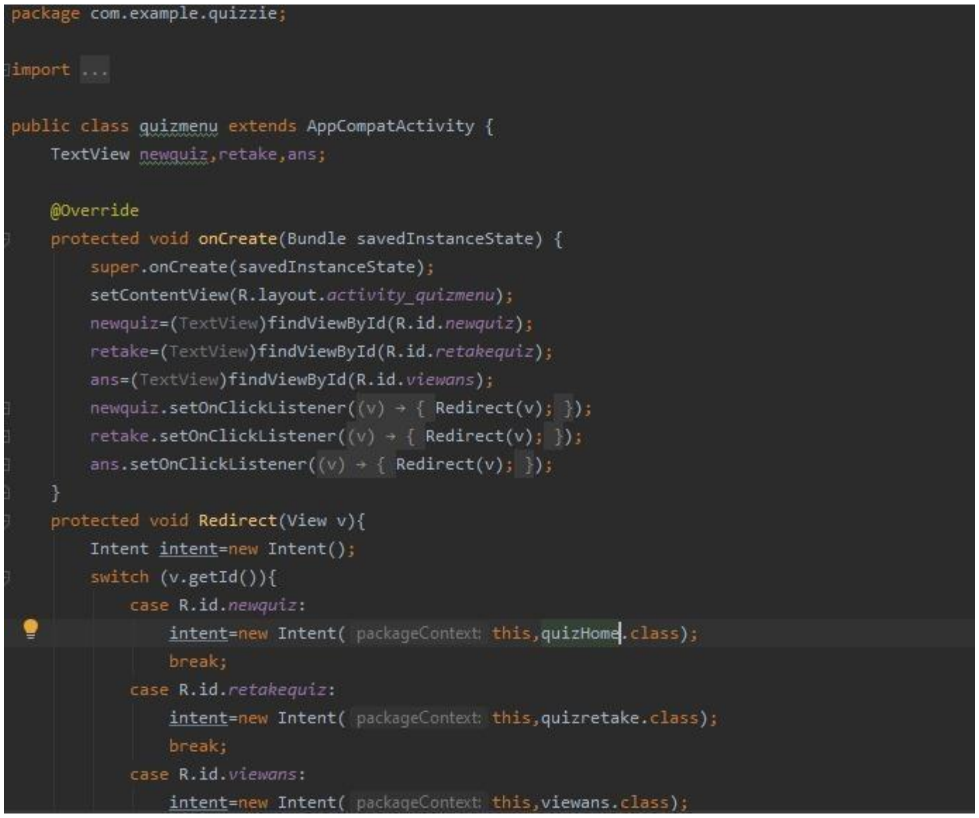
```
    private  void  message(String s) { Toast.makeText( context: this,s,Toast.LENGTH_LONG).show(); }
    private void addUser(){
        System.out.println("useradded");
        User user=new User(mail.replace( oldChar: '.', newChar: ' '),fname,lname,mail,pass,phone,Age,gender);
        myRef.child("User_Details").child(mail.replace( oldChar: '.', newChar: ' ')).setValue(user);
        Toast.makeText( context: this, text: "Registered Succesfully",Toast.LENGTH_LONG).show();
        Handler handler = new Handler();
        handler.postDelayed(() → {
                register.setVisibility(View.INVISIBLE);
                Intent intent=new Intent( packageContext: Register.this,Login.class);
                startActivity(intent);
//                my_button.setBackgroundResource(R.drawable.defaultcard);
        }, delayMillis: 3000);
    }
}
```



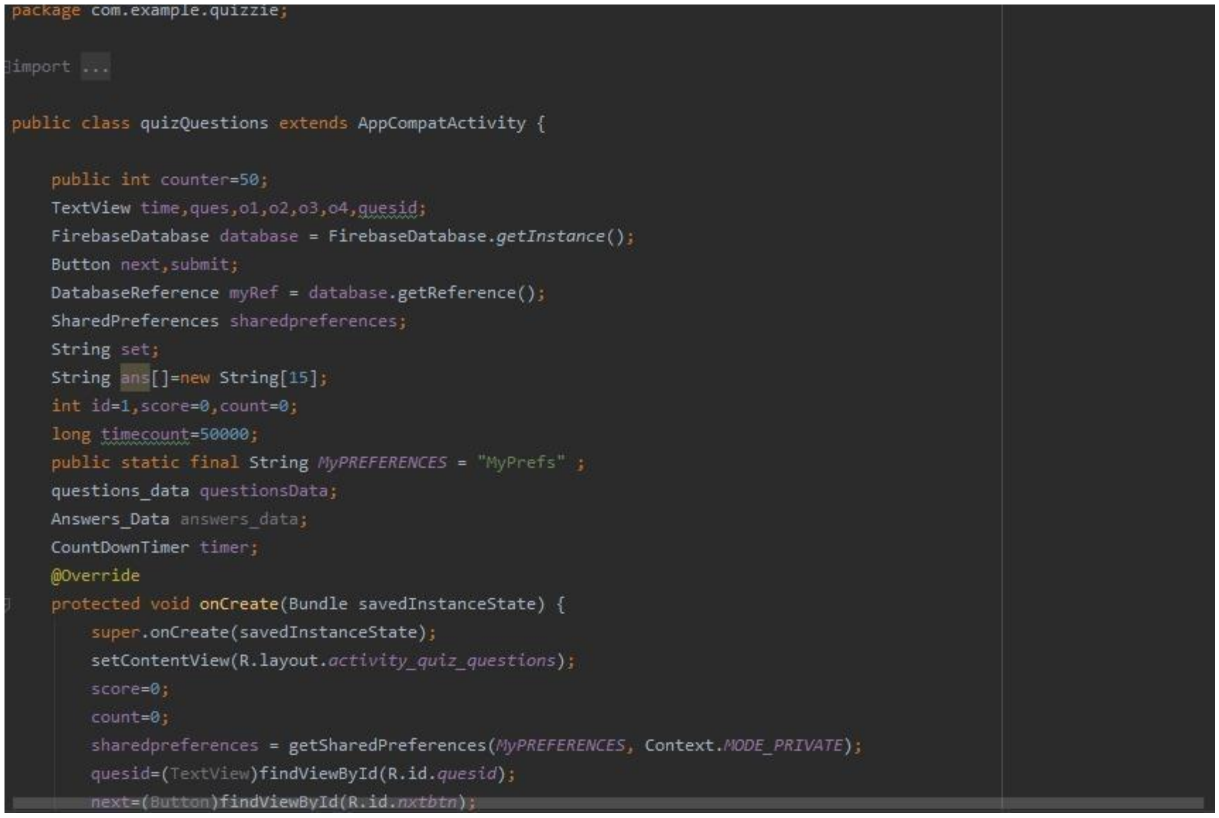
```
package com.example.quizzie;

import ...

public class quizfinal extends AppCompatActivity {
    Button home;
    TextView attemptcount,correctcount,scorecount;
    SharedPreferences sharedpreferences;
    public static final String MyPREFERENCES = "MyPrefs" ;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_quizfinal);
        sharedpreferences = getSharedPreferences(MyPREFERENCES, Context.MODE_PRIVATE);
        home=(Button)findViewById(R.id.button);
        attemptcount=(TextView)findViewById(R.id.attemptcount);
        correctcount=(TextView)findViewById(R.id.correctcount);
        scorecount=(TextView)findViewById(R.id.scorecount);
        attemptcount.setText(sharedpreferences.getString( key: "attempt", defValue: ""));
        correctcount.setText(sharedpreferences.getString( key: "score", defValue: ""));
        System.out.println(sharedpreferences.getString( key: "score", defValue: ""));
        double score=Double.parseDouble(correctcount.getText().toString())*(100.0/8.0);
        score=Double.parseDouble(correctcount.getText().toString())*12.5;
        System.out.println(score);
        scorecount.setText(String.valueOf(score)+"/100");
        home.setOnClickListener((v) → {
                Intent intent=new Intent( packageContext: quizfinal.this,Home.class);
                finish();
```



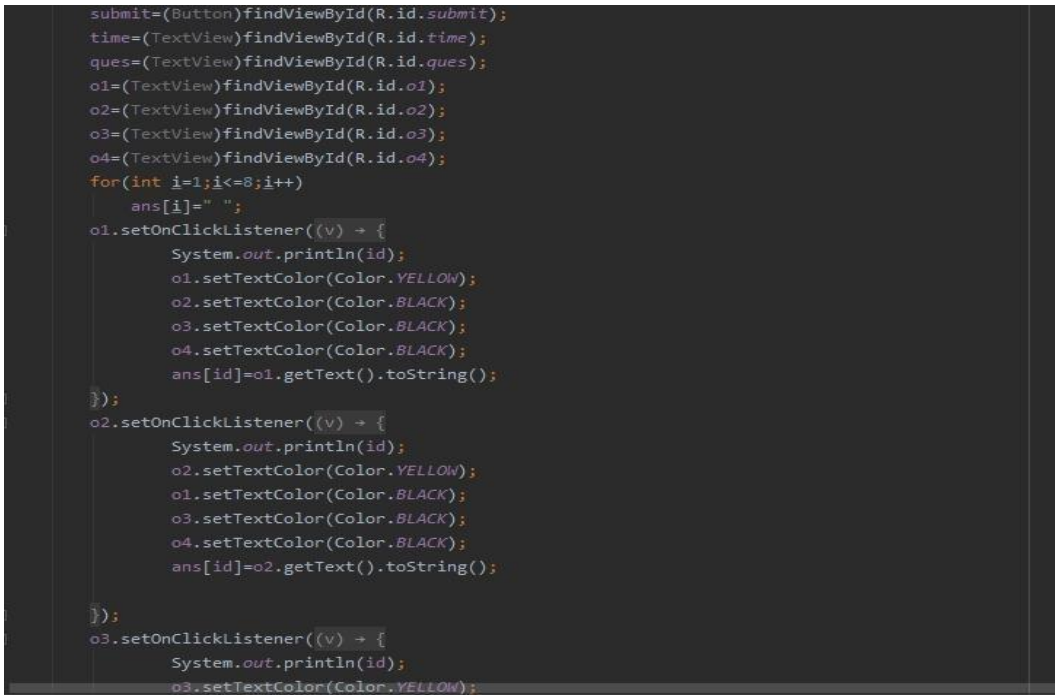
```
                startActivity(intent);
        });
    }
    @Override
    public void onBackPressed(){

        Intent intent=new Intent( packageContext: this,Home.class);
        startActivity(intent);
        finish();

    }
}
```



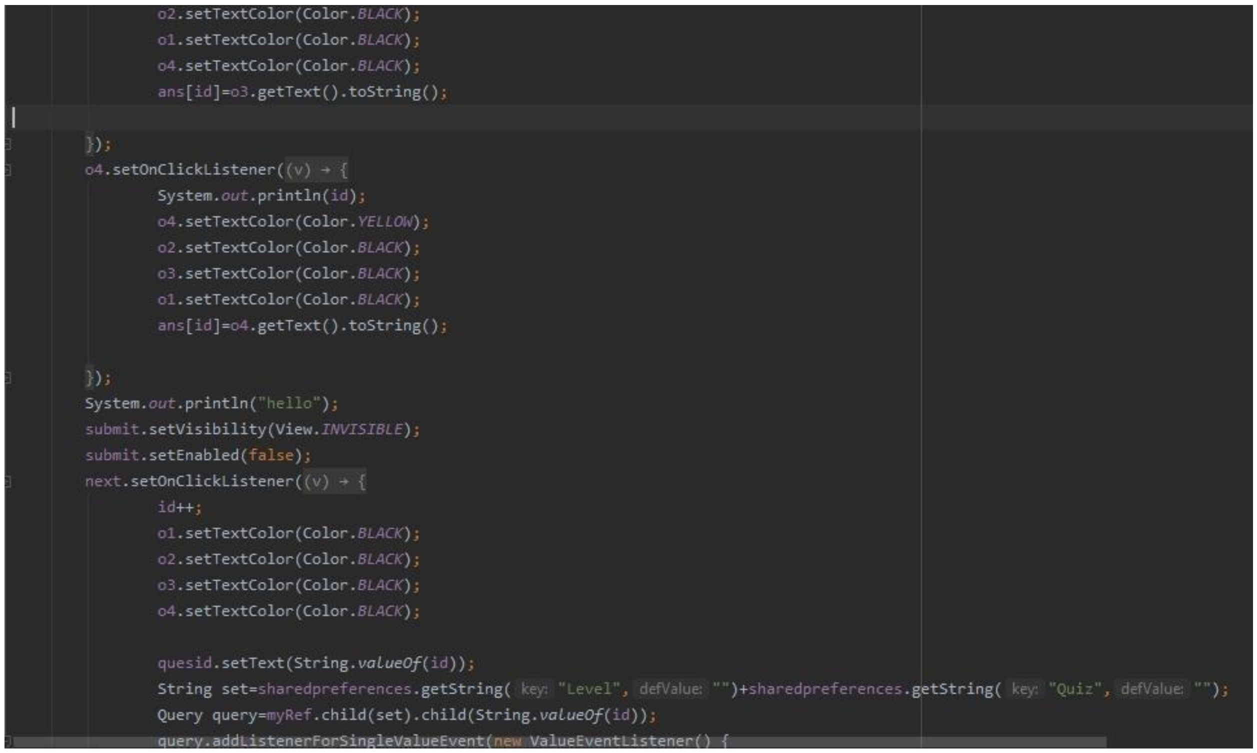
```java
package com.example.quizzie;

import ...

public class quizmenu extends AppCompatActivity {
    TextView newquiz,retake,ans;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_quizmenu);
        newquiz=(TextView)findViewById(R.id.newquiz);
        retake=(TextView)findViewById(R.id.retakequiz);
        ans=(TextView)findViewById(R.id.viewans);
        newquiz.setOnClickListener((v) → { Redirect(v); });
        retake.setOnClickListener((v) → { Redirect(v); });
        ans.setOnClickListener((v) → { Redirect(v); });
    }
    protected void Redirect(View v){
        Intent intent=new Intent();
        switch (v.getId()){
            case R.id.newquiz:
                intent=new Intent( packageContext: this,quizHome.class);
                break;
            case R.id.retakequiz:
                intent=new Intent( packageContext: this,quizretake.class);
                break;
            case R.id.viewans:
                intent=new Intent( packageContext: this,viewans.class);
```



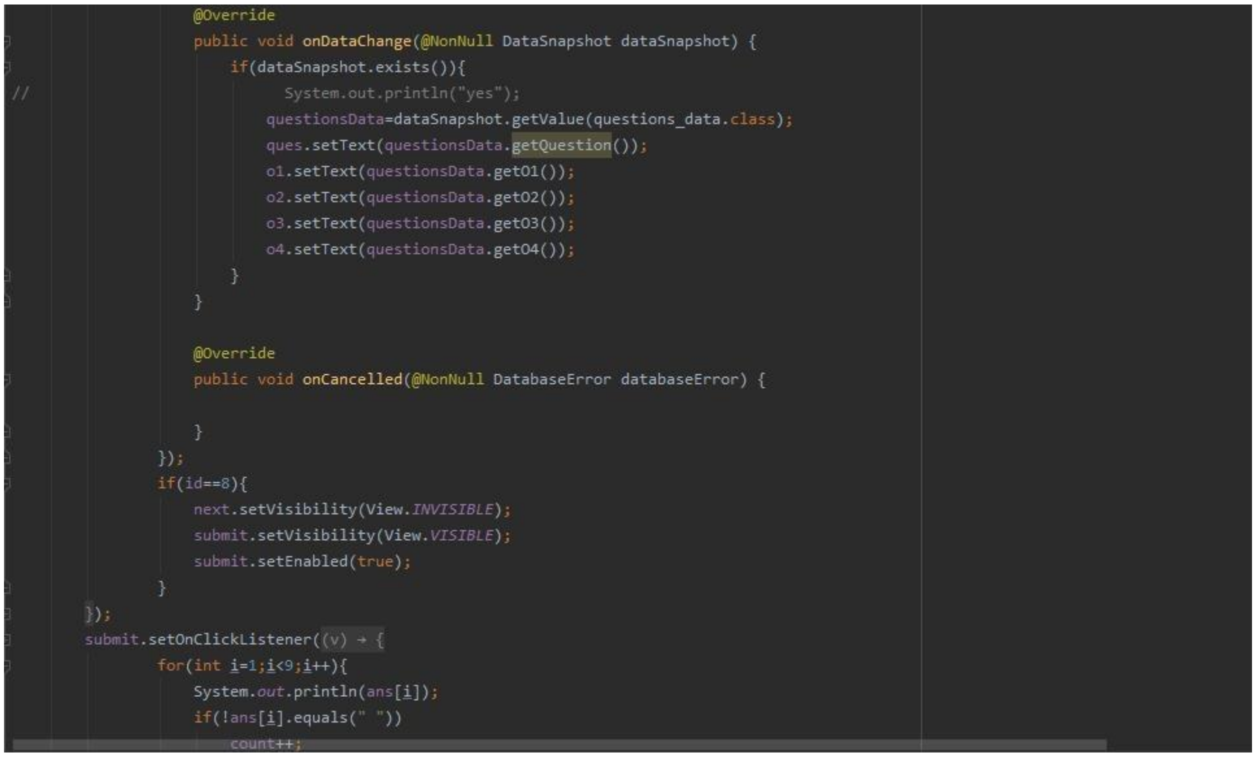
```java
package com.example.quizzie;

import ...

public class quizQuestions extends AppCompatActivity {

    public int counter=50;
    TextView time,ques,o1,o2,o3,o4,quesid;
    FirebaseDatabase database = FirebaseDatabase.getInstance();
    Button next,submit;
    DatabaseReference myRef = database.getReference();
    SharedPreferences sharedpreferences;
    String set;
    String ans[]=new String[15];
    int id=1,score=0,count=0;
    long timecount=50000;
    public static final String MyPREFERENCES = "MyPrefs" ;
    questions_data questionsData;
    Answers_Data answers_data;
    CountDownTimer timer;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_quiz_questions);
        score=0;
        count=0;
        sharedpreferences = getSharedPreferences(MyPREFERENCES, Context.MODE_PRIVATE);
        quesid=(TextView)findViewById(R.id.quesid);
        next=(Button)findViewById(R.id.nxtbtn);
```



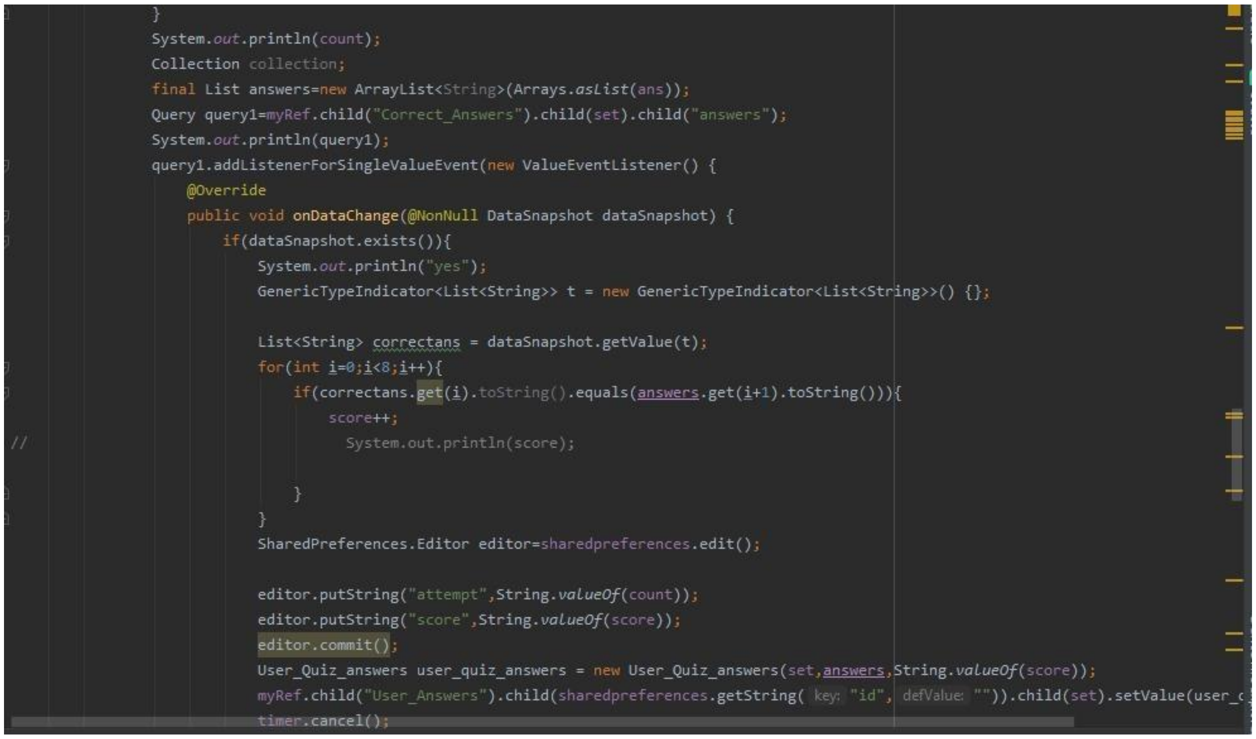
```
        submit=(Button)findViewById(R.id.submit);
        time=(TextView)findViewById(R.id.time);
        ques=(TextView)findViewById(R.id.ques);
        o1=(TextView)findViewById(R.id.o1);
        o2=(TextView)findViewById(R.id.o2);
        o3=(TextView)findViewById(R.id.o3);
        o4=(TextView)findViewById(R.id.o4);
        for(int i=1;i<=8;i++)
            ans[i]=" ";
        o1.setOnClickListener((v) → {
                System.out.println(id);
                o1.setTextColor(Color.YELLOW);
                o2.setTextColor(Color.BLACK);
                o3.setTextColor(Color.BLACK);
                o4.setTextColor(Color.BLACK);
                ans[id]=o1.getText().toString();
        });
        o2.setOnClickListener((v) → {
                System.out.println(id);
                o2.setTextColor(Color.YELLOW);
                o1.setTextColor(Color.BLACK);
                o3.setTextColor(Color.BLACK);
                o4.setTextColor(Color.BLACK);
                ans[id]=o2.getText().toString();

        });
        o3.setOnClickListener((v) → {
                System.out.println(id);
                o3.setTextColor(Color.YELLOW);
```



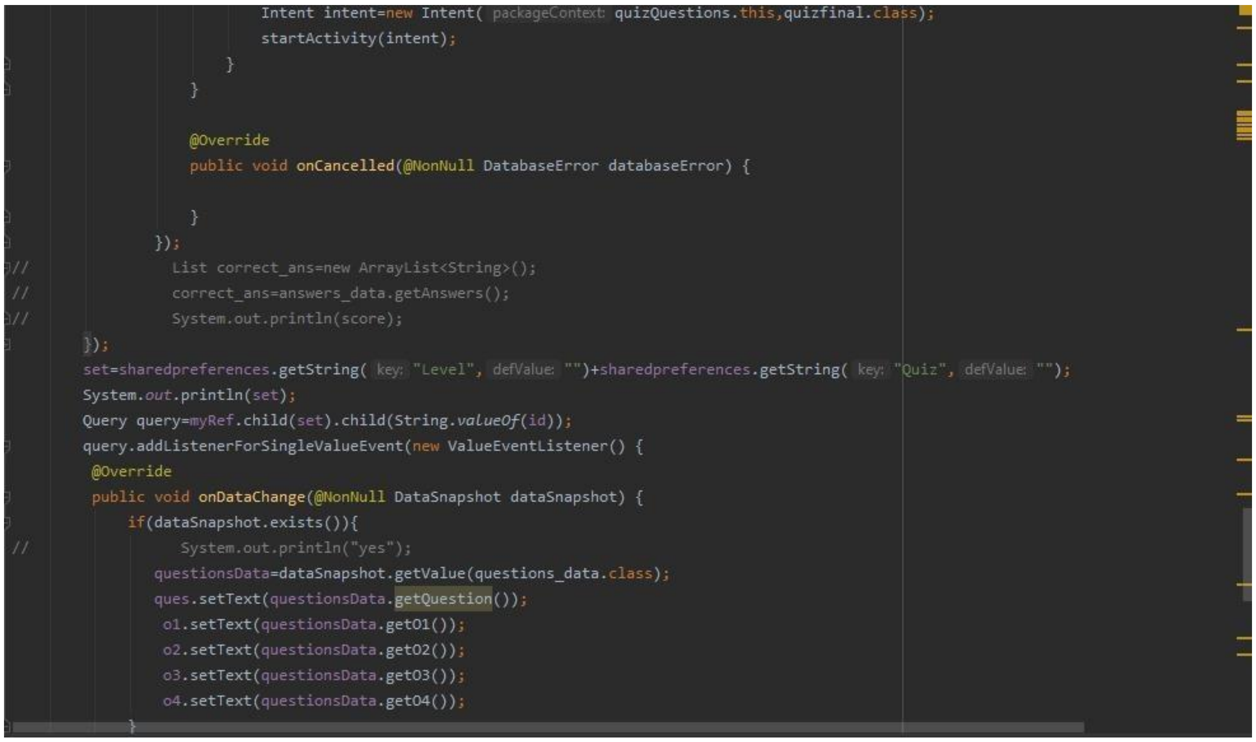
```
                o2.setTextColor(Color.BLACK);
                o1.setTextColor(Color.BLACK);
                o4.setTextColor(Color.BLACK);
                ans[id]=o3.getText().toString();

        });
        o4.setOnClickListener((v) → {
                System.out.println(id);
                o4.setTextColor(Color.YELLOW);
                o2.setTextColor(Color.BLACK);
                o3.setTextColor(Color.BLACK);
                o1.setTextColor(Color.BLACK);
                ans[id]=o4.getText().toString();

        });
        System.out.println("hello");
        submit.setVisibility(View.INVISIBLE);
        submit.setEnabled(false);
        next.setOnClickListener((v) → {
                id++;
                o1.setTextColor(Color.BLACK);
                o2.setTextColor(Color.BLACK);
                o3.setTextColor(Color.BLACK);
                o4.setTextColor(Color.BLACK);

                quesid.setText(String.valueOf(id));
                String set=sharedpreferences.getString( key: "Level", defValue: "")+sharedpreferences.getString( key: "Quiz", defValue: "");
                Query query=myRef.child(set).child(String.valueOf(id));
                query.addListenerForSingleValueEvent(new ValueEventListener() {
```



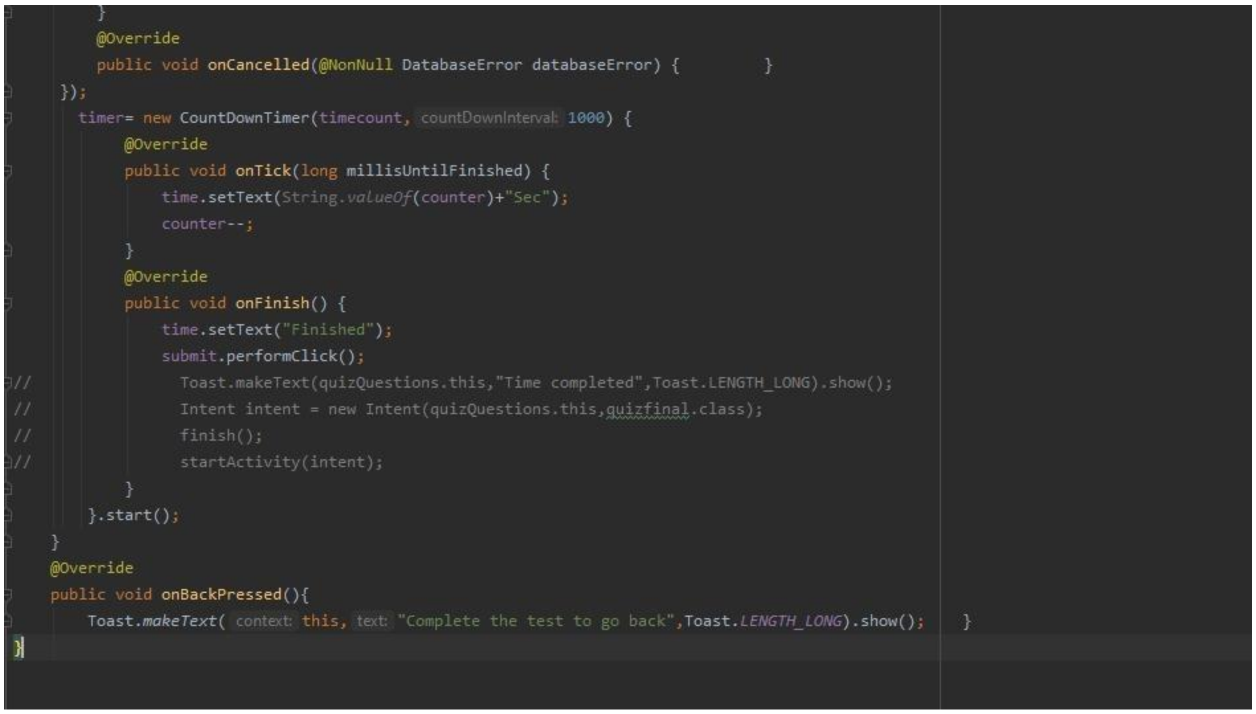
```java
                    @Override
                    public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                        if(dataSnapshot.exists()){
//                              System.out.println("yes");
                            questionsData=dataSnapshot.getValue(questions_data.class);
                            ques.setText(questionsData.getQuestion());
                            o1.setText(questionsData.get01());
                            o2.setText(questionsData.get02());
                            o3.setText(questionsData.get03());
                            o4.setText(questionsData.get04());
                        }
                    }

                    @Override
                    public void onCancelled(@NonNull DatabaseError databaseError) {

                    }
                });
                if(id==8){
                    next.setVisibility(View.INVISIBLE);
                    submit.setVisibility(View.VISIBLE);
                    submit.setEnabled(true);
                }
        });
        submit.setOnClickListener((v) → {
                for(int i=1;i<9;i++){
                    System.out.println(ans[i]);
                    if(!ans[i].equals(" "))
                        count++;
```



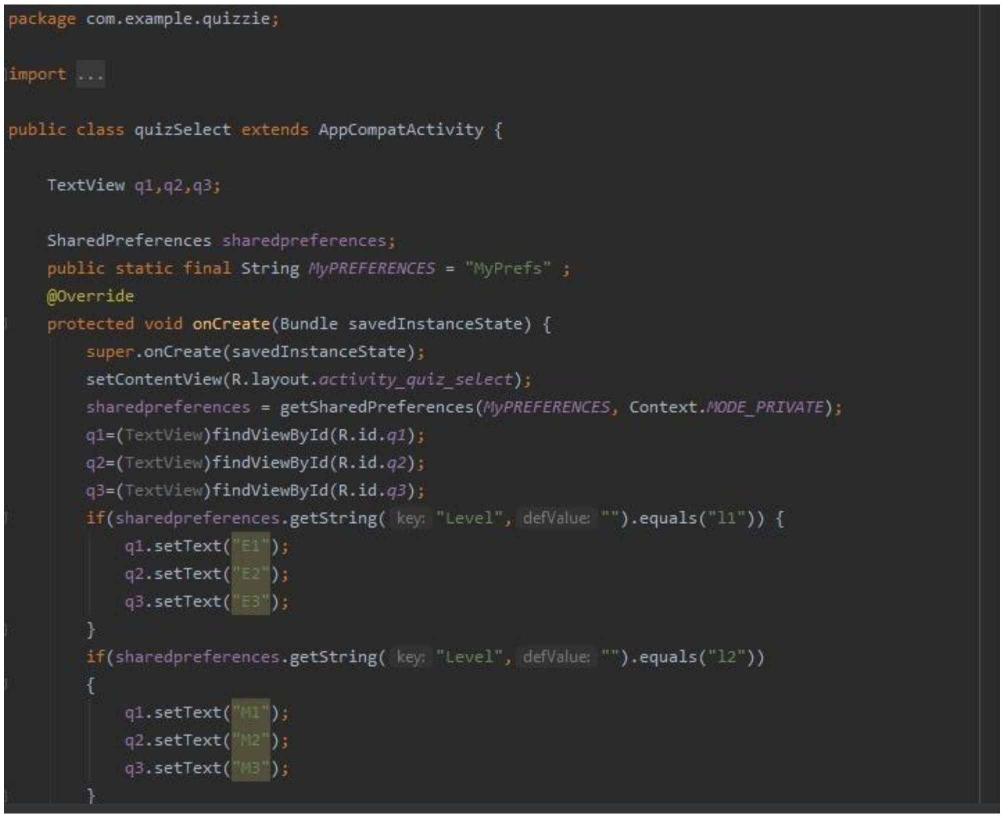
```java
                }
                System.out.println(count);
                Collection collection;
                final List answers=new ArrayList<String>(Arrays.asList(ans));
                Query query1=myRef.child("Correct_Answers").child(set).child("answers");
                System.out.println(query1);
                query1.addListenerForSingleValueEvent(new ValueEventListener() {
                    @Override
                    public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                        if(dataSnapshot.exists()){
                            System.out.println("yes");
                            GenericTypeIndicator<List<String>> t = new GenericTypeIndicator<List<String>>() {};

                            List<String> correctans = dataSnapshot.getValue(t);
                            for(int i=0;i<8;i++){
                                if(correctans.get(i).toString().equals(answers.get(i+1).toString())){
                                    score++;
//                                    System.out.println(score);

                                }
                            }
                            SharedPreferences.Editor editor=sharedpreferences.edit();

                            editor.putString("attempt",String.valueOf(count));
                            editor.putString("score",String.valueOf(score));
                            editor.commit();
                            User_Quiz_answers user_quiz_answers = new User_Quiz_answers(set,answers,String.valueOf(score));
                            myRef.child("User_Answers").child(sharedpreferences.getString( key: "id", defValue: "")).child(set).setValue(user_
                            timer.cancel();
```



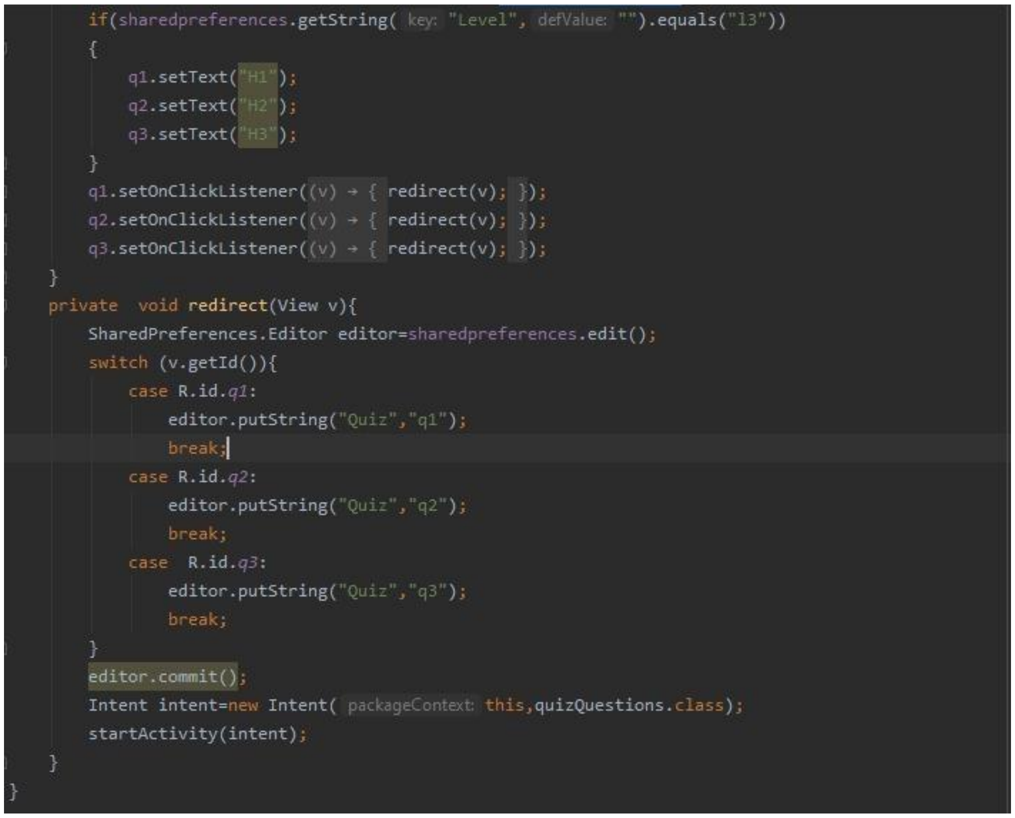
```
                        Intent intent=new Intent( packageContext: quizQuestions.this,quizfinal.class);
                        startActivity(intent);
                    }
                }

                @Override
                public void onCancelled(@NonNull DatabaseError databaseError) {

                }
            });
//              List correct_ans=new ArrayList<String>();
//              correct_ans=answers_data.getAnswers();
//              System.out.println(score);
        });
        set=sharedpreferences.getString( key: "Level", defValue: "")+sharedpreferences.getString( key: "Quiz", defValue: "");
        System.out.println(set);
        Query query=myRef.child(set).child(String.valueOf(id));
        query.addListenerForSingleValueEvent(new ValueEventListener() {
         @Override
         public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
             if(dataSnapshot.exists()){
//                   System.out.println("yes");
                 questionsData=dataSnapshot.getValue(questions_data.class);
                 ques.setText(questionsData.getQuestion());
                  o1.setText(questionsData.getO1());
                  o2.setText(questionsData.getO2());
                  o3.setText(questionsData.getO3());
                  o4.setText(questionsData.getO4());
             }
```



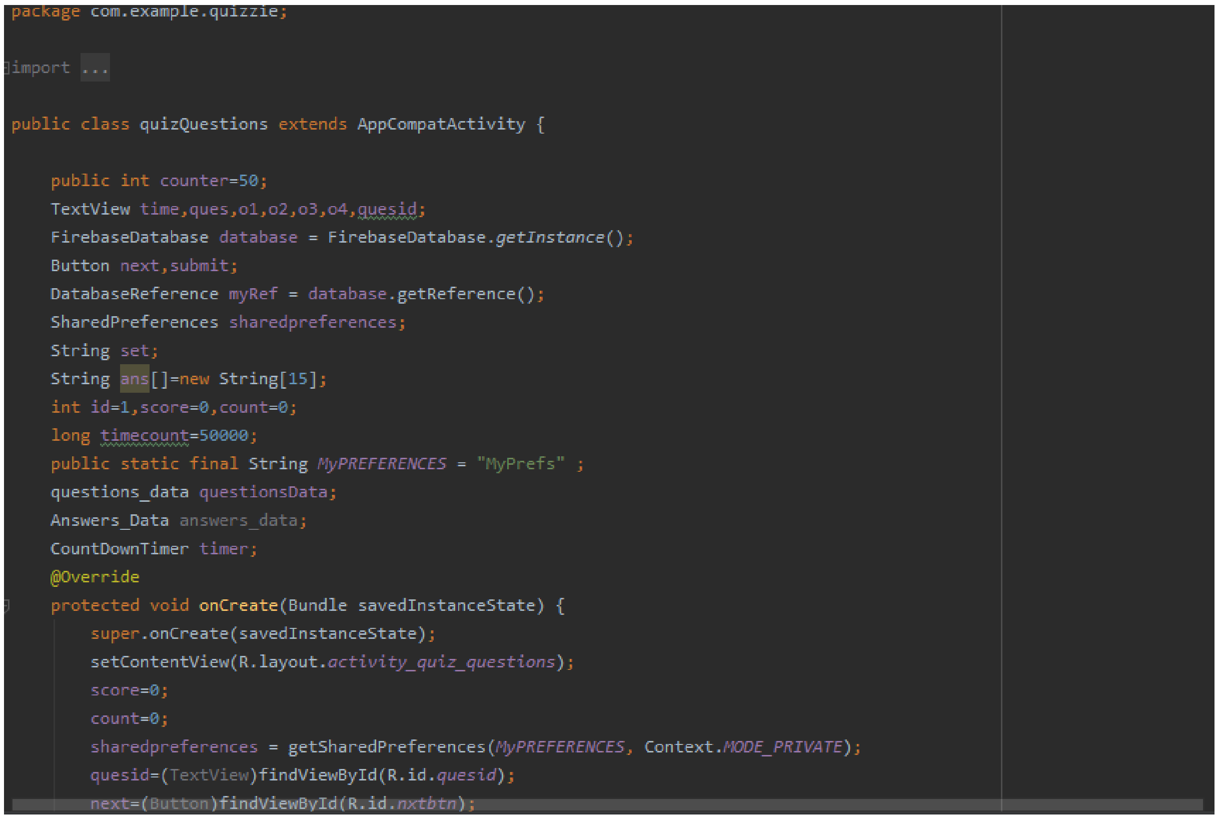
```
          }
          @Override
          public void onCancelled(@NonNull DatabaseError databaseError) {          }
      });
        timer= new CountDownTimer(timecount, countDownInterval: 1000) {
            @Override
            public void onTick(long millisUntilFinished) {
                time.setText(String.valueOf(counter)+"Sec");
                counter--;
            }
            @Override
            public void onFinish() {
                time.setText("Finished");
                submit.performClick();
//                Toast.makeText(quizQuestions.this,"Time completed",Toast.LENGTH_LONG).show();
//                Intent intent = new Intent(quizQuestions.this,quizfinal.class);
//                finish();
//                startActivity(intent);
            }
        }.start();
    }
    @Override
    public void onBackPressed(){
        Toast.makeText( context: this, text: "Complete the test to go back",Toast.LENGTH_LONG).show();    }
}
```



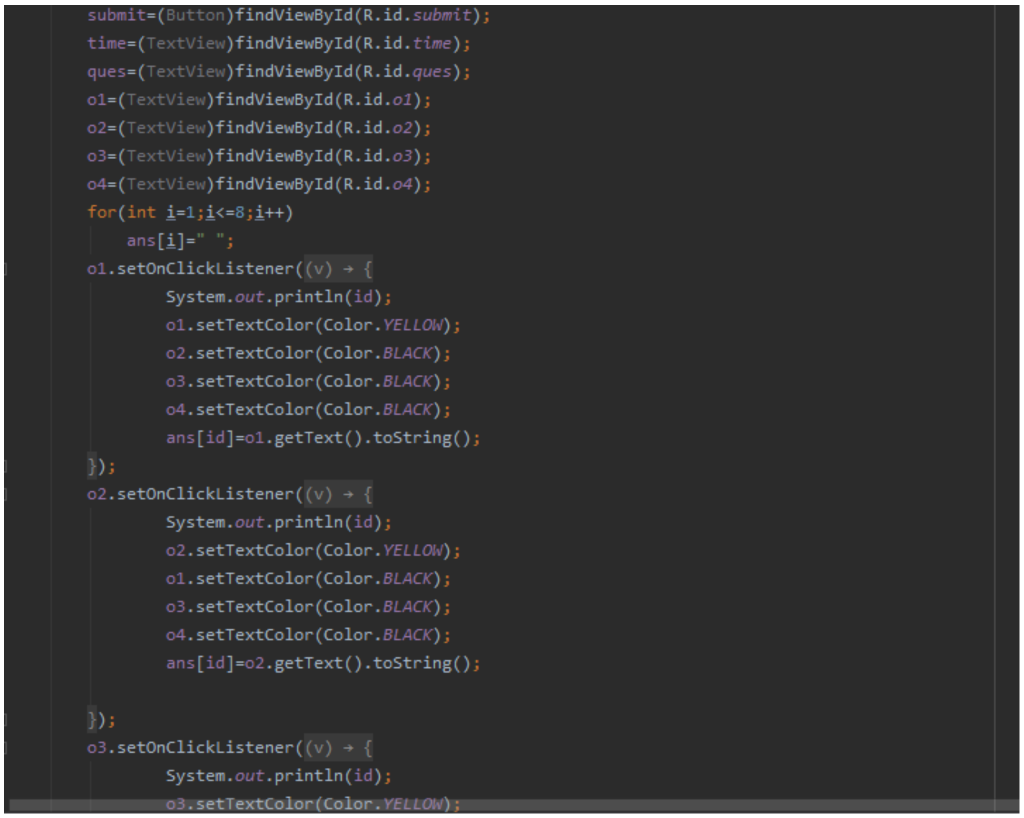
```java
package com.example.quizzie;

import ...

public class quizSelect extends AppCompatActivity {

    TextView q1,q2,q3;

    SharedPreferences sharedpreferences;
    public static final String MyPREFERENCES = "MyPrefs" ;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_quiz_select);
        sharedpreferences = getSharedPreferences(MyPREFERENCES, Context.MODE_PRIVATE);
        q1=(TextView)findViewById(R.id.q1);
        q2=(TextView)findViewById(R.id.q2);
        q3=(TextView)findViewById(R.id.q3);
        if(sharedpreferences.getString( key: "Level", defValue: "").equals("l1")) {
            q1.setText("E1");
            q2.setText("E2");
            q3.setText("E3");
        }
        if(sharedpreferences.getString( key: "Level", defValue: "").equals("l2"))
        {
            q1.setText("M1");
            q2.setText("M2");
            q3.setText("M3");
        }
```



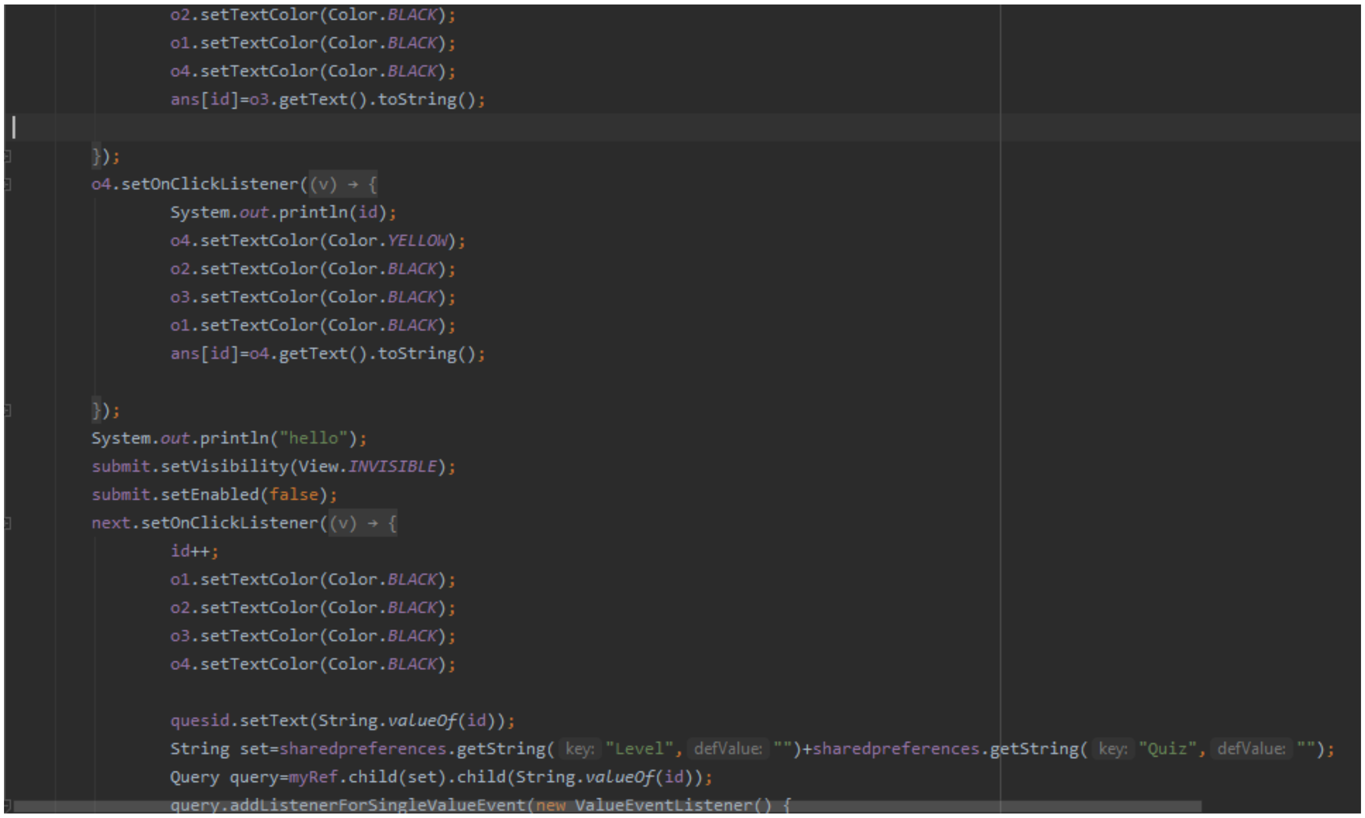
```java
        if(sharedpreferences.getString( key: "Level", defValue: "").equals("l3"))
        {
            q1.setText("H1");
            q2.setText("H2");
            q3.setText("H3");
        }
        q1.setOnClickListener((v) → { redirect(v); });
        q2.setOnClickListener((v) → { redirect(v); });
        q3.setOnClickListener((v) → { redirect(v); });
    }
    private  void redirect(View v){
        SharedPreferences.Editor editor=sharedpreferences.edit();
        switch (v.getId()){
            case R.id.q1:
                editor.putString("Quiz","q1");
                break;
            case R.id.q2:
                editor.putString("Quiz","q2");
                break;
            case  R.id.q3:
                editor.putString("Quiz","q3");
                break;
        }
        editor.commit();
        Intent intent=new Intent( packageContext: this,quizQuestions.class);
        startActivity(intent);
    }
}
```



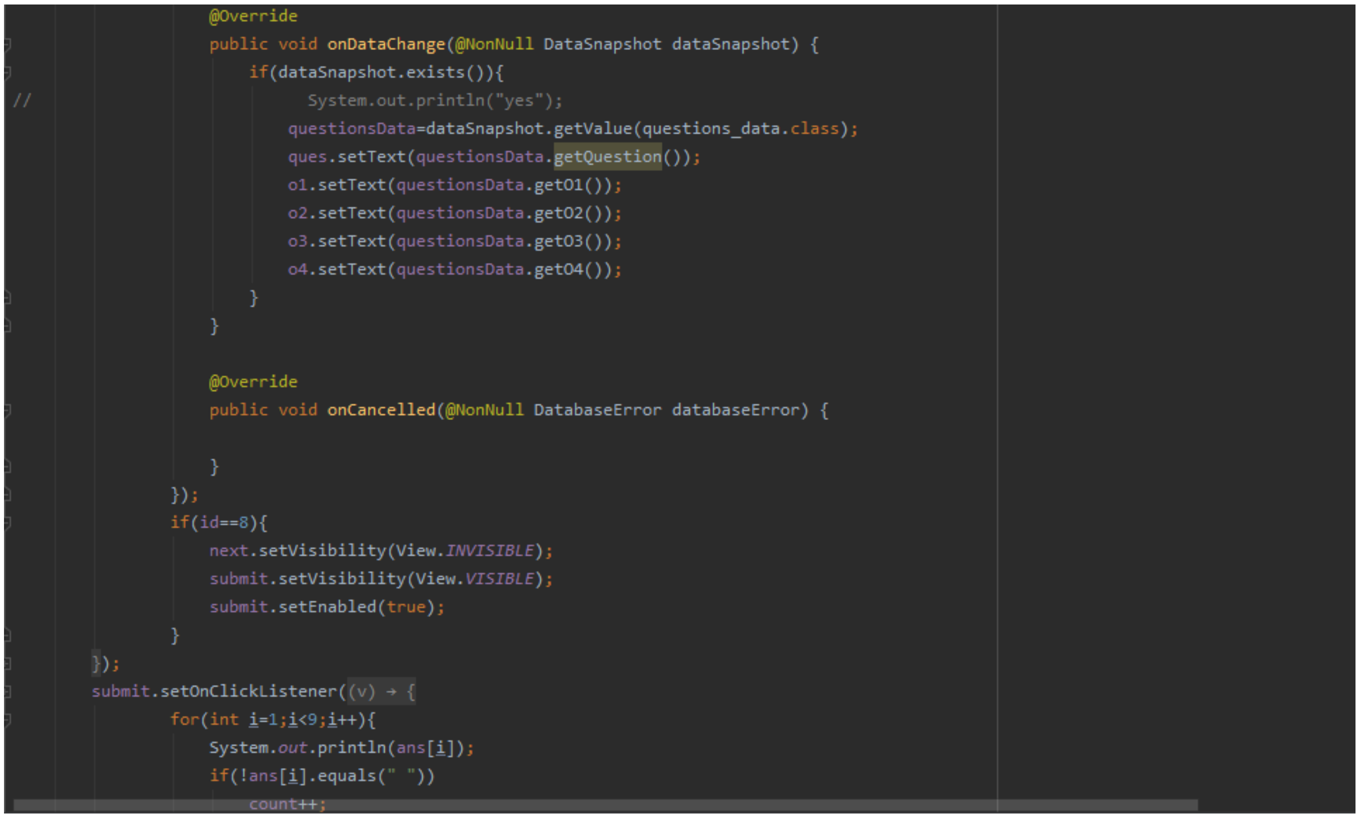
```
package com.example.quizzie;

import ...

public class quizQuestions extends AppCompatActivity {

    public int counter=50;
    TextView time,ques,o1,o2,o3,o4,quesid;
    FirebaseDatabase database = FirebaseDatabase.getInstance();
    Button next,submit;
    DatabaseReference myRef = database.getReference();
    SharedPreferences sharedpreferences;
    String set;
    String ans[]=new String[15];
    int id=1,score=0,count=0;
    long timecount=50000;
    public static final String MyPREFERENCES = "MyPrefs" ;
    questions_data questionsData;
    Answers_Data answers_data;
    CountDownTimer timer;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_quiz_questions);
        score=0;
        count=0;
        sharedpreferences = getSharedPreferences(MyPREFERENCES, Context.MODE_PRIVATE);
        quesid=(TextView)findViewById(R.id.quesid);
        next=(Button)findViewById(R.id.nxtbtn);
```



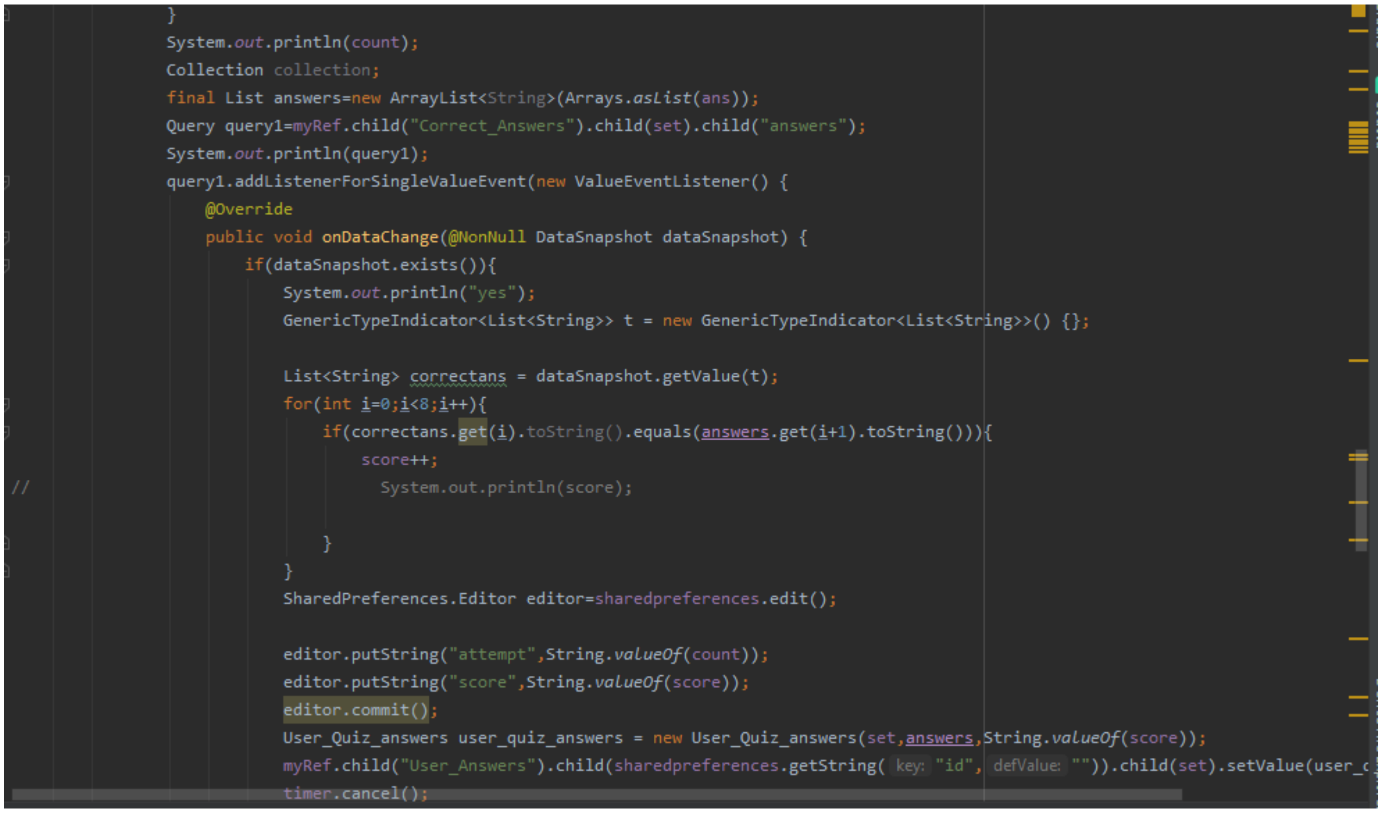
```
        submit=(Button)findViewById(R.id.submit);
        time=(TextView)findViewById(R.id.time);
        ques=(TextView)findViewById(R.id.ques);
        o1=(TextView)findViewById(R.id.o1);
        o2=(TextView)findViewById(R.id.o2);
        o3=(TextView)findViewById(R.id.o3);
        o4=(TextView)findViewById(R.id.o4);
        for(int i=1;i<=8;i++)
            ans[i]=" ";
        o1.setOnClickListener((v) → {
                System.out.println(id);
                o1.setTextColor(Color.YELLOW);
                o2.setTextColor(Color.BLACK);
                o3.setTextColor(Color.BLACK);
                o4.setTextColor(Color.BLACK);
                ans[id]=o1.getText().toString();
        });
        o2.setOnClickListener((v) → {
                System.out.println(id);
                o2.setTextColor(Color.YELLOW);
                o1.setTextColor(Color.BLACK);
                o3.setTextColor(Color.BLACK);
                o4.setTextColor(Color.BLACK);
                ans[id]=o2.getText().toString();

        });
        o3.setOnClickListener((v) → {
                System.out.println(id);
                o3.setTextColor(Color.YELLOW);
```



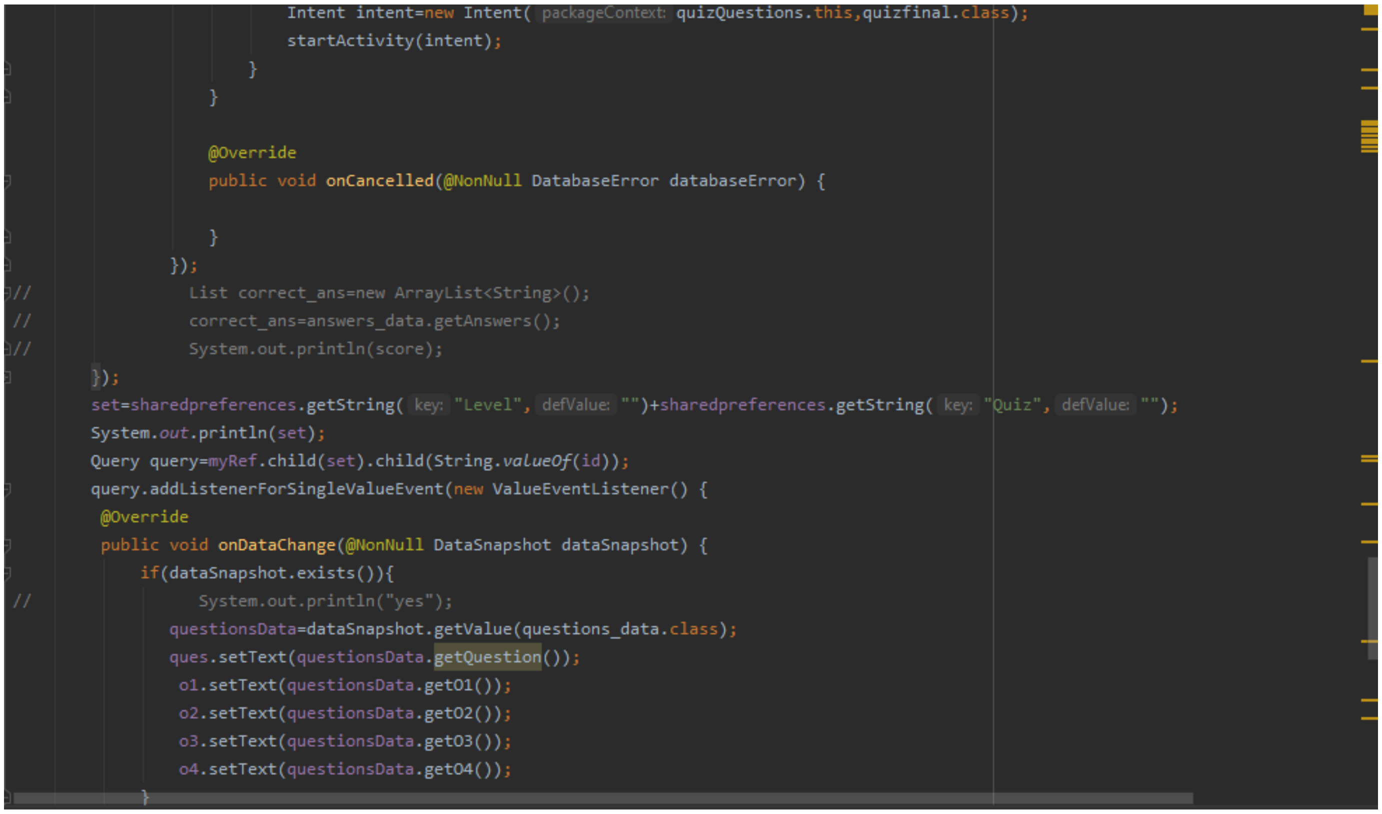
```
                o2.setTextColor(Color.BLACK);
                o1.setTextColor(Color.BLACK);
                o4.setTextColor(Color.BLACK);
                ans[id]=o3.getText().toString();

        });
        o4.setOnClickListener((v) → {
                System.out.println(id);
                o4.setTextColor(Color.YELLOW);
                o2.setTextColor(Color.BLACK);
                o3.setTextColor(Color.BLACK);
                o1.setTextColor(Color.BLACK);
                ans[id]=o4.getText().toString();

        });
        System.out.println("hello");
        submit.setVisibility(View.INVISIBLE);
        submit.setEnabled(false);
        next.setOnClickListener((v) → {
                id++;
                o1.setTextColor(Color.BLACK);
                o2.setTextColor(Color.BLACK);
                o3.setTextColor(Color.BLACK);
                o4.setTextColor(Color.BLACK);

                quesid.setText(String.valueOf(id));
                String set=sharedpreferences.getString( key: "Level", defValue: "")+sharedpreferences.getString( key: "Quiz", defValue: "");
                Query query=myRef.child(set).child(String.valueOf(id));
                query.addListenerForSingleValueEvent(new ValueEventListener() {
```



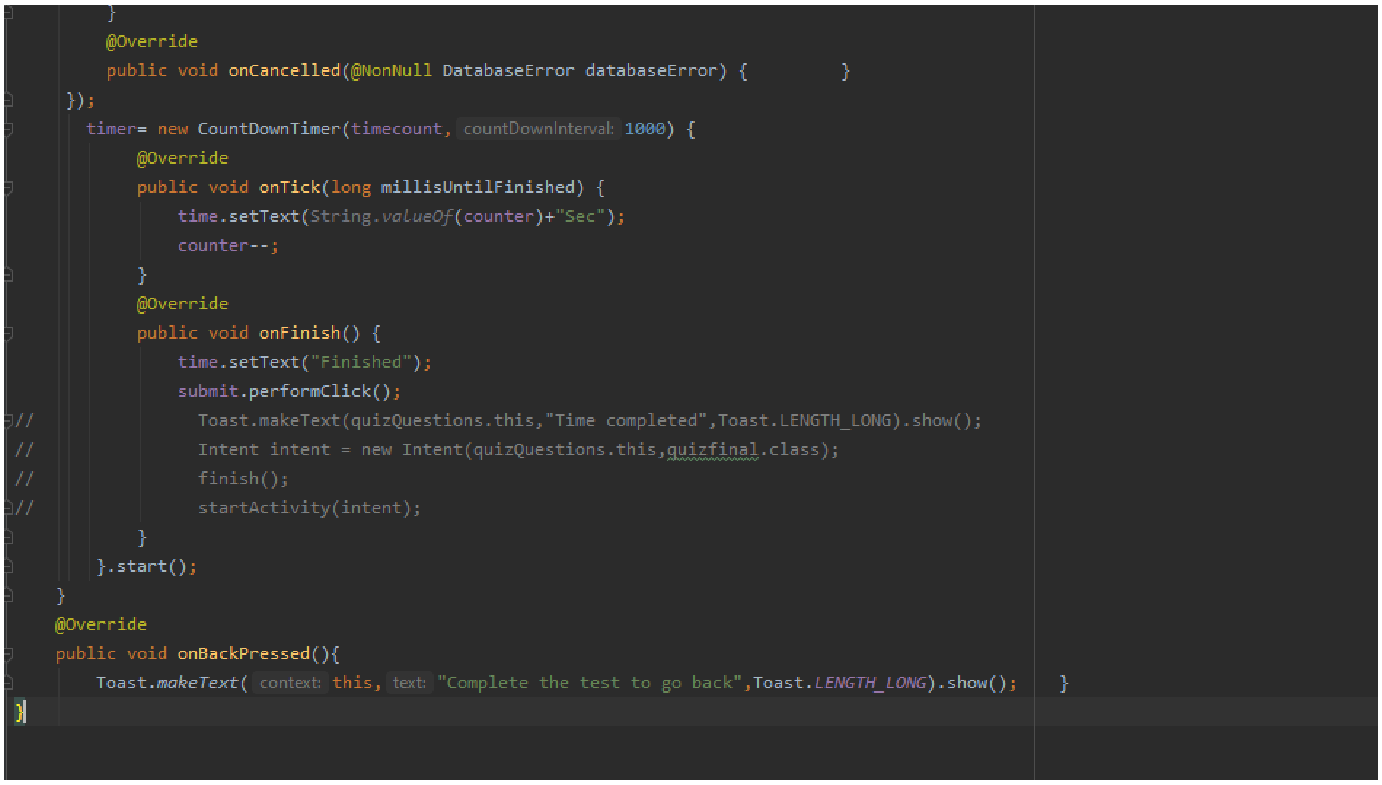
```
                    @Override
                    public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                        if(dataSnapshot.exists()){
//                            System.out.println("yes");
                            questionsData=dataSnapshot.getValue(questions_data.class);
                            ques.setText(questionsData.getQuestion());
                            o1.setText(questionsData.getO1());
                            o2.setText(questionsData.getO2());
                            o3.setText(questionsData.getO3());
                            o4.setText(questionsData.getO4());
                        }
                    }

                    @Override
                    public void onCancelled(@NonNull DatabaseError databaseError) {

                    }
                });
                if(id==8){
                    next.setVisibility(View.INVISIBLE);
                    submit.setVisibility(View.VISIBLE);
                    submit.setEnabled(true);
                }
        });
        submit.setOnClickListener((v) → {
                for(int i=1;i<9;i++){
                    System.out.println(ans[i]);
                    if(!ans[i].equals(" "))
                        count++;
```



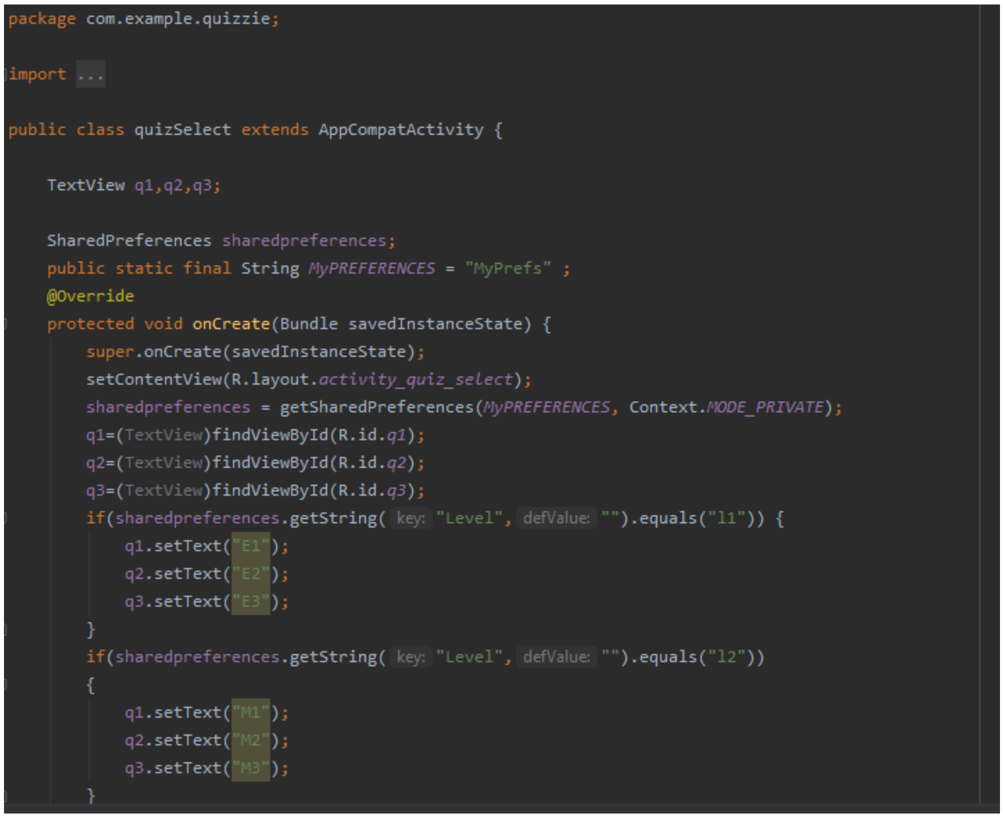
```
                }
                System.out.println(count);
                Collection collection;
                final List answers=new ArrayList<String>(Arrays.asList(ans));
                Query query1=myRef.child("Correct_Answers").child(set).child("answers");
                System.out.println(query1);
                query1.addListenerForSingleValueEvent(new ValueEventListener() {
                    @Override
                    public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                        if(dataSnapshot.exists()){
                            System.out.println("yes");
                            GenericTypeIndicator<List<String>> t = new GenericTypeIndicator<List<String>>() {};

                            List<String> correctans = dataSnapshot.getValue(t);
                            for(int i=0;i<8;i++){
                                if(correctans.get(i).toString().equals(answers.get(i+1).toString())){
                                    score++;
//                                    System.out.println(score);

                                }
                            }
                            SharedPreferences.Editor editor=sharedpreferences.edit();

                            editor.putString("attempt",String.valueOf(count));
                            editor.putString("score",String.valueOf(score));
                            editor.commit();
                            User_Quiz_answers user_quiz_answers = new User_Quiz_answers(set,answers,String.valueOf(score));
                            myRef.child("User_Answers").child(sharedpreferences.getString( key: "id", defValue: "")).child(set).setValue(user_c
                            timer.cancel();
```



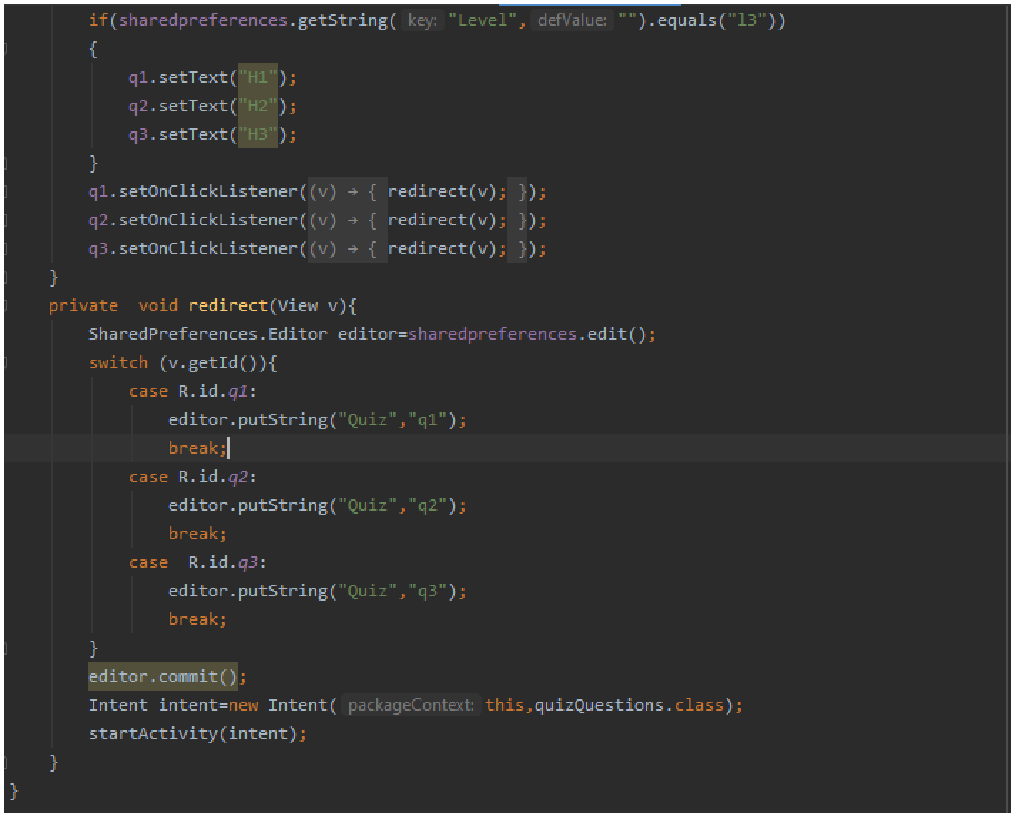
```
                            Intent intent=new Intent( packageContext: quizQuestions.this,quizfinal.class);
                            startActivity(intent);
                        }
                    }

                    @Override
                    public void onCancelled(@NonNull DatabaseError databaseError) {

                    }
                });
//                List correct_ans=new ArrayList<String>();
//                correct_ans=answers_data.getAnswers();
//                System.out.println(score);
        });
        set=sharedpreferences.getString( key: "Level", defValue: "")+sharedpreferences.getString( key: "Quiz", defValue: "");
        System.out.println(set);
        Query query=myRef.child(set).child(String.valueOf(id));
        query.addListenerForSingleValueEvent(new ValueEventListener() {
         @Override
         public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
             if(dataSnapshot.exists()){
//                   System.out.println("yes");
                questionsData=dataSnapshot.getValue(questions_data.class);
                ques.setText(questionsData.getQuestion());
                 o1.setText(questionsData.get01());
                 o2.setText(questionsData.get02());
                 o3.setText(questionsData.get03());
                 o4.setText(questionsData.get04());
             }
```



```
        }
        @Override
        public void onCancelled(@NonNull DatabaseError databaseError) {         }
    });
      timer= new CountDownTimer(timecount, countDownInterval: 1000) {
          @Override
          public void onTick(long millisUntilFinished) {
              time.setText(String.valueOf(counter)+"Sec");
              counter--;
          }
          @Override
          public void onFinish() {
              time.setText("Finished");
              submit.performClick();
//              Toast.makeText(quizQuestions.this,"Time completed",Toast.LENGTH_LONG).show();
//              Intent intent = new Intent(quizQuestions.this,quizfinal.class);
//              finish();
//              startActivity(intent);
          }
      }.start();
  }
  @Override
  public void onBackPressed(){
      Toast.makeText( context: this, text: "Complete the test to go back",Toast.LENGTH_LONG).show();    }
}
```



```
package com.example.quizzie;

import ...

public class quizSelect extends AppCompatActivity {

    TextView q1,q2,q3;

    SharedPreferences sharedpreferences;
    public static final String MyPREFERENCES = "MyPrefs" ;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_quiz_select);
        sharedpreferences = getSharedPreferences(MyPREFERENCES, Context.MODE_PRIVATE);
        q1=(TextView)findViewById(R.id.q1);
        q2=(TextView)findViewById(R.id.q2);
        q3=(TextView)findViewById(R.id.q3);
        if(sharedpreferences.getString( key: "Level", defValue: "").equals("l1")) {
            q1.setText("E1");
            q2.setText("E2");
            q3.setText("E3");
        }
        if(sharedpreferences.getString( key: "Level", defValue: "").equals("l2"))
        {
            q1.setText("M1");
            q2.setText("M2");
            q3.setText("M3");
        }
```


```
        if(sharedpreferences.getString( key: "Level", defValue: "").equals("13"))
        {
            q1.setText("H1");
            q2.setText("H2");
            q3.setText("H3");
        }
        q1.setOnClickListener((v) → { redirect(v); });
        q2.setOnClickListener((v) → { redirect(v); });
        q3.setOnClickListener((v) → { redirect(v); });
    }
    private  void redirect(View v){
        SharedPreferences.Editor editor=sharedpreferences.edit();
        switch (v.getId()){
            case R.id.q1:
                editor.putString("Quiz","q1");
                break;
            case R.id.q2:
                editor.putString("Quiz","q2");
                break;
            case  R.id.q3:
                editor.putString("Quiz","q3");
                break;
        }
        editor.commit();
        Intent intent=new Intent( packageContext: this,quizQuestions.class);
        startActivity(intent);
    }
}
```

You-tube Link:

https://youtu.be/g8wY
QkOFBy8