PROJECT REPORT ON

QUIZED

(UNDER THE PARTIAL FULFILLMENT OF THE UNIVERSITY FOR COURSE OF T.Y.BSC COMPUTER SCIENCE)

SUBMITTED BY:

Ms. JANHAVI SHARAD SHEDGE

GUIDED BY:

MS. BHOOMIKA PANSARE

DEPARTMENT OF COMPUTER SCIENCE
PARLE TILAK VIDYALAYA ASSOCIATION'S
MULUND COLLEGE OF COMMERCE S.N. ROAD,
MULUND (WEST), MUMBAI-80
UNIVERSITY OF MUMBAI 2022-2023

Plagiarism Checker X - Report

Originality Assessment

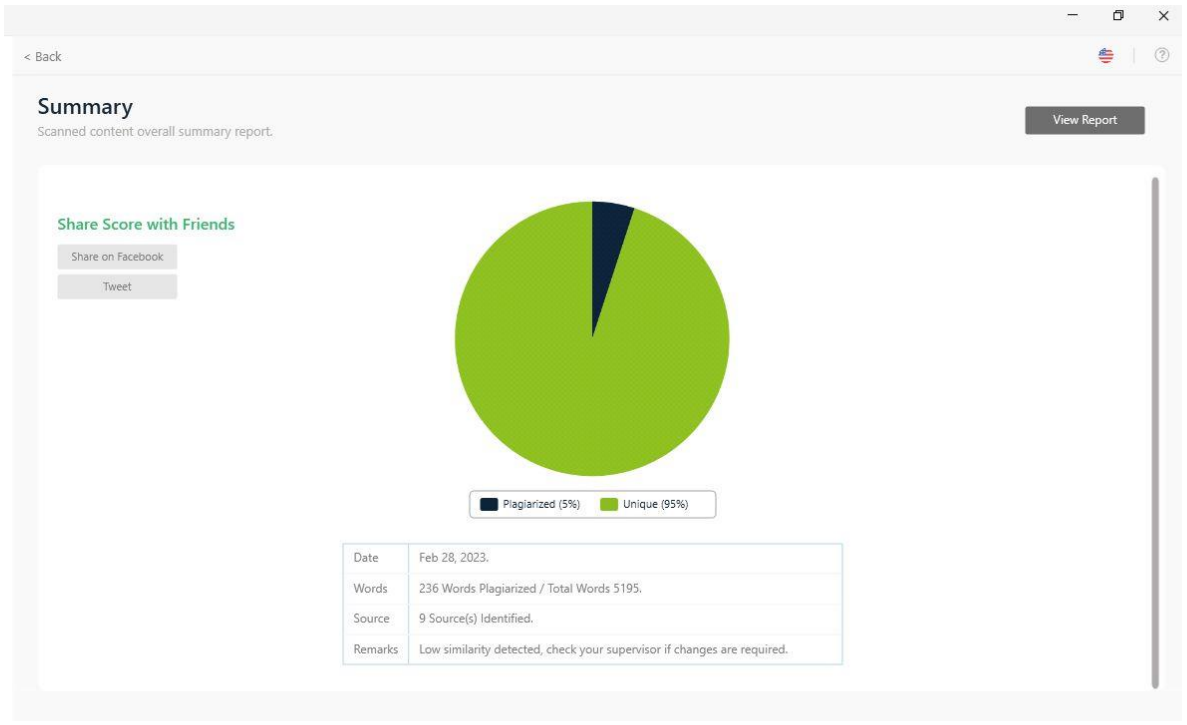
5%

Overall Similarity

Date: Feb 28, 2023
Matches: 236 / 5195 words
Sources: 9

Remarks: Low similarity detected, check with your supervisor if changes are required.

Verify Report:
Scan this QR Code



Summary

Scanned content overall summary report.

■ Plagiarized (5%) 🟩 Unique (95%)

Date	Feb 28, 2023.
Words	236 Words Plagiarized / Total Words 5195.
Source	9 Source(s) Identified.
Remarks	Low similarity detected, check your supervisor if changes are required.

ACKNOWLEDGEMENT

I like to extend my gratitude to Dr. Sonali Pednekar, our principal and all staff of Mulund College of Commerce for providing us moral support, conducive work environment and the much needed inspiration to complete this project on time.

I also take this opportunity to thank our Course Coordinator Dr. Reena Nagda and all the faculty members of Department of Computer Science for giving us the most needed guidance and continuous encouragement throughout the duration of the Programme.

I wish to extend my deepest gratitude and special thanks to my project guide Ms. Bhoomika Pansare. I perceive the skills and knowledge which I have gained throughout my project work as a very valuable component in my future career development.

I would like to convey my special thanks to the Management and all the staff of the college for providing the required infrastructure and resource to enable the completion and enrichment of my project.

I am also extremely grateful to the University of Mumbai for having prescribed this project work to me as a part of the academic requirement in the final year of Bachelor of Science in Computer Science.

Finally I thank all my friends and family members who have directly or indirectly helped me in completing my project.

Janhavi Sharad Shedge

INDEX

1.Title:

QuizEd

2. INTRODUCTION

2.1 Objective of the Project

Design a system for conducting multiple choice tests for an institution and automatically grade the students for the test

2.2 Description of the Current System

Institutions currently use a proper-based system where the questions are printed and the answers are eithher OCR graded or manually graded.

2.3 Limitations of Current System

Current System relies on the examiner to grade the papers,which is time consuming.The System consumes a lot of paper for every test being performed.There is nno way to display the grades of the students in a table unless it is manually mode.

2.4 Description of Proposed System

Proposed system makes use of web server to securely store the tests and manage all results.The teachers can make tests on the system and deploy the tests to students.Students can attempt tests deployed to them in the given time limit.

2.5 Advantages of Proposed System

Proposed system does not rely on the examiner to grade the tests, as soon as a student finishes his tests or runs out of time,the test is graded and the result is saved on the database.Each test has a dynamically generated page which displays the test results of the students automatically.

3. REQUIREMENT SPECIFICATION

3.1 Software Requirements:

- Application Software: Android Studio
- Front end:Java
- Backend: Firebase

3.2 Hardware Requirements:

- RAM – 4GB or 8GB
- 64 bit Operating System

3.3 Data Requirements:

- Email
- Password
- Phone Number

3.4 Fact Finding questions

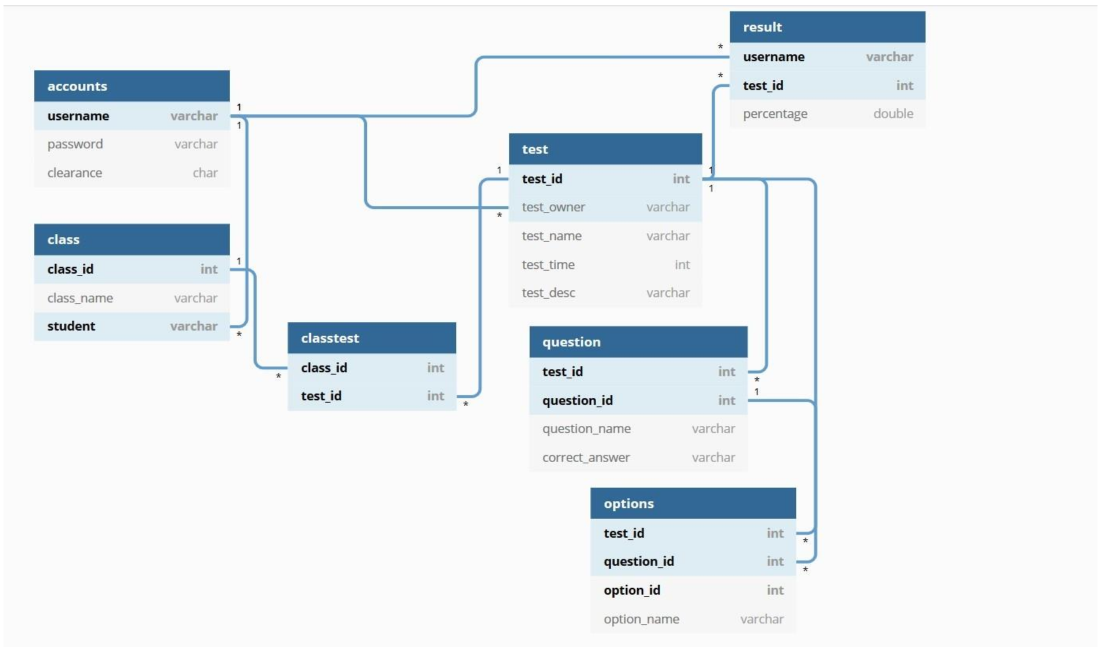
- Who will be accessing the quizzes?
- Who will be creating the quizzes?
- Will the time-limit for the quizzes differ from quiz-quiz?
- Will the number of questions differ from quiz-quiz?
- How will the quizes be graded?
- Do the results need to be segregrated?
- Who will be viewing the results?

4. SYSTEM DESIGN DETAILS

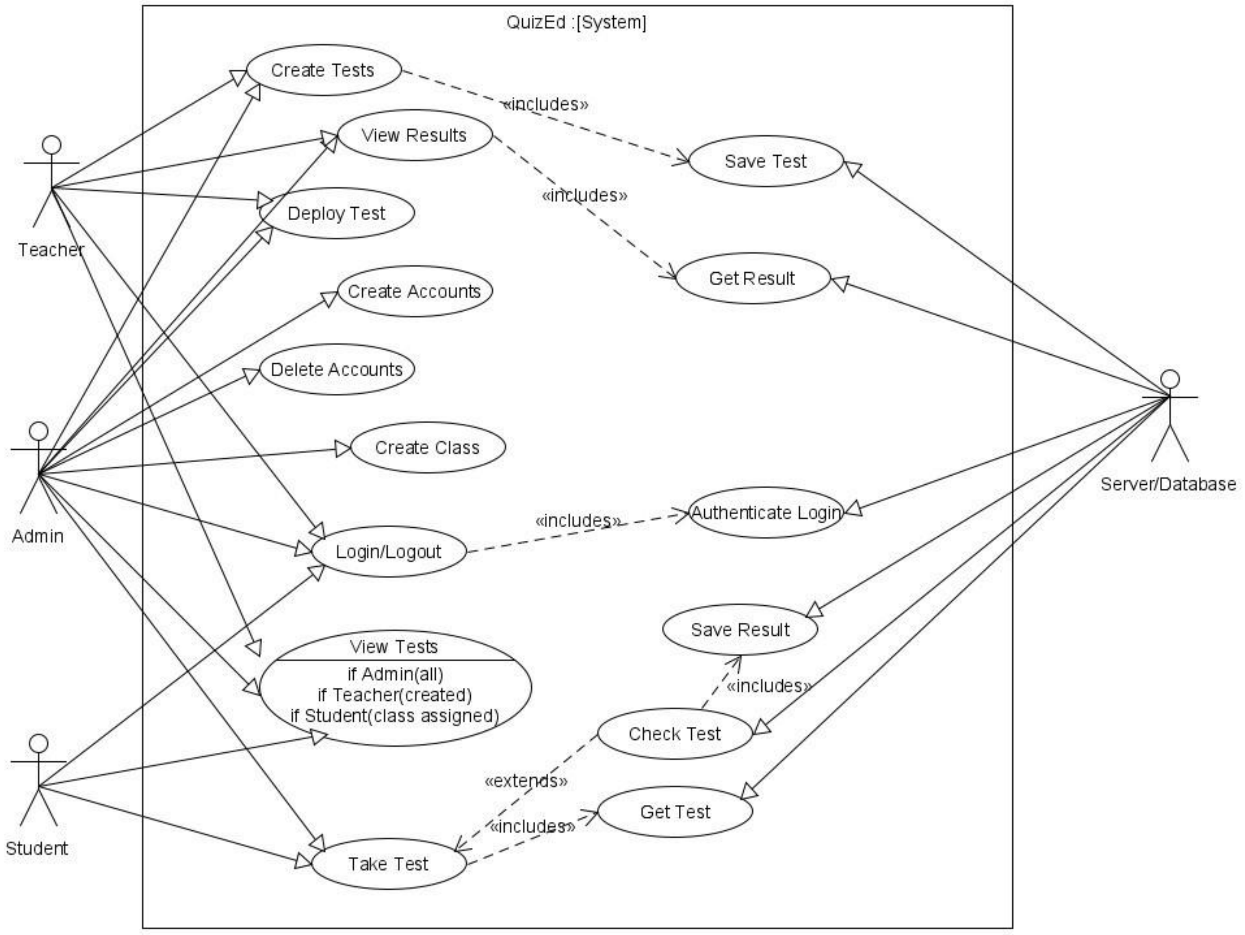
4.1 Event Table

No.	Event	Trigger	Source	Activity	Response	Destination
1.	Create Accounts	Div createaccount	admin/ teacher	Creates new account	Adds new account to db, redirect to home	Server
2.	Login	Index page	admin/ teacher/ student	Checks username and password to login	Redirect to home	Server
3.	Create Test	createtest hyperlinks	admin/ teacher	Creates a new test with the testowner as the user signed in	Redirect to home	Server
4.	Add Question	btnadd	admin/ teacher	Adds new question to create test menu	Displays new question in question list	Local
5.	Add Options	btnOadd	Admin/ teacher	Adds new option to the question selected	Displays a new option in the options list	Local
6.	Save Test	btnTestSubmit	Admin/ teacher	Saves created test	Saves Test details to database	Server
7.	Deploy Test	ddlDeploy	Admin/ teacher	Deploys test to student accounts	Deploys test to student accounts under selected class	Server

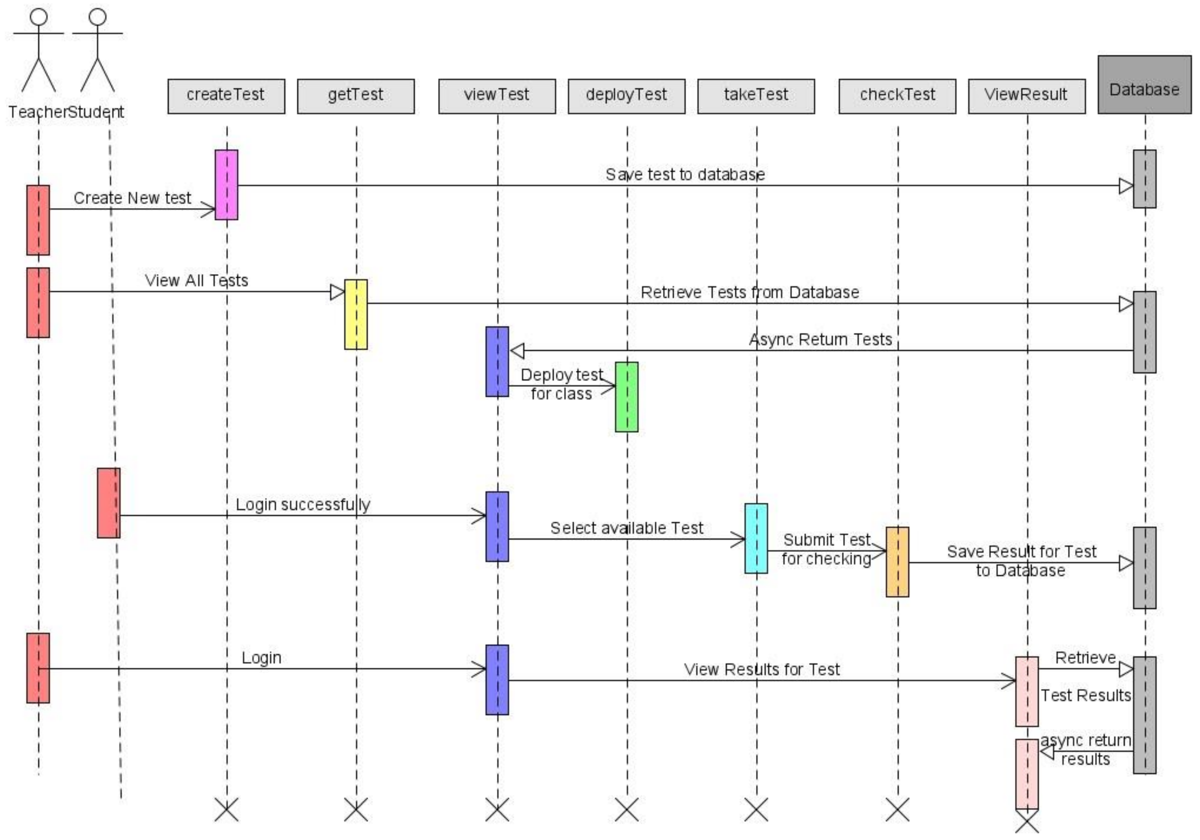
4.2 Class Diagram



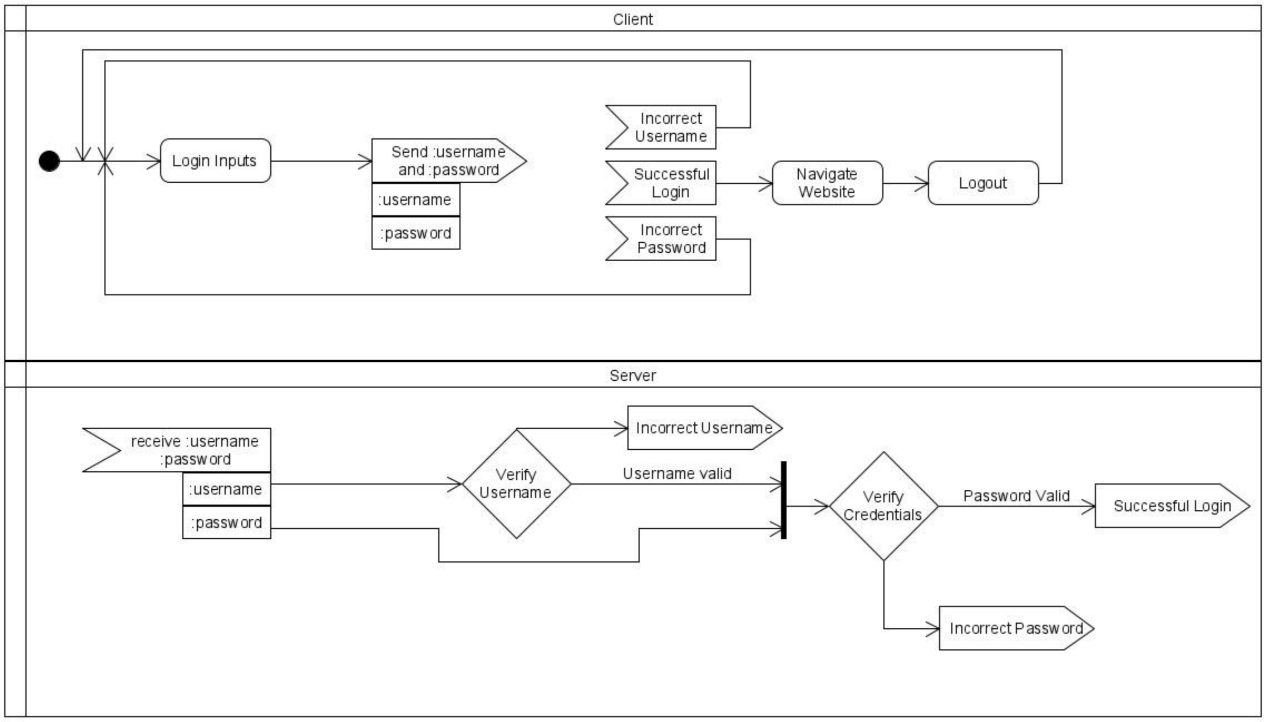
4.3 Use Case Diagram



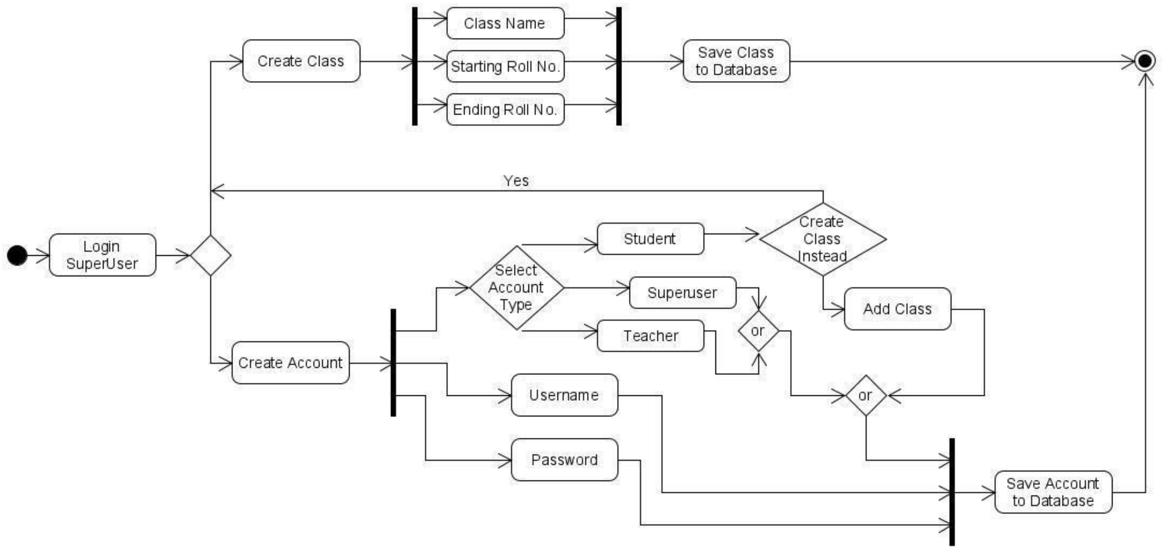
4.4 Sequence Diagram



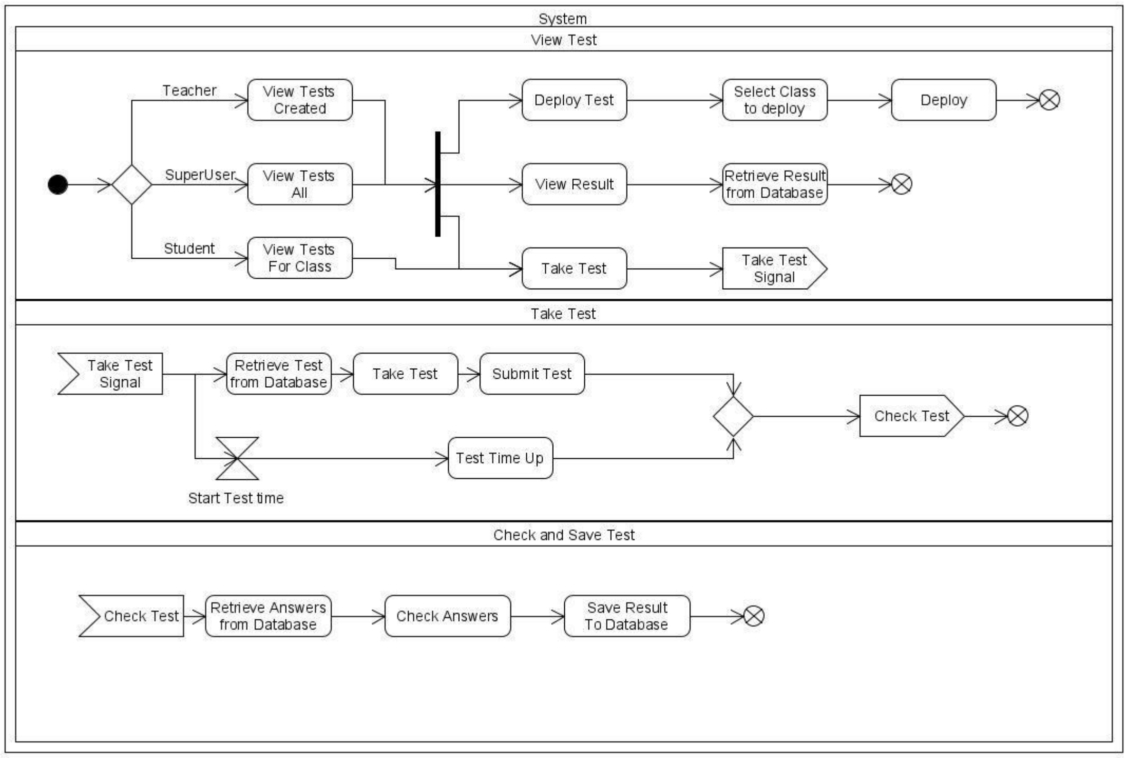
4.5 Activity Diagram



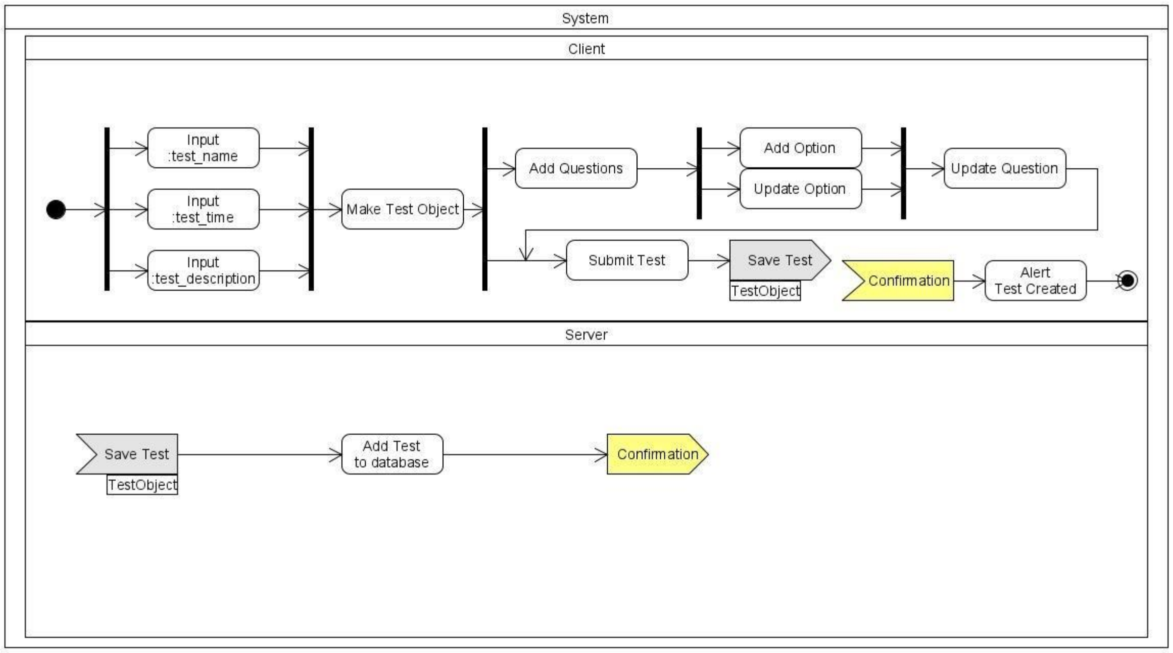
4.5.1 Login Activity



4.5.2 Create Accounts Activity

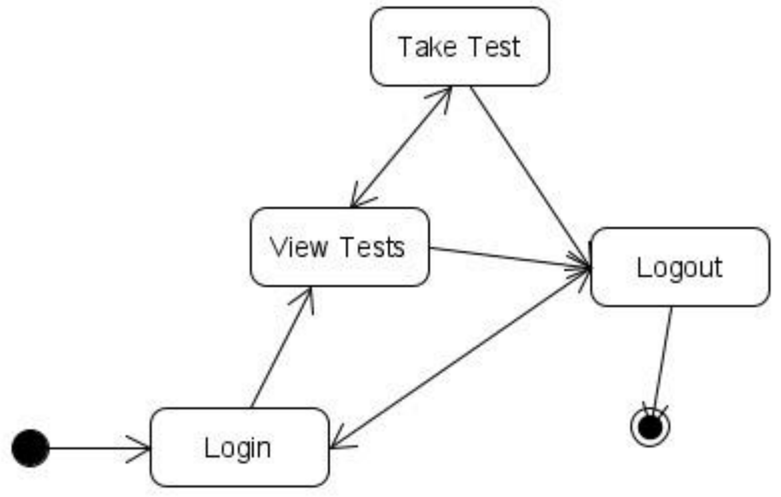


4.5.3 Test Activity

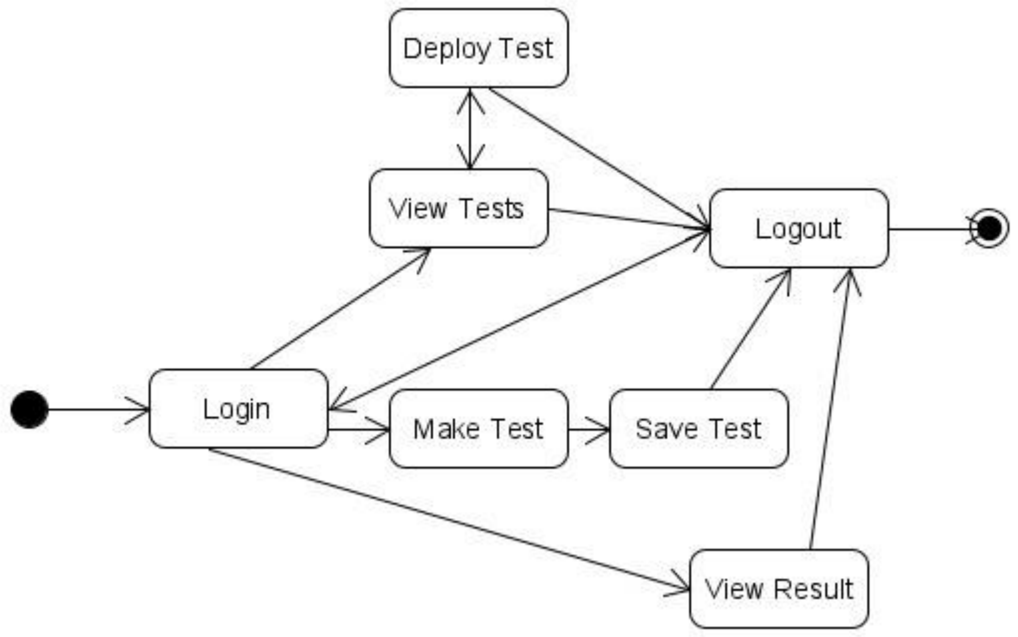


4.5.4 Teacher Test Activity

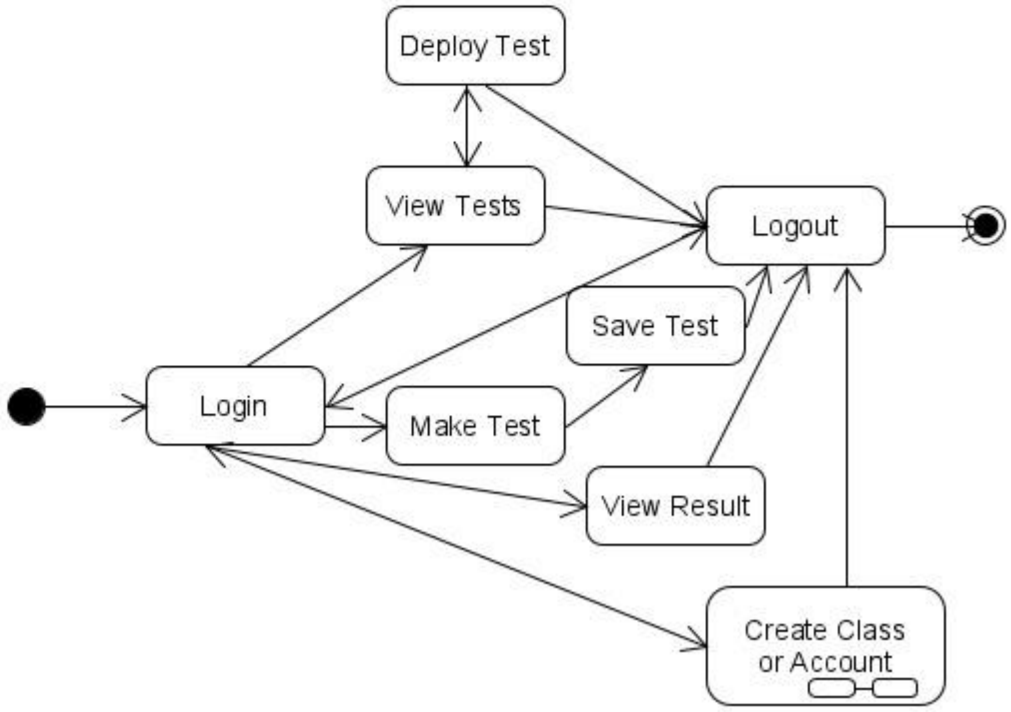
4.6 State Diagram



4.6.1 Student Side

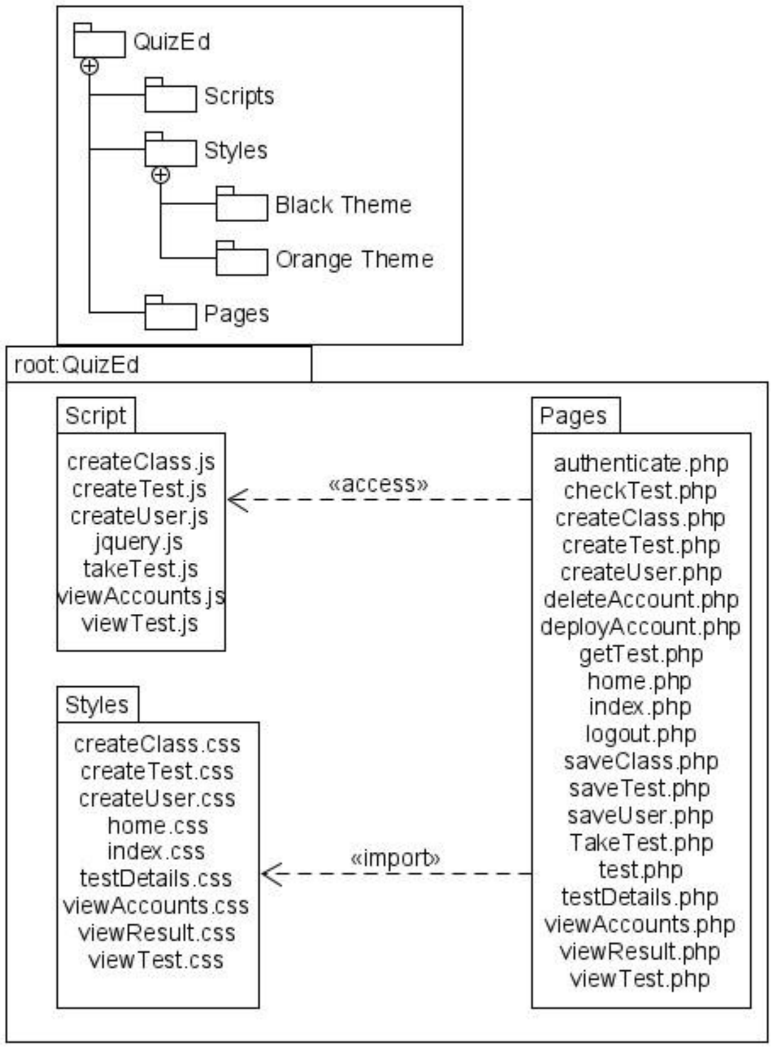


4.6.2 Teacher Side

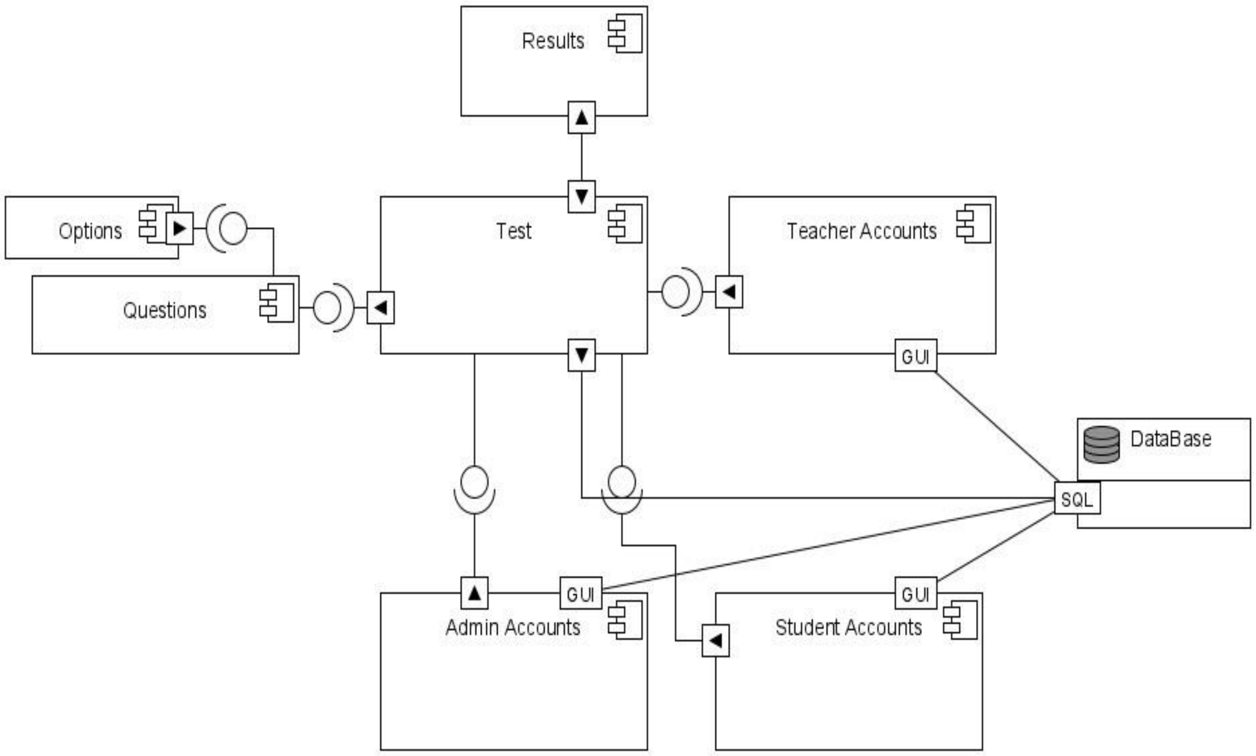


4.6.3 Admin Side

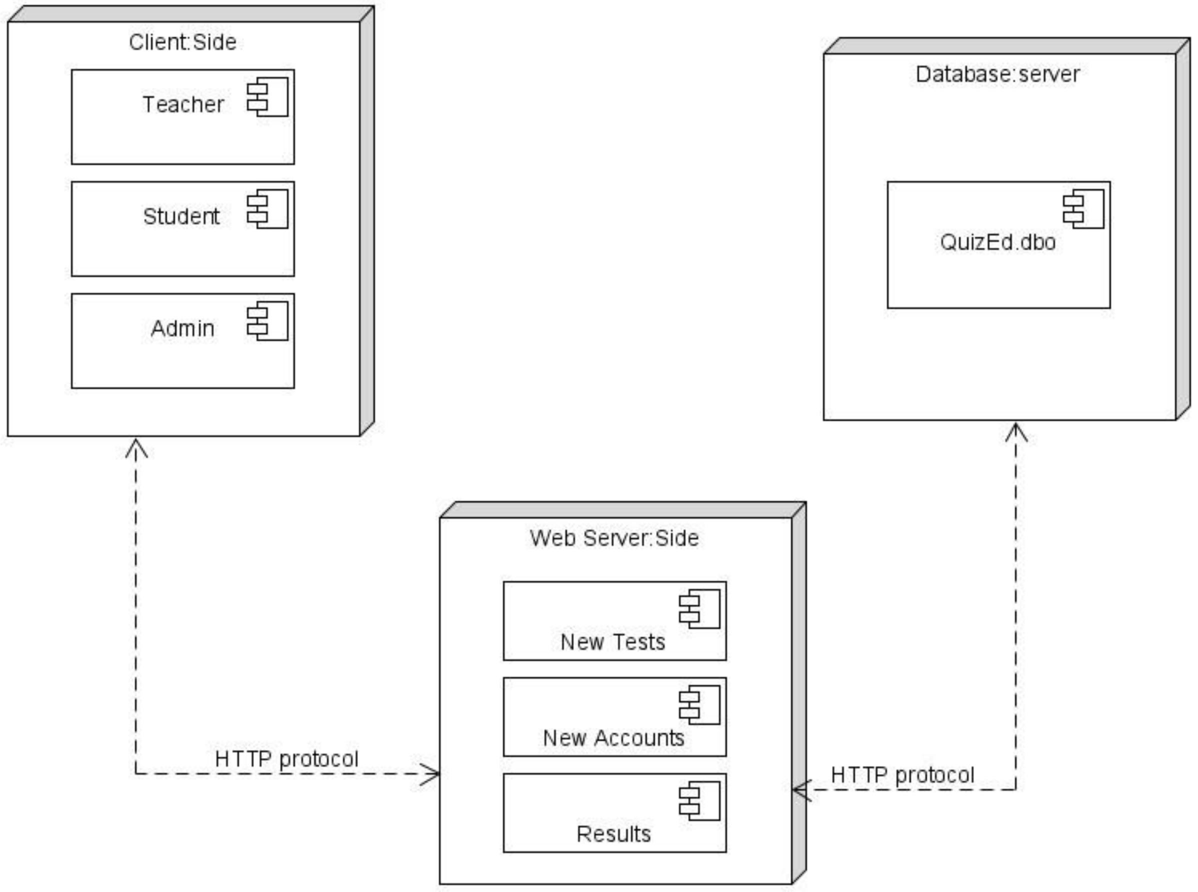
4.7 Package Diagram



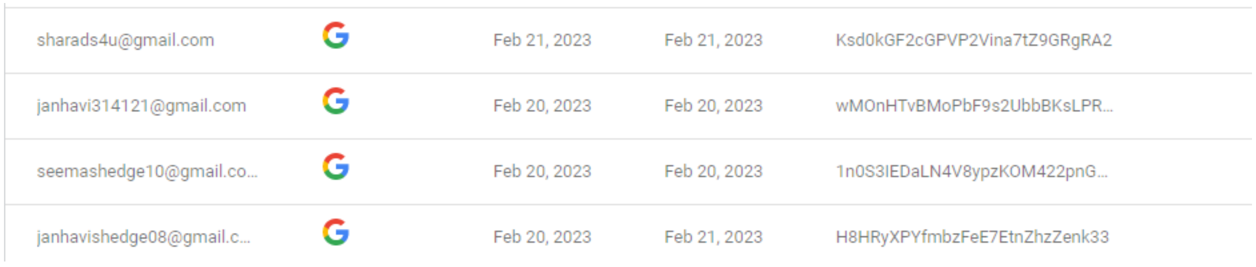
4.8 Component Diagram



4.9 Deployment Diagram

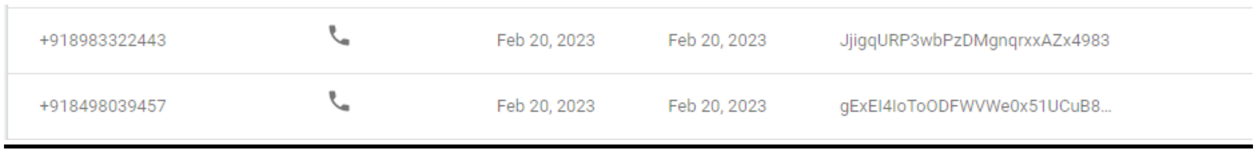


4.10 Database Diagram



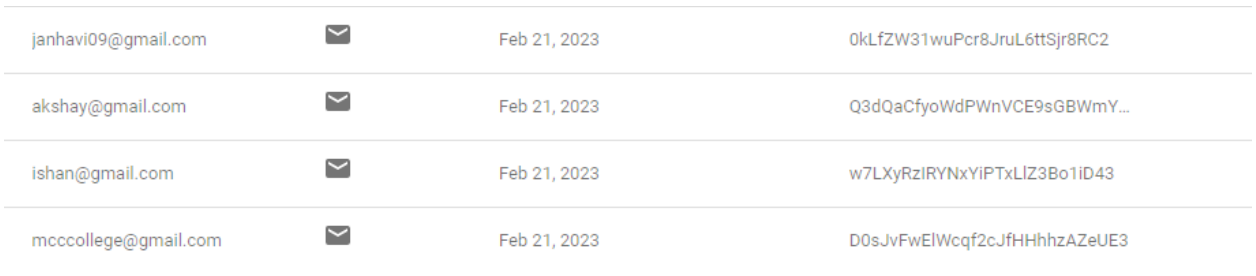
sharads4u@gmail.com	G	Feb 21, 2023	Feb 21, 2023	Ksd0kGF2cGPVP2Vina7tZ9GRgRA2
janhavi314121@gmail.com	G	Feb 20, 2023	Feb 20, 2023	wMOnHTvBMoPbF9s2UbbBKsLPR...
seemashedge10@gmail.co...	G	Feb 20, 2023	Feb 20, 2023	1n0S3lEDaLN4V8ypzKOM422pnG...
janhavishedge08@gmail.c...	G	Feb 20, 2023	Feb 21, 2023	H8HRyXPYfmbzFeE7EtnZhzZenk33

4.10.1 Google Login



+918983322443	📞	Feb 20, 2023	Feb 20, 2023	JjigqURP3wbPzDMgnqrxxAZx4983
+918498039457	📞	Feb 20, 2023	Feb 20, 2023	gExEI4loToODFWVWe0x51UCuB8...

4.10.2 Phone Login



janhavi09@gmail.com	✉	Feb 21, 2023	0kLfZW31wuPcr8JruL6ttSjr8RC2
akshay@gmail.com	✉	Feb 21, 2023	Q3dQaCfyoWdPWnVCE9sGBWmY...
ishan@gmail.com	✉	Feb 21, 2023	w7LXyRzIRYNxYiPTxLlZ3Bo1iD43
mcccollege@gmail.com	✉	Feb 21, 2023	D0sJvFwElWcqf2cJfHHhhzAZeUE3

4.10.3 Email Login

5 System Implementation

MainActivity.java

```
package com.app.quized;

import android.content.Intent;
import android.support.annotation.NonNull;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.support.v7.widget.GridLayoutManager;
import android.support.v7.widget.RecyclerView;
//import android.widget.Toolbar;
import android.support.v7.widget.Toolbar;
import android.view.LayoutInflater;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.CheckBox;
import android.widget.Toast;

import com.afollestad.materialdialogs.DialogAction;
import com.afollestad.materialdialogs.MaterialDialog;
import com.firebase.ui.auth.AuthUI;
import com.firebase.ui.auth.IdpResponse;
import com.github.javiersantos.materialstyleddialogs.MaterialStyledDialog;
import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.OnFailureListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.app.quized.Adapter.CategoryAdapter;
import com.app.quized.Common.Common;
import com.app.quized.Common.SpaceDecoration;
import com.app.quized.DBHelper.DBHelper;

import java.util.Arrays;
import java.util.List;

import io.paperdb.Paper;

public class MainActivity extends AppCompatActivity
{
    Toolbar toolbar;
    RecyclerView recycler_category;

    Button btn_sign_out;


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


        Paper.init(this);


        Common.isOnlineMode = Paper.book().read(Common.KEY_SAVE_ONLINE_MODE,
false);

        toolbar = findViewById(R.id.toolbar);
        toolbar.setTitle("Science Quiz");
        toolbar.setTextAlignment(View.TEXT_ALIGNMENT_CENTER);
        setSupportActionBar(toolbar);

        recycler_category = findViewById(R.id.recyler_category);
        recycler_category.setHasFixedSize(true);
        recycler_category.setLayoutManager(new GridLayoutManager(this, 1));


        CategoryAdapter adapter = new CategoryAdapter(MainActivity.this,
DBHelper.getInstance(this).getAllCategories());
        int spaceInPixel = 4;
        recycler_category.addItemDecoration(new SpaceDecoration(spaceInPixel));
        recycler_category.setAdapter(adapter);
""+e.getMessage(), Toast.LENGTH_SHORT).show();

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

    @Override
    public boolean onOptionsItemSelected(MenuItem item)
    {
        if(item.getItemId() == R.id.menu_settings)
        {
           showSettings();
        }
        if(item.getItemId() == R.id.btn_sign_out)
        {
           AuthUI.getInstance()
                   .signOut(MainActivity.this)
                   .addOnCompleteListener(new OnCompleteListener<Void>()
                   {
                      public void onComplete(@NonNull Task<Void> task)
                      {
//                          btn_sign_out.setEnabled(false);
                          Intent intent = new Intent(MainActivity.this,MainLoginActivity.class);
                          intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                          finish();
                          startActivity(intent);
                      }
                   }).addOnFailureListener(new OnFailureListener()
           {
              @Override
              public void onFailure(@NonNull Exception e)
              {
                 Toast.makeText(MainActivity.this, ""+e.getMessage(),
Toast.LENGTH_SHORT).show();
              }
           });
        }

        return true;
    }

    private void showSettings()
    {
        View setting_layout = LayoutInflater.from(this).inflate(R.layout.settings_layout, null);
        final CheckBox ckb_online_mode = setting_layout.findViewById(R.id.ckb_online_mode);

        //Load data from paper, if not available just init default false
        ckb_online_mode.setChecked(Paper.book().read(Common.KEY_SAVE_ONLINE_MODE,
false));

        //Show Dialog
        new MaterialStyledDialog.Builder(MainActivity.this)
              .setIcon(R.drawable.ic_settings_white_24dp)
              .setTitle("Settings")
              .setDescription("Please choose action")
```

```
                .setCustomView(setting_layout)
                .setNegativeText("DISMISS")
                .onNegative(new MaterialDialog.SingleButtonCallback()
                {

                    @Override
                    public void onClick(@NonNull MaterialDialog dialog, @NonNull DialogAction
which)
                    {
                        if(ckb_online_mode.isChecked())
                        {
                            Common.isOnlineMode = true;
                        }
                        else
                        {
                            Common.isOnlineMode = false;
                        }

                        //Save
                        Paper.book().write(Common.KEY_SAVE_ONLINE_MODE,
ckb_online_mode.isChecked());
                    }
                }).show();
    }
}
```

MainLoginActivity.java

```
package com.app.quized;

import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.Button;
import android.widget.Toast;

import com.firebase.ui.auth.AuthUI;
import com.firebase.ui.auth.IdpResponse;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;

import java.util.Arrays;
import java.util.List;

public class MainLoginActivity extends AppCompatActivity
{

   private static final int MY_REQUEST_CODE = 7117;
   List<AuthUI.IdpConfig> providers;
   Button btn_sign_out;

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


      providers = Arrays.asList(
            new AuthUI.IdpConfig.EmailBuilder().build(),
            new AuthUI.IdpConfig.PhoneBuilder().build(),
            new AuthUI.IdpConfig.GoogleBuilder().build());




      showSignInOptions();

   }

   private void showSignInOptions()
   {
        startActivityForResult(
                AuthUI.getInstance()
                        .createSignInIntentBuilder()
                        .setAvailableProviders(providers)
                        .setTheme(R.style.Mytheme)
                        .build(),MY_REQUEST_CODE
        );
    }

    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        super.onActivityResult(requestCode, resultCode, data);

        if (requestCode == MY_REQUEST_CODE)
        {
            IdpResponse response = IdpResponse.fromResultIntent(data);

            if (resultCode == RESULT_OK) {

                FirebaseUser user = FirebaseAuth.getInstance().getCurrentUser();
                startActivity(new Intent(MainLoginActivity.this,MainActivity.class)
                        .addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP));
                finish();

            } else {

                Toast.makeText(this, ""+response.getError().getMessage(),
Toast.LENGTH_SHORT).show();
            }
        }
    }
}
```

QuestionActivity.java

```
package com.app.quized;

import android.app.Activity;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import android.os.CountDownTimer;
import android.support.annotation.NonNull;
import android.support.annotation.Nullable;
import android.support.constraint.ConstraintLayout;
import android.support.design.widget.TabLayout;
import android.support.v4.view.ViewPager;
import android.support.v7.widget.GridLayoutManager;
import android.support.v7.widget.RecyclerView;
import android.text.TextUtils;
import android.view.Gravity;
import android.view.MotionEvent;
import android.view.View;
import android.support.v4.view.GravityCompat;
import android.support.v7.app.ActionBarDrawerToggle;
import android.view.MenuItem;
import android.support.design.widget.NavigationView;
import android.support.v4.widget.DrawerLayout;

import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.view.Menu;
import android.widget.Button;
import android.widget.LinearLayout;
import android.widget.TextView;
import android.widget.Toast;

import com.afollestad.materialdialogs.DialogAction;
import com.afollestad.materialdialogs.MaterialDialog;
import com.firebase.ui.auth.AuthUI;
import com.github.javiersantos.materialstyleddialogs.MaterialStyledDialog;
import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.OnFailureListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.database.FirebaseDatabase;
import com.google.gson.Gson;
import com.app.quized.Adapter.AnswerSheetAdapter;
import com.app.quized.Adapter.AnswerSheetHelperAdapter;
import com.app.quized.Adapter.QuestionFragmentAdapter;
import com.app.quized.Common.Common;
import com.app.quized.Common.SpaceDecoration;
import com.app.quized.DBHelper.DBHelper;
import com.app.quized.DBHelper.OnlineDBHelper;
import com.app.quized.Interface.MyCallback;
import com.app.quized.Misc.CustomViewPager;
import com.app.quized.model.CurrentQuestion;
import com.app.quized.model.Question;

import java.util.List;
import java.util.concurrent.TimeUnit;

public class QuestionActivity extends AppCompatActivity implements
NavigationView.OnNavigationItemSelectedListener
{
    private static final int CODE_GET_RESULT = 9999;
    int time_play = Common.TOTAL_TIME;
    boolean isAnswerModeView = false;

    TextView txt_right_answer, txt_timer, txt_wrong_answer;

    RecyclerView answer_sheet_view, answer_sheet_helper;
    AnswerSheetAdapter answerSheetAdapter;
    AnswerSheetHelperAdapter answerSheetHelperAdapter;


    CustomViewPager viewPager;
    TabLayout tabLayout;
    DrawerLayout drawer;

    //personal modifications
    Button btn_sign_out;
    private boolean isDoItAgainMode = false;

    @Override
    protected void onDestroy()
    {
        if(Common.countDownTimer != null)
            Common.countDownTimer.cancel();
        super.onDestroy();
    }

    BroadcastReceiver gotoQuestionNum = new BroadcastReceiver()
    {
        @Override
        public void onReceive(Context context, Intent intent)
        {
            if(intent.getAction().toString().equals(Common.KEY_GO_TO_QUESTION))
            {
                int question = intent.getIntExtra(Common.KEY_GO_TO_QUESTION, -1);
                if(question != -1)
                    viewPager.setCurrentItem(question); //Got to question
                drawer.closeDrawer(Gravity.START); //Close menu
            }
        }
    };

    Button btn_done;

    MenuItem menu_sign_out;
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_question);
        Toolbar toolbar = findViewById(R.id.toolbar);
        toolbar.setTitle(Common.selectedCategory.getName());
        setSupportActionBar(toolbar);

        isDoItAgainMode = false;
        drawer = findViewById(R.id.drawer_layout);
        ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(
                this, drawer, toolbar, R.string.navigation_drawer_open,
R.string.navigation_drawer_close);
        drawer.addDrawerListener(toggle);
        toggle.syncState();
        NavigationView navigationView = findViewById(R.id.nav_view);
        navigationView.setNavigationItemSelectedListener(this);

        View hView = navigationView.getHeaderView(0);
        answer_sheet_helper = hView.findViewById(R.id.answer_sheet);
        answer_sheet_helper.setHasFixedSize(true);
        answer_sheet_helper.setLayoutManager(new GridLayoutManager(this, 3));
        answer_sheet_helper.addItemDecoration(new SpaceDecoration(2));
        answerSheetHelperAdapter = new AnswerSheetHelperAdapter(this,
Common.answerSheetList);
        answer_sheet_helper.setAdapter(answerSheetHelperAdapter);


        btn_done = hView.findViewById(R.id.btn_done);
        btn_done.setOnClickListener(new View.OnClickListener()
        {
            @Override
            public void onClick(View v)
            {
                //Here we will copy code on menu_finish_game
                if(!isAnswerModeView)
                {



                    new MaterialStyledDialog.Builder(QuestionActivity.this)
                            .setTitle("Finish?")
                            .setIcon(R.drawable.ic_mood_black_24dp)
                            .setDescription("Do you really want to finish?")
                            .setNegativeText("No")
                            .onNegative(new MaterialDialog.SingleButtonCallback()
                            {
                                @Override
                                public void onClick(@NonNull MaterialDialog dialog, @NonNull
DialogAction which)
                                {
                                    dialog.dismiss();
                                }
                            })
                            .setPositiveText("Yes")
                            .onPositive(new MaterialDialog.SingleButtonCallback()
                            {
                                @Override
                                public void onClick(@NonNull MaterialDialog dialog, @NonNull
DialogAction which)
                                {
                                    dialog.dismiss();
                                    finishGame();
                                    drawer.closeDrawer(Gravity.START);
                                }
                            }).show();
                }
                else
                    finishGame();
            }
        });



        Common.fragmentsList.clear();
        Common.questionList.clear();
        Common.right_answer_count = 0;
        Common.wrong_answer_count = 0;


        takeQuestion();

    }


    private void finishGame()
    {
        //For the last question-answer page
        int position = viewPager.getCurrentItem();
        QuestionFragment questionFragment = Common.fragmentsList.get(position);
        CurrentQuestion question_state = questionFragment.getSelectedAnswer();
        if(!isAnswerModeView)
        {

            Common.answerSheetList.set(position, question_state); // Set question answer for answer
sheet
            answerSheetAdapter.notifyDataSetChanged(); //Change color in answer sheet
            answerSheetHelperAdapter.notifyDataSetChanged();
            Common.countDownTimer.cancel();

            countCorrectAnswer();


            txt_right_answer.setText(new StringBuilder(String.format("%d",
Common.right_answer_count))
                    .append("/")
                    .append(String.format("%d", Common.questionList.size())).toString());
            txt_wrong_answer.setText(String.valueOf(Common.wrong_answer_count));
        }




        if(question_state!=null)
        {
            if (question_state.getType() == Common.ANSWER_TYPE.NO_ANSWER)
            {
                questionFragment.showCorrectAnswer();
                questionFragment.disableAnswer();
            }
        }

        //We will navigate to result activity here
        Intent intent = new Intent(QuestionActivity.this, ResultActivity.class);
        Common.timer = Common.TOTAL_TIME - time_play;
        Common.no_answer_count = Common.questionList.size() -
(Common.wrong_answer_count + Common.right_answer_count);
        Common.data_question = new StringBuilder(new
Gson().toJson(Common.answerSheetList));

        startActivityForResult(intent, CODE_GET_RESULT);
    }

    private void countCorrectAnswer()
    {
        //Reset variable
        Common.right_answer_count = Common.wrong_answer_count = 0;
        for(CurrentQuestion item:Common.answerSheetList)
        {
            if(item!=null)
            {
                if (item.getType() == Common.ANSWER_TYPE.RIGHT_ANSWER)
                    Common.right_answer_count++;
                else if (item.getType() == Common.ANSWER_TYPE.WRONG_ANSWER)
                    Common.wrong_answer_count++;
            }
    }

    private void genFragmentList()
    {
        for(int i=0; i<Common.questionList.size(); i++)
        {
            Bundle bundle = new Bundle();
            bundle.putInt("index", i);
            QuestionFragment fragment = new QuestionFragment();
            fragment.setArguments(bundle);

            Common.fragmentsList.add(fragment);

        }
    }


    private void countTimer()
    {
        if(Common.countDownTimer == null)
        {
            Common.countDownTimer = new CountDownTimer(Common.TOTAL_TIME, 1000)
            {
```

```
            @Override
            public void onTick(long millisUntilFinished)
            {
                txt_timer.setText(String.format("%02d:%02d",
                        TimeUnit.MILLISECONDS.toMinutes(millisUntilFinished),
                        TimeUnit.MILLISECONDS.toSeconds(millisUntilFinished) -

TimeUnit.MINUTES.toSeconds(TimeUnit.MILLISECONDS.toMinutes(millisUntilFinished))));
                time_play-=1000;

            }



            @Override
            public void onFinish()
            {
                //Finish Game
                Toast.makeText(QuestionActivity.this, "Time finished!",
Toast.LENGTH_SHORT).show();
                finishGame();
            }
        }.start();
    }
    else
    {
        Common.countDownTimer.cancel();
        Common.countDownTimer = new CountDownTimer(Common.TOTAL_TIME, 1000)
        {
            @Override
            public void onTick(long millisUntilFinished)
            {
                txt_timer.setText(String.format("%02d:%02d",
                        TimeUnit.MILLISECONDS.toMinutes(millisUntilFinished),
                        TimeUnit.MILLISECONDS.toSeconds(millisUntilFinished) -

TimeUnit.MINUTES.toSeconds(TimeUnit.MILLISECONDS.toMinutes(millisUntilFinished))));
                time_play-=1000;

            }

            @Override
            public void onFinish()
            {
                //Finish Game
                Toast.makeText(QuestionActivity.this, "Time finished!",
Toast.LENGTH_SHORT).show();
                finishGame();
            }
        }.start();
    }
}

private void takeQuestion()
{
    Common.questionList.clear();
    Common.selected_values.clear();
    if(!Common.isOnlineMode)
    {
        Common.questionList =
DBHelper.getInstance(this).getQuestionByCategory(Common.selectedCategory.getId(),
Common.selectedCategory.getClassId());
        if(Common.questionList.size() == 0)
        {
            //if no question
            new MaterialStyledDialog.Builder(this)
                    .setTitle("Oops !")
                    .setIcon(R.drawable.ic_sentiment_very_dissatisfied_black_24dp)
                    .setDescription("We don't have any question in this
"+Common.selectedCategory.getName()+" category ")
                    .setPositiveText("OK")
                    .onPositive(new MaterialDialog.SingleButtonCallback()
                    {
                        @Override
                        public void onClick(@NonNull MaterialDialog dialog, @NonNull
DialogAction which)
                        {
                            dialog.dismiss();
                            finish();
                        }
                    }).show();
        }
        else
        {

            if(Common.answerSheetList.size() > 0)
                Common.answerSheetList.clear();
            //Gen answerSheet item from question
            //30 question = 30 answer sheet item
            //1 question = 1 answer sheet item
            for(int i = 0; i<Common.questionList.size(); i++)
            {
                Common.answerSheetList.add(new CurrentQuestion(i,
Common.ANSWER_TYPE.NO_ANSWER)); //Default all answer is no answer
            }
            Common.TOTAL_TIME = 15000*Common.questionList.size();

        }

        setUpQuestion();
    }
    else
    {
        OnlineDBHelper.getInstance(this, FirebaseDatabase.getInstance())
                .readData(new MyCallback()
                {
                    @Override
                    public void setQuestionList(List<Question> questionList)
                    {
                        Common.questionList.clear();
                        Common.questionList = questionList;

                        if(Common.questionList.size() == 0)
                        {
                            //if no question
                            new MaterialStyledDialog.Builder(QuestionActivity.this)
                                    .setTitle("Oops !")
                                    .setIcon(R.drawable.ic_sentiment_very_dissatisfied_black_24dp)
                                    .setDescription("We don't have any question in this
"+Common.selectedCategory.getName()+" category ")
                                    .setPositiveText("OK")
                                    .onPositive(new MaterialDialog.SingleButtonCallback()
                                    {
                                        @Override
                                        public void onClick(@NonNull MaterialDialog dialog, @NonNull
DialogAction which)
                                        {
                                            dialog.dismiss();
                                            finish();
                                        }
                                    }).show();
                        }
                        else
                        {

                            if(Common.answerSheetList.size() > 0)
                                Common.answerSheetList.clear();
                            //Gen answerSheet item from question
                            //30 question = 30 answer sheet item
                            //1 question = 1 answer sheet item
                            for(int i = 0; i<Common.questionList.size(); i++)
                            {
                                Common.answerSheetList.add(new CurrentQuestion(i,
Common.ANSWER_TYPE.NO_ANSWER)); //Default all answer is no answer
                            }

                        }

                        setUpQuestion();

                    }
                }, Common.selectedCategory.getName().replace(" ", "").replace("/", "_"),
String.valueOf(Common.selectedCategory.getClassId()));
    }

}


private void setUpQuestion()
{
    if(Common.questionList.size() > 0)
    {
        //Show TextView right answer and TextView Timer
        txt_right_answer = findViewById(R.id.txt_answer_right);
        txt_timer = findViewById(R.id.txt_timer);

        txt_timer.setVisibility(View.VISIBLE);
        txt_right_answer.setVisibility(View.VISIBLE);

        txt_right_answer.setText(new StringBuilder(String.format("%d/%d",
Common.right_answer_count, Common.questionList.size())));

        countTimer();



        //View
        answer_sheet_view = findViewById(R.id.grid_answer);
        answer_sheet_view.setHasFixedSize(true);
        if(Common.questionList.size() > 5) // If question List have size > 5, we will separate 2
rows
            answer_sheet_view.setLayoutManager(new GridLayoutManager(this,
Common.questionList.size()/2));
        answerSheetAdapter = new AnswerSheetAdapter(this, Common.answerSheetList);
        answer_sheet_view.setAdapter(answerSheetAdapter);

        viewPager = findViewById(R.id.viewpager);
        tabLayout = findViewById(R.id.sliding_tabs);

        genFragmentList();


        QuestionFragmentAdapter questionFragmentAdapter = new
QuestionFragmentAdapter(getSupportFragmentManager(),
                this,
                Common.fragmentsList);
        viewPager.setAdapter(questionFragmentAdapter);

        tabLayout.setupWithViewPager(viewPager);



        //Event
        viewPager.addOnPageChangeListener(new ViewPager.OnPageChangeListener()
        {
            int SCROLLING_RIGHT = 0;
            int SCROLLING_LEFT = 1;
            int SCROLLING_UNDETERMINED = 2;

            int currentScrollDirection = 2;

            private void setScrollingDirection(float positionOffset)
            {
                if((1-positionOffset) >= 0.5)
                    this.currentScrollDirection=SCROLLING_RIGHT;
                else if((1-positionOffset) <= 0.5)
                    this.currentScrollDirection=SCROLLING_LEFT;
            }

            private boolean isScrollDirectionUndetermined()
            {
                return (currentScrollDirection == SCROLLING_UNDETERMINED);
            }

            private boolean isScrollingRight()
            {
                return (currentScrollDirection == SCROLLING_RIGHT);
            }

            private boolean isScrollingLeft()
```

```
{
   return (currentScrollDirection == SCROLLING_LEFT);
}

@Override
public void onPageScrolled(int i, float v, int i1)
{
   if(isScrollDirectionUndetermined())
      setScrollingDirection(v);



}
QuestionFragment questionFragment;
@Override
public void onPageSelected(int i)
{

   int position = 0;
   if(i>0)
   {
      if(isScrollingRight())
         questionFragment = Common.fragmentsList.get(i-1);
         position = i-1;

      }
      else if(isScrollingLeft())
      {
                              questionFragment = Common.fragmentsList.get(i+1);
         position = i+1;
      }
      else
      {
         questionFragment = Common.fragmentsList.get(position);
      }
   }
   else
   {
      questionFragment = Common.fragmentsList.get(0);
      position = 0;

   }


   CurrentQuestion question_state = questionFragment.getSelectedAnswer();
   if(!isAnswerModeView)
```

```
          {

             Common.answerSheetList.set(position, question_state);
answerSheetAdapter.notifyDataSetChanged();
answerSheetHelperAdapter.notifyDataSetChanged();
             countCorrectAnswer();
          }


          if(isAnswerModeView)
          {
             questionFragment.showCorrectAnswer();
             questionFragment.disableAnswer();

          }
          if(isDoItAgainMode)
          {
             questionFragment.resetQuestion();
          }

          txt_right_answer.+




setText(new StringBuilder(String.format("%d", Common.right_answer_count))
                .append("/")
                .append(String.format("%d", Common.questionList.size())).toString());
          txt_wrong_answer.setText(String.valueOf(Common.wrong_answer_count));





       }

       @Override
       public void onPageScrollStateChanged(int i)
       {
          if(i == ViewPager.SCROLL_STATE_IDLE)
             this.currentScrollDirection = SCROLLING_UNDETERMINED;


          if(!isAnswerModeView && viewPager!=null)
          {
             if(viewPager.getCurrentItem() < (Common.answerSheetList.size()-1))
```

```
                    {
                        btn_finish.setVisible(false);
                        btn_finish.setEnabled(false);
                    }
                    else if(viewPager.getCurrentItem() == (Common.answerSheetList.size()-1))
                    {
                        btn_finish.setVisible(true);
                        btn_finish.setEnabled(true);
                    }
                }
                else
                {
                    btn_finish.setVisible(true);
                    btn_finish.setEnabled(true);
                }


            }
        });

    }
}

@Override
public void onBackPressed()
{
    DrawerLayout drawer = findViewById(R.id.drawer_layout);
    if (drawer.isDrawerOpen(GravityCompat.START))
    {
        drawer.closeDrawer(GravityCompat.START);
    } else
    {
        super.onBackPressed();
    }
}
MenuItem btn_finish;
@Override
public boolean onPrepareOptionsMenu(Menu menu)
{
    MenuItem item = menu.findItem(R.id.menu_wrong_answer);
    ConstraintLayout constraintLayout = (ConstraintLayout)item.getActionView();
    txt_wrong_answer = constraintLayout.findViewById(R.id.txt_wrong_answer);
    txt_wrong_answer.setText(String.valueOf(Common.wrong_answer_count));


    btn_finish = menu.findItem(R.id.menu_finish_game);
```

```
menu_sign_out = menu.findItem(R.id.btn_sign_out);

if(!isAnswerModeView && viewPager!=null)
{
   if(viewPager.getCurrentItem() < (Common.answerSheetList.size()-1))
   {
      btn_finish.setVisible(false);
      btn_finish.setEnabled(false);
   }
   else if(viewPager.getCurrentItem() == (Common.answerSheetList.size()-1))
   {
      btn_finish.setVisible(true);
      btn_finish.setEnabled(true);
   }
}
else
{
   btn_finish.setVisible(true);
   btn_finish.setEnabled(true);
}

LinearLayout tabStrip = ((LinearLayout)tabLayout.getChildAt(0));
for(int i = 0; i < tabStrip.getChildCount(); i++) {
   tabStrip.getChildAt(i).setOnTouchListener(new View.OnTouchListener() {
      @Override
      public boolean onTouch(View v, MotionEvent event) {
         return true;
      }
   });
}
if(viewPager!=null)
   viewPager.setAllowedSwipeDirection(Common.SwipeDirection.right);

new MaterialStyledDialog.Builder(this)
      .setTitle("Warning")
      .setIcon(R.drawable.ic_error_outline_black_24dp)
      .setDescription("You can only swipe right.\nOnce you swipe right, you won't be able
to go back left.\n\nHappy Quizzing!")
      .setPositiveText("OK")
      .onPositive(new MaterialDialog.SingleButtonCallback()
      {
         @Override
         public void onClick(@NonNull MaterialDialog dialog, @NonNull DialogAction
which)
         {
            dialog.dismiss();
```

```
                }
            }).show();

        return true;
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu)
    {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.question, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item)
    {
        .
        int id = item.getItemId();



        if(id == R.id.btn_sign_out)
        {
            AuthUI.getInstance()
                    .signOut(QuestionActivity.this)
                    .addOnCompleteListener(new OnCompleteListener<Void>()
                    {
                        public void onComplete(@NonNull Task<Void> task)
                        {
//                            btn_sign_out.setEnabled(false);
                            startActivity(new Intent(QuestionActivity.this, MainActivity.class));
                        }
                    }).addOnFailureListener(new OnFailureListener()
            {
                @Override
                public void onFailure(@NonNull Exception e)
                {
                    Toast.makeText(QuestionActivity.this, ""+e.getMessage(),
Toast.LENGTH_SHORT).show();
                }
            });
        }
        if (id == R.id.menu_finish_game)
        {
            if(!isAnswerModeView)
            {
                new MaterialStyledDialog.Builder(this)
                        .setTitle("Finish?")
                        .setIcon(R.drawable.ic_mood_black_24dp)
                        .setDescription("Do you really want to finish?")
                        .setNegativeText("No")
                        .onNegative(new MaterialDialog.SingleButtonCallback()
                        {
                            @Override
                            public void onClick(@NonNull MaterialDialog dialog, @NonNull
DialogAction which)
                            {
                                dialog.dismiss();
                            }
                        })
                        .setPositiveText("Yes")
                        .onPositive(new MaterialDialog.SingleButtonCallback()
                        {
                            @Override
                            public void onClick(@NonNull MaterialDialog dialog, @NonNull
DialogAction which)
                            {
                                dialog.dismiss();
                                finishGame();
                            }
                        }).show();
            }
            else
                finishGame();
            return true;
        }

        return super.onOptionsItemSelected(item);
    }



    @SuppressWarnings("StatementWithEmptyBody")
    @Override
    public boolean onNavigationItemSelected(MenuItem item)
    {
        // Handle navigation view item clicks here.
        int id = item.getItemId();
        if (id == R.id.nav_home)
        {
            // Handle the camera action
        } else if (id == R.id.nav_gallery)
        {

        } else if (id == R.id.nav_slideshow)
        {

        } else if (id == R.id.nav_tools)
        {

        } else if (id == R.id.nav_share)
        {

        } else if (id == R.id.nav_send)
        {


        }

        DrawerLayout drawer = findViewById(R.id.drawer_layout);
        drawer.closeDrawer(GravityCompat.START);
        return true;
    }

    @Override
    protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data)
    {
        super.onActivityResult(requestCode, resultCode, data);

        if(requestCode == CODE_GET_RESULT)
        {
            if(resultCode == Activity.RESULT_OK)
            {
                String action = data.getStringExtra("action");
//                Toast.makeText(QuestionActivity.this, "action = "+action,
Toast.LENGTH_SHORT).show();
                if(action == null || TextUtils.isEmpty(action))
                {


                    int questionNum = data.getIntExtra(Common.KEY_BACK_FROM_RESULT, -1);
                    viewPager.setCurrentItem(questionNum);

                    isAnswerModeView = true;
                    Common.countDownTimer.cancel();
```

```
          //
          int position = viewPager.getCurrentItem();
          QuestionFragment questionFragment = Common.fragmentsList.get(position);
          CurrentQuestion question_state = questionFragment.getSelectedAnswer();


          if(question_state!=null)
          {
             if (question_state.getType() == Common.ANSWER_TYPE.NO_ANSWER ||
question_state.getType() == Common.ANSWER_TYPE.WRONG_ANSWER)
             {
                questionFragment.showCorrectAnswer();
                questionFragment.disableAnswer();
             }
          }
                     LinearLayout tabStrip = ((LinearLayout)tabLayout.getChildAt(0));
          for(int i = 0; i < tabStrip.getChildCount(); i++) {
             tabStrip.getChildAt(i).setOnTouchListener(new View.OnTouchListener() {
                @Override
                public boolean onTouch(View v, MotionEvent event) {
                   return true;
                }
             });
          }
          viewPager.setAllowedSwipeDirection(Common.SwipeDirection.none);
          //


          txt_wrong_answer.setVisibility(View.GONE);
          txt_right_answer.setVisibility(View.GONE);
          txt_timer.setVisibility(View.GONE);

       }
       else
       {

          if(action.equals("viewquizanswer"))
          {

             viewPager.setAllowedSwipeDirection(Common.SwipeDirection.all);
             viewPager.setCurrentItem(0);

             isAnswerModeView = true;
             Common.countDownTimer.cancel();
```

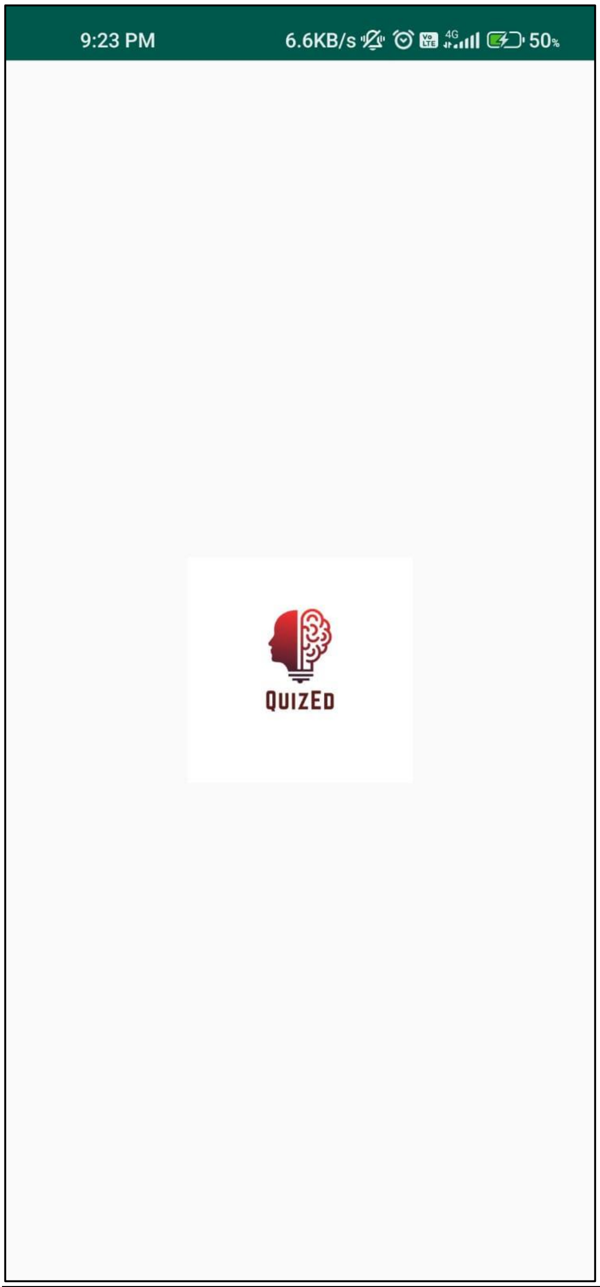
```
                        txt_wrong_answer.setVisibility(View.GONE);
                        txt_right_answer.setVisibility(View.GONE);
                        txt_timer.setVisibility(View.GONE);


                    }
                    else if(action.equals("doitagain"))
                    {

                        isDoItAgainMode = true;

                        viewPager.setCurrentItem(0);
                        viewPager.setAllowedSwipeDirection(Common.SwipeDirection.right);

                        isAnswerModeView = false;
                        Common.right_answer_count = 0;
                        Common.wrong_answer_count = 0;
                        Common.selected_values.clear();
                        countTimer();

                        txt_wrong_answer.setVisibility(View.VISIBLE);
                        txt_right_answer.setVisibility(View.VISIBLE);
                        txt_timer.setVisibility(View.VISIBLE);

                        for(CurrentQuestion item:Common.answerSheetList)
                            item.setType(Common.ANSWER_TYPE.NO_ANSWER);
answerSheetAdapter.notifyDataSetChanged();
                        answerSheetHelperAdapter.notifyDataSetChanged();

                    }
                }
            }
        }
    }
}
```

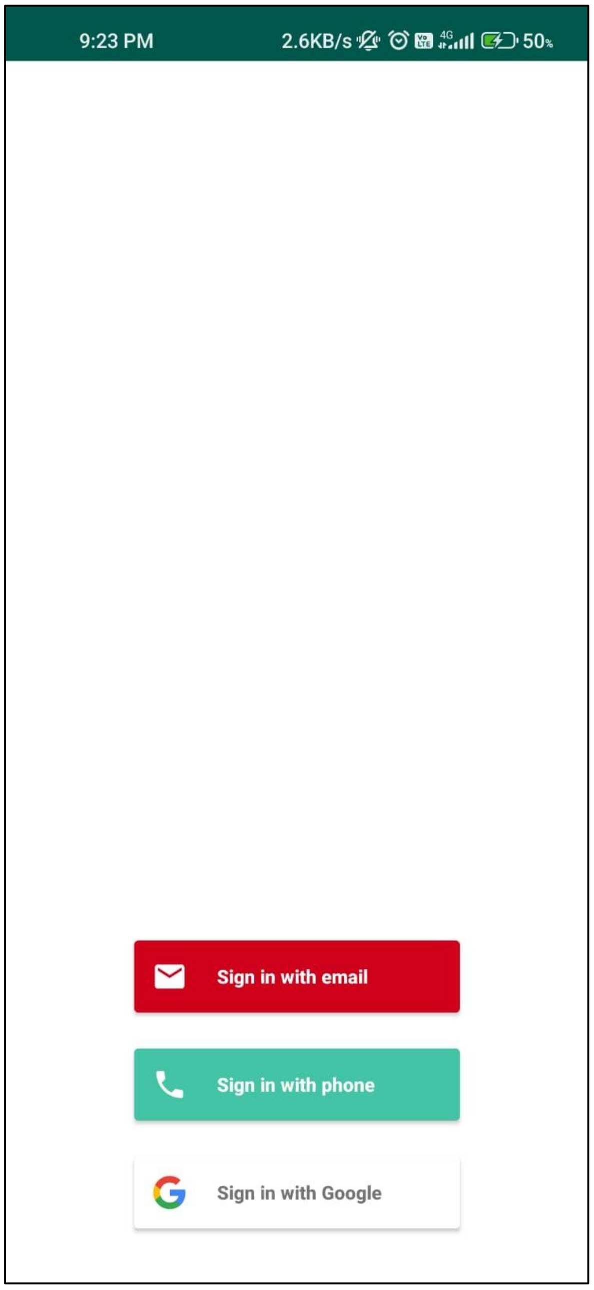
5. Result

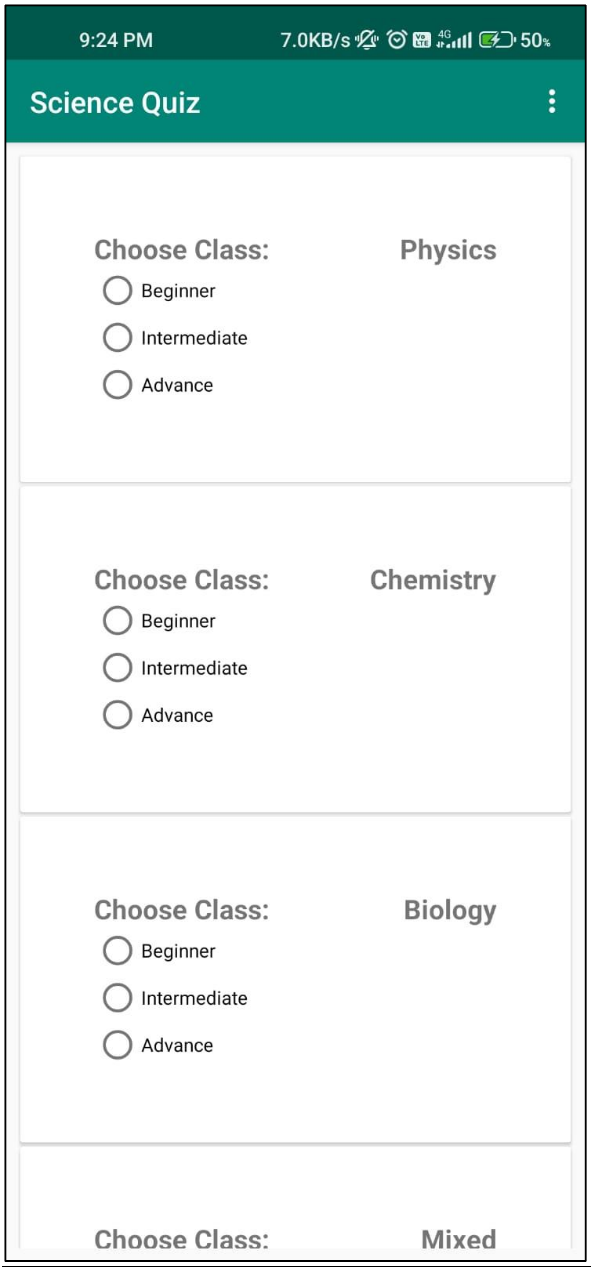
5.1 Validation and Naming Conventions

Sr.No	Control ID	Validation Used	Reason
1	username	RequiredFieldValidator	Username cannot be null
2	pswd	RequiredFieldValidator	Password cannot be null
3	testname	RequiredFieldValidator	Testname cannot be null
4	testtime	RequiredFieldValidator, NumericValidator	Time cannot be null and should be a number.
5	testdetails	RequiredFieldValidator	Details cannot be null
6	question	RequiredFieldValidator	Question Name cannot be null
7	option	RequiredFieldValidator	Option Field cannot be null
8	classname	RequiredFieldValidator	Class Name cannot be null
9	startnum	RequiredFieldValidator, NumericValidator	Starting roll number cannot be null and should be a number
10	endnum	RequiredFieldValidator, NumericValidator	Ending roll number cannot be null and should be a number

5.2 Screenshots







9:24 PM 7.0KB/s 50%

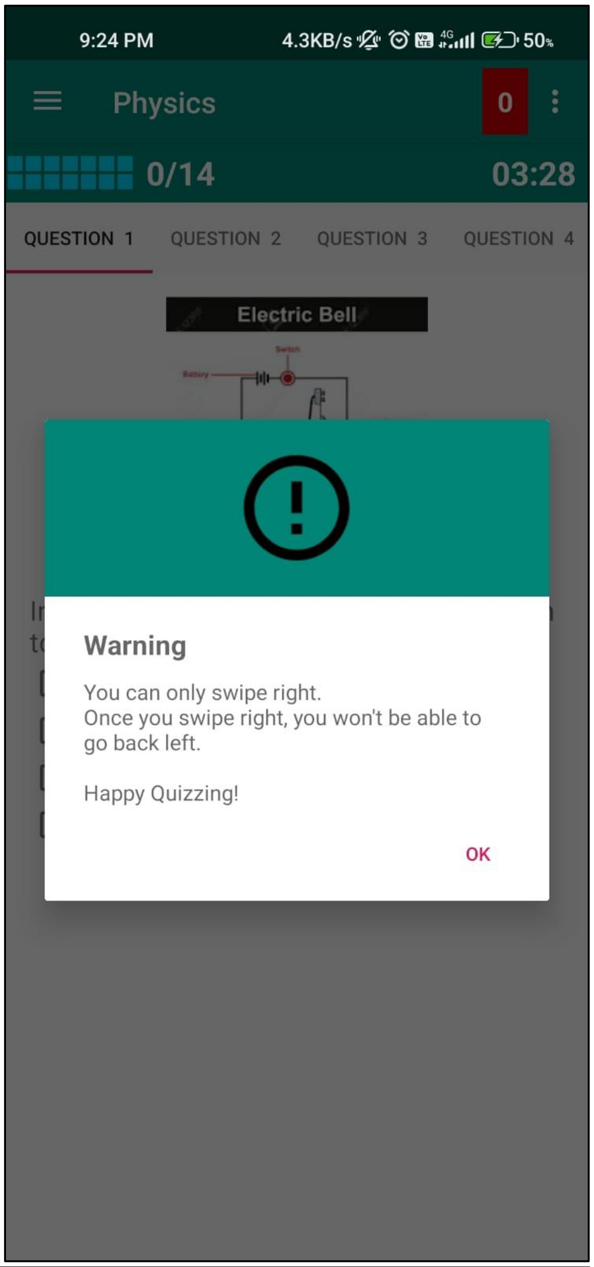
Science Quiz

Choose Class: Physics
- Beginner
- Intermediate
- Advance

Choose Class: Chemistry
- Beginner
- Intermediate
- Advance

Choose Class: Biology
- Beginner
- Intermediate
- Advance

Choose Class: Mixed



9:24 PM 4.3KB/s 50%

Physics 0

0/14 03:28

QUESTION 1 QUESTION 2 QUESTION 3 QUESTION 4

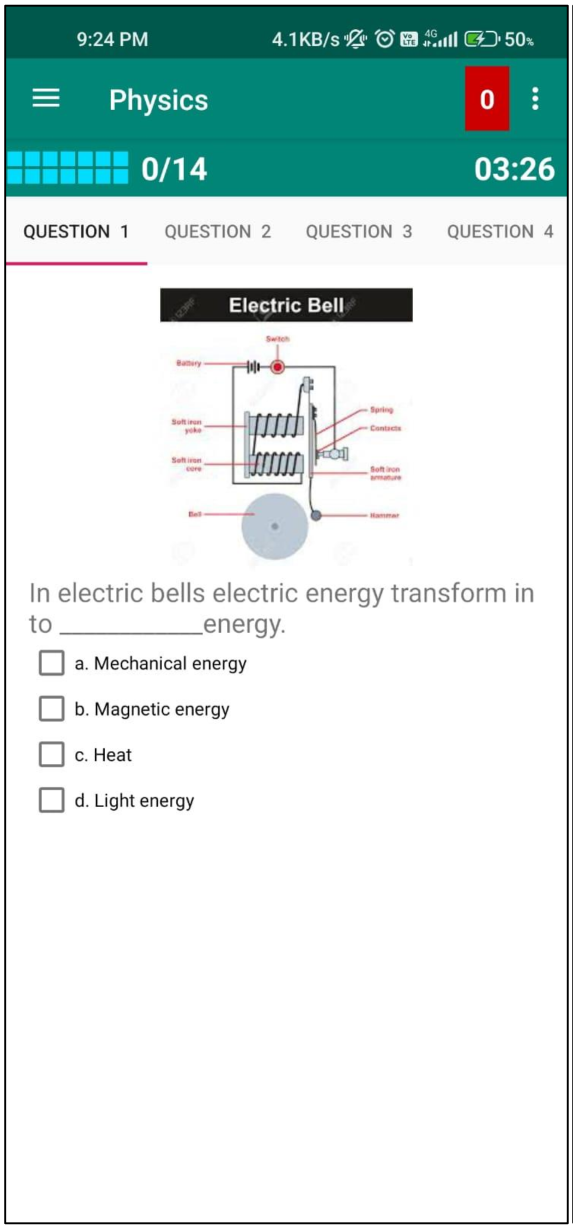
Electric Bell

Warning

You can only swipe right.
Once you swipe right, you won't be able to go back left.

Happy Quizzing!

OK

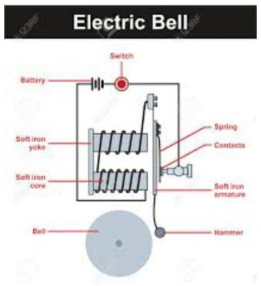


9:24 PM 4.1KB/s 50%

Physics 0

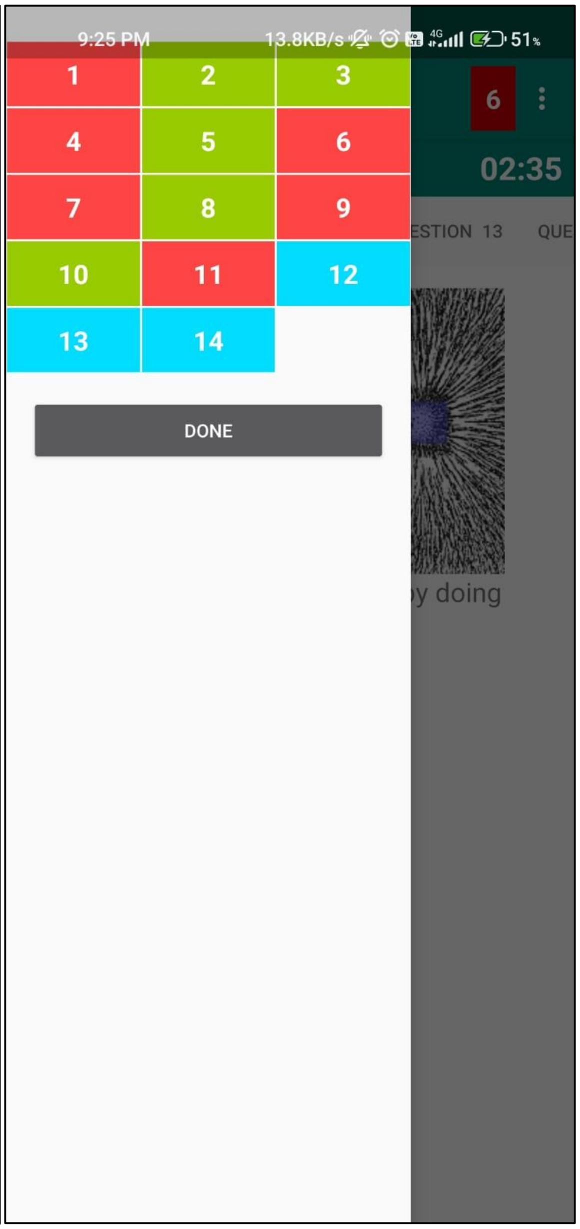
0/14 03:26

QUESTION 1 QUESTION 2 QUESTION 3 QUESTION 4



In electric bells electric energy transform in to ___________energy.

- [ ] a. Mechanical energy
- [ ] b. Magnetic energy
- [ ] c. Heat
- [ ] d. Light energy



9:25 PM 13.8KB/s 51%

1	2	3
4	5	6
7	8	9
10	11	12
13	14	

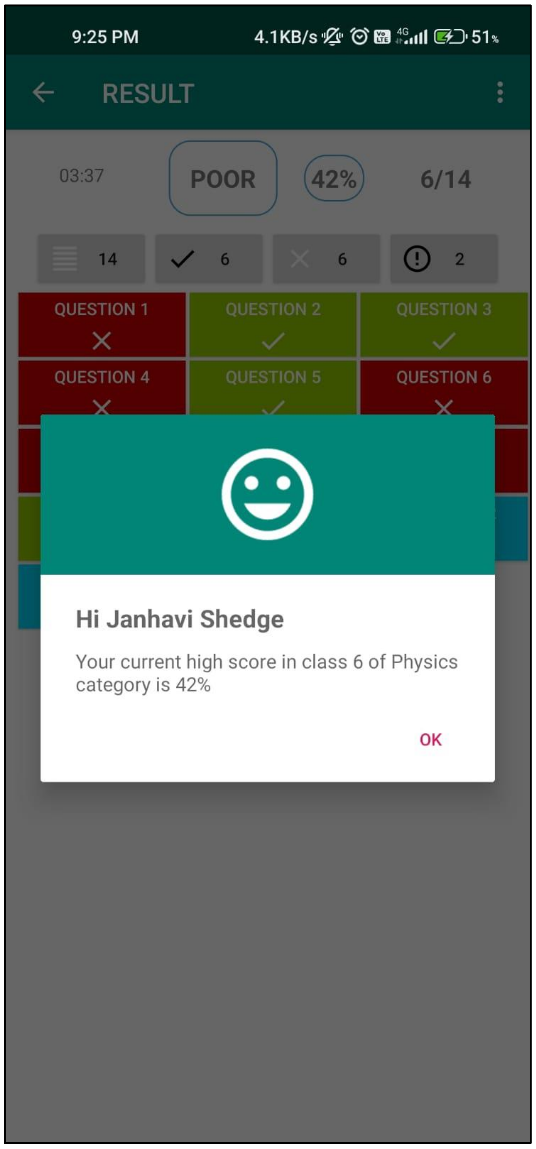
DONE

6

02:35

ESTION 13 QUE

y doing



9:25 PM 4.1KB/s 51%

RESULT

03:37 POOR 42% 6/14

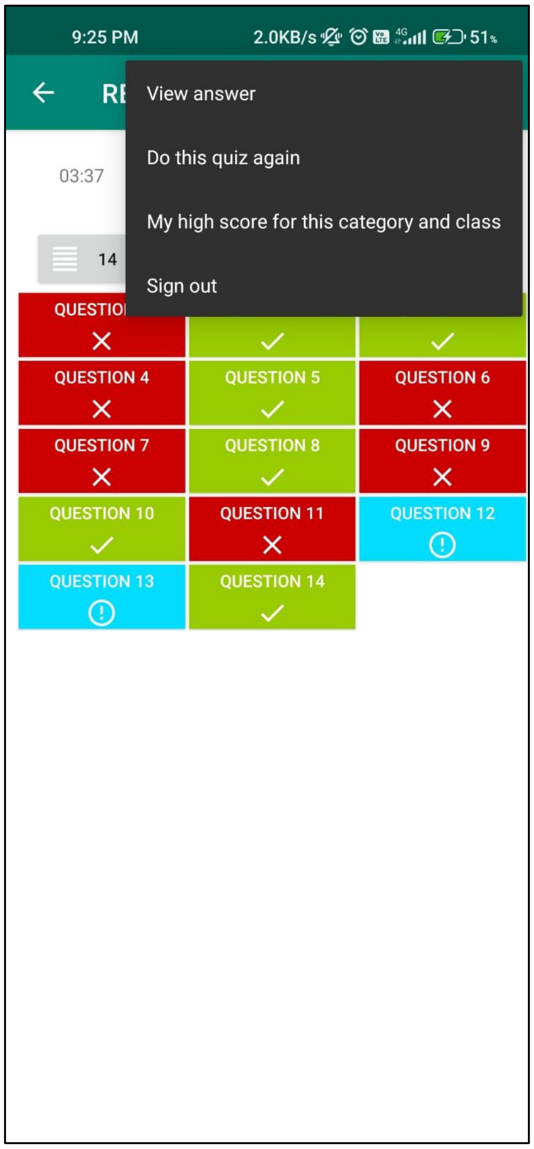
14 ✓ 6 ✗ 6 ! 2

QUESTION 1 QUESTION 2 QUESTION 3 QUESTION 4 QUESTION 5 QUESTION 6

Hi Janhavi Shedge

Your current high score in class 6 of Physics category is 42%

OK



9:25 PM 2.0KB/s 51%

View answer

Do this quiz again

My high score for this category and class

Sign out

03:37

14

QUESTION 1 ✗	✓	✓
QUESTION 4 ✗	QUESTION 5 ✓	QUESTION 6 ✗
QUESTION 7 ✗	QUESTION 8 ✓	QUESTION 9 ✗
QUESTION 10 ✓	QUESTION 11 ✗	QUESTION 12 !
QUESTION 13 !	QUESTION 14 ✓	

6. CONCLUSION AND FUTURE SCOPE

6.1 Future Enhancement

- Security will be broadened.
- Custom Teacher/Student sign-up will be implemented
- Test undertaking proctoring will be added for enhanced security.
- Test will have a live timer.
- Teachers will be able to track student's progress.

6.2 Conclusion

- In the current phase of pandemic, systems like QuizEd are required for the maintenance of education systems functioning.
- With the help of QuizEd, teachers do not have to check papers instead it is done for them directly by the system
- Students can attempt the exams and not be worried about connectivity issues as such as the assessment only happens once the test is submitted

7. REFERENCES

- NCERT SCIENCE BOOKS FROM 6-8TH STANDARDS
- ONLINE PRACTICE QUESTIONS OF SCIENCE FOR 6-8TH STANDARDS
- NATIONAL SCIENCE OLYMPIAD QUESTIONS OF THE RESPECTIVE STANDARDS
- https://www.youtube.com/watch?v=KSW5jyWXs_Y&t=201s
- https://www.geeksforgeeks.org/android-tutorial/

8. ANNEXURE

8.1 Figures list

8.2 Table List