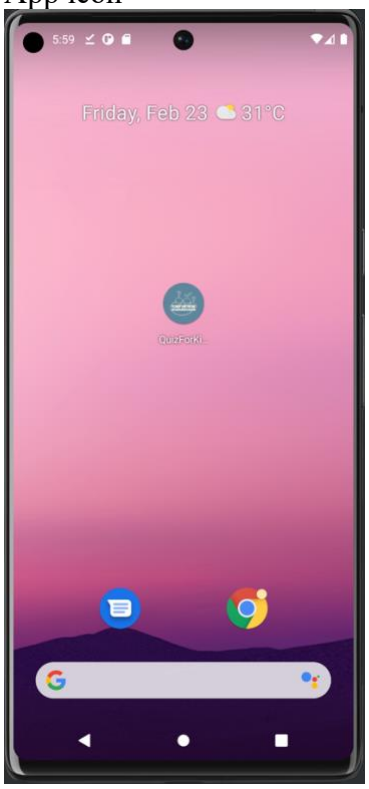
CSIT242 ASSIGNMENT 2

SHINE WAI LU

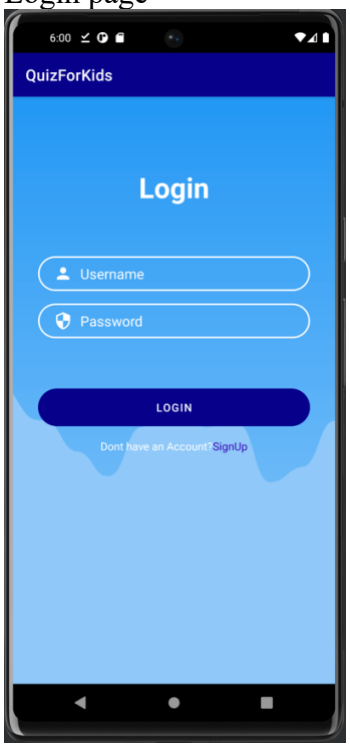
8039963

App icon

- Screen capture of App icon on the emulator
- App icon
- 

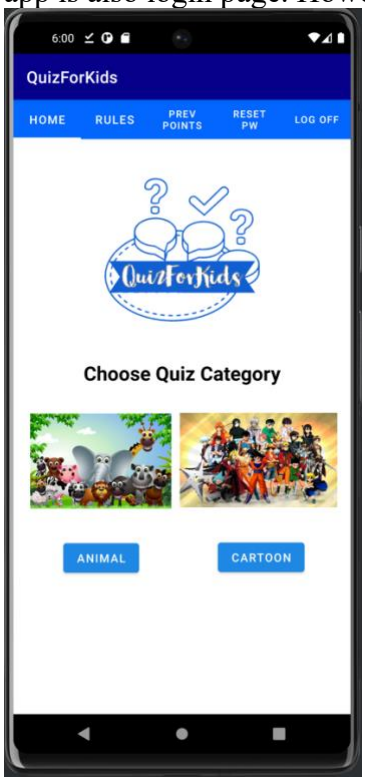
- App icon on the startup page
- 

App logo

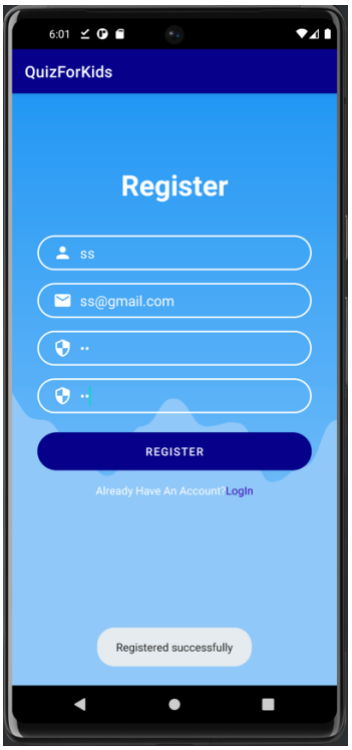
- Screen capture login page
- Login page
- 

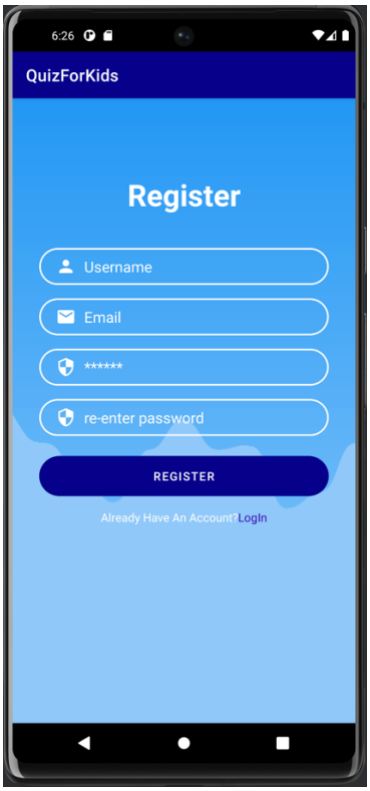
- Screen capture of first page of app

  Login page is the page that firstly seen when opening the app. So, the first page of the app is also login page. However, if the first page is the page after logging in,

- 

Registration

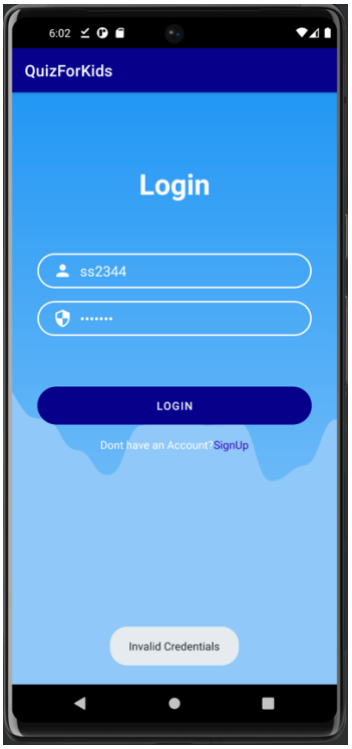
- Create a new user.
- 

- e-mail, username, and password
- 

Login

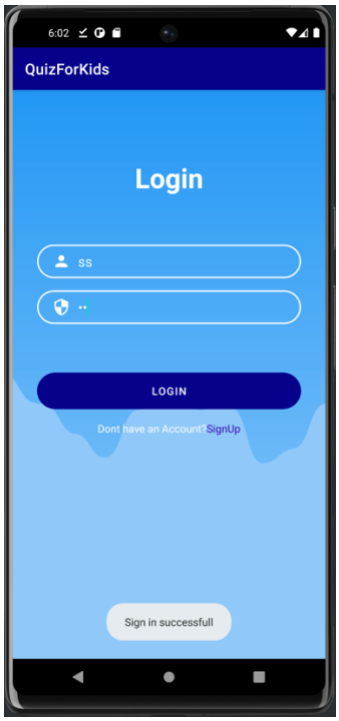
- A case of wrong password and/or wrong username

  Correct credentials are ss and ss, but now there are incorrect credentials.

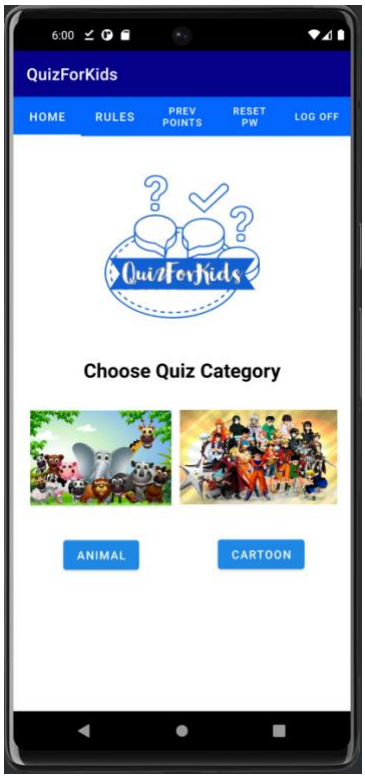
  

-
-
- success case and go to main page

  succeeded

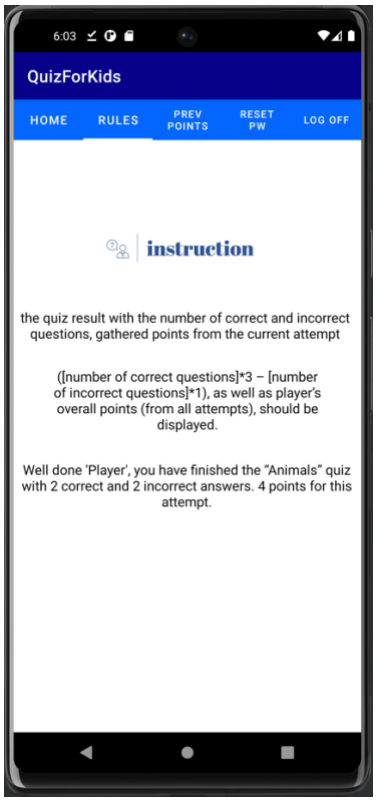


And it goes to the main page.



Instruction

- A page Simple instructions/quiz rules

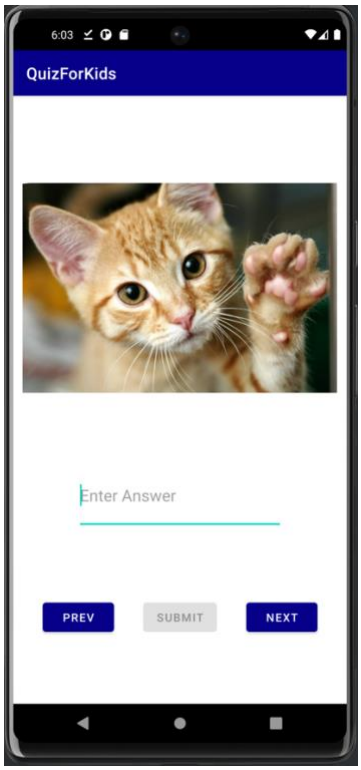


Main app – Animals

- In the form of image and fill in the blanks

The first quiz cannot press previous button, but only next.

- Able to go back and forward between questions.



Able to go back and forward but cannot submit yet.

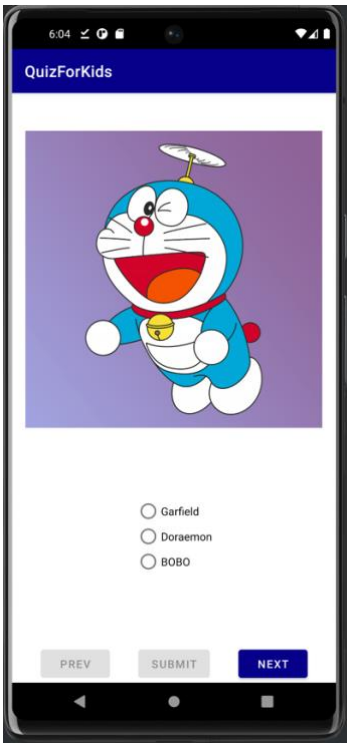
The last page only let you to go back but no more forwarding. Submit button available now.

- Answer 2 questions right and 2 questions wrong
- Show the current points (2*3-2=4 points)
- Show the overall points (4 points)

In the result page, it shows 2 correct answers and 2 incorrect answers, current quiz 4 points and overall points 4 points.

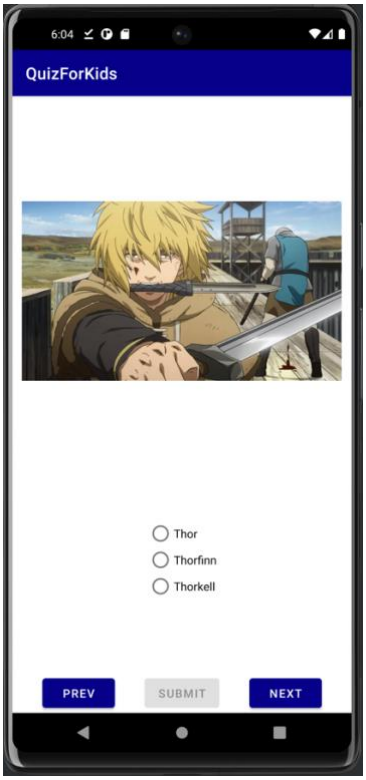
Main app – Cartoons

- In the form of image and MCQs

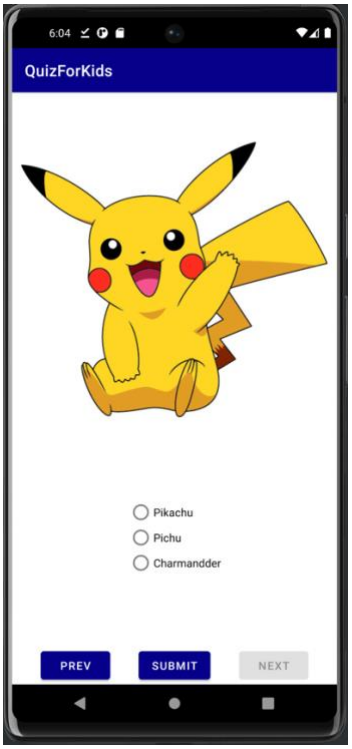


Just like the animal quiz, the prev button will not be available.

- Able to go back and forward between questions.

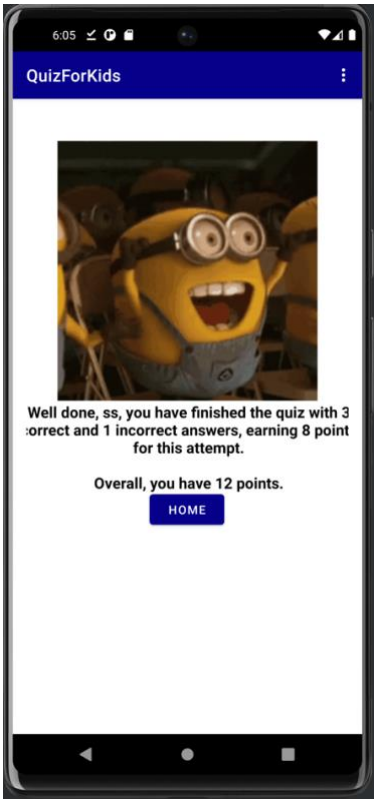


It allow the user to go back and forward.



The last page will let you submit.

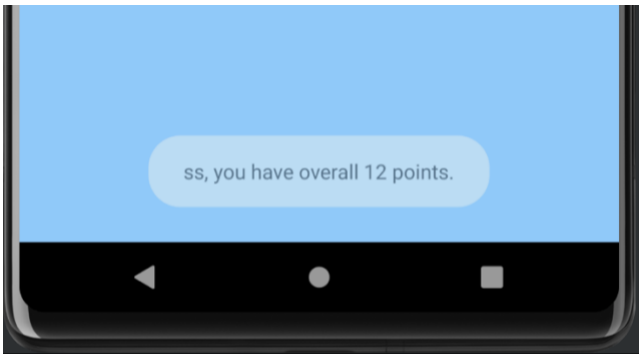
- Answer 3 questions right and 1 question wrong
- Show the current points (3*3-1=8 points)
- Show the overall points (12 points)



Then the user will be able to see 3 right questions and 1 wrong question. The current quiz point will be 8 but the overall will be 12.

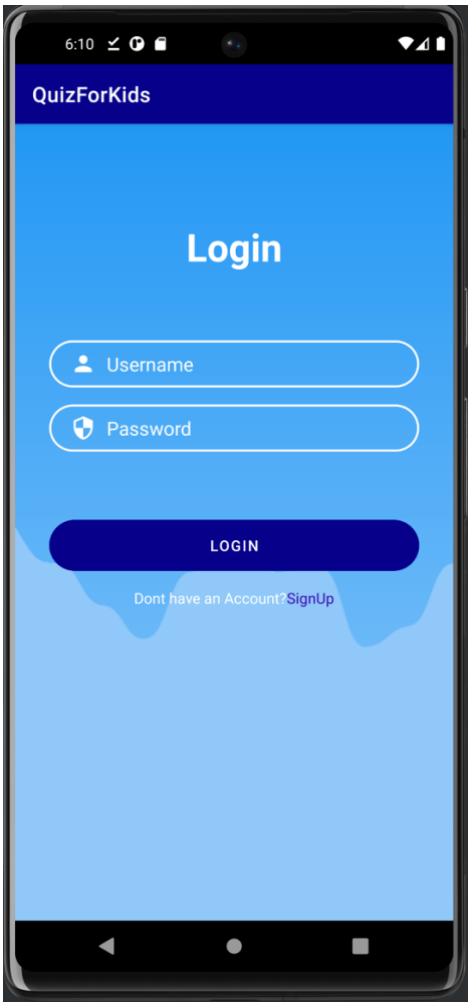
Logoff

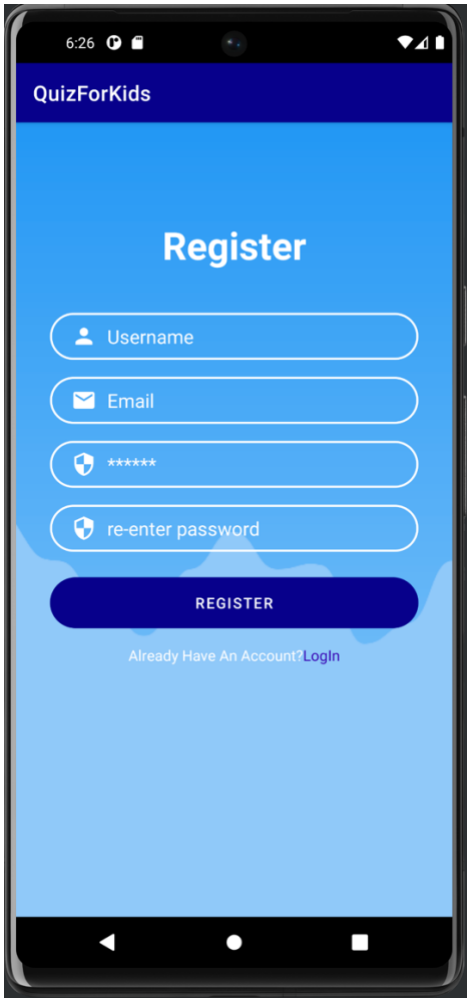
- overall points



The overall points will be shown as a Toast after logging out.

- go to login/register screen





After logging out, you can still relog in or register.

Attempt summary.

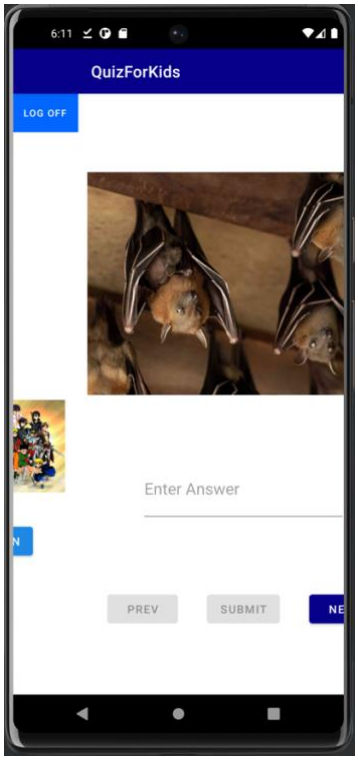
- Login again and display all previous user's attempts and earned point
- Order the previous attempts and earned points by date or by quiz area.

- First capture is sorted by the date and second one is by area. The user can see the overall total points, each quiz points and other respective data like date.

Animations

- Describe and show the various animations implemented in the app

  There are animations from one activity to another activity.



Here is that the user is choosing the animal quiz and the slide goes from the home page to the animal quiz page. The animation is implemented like this in the activity.

```
    @Override
    public void finish() {
        super.finish();
        overridePendingTransition(R.anim.slide_in_left, R.anim.out_r);
    }
}
```

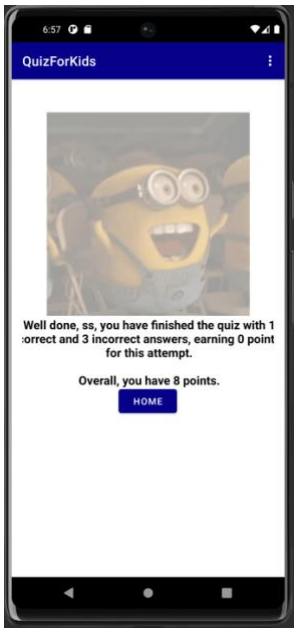
The xml file for those animation effects are as follows:

```
<?xml version="1.0" encoding="utf-8"?>

<set xmlns:android="http://schemas.android.com/apk/res/android">
    <translate
        android:duration="400"
        android:fromXDelta="0"
        android:toXDelta="100%p" />
</set>
```

```
<?xml version="1.0" encoding="utf-8"?>

<set xmlns:android="http://schemas.android.com/apk/res/android">
    <translate
        android:duration="400"
        android:fromXDelta="-100%p"
        android:toXDelta="0" />
</set>
```

All activities are applied with these animations.

The another animation is fading in.



The gif image is slowly fading in after submitting the quiz. The implementation in the result activities is as follows:

```
GifImageView imageView = findViewById(R.id.yay);
Animation animation = AnimationUtils.loadAnimation(getApplicationContext(), R.anim.fade_in);
imageView.startAnimation(animation);
```

The xml file is:

```
<!-- res/anim/fade_in.xml -->
<alpha xmlns:android="http://schemas.android.com/apk/res/android"
    android:interpolator="@android:anim/accelerate_interpolator"
    android:fromAlpha="0.0" android:toAlpha="1.0"
    android:duration="3000" />
```

Database

- A ERD design

- Code to create the database tables for
    a. User details

```java
import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
import androidx.annotation.Nullable;

15 usages
public class DBHelper extends SQLiteOpenHelper {
    1 usage
    public static final String DBNAME = "Login.db";
    7 usages
    public DBHelper(Context context) { super(context, DBNAME, factory: null, version: 1); }


    @Override
    public void onCreate(SQLiteDatabase MyDB) {
        MyDB.execSQL("create Table users(username TEXT primary key, email TEXT, password TEXT, point INTEGER)");
        MyDB.execSQL("create Table attempt(id INTEGER primary key, username TEXT, category TEXT, time TEXT, point INTEGER)");

    }

    10 usages
    @Override
    public void onUpgrade(SQLiteDatabase MyDB, int oldVersion, int newVersion) {
        MyDB.execSQL("DROP TABLE IF EXISTS users");
        MyDB.execSQL("DROP TABLE IF EXISTS attempt");
        onCreate(MyDB);
    }
```

This is my DBHelper for user data and points. First table is for the user detail.

Now this is the inserting data function.

```java
1 usage
public Boolean insertData(String email, String username, String password){
    SQLiteDatabase MyDB = this.getWritableDatabase();
    ContentValues contentValues= new ContentValues();
    contentValues.put("username", username);
    contentValues.put("email", email);
    contentValues.put("password", password);
    contentValues.put("point", 0);
    long result = MyDB.insert( table: "users",  nullColumnHack: null, contentValues);
    if(result==-1) return false;
    else
        return true;
}
```

This function is used for registration.

```java
signup.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        String mail = email.getText().toString();
        String user = username.getText().toString();
        String pass = password.getText().toString();
        String repass = repassword.getText().toString();

        if(user.equals("")||pass.equals("")||mail.equals(""))
            Toast.makeText( context: register.this,  text: "Please enter all the fields", Toast.LENGTH_SHORT).show();
        else{
            if(pass.equals(repass)) {
                Boolean checkuser = DB.checkusername(user);
                Boolean checkmail = DB.checkemail(mail);
                if (checkuser == false || checkmail ==false) {
                    Boolean insert = DB.insertData(mail, user, pass);
                    if (insert == true) {
                        Toast.makeText( context: register.this,  text: "Registered successfully", Toast.LENGTH_SHORT).show();
                        Intent intent = new Intent(getApplicationContext(), MainActivity.class);
                        startActivity(intent);
                    } else {
                        Toast.makeText( context: register.this,  text: "Registration failed", Toast.LENGTH_SHORT).show();
                    }
                } else {
                    Toast.makeText( context: register.this,  text: "User already exists! please sign in", Toast.LENGTH_SHORT).show();
                }
            }
        else{
            Toast.makeText( context: register.this,  text: "Passwords not matching", Toast.LENGTH_SHORT).show();
        }
    } }
});
```

Next functions are checkemail and checkusername:

```
1 usage
public Boolean checkusername(String username) {
    SQLiteDatabase MyDB = this.getWritableDatabase();
    Cursor cursor = MyDB.rawQuery( sql: "Select * from users where username = ?", new String[]{username});
    if (cursor.getCount() > 0)
        return true;
    else
        return false;
}

1 usage
public Boolean checkemail(String email) {
    SQLiteDatabase MyDB = this.getWritableDatabase();
    Cursor cursor = MyDB.rawQuery( sql: "Select * from users where email = ?", new String[]{email});
    if (cursor.getCount() > 0)
        return true;
    else
        return false;
}
```

These functions are used in checking whether same mail and username already existed in register activity.

```
    Toast.makeText( context: register.this,  text: "Please enter all the fields", Toast.LENGTH_SHORT).show();
else{
    if(pass.equals(repass)) {
        Boolean checkuser = DB.checkusername(user);
        Boolean checkmail = DB.checkemail(mail);
        if (checkuser == false || checkmail ==false) {
            Boolean insert = DB.insertData(mail, user, pass);
            if (insert == true) {
                Toast.makeText( context: register.this,  text: "Registered successfully", Toast.LENGTH_SHORT).show()
                Intent intent = new Intent(getApplicationContext(), MainActivity.class);
                startActivity(intent);
            } else {
                Toast.makeText( context: register.this,  text: "Registration failed", Toast.LENGTH_SHORT).show();
```

Next ones are checkusernamepassword function and updatePassword function used for checking credentials to login and resetting password.

```
1 usage
public Boolean checkusernamepassword(String username, String password){
    SQLiteDatabase MyDB = this.getWritableDatabase();
    Cursor cursor = MyDB.rawQuery( sql: "Select * from users where username = ? and password = ?", new String[] {username,password});
    if(cursor.getCount()>0)
        return true;
    else
        return false;
}

1 usage
public boolean updatePassword(String oldPassword, String newPassword) {
    SQLiteDatabase MyDB = this.getWritableDatabase();
    ContentValues contentValues = new ContentValues();
    contentValues.put("password", newPassword);

    int updatedRows = MyDB.update( table: "users", contentValues,  whereClause: "password = ?", new String[]{oldPassword});
    return updatedRows > 0;
}
```

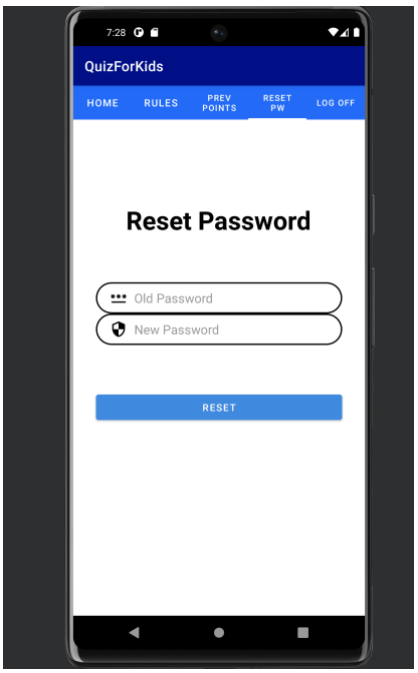
The first one is used in login activity.

```java
String user = username.getText().toString();
String pass = password.getText().toString();
uname = user;

if(user.equals("")||pass.equals(""))
    Toast.makeText( context: login.this,  text: "Please enter all the fields", Toast.LENGTH_SHORT).show();
else{
    Boolean checkuserpass = DB.checkusernamepassword(user, pass);
    if(checkuserpass==true){
        Toast.makeText( context: login.this,  text: "Sign in successfull", Toast.LENGTH_SHORT).show();
        Intent intent  = new Intent(getApplicationContext(), MainActivity.class);
        startActivity(intent);
    }else{
        Toast.makeText( context: login.this,  text: "Invalid Credentials", Toast.LENGTH_SHORT).show();
    }
}
}
);
```

The latter one is used in reset password activity.

```java
resetButton.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        String oldPw = oldPassword.getText().toString().trim();
        String newPw = newPassword.getText().toString().trim();

        if (oldPw.equals(newPw)) {
            Toast.makeText(getActivity(),  text: "Old and new passwords cannot be the same", Toast.LENGTH_SHORT).show();
        } else {
            // Call a method in DBHelper to update the password
            boolean updated = dbHelper.updatePassword(oldPw, newPw);
            if (updated) {
                Toast.makeText(getActivity(),  text: "Password updated successfully", Toast.LENGTH_SHORT).show();
            } else {
                Toast.makeText(getActivity(),  text: "Failed to update password. Please check your old password", Toast.LENGTH_SHOR
            }
        }
    }
});
```

This is UI for reset password that the user can choose as an option at the nav bar.



b. User scores

```java
import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
import androidx.annotation.Nullable;

15 usages
public class DBHelper extends SQLiteOpenHelper {
    1 usage
    public static final String DBNAME = "Login.db";
    7 usages
    public DBHelper(Context context) { super(context, DBNAME, factory: null, version: 1); }


    @Override
    public void onCreate(SQLiteDatabase MyDB) {
        MyDB.execSQL("create Table users(username TEXT primary key, email TEXT, password TEXT, point INTEGER)");
        MyDB.execSQL("create Table attempt(id INTEGER primary key, username TEXT, category TEXT, time TEXT, point INTEGER)");

    }

    10 usages
    @Override
    public void onUpgrade(SQLiteDatabase MyDB, int oldVersion, int newVersion) {
        MyDB.execSQL("DROP TABLE IF EXISTS users");
        MyDB.execSQL("DROP TABLE IF EXISTS attempt");
        onCreate(MyDB);
    }
```

Second table is the user attempt table which is mainly focus on the user scores.

The inserting data to the attempt table.

```java
2 usages
public long insertAttempt(String username, String cate, String time, int points) {
    SQLiteDatabase db = this.getWritableDatabase();
    ContentValues contentValues = new ContentValues();
    contentValues.put("username", username);
    contentValues.put("category", cate);
    contentValues.put("time", time);
    contentValues.put("point", points);
    return db.insert( table: "attempt",  nullColumnHack: null, contentValues);
}
```

This function is used in both result activities for animal quiz and cartoon. Here is some snip of the code.

```java
String area = intent.getStringExtra( name: "area");

if (uname != null) {
    // Add the attempt to the database
    long result = myDB.insertAttempt(uname, area, startTime, points);

    if (result != -1) {
        // Update the user's total points in the login table
        System.out.println(points);
        myDB.updateTotalPoints(uname, points);
        System.out.println("hello");

        // Retrieve the user's overall points
        int overallPoints = myDB.getOverallPoints(uname);

        // Display a congratulatory message with the details of the current attempt and the overall points
        String message = "Well done, " + uname + ", you have finished the quiz with "
                + correctCount + " correct and " + incorrectCount + " incorrect answers, earning "
                + points + " points for this attempt.\n\n" +
                "Overall, you have " + overallPoints + " points.";

        // Set the congratulatory message in the TextView
        text.setText(message);
    }else {
        Toast.makeText( context: this,  text: "Failed to add attempt.", Toast.LENGTH_SHORT).show();
    }
} else {
    Toast.makeText( context: this,  text: "Failed to retrieve username.", Toast.LENGTH_SHORT).show();
}
```

The rest are:

```java
2 usages
public boolean updateTotalPoints(String username, int newPoints) {
    SQLiteDatabase MyDB = this.getWritableDatabase();

    // Retrieve the existing total points for the user
    int existing = getOverallPoints(username);

    // Add the new points to the existing total points
    int total = existing + newPoints;

    ContentValues contentValues = new ContentValues();
    contentValues.put("point", total); // Update the TOTAL_POINTS column with the new total points
    int affectedRows = MyDB.update( table: "users", contentValues,  whereClause: "username = ?", new String[]{username});
    return affectedRows > 0;
}



7 usages
public int getOverallPoints(String username) {
    SQLiteDatabase MyDB = this.getReadableDatabase();
    Cursor cursor = MyDB.rawQuery( sql: "SELECT  point  FROM  users   WHERE  username =?", new String[]{username});
    int overallPoints = 0;
    if (cursor != null && cursor.moveToFirst()) {
        overallPoints = cursor.getInt(cursor.getColumnIndex( columnName: "point"));
        cursor.close();
    }
    return overallPoints;
}

1 usage
public Cursor getAttemptsByUsername(String username, String sortOrder) {
    SQLiteDatabase MyDB = this.getReadableDatabase();
    return MyDB.query( table: "attempt",  columns: null,  selection: "username =?", new String[]{username},  groupBy: null,  having: null, sortOrder);
}

}
```

First one is used for updating the total points every time the user finished the quizzes. It is used in result activity that needs to be updated after answering a quiz.

```java
if (result != -1) {
    // Update the user's total points in the login table
    System.out.println(points);
    myDB.updateTotalPoints(uname, points);
    System.out.println("hello");

    // Retrieve the user's overall points
    int overallPoints = myDB.getOverallPoints(uname);

    // Display a congratulatory message with the details of the current attempt and the overall
    String message = "Well done, " + uname + ", you have finished the quiz with "
            + correctCount + " correct and " + incorrectCount + " incorrect answers, earning "
            + points + " points for this attempt.\n\n" +
            "Overall, you have " + overallPoints + " points.";

    // Set the congratulatory message in the TextView
    text.setText(message);
}else {
    Toast.makeText( context: this,  text: "Failed to add attempt.", Toast.LENGTH_SHORT).show();
}
} else {
    Toast.makeText( context: this,  text: "Failed to retrieve username.", Toast.LENGTH_SHORT).show();
}
```

The second one is getting overall points. The method is used for many times in the code as it is the main focus of the user score. Here is an example snippet of the code.

```java
@Override
public void onViewCreated(@NonNull View view, @Nullable Bundle savedInstanceState) {
    super.onViewCreated(view, savedInstanceState);

    db = new DBHelper(requireContext());
    int overallPoints = db.getOverallPoints(uname);


    final Button btn = (Button) (this.getView().findViewById(R.id.logout));

    btn.setOnClickListener((v -> {
        String message = uname + ", you have overall " + overallPoints + " points.";
        Toast.makeText(requireContext(), message, Toast.LENGTH_SHORT).show();
        Intent intent  = new Intent(getActivity(), login.class);
        startActivity(intent);
        getActivity().finish();
    }

            ));


}
```

When logging off, to show the overall point, this function must be called.

c. Questions

For questions, I created 2 database helper functions for each quiz.

```java
1 usage
public AnimalDBHelper(@Nullable Context context, @Nullable String name,
                      @Nullable SQLiteDatabase.CursorFactory factory, int version) {
    super(context, name, factory, version);
    this.mContext = context;
    addDefaultData();
}

//creating the table
@Override
public void onCreate(SQLiteDatabase db) {
    db.execSQL("CREATE TABLE " + TABLE_NAME +
            "(ID INTEGER PRIMARY KEY AUTOINCREMENT," +
            "IMAGE TEXT, ANSWER TEXT)");


}

// called when the database needs to be upgraded to the new schema version
10 usages
@Override
public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
    db.execSQL("DROP TABLE IF EXISTS " + TABLE_NAME);
    onCreate(db);
}
```

Database helper for animal quiz.

```java
public CartoonDBHelper(@Nullable Context context, @Nullable String name,
                       @Nullable SQLiteDatabase.CursorFactory factory, int version) {
    super(context, name, factory, version);
    this.mContext = context;
    addDefaultData();
}

//creating the table
@Override
public void onCreate(SQLiteDatabase db) {
    db.execSQL("CREATE TABLE " + TABLE_NAME +
            "(ID INTEGER PRIMARY KEY AUTOINCREMENT," +
            "IMAGE TEXT, ANSWER TEXT, CHOICE1 TEXT, CHOICE2 TEXT, CHOICE3 TEXT)");
}

// called when the database needs to be upgraded to the new schema version
10 usages
@Override
public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
    db.execSQL("DROP TABLE IF EXISTS " + TABLE_NAME);
    onCreate(db);
}
```

Database helper for cartoon quiz.

- Code to create the actual questions.

For animal quiz,

```java
10 usages
public long insertAnimalAnswer(String image, String answer) {
    SQLiteDatabase db = this.getWritableDatabase();
    ContentValues contentValues = new ContentValues();
    contentValues.put(COL_2, image);
    contentValues.put(COL_3, answer);
    return db.insert(TABLE_NAME, nullColumnHack: null, contentValues);
}
```

Adding questions.

```java
1 usage
private void addDefaultData() {
    SQLiteDatabase db = this.getWritableDatabase();

    // Add sample data for animal quiz answers and image paths
    insertAnimalAnswer( image: "tiger",  answer: "tiger");
    insertAnimalAnswer( image: "panda",  answer: "panda");
    insertAnimalAnswer( image: "cat",  answer: "cat");
    insertAnimalAnswer( image: "dog",  answer: "dog");
    insertAnimalAnswer( image: "mouse",  answer: "mouse");
    insertAnimalAnswer( image: "lion",  answer: "lion");
    insertAnimalAnswer( image: "bat",  answer: "bat");
    insertAnimalAnswer( image: "zebra",  answer: "zebra");
    insertAnimalAnswer( image: "fish",  answer: "fish");
    insertAnimalAnswer( image: "owl",  answer: "owl");
}
```

For cartoon quiz,

```
10 usages
public long insertCartoonData(String image, String answer, String choice1, String choice2) {
    SQLiteDatabase db = this.getWritableDatabase();
    ContentValues contentValues = new ContentValues();
    contentValues.put(COL_2, image);
    contentValues.put(COL_3, answer);
    contentValues.put(COL_4, choice1);
    contentValues.put(COL_5, choice2);

    return db.insert(TABLE_NAME, nullColumnHack: null, contentValues);
}
```

Adding questions.

```
1 usage
private void addDefaultData() {
    SQLiteDatabase db = this.getWritableDatabase();

    // Add sample data for animal quiz answers and image paths
    insertCartoonData( image: "batman", answer: "BatMan", choice1: "BirdMan", choice2: "BlackMan");
    insertCartoonData( image: "pikachu", answer: "Pikachu", choice1: "Charmandder", choice2: "Pichu");
    insertCartoonData( image: "aang", answer: "Aang", choice1: "Killua", choice2: "Dororo");
    insertCartoonData( image: "tom", answer: "Tom", choice1: "Jerry", choice2: "Puss in boots");
    insertCartoonData( image: "doraemon", answer: "Doraemon", choice1: "Garfield", choice2: "BOBO");
    insertCartoonData( image: "goku", answer: "Goku", choice1: "Doku", choice2: "Deku");
    insertCartoonData( image: "luffy", answer: "Luffy", choice1: "Zoro", choice2: "Saitama");
    insertCartoonData( image: "naruto", answer: "Naruto", choice1: "Ichigo", choice2: "Boruto");
    insertCartoonData( image: "scoobydoo", answer: "Scooby-Doo", choice1: "Droopy", choice2: "Bolt");
    insertCartoonData( image: "thorfinn", answer: "Thorfinn", choice1: "Thor", choice2: "Thorkell");


}
```