Abstract:

We try to participate in a variety of activities at university rather than focusing only on our academics. Thus, the CLUB MANAGEMENT **(RUET DEBATING CLUB)** app, which is an Android app, could assist us in making clubbing more convenient while also balancing clubbing with our academics.

By this project, club members can easily stay connected with each other and updated. Also, the overall complexity of the registration process could be reduced. Time management might also be improved. Again, an overview of member details can be observed and maintained through this project.

Project IDE and tools :

- Android Studio
- Language : Java
- XML
- Database : Firebase

Main Features:

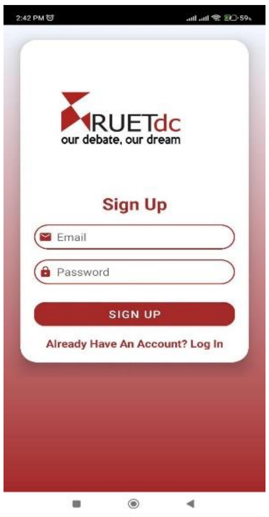
This Android app will include sections, for clients to login, register, and read notices uploaded by the admin/admins. For admins, there will be too section for login, uploading notices/activities and member summary.

The most crucial feature of the app is that the ADMIN is also a CLIENT/MEMBER. Thus, the admin too has to sign up and register form to proceed.

App Components:

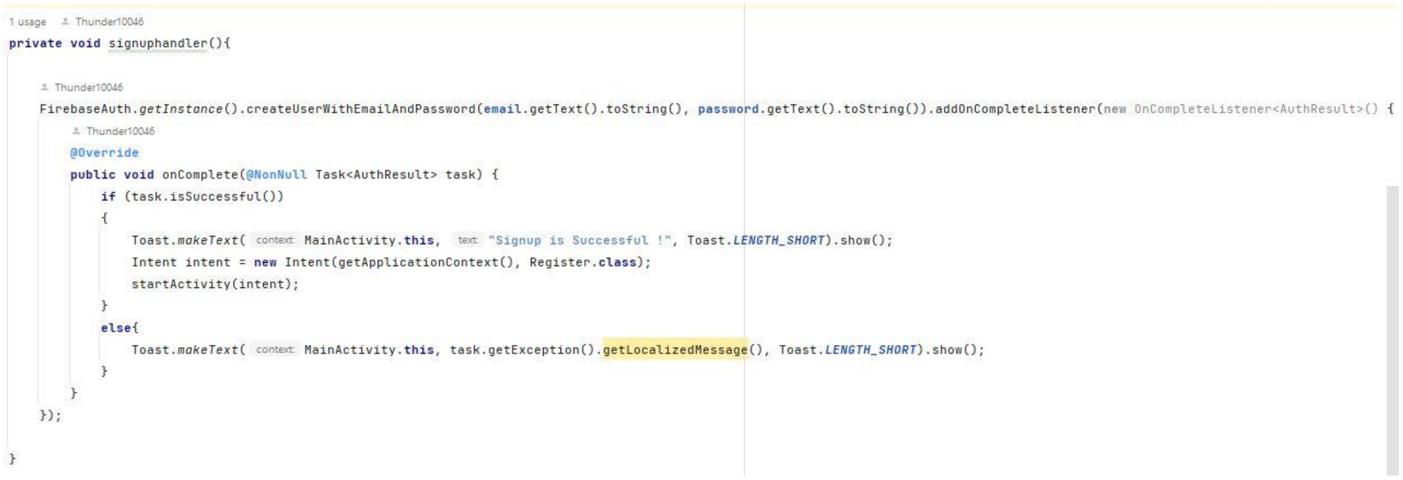
Using Android Studio, I have developed this app with xml layouts respective to the user interface and also for the admin respectively. To activate functionality to the design/xml files, I have used JAVA files respectively. Following is an overall description of the XML files along with their functionality:

1. activity_main.xml (Sign Up Activity) :



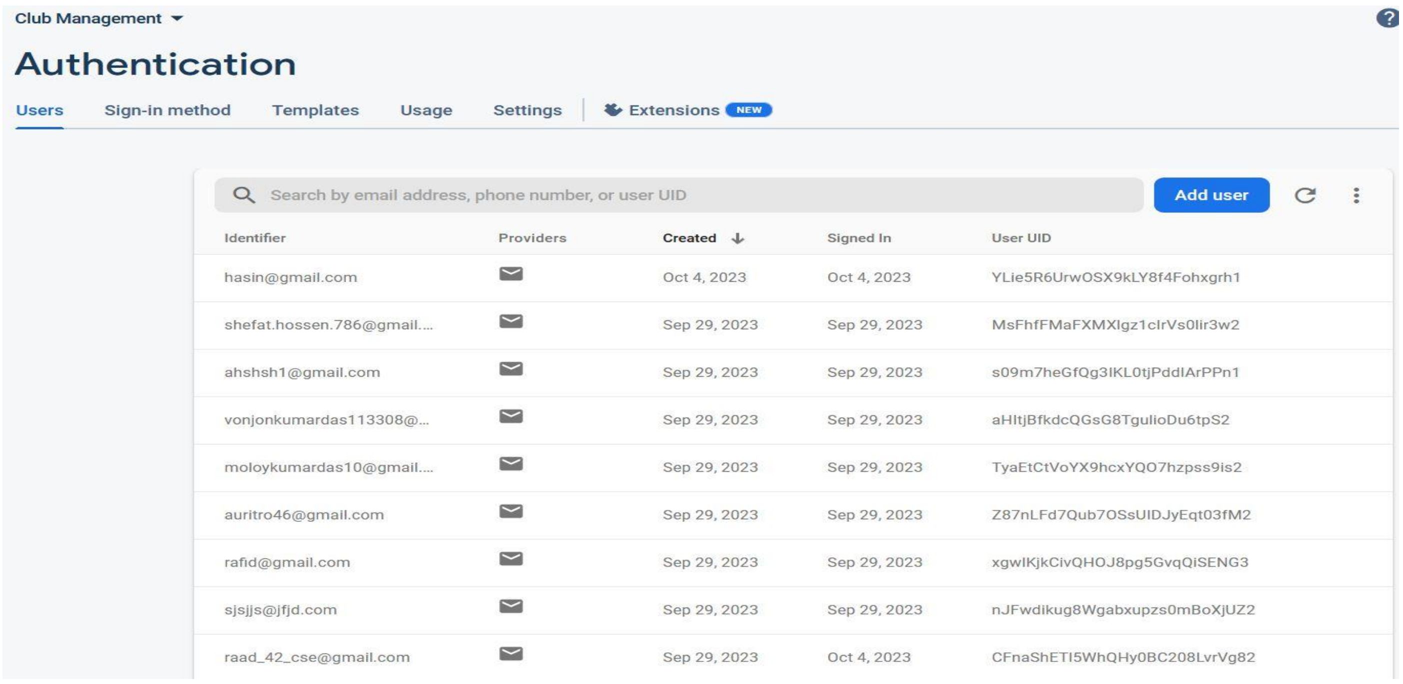
User has to sign up to proceed for registration by email and password. Two EditText field has been used with a button "SIGN UP". Below the button, a TextView is also used so that if the user is already registered, he/she should be moved to Log in activity.

Firebase, Inc. is a set of backend cloud computing services and application development platforms provided by Google. I have used FirebaseAuth class and its method creatueUserWithEmailAndPassword. This method requires two credentials – email and password. If there is no error, a new account would be created with an unique ID **(UID)** which is used afterwards for saving data using Realtime Database of Firebase. The code for creating account is showed below:



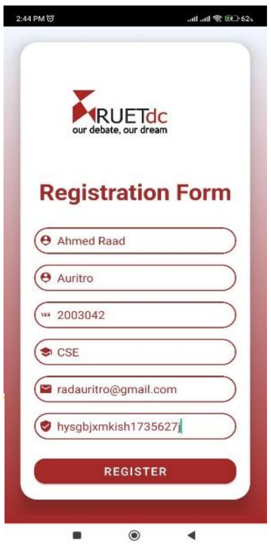
```
1 usage  Thunder10046
private void signuphandler(){

    Thunder10046
    FirebaseAuth.getInstance().createUserWithEmailAndPassword(email.getText().toString(), password.getText().toString()).addOnCompleteListener(new OnCompleteListener<AuthResult>() {
        Thunder10046
        @Override
        public void onComplete(@NonNull Task<AuthResult> task) {
            if (task.isSuccessful())
            {
                Toast.makeText( context MainActivity.this,  text "Signup is Successful !", Toast.LENGTH_SHORT).show();
                Intent intent = new Intent(getApplicationContext(), Register.class);
                startActivity(intent);
            }
            else{
                Toast.makeText( context MainActivity.this, task.getException().getLocalizedMessage(), Toast.LENGTH_SHORT).show();
            }
        }
    });

}
```

Created Users in the Firebase is stored as :



Identifier	Providers	Created	Signed In	User UID
hasin@gmail.com	✉	Oct 4, 2023	Oct 4, 2023	YLie5R6UrwOSX9kLY8f4Fohxgrh1
shefat.hossen.786@gmail....	✉	Sep 29, 2023	Sep 29, 2023	MsFhfFMaFXMXlgz1clrVs0lir3w2
ahshsh1@gmail.com	✉	Sep 29, 2023	Sep 29, 2023	s09m7heGfQg3IKL0tjPddIArPPn1
vonjonkumardas113308@...	✉	Sep 29, 2023	Sep 29, 2023	aHltjBfkdcQGsG8TgulioDu6tpS2
moloykumardas10@gmail....	✉	Sep 29, 2023	Sep 29, 2023	TyaEtCtVoYX9hcxYQO7hzpss9is2
auritro46@gmail.com	✉	Sep 29, 2023	Sep 29, 2023	Z87nLFd7Qub7OSsUIDJyEqt03fM2
rafid@gmail.com	✉	Sep 29, 2023	Sep 29, 2023	xgwIKjkCivQHOJ8pg5GvqQiSENG3
sjsjjs@jfjd.com	✉	Sep 29, 2023	Sep 29, 2023	nJFwdikug8Wgabxupzs0mBoXjUZ2
raad_42_cse@gmail.com	✉	Sep 29, 2023	Oct 4, 2023	CFnaShETI5WhQHy0BC208LvrVg82

2. activity_register.xml (Registration form) :



2:44 PM

RUETdc
our debate, our dream

Registration Form

Ahmed Raad

Auritro

2003042

CSE

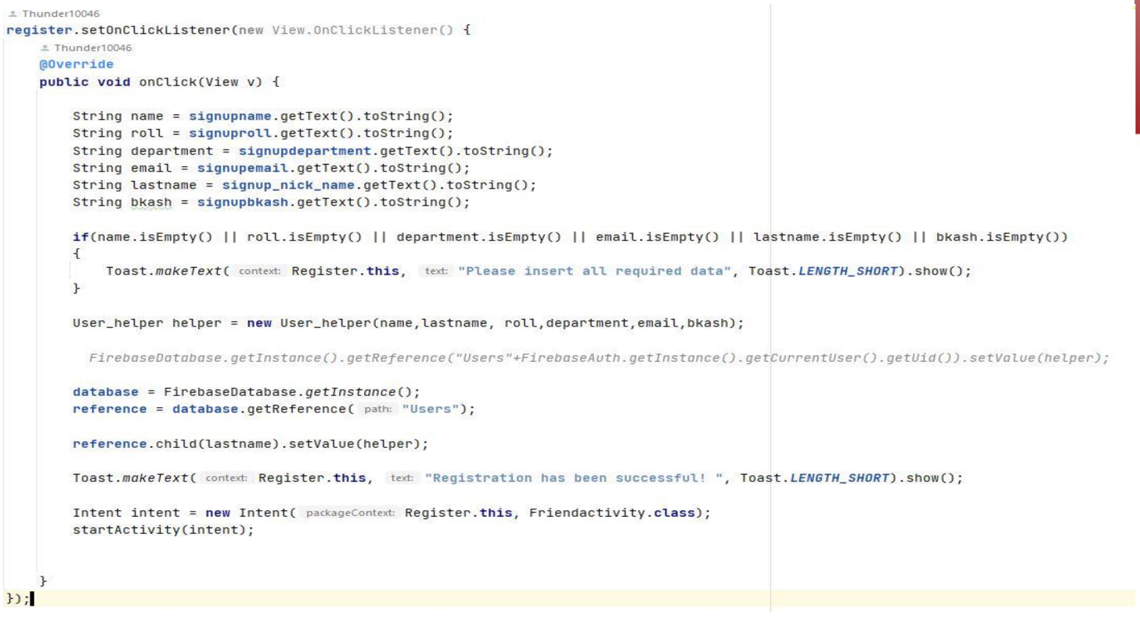
radauritro@gmail.com

hysgbjxmkish1735627j

REGISTER

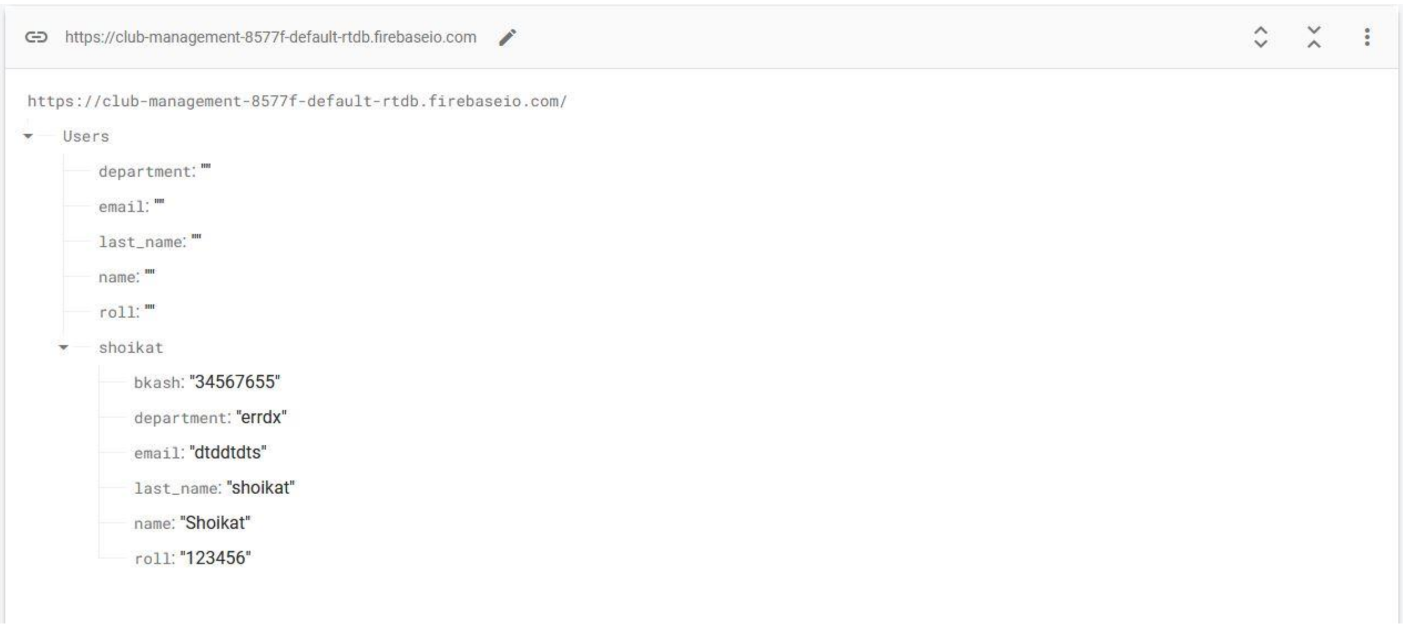
After a user is signed in, he/she has to fill up the registration form accordingly to UI.

After pressing the button, the functionality begins by calling a reference from Firebase Realtime Database respectively to the user's **UID.** A java class **User_helper** is used so that it can fetch values (parameters) of each user and set the values according to the **nickname** which will be the node for respective user in the Realtime Database. Also, all the users (nicknames) will be under a node named "Users". After registering, the user would move to a feed page which is defined as activity_friend.xml, which is generally shows the notices uploaded by the admin. But, the code for the functionality of Registration is shown below:



```
Thunder10046
register.setOnClickListener(new View.OnClickListener() {
    Thunder10046
    @Override
    public void onClick(View v) {

        String name = signupname.getText().toString();
        String roll = signuproll.getText().toString();
        String department = signupdepartment.getText().toString();
        String email = signupemail.getText().toString();
        String lastname = signup_nick_name.getText().toString();
        String bkash = signupbkash.getText().toString();

        if(name.isEmpty() || roll.isEmpty() || department.isEmpty() || email.isEmpty() || lastname.isEmpty() || bkash.isEmpty())
        {
            Toast.makeText( context: Register.this,  text: "Please insert all required data", Toast.LENGTH_SHORT).show();
        }

        User_helper helper = new User_helper(name,lastname, roll,department,email,bkash);

          FirebaseDatabase.getInstance().getReference("Users"+FirebaseAuth.getInstance().getCurrentUser().getUid()).setValue(helper);

        database = FirebaseDatabase.getInstance();
        reference = database.getReference( path: "Users");

        reference.child(lastname).setValue(helper);

        Toast.makeText( context: Register.this,  text: "Registration has been successful! ", Toast.LENGTH_SHORT).show();

        Intent intent = new Intent( packageContext: Register.this, Friendactivity.class);
        startActivity(intent);


    }
});
```

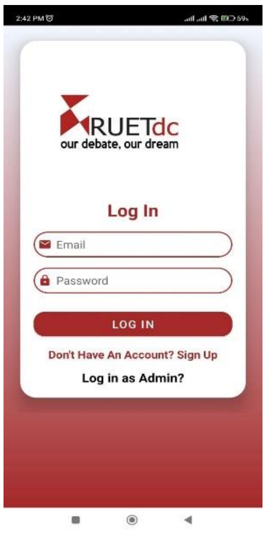
In Firebase Realtime Database, it shows as :



https://club-management-8577f-default-rtdb.firebaseio.com

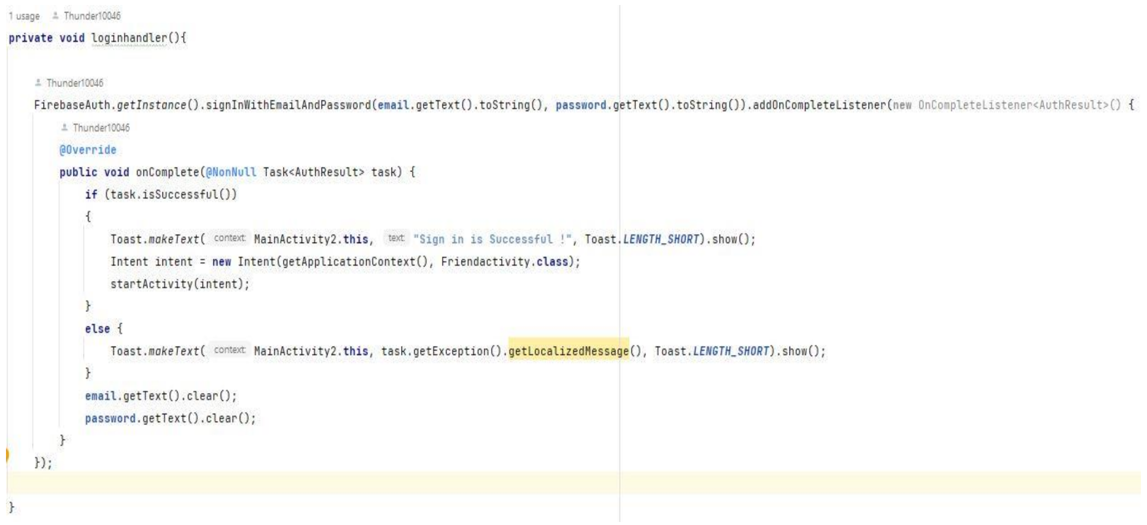
```
https://club-management-8577f-default-rtdb.firebaseio.com/
▾ Users
    department: ""
    email: ""
    last_name: ""
    name: ""
    roll: ""
  ▾ shoikat
      bkash: "34567655"
      department: "errdx"
      email: "dtddtdts"
      last_name: "shoikat"
      name: "Shoikat"
      roll: "123456"
```

3. activity_main2.xml (Log In Activity) :



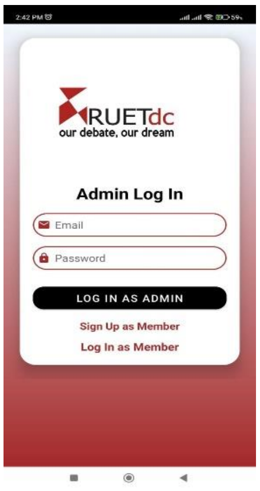
Similar to Sign Up activity, this activity is to log in. With valid email and password, users would log in and they would be moved to activity_friend.xml. Unless they have previously Signed Up, "LOG IN" button would through exceptions with generalized localized messages.

A TextView is also used so that if the user didn't create an account, he/she could move to Sign Up. Also, the message for Log in as Admin is shown here. If pressed, a new intent would open which mainly works for admin log in.



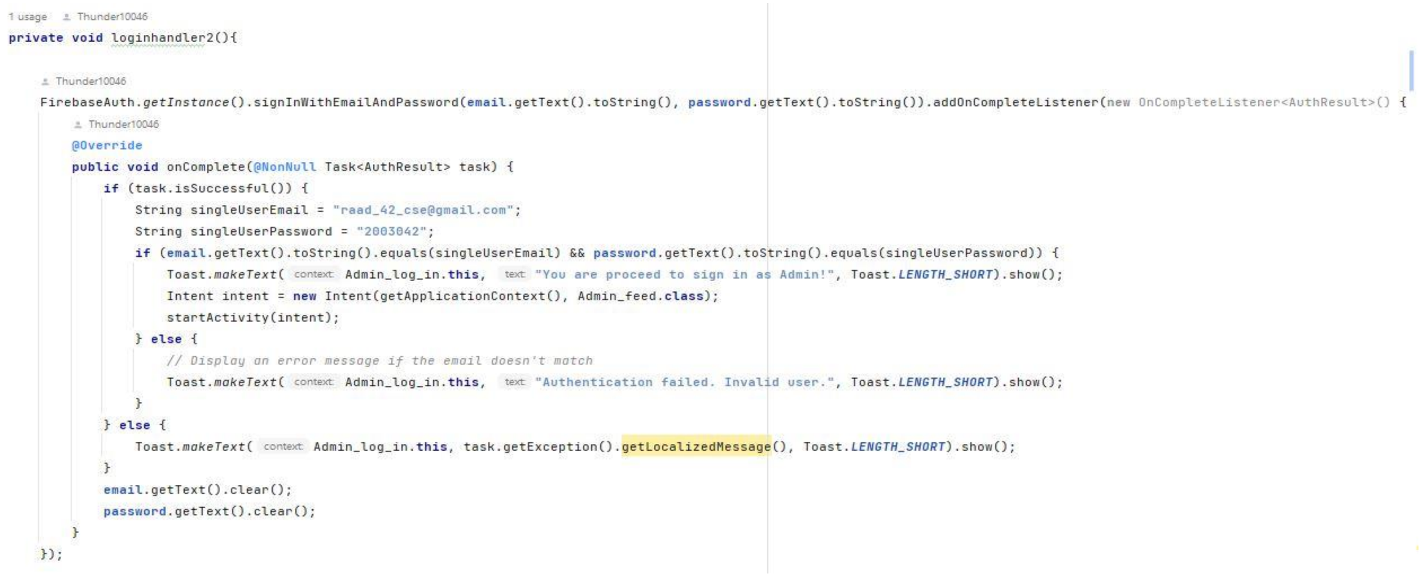
```
1 usage   Thunder10046
private void loginhandler(){

    Thunder10046
    FirebaseAuth.getInstance().signInWithEmailAndPassword(email.getText().toString(), password.getText().toString()).addOnCompleteListener(new OnCompleteListener<AuthResult>() {
        Thunder10046
        @Override
        public void onComplete(@NonNull Task<AuthResult> task) {
            if (task.isSuccessful())
            {
                Toast.makeText( context MainActivity2.this,  text "Sign in is Successful !", Toast.LENGTH_SHORT).show();
                Intent intent = new Intent(getApplicationContext(), Friendactivity.class);
                startActivity(intent);
            }
            else {
                Toast.makeText( context MainActivity2.this, task.getException().getLocalizedMessage(), Toast.LENGTH_SHORT).show();
            }
            email.getText().clear();
            password.getText().clear();
        }
    });

}
```

4. activity_admin_log_in.xml (Admin Log In Activity) :



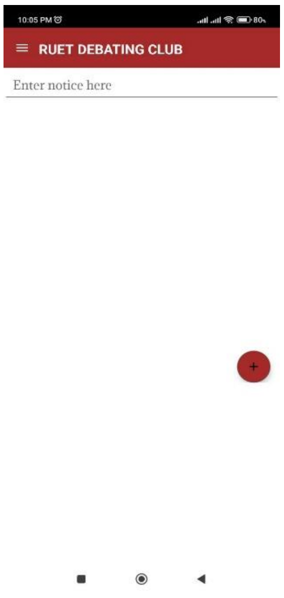
Log in for admin could only be proceed if two conditions are satisfied. One is, the admin must be pre-registered as "a user". Second one is, the email and password must match which must be predefined by the developer.

After pressing the button "LOG IN AS ADMIN", a new interface would open, named as activity_admin_feed.xml. Besides, if a user is introduced by this interface, he/she can go back to Sign Up or Log In by pressing following TextViews which are below the button respectively.



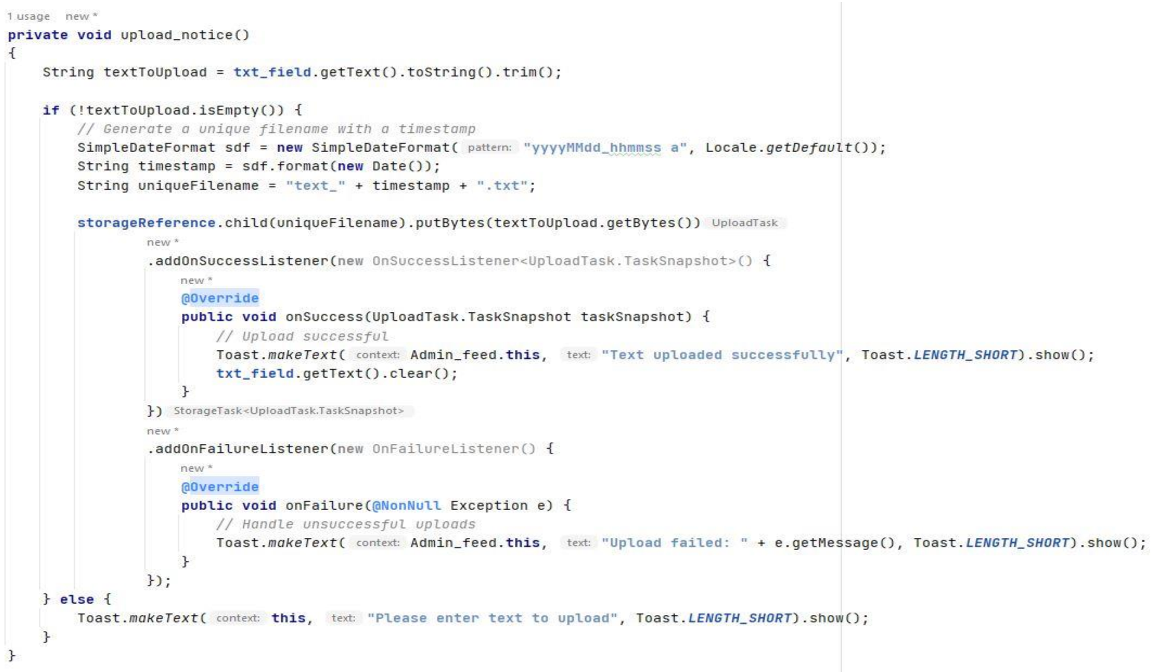
```
1 usage   Thunder10046
private void loginhandler2(){

    Thunder10046
    FirebaseAuth.getInstance().signInWithEmailAndPassword(email.getText().toString(), password.getText().toString()).addOnCompleteListener(new OnCompleteListener<AuthResult>() {
        Thunder10046
        @Override
        public void onComplete(@NonNull Task<AuthResult> task) {
            if (task.isSuccessful()) {
                String singleUserEmail = "raad_42_cse@gmail.com";
                String singleUserPassword = "2003042";
                if (email.getText().toString().equals(singleUserEmail) && password.getText().toString().equals(singleUserPassword)) {
                    Toast.makeText( context Admin_log_in.this,  text "You are proceed to sign in as Admin!", Toast.LENGTH_SHORT).show();
                    Intent intent = new Intent(getApplicationContext(), Admin_feed.class);
                    startActivity(intent);
                } else {
                    // Display an error message if the email doesn't match
                    Toast.makeText( context Admin_log_in.this,  text "Authentication failed. Invalid user.", Toast.LENGTH_SHORT).show();
                }
            } else {
                Toast.makeText( context Admin_log_in.this, task.getException().getLocalizedMessage(), Toast.LENGTH_SHORT).show();
            }
            email.getText().clear();
            password.getText().clear();
        }
    });
```

5. activity_admin_feed.xml (Admin Notice Upload Activity) :



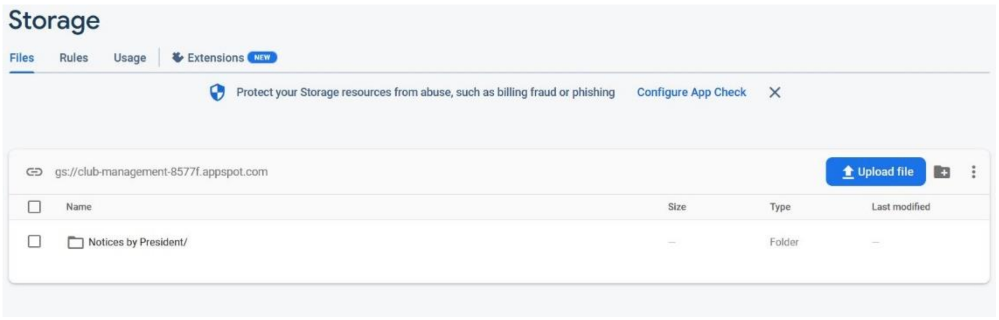
In this particular interface, functionality is given to the EditText field and a Floating Action Button. After writing the notice to the EditText field, if the admin press the button (symbolized as add), the notice would be updated and further, the updated/uploaded notice would be shown in activity_friend.xml interface which is developed for users so that they can see the notices.

Firebase Storage is used in this case and the code for the functionality is shown below:



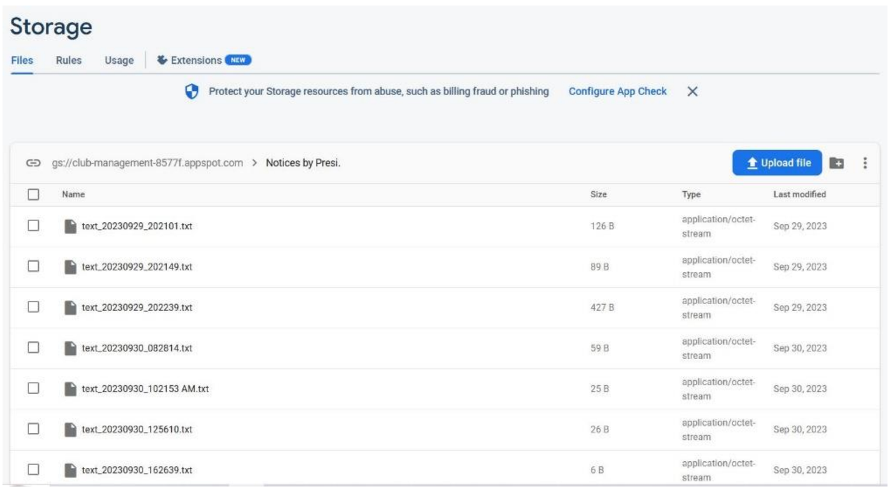
```
1 usage   new *
private void upload_notice()
{
    String textToUpload = txt_field.getText().toString().trim();

    if (!textToUpload.isEmpty()) {
        // Generate a unique filename with a timestamp
        SimpleDateFormat sdf = new SimpleDateFormat( pattern: "yyyyMMdd_hhmmss a", Locale.getDefault());
        String timestamp = sdf.format(new Date());
        String uniqueFilename = "text_" + timestamp + ".txt";

        storageReference.child(uniqueFilename).putBytes(textToUpload.getBytes()) UploadTask
                new *
                .addOnSuccessListener(new OnSuccessListener<UploadTask.TaskSnapshot>() {
                    new *
                    @Override
                    public void onSuccess(UploadTask.TaskSnapshot taskSnapshot) {
                        // Upload successful
                        Toast.makeText( context: Admin_feed.this,  text: "Text uploaded successfully", Toast.LENGTH_SHORT).show();
                        txt_field.getText().clear();
                    }
                }) StorageTask<UploadTask.TaskSnapshot>
                new *
                .addOnFailureListener(new OnFailureListener() {
                    new *
                    @Override
                    public void onFailure(@NonNull Exception e) {
                        // Handle unsuccessful uploads
                        Toast.makeText( context: Admin_feed.this,  text: "Upload failed: " + e.getMessage(), Toast.LENGTH_SHORT).show();
                    }
                });
    } else {
        Toast.makeText( context: this,  text: "Please enter text to upload", Toast.LENGTH_SHORT).show();
    }
}
```

Some updated/uploaded notices in Firebase storage :


Storage
Files
Rules
Usage
Extensions NEW
Protect your Storage resources from abuse, such as billing fraud or phishing
Configure App Check
Upload file

gs://club-management-8577f.appspot.com

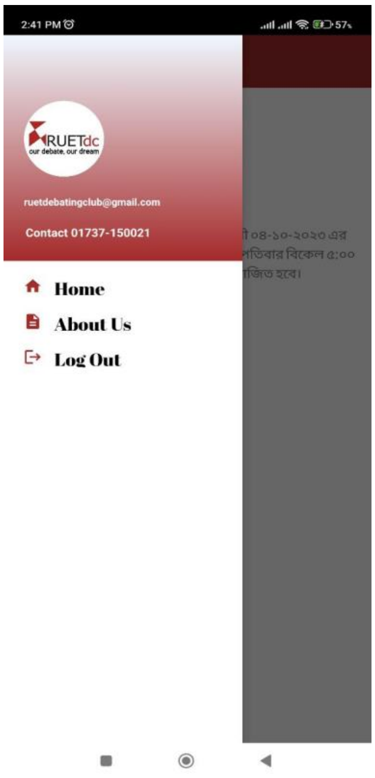
Name	Size	Type	Last modified
Notices by President/	—	Folder	—


Storage
Files
Rules
Usage
Extensions NEW
Protect your Storage resources from abuse, such as billing fraud or phishing
Configure App Check
Upload file

gs://club-management-8577f.appspot.com > Notices by Presi.

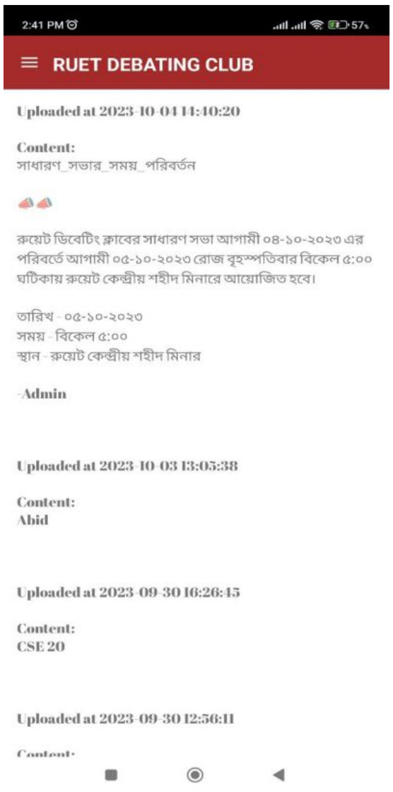
Name	Size	Type	Last modified
text_20230929_202101.txt	126 B	application/octet-stream	Sep 29, 2023
text_20230929_202149.txt	89 B	application/octet-stream	Sep 29, 2023
text_20230929_202239.txt	427 B	application/octet-stream	Sep 29, 2023
text_20230930_082814.txt	59 B	application/octet-stream	Sep 30, 2023
text_20230930_102153 AM.txt	25 B	application/octet-stream	Sep 30, 2023
text_20230930_125610.txt	26 B	application/octet-stream	Sep 30, 2023
text_20230930_162639.txt	6 B	application/octet-stream	Sep 30, 2023

6. nav_drawer.xml :



This interface is used for better UI experience. The Navigation Drawer opens by pressing Navigation Menu from both activity_admin_feed.xml and activity_friend.xml. By selecting “About Club”, the user would move to a new interface which has no functionality but briefing about club. Other than that, “Home” would move to the feed and “Log Out” would work for logging out for both user and admin and move to “Sign Up” activity. This interface is same for both user and admin.

7. activity_friend.xml (User Notice View Activity) :



Uploaded at 2023-10-04 14:40:20

Content:
সাধারণ_সভার_সময়_পরিবর্তন

রুয়েট ডিবেটিং ক্লাবের সাধারণ সভা আগামী ০৪-১০-২০২৩ এর পরিবর্তে আগামী ০৫-১০-২০২৩ রোজ বৃহস্পতিবার বিকেল ৫:০০ ঘটিকায় রুয়েট কেন্দ্রীয় শহীদ মিনারে আয়োজিত হবে।

তারিখ - ০৫-১০-২০২৩
সময় - বিকেল ৫:০০
স্থান - রুয়েট কেন্দ্রীয় শহীদ মিনার

-Admin

Uploaded at 2023-10-03 13:05:38

Content:
Abid

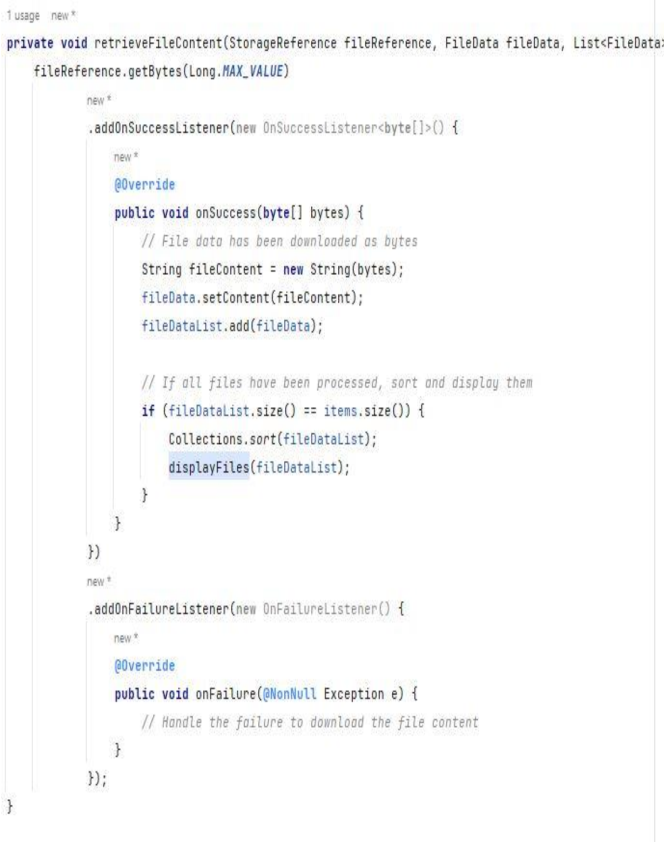
Uploaded at 2023-09-30 16:26:45

Content:
CSE 20

Uploaded at 2023-09-30 12:56:11

Content:

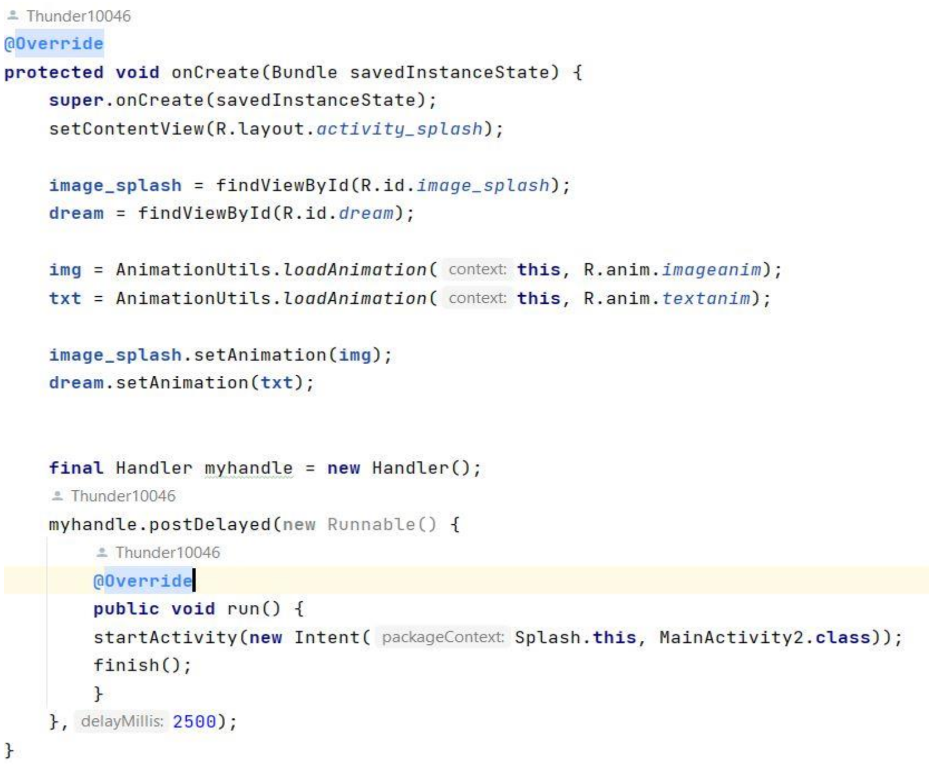
This is **the most pivotal part of the application**. For in this interface, the users can see the notices upload/updated by the admin. Also, the admin too can see by logging in as a member. Firebase Storage is used which is shown below:



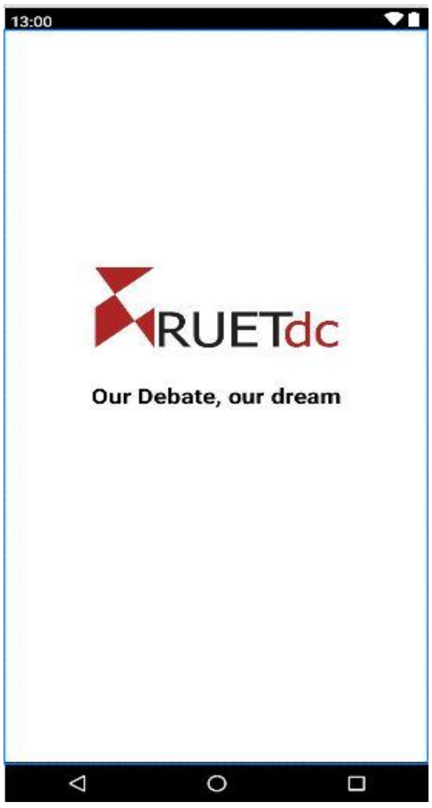
```java
1 usage   new *
private void retrieveFileContent(StorageReference fileReference, FileData fileData, List<FileData> fileDataList, List<StorageReference> items) {
    fileReference.getBytes(Long.MAX_VALUE)
            new *
            .addOnSuccessListener(new OnSuccessListener<byte[]>() {
                new *
                @Override
                public void onSuccess(byte[] bytes) {
                    // File data has been downloaded as bytes
                    String fileContent = new String(bytes);
                    fileData.setContent(fileContent);
                    fileDataList.add(fileData);

                    // If all files have been processed, sort and display them
                    if (fileDataList.size() == items.size()) {
                        Collections.sort(fileDataList);
                        displayFiles(fileDataList);
                    }
                }
            })
            new *
            .addOnFailureListener(new OnFailureListener() {
                new *
                @Override
                public void onFailure(@NonNull Exception e) {
                    // Handle the failure to download the file content
                }
            });
}
```

6. activity_splash.xml (Animation while launching) :

An animation is used to maintain better UI experience. This is the Launching activity. This is a simple xml file enabled functioning by using basic Java code, is shown below:



```
Thunder10046
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_splash);

    image_splash = findViewById(R.id.image_splash);
    dream = findViewById(R.id.dream);

    img = AnimationUtils.loadAnimation( context: this, R.anim.imageanim);
    txt = AnimationUtils.loadAnimation( context: this, R.anim.textanim);

    image_splash.setAnimation(img);
    dream.setAnimation(txt);


    final Handler myhandle = new Handler();
    Thunder10046
    myhandle.postDelayed(new Runnable() {
        Thunder10046
        @Override
        public void run() {
        startActivity(new Intent( packageContext: Splash.this, MainActivity2.class));
        finish();
        }
    }, delayMillis: 2500);
}
```



Conclusion :

This RUET DEBATING CLUB Management App serves various purposes, with a significant focus on effective time management. It helps users manage their time efficiently, allowing them to balance academics and other important tasks without distractions. One of the primary motivations behind developing this app was to streamline the registration process and keep club member information updated, eliminating the need for Excel sheets.

Also, by this app, University Projects can be handled by maintaining connection among Administrator and students. For this app is mainly built for updating notices by admin and retrieving notices by users. So, any kind of necessity of serving for such problems, this application could be a solution.

Future Improvements :

1. Messaging among members
2. Admin’s overview of all members
3. Individual profile layout