

## Chapter 4: IMPLEMENTATION

### 4.1 Model

#### AGILE MODEL

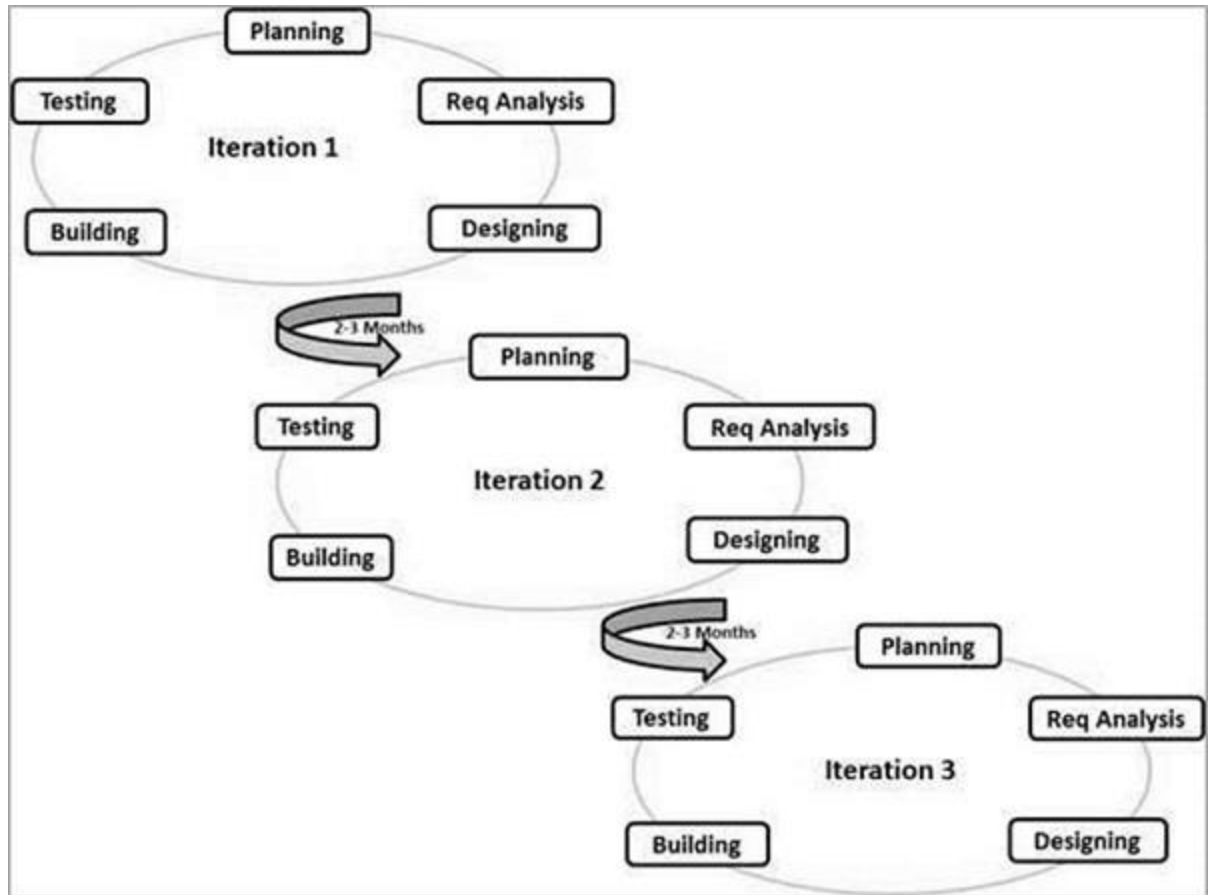
Agile is a software development methodology that is becoming more popular every day. It defines the mind set of many software development teams working across the globe.[2] This article will walk you through the entire Agile-scrum process and how, as a developer, you can contribute in an agile way and deliver value. BTW, Agile means ability to move quickly. [3]

#### Agenda

- Life Cycle of a Developer
- Barriers to value delivery
- Agile Adoption
- Plan Driven Vs. Value Driven
- Agile Manifesto
- Scrum End-to-End Process
- Agile Estimation
- Daily Standup
- Agile-Scrum Tools

#### Life Cycle of a Developer

The 1<sup>st</sup> interaction a developer has is with a BA. They are responsible for documenting, tracking and describing the user requirements. But there are often gaps in understanding therequirements and that can cause issues in various ways.The following image shows what the user wanted and what happened when it was implemented. [4]

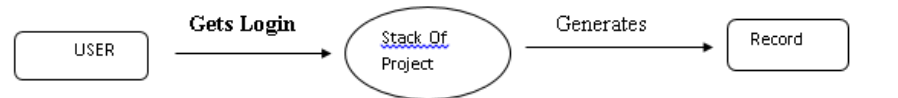


**Figure 4.1.1 agile software development model**

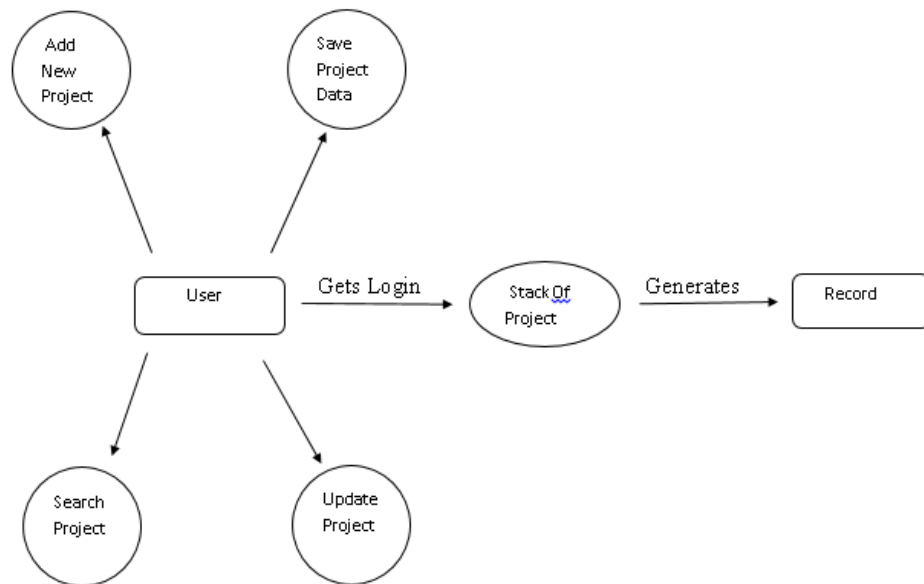
Predictive methods entirely depend on the requirement analysis and planning done in the beginning of cycle. Any changes to be incorporated go through a strict change control management and prioritization. Agile uses an adaptive approach where there is no detailed planning and there is clarity on future tasks only in respect of what features need to be developed. There is feature driven development and the team adapts to the changing product requirements dynamically. The product is tested very frequently, through the release iterations, minimizing the risk of any major failures in future. [3] Customer Interaction is the backbone of this agile methodology, and open communication with minimum documentation are the typical features of Agile development environment. The agile teams work in close collaboration with each other and are most often located in the same geographical location. [7]

## 4.2 Data Flow Diagram

### 4.2.1 DATA FLOW DIAGRAM



### 4.2.2 Level-1 DFD Of Stack Of Project



**Figure 4.2.1 first level dfd**

This level one DFD explains about the feature when user registers on our website. Users can upload their own project when logged in. Users can search and add new projects on our app. When user adds new project admin approval is required.

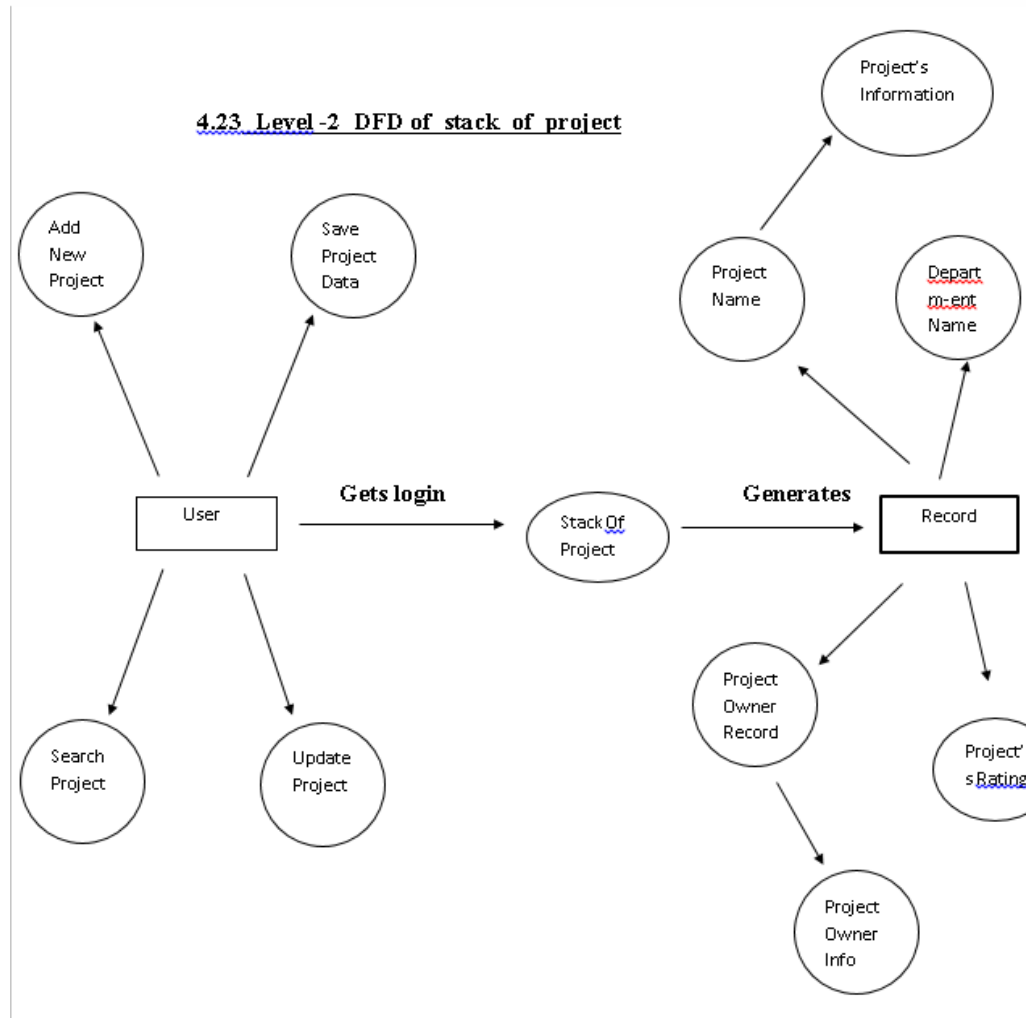


Figure 4.2.2 second level dfd

This level two DFD explains detailed procedure of how users enters their own project on our app. Users have to give their project basic information first to add their own project on our app. Users are asked for project name, its date of creation, little description about the project.

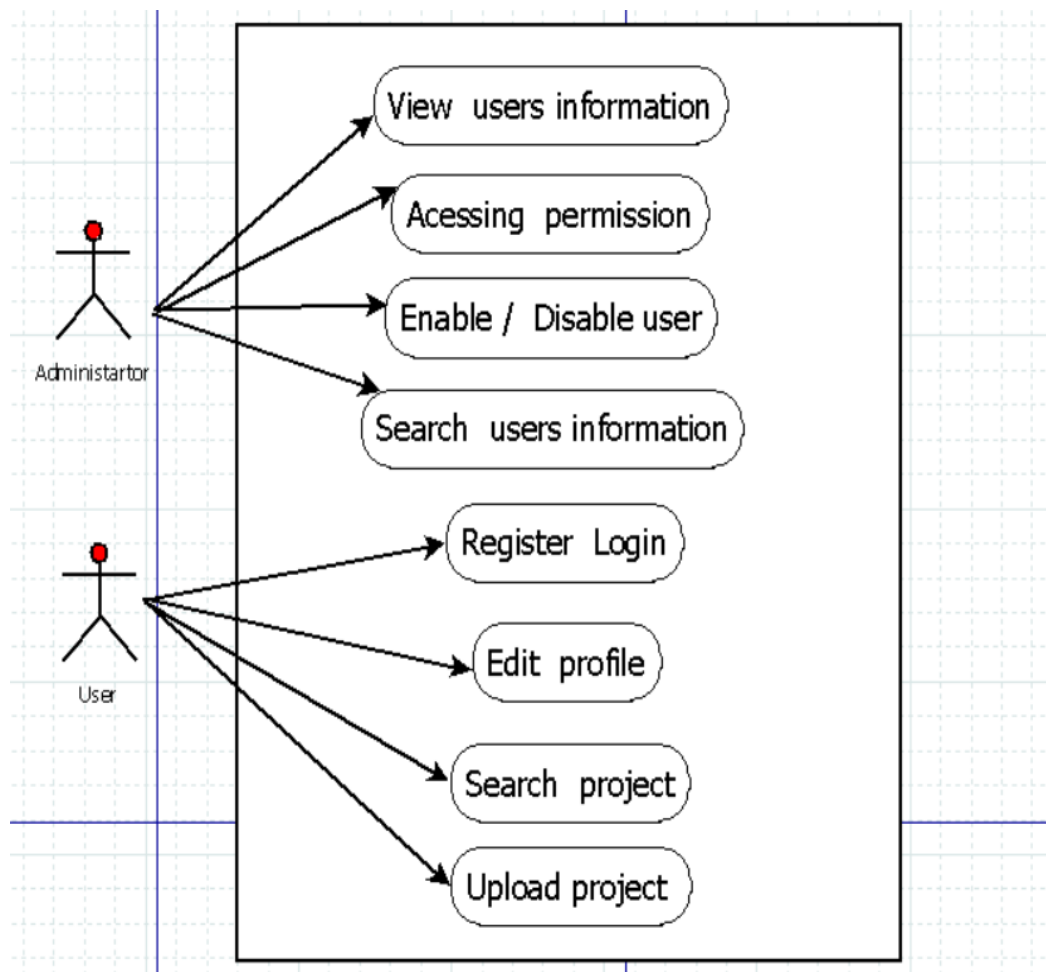


Figure 4.2.3 use case diagram

This use case diagram explains about the administrator functionalities and its benefits. It also gives information about the users who are registered on the app, When administrator accesses the content of app administrator has to login on the website. Administrator can enable and disable users and also projects hosted on the website. Users can search the projects and add their own projects on the app this use case explains about how users can do this.

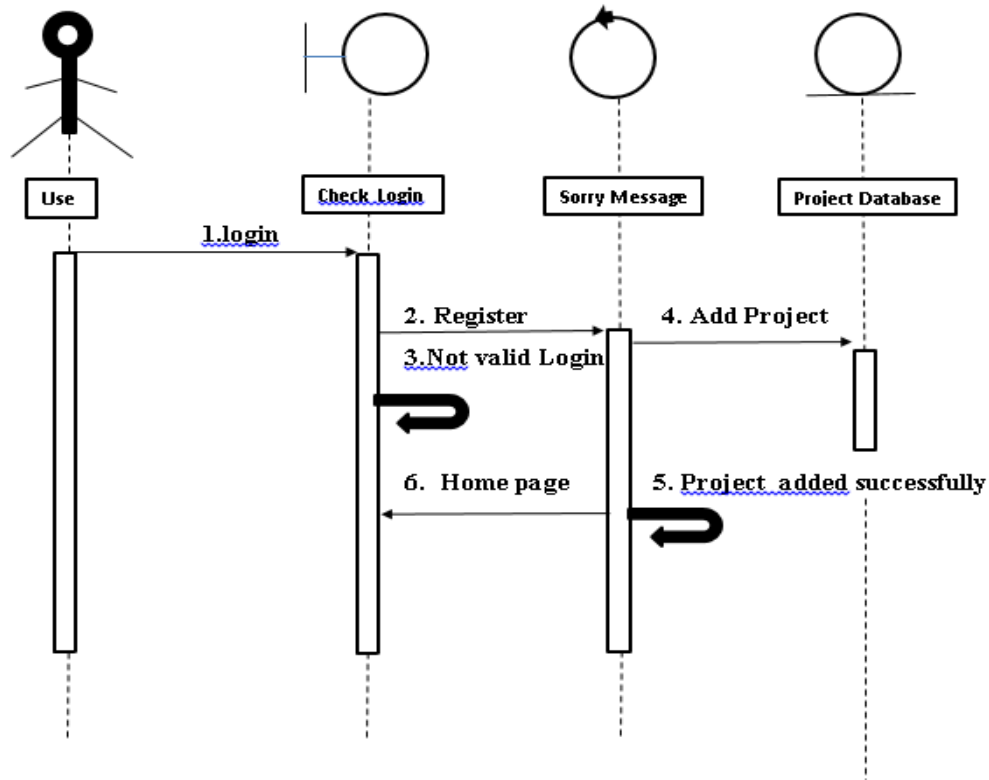
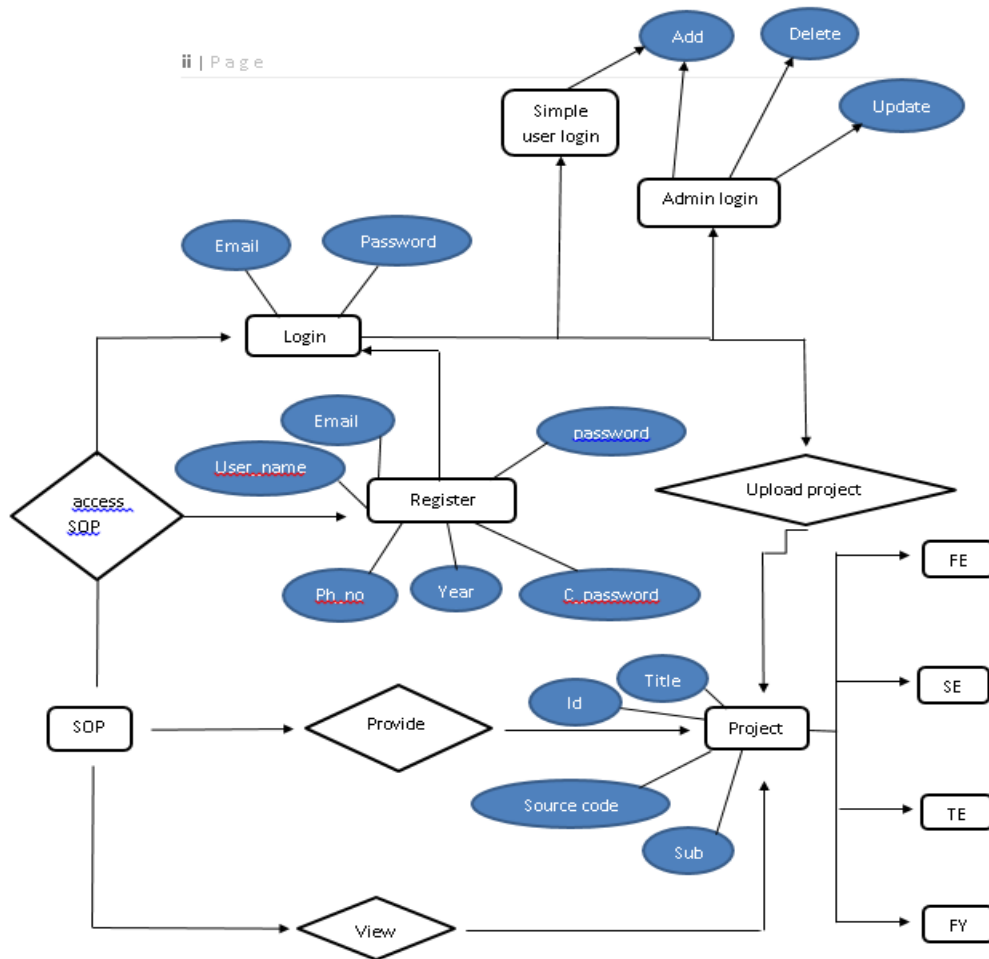
4.4 1. SEQUENCE DIGRAM

Figure 4.2.4 sequence diagram for login

This sequence diagram explains about how user logs in on our website. First users have to register, then they are asked for email id and password. When login is not valid then it sends a sorry message box on the interface and if it succeeds then it shows project interface.

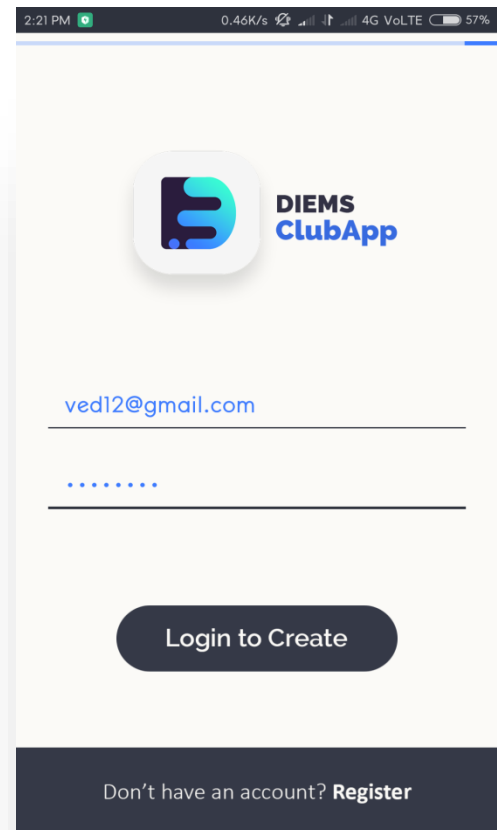
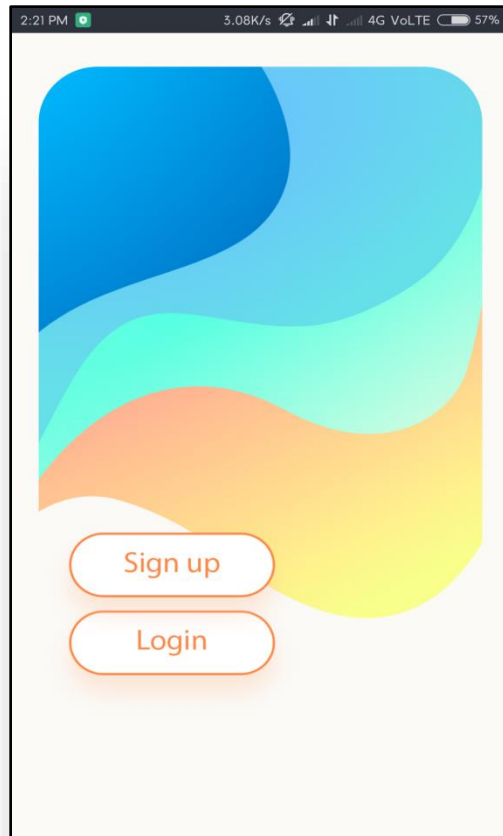
### 4.3 ER DIAGRAM



**Figure 4.3.1 ER diagram for app access**

This ER diagram explains about the how users and administrator login actually works on our app. When a person tries to log in our app, Administrator is asked for email and password. If it is admin login then administrator is provided with additional functionalities such as delete, update project and for normal user administrator is given only add new project functionality.

## 4.4 USER INTERFACE DESIGN



screenshot 4.4.1 user interface design-1

This page includes the login details of user. If user already registered then user login directly ,And if not then user have to register first by clicking on the create new account button.



**LoginActivity.java**

```
package com.clubapp.aniruddhadeshpande.dev;

import android.content.Intent;

import android.media.Image;

import android.support.annotation.NonNull;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.text.TextUtils;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.ImageView;

import android.widget.ProgressBar;

import android.widget.Toast;

import com.google.android.gms.tasks.OnCompleteListener;

import com.google.android.gms.tasks.Task;

import com.google.firebase.auth.AuthResult;

import com.google.firebase.auth.FirebaseAuth;

import com.google.firebase.auth.FirebaseUser;

public class LoginActivity extends AppCompatActivity {

    private EditText loginEmailText;

    private EditText loginPassText;

    private ImageView loginBtn;
```

```

private ImageView loginRegBtn;

private FirebaseAuth mAuth;

private ProgressBar loginProgress;

@Override

protected void onCreate(Bundle savedInstanceState) {

    super.onCreate(savedInstanceState);

    setContentView(R.layout.activity_login);

    mAuth = FirebaseAuth.getInstance();

    loginEmailText = findViewById(R.id.reg_email);

    loginPassText = findViewById(R.id.reg_confirm_pass);

    loginBtn = findViewById(R.id.login_btn);

    loginRegBtn = findViewById(R.id.login_reg_btn);

    loginProgress = findViewById(R.id.login_progress);

    loginRegBtn.setOnClickListener(new View.OnClickListener() {

        @Override

        public void onClick(View v) {

            Intent regIntent = new Intent(LoginActivity.this, RegisterActivity.class);

            startActivity(regIntent);

        }

    });

    loginBtn.setOnClickListener(new View.OnClickListener() {

        @Override

        public void onClick(View v) {

```

```

String loginEmail = loginEmailText.getText().toString();

String loginPass = loginPassText.getText().toString();

if(!TextUtils.isEmpty(loginEmail) && !TextUtils.isEmpty(loginPass)){

    loginProgress.setVisibility(View.VISIBLE);

    mAuth.signInWithEmailAndPassword(loginEmail,
loginPass).addOnCompleteListener(new OnCompleteListener<AuthResult>() {

        @Override

        public void onComplete(@NonNull Task<AuthResult> task) {

            if(task.isSuccessful()){

                sendToMain();

            } else {

                String errorMessage = task.getException().getMessage();

                Toast.makeText(LoginActivity.this, "Error : " + errorMessage,
Toast.LENGTH_LONG).show();

            }

            loginProgress.setVisibility(View.INVISIBLE);

        }

    });

}

else{

    if(TextUtils.isEmpty(loginEmail)){

        loginEmailText.setError("Please Enter Your Email");

        loginEmailText.requestFocus();

```

```

        return;
    }

    if(TextUtils.isEmpty(loginPass)){

        loginPassText.setError("Please Enter Your Email");

        loginPassText.requestFocus();

        return;
    }
}

});
}

@Override

protected void onStart() {

    super.onStart();

    FirebaseUser currentUser = mAuth.getCurrentUser();

    if(currentUser != null){

        sendToMain();

    }

}

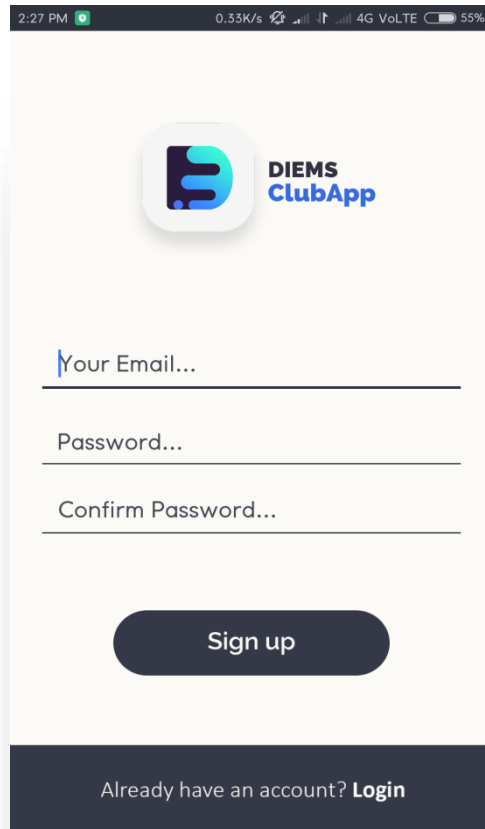
private void sendToMain() {

    Intent mainIntent = new Intent(LoginActivity.this, MainActivity.class);

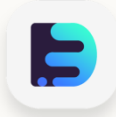
    startActivity(mainIntent);

    finish() } }

```

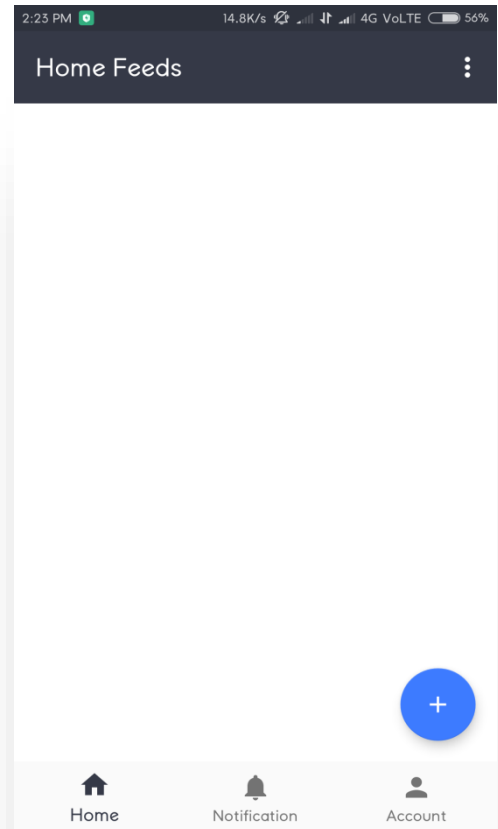


2:27 PM 0.53K/s 4G VoLTE 55%

 **DIEMS  
ClubApp**

**Sign up**

Already have an account? **Login**



**screenshot 4.4.2 user interface design-2**

This page includes the registration details of user. If user is not registered yet then user Can register by using this registration page by filling the required given fields.

**RegisterActivity.java**

```
package com.clubapp.aniruddhadeshpande.dev;

import android.content.Intent;

import android.os.Bundle;

import android.support.annotation.NonNull;

import android.support.v7.app.AppCompatActivity;

import android.text.TextUtils;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.ImageView;

import android.widget.ProgressBar;

import android.widget.Toast;

import com.google.android.gms.tasks.OnCompleteListener;

import com.google.android.gms.tasks.Task;

import com.google.firebase.auth.AuthResult;

import com.google.firebase.auth.FirebaseAuth;

import com.google.firebase.auth.FirebaseUser;

public class RegisterActivity extends AppCompatActivity {

    private EditText reg_email_field;

    private EditText reg_pass_field;

    private EditText reg_confirm_pass_field;
```

```

private ImageView reg_btn;

private ImageView reg_login_btn;

private ProgressBar reg_progress;

private FirebaseAuth mAuth;

@Override

protected void onCreate(Bundle savedInstanceState) {

    super.onCreate(savedInstanceState);

    setContentView(R.layout.activity_register);

    mAuth = FirebaseAuth.getInstance();

    reg_email_field = findViewById(R.id.reg_email);

    reg_pass_field = findViewById(R.id.reg_pass);

    reg_confirm_pass_field = findViewById(R.id.reg_confirm_pass);

    reg_btn = findViewById(R.id.reg_btn);

    reg_login_btn = findViewById(R.id.reg_login_btn);

    reg_progress = findViewById(R.id.reg_progress);

    reg_login_btn.setOnClickListener(new View.OnClickListener() {

        @Override

        public void onClick(View v) {

            finish();

        }

    });

    reg_btn.setOnClickListener(new View.OnClickListener() {

        @Override

```

```

public void onClick(View v) {

    String email = reg_email_field.getText().toString();

    String pass = reg_pass_field.getText().toString();

    String confirm_pass = reg_confirm_pass_field.getText().toString();

    if(!TextUtils.isEmpty(email) && !TextUtils.isEmpty(pass) &
!TextUtils.isEmpty(confirm_pass)){

        if(pass.equals(confirm_pass)){

            reg_progress.setVisibility(View.VISIBLE);

            mAuth.createUserWithEmailAndPassword(email,
pass).addOnCompleteListener(new OnCompleteListener<AuthResult>() {

                @Override

                public void onComplete(@NonNull Task<AuthResult> task) {

                    if(task.isSuccessful()){

                        Intent setupIntent = new Intent(RegisterActivity.this, SetupActivity.class);

                        startActivity(setupIntent);

                        finish();

                    } else {

                        String errorMessage = task.getException().getMessage();

                        Toast.makeText(RegisterActivity.this, "Error : " + errorMessage,
Toast.LENGTH_LONG).show();

                    }

                    reg_progress.setVisibility(View.INVISIBLE);

                }

            }

```



```

        });

    } else {

        Toast.makeText(RegisterActivity.this, "Confirm Password and Password Field
doesn't match.", Toast.LENGTH_LONG).show();

    }

}

else{

    if(TextUtils.isEmpty(email)){

        reg_email_field.setError("Please Enter Your Email");

        reg_email_field.requestFocus();

        return;

    }

    if(TextUtils.isEmpty(pass)){

        reg_pass_field.setError("Please Enter Your password");

        reg_pass_field.requestFocus();

        return;

    }

    if(TextUtils.isEmpty(confirm_pass)){

        reg_confirm_pass_field.setError("Please Enter Your confirm password");

        reg_confirm_pass_field.requestFocus();

        return;

    }

}

```

```
        }

    });

}

@Override

protected void onStart() {

    super.onStart();

    FirebaseUser currentUser = mAuth.getCurrentUser();

    if(currentUser != null){

        sendToMain();

    }

}

private void sendToMain() {

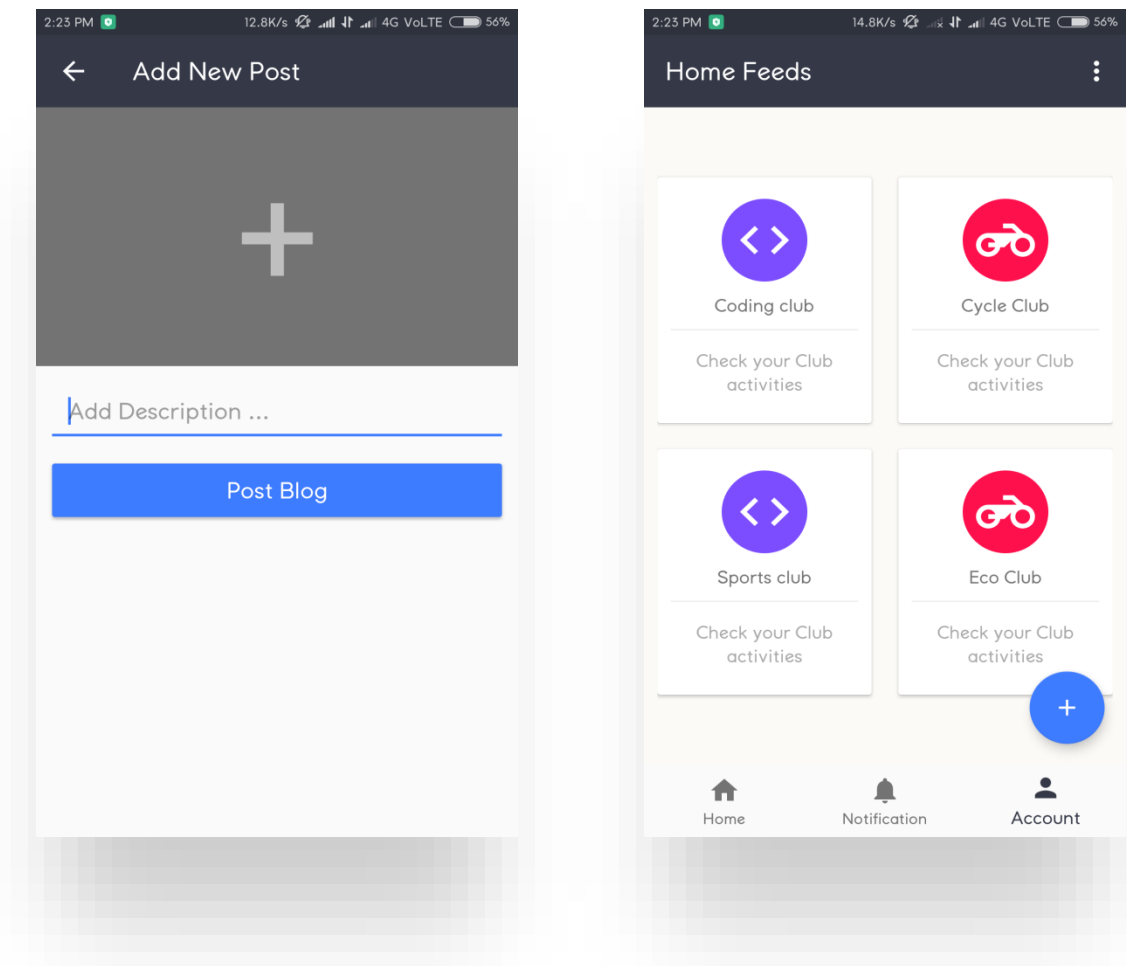
    Intent mainIntent = new Intent(RegisterActivity.this, MainActivity.class);

    startActivity(mainIntent);

    finish();

}

}
```



**screenshot 4.4.3 user interface design-3**

This page includes the post details given by user. If user is satisfied with application or not he can send feedback about that.

**NewPostActiivty.java**

```
package com.clubapp.aniruddhadeshpande.dev;

import android.content.Intent;

import android.graphics.Bitmap;

import android.net.Uri;

import android.os.AsyncTask;

import android.os.Bundle;

import android.os.StrictMode;

import android.support.annotation.NonNull;

import android.support.v7.app.AppCompatActivity;

import android.support.v7.widget.Toolbar;

import android.text.TextUtils;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.ImageView;

import android.widget.ProgressBar;

import android.widget.Toast;

import com.google.android.gms.tasks.OnCompleteListener;

import com.google.android.gms.tasks.OnFailureListener;

import com.google.android.gms.tasks.OnSuccessListener;

import com.google.android.gms.tasks.Task;

import com.google.firebase.auth.FirebaseAuth;
```

```

import com.google.firebase.firestore.DocumentReference;

import com.google.firebase.firestore.FieldValue;

import com.google.firebase.firestore.FirebaseFirestore;

import com.google.firebase.storage.FirebaseStorage;

import com.google.firebase.storage.StorageReference;

import com.google.firebase.storage.UploadTask;

import com.theartofdev.edmodo.cropper.CropImage;

import com.theartofdev.edmodo.cropper.CropImageView;

import java.io.ByteArrayOutputStream;

import java.io.File;

import java.io.IOException;

import java.io.OutputStream;

import java.net.HttpURLConnection;

import java.net.URL;

import java.util.HashMap;

import java.util.Map;

import java.util.Scanner;

import java.util.UUID;

import id.zelory.compressor.Compressor;

public class NewPostActivity extends AppCompatActivity {

    private Toolbar newPostToolbar;

    private ImageView newPostImage;

    private EditText newPostDesc;

```

```

private Button newPostBtn;

private Uri postImageUri = null;

private ProgressBar newPostProgress;

private StorageReference storageReference;

private FirebaseFirestore firebaseFirestore;

private FirebaseAuth firebaseAuth;

ListUsersPage page = FirebaseAuth.getInstance().listUsersAsync(null).get();

private String current_user_id;

private Bitmap compressedImageFile;

@Override

protected void onCreate(Bundle savedInstanceState) {

    super.onCreate(savedInstanceState);

    setContentView(R.layout.activity_new_post);

    storageReference = FirebaseStorage.getInstance().getReference();

    firebaseFirestore = FirebaseFirestore.getInstance();

    firebaseAuth = FirebaseAuth.getInstance();

    current_user_id = firebaseAuth.getCurrentUser().getUid();

    newPostToolbar = findViewById(R.id.new_post_toolbar);

    setSupportActionBar(newPostToolbar);

    getSupportActionBar().setTitle("Add New Post");

    getSupportActionBar().setDisplayHomeAsUpEnabled(true);

    newPostImage = findViewById(R.id.new_post_image);

    newPostDesc = findViewById(R.id.new_post_desc);

```

```

newPostBtn = findViewById(R.id.post_btn);

newPostProgress = findViewById(R.id.new_post_progress);

newPostImage.setOnClickListener(new View.OnClickListener() {

    @Override

    public void onClick(View v) {

        CropImage.activity()

            .setGuidelines(CropImageView.Guidelines.ON)

            .setMinCropResultSize(512, 512)

            .setAspectRatio(1, 1)

            .start(NewPostActivity.this);

    }

});

newPostBtn.setOnClickListener(new View.OnClickListener() {

    @Override

    public void onClick(View v) {

        final String desc = newPostDesc.getText().toString();

        if(!TextUtils.isEmpty(desc) && postImageUri != null){

            newPostProgress.setVisibility(View.VISIBLE);

            final String randomName = UUID.randomUUID().toString();

            // PHOTO UPLOAD

            File newImageFile = new File(postImageUri.getPath());

            try {

                compressedImageFile = new Compressor(NewPostActivity.this)

```

```

        .setMaxHeight(720)

        .setMaxWidth(720)

        .setQuality(50)

        .compressToBitmap(new ImageFile);

    } catch (IOException e) {

        e.printStackTrace();

    }

    ByteArrayOutputStream baos = new ByteArrayOutputStream();

    compressedImageFile.compress(Bitmap.CompressFormat.JPEG, 100, baos);

    byte[] imageData = baos.toByteArray();

    // PHOTO UPLOAD

    UploadTask filePath = storageReference.child("post_images").child(randomName +
    ".jpg").putBytes(imageData);

    filePath.addOnCompleteListener(new
    OnCompleteListener<UploadTask.TaskSnapshot>() {

        @Override

        public void onComplete(@NonNull final Task<UploadTask.TaskSnapshot> task)

        {

            final String downloadUri = task.getResult().getDownloadUrl().toString();

            if(task.isSuccessful()){

                File newThumbFile = new File(postImageUri.getPath());

                try {

                    compressedImageFile = new Compressor(NewPostActivity.this)

```



```

        .setMaxHeight(100)

        .setMaxWidth(100)

        .setQuality(1)

        .compressToBitmap(newThumbFile);
    } catch (IOException e) {

        e.printStackTrace();
    }

    ByteArrayOutputStream baos = new ByteArrayOutputStream();
    compressedImageFile.compress(Bitmap.CompressFormat.JPEG, 100, baos);
    byte[] thumbData = baos.toByteArray();

    UploadTask uploadTask = storageReference.child("post_images/thumbs")

        .child(randomName + ".jpg").putBytes(thumbData);

    uploadTask.addOnSuccessListener(new
    OnSuccessListener<UploadTask.TaskSnapshot>() {

        @Override

        public void onSuccess(UploadTask.TaskSnapshot taskSnapshot) {

            String downloadthumbUri = taskSnapshot.getDownloadUrl().toString();

            Map<String, Object> postMap = new HashMap<>();

            postMap.put("image_url", downloadUri);

            postMap.put("image_thumb", downloadthumbUri);

            postMap.put("desc", desc);

            postMap.put("user_id", current_user_id);

            postMap.put("timestamp", FieldValue.serverTimestamp());

```

```

firebaseFirestore.collection("Posts").add(postMap).addOnCompleteListener(new
OnCompleteListener<DocumentReference>() {

    @Override

    public void onComplete(@NonNull Task<DocumentReference>
task) {

        if(task.isSuccessful()){

            sendNotification();

            Toast.makeText(NewPostActivity.this, "Post was added",
Toast.LENGTH_LONG).show();

            Intent mainIntent = new Intent(NewPostActivity.this,
MainActivity.class);

            startActivity(mainIntent);

            finish();

        } else {

        }

        newPostProgress.setVisibility(View.INVISIBLE);

    }

});

}

}).addOnFailureListener(new OnFailureListener() {

    @Override

    public void onFailure(@NonNull Exception e) {

        //Error handling

```

```

        }

        });

    } else {

        newPostProgress.setVisibility(View.INVISIBLE);

    }

}

});

}

}

});

}

@Override

protected void onActivityResult(int requestCode, int resultCode, Intent data) {

    super.onActivityResult(requestCode, resultCode, data);

    if (requestCode == CropImage.CROP_IMAGE_ACTIVITY_REQUEST_CODE) {

        CropImage.ActivityResult result = CropImage.getActivityResult(data);

        if (resultCode == RESULT_OK) {

            postImageUri = result.getUri();

            newPostImage.setImageURI(postImageUri);

        } else if (resultCode ==

CropImage.CROP_IMAGE_ACTIVITY_RESULT_ERROR_CODE) {

            Exception error = result.getError();

        }

```

```

    }
}

private void sendNotification(){

    for()

    AsyncTask.execute(new Runnable() {

        @Override

        public void run() {

            int SDK_INT = android.os.Build.VERSION.SDK_INT;

            if (SDK_INT > 8) {

                StrictMode.ThreadPolicy policy = new StrictMode.ThreadPolicy.Builder()

                    .permitAll().build();

                StrictMode.setThreadPolicy(policy);

                String send_email;

                //This is a Simple Logic to Send Notification different Device Programmatically....

                if (MainActivity.LoggedIn_User_Email.equals("ved12@gmail.com") ) {

                    send_email = "user2@gmail.com";

                } else {

                    send_email = "user1@gmail.com";

                }

                try {

                    String jsonResponse;

                    URL url = new URL("https://onesignal.com/api/v1/notifications");

                    HttpURLConnection con = (HttpURLConnection) url.openConnection();

```

```

con.setUseCaches(false);

con.setDoOutput(true);

con.setDoInput(true);

con.setRequestProperty("Content-Type", "application/json; charset=UTF-8");

con.setRequestProperty("Authorization", "Basic
YTUzMWE2M2UtMTZiYy00M2FhLWEyMjltYWQ5YWI1MDgzM2U2");

con.setRequestMethod("POST");

String strJsonBody = "{"
    + "\"app_id\": \"0523d5af-d75a-4916-a8dd-3e9109e0f10b\","
    + "\"filters\": [{\"field\": \"tag\", \"key\": \"User_ID\", \"relation\": \"=\",
    + \"value\": \"\" + send_email + \"\"}],\"
    + \"data\": {\"foo\": \"bar\"},\"
    + \"contents\": {\"en\": \"English Message\"}\"
    + "}";

System.out.println("strJsonBody:\n" + strJsonBody);

byte[] sendBytes = strJsonBody.getBytes("UTF-8");

con.setFixedLengthStreamingMode(sendBytes.length);

OutputStream outputStream = con.getOutputStream();

outputStream.write(sendBytes);

int httpResponse = con.getResponseCode();

System.out.println("httpResponse: " + httpResponse);

if (httpResponse >= HttpURLConnection.HTTP_OK
    && httpResponse < HttpURLConnection.HTTP_BAD_REQUEST) {

```

```

        Scanner scanner = new Scanner(con.getInputStream(), "UTF-8");

        jsonResponse = scanner.useDelimiter("\\A").hasNext() ? scanner.next() : "";

        scanner.close();

    } else {

        Scanner scanner = new Scanner(con.getErrorStream(), "UTF-8");

        jsonResponse = scanner.useDelimiter("\\A").hasNext() ? scanner.next() : "";

        scanner.close();

    }

    System.out.println("jsonResponse:\n" + jsonResponse);

} catch (Throwable t) {

    t.printStackTrace();

}

}

});

}

}

```