
# CITIZEN COP

## A CLOUD BASED CRIME REPORT APPLICATION.

**PRESENTED BY :-**

SOUVIK ROY - IT2014/066

AWSAF AMBAR - IT2014/067

SAYANI DUTTA - IT2014/089

# CITIZEN COP

## A CLOUD BASED CRIME REPORTING APP

REPORT OF PROJECT SUBMITTED FOR PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE DEGREE OF BACHELOR OF TECHNOLOGY In INFORMATION TECHNOLOGY

By

SOUVIK ROY

REGISTRATION NO – 141170110165

UNIVERSITY ROLL NO –11700214070

AWSAF AMBAR

REGISTRATION NO-141170110120

UNIVERSITY ROLL NO-11700214025

SAYANI DUTTA

REGISTRATION NO-141170110157

UNIVERSITY ROLL NO -11700214062

UNDER THE SUPERVISION OF DR ABHIJIT DAS

DEPARTMENT OF INFORMATION TECHNOLOGY


AT

RCC INSTITUTE OF INFORMATION TECHNOLOGY

[Affiliated to Maulana Abul kalam Azad University of Technology]

CANAL SOUTH ROAD, BELIAGHATA, KOLKATA – 700 015


# CERTIFICATE

The report of the Project titled **CITIZEN COP** submitted by Souvik Roy, Awsaf Ambar, Sayani Dutta of B. Tech. (IT) 8th Semester of 2018 has been prepared under our supervision for the partial fulfilment of the requirements for B Tech (IT) degree in Maulana Abul Kalam Azad University of Technology. The report is hereby forwarded.

---

SIGNATURE OF THE GUIDE

ASSOCIATE PROF. ABHIJIT DAS

HEAD OF THE DEPARTMENT

DEPARTMENT OF INFORMATION TECHNOLOGY

RCCIIT, Kolkata

---

SIGNATURE OF THE HEAD

ASSOCIATE PROF.ABHIJIT DAS

HEAD OF THE DEPARTMENT

DEPARTMENT OF INFORMATION TECHNOLOGY

RCCIIT, KOLKATA

# ACKNOWLEDGEMENT

We would like to express our sincere gratitude to Dr Abhijit Das, Associate Professor and the Head of the Department of the department of Information Technology, whose role as project guide was invaluable for the project. We are extremely thankful for the keen interest he took in advising us, for the books and reference materials provided for the moral support extended to us.

Last but not the least we convey our gratitude to all the teachers for providing us the technical skill that will always remain as our asset and to all non-teaching staff for the gracious hospitality they offered us.

Place: RCCIIT, Kolkata

Date: 11.5.18

_____

SOUVIK ROY(IT2014/066)

_____

AWSAF AMBAR(IT 2014/067)

_____

SAYANI DUTTA(IT2014/089)

# CONTENTS

## TABLE INDEX:



## FIGURE INDEX:

# INTRODUCTION

## 1.1 PURPOSE & MOTIVATION

There are numerous reasons to go for this project, but few prime reasons are as follows: -

- Cloud is an advance technology and trending also, lots of projects involved websites, apps, services for remote public is done using cloud. So, to get a taste of that technology it is important to work on it.
- We as a group always interested to go and develop for an android app rather than any other website designing as, we want to continue to be app developer. To be a good app developer it is important to work closely with integrating API'S and other technology keeping trending demands.
- After cloud + android app, we need an app to be a bit different with basic functionality that will explore various cloud features, so we thought of a crime blog app so that after proper working we can place it over Play Store and people can be benefitted.

## 1.2 SCOPE

Android is an open-source Linux-based operating system designed mainly for smart phones and tablets. It is maintained as an open source project by Google. This open source code and licensing allows the developers and device manufacturers to modify the software according to their needs. Android platform has brought about cutting-edge technologies in app development.
Owing to the popularity of Android, Mobile Apps development industries are considering Android Application Development as one of the best remunerative business opportunities. The need to hire knowledgeable mobile application developer is intense.

Firebase projects are Google Cloud Platform projects that use Firebase services. This means that:

- Billing and permissions for projects are shared across consoles.
- Projects that appear in the Firebase console also appear in the Google Cloud Platform and Google APIs consoles.
- Deleting a project deletes it across all consoles. Projects let you share users, data, and analytics across platforms (Android, iOS, and web) so your users have the same experience whatever device they're on. Every app using Firebase is connected to a single Firebase project, so you can manage all versions of your app through the Firebase console.

Recalling Eric Schmidt's statement – "Mobile is the future of Software Development"- Android is on the path of proving the same.

## 1.3 DEFINITIONS, ACRONYMS, AND ABBREVIATIONS

Table 1 - Definitions

| **Term** | **Definition** |
|---|---|
| User | Someone who interacts with the mobile phone application |
| Admin/Administrator | System administrator who is given specific permission for managing and controlling the system |
| Application Store | An installed application on mobile phone which helps user to find new compatible applications with mobile phone platform and download them from Internet |
| Stakeholder | Any person who interacts with the system who is not a developer |
| APP | An android app with setup file as .APK and can be installed in any android phones |
| UI | the user interface (UI) is everything designed into an information device with which a person may interact. This can include display screens, keyboards, a mouse and the appearance of a desktop. It is also the way through which a user interacts with an application or a website |
| | |
| CLOUD | Simply put, cloud computing is the delivery of computing services—servers, storage, databases, networking, software, analytics and more—over the Internet **("the cloud")** [1] |
| PaaS | Platform as a service (PaaS) is a complete development and deployment environment in the cloud, with resources that enable you to deliver everything from simple cloud-based apps to sophisticated, cloud-enabled enterprise applications [2] |
| Robo test | Robo test analyses the structure of your app's UI and then explores it methodically, automatically simulating user activities [3] |
| Unit Test | Unit tests are the fundamental tests in your app testing strategy. By creating and running unit tests against your code. |

# PROJECT OVERVIEW

## 2.1 PROBLEM STATEMENT

As of today, the reporting system in the country is must be reported personally. The concern citizen must go to the nearest police station to report a crime or incident or the person need to call the nearest police station for faster action of the authority. As a result, the possibility of neglecting of reporting the crime by the concern citizen is enormous.

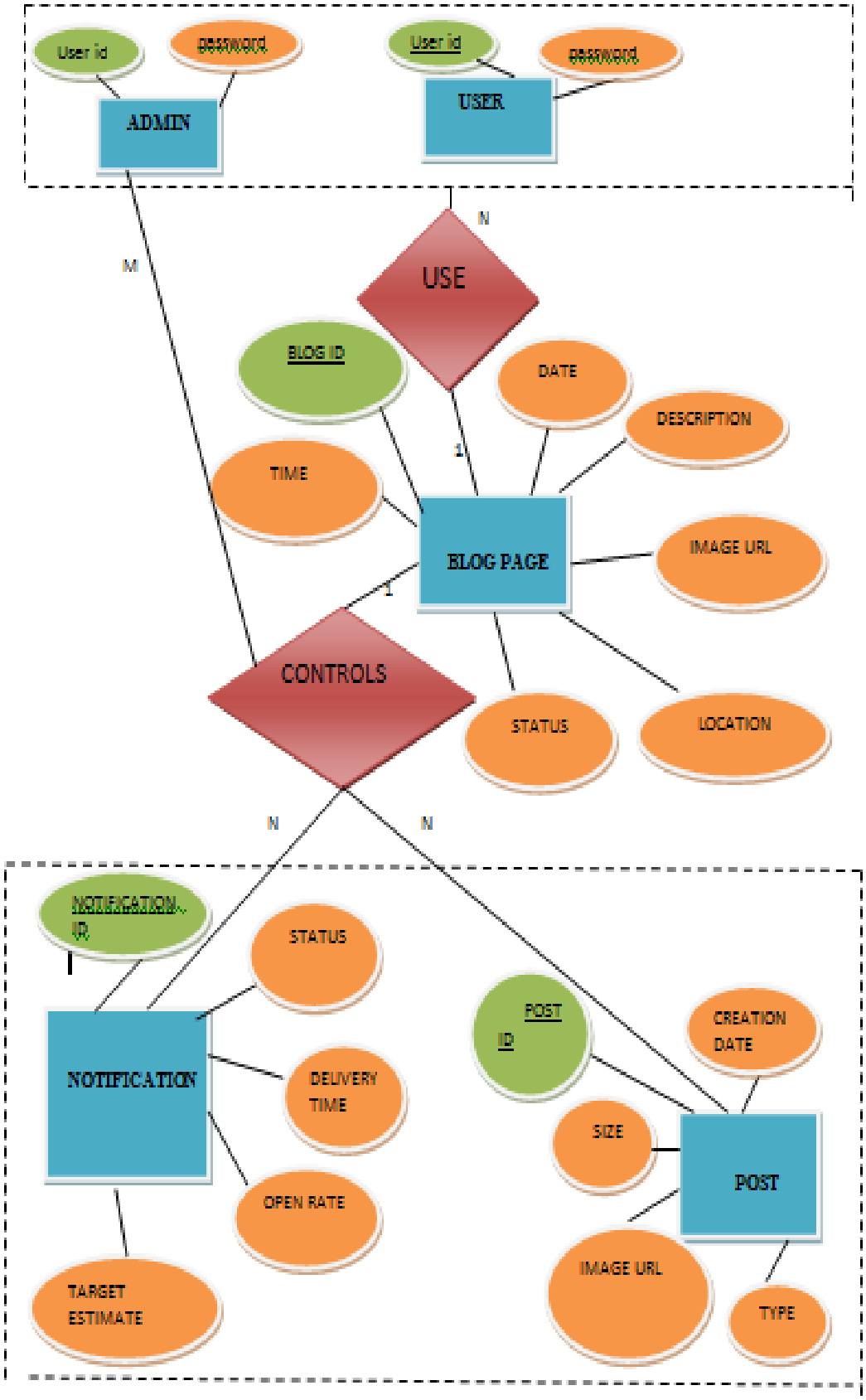
The citizen neglect to report incidents to less hassle and because of unknowledgeable in law. The authorities also lack in spreading the police station hotline in communities, most of the people in the community don't know the nearest police station hotline. The police station must have cell phone number hotline for the complainant that is only using promo loads. Some of complainants are not familiar with the place and didn't know where the police station is. It is considered the citizens that are trap in hazardous place like floods and landslides.

It also considered the most of the people in the community doubt in reporting especially if the person that will be complained is a big syndicate, life threatening situation leads the citizen to silence. The long record of misbehave police also affect the trust of the citizen to the authority. The inadequate number of the authority is also considered and it’s the causes why the response of the authority takes longer.

## 2.2 Our solution

Instead of focusing only on the various technological possibilities of computers such as report by email in fear of revealing the ID we have gone for android app which is legal and with security to report crime irrespective of the age, gender or any other discrimination. It must be blogged so that the people who witnessed may get some courage to report or add comment in favour of such crime. DFD’s will explain the working of our solution.

- Our cloud platform which consists of storage, database, authentication, testing platform and many more




**ENTITY RELATIONSHIP DIAGRAM**

Fig.1

## 2.3 CITIZEN COP APPLICATION

With the help of Google firebase which is a cloud platform[6] we build an application which aims to report the nearby crime.

- First when we open our application a user interface of registration page is opened.
- In the registration page there is a link to link to the login page if the user is already a member.
- There is also an option of “ADMIN” section in the menu.
- When user click on register after entering the email id, after successful checks if it is, that user will be redirected to login page and continue.
- When he enter its own correct login id and password it will redirected to stories page.
- Stories page consist of all the crime which was reported by people.
- If user once logged in, it is not required to login again as directly “STORIES” page will be open.
- From there, there will be options to add post and logout.
- If one logout its instance get cleared and so directed to login page.
- If one click on “circular add post icon” on below it will be redirected to a new page.
- This page will have options to report a crime by attaching details like title, date, time, location, description, image.
- After entering details it will be redirected to again “stories” page with the post which is recently added on to the top.
- Admin have their own login section from which on successful login redirected to again “stories” page.
- They have the right to change the status of the post as by default a post was submitted with not verified.
- Admin can also delete post also.

**FOLLOWING DFDS WILL HELP**

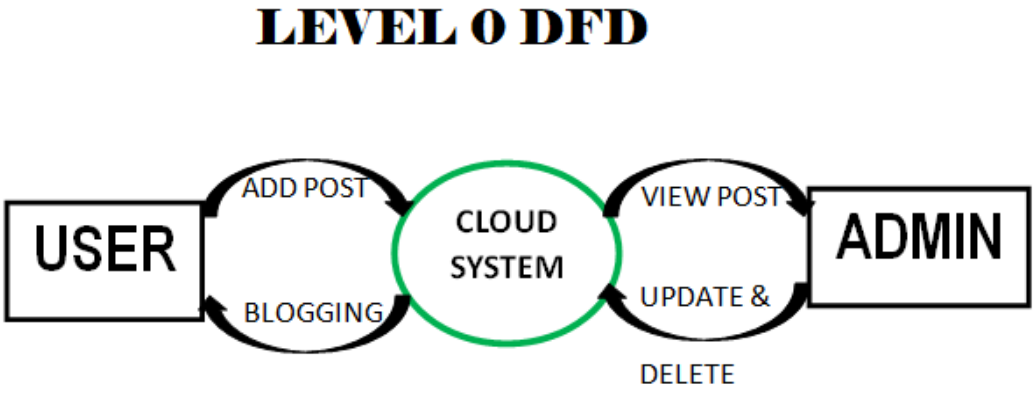



**FIG 2. ZERO LEVEL DFD**

# LEVEL 1 DFD

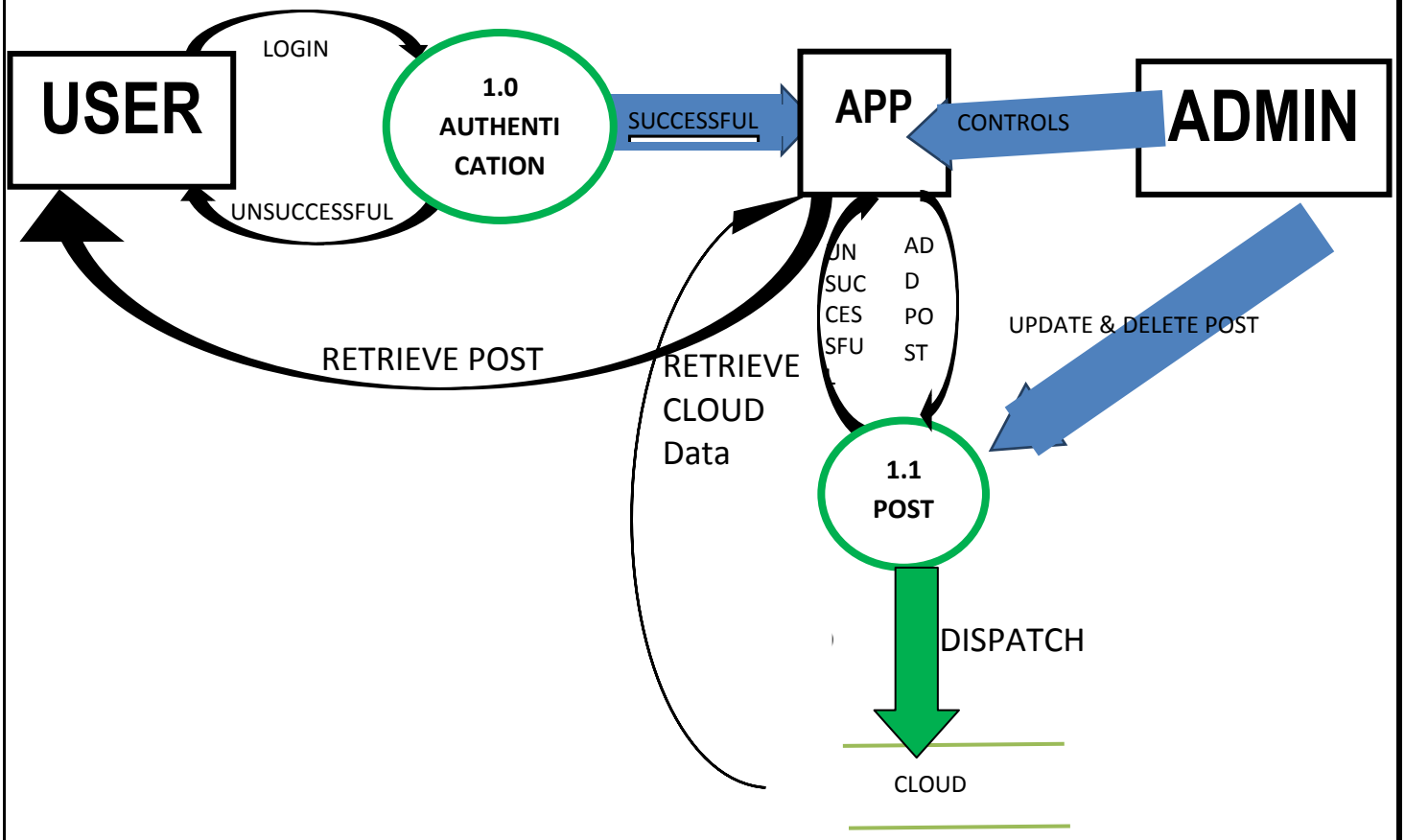



**FIG. 3. ONE LEVEL DFD**

# LEVEL 2 DFD

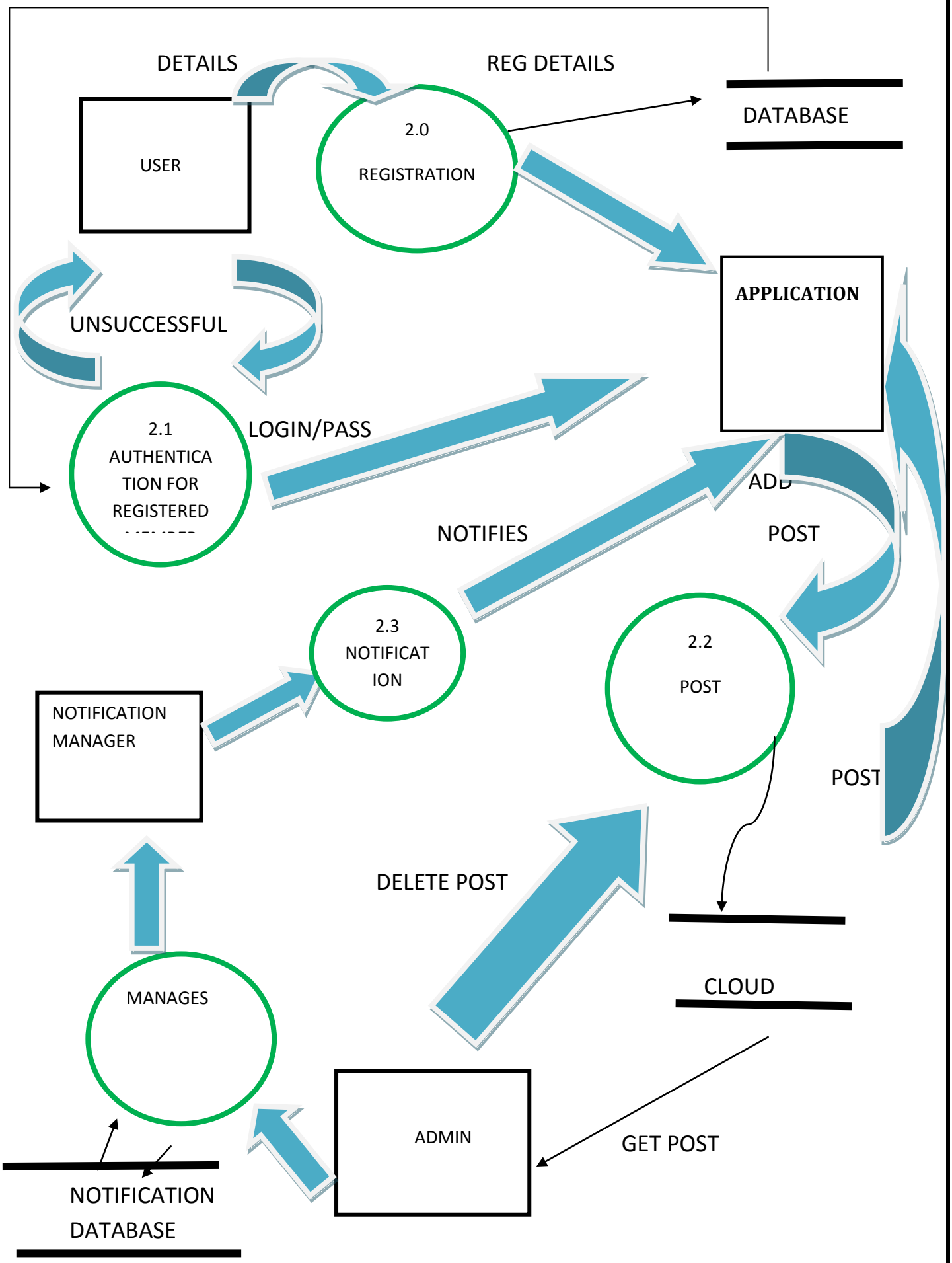



**FIG 4. TWO LEVEL DFD**

# SOFTWARE REQUIREMENTS SPECIFICATION

This section contains all of the functional and quality requirements of the system. It gives a detailed description of the system and all its features.

## 3.1 FUNCTIONAL SPECIFICATION

### 3.1.1 User interfaces

This section provides a detailed description of all inputs into and outputs from the system. It also gives a description of the hardware, software and communication interfaces and provides basic prototypes of the user interface.

A first-time user of the mobile application should see the log-in page when he/she opens the application, if the user has not registered, he/she should be able to do that on the log-in page.

If the user is not a first-time user, he/she should be able to see the story page directly when the application is opened (figure **17**.), here the user chooses the type of story he wants to see. Every user should have a profile page where they can edit their e-mail address, phone number and password.

The add icon in the story page below as shown in the figure **20**, which redirects to add new post PAGE. Add new page contains “Report” and “Date/Time” button to add time and date.

### 3.1.2 Hardware interfaces

Since neither the mobile application nor the web portal have any designated hardware, it does not have any direct hardware interfaces. The mobile phone and the hardware connection to the database server is managed by the underlying operating system on the mobile phone and the web server.

### 3.1.3 Software interfaces

The mobile application communicates with the Google Firebase which is cloud platform.
Software used: - Android Studio,
Frontend used: - Xml, android widgets.
Backend used: - cloud storage (google firebase storage)
The user is located and the visual representation of it, and with the database in order to get the information about the stories, see Figure 1.

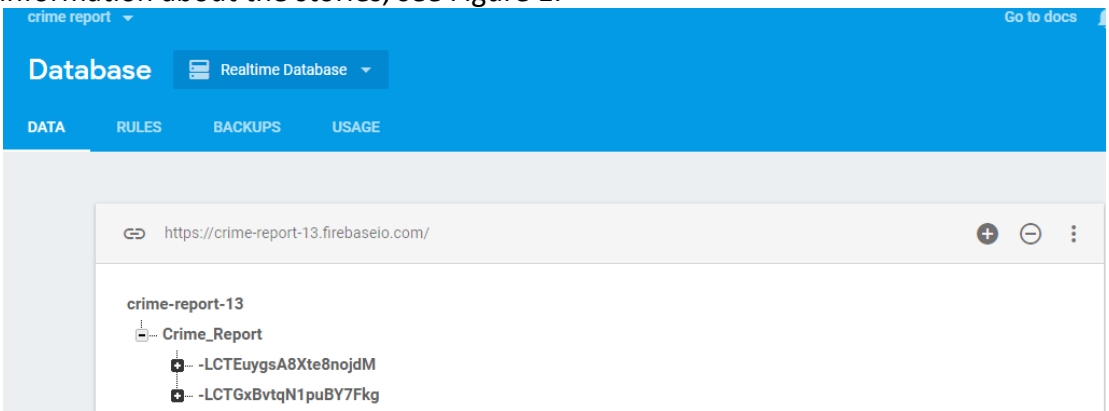



**FIG 5. DATABASE OF BLOG**

### 3.1.4 Communications interfaces

The communication between the cloud and the android app consists of operation concerning both reading and modifying the data as shown in the figure 6.

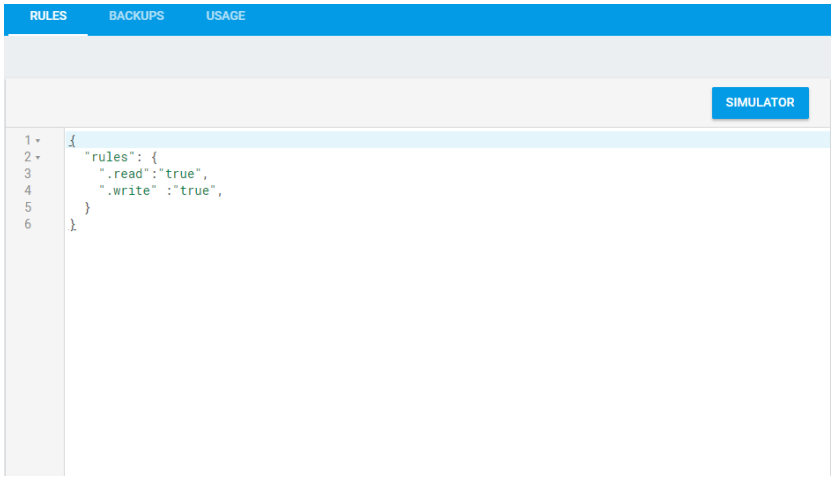



**FIG 6. DATABASE RULES FOR MODIFICATION**

While the communication between the user and the mobile application consists of only reading operations. The communication between the different parts of the system is important since they depend on each other. There is another way to communicate with the app using users and it is Notification. ADMIN creates the notification to redirect it to other pages.

## 3.2 NON-FUNCTIONAL REQUIREMENTS

### 3.2.1 Performance requirements

The requirements in this section provide a detailed specification of the user interaction with the software and measurements placed on the system performance.


**FIG 7. PERFORMANCE OVER TIME**

## 3.2.2 User engagement

There are certain events linked with the app Citizen Cop. So involvement of the users in the app for certain events is managed by cloud atmosphere.


**FIG 8. EVENT INVOLVEMENT OF THE USERS**

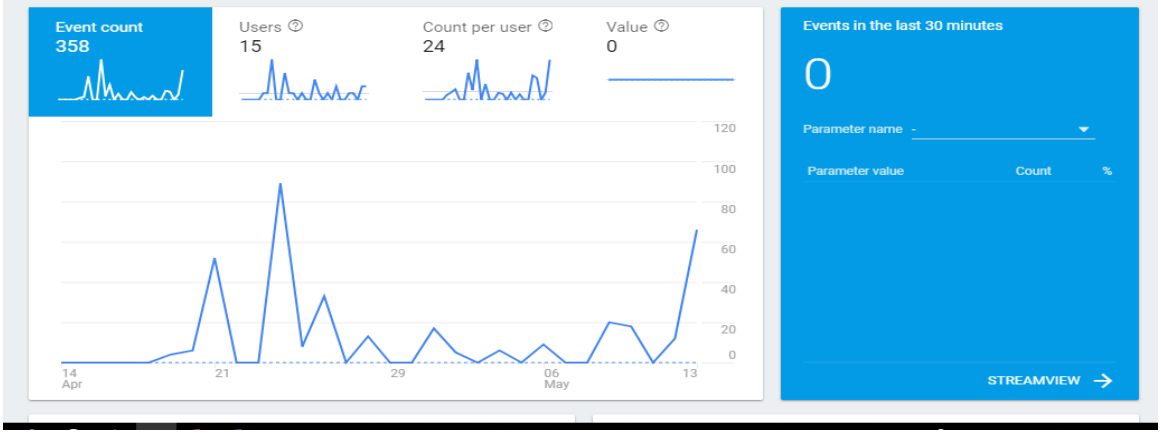



**FIG 9. EVEN VIEW AND COUNT OF THE APP BY THE USERS**

## 3.2.3 Security

Since Cloud aims to reach billions and millions of people, security is the most important issue that must be considered. To achieve this goal, no one can see another user's information without his or her permission. ACL (Access Control List) is used to construct a secure structure.

# DATA DICTIONARIES AND CLOUD STORAGE


**FIG 10. BLOG DATABASE**

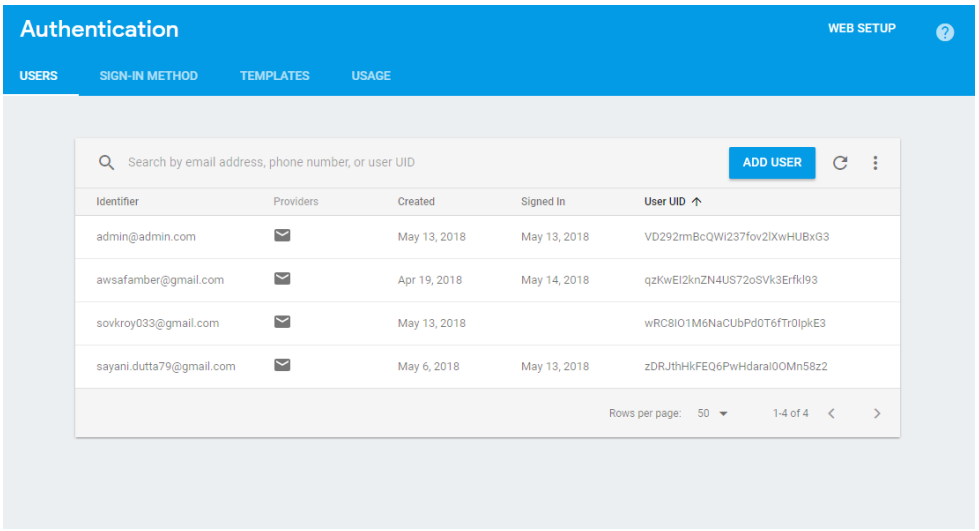



**FIG 11. USER AND ADMIN DATA**

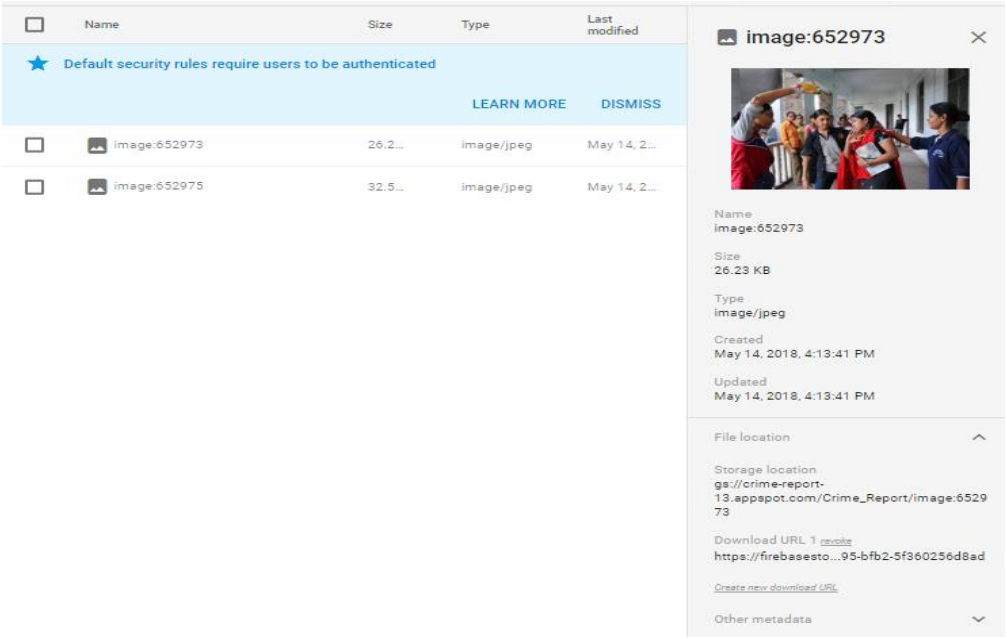



**FIG 12. POST DATABASE**

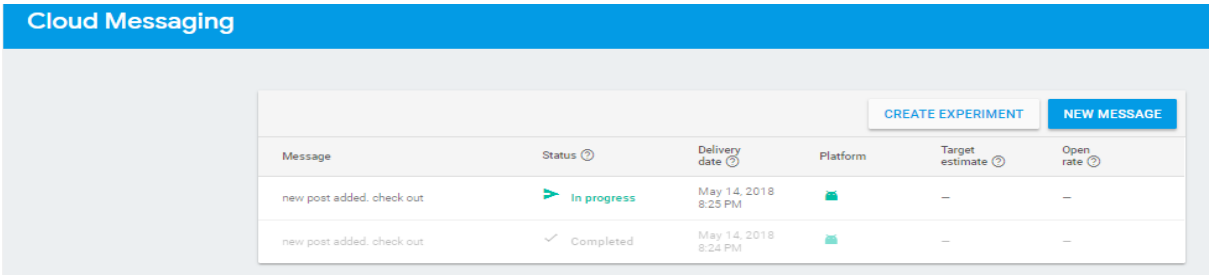



**FIG 13. NOTIFICATION DATABASE**

# PLANNING

In order to get a view of how to divide the requirements into different releases and what requirements should be included in which release, a prioritization of the requirements is needed. This section discusses the choice of prioritization methods and gives a suggestion of how the release plan for these requirements could look like

## 5.1 CHOICE OF PRIORITIZATION METHOD

When prioritizing the requirements, the 5 most important ones were picked out first. This was done with a simple "1 to 5" ranking method, with one being "very important" and five "not important" as shown in table II. Based on the elicitation meetings, and the perceived ideas of what was important to the different users, a number was set for each requirement. The numbers were then summed up for each requirement and the ten with the least score were chosen to be prioritized with the time value approach. These requirements were then prioritized according to the time value approach and the results can be viewed under Appendix II. This method was chosen since it gives the different users the same importance and has an enough wide range for determining which requirement is more important than the other. This requirement is important, but it might still get the most votes since one person cared about it. The yes-no vote method might be fairly simple to carry out, however the range is too narrow. For instance, if two requirements are not very important it would be hard to determine which of those requirements that is more important than the other. In conclusion, weighing the disadvantages and advantages of these methods against each other lead us to choose the five-way priority scheme.

## 5.1 RELEASE PLAN

The requirements were divided into three releases based on the prioritization and their dependencies. The three different releases were assembled so that each would work as a fully functional application. In the first release the requirements that build up the foundation of the application were included, together with the most highly prioritized requirements and their dependencies. The second release also includes important requirements. However, these requirements are not vital for a functional application. They are more suited to act as additional features that can contribute to making the software product more attractive. The third release includes the requirements that can be afforded to discard if the project gets delayed or overruns the budget. For further details about the release plan, see table III.

Prioritization Result of 5 selected Requirements Using Cost-Value Approach

**TABLE – II MOST IMPORTANT REQUIREMENTS**

| Requirement ID | Title | Requirement Type |
|---|---|---|
| F1 | BUDGET CLOUD SEARCH | Function |
| F2 | MOBILE PLATFORM SEARCH- ANDROID/ IOS/WINDOWS | Function |
| F3 | MOBILE APPLICATION ADD IMAGE WITH TEXT | Function |
| F4 | MOBILE APPLICATION - Search result in a list view | Function |
| Q1 | INTERNET CONNECTION | Quality |

**TABLE III – RELEASE PLAN**

| RE: Release EID | Dependencies | Description | Motivation | Duration(DAYS) |
|---|---|---|---|---|
| FR1 | F2,F1 | MOBILE APPLICATION ADD IMAGE WITH TEXT | This requirement makes the application available for users and is therefore an important requirement to include in the first release. | 10 |
| FR2 | F3,Q1 | User registration | For the user to be able to use the application, the user has to registrant. Consequently, this requirement needs to be met in the first release. | 5 |
| FR3 | FR2 | User Log-in | For the user to be able to use the application, the user has to log in. Consequently, this requirement needs to be met in the first release. | 2 |
| FR4 | FR3,FR1,Q1 | Search result in a list view | The ability to show the search result in a list view is part of the basic goal of the program. We have decided to put this one in the second release and only include the map result view in the first release. | 5 |
| FR5 | F1 | Log-in - admin | This should be included in the first release because the administrator needs to be able to log in so she/hi can manage the system. | 2 |
| FR6 | FR4 | Manage crime post. types - administrator | The type search will be added in the second release. | 5 |
| FR7 | FR3 | Verify POST-ADMIN | It's important for the admin to be able to verify the crime application for registration and post. | 2 |

# TESTING

Firebase Test Lab for Android lets you run the following types of tests:

- Espresso, Robotium or UI Automator 2.0 instrumentation tests written specifically to exercise your app.
- Robo test, which analyses the structure of your app's user interface and then explores it automatically by simulating user activities. [3]. Before you get started, you need to enable billing for your project. If you don't have an active billing account, add one and then connect your project to that billing account. You will need ownership or edit permissions in your project.

## To run a Robo test

1. On the Firebase console navigation bar, click **Test Lab**, and then click **Get Started -> Run a Robo test**.
2. Click **Browse**, browse to your app APK, and then click **Continue**.
3. Define your test matrix by selecting which devices, Android API levels, screen orientations and locales you want to test your app against.
4. **(Optional)** Click **Show advanced options** to change the following options:

- **Test timeout** determines the maximum duration of each test execution.
- **Max depth** determines how thoroughly Robo test explores a particular branch of your app's UI before returning to the root of the UI (the main screen) to explore another branch.
- **Test account credentials** is used to provide credentials for a test account. **Note:** This feature should never be used with real user accounts.
- **Additional fields** is used to provide text input for other text fields in your app. **Note:** To learn more about Test account credentials and additional fields, see Test account sign-in and predefined text.

5. Click **Start <N> Tests**, where **<N>** is the number of valid test configurations from the test matrix that you define on this screen. Each pending test is shown with a blue clock icon while it is waiting to run, and that icon changes to a green check when the test has completed.
6. After each test has run, click the device listed in the **Test Execution** column to see test results, including test cases, logs, screenshots and videos.

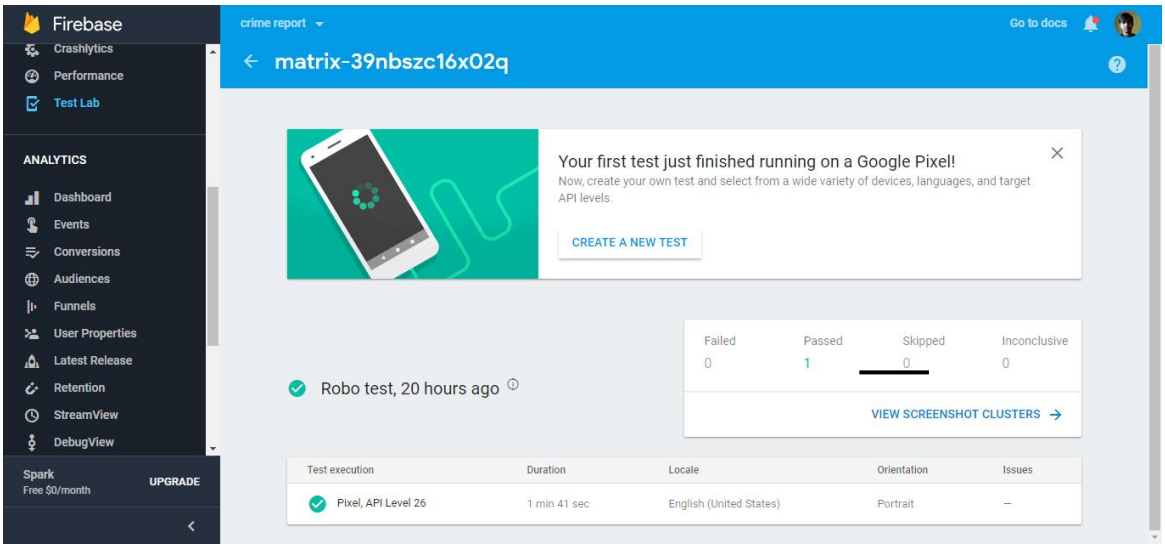



**FIG 14. ROBO TEST RESULTS**

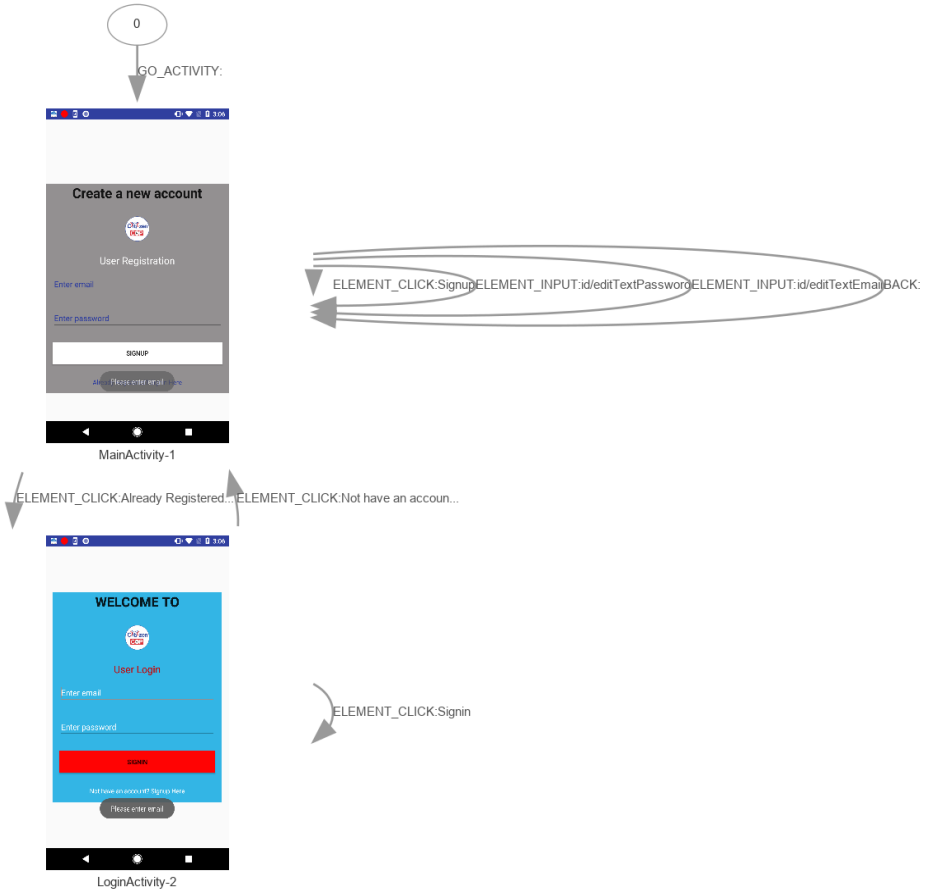



**FIG 15 ACTIVITY DIAGRAM**

## To run a unit test

1 Add a new module and then open AndroidMainfest.xml

2 Change intent of MainActivity or any default java activity to one which is newly added

3 Build apk or run the app.

4 It will change the launching default activity to one which is having <intent> in javaactivity

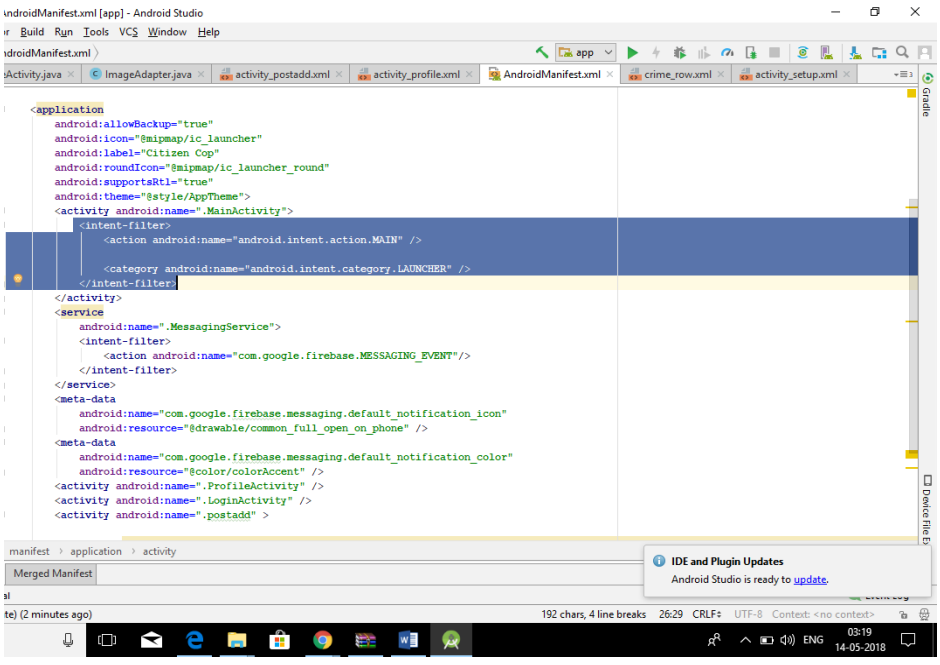



**FIG 16. HOW TO DO UNIT TESTING**

# MODULAR CODING & DESIGN

## AndroidMainfest.xml

Every application must have an AndroidManifest.xml file (with precisely that name) in its root directory. The manifest presents essential information about the application to the Android system, information the system must have before it can run any of the application's code [4].

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
package="com.example.sayanidutta.crimereport">

<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />

<" "-- To auto-complete the email text field in the login form with the user's emails
-->
<uses-permission android:name="android.permission.GET_ACCOUNTS" />
<uses-permission android:name="android.permission.READ_PROFILE" />
<uses-permission android:name="android.permission.READ_CONTACTS" />

<application
android:allowBackup="true"
android:icon="@mipmap/ic_launcher"
android:label="Citizen Cop"
android:roundIcon="@mipmap/ic_launcher_round"
android:supportsRtl="true"
android:theme="@style/AppTheme">
<activity android:name=".MainActivity">
<intent-filter>
<action android:name="android.intent.action.MAIN" />

<category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
</activity>
<service android:name=".MessagingService">
<intent-filter>
<action android:name="com.google.firebase.MESSAGING_EVENT" />
</intent-filter>
</service>

<meta-data
android:name="com.google.firebase.messaging.default_notification_icon"
android:resource="@drawable/common_full_open_on_phone" />
<meta-data
android:name="com.google.firebase.messaging.default_notification_color"
android:resource="@color/colorAccent" />

<activity android:name=".ProfileActivity" />
<activity android:name=".LoginActivity" />
<activity android:name=".postadd">

</activity>>
<activity android:name=".SetupActivity" />
<activity android:name=".admin"></activity>
</application>

</manifest>
```

**MainActivity.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayoutxmlns:android="http://schemas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context="com.example.sayanidutta.crimereport.MainActivity">

<android.support.v7.widget.Toolbar
android:id="@+id/toolbar_admin"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_alignParentTop="true"
android:layout_centerHorizontal="true"
android:background="#d21f1f">

</android.support.v7.widget.Toolbar>

<LinearLayout
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_alignParentBottom="true"

android:layout_marginBottom="60dp"
android:background="@color/colorAccent"
android:orientation="vertical">

<TextView
android:id="@+id/textViewz"
android:layout_width="match_parent"
android:layout_height="45dp"
android:layout_gravity="center"
android:freezesText="true"
android:text="Create a new account"
android:textAlignment="center"
android:textAppearance="@style/TextAppearance.AppCompat.Menu"
android:textSize="30sp"
android:textStyle="bold" />

<ImageView
android:id="@+id/imageView4"
android:layout_width="match_parent"
android:layout_height="wrap_content"
app:srcCompat="@mipmap/ic_launcher_foreground" />

<TextView
android:id="@+id/textView8"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_gravity="center_horizontal"
android:text="User Registration"
android:textAppearance="?android:attr/textAppearanceLarge"
android:textColor="@android:color/background_light" />

<EditText
android:id="@+id/editTextEmail"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_margin="15dp"
android:hint="Enter email"
android:inputType="textEmailAddress"
android:textColorHint="@color/colorPrimaryDark" />

<EditText
android:id="@+id/editTextPassword"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_margin="15dp"
android:hint="Enter password"
android:inputType="textPassword"
android:textColorHint="@color/colorPrimaryDark" />

<Button
```

```
    android:id="@+id/buttonSignup"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="15dp"
    android:background="@android:color/background_light"
    android:text="Signup" />

    <TextView
    android:id="@+id/textViewSignin"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="15dp"
    android:text="Already Registered? Signin Here"
    android:textAlignment="center"
    android:textColor="@color/colorPrimaryDark" />

    </LinearLayout>

</RelativeLayout>
```

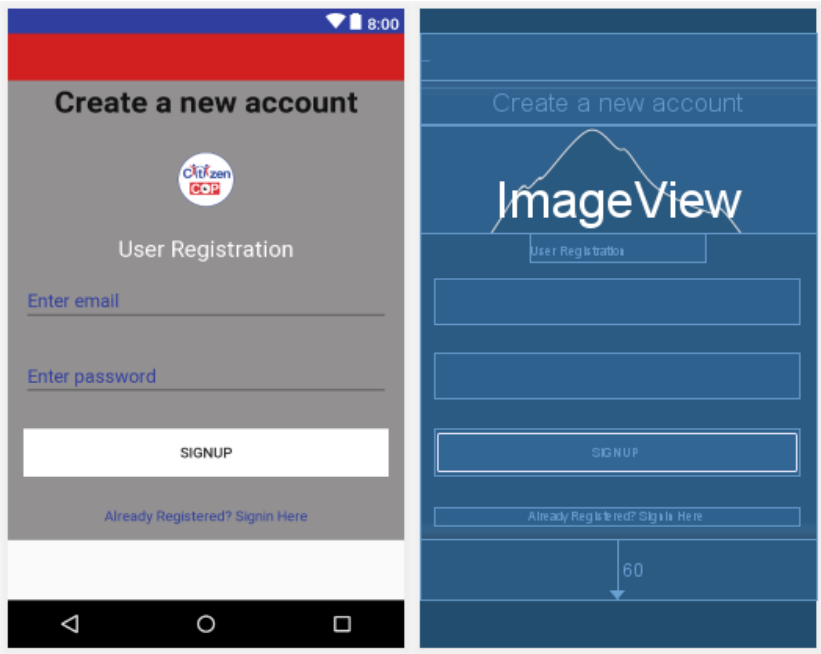



**FIG 17 DEFAULT/REGISTER PAGE DESIGN**

## MainActivity.java

```
package com.example.sayanidutta.crimereport;

import android.app.ProgressDialog;
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.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;
import android.widget.Toolbar;

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 static com.example.sayanidutta.crimereport.R.id.action_admin;
```

```
public class MainActivityextends AppCompatActivityimplements View.OnClickListener {

private Button buttonSignup;
private TextViewtextViewSignin;

private EditTexteditTextEmail;
private EditTexteditTextPassword;
private ProgressDialogprogressDialog;
private FirebaseAuthfirebaseAuth;
private android.support.v7.widget.Toolbar toolbar_ad;
@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);


//initializing firebase auth object

firebaseAuth= FirebaseAuth.getInstance();
toolbar_ad= (android.support.v7.widget.Toolbar) findViewById( R.id.toolbar_admin);
setSupportActionBar(toolbar_ad);
getSupportActionBar().setTitle( " CITIZEN COP " );

//if getCurrentUser does not returns null
if (firebaseAuth.getCurrentUser() ‟ ‟= null) {
//that means user is already logged in
            //so close this activity
finish();

//and open profile activity
startActivity(new Intent(getApplicationContext(), ProfileActivity.class));
        }

buttonSignup= (Button) findViewById(R.id.buttonSignup);
textViewSignin= (TextView) findViewById(R.id.textViewSignin);


//attaching listener to button
buttonSignup.setOnClickListener(this);
textViewSignin.setOnClickListener(this);
firebaseAuth= FirebaseAuth.getInstance();
//initializing views
editTextEmail= (EditText) findViewById(R.id.editTextEmail);
editTextPassword= (EditText) findViewById(R.id.editTextPassword);
progressDialog= new ProgressDialog(this);
registerUser();
    }
@Override
public booleanonCreateOptionsMenu(Menu menu) {
getMenuInflater().inflate( R.menu.admin_menu, menu );
return super.onCreateOptionsMenu( menu );
    }

@Override
public booleanonOptionsItemSelected(MenuItem item) {
if (item.getItemId() == R.id.action_admin)
        {
startActivity(new Intent(this, admin.class));
        }
return super.onOptionsItemSelected( item );
    }

    private void registerUser(){
    //getting email and password from edit texts
    String email = editTextEmail.getText().toString().trim();
        String password  = editTextPassword.getText().toString().trim();
//checking if email and passwords are empty
if(TextUtils.isEmpty(email)){
Toast.makeText(this,"Please enter email",Toast.LENGTH_LONG).show();
return;
        }
if(TextUtils.isEmpty(password)){
Toast.makeText(this,"Please enter password",Toast.LENGTH_LONG).show();
return;
        }
//if the email and password are not empty
```

```
        //displaying a progress dialog

progressDialog.setMessage("Registering Please Wait...");
progressDialog.show();

//creating a new user
firebaseAuth.createUserWithEmailAndPassword(email, password)
                .addOnCompleteListener(this, new OnCompleteListener<AuthResult>() {
@Override
public void onComplete(@NonNullTask<AuthResult> task) {
//checking if success
if(task.isSuccessful()){
//display some message here
Toast.makeText(MainActivity.this,"Successfully registered",Toast.LENGTH_LONG).show();
                        }else{
//display some message here
Toast.makeText(MainActivity.this,"Registration Error",Toast.LENGTH_LONG).show();
                        }
progressDialog.dismiss();
                    }
                }); }

@Override
public void onClick(View view) {
if (view == buttonSignup) {
//calling register method on click
registerUser();
        }
if (view == textViewSignin) {
//open login activity when user taps on the already registered textview
startActivity(new Intent(this, LoginActivity.class));
        }

    }
}
```

**Activity_login.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayoutxmlns:android="http://schemas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:paddingBottom="@dimen/activity_vertical_margin"
android:paddingLeft="@dimen/activity_horizontal_margin"
android:paddingRight="@dimen/activity_horizontal_margin"
tools:context="com.example.sayanidutta.crimereport.LoginActivity">

<android.support.v7.widget.Toolbar
android:id="@+id/toolbar_login"
android:layout_width="match_parent"
android:layout_height="67dp"
android:layout_alignParentTop="true"
android:layout_centerHorizontal="true"
android:layout_marginTop="5dp"
android:background="@android:color/darker_gray">

</android.support.v7.widget.Toolbar>

<LinearLayout
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_centerVertical="true"
android:background="@android:color/holo_blue_light"
android:orientation="vertical">

<TextView
android:id="@+id/textView2"
android:layout_width="match_parent"
android:layout_height="45dp"
android:layout_gravity="center"
android:freezesText="true"
android:text="WELCOME TO"
android:textAlignment="center"
android:textAppearance="@style/TextAppearance.AppCompat.Menu"
android:textSize="30sp"
android:textStyle="bold" />

<ImageView
android:id="@+id/imageView"
android:layout_width="match_parent"
android:layout_height="wrap_content"
app:srcCompat="@mipmap/ic_launcher_foreground" />

<TextView
android:id="@+id/textView"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_gravity="center_horizontal"
android:text="User Login"
android:textAppearance="?android:attr/textAppearanceLarge"
android:textColor="@android:color/holo_red_dark" />

<EditText
android:id="@+id/editTextEmail"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_margin="15dp"
android:hint="Enter email"
android:inputType="textEmailAddress"
android:singleLine="false"
android:textColorHint="@color/cardview_light_background" />

<EditText
android:id="@+id/editTextPassword"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_margin="15dp"
android:hint="Enter password"
```

```
android:inputType="textPassword"
android:textColorHint="@color/cardview_light_background" />
        <Button
android:id="@+id/buttonSignin"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_margin="15dp"
android:background="#ff0303"
android:text="Signin" />

<TextView
android:id="@+id/textViewSignUp"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_margin="15dp"
android:text="Not have an account? Signup Here"
android:textAlignment="center"
android:textColor="@android:color/background_light" />

</LinearLayout>

</RelativeLayout>
```

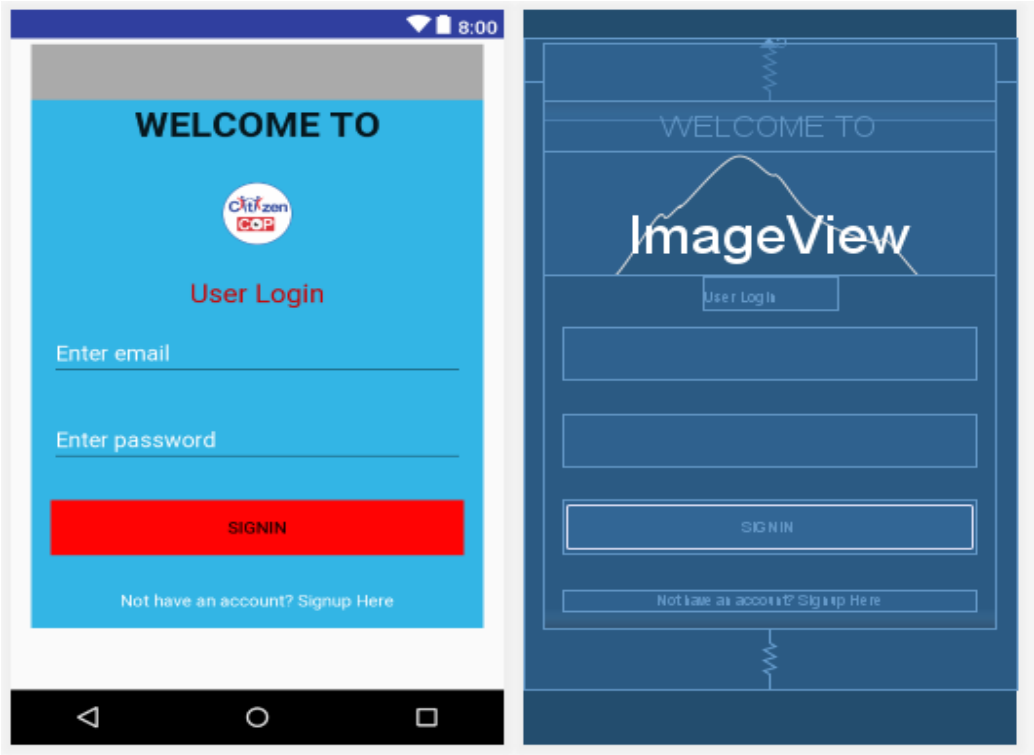



**FIG 18 LOGIN PAGE**

## LoginAvtivity.java

```
package com.example.sayanidutta.crimereport;

import android.app.ProgressDialog;
import android.content.Intent;
import android.support.annotation.NonNull;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
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.TextView;
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;
```

```
public class LoginActivityextends AppCompatActivityimplements View.OnClickListener {
//defining views
private Button buttonSignIn;
private EditTexteditTextEmail;
private EditTexteditTextPassword;
private TextViewtextViewSignup;
private Toolbar toolbar_lg;

//firebase auth object
private FirebaseAuthfirebaseAuth;

//progress dialog
private ProgressDialogprogressDialog;


@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_login);
toolbar_lg= (android.support.v7.widget.Toolbar) findViewById( R.id.toolbar_login);
setSupportActionBar(toolbar_lg);
getSupportActionBar().setTitle( " CITIZEN COP " );

//getting firebase auth object
firebaseAuth= FirebaseAuth.getInstance();

//if the objects getcurrentuser method is not null
        //means user is already logged in
if(firebaseAuth.getCurrentUser() ‟ ‟= null){
//close this activity
finish();
//opening profile activity
startActivity(new Intent(getApplicationContext(), ProfileActivity.class));
        }

//initializing views
editTextEmail= (EditText) findViewById(R.id.editTextEmail);
editTextPassword= (EditText) findViewById(R.id.editTextPassword);
buttonSignIn= (Button) findViewById(R.id.buttonSignin);
textViewSignup= (TextView) findViewById(R.id.textViewSignUp);

progressDialog= new ProgressDialog(this);

//attaching click listener
buttonSignIn.setOnClickListener(this);
textViewSignup.setOnClickListener(this);
    }

//method for user login
private void userLogin(){
        String email = editTextEmail.getText().toString().trim();
        String password  = editTextPassword.getText().toString().trim();


//checking if email and passwords are empty
if(TextUtils.isEmpty(email)){
Toast.makeText(this,"Please enter email",Toast.LENGTH_LONG).show();
return;
        }

//checking if email and passwords are empty
if(TextUtils.isEmpty(email)){
Toast.makeText(this,"Please enter email",Toast.LENGTH_LONG).show();
return;
    }

if(TextUtils.isEmpty(password)){
Toast.makeText(this,"Please enter password",Toast.LENGTH_LONG).show();
return;
    }
//if the email and password are not empty
        //displaying a progress dialog

progressDialog.setMessage("Registering Please Wait...");
progressDialog.show();
```

```
//logging in the user
firebaseAuth.signInWithEmailAndPassword(email, password)
                .addOnCompleteListener(this, new OnCompleteListener<AuthResult>() {
@Override
public void onComplete(@NonNullTask<AuthResult> task) {
progressDialog.dismiss();
//if the task is successfull
if(task.isSuccessful()){
//start the profile activity
finish();
startActivity(new Intent(getApplicationContext(), ProfileActivity.class));
                        }
                    }
                });
    }

@Override
public void onClick(View view) {
if(view == buttonSignIn){
userLogin();
        }

if(view == textViewSignup){
             finish();
startActivity(new Intent(this, MainActivity.class));
        }
    }
}
```

**ProfileActivity.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayoutxmlns:android="http://schemas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:layout_margin="5dp"
tools:context=".ProfileActivity">

<android.support.v7.widget.Toolbar
android:id="@+id/toolbar_STORIES"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:background="@android:color/holo_purple" />

<android.support.design.widget.FloatingActionButton
android:id="@+id/floatingActionButton2"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignParentBottom="true"
android:layout_alignParentEnd="true"
android:layout_marginBottom="31dp"
android:layout_marginEnd="31dp"
android:background="#fffffe"
android:clickable="true"
android:focusable="true"
app:backgroundTint="@android:color/holo_red_dark"
app:backgroundTintMode="add"
app:srcCompat="@android:drawable/ic_menu_add" />

<TextView
android:id="@+id/textViewUserEmail"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_alignParentStart="true"
android:layout_below="@+id/toolbar_STORIES"
android:layout_gravity="center_horizontal"
android:text="Large Text"
android:textAppearance="?android:attr/textAppearanceLarge" />

<ProgressBar
android:id="@+id/progress_circle"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignParentTop="true"
android:layout_centerHorizontal="true"
android:layout_marginTop="70dp" />

<android.support.v7.widget.RecyclerView
android:id="@+id/crime_List"
android:layout_width="match_parent"
android:layout_height="504dp"
android:layout_alignParentBottom="true"
android:layout_alignParentStart="true"
android:padding="5dp"
android:paddingBottom="5dp"
android:paddingLeft="5dp"
android:paddingRight="5dp"
android:paddingTop="5dp"
android:theme="?android:attr/alertDialogTheme">

</android.support.v7.widget.RecyclerView>
</RelativeLayout>
```

**CrimeRecyclerView.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.v7.widget.CardView
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
```

```
android:layout_height="match_parent"
android:layout_marginLeft="5dp"
android:layout_marginRight="5dp">

<RelativeLayout
android:layout_width="match_parent"
android:layout_height="613dp"
android:orientation="vertical"
android:padding="5dp"
android:paddingBottom="5dp"
android:paddingLeft="5dp"
android:paddingRight="5dp"
android:paddingTop="5dp">

<TextView
android:id="@+id/show_date"
android:layout_width="73dp"
android:layout_height="wrap_content"
android:layout_alignParentLeft="true"
android:layout_alignParentStart="true"
android:layout_below="@+id/show_time"
android:text="TextView"
android:textSize="11sp" />

<TextView
android:id="@+id/show_time"
android:layout_width="73dp"
android:layout_height="wrap_content"
android:layout_alignParentLeft="true"
android:layout_alignParentStart="true"
android:text="TextView"
android:textSize="10sp" />

<ImageView
android:id="@+id/crime_images"
android:layout_width="match_parent"
android:layout_height="214dp"
android:layout_alignParentStart="true"
android:layout_alignParentTop="true"
android:layout_marginTop="93dp"
android:adjustViewBounds="true"
android:contentDescription="@string/crime_scenes"
android:padding="5dp"
android:paddingLeft="5dp"
android:paddingRight="5dp"
tools:src="@mipmap/add_btn" />

<TextView
android:id="@+id/crime_title"
android:layout_width="260dp"
android:layout_height="39dp"
android:layout_alignEnd="@+id/crime_images"
android:layout_alignParentTop="true"
android:layout_alignRight="@+id/crime_images"
android:paddingBottom="10dp"
android:paddingLeft="5dp"
android:paddingRight="5dp"
android:text="@string/the_subject_of_crime_report"
android:textStyle="bold"
tools:padding="5dp"
tools:textSize="14sp" />

<TextView
android:id="@+id/crime_description"
android:layout_width="match_parent"
android:layout_height="87dp"
android:layout_alignParentStart="true"
android:layout_below="@+id/crime_images"
android:layout_marginTop="-5dp"
android:foregroundGravity="center"
android:padding="5dp"
android:paddingLeft="5dp"
android:paddingRight="5dp"
android:text="@string/the_crime_scene"
tools:textSize="14sp"
android:layout_alignParentLeft="true" />
```

```
<EditText
android:id="@+id/show_status"
android:layout_width="148dp"
android:layout_height="wrap_content"
android:layout_above="@+id/crime_images"
android:layout_alignParentLeft="true"
android:layout_alignParentStart="true"
android:layout_marginBottom="-93dp"
android:textSize="12sp" />

<EditText
android:id="@+id/show_location"
android:layout_width="192dp"
android:layout_height="wrap_content"
android:layout_above="@+id/crime_images"
android:layout_alignParentEnd="true"
android:layout_marginBottom="-93dp"
android:layout_marginEnd="1dp"
android:textSize="12sp" />
</RelativeLayout>


</android.support.v7.widget.CardView>
```

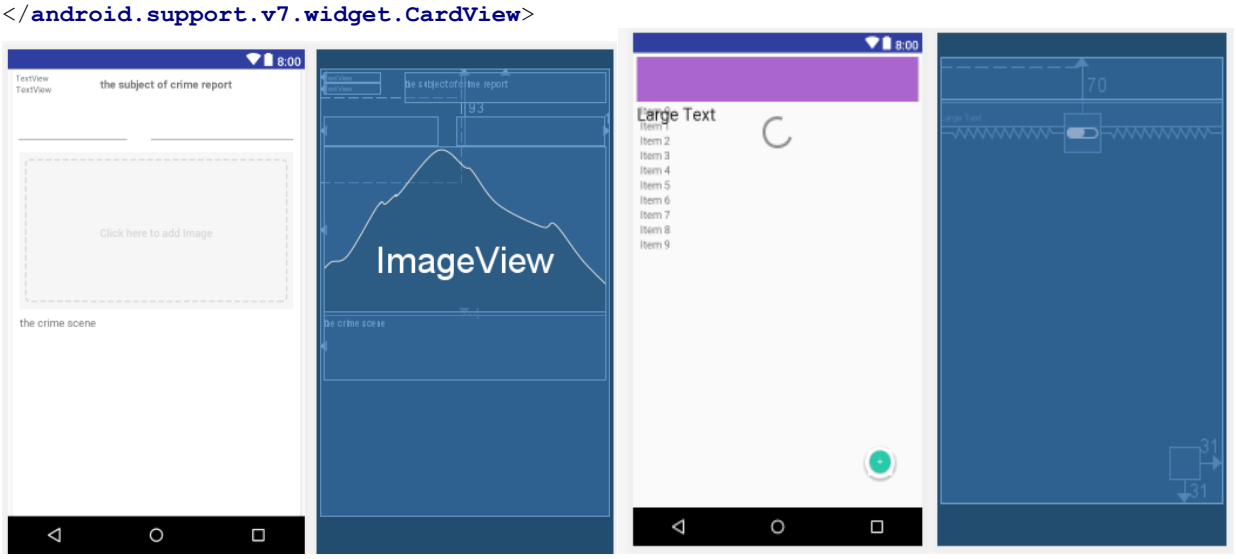



**FIG . 19 RECYCLER VIEW**

## StorageConstruction.java

```
package com.example.sayanidutta.crimereport;


import android.app.Fragment;
import android.os.Bundle;
import android.support.annotation.Nullable;

import com.google.firebase.database.Exclude;

public class Crime_Report {
private String mTitle,mDesc,mDate,mTime,mLocation,mStatus;
private String mImageUrl;
private String mKey;

public Crime_Report(){}
public Crime_Report(String title, String image, String description, String date,
String time, String location, String status){
mTitle= title;
mImageUrl=image;
mDesc= description;
mTime=time;
mDate=date;
mLocation= location;
mStatus= status;
    }
```

```
public String getImage() { return mImageUrl; }

public void setImage(String image) {  mImageUrl= image  }

public String getDescription() { return mDesc; }

public void setDescription(String description) { mDesc= description; }

public String getTitle() {   return mTitle }

public void setTitle(String title) { mTitle= title;}

public String getDate() {  return mDate;}

public void setDate(String date) {   mDate= date; }

public String getLocation(){ return mLocation; }

public void setLocation(String location){ mLocation= location; }

public String getStatus(){return mStatus;}

public void setStatus(String status){ mStatus= status;}

public String getTime() { return mTime; }

public void setTime(String time){ mTime= time;}


@Exclude
public String getKey() {
return mKey;
    }

@Exclude
public void setKey(String key) {
mKey= key;
    }

}
```

## Firebaseadapter.java

```
package com.example.sayanidutta.crimereport;

import android.content.Intent;
import android.support.v7.widget.RecyclerView;
import android.content.Context;
import android.view.ContextMenu;
import android.view.LayoutInflater;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.TextView;

import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.squareup.picasso.Picasso;

import java.util.List;

public class ImageAdapterextends RecyclerView.Adapter<ImageAdapter.ImageViewHolder> {
private Context mContext;
private List<Crime_Report>mCrimeBlog;
private OnItemClickListenermListener;


public ImageAdapter(Context context, List<Crime_Report>CrimeBlog) {
mContext= context;
mCrimeBlog= CrimeBlog;
```

```
    }

@Override
public ImageViewHolderonCreateViewHolder(ViewGroup parent, int viewType) {
        View v = LayoutInflater.from( mContext).inflate( R.layout.crime_row, parent,
false );
return new ImageViewHolder( v );
    }

@Override
public void onBindViewHolder(ImageViewHolder holder, int position) {
Crime_ReportuploadCurrent = mCrimeBlog.get( position );
holder.textViewName.setText( uploadCurrent.getTitle() );
holder.textViewNameb.setText( uploadCurrent.getDescription() );
holder.showstatus.setText( uploadCurrent.getStatus() );
holder.showdate.setText( uploadCurrent.getDate() );
holder.showlocation.setText( uploadCurrent.getLocation() );
holder.showtime.setText( uploadCurrent.getTime() );

Picasso.get()
                .load( uploadCurrent.getImage() )
                .placeholder( R.mipmap.ic_launcher)
                .fit()
                .centerCrop()
                .into( holder.imageView);


    }

@Override
public int getItemCount() {
return mCrimeBlog.size();
    }

public class ImageViewHolderextends RecyclerView.ViewHolderimplements
View.OnClickListener,
View.OnCreateContextMenuListener, MenuItem.OnMenuItemClickListener{
public TextViewtextViewName, textViewNameb,showlocation,showdate,showtime;
public ImageViewimageView;
public EditTextshowstatus;

public ImageViewHolder(View itemView) {
super( itemView );

textViewName= itemView.findViewById( R.id.crime_title);
showdate= itemView.findViewById( R.id.show_date);
showtime = itemView.findViewById( R.id.show_time);
showlocation= itemView.findViewById( R.id.show_location);
showstatus= itemView.findViewById( R.id.show_status);
textViewNameb= itemView.findViewById( R.id.crime_description);
imageView= itemView.findViewById( R.id.crime_images);
itemView.setOnClickListener( this );
itemView.setOnCreateContextMenuListener( this );

        }
public ImageViewHolder(View itemView) {
super( itemView );

textViewName= itemView.findViewById( R.id.crime_title);
showdate= itemView.findViewById( R.id.show_date);
showtime = itemView.findViewById( R.id.show_time);
showlocation= itemView.findViewById( R.id.show_location);
showstatus= itemView.findViewById( R.id.show_status);
textViewNameb= itemView.findViewById( R.id.crime_description);
imageView= itemView.findViewById( R.id.crime_images);
itemView.setOnClickListener( this );
itemView.setOnCreateContextMenuListener( this );

        }
@Override
public void onClick(View v) {
if (mListener" "= null) {
int position = getAdapterPosition();
if (position " "= RecyclerView.NO_POSITION) {
mListener.onItemClick(position);
                }
```

```
            }
        }

@Override
public void onCreateContextMenu(ContextMenu menu, View v,
ContextMenu.ContextMenuInfomenuInfo) {
menu.setHeaderTitle("Select Action");
MenuItemdoWhatever = menu.add(Menu.NONE, 1, 1, "Status");
MenuItem delete = menu.add( Menu.NONE, 2, 2, "Delete");

doWhatever.setOnMenuItemClickListener(this);
delete.setOnMenuItemClickListener(this);
        }
@Override
public booleanonMenuItemClick(MenuItem item) {
if (mListener‟ ‟= null) {
int position = getAdapterPosition();
if (position ‟ ‟= RecyclerView.NO_POSITION) {

switch (item.getItemId()) {
case 1:
mListener.onChangeAction(position);
return true;
case 2:
mListener.onDeleteClick(position);
return true;
                    }
                }
            }
return false;

        }
    }
public interface OnItemClickListener {
void onItemClick(int position);

void onChangeAction(int position);

void onDeleteClick(int position);
    }

public void setOnItemClickListener(OnItemClickListener listener) {
mListener= listener;
    }
}
```

**StoryView.java**

```
package com.example.sayanidutta.crimereport;

import android.app.ProgressDialog;
import android.content.Context;
import android.content.Intent;
import android.graphics.Bitmap;
import android.os.Bundle;
import android.provider.MediaStore;
import android.support.annotation.NonNull;
import android.support.design.widget.FloatingActionButton;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.LinearLayoutManager;
import android.support.v7.widget.RecyclerView;
import android.support.v7.widget.Toolbar;
import android.view.LayoutInflater;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.ProgressBar;
import android.widget.TextView;
import android.widget.Toast;

import com.firebase.ui.database.FirebaseRecyclerAdapter;
import com.firebase.ui.database.FirebaseRecyclerOptions;
```

```
import com.google.android.gms.tasks.OnFailureListener;
import com.google.android.gms.tasks.OnSuccessListener;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;
import com.google.firebase.storage.FirebaseStorage;
import com.google.firebase.storage.OnProgressListener;
import com.google.firebase.storage.StorageReference;
import com.google.firebase.storage.UploadTask;
import com.squareup.picasso.Picasso;

import java.io.IOException;
import java.util.ArrayList;
import java.util.List;


//import android.


public class ProfileActivityextends AppCompatActivityimplements View.OnClickListener,
ImageAdapter.OnItemClickListener {

private RecyclerViewmRecyclerView;
private ImageAdaptermAdapter;
private List<Crime_Report>blog_list;
private ProgressBarmProgressCircle;
private FloatingActionButtonadd;
private ValueEventListenermDBListener;
//initializing firebase authentication object
    //firebase auth object
public FirebaseAuthfirebaseAuth;

public StorageReferencemStorageRef;
private FirebaseStoragemStorage;

private DatabaseReferencemDatabase;
private TextViewtextViewUserEmail;
private Toolbar tool_story;


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

tool_story= (android.support.v7.widget.Toolbar) findViewById( R.id.toolbar_STORIES);
setSupportActionBar( tool_story);
getSupportActionBar().setTitle( " STORIES " );

//if the user is not logged in
        //that means current user will return null
{
firebaseAuth= FirebaseAuth.getInstance();
textViewUserEmail= findViewById( R.id.textViewUserEmail);

mStorageRef= FirebaseStorage.getInstance().getReference();
mStorage= FirebaseStorage.getInstance();
mProgressCircle= findViewById( R.id.progress_circle);
add = findViewById( R.id.floatingActionButton2 );


        }
mRecyclerView= findViewById( R.id.crime_List);
mRecyclerView.setHasFixedSize( true );
mRecyclerView.setLayoutManager( new LinearLayoutManager( this ) );
blog_list= new ArrayList<>();
mDatabase= FirebaseDatabase.getInstance().getReference( "Crime_Report" );

mStorageRef= FirebaseStorage.getInstance().getReference();
mAdapter= new ImageAdapter( ProfileActivity.this, blog_list);
mProgressCircle= findViewById( R.id.progress_circle);
mRecyclerView.setAdapter( mAdapter);
```

```
FirebaseUser user = firebaseAuth.getCurrentUser();
mAdapter.setOnItemClickListener( ProfileActivity.this);
add.setOnClickListener( this );
mDBListener= mDatabase.addValueEventListener( new ValueEventListener() {
@Override
public void onDataChange(DataSnapshotdataSnapshot) {
blog_list.clear();
for (DataSnapshotpostSnapshot : dataSnapshot.getChildren()) {
Crime_Report upload = postSnapshot.getValue( Crime_Report.class);
upload.setKey( postSnapshot.getKey() );
blog_list.add( 0, upload );
            }

mAdapter.notifyDataSetChanged();

mProgressCircle.setVisibility( View.INVISIBLE);
        }

@Override
public void onCancelled(DatabaseErrordatabaseError) {
Toast.makeText( ProfileActivity.this, databaseError.getMessage(),
Toast.LENGTH_SHORT).show();
mProgressCircle.setVisibility( View.INVISIBLE);
        }
    } );
if (firebaseAuth.getCurrentUser() == null) {
//closing this activity
finish();
//starting login activity
startActivity( new Intent( this, LoginActivity.class) );
    }

//getting current user


    //displaying logged in user name
textViewUserEmail.setText( "Welcome " + user.getEmail() );


}
@Override
public booleanonCreateOptionsMenu(Menu menu) {
getMenuInflater().inflate( R.menu.menu_main, menu );
return super.onCreateOptionsMenu( menu );
    }

@Override
public booleanonOptionsItemSelected(MenuItem item) {
if (item.getItemId() == R.id.Add_Pst) {
            Intent intent = new Intent( ProfileActivity.this, postadd.class);
startActivity( intent );
        }
if (item.getItemId() == R.id.Logout) {
//logging out the user
firebaseAuth.signOut();
//closing activity
finish();
//starting login activity
startActivity( new Intent( this, LoginActivity.class) );
        }
if (item.getItemId() == R.id.action_settings) {

        }
return super.onOptionsItemSelected( item );
    }


@Override
public void onClick(View v) {
if (v == add) {
            Intent intent = new Intent( ProfileActivity.this, postadd.class);
startActivity( intent );

        }
    }
```

```
@Override
public void onItemClick(int position) {
Toast.makeText( this, "Normal click at position: " + position,
Toast.LENGTH_SHORT).show();

    }
@Override
public void onChangeAction(int position) {
Toast.makeText( this, "change status Clicked at position: " + position,
Toast.LENGTH_SHORT).show();
    }
@Override
public void onDeleteClick(int position) { Crime_ReportselectedItem = blog_list.get(
position );
final String selectedKey = selectedItem.getKey();

StorageReferenceimageRef = mStorage.getReferenceFromUrl( selectedItem.getImage() );
Toast.makeText( this, "Delete click at position: " + imageRef + "   " + selectedKey,
Toast.LENGTH_SHORT).show();
imageRef.delete().addOnSuccessListener( new OnSuccessListener<Void>() {
@Override
public void onSuccess(Void aVoid) {
mDatabase.child( selectedKey).removeValue();
Toast.makeText( ProfileActivity.this, "Item deleted", Toast.LENGTH_SHORT).show();
            }
        } );
    }

@Override
protected void onDestroy() {
super.onDestroy();
mDatabase.removeEventListener( mDBListener);
    }
}
```

## Add post.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayoutxmlns:android="http://schemas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:layout_margin="5dp"
android:padding="5dp"
tools:context=".postadd">

<android.support.v7.widget.Toolbar
android:id="@+id/toolbar_addpost"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_alignParentStart="true"
android:layout_alignParentTop="true"
android:background="?attr/colorPrimary"
android:minHeight="?attr/actionBarSize"
android:theme="?attr/actionBarTheme" />

<ImageButton
android:id="@+id/choose"
android:layout_width="match_parent"
android:layout_height="221dp"
android:layout_alignParentStart="true"
android:layout_below="@+id/toolbar_addpost"
android:adjustViewBounds="true"
android:background="@mipmap/add_btn"
android:contentDescription="@string/todo"
android:cropToPadding="true"
android:paddingLeft="5dp"
android:paddingRight="5dp" />

<EditText
android:id="@+id/storyHeading"
android:layout_width="match_parent"
android:layout_height="46dp"
android:layout_alignParentStart="true"
android:layout_below="@+id/choose"
android:hint="@string/add_title_here"
android:inputType="text|textAutoComplete"
android:singleLine="false" />

<Button
android:id="@+id/load"
android:layout_width="match_parent"
android:layout_height="32dp"
android:layout_alignParentBottom="true"
android:background="@android:color/holo_red_dark"
android:text="REPORT"
android:textColor="@android:color/white" />

<TextView
android:id="@+id/timehr"
android:layout_width="109dp"
android:layout_height="34dp"
android:layout_alignParentStart="true"
android:layout_below="@+id/storyHeading"
android:layout_marginStart="17dp"
android:hint="time"
android:textColorHint="@android:color/black" />

<EditText
android:id="@+id/Description"
android:layout_width="match_parent"
android:layout_height="130dp"
android:layout_above="@+id/load"
android:layout_alignParentStart="true"
android:hint="@string/add_description_here"
android:inputType="textMultiLine"
android:maxLength="500"
android:singleLine="false"
android:textSize="14sp" />
```

```
<Button
android:id="@+id/timeStamPicker"
android:layout_width="70dp"
android:layout_height="35dp"
android:layout_alignParentEnd="true"
android:layout_below="@+id/storyHeading"
android:text="date/time"
android:textSize="8sp" />

<TextView
android:id="@+id/datehr"
android:layout_width="120dp"
android:layout_height="34dp"
android:layout_below="@+id/storyHeading"
android:layout_toEndOf="@+id/timehr"
android:hint="date"
android:textColorHint="@android:color/black" />

<EditText
android:id="@+id/locationhr"
android:layout_width="match_parent"
android:layout_height="34dp"

android:layout_alignParentStart="true"
android:layout_below="@+id/timehr"
android:hint="location"
android:textColorHint="@android:color/black"
android:textSize="12sp" />


</RelativeLayout>
```

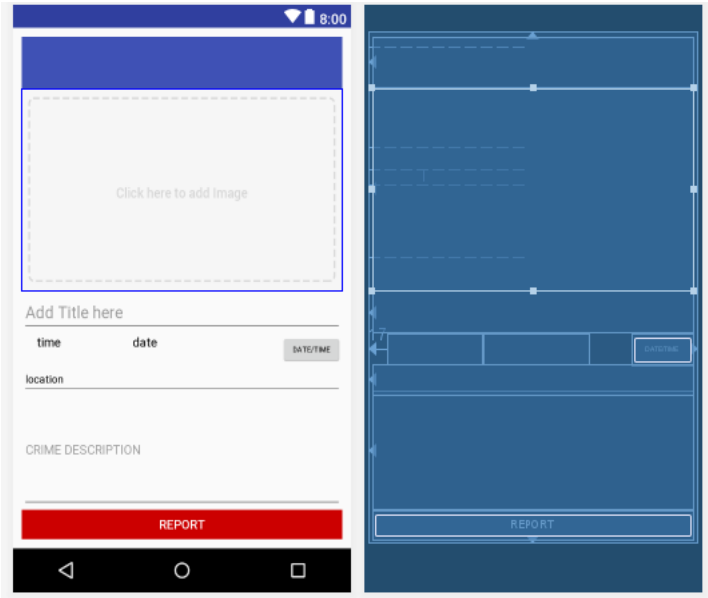



**FIG . 20 ADD NEW POST UI**

## AddPost.java

```
package com.example.sayanidutta.crimereport;

import android.app.DatePickerDialog;
import android.app.ProgressDialog;
import android.app.TimePickerDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.graphics.Bitmap;
import android.net.Uri;
import android.provider.MediaStore;
import android.support.annotation.NonNull;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.text.TextUtils;
import android.util.EventLogTags;
import android.view.View;
```

```
import android.widget.Button;
import android.widget.DatePicker;
import android.widget.EditText;
import android.widget.ImageButton;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.TimePicker;
import android.widget.Toolbar;

import com.google.android.gms.tasks.OnFailureListener;
import com.google.android.gms.tasks.OnSuccessListener;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.storage.FirebaseStorage;
import com.google.firebase.storage.OnProgressListener;
import com.google.firebase.storage.StorageReference;
import com.google.firebase.storage.UploadTask;

import java.io.IOException;
import java.text.DateFormat;
import java.util.Calendar;

import static android.widget.TimePicker.*;

public class postadd extends AppCompatActivity implements
OnClickListener,DatePickerDialog.OnDateSetListener, TimePickerDialog.OnTimeSetListener
{


    int day,month,year,hour,minutes;
    int dayFinal,monthFinal,yearFinal,hourFinal,minuteFinal;
    String FirDate,FirTime;
    private ImageButton slct;
    private Button uplood,datePicker;
    private Uri FilePath;
    private static final int glr = 2;
   // private FirebaseAuth firebaseAuth;
    public StorageReference mStorageRef;
    private EditText StoryHeading,description,Firlocation;
    private DatabaseReference mDatabase;
    private  ProgressDialog progressDialog;
    private TextView CrimeDate,CrimeTime;
    private android.support.v7.widget.Toolbar add_post_tool;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_postadd);
        slct = (ImageButton) findViewById(R.id.choose);
        add_post_tool = (android.support.v7.widget.Toolbar) findViewById(
R.id.toolbar_addpost );
        setSupportActionBar(add_post_tool);
        getSupportActionBar().setTitle( " ADD POST " );
        StoryHeading = (EditText) findViewById(R.id.storyHeading);
        description = (EditText) findViewById(R.id.Description);
        Firlocation=(EditText) findViewById( R.id.locationhr );
        CrimeDate=(TextView) findViewById( R.id.datehr );
        CrimeTime=(TextView) findViewById( R.id.timehr );
        datePicker = findViewById( R.id.timeStamPicker );
        uplood = findViewById(R.id.load);
        slct.setOnClickListener(this);
        uplood.setOnClickListener(this);
        datePicker.setOnClickListener( this );
      /*  FirebaseUser user = firebaseAuth.getCurrentUser();*/
        mStorageRef= FirebaseStorage.getInstance().getReference();
        mDatabase=
FirebaseDatabase.getInstance().getReference().child("Crime_Report");

    }

    @Override
    public void onClick(View v) {
        if (v == slct) {
            Intent intent = new Intent();
            intent.setType("image/*");
```

```
            intent.setAction(Intent.ACTION_GET_CONTENT);
            startActivityForResult(Intent.createChooser(intent, "select an image"),
glr);
        }
        if(v== datePicker){
            Calendar c=Calendar.getInstance();
            year = c.get(Calendar.YEAR);
            month = c.get(Calendar.MONTH);
            day = c.get(Calendar.DAY_OF_MONTH);

            DatePickerDialog datePickerDialog = new DatePickerDialog(
postadd.this,postadd.this,year,month,day );
            datePickerDialog.show();

        }
        if (v == uplood) {

            final String StoryHeading_val= StoryHeading.getText().toString().trim();
            final String Description_val = description.getText().toString().trim();
            final String Crime_Time =CrimeTime.getText().toString().trim();
            final String Crime_Date =CrimeDate.getText().toString().trim();
            final String Crime_location =Firlocation.getText().toString().trim();


                    if (!TextUtils.isEmpty(StoryHeading_val) &&
!TextUtils.isEmpty(Description_val) && !TextUtils.isEmpty(Crime_Date) &&
!TextUtils.isEmpty(Crime_Time) &&  FilePath!= null)
                    { progressDialog = new ProgressDialog(this);
                progressDialog.setTitle("Uploading.....");
                progressDialog.show();
                StorageReference filepath =
mStorageRef.child("Crime_Report").child(FilePath.getLastPathSegment());

               filepath.putFile(FilePath)
                        .addOnProgressListener(new
OnProgressListener<UploadTask.TaskSnapshot>() {
                            @Override
                            public void onProgress(UploadTask.TaskSnapshot
taskSnapshot) {
                                double progress
=(100*taskSnapshot.getBytesTransferred())/taskSnapshot.getTotalByteCount();
                                progressDialog.setMessage("Uploading " + ((int)
progress) + "%...");
                                progressDialog.show();
                            }
                        })
                         .addOnSuccessListener(new
OnSuccessListener<UploadTask.TaskSnapshot>() {
                            @Override
                            public void onSuccess(UploadTask.TaskSnapshot
taskSnapshot) {
                                //
                                //Get a URL to the uploaded content
                                    Uri downloadUrl = taskSnapshot.getDownloadUrl();
                                    DatabaseReference newPost =mDatabase.push();

                                    newPost.child("title").setValue(StoryHeading_val);

newPost.child("description").setValue(Description_val);

newPost.child("image").setValue(downloadUrl.toString());
                                    newPost.child( "date" ).setValue( Crime_Date );
                                    newPost.child( "time" ).setValue( Crime_Time );
                                    newPost.child( "status" ).setValue( "Not Verified"
);
                                    newPost.child( "location").setValue(
Crime_location  );
                                    progressDialog.dismiss();

                            }
                        })
                        .addOnFailureListener(new OnFailureListener() {
                            @Override
                            public void onFailure(@NonNull Exception exception) {
                                // Handle unsuccessful uploads
                                // ...
```

```
                                }

                            });

                }

            }
        }

        @Override
        protected void onActivityResult(int requestCode, int resultCode, Intent data) {
            super.onActivityResult(requestCode, resultCode, data);
            if (requestCode == glr && resultCode == RESULT_OK && data != null &&
data.getData() != null) {
                FilePath = data.getData();
                try {
                    Bitmap bitmap =
MediaStore.Images.Media.getBitmap(getContentResolver(), FilePath);
                    slct.setImageBitmap(bitmap);

                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
        }

        @Override
        public void onDateSet(DatePicker view, int year, int month, int dayOfMonth) {
        yearFinal = year;
        monthFinal = month+1;
        dayFinal = dayOfMonth;

        Calendar c = Calendar.getInstance();
        hour=c.get( Calendar.HOUR_OF_DAY );
        minutes=c.get( Calendar.MINUTE  );

        TimePickerDialog timePickerDialog = new TimePickerDialog(
postadd.this,postadd.this,hour,minutes, android.text.format.DateFormat.is24HourFormat(
this ));
        timePickerDialog.show();


        }

        @Override
        public void onTimeSet(TimePicker view, int hourOfDay, int minute) {
            hourFinal = hourOfDay;
            minuteFinal = minute;
            FirDate = Integer.toString(dayFinal)+": "+Integer.toString(monthFinal)+":
"+Integer.toString(yearFinal);
            FirTime = Integer.toString(hourFinal)+"HH:
"+Integer.toString(minuteFinal)+"MM";
            CrimeDate.setText( FirDate );
            CrimeTime.setText( FirTime );

        }
}
```

**Admin.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayoutxmlns:android="http://schemas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context=".admin">

<android.support.v7.widget.Toolbar
android:id="@+id/admin_tool"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:background="@android:color/background_dark"
app:subtitle="ADMIN"
app:subtitleTextColor="@android:color/background_light"></android.support.v7.widget.To
olbar>

<android.support.v7.widget.LinearLayoutCompat
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_centerVertical="true"
android:orientation="vertical">

<TextView
android:id="@+id/textViewx"
android:layout_width="match_parent"
android:layout_height="45dp"
android:layout_gravity="center"
android:freezesText="true"
android:text="WELCOME TO"
android:textAlignment="center"
android:textAppearance="@style/TextAppearance.AppCompat.Menu"
android:textSize="30sp"
android:textStyle="bold" />

<ImageView
android:id="@+id/imageView4a"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_below="@+id/textViewx"
app:srcCompat="@mipmap/ic_launcher_foreground" />

<EditText
android:id="@+id/admin_uid"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_below="@+id/admin_tool"
android:hint="Username" />

<EditText
android:id="@+id/admin_upass"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_marginTop="0dp"
android:hint="Password" />

<Button
android:id="@+id/admin_signIn"
android:layout_width="match_parent"
android:layout_height="wrap_content"
</android.support.v7.widget.LinearLayoutCompat>

</RelativeLayout>
```

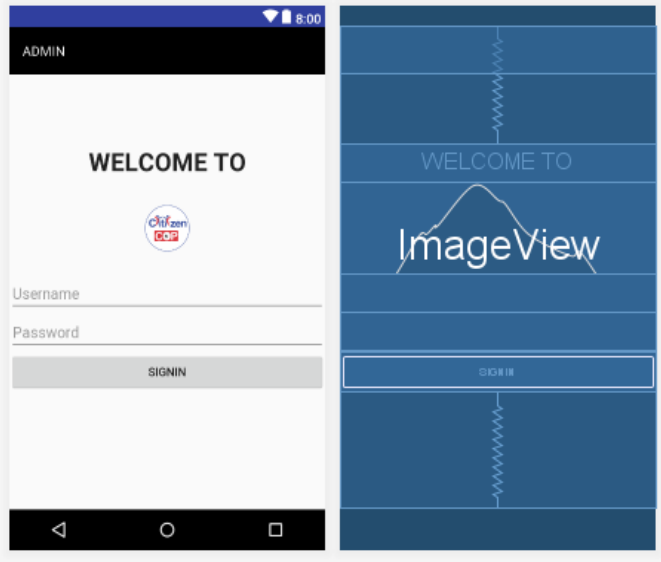



**FIG 21 ADMIN CONTROL SYSTEMS**

## Admin.java

```
package com.example.sayanidutta.crimereport;

import android.content.Intent;
import android.net.Uri;
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.TextView;
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;

public class admin extends AppCompatActivityimplements View.OnClickListener {
private EditTextUid;
private EditTextUpass;
private TextViewAdminSignIn;
private FirebaseAuthfirebaseAuth;
@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate( savedInstanceState );
setContentView( R.layout.activity_admin);
firebaseAuth= FirebaseAuth.getInstance();

//if the objects getcurrentuser method is not null
        //means user is already logged in
if(firebaseAuth.getCurrentUser() " "= null){
//close this activity
finish();
//opening profile activity
startActivity(new Intent(getApplicationContext(), ProfileActivity.class));
        }
//initializing views
Uid= (EditText) findViewById(R.id.admin_uid);
Upass= (EditText) findViewById(R.id.admin_upass);
AdminSignIn= (Button) findViewById(R.id.admin_signIn);

AdminSignIn.setOnClickListener(this);
}
private void AdminLogin() {
    String email = Uid.getText().toString().trim();
    String password = Upass.getText().toString().trim();
//checking if email and passwords are empty
```

```
if (TextUtils.isEmpty( email )) {
Toast.makeText( this, "Please enter email", Toast.LENGTH_LONG).show();
return;
    }


if (TextUtils.isEmpty( password )) {
Toast.makeText( this, "Please enter password", Toast.LENGTH_LONG).show();
return;
        }
//if the email and password are not empty
        //displaying a progress dialog
{
if (email.equals("admin@admin.com")) {
//logging in the user
firebaseAuth.signInWithEmailAndPassword( email, password )
                        .addOnCompleteListener( this, new
OnCompleteListener<AuthResult>() {
@Override
public void onComplete(@NonNullTask<AuthResult> task) {
//if the task is successfull
if (task.isSuccessful()) {
//start the profile activity
finish();
startActivity( new Intent( getApplicationContext(), ProfileActivity.class) );
                                    }
                                }
                            } );
  }    }      }

@Override
public void onClick(View v) {
if(v == AdminSignIn){
AdminLogin();
 }    }     }
```

## NotificationManager.java

```
package com.example.sayanidutta.crimereport;

import...

public class MessagingServiceextends FirebaseMessagingService {

@Override
public void onMessageReceived(RemoteMessageremoteMessage) {
showNotification(remoteMessage.getNotification().getBody());
    }

public void showNotification(String message) {
PendingIntent pi = PendingIntent.getActivity(this, 0, new Intent(this,
MainActivity.class), 0);
      Notification notification = new NotificationCompat.Builder( this )
                .setSmallIcon(R.mipmap.ic_launcher)
                .setContentTitle("CITIZEN COP")
                .setContentText(message)
                .setContentIntent(pi)
                .setAutoCancel(true)
                .build();

NotificationManagernotificationManager = (NotificationManager)
getSystemService(NOTIFICATION_SERVICE);
notificationManager.notify(0, notification);
    }
}
```

# RESULTS AND DISCUSSIONS

- After making connection with cloud storage and designing the whole app along with the .Json file to link with the cloud server.
  This UI takes all the details in terms of FIR (first Information Report) which must have information as Date, Time, Location and Crime Scene

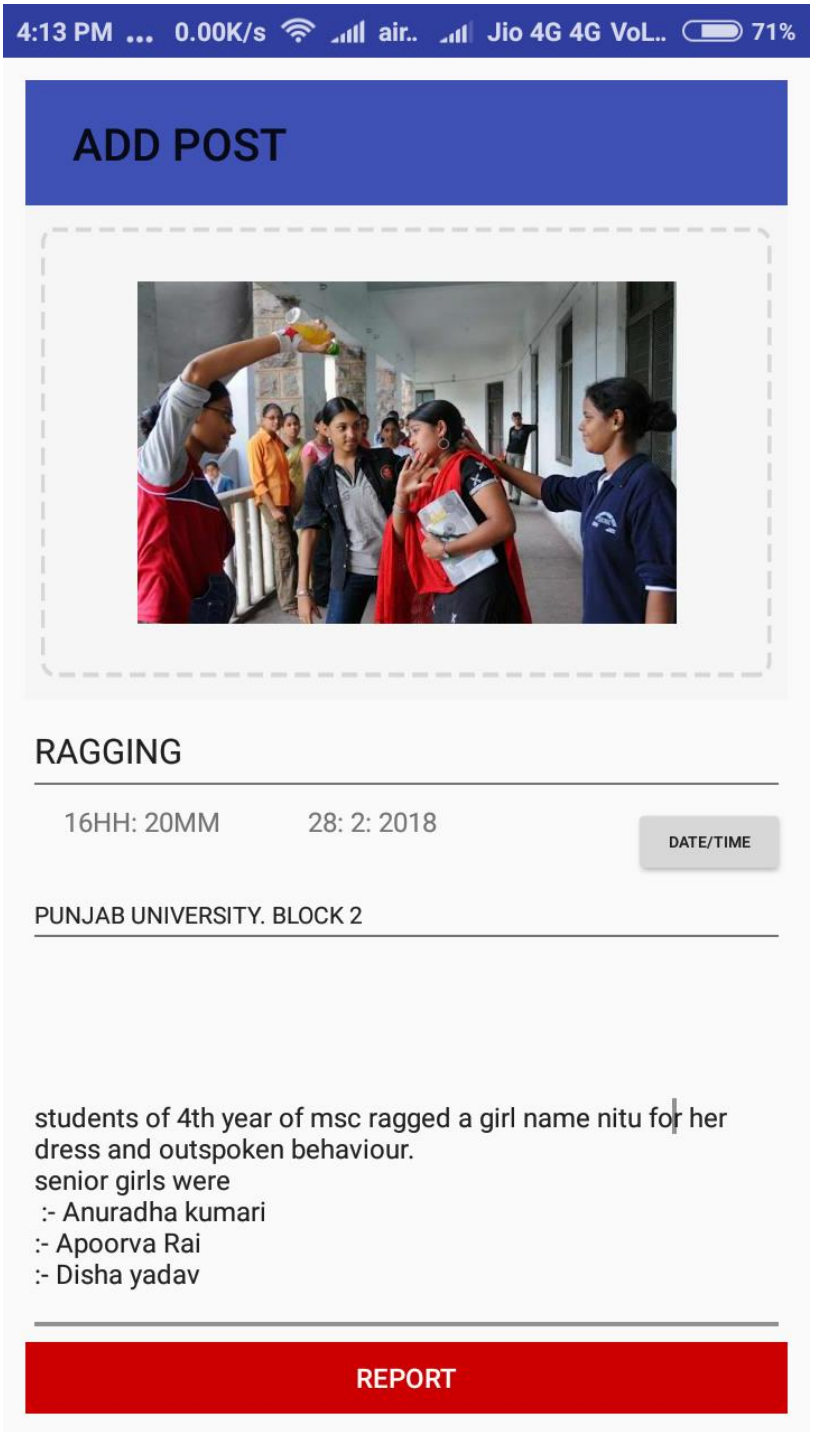



**FIG 22 . NEW POST ADDING UI**

- This is the card View of the post which was made previously. In this all the post appear in the list form.
  Here we took help from the picasso image viewer[5].
  All the details which is basically FiR report was shown along with the images.

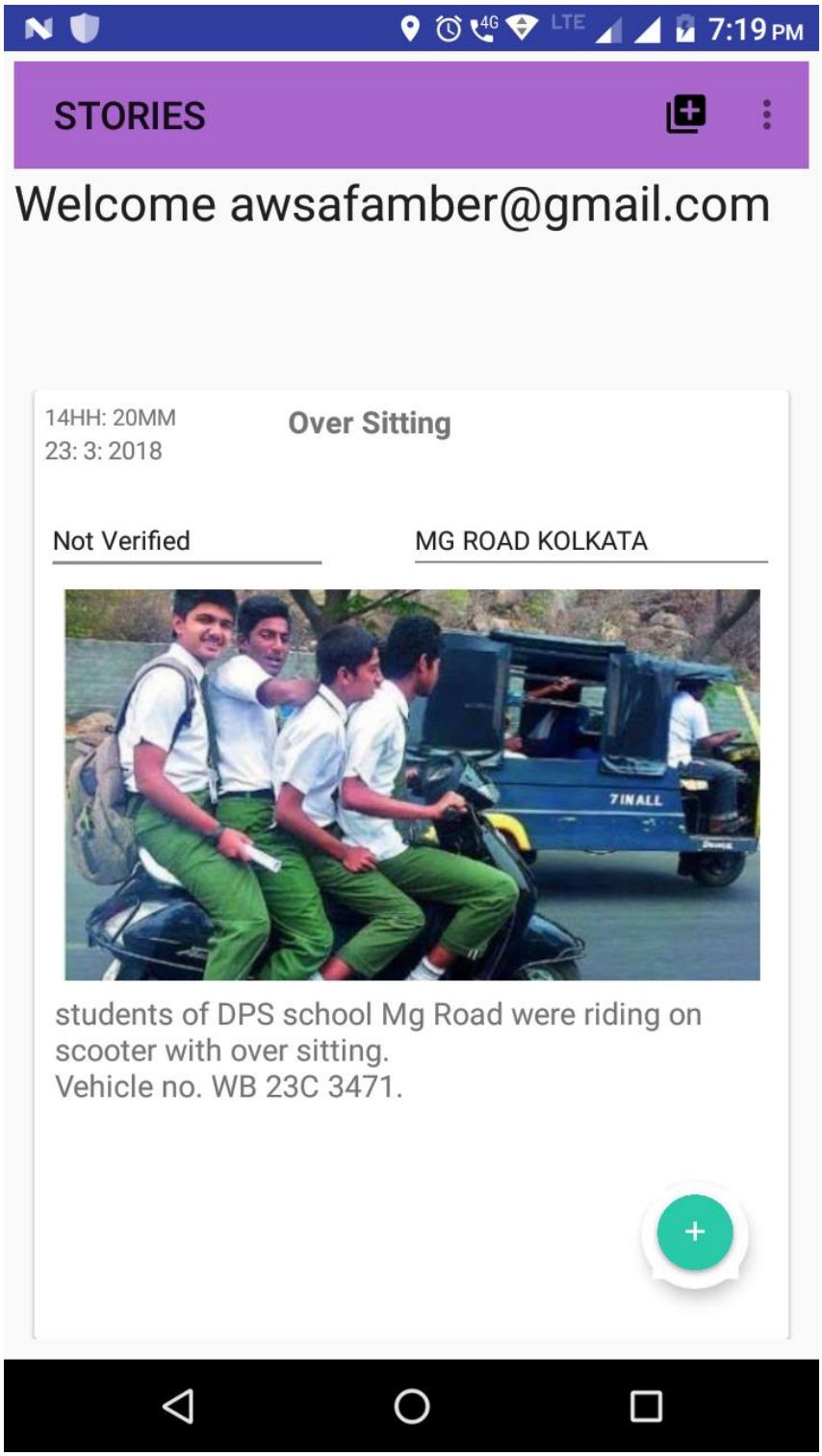



**FIG 23. VIEW CRIME BLOG**

- This is the view of the notification with a message on it. On clicking Main Activity is opened and you can view new added post. Along with that if there is any emergency warning or crime to be highlighted to all a notification is sent. We can send dynamic links on it.
  Notification manager sets the duration of the notification and check for every opened notification by any user.

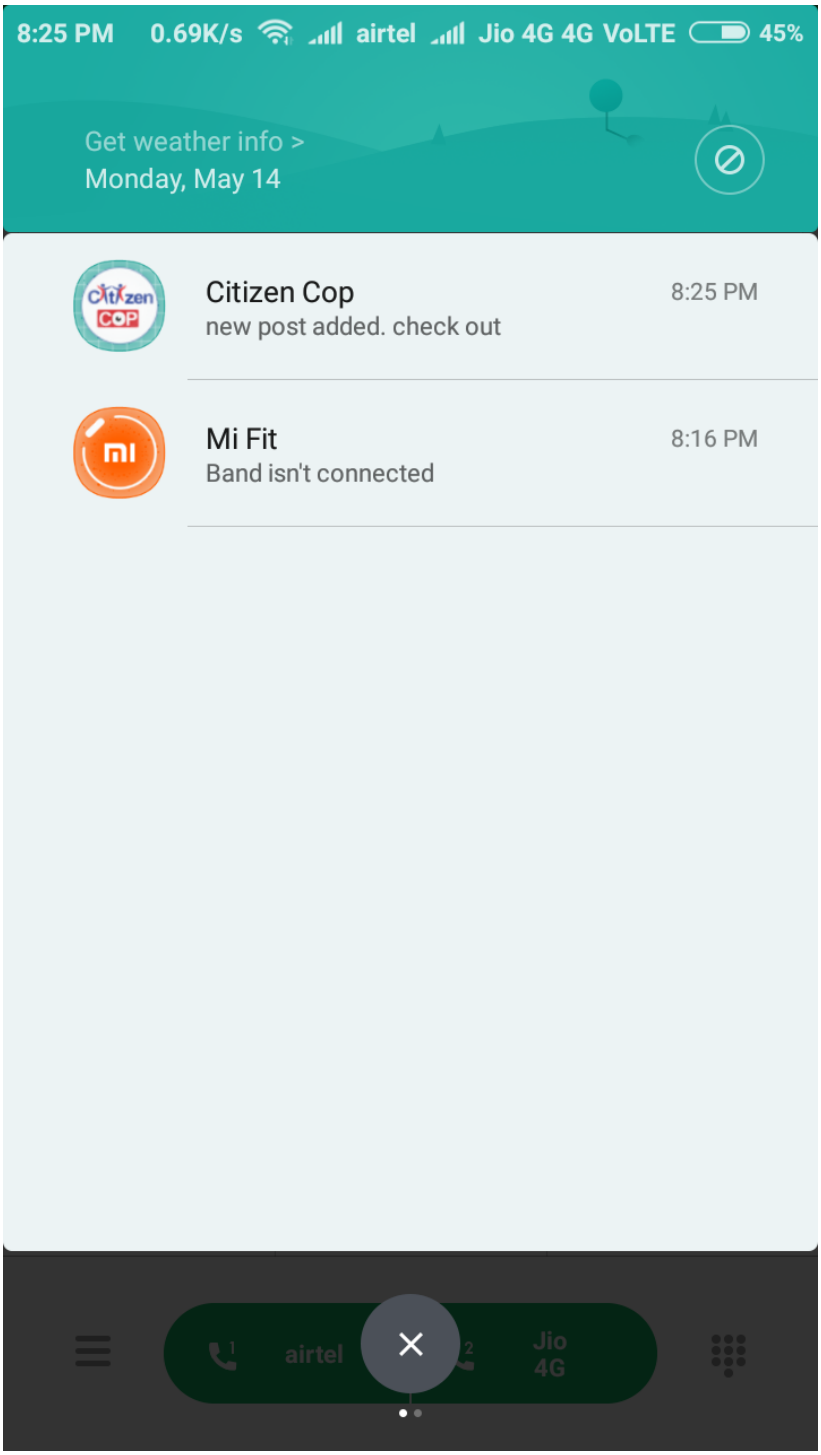



**FIG 24 . NOTIFICATION ALERT**

# PROBLEM ENCOUTERED

## 9.1 PROBLEM FACED

- As the ide of the android studio keep on updating. Newer versions of android studio keep on enrolling and so few functions like “compile” changes to “implementation” and “RECYCLERVIEW” classes’ functions changes very much.
- Firebase and Picasso functions must be integrated with android studio with specific version of android studio.
- We have not developed any websites for our app rather than code whole design with xml for specific app purpose. Casting a website is comparatively easily to host in cloud than any app.

## 9.2 PROBLEM SOLVED

- Android studio tutorials and GitHub helped to understand, how to implement updated functions and classes.
- Proper Documentations for Picasso API and Google firebase helped for integration along with Quora searches related to blogging app.
- We are always okay to make it simple and attractive UI with having proper planning and extra time to manage the changes.

# CONCLUSION AND FUTURE SCOPE

We hereby conclude that our project is successfully completed. The system is made for the benefit of all the citizens wherein, an individual can sign in to post any crime with a picture and a proper title for the same. The admin verifies if the crime happening and the location is authentic or not. If yes, then the cops will be immediately informed. Thus, the title stands justified.

Usually crime takes place uninformed. So, this application in our android devices will help us catch hold of the defaulters and stop crime.

Future scope:

1. We are looking forward to more account creation methods
2. A more authentic and secured platform
3. Getting to upload the app in Google play store

# BIBLIOGRAPHY/REFERENCES


1. https://azure.microsoft.com/en-in/overview/what-is-cloud-computing/
2. https://azure.microsoft.com/en-in/overview/what-is-paas/
3. https://firebase.google.com/docs/test-lab/android/robo-ux-test
4. https://stuff.mit.edu/afs/sipb/project/android/docs/guide/topics/manifest/manifest-intro.html
5. http://square.github.io/picasso/
6. https://github.com/hdodenhof/CircleImageView
7. https://www.youtube.com/playlist?list=PLGCjwl1RrtcR4ptHvrc_PQIxDBB5MGiJA