VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

Mobile Application Development

FACULTY**: Srinivasan P**

STUDENT ATTENDANCE SCANNER

Prateek D	19BIT0229
Vihaan S	19BIT0243
Priyal B	18BIT0272

Introduction

Our project aims to build an efficient system that helps to take attendance. Currently we can see a transition in many parts of the country from online mode to offline mode of education.

But the teachers are still taking manual attendance in places where there was previously fingerprint attendance. We wanted to change this system and implement a application that saves a lot of time while at the same time maintain proper safety protocols.

The issue with the manual attendance way is that teachers on a average spend a minimum of 7-10 mins for taking attendance by calling roll number and verifying the presence of each student in the class. But this impacts highly on a 40 mins class as 10 mins is gone to take attendance, leaving only 30 mins of teaching time for the teacher.

Considering the finger print attendance scanner, it is not a viable solution as a lot of people comes in contact with it and thus increasing the chance of spreading infections

Abstract

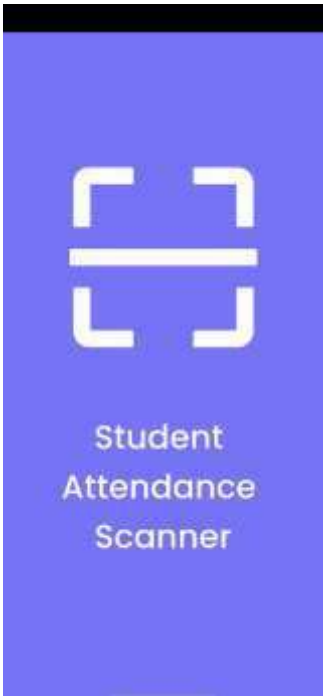
We aim to build an android mobile application that helps in taking attendance in an efficient and a safe manner. The app will have various functions that will help teachers take attendance and store them in a proper way. The teacher needs to login in to her class and open her camera and the students can show their VIT ID card from which their attendance can be scanned. The app will be made using Java and Android SDK. Firebase will be used for providing backend services

The app will also contain additional features such as view attendance history, and options to add new set of classroom and students. Also the login system will require the user to login only once in.

Methods

BASE ALGORITHM

Scan the Barcode in the ID card using the scanner, we get the registration number of the students.



This registration number is compared with a list of registration numbers in the database and if a match is found attendance is marked for the student.

After scanning multiple ID cards we can see the list of students present on that day and we can update this list to a database for future reference.

Demo:

https://drive.google.com/file/d/1E7ADg47t94-OTTNsE9-yGAECWcKde2rQ/view

Authentication

The app uses Firebase Email authentication for providing a secure login experience. The authentication is required so that there is no leak data, and the authentication system also helps in maintaining the database of a particular user accessible to them.

The signup page helps with new users who are joining the application and once the authentication that is the login/signup part is done the user need not perform them again and again when using the app.

Profile

The profile system is mainly used to keep a track the user and his activity. The user has the options to logout or add a new classroom with the set of students in the class.

In this part of the application we get the user name, email and the classroom details all these , that will be used in the rest of the app. In short we use this part to get various user information to provide them a seamless experience when using the app

Attendance History : This part of the app is used to keep track of the attendance taken by the user.

The attendance is stored on the local device in an encrypted format and when everytime the user opens the app they can click on the card to see the attendance details of the particular class. The user also has the option to delete the attendance that is once taken.

Scanning Barcode

The scanning section is used continuously to scan the ID card's barcode and extracting the student registration number from.

After getting the registration number it is then compared with the students list of that particular class to check if the registration number is present. After checking it marks the attendance.

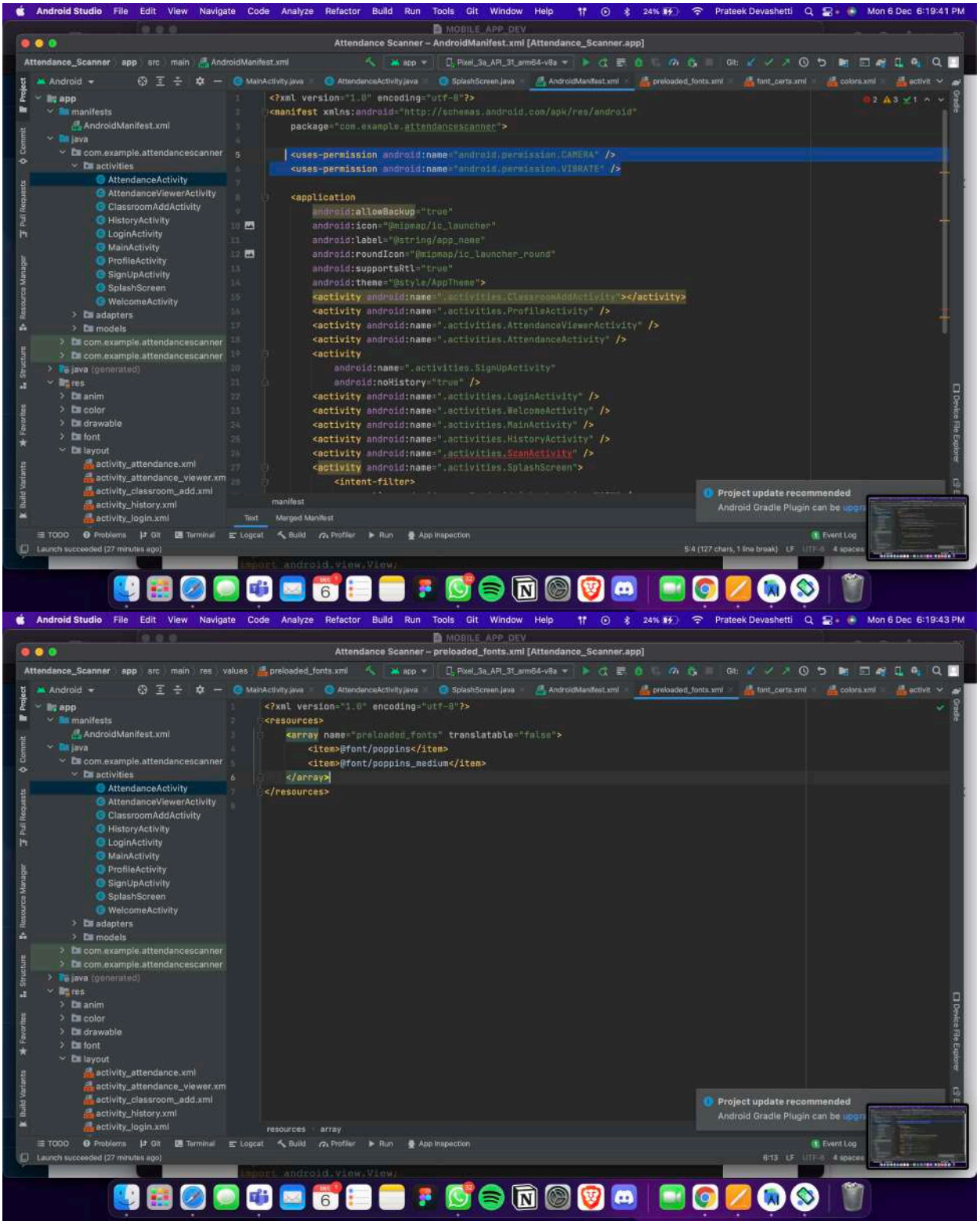
This system doesnot allow any barcode to be scanned and marked for attendance as the value is checked with the database for marking the attendance. We can't also scan the same card twice.

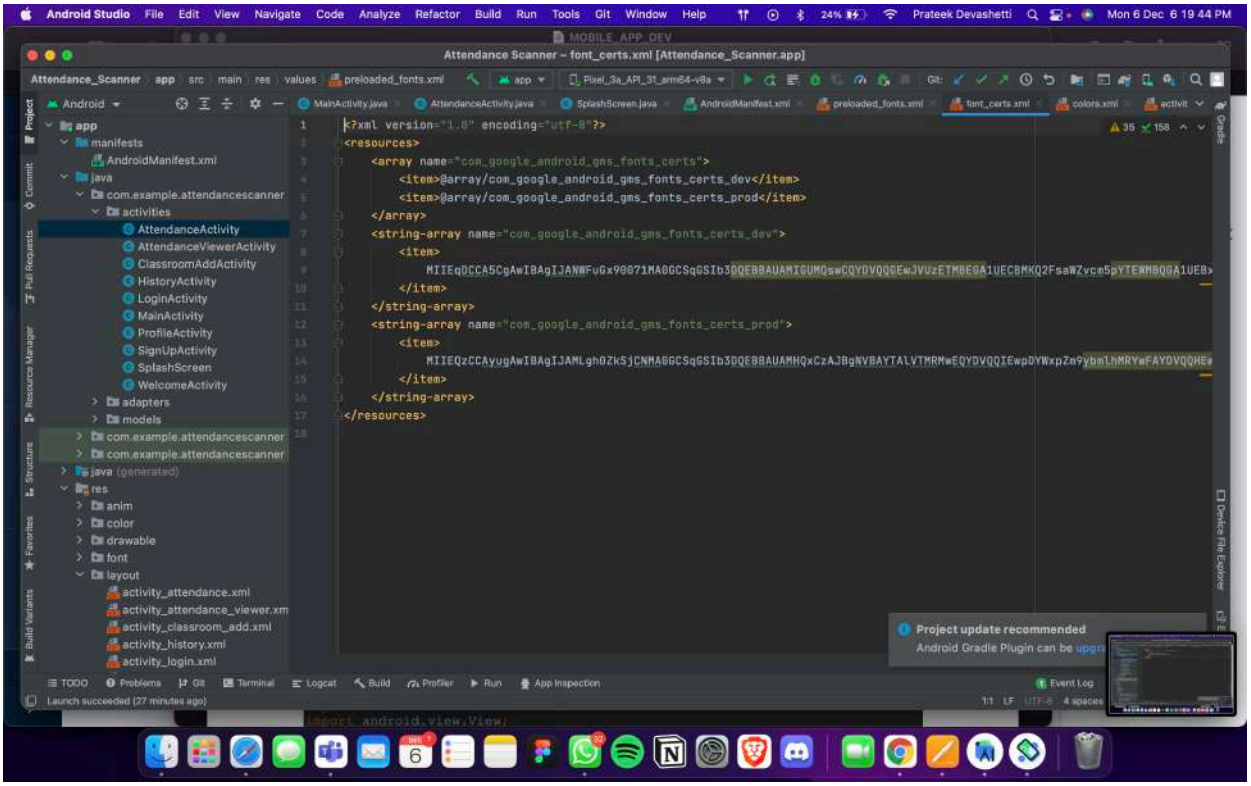
Coding:

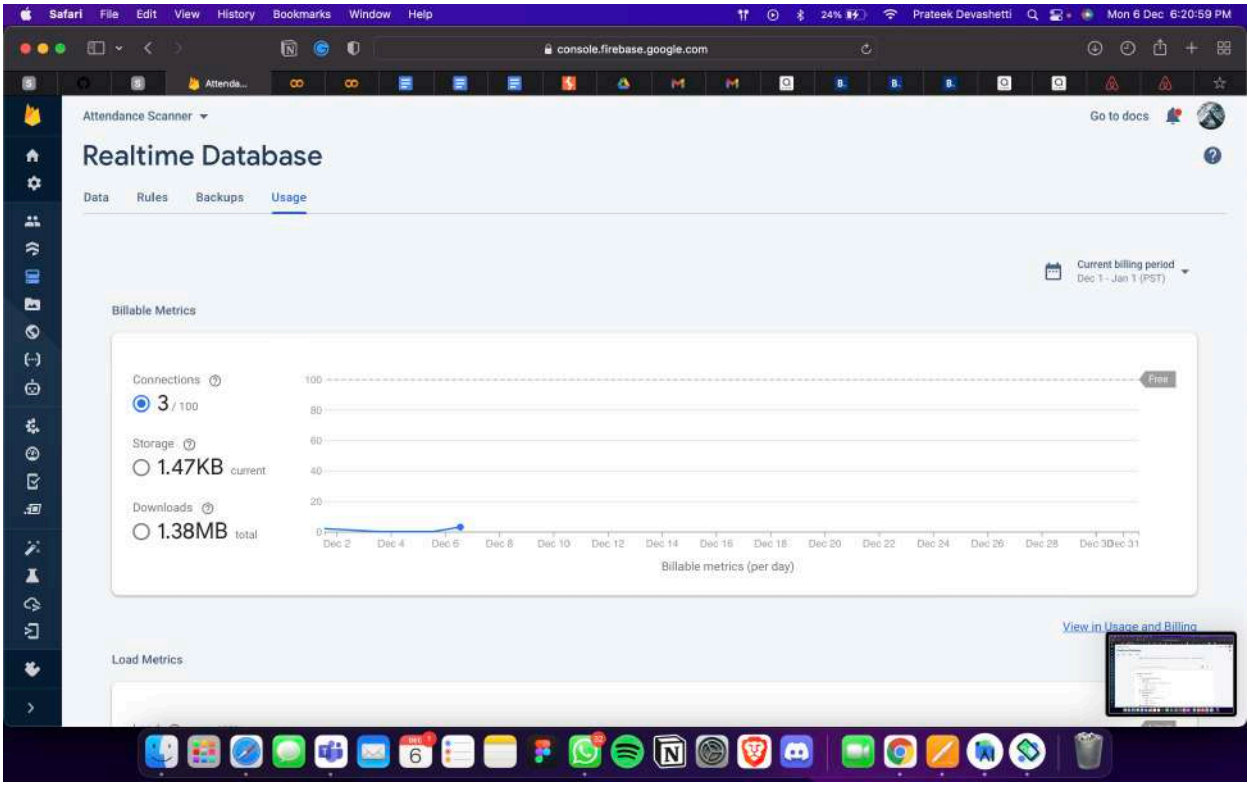
```
package com.example.attendancescanner.activities;

import androidx.appcompat.app.AppCompatActivity;
import androidx.constraintlayout.widget.ConstraintLayout;
import androidx.core.app.ActivityCompat;

import android.Manifest;
import android.content.Context;
import android.content.pm.PackageManager;
import android.os.Build;
import android.os.Bundle;
import android.os.VibrationEffect;
import android.os.Vibrator;
import android.util.Log;
import android.util.SparseArray;
import android.view.Gravity;
import android.view.SurfaceHolder;
import android.view.SurfaceView;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;

import com.example.attendancescanner.R;
import com.example.attendancescanner.models.AttendanceModel;
import com.google.android.gms.vision.CameraSource;
import com.google.android.gms.vision.Detector;
import com.google.android.gms.vision.barcode.Barcode;
import com.google.android.gms.vision.barcode.BarcodeDetector;
import com.google.android.material.bottomsheet.BottomSheetBehavior;
import com.google.gson.Gson;

import java.io.IOException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
import java.util.Locale;

public class AttendanceActivity extends AppCompatActivity {

    SurfaceView cameraView;
    TextView barcodeInfo;
    ListView listView;
    ArrayList<String> arrayList;
    ArrayAdapter arrayAdapter;
    BottomSheetBehavior bottomSheetBehavior;
    ArrayList<String> students_check;


    @Override
    public void onBackPressed() {
        super.onBackPressed();

overridePendingTransition(R.anim.slide_in_left,R.anim.slide_out_right);
    }

    public void onClickTick(View view){
```

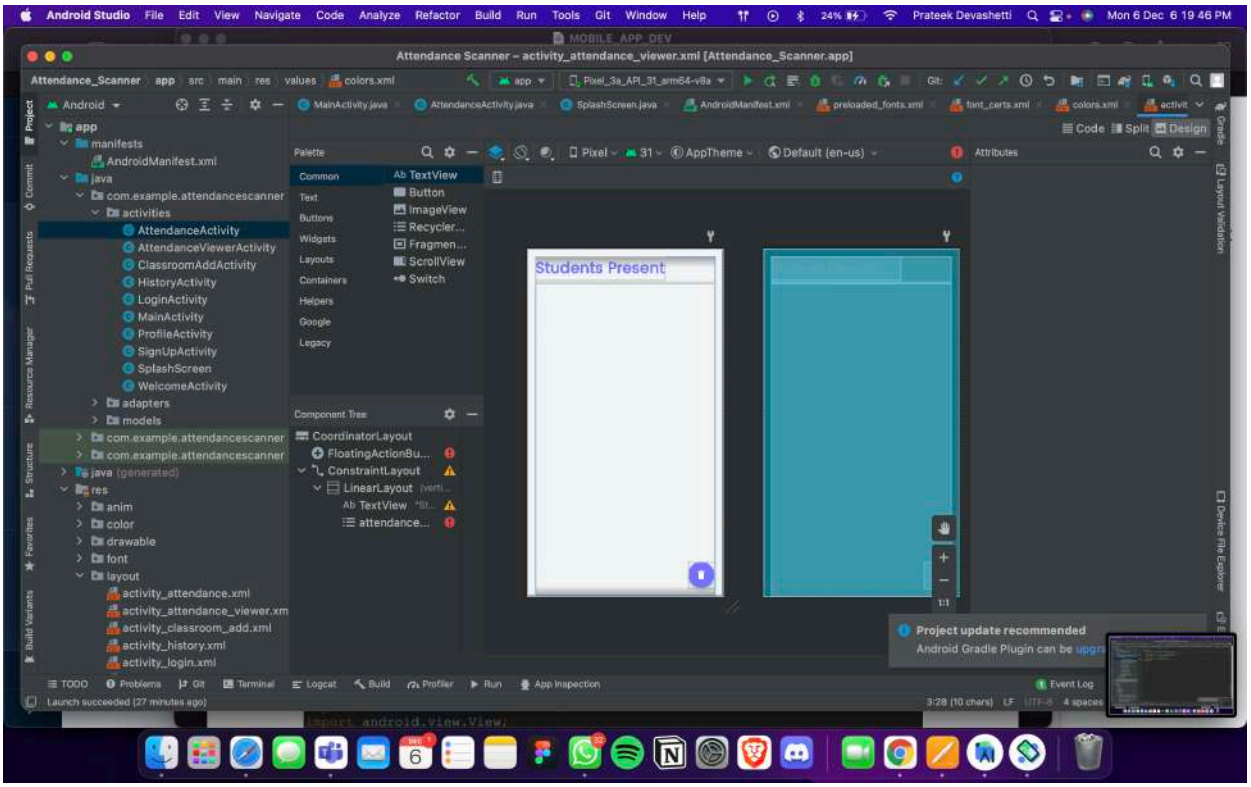
```
if(bottomSheetBehavior.getState()==bottomSheetBehavior.STATE_COLLAPSED){

bottomSheetBehavior.setState(bottomSheetBehavior.STATE_EXPANDED);
        }else{
            if(arrayList.size()>0){
                String date = new SimpleDateFormat("yyyy-MM-dd",
Locale.getDefault()).format(new Date());
                MainActivity.attendanceModels.add(new
AttendanceModel(arrayList,date));
                MainActivity.adapter.notifyDataSetChanged();
                try {
                    Gson gson = new Gson();
                    String response =
gson.toJson(MainActivity.attendanceModels);
MainActivity.sharedPreferences.edit().putString("attendance",response).app
l y();
                    Log.i("add attendance", "attendance: added");
                }catch (Exception e)
                    { e.printStackTrace(
                    );
                }
                MainActivity.recyclerView.setVisibility(View.VISIBLE);
                MainActivity.constraintLayout.setVisibility(View.GONE);
                finish();
            }else{
                Toast.makeText(this,"Attendance list cannot be
empty",Toast.LENGTH_SHORT).show();
            }
        }
    }
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_attendance);
        students_check = new ArrayList<>();
        students_check = (ArrayList<String>)
getIntent().getSerializableExtra("STUDENT_LIST");
        Log.i("CHECK INTENT",students_check.toString());
        cameraView = findViewById(R.id.camera_view);
        barcodeInfo = findViewById(R.id.text_view);
        listView = findViewById(R.id.list_view);
        arrayList = new ArrayList<>();
        arrayAdapter = new ArrayAdapter<>(this,
R.layout.list_item,R.id.list_text_view,arrayList);
        listView.setDivider(null);
        listView.setAdapter(arrayAdapter);
        BarcodeDetector barcodeDetector = new
BarcodeDetector.Builder(this).setBarcodeFormats(Barcode.CODE_128).build();
        final CameraSource cameraSource = new CameraSource.Builder(this,
barcodeDetector).setAutoFocusEnabled(true).setRequestedPreviewSize(1280,720
).setRequestedFps(30).build();
        cameraView.getHolder().addCallback(new SurfaceHolder.Callback() {
            @Override
            public void surfaceCreated(SurfaceHolder holder) {
                try {
                    if
(ActivityCompat.checkSelfPermission(getApplicationContext(),
Manifest.permission.CAMERA) != PackageManager.PERMISSION_GRANTED) {
                        // TODO: Consider calling
```

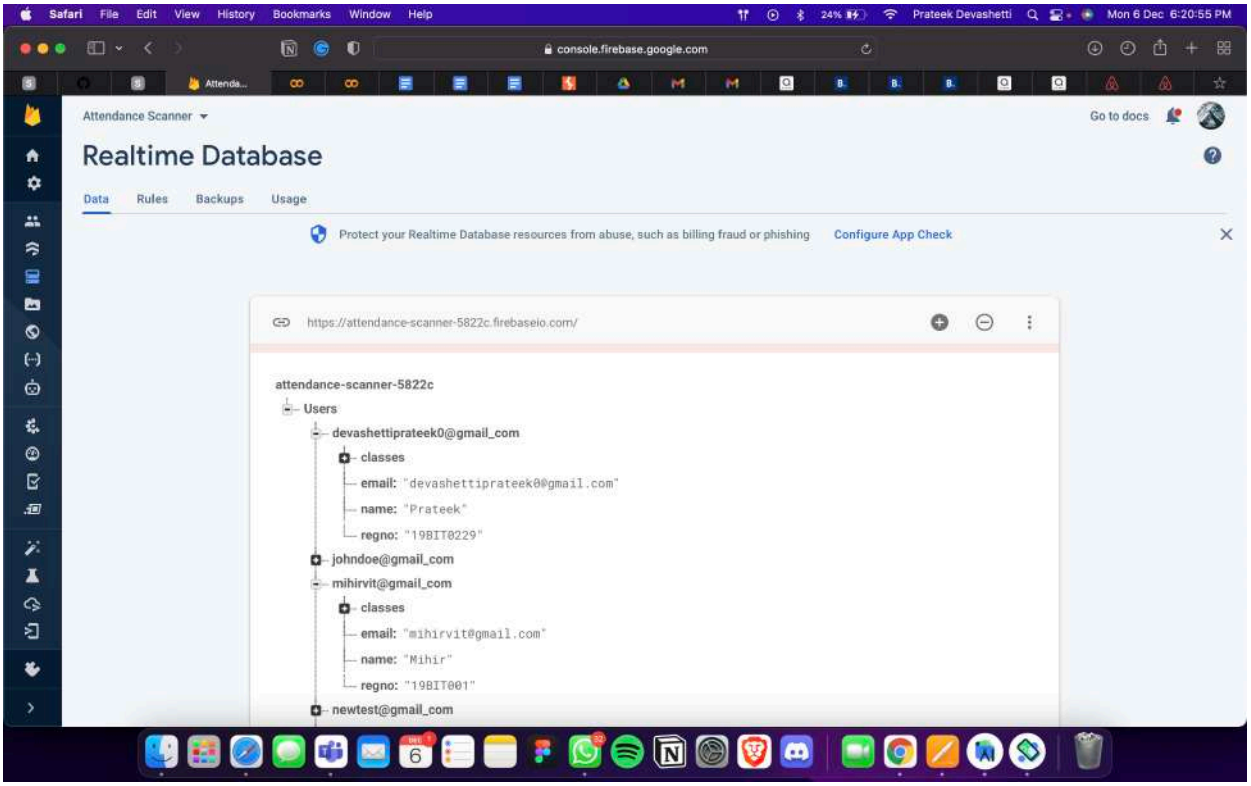
```
                        //    ActivityCompat#requestPermissions
                        // here to request the missing permissions, and
then overriding
                        //   public void onRequestPermissionsResult(int

requestCode, String[] permissions,
                        //                                          int[]
grantResults)
                        // to handle the case where the user grants the
permission. See the documentation
                        // for ActivityCompat#requestPermissions for more

details.
                        return;
                    }

                    cameraSource.start(cameraView.getHolder());
                } catch (IOException ie) {
                    Log.e("CAMERA SOURCE", ie.getMessage());
                }
            }


            @Override
            public void surfaceChanged(SurfaceHolder holder, int format,
int width, int height) {

            }

            @Override
            public void surfaceDestroyed(SurfaceHolder holder) {
                cameraSource.stop();
            }
        });
        barcodeDetector.setProcessor(new Detector.Processor<Barcode>() {
            @Override
            public void release() {

            }

            @Override
            public void receiveDetections(Detector.Detections<Barcode>
detections) {
                final SparseArray<Barcode> barcodes =
detections.getDetectedItems();

                if (barcodes.size() != 0) {

//                    barcodeInfo.post(new Runnable() {    // Use the post
method of the TextView
//                        public void run() {
//
Log.i("barcode",barcodes.valueAt(0).displayValue)
;
//                            barcodeInfo.setText(    // Update the
TextView
//                                    barcodes.valueAt(0).displayValue
//                            );
//                        }
//                    });
                    String name = barcodes.valueAt(0).displayValue;
                    Log.i("NAME IS ", name);
                    if (students_check.contains(name)) {
                        if
(!arrayList.contains(barcodes.valueAt(0).displayValue)) {
                            Vibrator vibrator = (Vibrator)
```

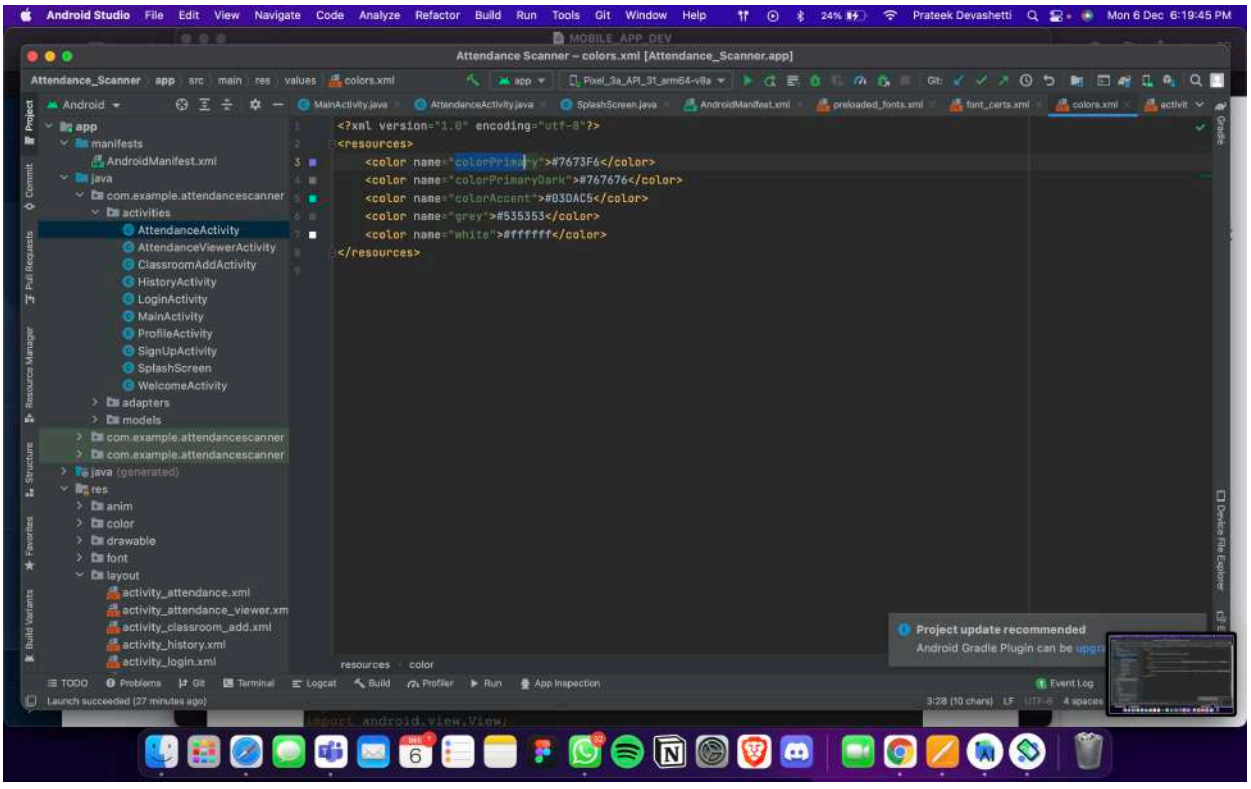

Android Studio File Edit View Navigate Code Analyze Refactor Build Run Tools Git Window Help
Prateek Devashetti
Mon 6 Dec 6:19:41 PM
Attendance Scanner – AndroidManifest.xml [Attendance_Scanner.app]
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
package="com.example.attendancescanner">
<uses-permission android:name="android.permission.CAMERA" />
<uses-permission android:name="android.permission.VIBRATE" />
<application
android:allowBackup="true"
android:icon="@mipmap/ic_launcher"
android:label="@string/app_name"
android:roundIcon="@mipmap/ic_launcher_round"
android:supportsRtl="true"
android:theme="@style/AppTheme">
<activity android:name=".activities.ClassroomAddActivity"></activity>
<activity android:name=".activities.ProfileActivity" />
<activity android:name=".activities.AttendanceViewerActivity" />
<activity android:name=".activities.AttendanceActivity" />
<activity
android:name=".activities.SignUpActivity"
android:noHistory="true" />
<activity android:name=".activities.LoginActivity" />
<activity android:name=".activities.WelcomeActivity" />
<activity android:name=".activities.MainActivity" />
<activity android:name=".activities.HistoryActivity" />
<activity android:name=".activities.ScanActivity" />
<activity android:name=".activities.SplashScreen">
<intent-filter>
Project update recommended
Android Gradle Plugin can be upgr
Android Studio File Edit View Navigate Code Analyze Refactor Build Run Tools Git Window Help
Prateek Devashetti
Mon 6 Dec 6:19:43 PM
Attendance Scanner – preloaded_fonts.xml [Attendance_Scanner.app]
<?xml version="1.0" encoding="utf-8"?>
<resources>
<array name="preloaded_fonts" translatable="false">
<item>@font/poppins</item>
<item>@font/poppins_medium</item>
</array>
</resources>
Project update recommended
Android Gradle Plugin can be upgr


Android Studio File Edit View Navigate Code Analyze Refactor Build Run Tools Git Window Help
Attendance Scanner – font_certs.xml [Attendance_Scanner.app]
app
manifests
AndroidManifest.xml
java
com.example.attendancescanner
activities
AttendanceActivity
AttendanceViewerActivity
ClassroomAddActivity
HistoryActivity
LoginActivity
MainActivity
ProfileActivity
SignUpActivity
SplashScreen
WelcomeActivity
adapters
models
res
anim
color
drawable
font
layout
activity_attendance.xml
activity_attendance_viewer.xm
activity_classroom_add.xml
activity_history.xml
activity_login.xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
<array name="com_google_android_gms_fonts_certs">
<item>@array/com_google_android_gms_fonts_certs_dev</item>
<item>@array/com_google_android_gms_fonts_certs_prod</item>
</array>
<string-array name="com_google_android_gms_fonts_certs_dev">
<item>
MIIEqDCCA5CgAwIBAgIJANWFuGx90071MA0GCSqGSIb3DQEBBAUAMIGUMQswCQYDVQQGEwJVUzETMBEGA1UECBMKQ2FsaWZvcm5pYTEWMBQGA1UE8
</item>
</string-array>
<string-array name="com_google_android_gms_fonts_certs_prod">
<item>
MIIEQzCCAyugAwIBAgIJAMLgh0ZkSjCNMA0GCSqGSIb3DQEBBAUAMHQxCzAJBgNVBAYTAlVTMRMwEQYDVQQIEwpDYWxpZm9ybmlhMRYwFAYDVQQHEw
</item>
</string-array>
</resources>
Project update recommended
Android Gradle Plugin can be upgr
Launch succeeded (27 minutes ago)


Safari File Edit View History Bookmarks Window Help
console.firebase.google.com
Attendance Scanner
Go to docs
Realtime Database
Data
Rules
Backups
Usage
Current billing period
Dec 1 - Jan 1 (PST)
Billable Metrics
Connections
3 / 100
Storage
1.47KB current
Downloads
1.38MB total
Free
Billable metrics (per day)
View in Usage and Billing
Load Metrics

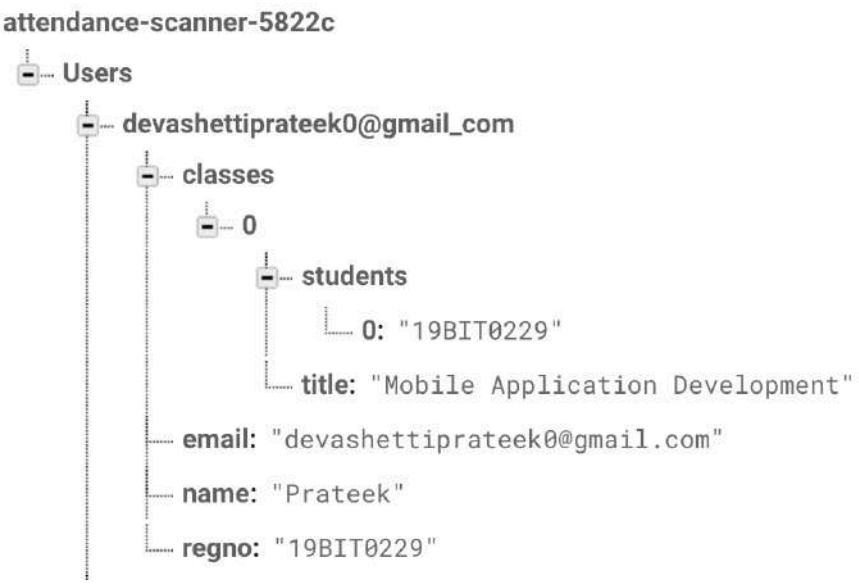

Android Studio File Edit View Navigate Code Analyze Refactor Build Run Tools Git Window Help
Attendance Scanner – activity_attendance_viewer.xml [Attendance_Scanner.app]
Students Present
Project update recommended


Safari File Edit View History Bookmarks Window Help
Prateek Devashetti
Mon 6 Dec 6:20:55 PM
console.firebase.google.com
Attendance Scanner
Go to docs
Realtime Database
Data
Rules
Backups
Usage
Protect your Realtime Database resources from abuse, such as billing fraud or phishing
Configure App Check
https://attendance-scanner-5822c.firebaseio.com/
attendance-scanner-5822c
Users
devashettiprateek0@gmail_com
classes
email: "devashettiprateek0@gmail.com"
name: "Prateek"
regno: "19BIT0229"
johndoe@gmail_com
mihirvit@gmail_com
classes
email: "mihirvit@gmail.com"
name: "Mihir"
regno: "19BIT001"
newtest@gmail_com


Android Studio File Edit View Navigate Code Analyze Refactor Build Run Tools Git Window Help
Prateek Devashetti
Mon 6 Dec 6:19:45 PM
Attendance Scanner – colors.xml [Attendance_Scanner.app]
<?xml version="1.0" encoding="utf-8"?>
<resources>
<color name="colorPrimary">#7673F6</color>
<color name="colorPrimaryDark">#767676</color>
<color name="colorAccent">#03DAC5</color>
<color name="grey">#535353</color>
<color name="white">#ffffff</color>
</resources>
Project update recommended
Android Gradle Plugin can be upgr

```
getSystemService(Context.VIBRATOR_SERVICE);
                            if (Build.VERSION.SDK_INT >=
Build.VERSION_CODES.O) {

vibrator.vibrate(VibrationEffect.createOneShot(500,
VibrationEffect.DEFAULT_AMPLITUDE));
                            } else {
                                //deprecated in API 26
                                vibrator.vibrate(500);
                            }

arrayList.add(barcodes.valueAt(0).displayValue);
                            listView.post(new Runnable() {
                                @Override
                                public void run() {
                                    arrayAdapter.notifyDataSetChanged();
                                }
                            });
                        }
                        else{
                        }
                    }
                    else {
                    }
                }
            }
        });
        ConstraintLayout bottomSheetLayout =
findViewById(R.id.bottom_sheet_layout);
        bottomSheetBehavior = BottomSheetBehavior.from(bottomSheetLayout);
    }

}
```

Database



Conclusion:

From our output we can draw the conclusion that we have built a efficient and a new alternative system to scan attendance. The attendance system is highly securable as we use the barcode present in the ID cards storing it in a private database. The system is very reliable as it scans the bar code present in the ID card of the user. Since each user has a unique barcode over their ID card the overall operations become easier and more reliable.

The use of the app can be extended from teachers to event organisers, hostel wardens and exam invigilators. Also as it currently only scans VIT ID cards for the future work we can make it to scan other college ID cards or any ID card with a barcode in it.

Reference: https://

developers.google.com/ml-kit

https://developers.google.com/ml-kit/vision/barcode-scanning

https://firebase.google.com/docs/database/android/start

https://firebase.google.com/docs/android/setup https://

developer.android.com/guide/topics/media/camera https://

www.scnsoft.com/blog/software-development-models