PROJECT REPORT ON

Pocket College App

Developed by

Group Members	Seat No.
Rohit Ahuja	125403
Kunal Hemnani	125416
Tarun Pahilajani	125430
Tejas Pundpal	125433

UNDER THE GUIDANCE OF

Mrs. Shilpa Ingoley

Department of Computer Technology

S.H.M Institute of Technology

Ulhasnagar-421003

Academic Year: 2015-2016

Certificate

This is to certify that Rohit Ahuja, Kunal Hemnani, Tarun Pahilajani and Tejas Pundpal, the students of Third Year Computer Technology have completed their Major project on College Android app called as “POCO (Pocket College)” for Android in the academic year 2015-2016, as a partial fulfillment of the Diploma course in engineering/technology as prescribed by the Maharashtra State Board of Technical Education (MSBTE), Mumbai for academic year 2015-2016.

And I have accessed the said work and I am Satisfied that the name is up to standard for the level of course.

And, said work may be presented to the External Examiner.

Date: _____ _____

(Guide’s Signature)

_____ _____

(HOD’s Signature) (Principal’s Signature)

_____ _____

(Internal Examiner’s Signature) (External Examiner Signature)

Submission

We,

Mr. Rohit Ahuja	1301120104
Mr. Kunal Hemnani	1301120118
Mr. Tarun Pahilajani	1301120133
Mr. Tejas Pundpal	1301120138

The Student of Third Year of Computer Technology branch from SHMIT Institute accept that you have submitted the Detailed Project Report POCO (Pocket College) and that, the following student(s) were associated with this work.

And, that we have not copied the report or any of its

appreciable part from any other literature in contravention of the academic's ethics.

Date: _____

Name: Signature:

Rohit Ahuja

Kunal Hemnani

Tarun Pahilajani

Tejas Pundpal

Acknowledgement

We would like to express our gratitude to all those who gave us the possibility to undertake our Project for final year POCO (Pocket College) App for Android.

We would like to take this opportunity to express our deep sense of gratitude towards all of them who spanned their time, energy and intelligence towards the beautification of our project. This justification will never sound good if we don't express our vote of thanks to **Prof. Shilpa Ingoley** who provided us innumerable guidance and even extended their support in execution and preparation of our Project. They motivated us and gave their suggestions to add new features in our app to make it look better and function according to the needs of user.

There are times in projects when your clock beats your time again and again and you run of the time and dangling to complete the project. Our Head of Department **Mr. D.V. Chawak** and Principal **Mr. A.B. Patil** made us endure such times with their unfailing support and help.

Really it is highly impossible to repay the debt of all the people who have directly or indirectly helped us for performing this project.

Index

INTRODUCTION

POCO (Pocket College) is utility app made for the S.H.M.I.T college students and staff. It is small and a sober Android app free from complexities and technical jargons. The main aim of POCO is to provide students an easy access to the Time table, Exam Question Papers, Notes, Notices of the college. It is an endeavor to make such a generic app that can be used in a wide range of technical institutes.

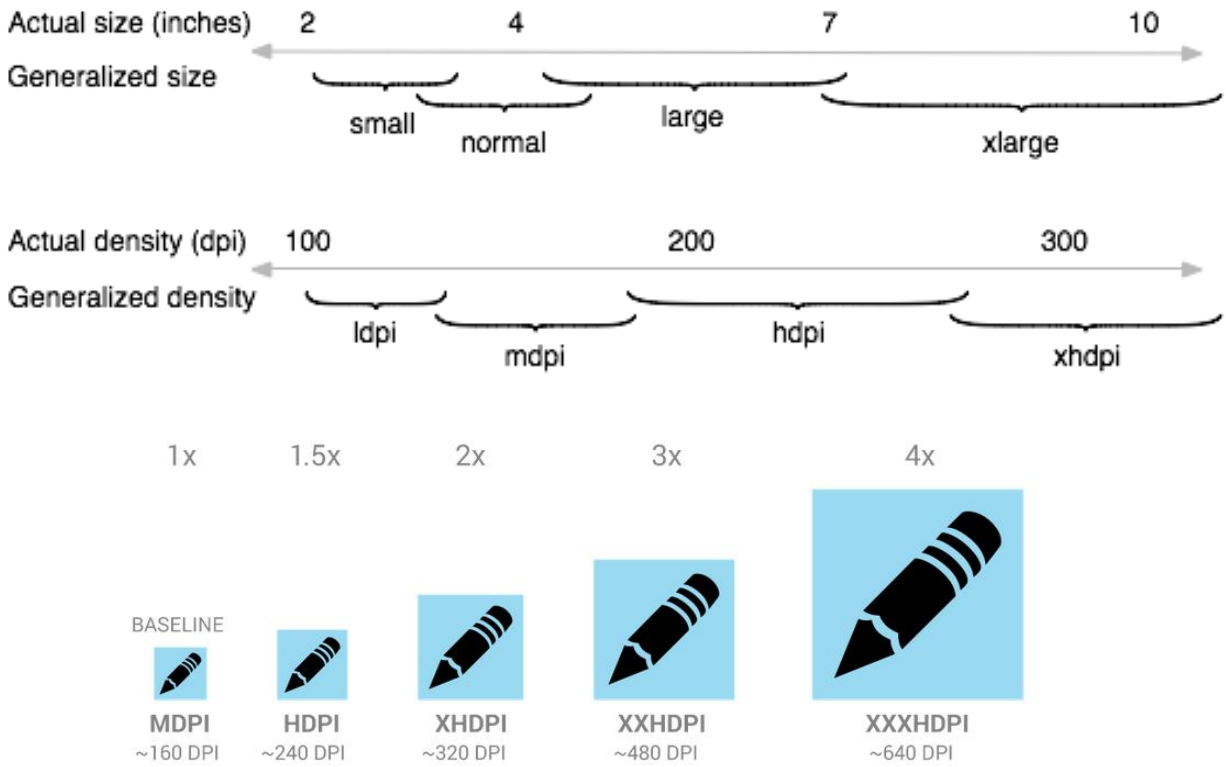
Why we named it Poco? - The name POCO is defined to app because it provides users push notification for tests, new timetable uploads, Question Papers, information about the Faculty and Dashboard Notices, Upload a Notice, Anywhere & Anytime.

The app is very easy to use and is as per student's requirement. This app allows user to get all the data for any of the classes present within its database. User can view Timetable of their respective class. Users can view Question papers of their respective year and semester. As, I mentioned above, users can even upload a notice in this app. Once user uploads a notice, the notice is uploaded in the database and that notice can be viewed by Administrator only. If the Administrator verifies that the uploaded notice is correct, He/she will display that uploaded notice in the Dashboard Notices Section. If the notice is not correct/genuine, then it will be rejected and will not be displayed in the Dashboard Notice Section.

The main highlights of Pocket College are its simple and clean interface that pleases the eye. The app tries to cope up with all the existing base design principles of the Android Architecture.

Hardware Aspects: The app Pocket College is a lightweight app that resides within the internal memory of user's Android cell phone. Theirs is also an option to move it to SD Card to free up the internal space.

The app renders itself quite comfortably on Android phones of various screen densities. The following image gives us an idea of segmentation terminology used within the Android universe to differentiate between different screen rendering specifications in devices.



Software Aspects: Pocket College is made for Android Devices that operate on Android Version 4.1.1 i.e. API Level 15 and above.

It harnesses custom Views i.e. UI controls of the Android architecture itself and also some third party custom views. Since it is an Android app it is written in java for obvious reasons and also includes XML for data transfer and layout rendering. It doesn't require any huge graphic support to be present on the user's phone to work.

Pocket College uses Parse backend server for Database. As all the Question Papers, Time table, Notices rely on the Parse.com Database. Question papers are uploaded at Parse.com SDK and through Google Docs user can view Question Paper PDF easily. Parse.com server is free for every developer, there is no need to download any external third party software/app to fully utilize the functionalities of Pocket College.

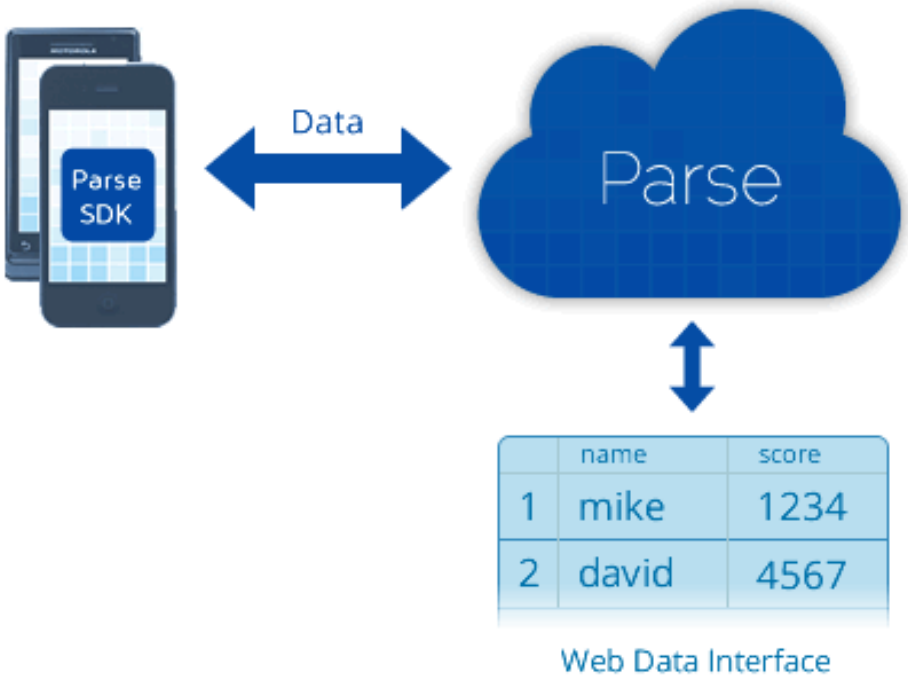
What is Parse?

Parse is a Backend as a Service providing company which was acquired by Facebook in 2013. In other words, they provide you with an application development platform in the cloud. BaaS lets you focus more on the frontend and enhance user experience as the backend is being taken care of by the BaaS provider.

Parse offer a backend to store data, push notifications, social media integration for your app etc.

The features provided tend to be helpful in prototyping quickly. Features as iterated by Parse:

- Parse Core
- Parse Push
- Parse Social
- Parse Analytics
- Parse Hosting
- Cloud Code



PROJECT OBJECTIVES AND SCOPE

Going College is a routine for all the students out there. It's kind of activity that has to be performed for the studies, examination and Student's future scope. Don't you think that many students don't get informed / get confused about Examination timings. Students have to visit College again and again to view any new Important Notice displayed on the Notice board. Students need to search for Question papers here and there. No doubt, It's job of the Students to visit college and get Noticed, but don't you think this job can be done in easy way?

In this age of technological advancement, it is a shocker to see that no efforts are made to normalize this procedure and make at least not look dull and burdensome. There are institutes that harness the new age technology for the same like, providing Timetable, Question papers, Notices directly on Laptops, PDA's or app provided to students by the institution itself. The problem with this approach is that it is a costly affair for the institute and that too for a work that is not considered as a tiresome one. So, it is used on a very small scale in some countries that too where the institutes are self-sufficient to go out of their way to provide these kind of peripheral facilities to their students. It would be great to be able to find a way to make it work for both the institutes and students.

Pocket College, does the same for every student out there. It is an attempt to digitize the hectic procedure of visiting college again and again for Notices, searching for Question Papers, etc. It utilizes the **Android Platform** that is famous and used all around the world each genre of users possible – from small kids and grown-ups to adults and senior citizens. And since there is still no perfect alternative available to Android users for this purpose on the Google Play Store, it would be great to be providing this app as a utility to the needful and capable.

It can possibly be seen as a non-costly affair for the institutions due to the extent to which Android is used today; it can be implemented at a large scale in many places.

The aim is- providing a reasonable solution that could work without the institutions having to go out of way, also to provide students an easy access to the Time table, Exam Question Papers, Notes, Notices of the College; so **Pocket College will be available for FREE** on the **Play Store.** This would in effect, increase the scope of this project and will help the developers in accomplishing their goal. Since it will be available for free to every user, it would act as catalyst and might get implemented in various places. This **Android application** can be readily used by non-programming personal avoiding human handled chance of error.

This **App** has feature that are:

- **Simple** app that is quite easy to use and needs no time to get along with.
- Stays within **Pocket.**
- **Timetable**(FY-SY-TY)
- **Notes** and **Question-Papers** (PDF Format-Students can download the notes easily).
- **Notice (Important Notices-**E.g. Examination timings, Question Bank**)**.
- **About the college** (History, Faculty, Working hours).
- **Dashboard Notices** (Any important notice/Image will be displayed here).
- **Push Notifications** (User can directly receive Push Notifications on their Android Phone).
- **Upload a Notice** (User can upload a notice).
- **Data** is stored directly on the **Server.**

The objectives of this **App** are: -

- To reduce **paperwork.**
- Increased **accuracy and reliability**.
- Data **security.**
- Real-time Information Publishing through **Push** Noti.
- Saves time of **Students and Teachers.**

ANDROID OS AND THE SOFTWARE STACK

ANDROID OS

Android is an operating system based on the Linux Kernel, and designed primarily for touchscreen mobile devices such as smartphones and tablet computers. Initially developed by android, Inc., which Google backed financially and later bought in 2005. Android's source code is released by Google under the Apache License this permissive licensing allows the software to be freely modified and distributed by device allows the software to be freely modified and distributed by device manufactures, wireless carriers and enthusiast developers. Most Android devices ship with a combination of open source and proprietary software.

As of May 2012, Android became the most popular mobile OS, having the largest installed base, and is a market leader in most countries including the United States; there it has had the highest installed base of mobile phone for years. Android is popular with technology companies which require a ready-made, low-cost and customizable operating system for high-tech devices. Despite being primarily designed for phones and tablets, it also has been used in televisions, games consoles, cameras & other electronics.

Android app programming is basically a mixture of pure java and XML concepts. The designing and lay outing is done via XML

Whereas the dynamic lay outing and programming stuff is made with the help of java Language. These is a special API made for the app developers, to work on. The packages in the API are already present in every Android device that is released. Android updates its' API quite a lot frequently while releasing the new Android versions. Android apps support forward compatibility, by default. Even the backward compatibility is possible, but, with the help of support libraries which is to be included in the setup file of an app.

Android software development is the process by which new apps are created for the android OS. Apps are usually developed in the java programming language using the Android Software Development Kit, but other development tools are available, which wrap up long-lasting things in seconds.

JAVA AS A SOFTWARE PLATFORM

Java is a set of several computer software products and specifications from sun microsystems (which has since merged with oracle Corporation), that together provides a system for developing applications software and deploying it in a cross-platform computing environment. Java is used in a wide variety of computing platforms from embedded devices and mobile phones on the low end, to enterprise servers and supercomputers on the high end. While less common, java applets are sometimes used to provide improved and secure

functions while browsing the World Wide Web on desktop computers.

Writing in the java programming language is the primary way to produce code that will be deployed as java bytecode. There are, however, bytecode compilers available for other languages have been designed to run natively on the java Virtual Machine (JVM), such as Scala, Clojure and Groovy. Java syntax borrows heavily from C and C++, but object-oriented features are modelled after Smalltalk and Objective-c. Java eliminates certain low-level constructs such as pointers and has a very simple memory model where every object is allocated on the heap and all variables of object types are references. Memory management is handled through integrated automatic garbage collection performed by the JVM. On November 13, 2006, Sun Microsystems made the bulk of its implementation of java available under the GNU General Public License (GPL).

JAVA AS A PROGRAMMING LANGUAGE

Java is a computer programming language that is concurrent, class based, object-oriented, and specifically designed to have as few implementation dependencies as possible. It is intended to let app developers “write once, run anywhere” (WORA), meaning that code that runs on one platform does not need to be recompiled to run on another. Java applications are typically compiled to byte code (class file) that

can run on any java virtual machine (JVM) regardless of computer architecture. Java is, as of 2014, one of the most popular programming languages in use, particularly for client-server web apps, with a reported 9 million developers .Java was originally developed by James Gosling at Sun Microsystems (which has since merged into Oracle Corporation) and released in 1995 as a core component of Sun Microsystems' java platform. The language derives much of its syntax from C and C++, but it has fewer low-level facilities than either of them.

The original and reference implementation java compilers, virtual machines, and class libraries were developed by Sun from 1991 and first released in 1995. As of May 2007, in compliance with the specification of the java community process, Sun relicensed most of its java technologies under the GNU General Public License. Others have also developed alternative implementations of these Sun technologies, such as the GNU Compiler for java (byte code compiler), GNU Class path (standard libraries), and Iced Tea-Web (browser plugin for applets).

XML

Extensible Mark Up Language (XML) is a Mark Up language that defines a set of rules for encoding documents in a format that is both human readable and machine-readable. It is defined in the XML 1.0 Specification produced by the W3C, and several other related specifications, all free open standards.

The design goals of XML emphasize simplicity, generality, and usability over the internet. It is a textual data format with strong support via Unicode for the languages of the world. Although the design of XML focuses on documents, it is widely used for the representation of arbitrary data structures, for example in web services. Many application programming interfaces (APIs) have been developed to aid software developers with processing XML data, and several schema systems exist to aid in the definition of XML-based languages.

SOFTWARE DEVELOPMENT KIT

The Android software development kit (SDK) includes a comprehensive set of development tools. These include a debugger, libraries, a handset emulator based on QEMU, documentation, sample code, and tutorials. Currently supported development platforms include computers running Linux (any modern desktop Linux distribution), Mac OS X 10.5.8 or later; for the moment one can develop Android software on Android itself by using [AIDE – Android IDE – Java, C++] app and [Android java editor] app. The officially supported integrated development environment (IDE) is Eclipse using the Android Development Tools (ADT) Plugin, though IntelliJ IDEA IDE (all editions) fully supports Android development out of the box, and NetBeans IDE also supports Android Development kit via a plugin. Additionally, developers may use any text editors to edit java and XML files, then use command line tools (java Development Kit and Apache

Ant are required) to create, build and debug Android apps as well as control attached Android devices (e.g., triggering a reboot, installing software package(s) remotely).

Android apps are packaged in. apk format and stored under /data/app folder on the Android OS (the folder is accessible only to the root user for security reason). APK package contains. dex files (compiled byte code files called Dalvik executables), resource files, etc.

ANDROID STUDIO

Android studio is a new Android development environment based on IntelliJ IDEA. Similar to Eclipse with the ADT Plugin, Android Studio provides integrated Android developers tools for development and debugging. On top of the capabilities you expect from IntelliJ, Android Studio offers:

- Gradle-based build support.
- Android-specific refactoring and quick fixes.
- Lint tools to catch performance, usability, version compact & other problems.
- ProGuard and app-signing capabilities.
- Template-based wizards to create common Android designs and components.

GENY MOTION

Geny motion is the next generation of the AndroVM open source project, already trusted by 900000 developers. It's even easier to use and has lots more functionalities. It is an emulator using x86 architecture virtualization, making it much more efficient! Talking advantages of OpenGL hardware acceleration, it allows one to test his/her apps with amazing 3D performance. It perfectly integrates in the development environment with Eclipse plugin. It is packaged for windows, Mac and Linux.

GenyMotion Features:

- Easily download and run pre-configured virtual images covering a range of android versions from2.x onwards, and various phone and tablet screen sizes
- Networking: Ethernet (emulates Wi-Fi connection)
- GPS (with configurable coordinates) and battery (with configurable battery levels) emulation widgets
- Display: OpenGL hardware acceleration, multiscreen, full screen display
- GenyMotion shell which allows you to interact with your VM using a command line
- ADB support
- Eclipse and android studio plugins
- Supports Linux, windows and mac
- "Drag & Drop" apk installs
- "Drag & Drop" zip support for system updates/ patches

MODELLING DIAGRAMS

Modeling diagrams basically help you understand, clarify, and communicate ideas about your code and the user requirements that your software system must support. For example, to describe and communicate user requirements, you can use Unified Modeling Language (UML) use case, activity, class, and sequence diagrams. To describe and communicate the functionality of your system, you can use UML component, class, activity, and sequence diagrams.

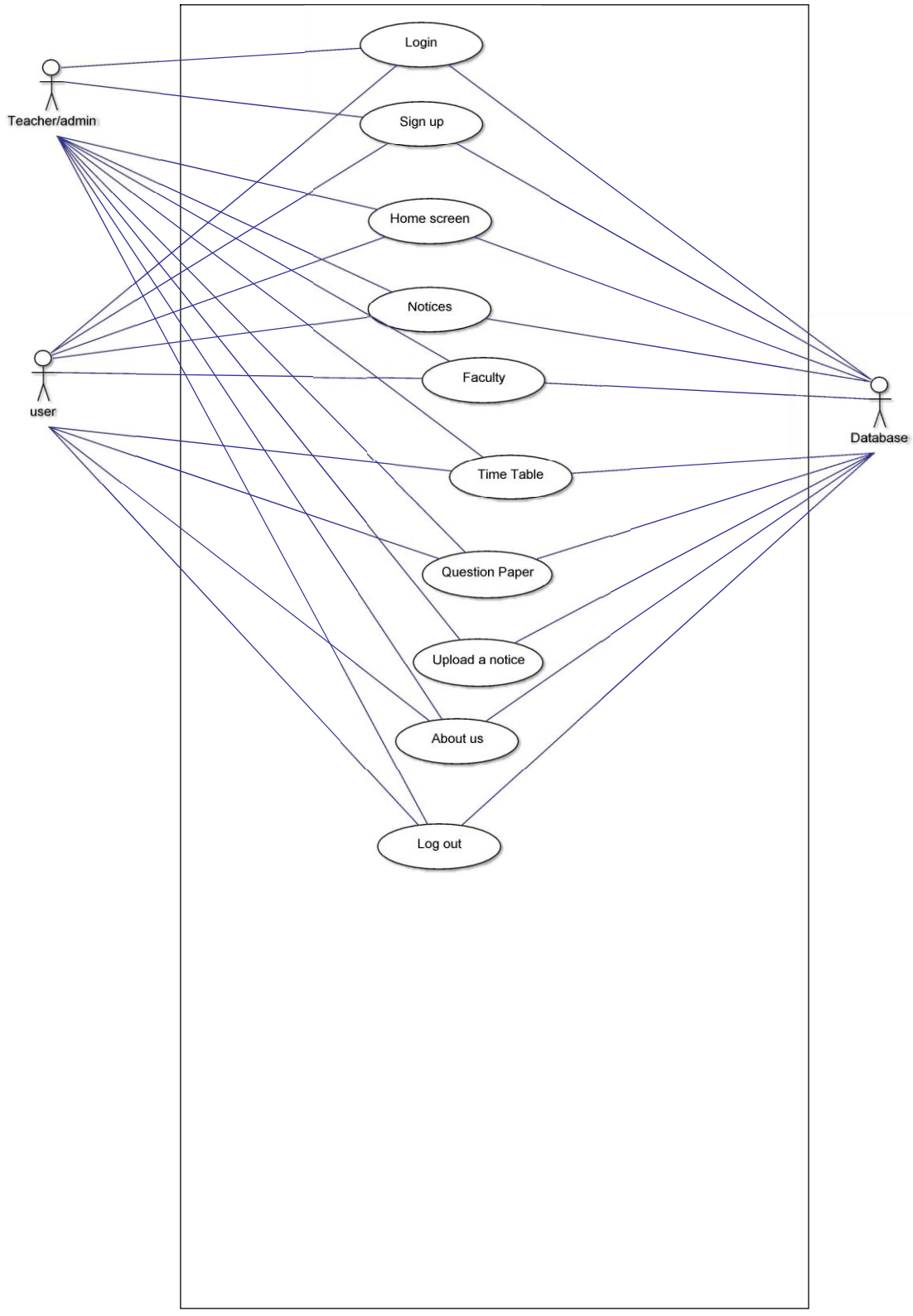
Whereas, the Unified Modeling Language (UML) is a general-purpose modeling language in the field of software engineering. It provides a set of graphic notation techniques to create visual models of object -oriented software -intensive systems. It was developed by Grady Booch, Ivar Jacobson and James Rambaughat Rational Software in the 1990s. It was adopted by the Object Management Group (OMG) in 1997, and has been managed by this organization ever since. In 2000 the Unified Modeling Language was accepted by the International Organization for Standardization (ISO) as a standard for modeling software -intensive systems.

There is a rich -variety for a developer to choose among the two types of UML diagrams, viz. Static diagrams like the class diagram and the Dynamic Diagrams like the sequence diagram. The Pocket College app has been outlined with the help of four different UML diagrams.

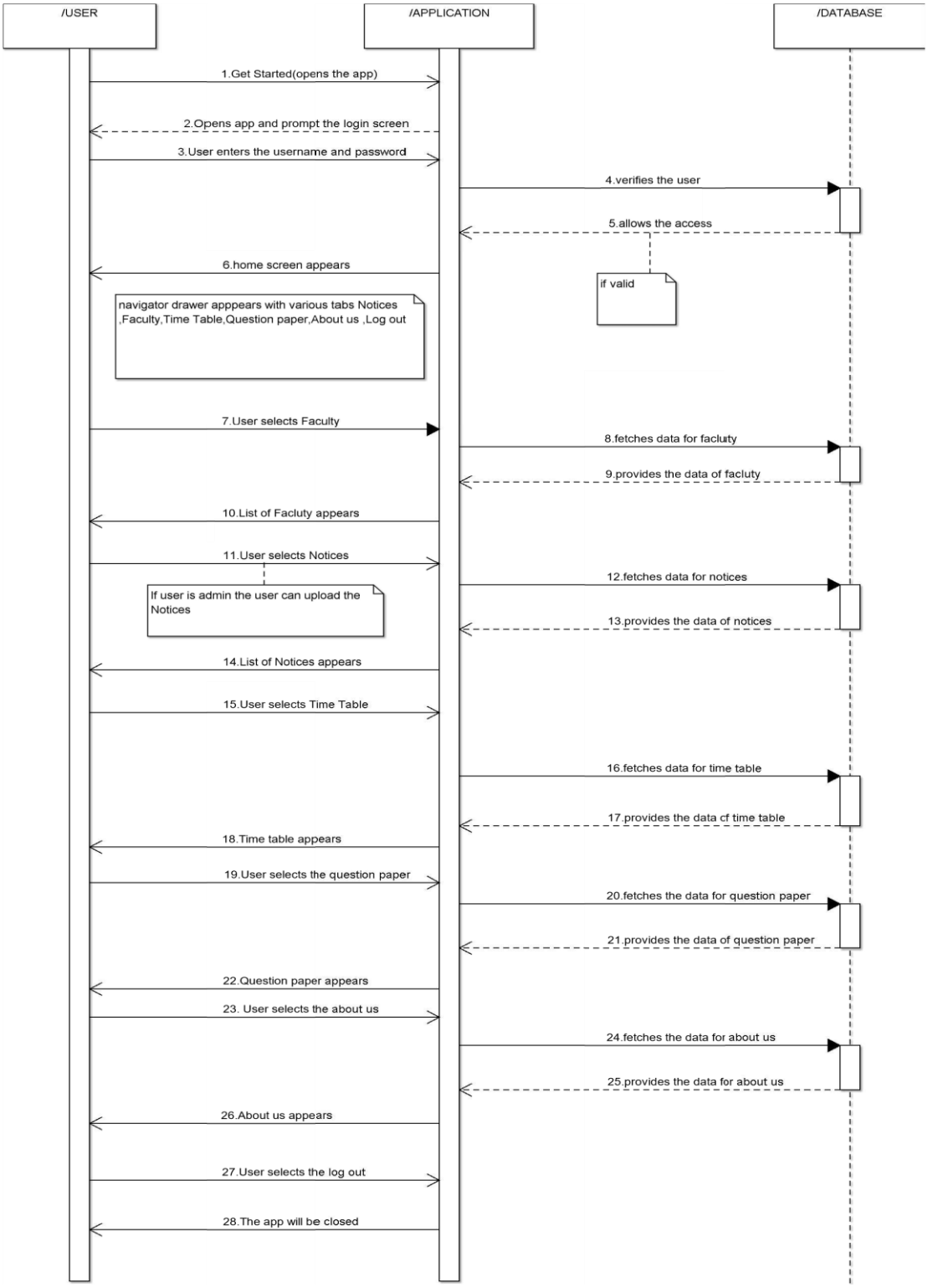
- Use -case diagram highlighting various actors involved
- Sequence diagram emphasizing the operation process

- Component Diagram
- Collaboration Diagram
- Data Flow Diagram
- Flow Diagram

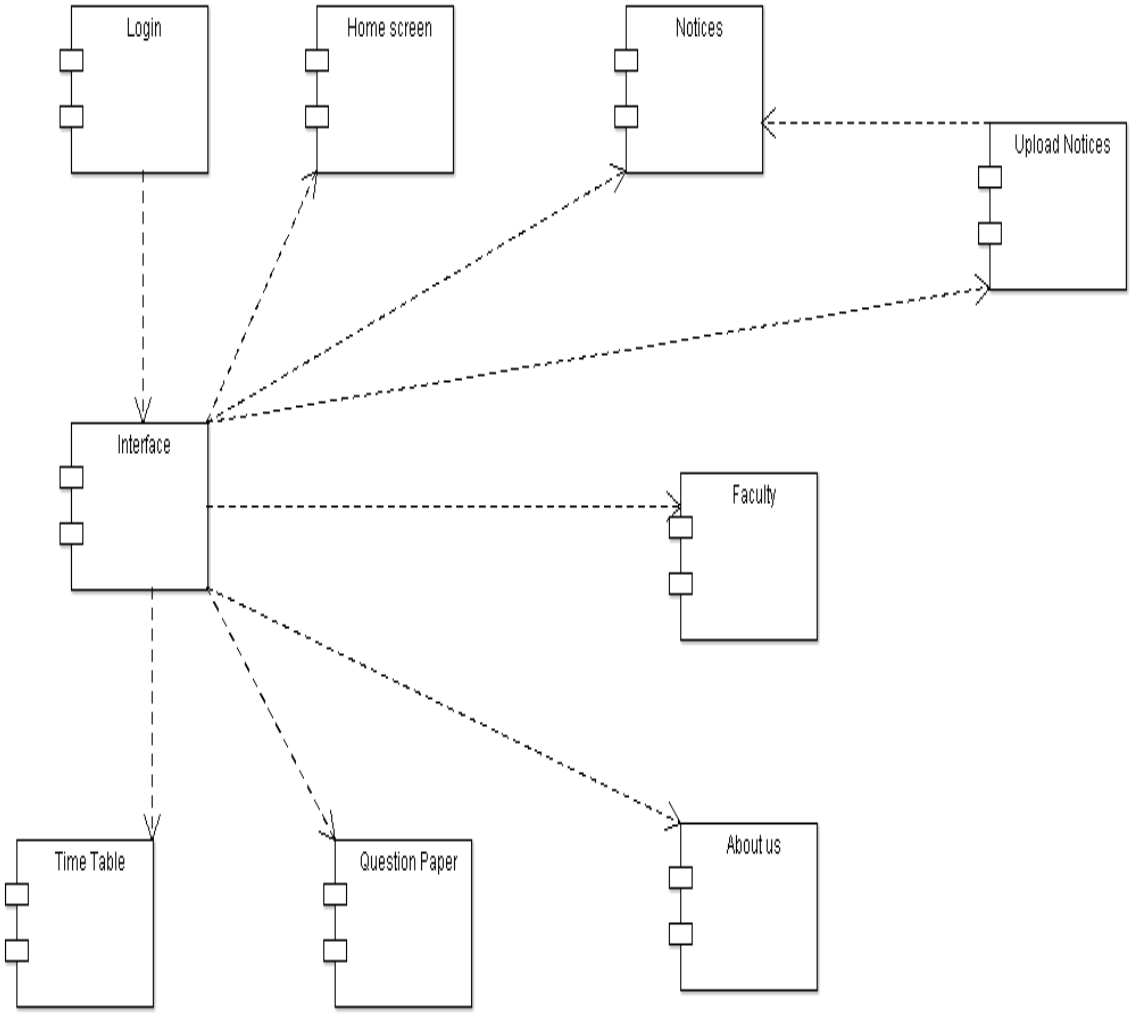
Use Case Diagram



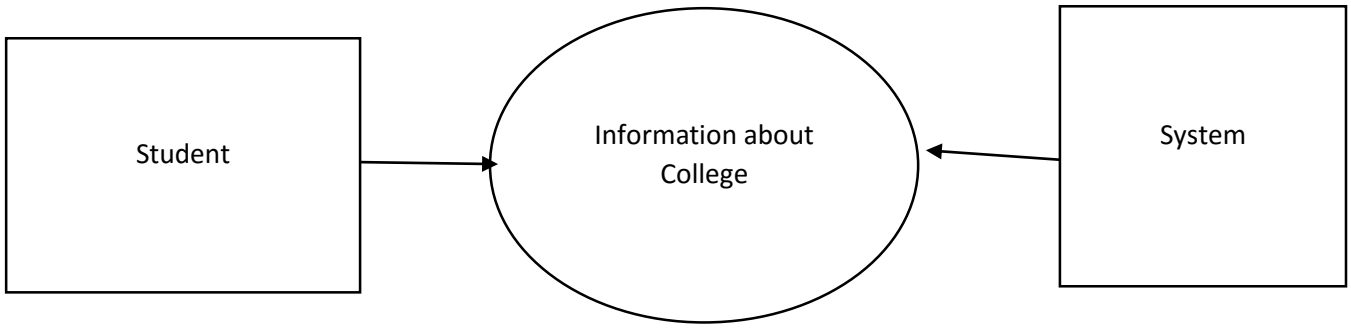
Sequence Diagram



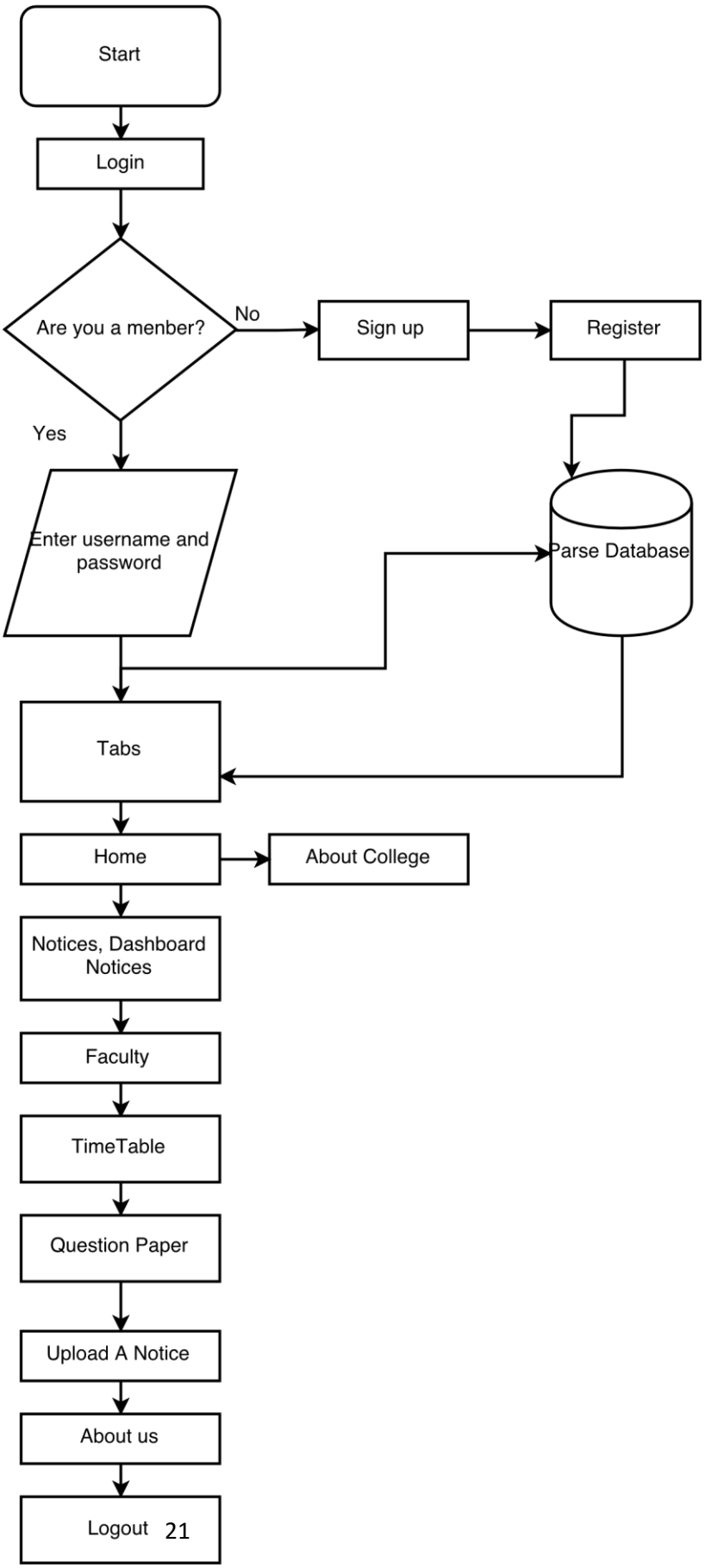
Component Diagram



Data Flow Diagram



Flow Diagram


Start
Login
Are you a menber?
No
Sign up
Register
Yes
Enter username and password
Parse Database
Tabs
Home
About College
Notices, Dashboard Notices
Faculty
TimeTable
Question Paper
Upload A Notice
About us
Logout

CODING

Code for Welcome Screen:

```
package com.college.poco.utils.welcomescreen;

/**
 * Created by ahuja on 3/19/2016.
 */
import android.animation.ObjectAnimator;
import android.animation.StateListAnimator;
import android.app.Activity;
import android.content.Intent;
import android.content.pm.ActivityInfo;
import android.database.DataSetObserver;
import android.os.Build;
import android.os.Bundle;
import android.os.Parcelable;
import android.support.v4.view.PagerAdapter;
import android.support.v4.view.ViewPager;
import android.view.View;
import android.view.ViewGroup;
import android.view.Window;
import android.view.animation.Animation;
import android.view.animation.AnimationUtils;
import android.widget.ImageView;
import android.widget.TextView;

import com.college.poco.R;
import com.college.poco.utils.SplashScreen;
import com.college.poco.utils.Welcome;
import com.viewpagerindicator.CirclePageIndicator;
import com.viewpagerindicator.PageIndicator;

public class IntroActivity extends Activity {
    private ViewPager viewPager;
    private PageIndicator indicator;
    private ImageView topImage1;
    private ImageView topImage2;
    private int lastPage = 0;
    private boolean startPressed = false;
    private int[] icons;
    private int[] messages;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        requestWindowFeature(Window.FEATURE_NO_TITLE);

        setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_PORTRAIT);
        setContentView(R.layout.intro_layout);

        icons = new int[]{
                R.drawable.app_overview_2_smartphone,
                R.drawable.app_overview_1_buy,
                R.drawable.app_overview_3_save_money,
                R.drawable.app_overview_4_coupons
        };
        messages = new int[]{
                R.string.app_overview_1st_item_text,
```

```
                R.string.app_overview_2nd_item_text,
                R.string.app_overview_3rd_item_text,
                R.string.app_overview_4th_item_text
        };
        viewPager = (ViewPager) findViewById(R.id.intro_view_pager);
        TextView startMessagingButton = (TextView)
findViewById(R.id.start_messaging_button);
        startMessagingButton.setText(getString(R.string.StartMessaging));
        if (Build.VERSION.SDK_INT >= 21) {
            StateListAnimator animator = new StateListAnimator();
            animator.addState(new int[]{android.R.attr.state_pressed},
ObjectAnimator.ofFloat(startMessagingButton, "translationZ", AndroidUtilities.dp(2,
this), AndroidUtilities.dp(4, this)).setDuration(200));
            animator.addState(new int[]{},
ObjectAnimator.ofFloat(startMessagingButton, "translationZ", AndroidUtilities.dp(4,
this), AndroidUtilities.dp(2, this)).setDuration(200));
            startMessagingButton.setStateListAnimator(animator);
        }
        topImage1 = (ImageView) findViewById(R.id.icon_image1);
        topImage2 = (ImageView) findViewById(R.id.icon_image2);
        indicator = (CirclePageIndicator) findViewById(R.id.indicator);

        topImage2.setVisibility(View.GONE);
        viewPager.setAdapter(new IntroAdapter());
        viewPager.setPageMargin(0);
        viewPager.setOffscreenPageLimit(1);
        indicator.setViewPager(viewPager);
        indicator.setOnPageChangeListener(new ViewPager.OnPageChangeListener() {
            @Override
            public void onPageScrolled(int position, float positionOffset, int
positionOffsetPixels) {

                if (lastPage != viewPager.getCurrentItem()) {
                    lastPage = viewPager.getCurrentItem();

                    final ImageView fadeoutImage;
                    final ImageView fadeinImage;
                    if (topImage1.getVisibility() == View.VISIBLE) {
                        fadeoutImage = topImage1;
                        fadeinImage = topImage2;

                    } else {
                        fadeoutImage = topImage2;
                        fadeinImage = topImage1;
                    }

                    fadeinImage.bringToFront();
                    fadeinImage.setImageResource(icons[lastPage]);
                    fadeinImage.clearAnimation();
                    fadeoutImage.clearAnimation();

                    Animation outAnimation =
AnimationUtils.loadAnimation(IntroActivity.this, R.anim.icon_anim_fade_out);
                    outAnimation.setAnimationListener(new
Animation.AnimationListener() {
                        @Override
                        public void onAnimationStart(Animation animation) {
                        }

                        @Override
                        public void onAnimationEnd(Animation animation) {
                            fadeoutImage.setVisibility(View.GONE);
                        }
```

```
                        @Override
                        public void onAnimationRepeat(Animation animation) {

                        }
                    });

                    Animation inAnimation =
AnimationUtils.loadAnimation(IntroActivity.this, R.anim.icon_anim_fade_in);
                    inAnimation.setAnimationListener(new Animation.AnimationListener()
{
                        @Override
                        public void onAnimationStart(Animation animation) {
                            fadeinImage.setVisibility(View.VISIBLE);
                        }

                        @Override
                        public void onAnimationEnd(Animation animation) {
                        }

                        @Override
                        public void onAnimationRepeat(Animation animation) {

                        }
                    });


                    fadeoutImage.startAnimation(outAnimation);
                    fadeinImage.startAnimation(inAnimation);
                }
            }

            @Override
            public void onPageSelected(int position) {

            }

            @Override
            public void onPageScrollStateChanged(int state) {

            }
        });

        startMessagingButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                if (startPressed) {
                    return;
                }
                startPressed = true;
                Intent intent2 = new Intent(IntroActivity.this, SplashScreen.class);
                startActivity(intent2);
                finish();
            }
        });
    }

    private class IntroAdapter extends PagerAdapter {

        @Override
        public int getCount() {
            return 4;
        }
```

```
        @Override
        public boolean isViewFromObject(View view, Object object) {
            return view.equals(object);
        }

        @Override
        public Object instantiateItem(ViewGroup container, int position) {
            View view = View.inflate(container.getContext(),
R.layout.intro_view_layout, null);
            TextView messageTextView = (TextView)
view.findViewById(R.id.message_text);
            container.addView(view, 0);

messageTextView.setText(AndroidUtilities.replaceTags(getString(messages[position]),
getApplicationContext()));
            return view;
        }

        @Override
        public void destroyItem(ViewGroup container, int position, Object object) {
            ((ViewPager) container).removeView((View) object);
        }
    }
}
```

Code for Splash Screen:

```
package com.college.poco.utils;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;

import com.college.poco.R;

/**
 * Created by ahuja on 3/3/2016.
 */
public class SplashScreen extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        // TODO Auto-generated method stub
        super.onCreate(savedInstanceState);
        setContentView(R.layout.splash);

        Thread timerThread = new Thread(){
            public void run(){
                try{
                    sleep(3000);
                }catch(InterruptedException e){
                    e.printStackTrace();
                }finally{
                    if(!isFinishing()) {
                        Intent intent = new Intent(SplashScreen.this, Welcome.class);
                        startActivity(intent);
                        finish();
                    }
                }
```

```
            }
        };

        timerThread.start();
    }
    @Override
    public void onBackPressed() {

        super.onBackPressed();
        finish();
    } }
```

XML for Splash Screen

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/splash">

</LinearLayout>
```

Code for Login Screen

```
package com.college.poco.utils;
import android.app.Activity;
import android.app.ProgressDialog;
import android.content.Context;
import android.content.Intent;
import android.graphics.Color;
import android.graphics.Typeface;
import android.os.Bundle;
import android.os.Handler;
import android.support.design.widget.Snackbar;
import android.support.design.widget.TextInputLayout;
import android.util.Log;
import android.view.KeyEvent;
import android.view.MotionEvent;
import android.view.View;
import android.view.inputmethod.InputMethodManager;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import com.college.poco.FirstActivity;
import com.college.poco.R;
import com.parse.LogInCallback;
import com.parse.ParseException;
import com.parse.ParseUser;

import android.view.View;


/**
 * Created by hemna on 26-12-2015.
 */
public class LogIn extends Activity {
```

```
    EditText si_un, si_pwd;
    Button signUp, logIn;
    String stUN, stPWD;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);
        logIn = (Button) findViewById(R.id.li_btn);
        si_un = (EditText) findViewById(R.id.li_un);
        si_pwd = (EditText) findViewById(R.id.li_pwd);
        signUp = (Button) findViewById(R.id.si_btn);


        signUp.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(LogIn.this, SignUp.class);
                startActivity(intent);
            }
        });


        logIn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                final ProgressDialog progressDialog = new ProgressDialog(LogIn.this);
                progressDialog.setMessage("Logging in...");
                progressDialog.show();
                stUN = si_un.getText().toString();
                stPWD = si_pwd.getText().toString();
                ParseUser.logInInBackground(stUN, stPWD, new LogInCallback() {
                    @Override
                    public void done(ParseUser parseUser, ParseException e) {
                        if (e == null) {
//                            View view1 = findViewById(android.R.id.content);
//                            Snackbar.make(view1, "User Logged in.",
Snackbar.LENGTH_LONG).show();
                            Toast.makeText(LogIn.this, "User logged in successfully.",
Toast.LENGTH_LONG).show();
                            Intent intent = new Intent(LogIn.this,
FirstActivity.class);
                            intent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
                            intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TASK);
                            startActivity(intent);
                        } else {
                            progressDialog.dismiss();
                            View view2 = findViewById(android.R.id.content);
                            Snackbar.make(view2, "Error. Wrong Username or Password",
Snackbar.LENGTH_LONG).show();

view2.setBackgroundColor(getResources().getColor(R.color.RED));

//                            Toast.makeText(LogIn.this, "Error..!!",
Toast.LENGTH_LONG).show();
                        }

                    }

                });
            }
        });
```

```
    //    For disabling keyboard in EditText
    @Override
    public boolean dispatchTouchEvent(MotionEvent event) {

        View v = getCurrentFocus();
        boolean ret = super.dispatchTouchEvent(event);

        if (v instanceof EditText) {
            View w = getCurrentFocus();
            int scrcoords[] = new int[2];
            w.getLocationOnScreen(scrcoords);
            float x = event.getRawX() + w.getLeft() - scrcoords[0];
            float y = event.getRawY() + w.getTop() - scrcoords[1];

            Log.d("Activity", "Touch event " + event.getRawX() + "," + event.getRawY()
+ " " + x + "," + y + " rect " + w.getLeft() + "," + w.getTop() + "," + w.getRight() +
"," + w.getBottom() + " coords " + scrcoords[0] + "," + scrcoords[1]);
            if (event.getAction() == MotionEvent.ACTION_UP && (x < w.getLeft() || x >=
w.getRight() || y < w.getTop() || y > w.getBottom()) ) {

                InputMethodManager imm =
(InputMethodManager)getSystemService(Context.INPUT_METHOD_SERVICE);

imm.hideSoftInputFromWindow(getWindow().getCurrentFocus().getWindowToken(), 0);
            }
        }
        return ret;
    }

    @Override
    public void onBackPressed() {

// make sure you have this outcommented
// super.onBackPressed();
    Intent intent = new Intent(Intent.ACTION_MAIN);
    intent.addCategory(Intent.CATEGORY_HOME);
    intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
    startActivity(intent);} }
```

XML for Login Screen:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    xmlns:tools="http://schemas.android.com/tools"
    android:background="@color/PrimaryColor"
    android:gravity="center"
    android:orientation="vertical"
    android:padding="10dp"
    tools:context="com.college.poco.utils.LogIn">

    <LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:orientation="vertical"
        android:paddingLeft="20dp"
```

```
    android:paddingRight="20dp" >

    <ImageView android:layout_gravity="center"
        android:layout_width="500.0dip"
        android:layout_height="100.0dip"
        android:src="@drawable/poco"/>



    <EditText
        android:id="@+id/li_un"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_marginBottom="10dp"
        android:background="@color/white"
        android:hint="Enter Username"
        android:inputType="textEmailAddress"
        android:padding="10dp"
        android:singleLine="true"
        android:textColor="@color/black"
        android:textColorHint="@color/grey" />

    <EditText
        android:id="@+id/li_pwd"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_marginBottom="10dp"
        android:background="@color/white"
        android:hint="Password"
        android:inputType="textPassword"
        android:padding="10dp"
        android:singleLine="true"
        android:textColor="@color/black"
        android:textColorHint="@color/grey"
        />



    <!-- Login Button -->

    <Button
        android:id="@+id/li_btn"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="20dip"
        android:background="@color/white"
        android:text="Login"
        android:textColor="@color/PrimaryColor" />

    <!-- Link to Login Screen -->

    <Button
        android:id="@+id/si_btn"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="40dip"
        android:background="@null"
        android:text="Not a member? Sign up now"
        android:textAllCaps="false"
        android:textColor="@color/white"
        android:textSize="15dp" />
</LinearLayout>
```

```
</LinearLayout>
```

Code for Signup Screen:

```
package com.college.poco.utils;

import android.app.Activity;
import android.app.ProgressDialog;
import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import android.support.design.widget.Snackbar;
import android.util.Log;
import android.view.MotionEvent;
import android.view.View;
import android.view.inputmethod.InputMethodManager;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import com.college.poco.FirstActivity;
import com.college.poco.R;
import com.parse.ParseException;
import com.parse.ParseUser;
import com.parse.SignUpCallback;

/**
 * Created by hemna on 26-12-2015.
 */
public class SignUp extends Activity {
    EditText username,pwd;
    EditText mail_id;
    Button submit,log;
    String stUN,stPWD,stMI;


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_signup);
        username=(EditText)findViewById(R.id.username);
        pwd=(EditText)findViewById(R.id.pwd);
        mail_id=(EditText)findViewById(R.id.mailadd);
        submit=(Button)findViewById(R.id.submit);
        log=(Button)findViewById(R.id.log_btn);

        log.setOnClickListener(new View.OnClickListener() {

            public void onClick(View view) {
                Intent i = new Intent(getApplicationContext(),
                        LogIn.class);
                startActivity(i);
                finish();
            }
        });


        submit.setOnClickListener(new View.OnClickListener() {
```

```
            @Override
            public void onClick(View v) {
                final ProgressDialog progressDialog = new ProgressDialog(SignUp.this);
                progressDialog.setMessage("Signing Up...");
                progressDialog.show();
                stUN=username.getText().toString();
                stPWD=pwd.getText().toString();
                stMI=mail_id.getText().toString();
                ParseUser user=new ParseUser();
                user.setUsername(stUN);
                user.setPassword(stPWD);
                user.setEmail(stMI);
                user.signUpInBackground(new SignUpCallback() {
                    public void done(ParseException e) {
                        if (e == null) {
                            Toast.makeText(SignUp.this, "User created successfully.",
Toast.LENGTH_LONG).show();
                            Intent intent = new Intent(SignUp.this,
FirstActivity.class);
                            intent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
                            intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TASK);
                            startActivity(intent);
                        } else {
                            progressDialog.dismiss();
                            View view2 = findViewById(android.R.id.content);
                            Snackbar.make(view2, "Please Check your parameters and /
or Your internet connectivity.Thank You", Snackbar.LENGTH_LONG).show();
//                              Toast.makeText(SignUp.this, "Please Check your
parameters and / or Your internet connectivity.Thank You", Toast.LENGTH_LONG).show();
                        }
                    }
                });

            }
        }

        );



    }

    //    For disabling keyboard in EditText
    @Override
    public boolean dispatchTouchEvent(MotionEvent event) {

        View v = getCurrentFocus();
        boolean ret = super.dispatchTouchEvent(event);

        if (v instanceof EditText) {
            View w = getCurrentFocus();
            int scrcoords[] = new int[2];
            w.getLocationOnScreen(scrcoords);
            float x = event.getRawX() + w.getLeft() - scrcoords[0];
            float y = event.getRawY() + w.getTop() - scrcoords[1];

            Log.d("Activity", "Touch event " + event.getRawX() + "," + event.getRawY()
+ " " + x + "," + y + " rect " + w.getLeft() + "," + w.getTop() + "," + w.getRight() +
"," + w.getBottom() + " coords " + scrcoords[0] + "," + scrcoords[1]);
            if (event.getAction() == MotionEvent.ACTION_UP && (x < w.getLeft() || x >=
w.getRight() || y < w.getTop() || y > w.getBottom()) ) {

                InputMethodManager imm =
```

```
(InputMethodManager)getSystemService(Context.INPUT_METHOD_SERVICE);

imm.hideSoftInputFromWindow(getWindow().getCurrentFocus().getWindowToken(), 0);
            }
        }
        return ret;
    }

    @Override
    public void onBackPressed() {
        super.onBackPressed();
        finish();
    }
}
```

XML for Login Screen

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="@color/black"
    android:gravity="center"
    android:orientation="vertical"
    android:padding="10dp" >

    <LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:orientation="vertical"
        android:paddingLeft="20dp"
        android:paddingRight="20dp" >

        <EditText
            android:id="@+id/username"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginBottom="10dp"
            android:background="@color/white"
            android:hint="Username"
            android:padding="10dp"
            android:singleLine="true"
            android:inputType="textCapWords"
            android:textColor="@color/black"
            android:textColorHint="@color/grey" />

        <EditText
            android:id="@+id/mailadd"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginBottom="10dp"
            android:background="@color/white"
            android:hint="Email address"
            android:inputType="textEmailAddress"
            android:padding="10dp"
            android:singleLine="true"
            android:textColor="@color/black"
            android:textColorHint="@color/grey" />
```

```
        <EditText
            android:id="@+id/pwd"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginBottom="10dp"
            android:background="@color/white"
            android:hint="Password"
            android:inputType="textPassword"
            android:padding="10dp"
            android:singleLine="true"
            android:textColor="@color/black"
            android:textColorHint="@color/grey" />

        <!-- Login Button -->

        <Button
            android:id="@+id/submit"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="20dip"
            android:background="#ea4c88"
            android:text="SignUp"
            android:textColor="@color/white" />

        <!-- Link to Login Screen -->

        <Button
            android:id="@+id/log_btn"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="40dip"
            android:background="@null"
            android:text="Already registered!Log me"
            android:textAllCaps="false"
            android:textColor="@color/white"
            android:textSize="15dp" />
    </LinearLayout>

</LinearLayout>
```

Code for Parse Application

```
package com.college.poco.utils;

import android.app.Application;

import com.college.poco.R;
import com.parse.Parse;
import com.parse.ParseInstallation;

/**
 * Created by ahuja on 3/18/2016.
 */
public class ParseApplication extends Application {

    @Override
    public void onCreate() {
        super.onCreate();
```

```
        // Add your initialization code here
        Parse.initialize(this, getString(R.string.parse_app_id),
getString(R.string.parse_client_id));
        ParseInstallation.getCurrentInstallation().saveInBackground();
//        ParseUser.enableAutomaticUser();
//        ParseACL defaultACL = new ParseACL();
//
//        // If you would like all objects to be private by default, remove this
//        // line.
//        defaultACL.setPublicReadAccess(true);
//
//        ParseACL.setDefaultACL(defaultACL, true);
    }

}
```

Code for Home Screen

```
package com.college.poco;

import android.app.ProgressDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.os.Handler;
import android.support.v7.app.AlertDialog;
import android.support.v7.widget.CardView;
import android.widget.Toast;

import com.college.poco.utils.LogIn;
import com.parse.ParseUser;

import java.io.FileReader;

public class FirstActivity extends BaseDrawerActivity {

    protected int getSelfNavDrawerItem() {
        return NAVDRAWER_ITEM_HOME;
    }

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

    @Override
    public void onBackPressed() {
        AlertDialog.Builder alertDialogBuilder = new
AlertDialog.Builder(FirstActivity.this);
        // set title
        alertDialogBuilder.setTitle("Close App?");
        // set dialog message
        alertDialogBuilder
                .setMessage("Do you really want to close this beautiful app?")
                .setCancelable(false)
                .setPositiveButton("Yes", new DialogInterface.OnClickListener() {
                    public void onClick(DialogInterface dialog, int id) {
                        Intent intent = new Intent(Intent.ACTION_MAIN);
                        intent.addCategory(Intent.CATEGORY_HOME);
                        intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
```

```
                        startActivity(intent);
                        //do whatever you want to do when user clicks ok
                    }
                })
                .setNegativeButton("No", new DialogInterface.OnClickListener() {
                    public void onClick(DialogInterface dialog, int id) {
                        // if this button is clicked, just close
                        // the dialog box and do nothing
                        dialog.cancel();
                    }
                });
        // create alert dialog
        AlertDialog alertDialog = alertDialogBuilder.create();
        // show it
        alertDialog.show();
    }
}
```

XML for Home Screen

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.design.widget.CoordinatorLayout
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"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    tools:context="com.college.poco.FirstActivity"
    android:background="@drawable/bkground"
    app:layout_behavior="@string/appbar_scrolling_view_behavior">

  <android.support.design.widget.AppBarLayout
    android:id="@+id/app_bar_layout"
    android:layout_width="match_parent"
    android:layout_height="wrap_content">

  <android.support.v7.widget.Toolbar
    android:id="@+id/toolbar"
    android:layout_width="match_parent"
    android:layout_height="?attr/actionBarSize"
    android:theme="@style/ThemeOverlay.AppCompat.Dark.ActionBar" />
  </android.support.design.widget.AppBarLayout>
    <android.support.v7.widget.CardView
        android:orientation="horizontal"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        app:layout_behavior="@string/appbar_scrolling_view_behavior"
        card_view:cardCornerRadius="5.0dip"
        card_view:cardUseCompatPadding="true"
        xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:card_view="http://schemas.android.com/apk/res-auto">
    <RelativeLayout android:background="#ffffffff" android:layout_width="fill_parent"
android:layout_height="fill_parent">
  <ImageView android:layout_gravity="top" android:id="@id/imageView3"
android:layout_width="500.0dip" android:layout_height="100.0dip"
android:layout_marginTop="10.0dip" android:src="@drawable/mainicon"
      app:layout_behavior="@string/appbar_scrolling_view_behavior"/>
  <TextView android:textColor="#ff000000" android:paddingLeft="10.0dip"
android:paddingRight="10.0dip" android:layout_width="match_parent"
android:layout_height="match_parent" android:text="@string/aboutcollege"
```

```
      app:layout_behavior="@string/appbar_scrolling_view_behavior"/>
    </RelativeLayout>
    </android.support.v7.widget.CardView>

</android.support.design.widget.CoordinatorLayout>
```

Code for Navigation Drawer

```
package com.college.poco;

import android.app.ProgressDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.os.Handler;
import android.support.annotation.Nullable;
import android.support.design.widget.NavigationView;
import android.support.v4.view.GravityCompat;
import android.support.v4.widget.DrawerLayout;
import android.support.v7.app.AlertDialog;
import android.view.LayoutInflater;
import android.view.MenuItem;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageView;
import android.widget.Toast;

import butterknife.Bind;
import butterknife.BindString;

import com.college.poco.utils.LogIn;
import com.college.poco.utils.uploadanotice.MainActivity;
import com.parse.ParseUser;
import com.squareup.picasso.Picasso;
import timber.log.Timber;

public class BaseDrawerActivity extends BaseActivity
    implements NavigationView.OnNavigationItemSelectedListener {

  @Bind(R.id.drawer_layout)
  DrawerLayout drawerLayout;
  @Bind(R.id.nav_view)
  NavigationView navigationView;
  @Nullable
  @Bind(R.id.main_content)
  View mainContent;

  @BindString(R.string.user_profile_photo)
  String profilePhoto;

  // delay to launch nav drawer item, to allow close animation to play
  private static final int NAVDRAWER_LAUNCH_DELAY = 250;

  protected static final int NAVDRAWER_ITEM_INVALID = -1;

  protected static final int NAVDRAWER_ITEM_HOME = R.id.nav_home;
  protected static final int NAVDRAWER_ITEM_NOTICES = R.id.nav_notices;
  protected static final int NAVDRAWER_ITEM_FACULTY = R.id.nav_faculty;
```

```
  protected static final int NAVDRAWER_ITEM_QUESTION = R.id.nav_question;
  protected static final int NAVDRAWER_ITEM_LOGOUT = R.id.nav_logout;
  protected static final int NAVDRAWER_ITEM_TIMETABLE = R.id.nav_timetable;
  protected static final int NAVDRAWER_ITEM_ABOUTUS = R.id.nav_aboutus;
  protected static final int NAVDRAWER_ITEM_UPLOADANOTICE = R.id.nav_uploadanotice;
  protected static final int NAVDRAWER_ITEM_NOTIIMAGE = R.id.nav_notiimage;

  // fade in and fade out durations for the main content when switching between
  // different Activities of the app through the Nav Drawer
  private static final int MAIN_CONTENT_FADEOUT_DURATION = 150;

  private static final int MAIN_CONTENT_FADEIN_DURATION = 250;

  /**
   * Returns the navigation drawer item that corresponds to this Activity. Subclasses
   * of BaseActivity override this to indicate what nav drawer item corresponds to
them
   * Return NAVDRAWER_ITEM_INVALID to mean that this Activity should not have a Nav
Drawer.
   */
  protected int getSelfNavDrawerItem() {
    return NAVDRAWER_ITEM_INVALID;
  }

  @Override
  public void setContentView(int layoutResID) {
    super.setContentViewWithoutInject(R.layout.activity_drawer);
    ViewGroup viewGroup = (ViewGroup) findViewById(R.id.content_frame);
    LayoutInflater.from(this).inflate(layoutResID, viewGroup, true);
    bindViews();
    setupHeader();

    navigationView.setNavigationItemSelectedListener(this);

    if (getSelfNavDrawerItem() != NAVDRAWER_ITEM_INVALID) {
      navigationView.getMenu().findItem(getSelfNavDrawerItem()).setChecked(true);
    }
  }

  @Override
  protected void onPostCreate(Bundle savedInstanceState) {
    super.onPostCreate(savedInstanceState);

    if (mainContent != null) {
      mainContent.setAlpha(0);
      mainContent.animate().alpha(1).setDuration(MAIN_CONTENT_FADEIN_DURATION);
    } else {
      Timber.w("No view with ID main_content to fade in.");
    }
  }

  @Override
  protected void setupToolbar() {
    super.setupToolbar();
    if (toolbar != null) {
      toolbar.setNavigationIcon(R.drawable.ic_menu_white);
      toolbar.setNavigationOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
          drawerLayout.openDrawer(GravityCompat.START);
        }
      });
    }
```

```
  }

  private void setupHeader() {
    View headerView = navigationView.getHeaderView(0);
    ImageView avatarView = (ImageView) headerView.findViewById(R.id.avatar);

    Picasso.with(BaseDrawerActivity.this).load(profilePhoto).into(avatarView);
  }

  @Override
  public boolean onNavigationItemSelected(final MenuItem item) {

    item.setChecked(true);

    if (getSelfNavDrawerItem() != item.getItemId()) {

      Handler mHandler = new Handler();
      mHandler.postDelayed(new Runnable() {
        @Override
        public void run() {
          final Intent intent;
          switch (item.getItemId()) {
            case NAVDRAWER_ITEM_HOME:
              intent = new Intent(BaseDrawerActivity.this, FirstActivity.class);
              intent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
              intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TASK);
              intent.addFlags(Intent.FLAG_ACTIVITY_NO_ANIMATION);
              startActivity(intent);
              finish();
              break;
            case NAVDRAWER_ITEM_NOTICES:
              intent = new Intent(BaseDrawerActivity.this,
com.college.poco.utils.noti.MainActivity.class);
              intent.addFlags(Intent.FLAG_ACTIVITY_NO_ANIMATION);
              startActivity(intent);
              finish();
              break;
            case NAVDRAWER_ITEM_FACULTY:
              intent = new Intent(BaseDrawerActivity.this,
com.college.poco.utils.faculty.MainActivity.class);
              intent.addFlags(Intent.FLAG_ACTIVITY_NO_ANIMATION);
              startActivity(intent);
              finish();
              break;
            case NAVDRAWER_ITEM_QUESTION:
              intent = new Intent(BaseDrawerActivity.this,
com.college.poco.utils.questionpaper2.QuestionPaperTrial.class);
              intent.addFlags(Intent.FLAG_ACTIVITY_NO_ANIMATION);
              startActivity(intent);
              finish();
              break;
            case NAVDRAWER_ITEM_LOGOUT:
              AlertDialog.Builder alertDialogBuilder = new
AlertDialog.Builder(BaseDrawerActivity.this);
              // set title
              alertDialogBuilder.setTitle("Alert");
              // set dialog message
              alertDialogBuilder
                      .setMessage("Do you really want to Logout?")
                      .setCancelable(false)
                      .setPositiveButton("Yes",new DialogInterface.OnClickListener() {
                        public void onClick(DialogInterface dialog,int id) {
                          ParseUser.logOut();
```

```
                            ProgressDialog progressDialog = new
ProgressDialog(BaseDrawerActivity.this);
                            progressDialog.setMessage("Logging Out...");
                            progressDialog.show();
                            Intent intent = new Intent(BaseDrawerActivity.this,
LogIn.class);
                            intent.addFlags(Intent.FLAG_ACTIVITY_NO_ANIMATION);
                            startActivity(intent);
                            //do whatever you want to do when user clicks ok
                          }
                        })
                        .setNegativeButton("No",new DialogInterface.OnClickListener() {
                          public void onClick(DialogInterface dialog,int id) {
                            // if this button is clicked, just close
                            // the dialog box and do nothing
                            dialog.cancel();
                          }
                        });
                // create alert dialog
                AlertDialog alertDialog = alertDialogBuilder.create();
                // show it
                alertDialog.show();
                break;
              case NAVDRAWER_ITEM_TIMETABLE:
                intent = new Intent(BaseDrawerActivity.this,
com.college.poco.utils.timetable2.TimeTable.class);
                intent.addFlags(Intent.FLAG_ACTIVITY_NO_ANIMATION);
                startActivity(intent);
                finish();
                break;
              case NAVDRAWER_ITEM_ABOUTUS:
                intent = new Intent(BaseDrawerActivity.this,
com.college.poco.utils.AboutUs.class);
                intent.addFlags(Intent.FLAG_ACTIVITY_NO_ANIMATION);
                startActivity(intent);
                finish();
                break;
              case NAVDRAWER_ITEM_UPLOADANOTICE:
                intent = new Intent(BaseDrawerActivity.this, MainActivity.class);
                intent.addFlags(Intent.FLAG_ACTIVITY_NO_ANIMATION);
                startActivity(intent);
                finish();
                break;
              case NAVDRAWER_ITEM_NOTIIMAGE:
                intent = new Intent(BaseDrawerActivity.this,
com.college.poco.utils.notiimage.MainActivity.class);
                intent.addFlags(Intent.FLAG_ACTIVITY_NO_ANIMATION);
                startActivity(intent);
                finish();
                break;


            }
          }
        }, NAVDRAWER_LAUNCH_DELAY);

        if (mainContent != null) {
          mainContent.animate().alpha(0).setDuration(MAIN_CONTENT_FADEOUT_DURATION);
        }
      }

      drawerLayout.closeDrawers();
      return true; }
```

Menu layout for Navigation Drawer

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto">


  <group android:checkableBehavior="single">
    <item
      android:id="@+id/nav_home"
      android:icon="@drawable/ic_home"
        app:itemIconTint="#000000"
      android:title="Home" />
    <item
      android:id="@+id/nav_notices"
      android:icon="@drawable/ic_notices"
        app:itemIconTint="#000000"
      android:title="Notices" />
      <item
          android:id="@+id/nav_notiimage"
          android:icon="@drawable/ic_menu_gallery"
          app:itemIconTint="#000000"
          android:title="Dashboard Notice" />
    <item
      android:id="@+id/nav_faculty"
        android:icon="@drawable/ic_faculty"
        app:itemIconTint="#000000"
        android:title="Faculty" />

    <item
        android:id="@+id/nav_timetable"
        android:icon="@drawable/ic_time_table"
        app:itemIconTint="#000000"
        android:title="Time Table"/>


    <item
      android:id="@+id/nav_question"
        android:icon="@drawable/ic_question"
        app:itemIconTint="#000000"
      android:title="Question Papers"/>

      <item  android:id="@+id/nav_uploadanotice"
          android:icon="@drawable/ic_menu_camera"
          app:itemIconTint="#000000"
          android:title="Upload a Notice"/>

      <item
          android:id="@+id/nav_aboutus"
          android:icon="@drawable/ic_aboutus"
          app:itemIconTint="#000000"
          android:title="About Us"/>

      <item
          android:id="@+id/nav_logout"
          android:icon="@drawable/ic_logout"
          app:itemIconTint="#000000"
          android:title="Log Out"/>
        </group>
        <menu>
```

DESIGN SPECIFICATION AND SCREENSHOTS

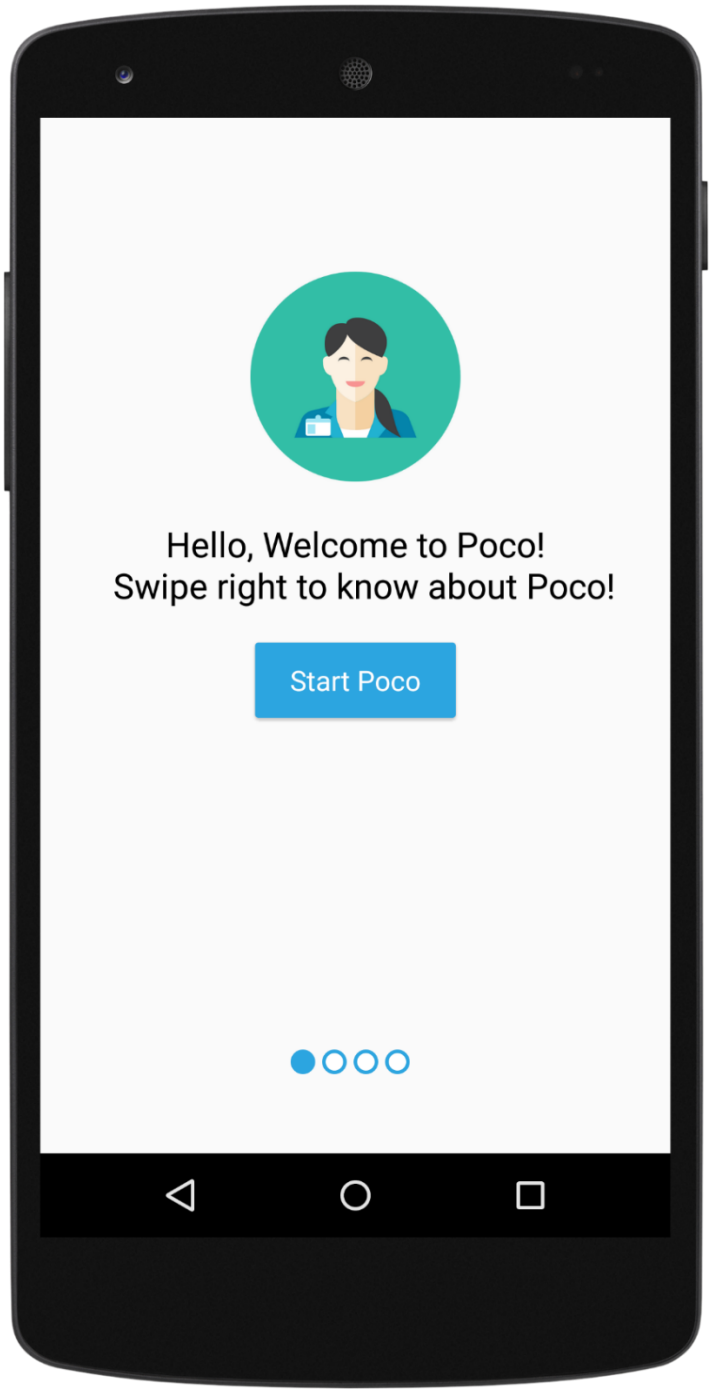
When learning, how to build applications on a new platform, it's important to first learn what APIs are available and how to use them. Later, the nuances of the platform are learnt. Put another way: first you learn how you can build app; later you can learn how you should build them.to ensure that your apps offer outstanding performance and a great user experience.

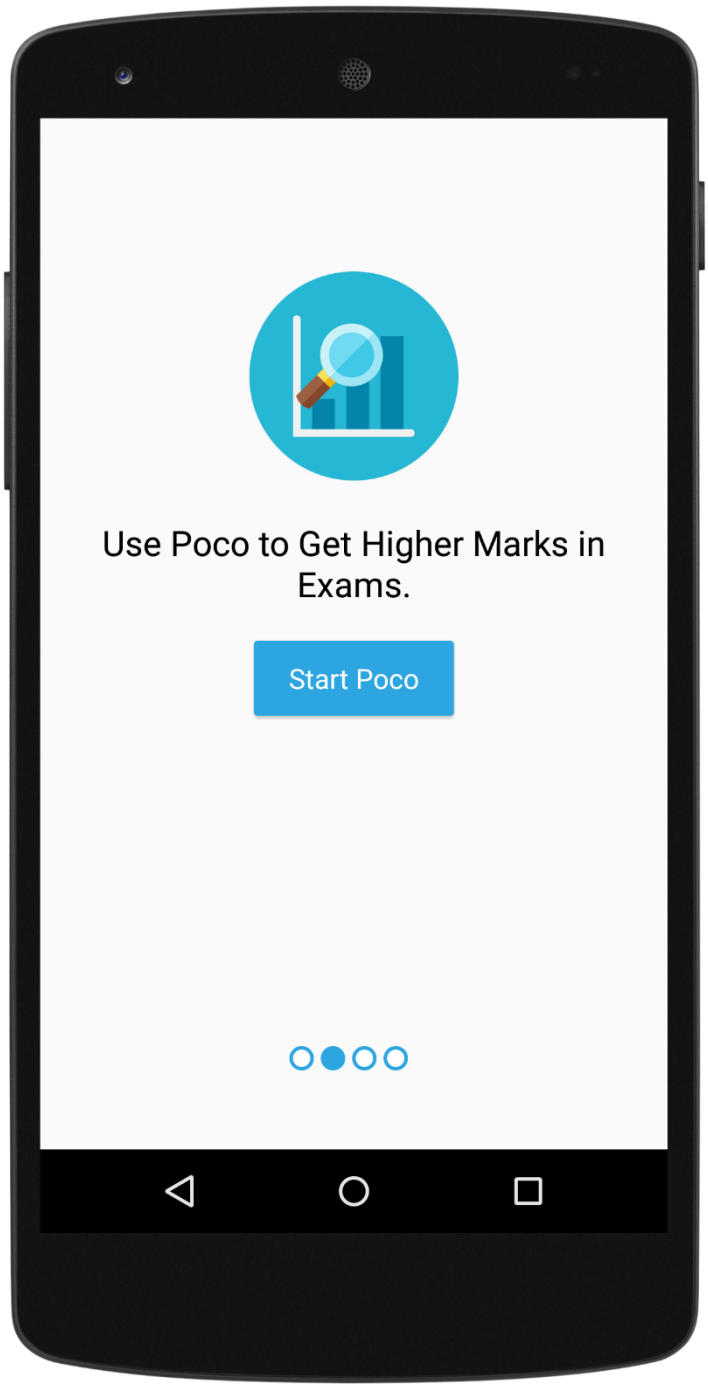
Successful mobile apps offer an outstanding user experience, in addition to a compelling technical feature set. The user experience is more than just its visual design or UI flow. It is also influenced by how well it interacts with other apps, and how fully and efficiently it uses device and system capabilities: it is fast: it is responsive: and it is seamless. Of course every platform achieves them in different ways. While Android also tries to match the same objectives, it as its design principal which are as:

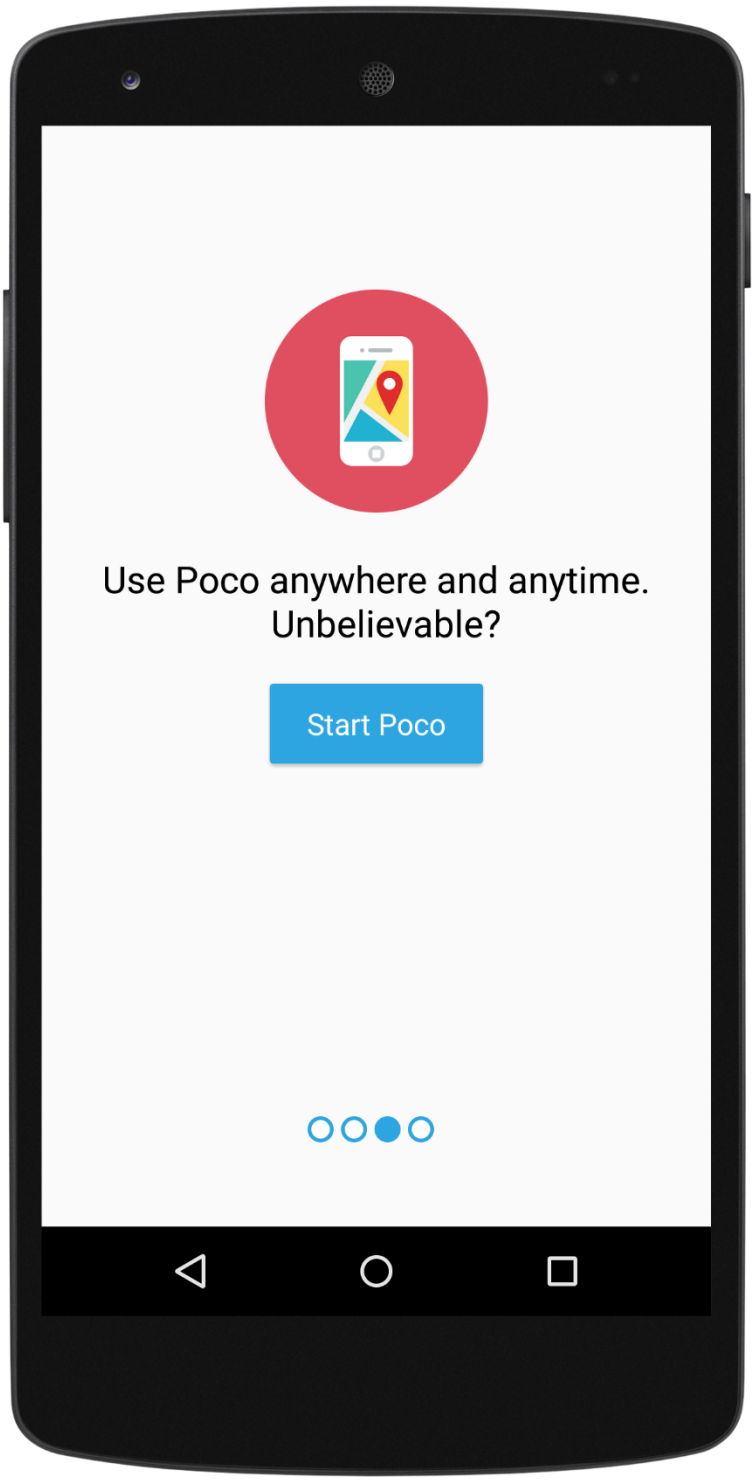
App Logo

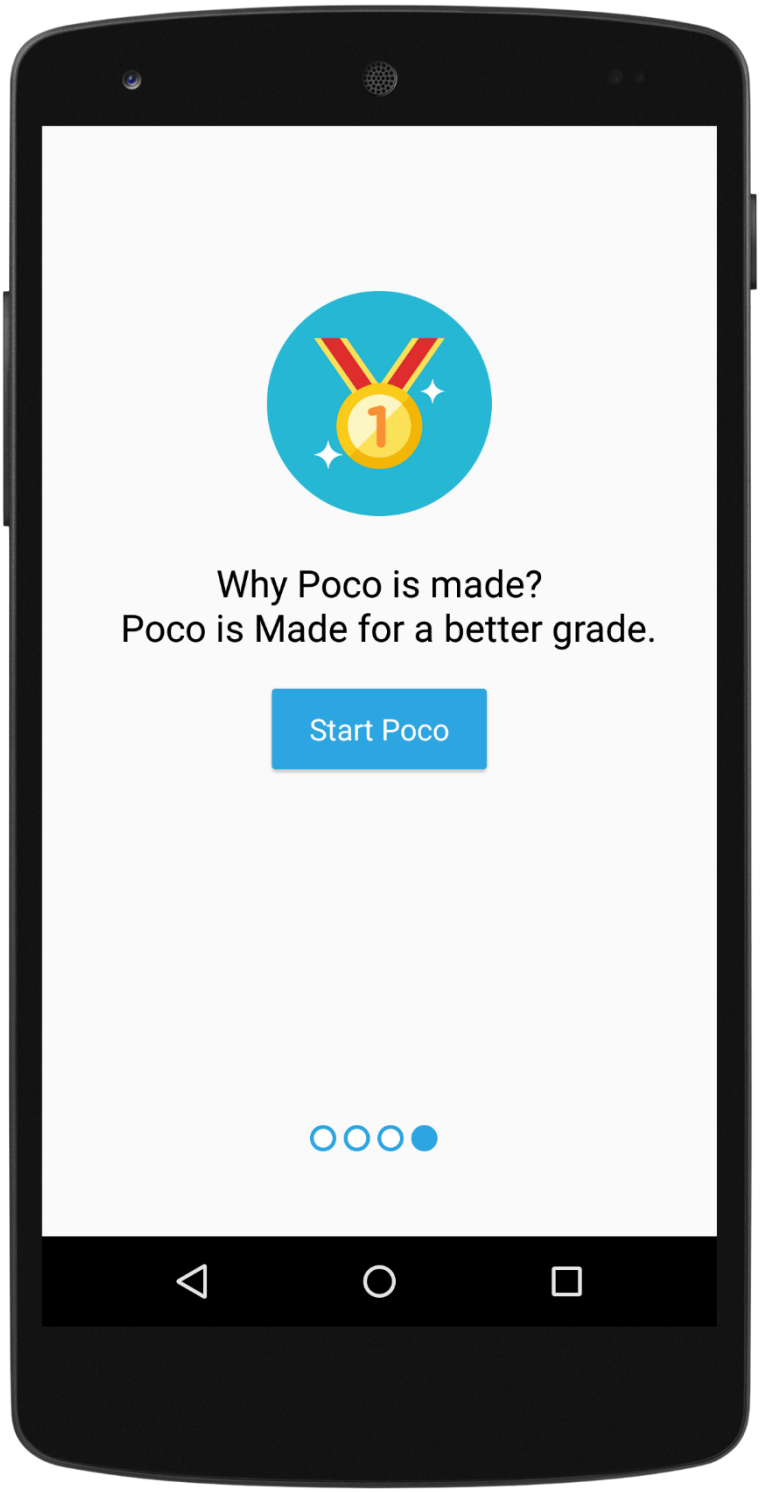
Welcome Screen

A beautiful surface, a carefully placed **Introduction** Activity:



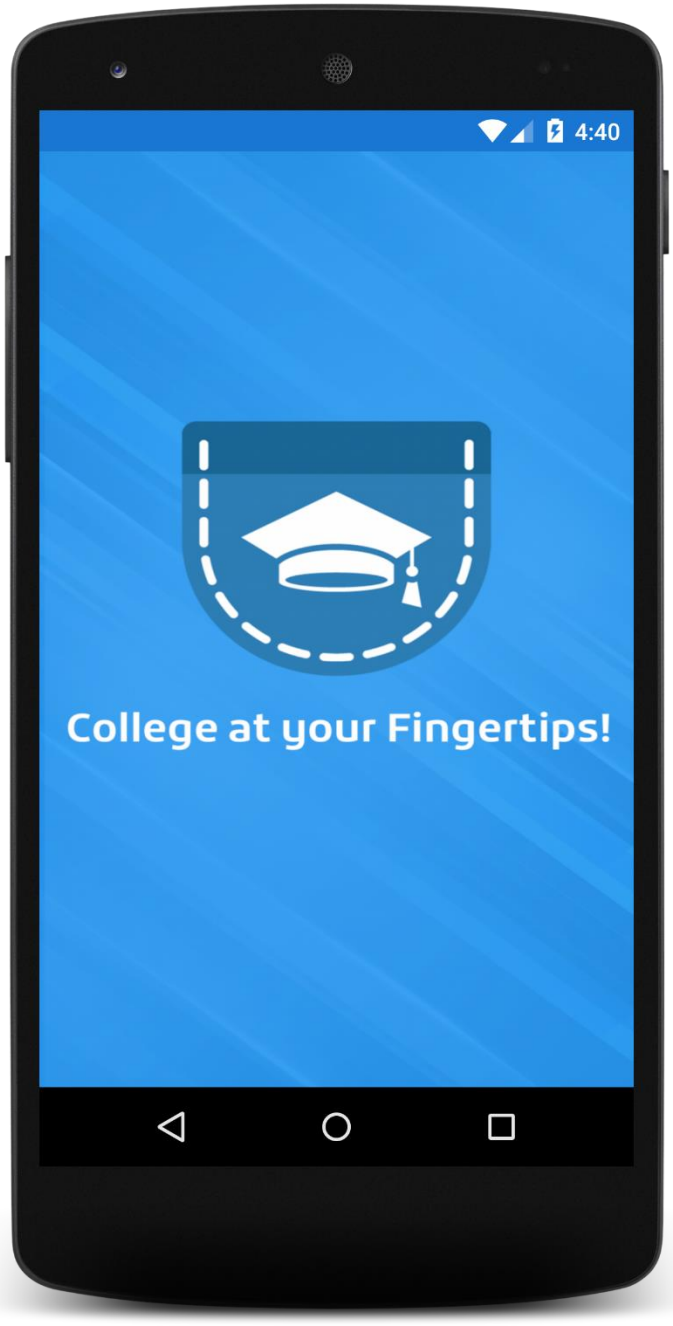

Use Poco to Get Higher Marks in Exams.
Start Poco


Use Poco anywhere and anytime.
Unbelievable?
Start Poco


Why Poco is made?
Poco is Made for a better grade.
Start Poco

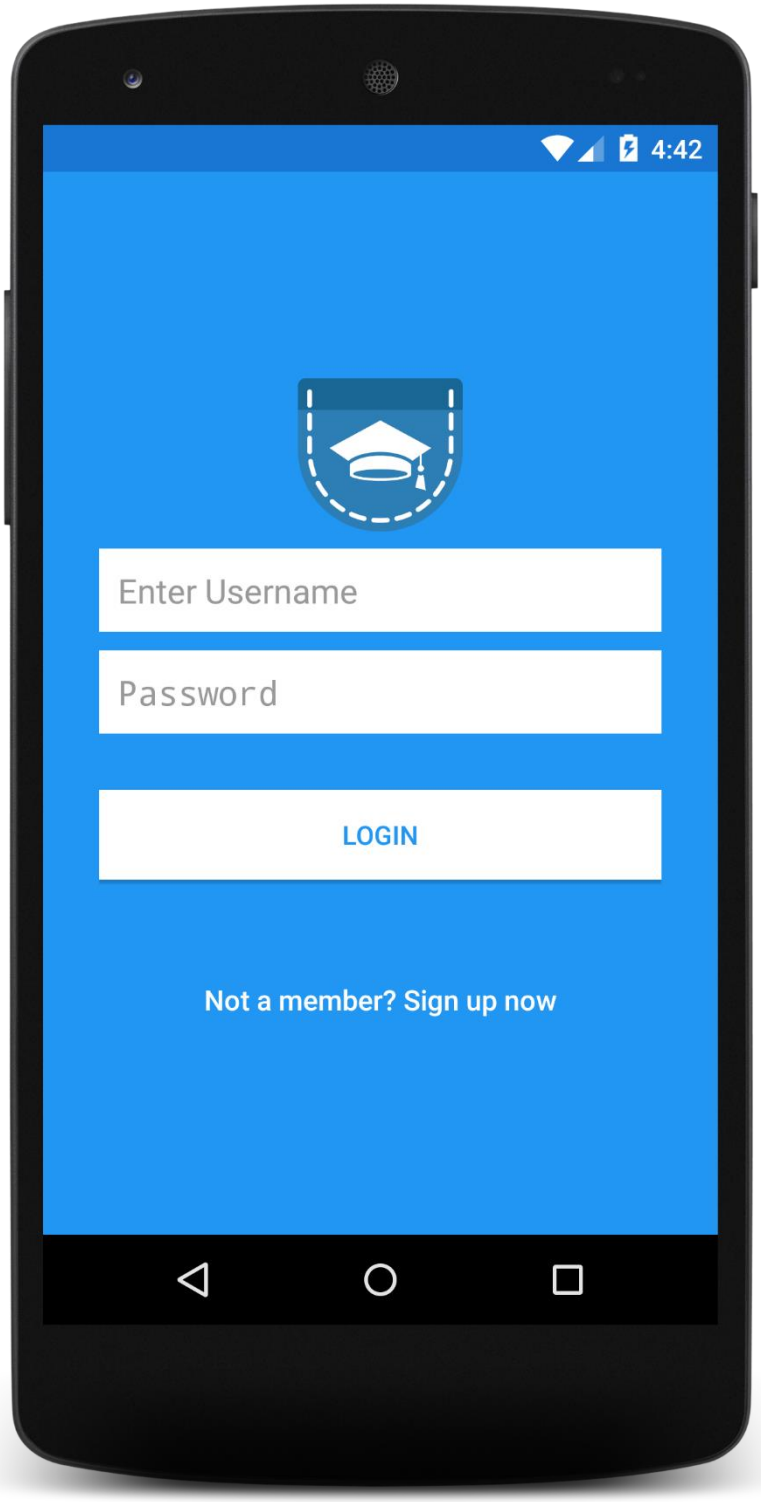
Splash Screen

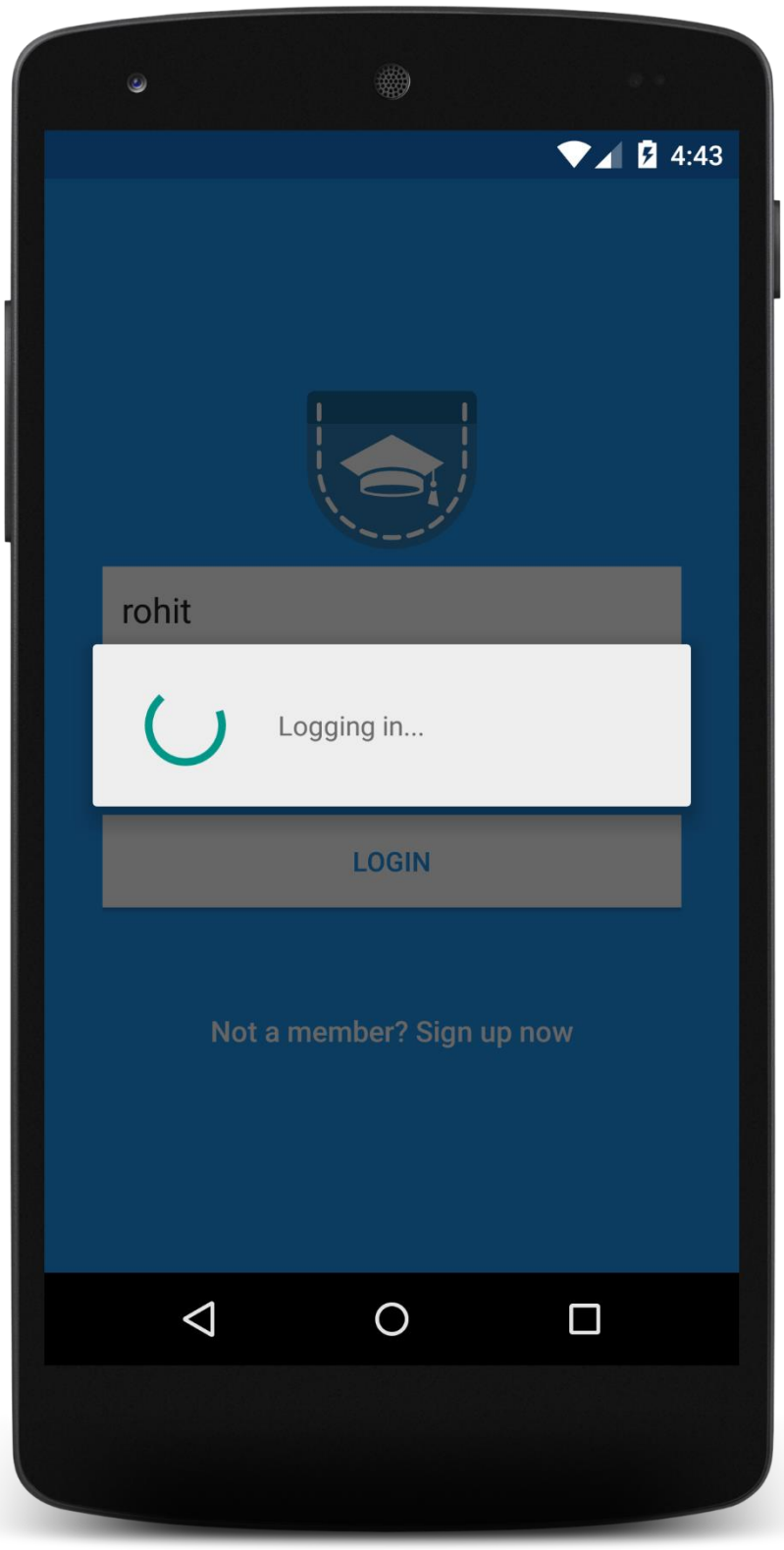
A **splash screen** is a graphical control element consisting of window containing an image, a logo and the current version of the software. A splash screen usually appears while a game or program is launching



Login Screen

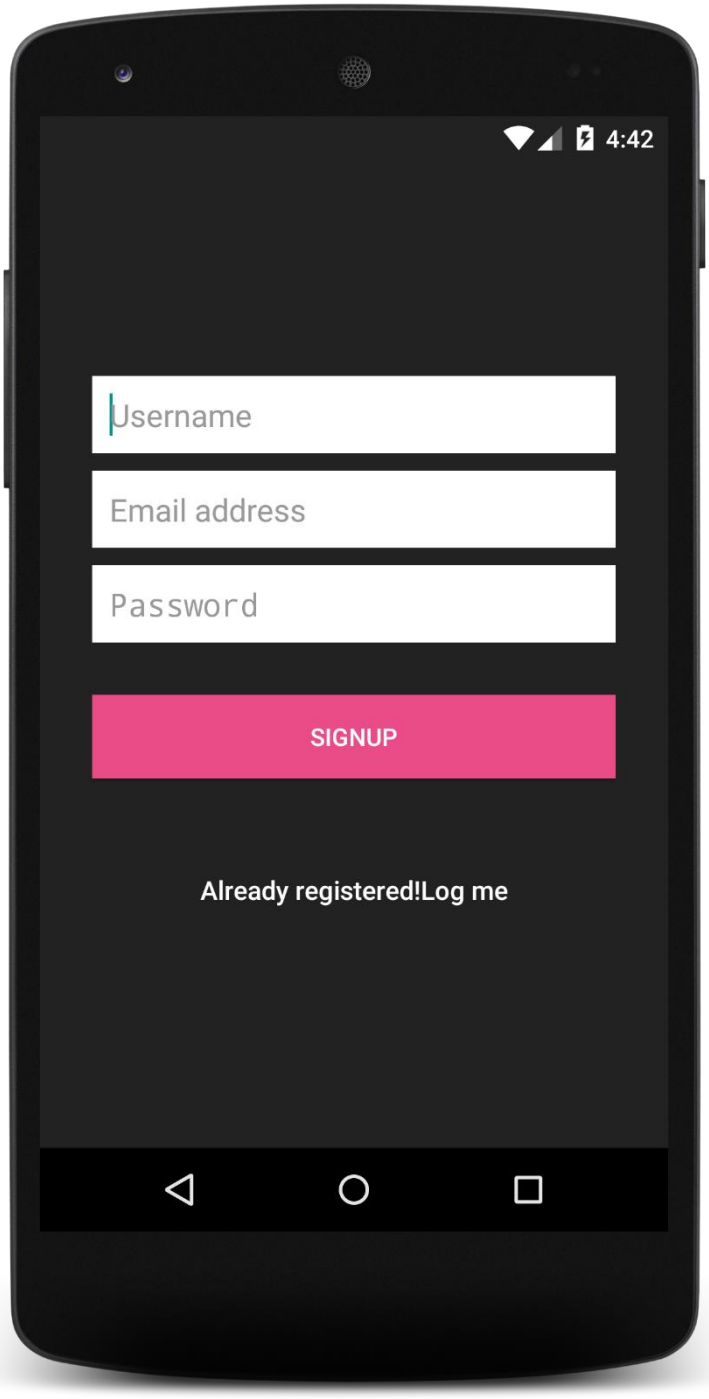
User can **Login** into app by entering Username and Password

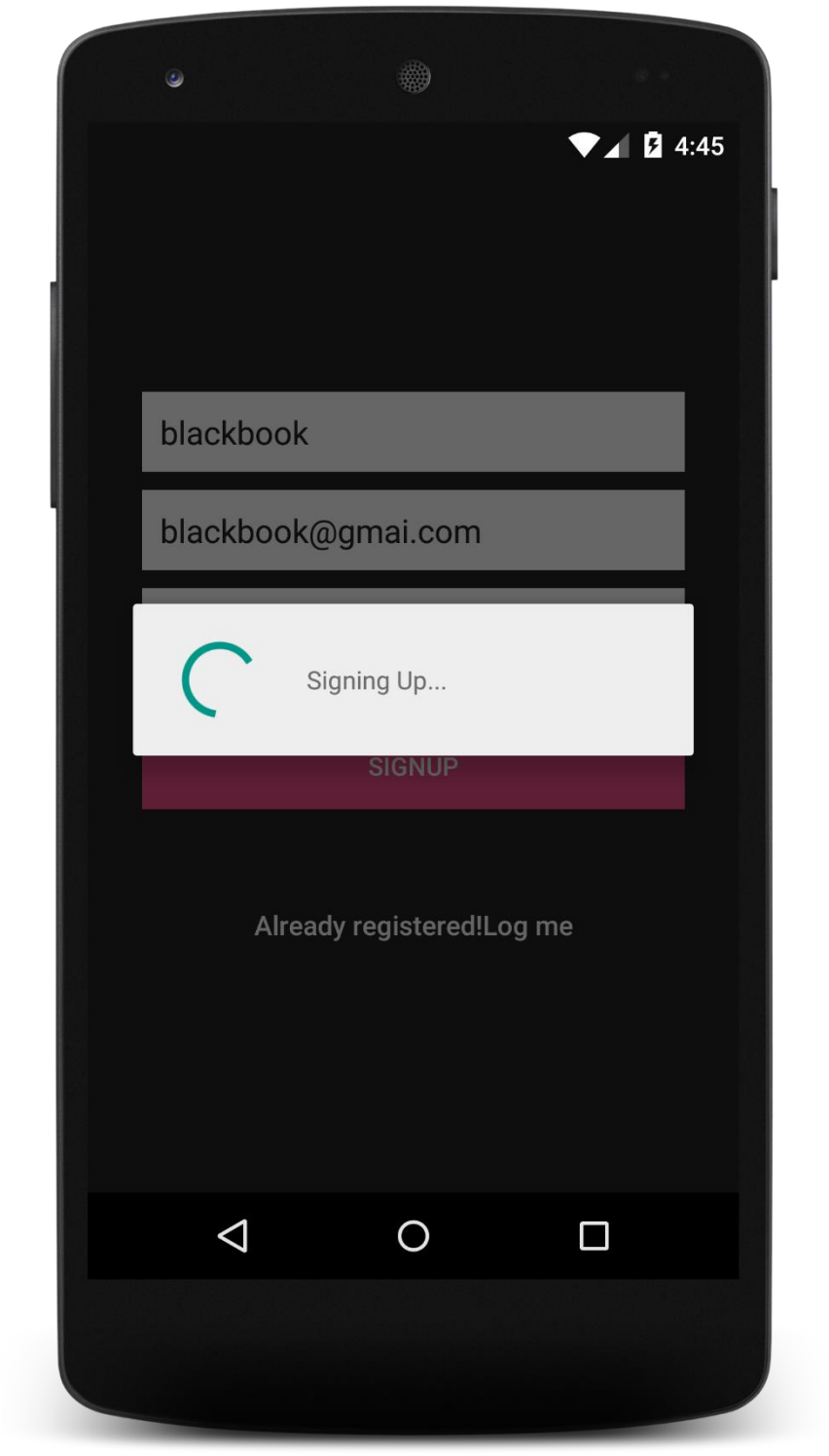



4:43
rohit
Logging in...
LOGIN
Not a member? Sign up now

Signup Screen:

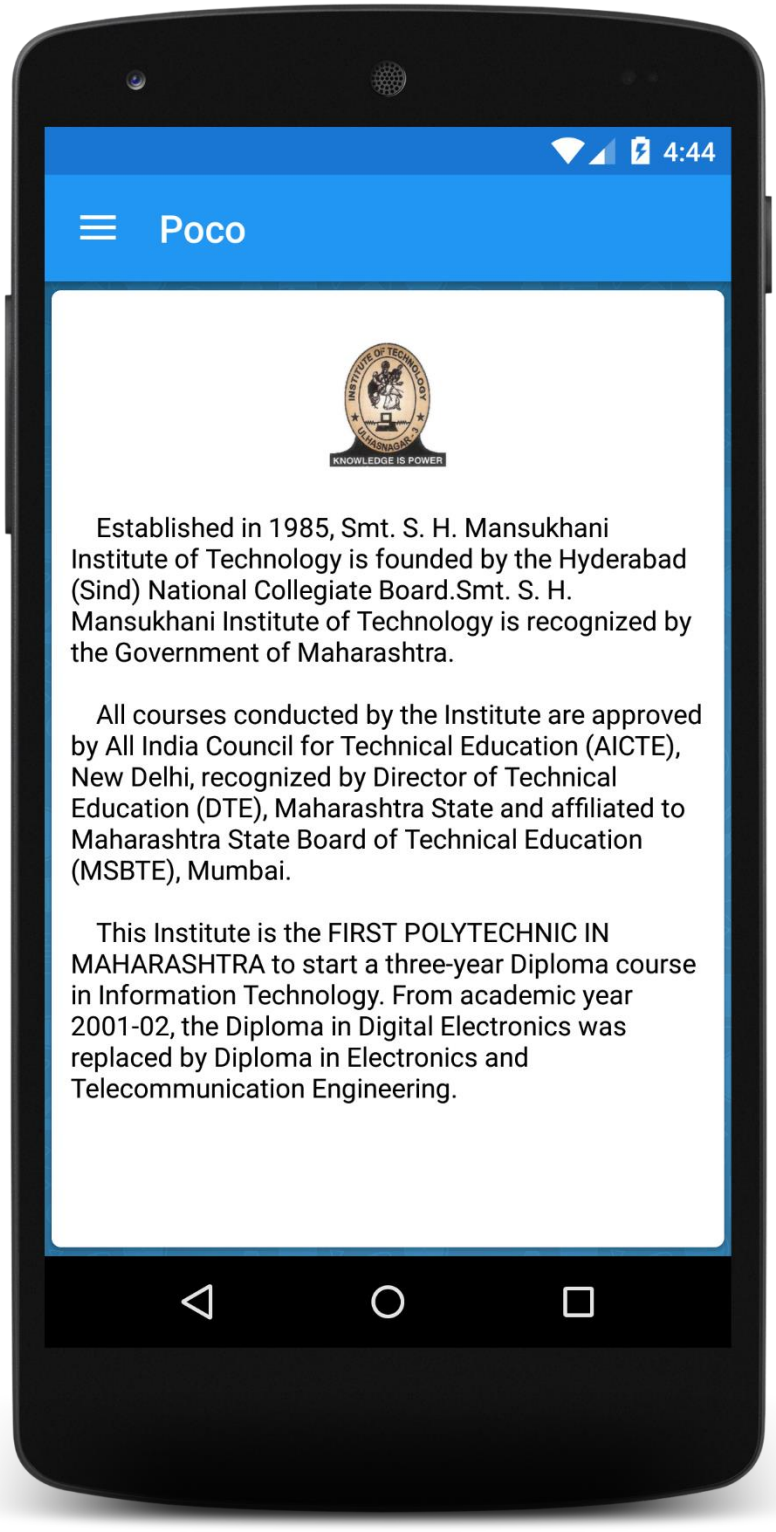
User must enter username, correct email id followed by **@** and password. Following these steps, User account is successfully created.




4:45
blackbook
blackbook@gmai.com
Signing Up...
SIGNUP
Already registered!Log me

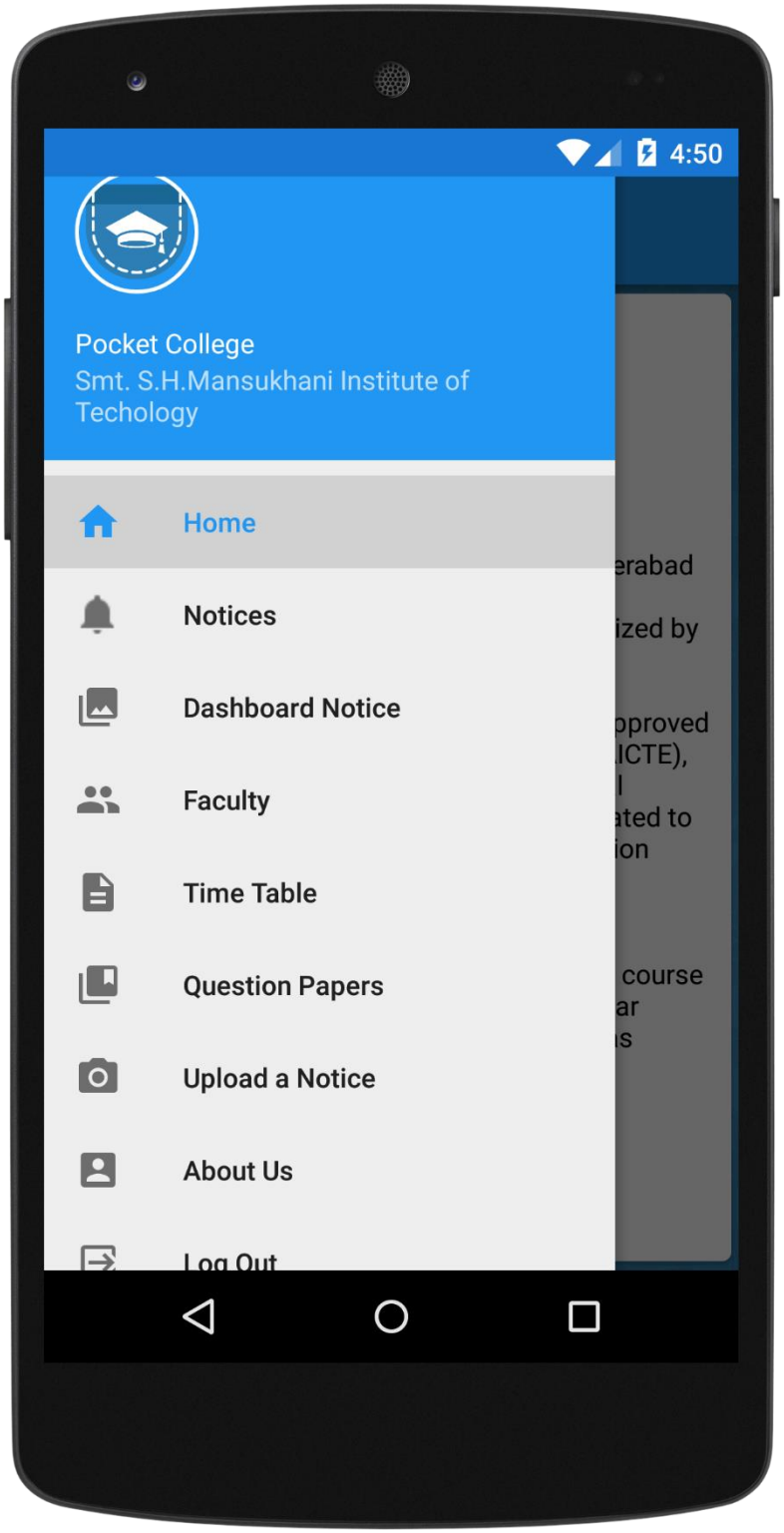
Home Screen

Home screen displays **College Information.**



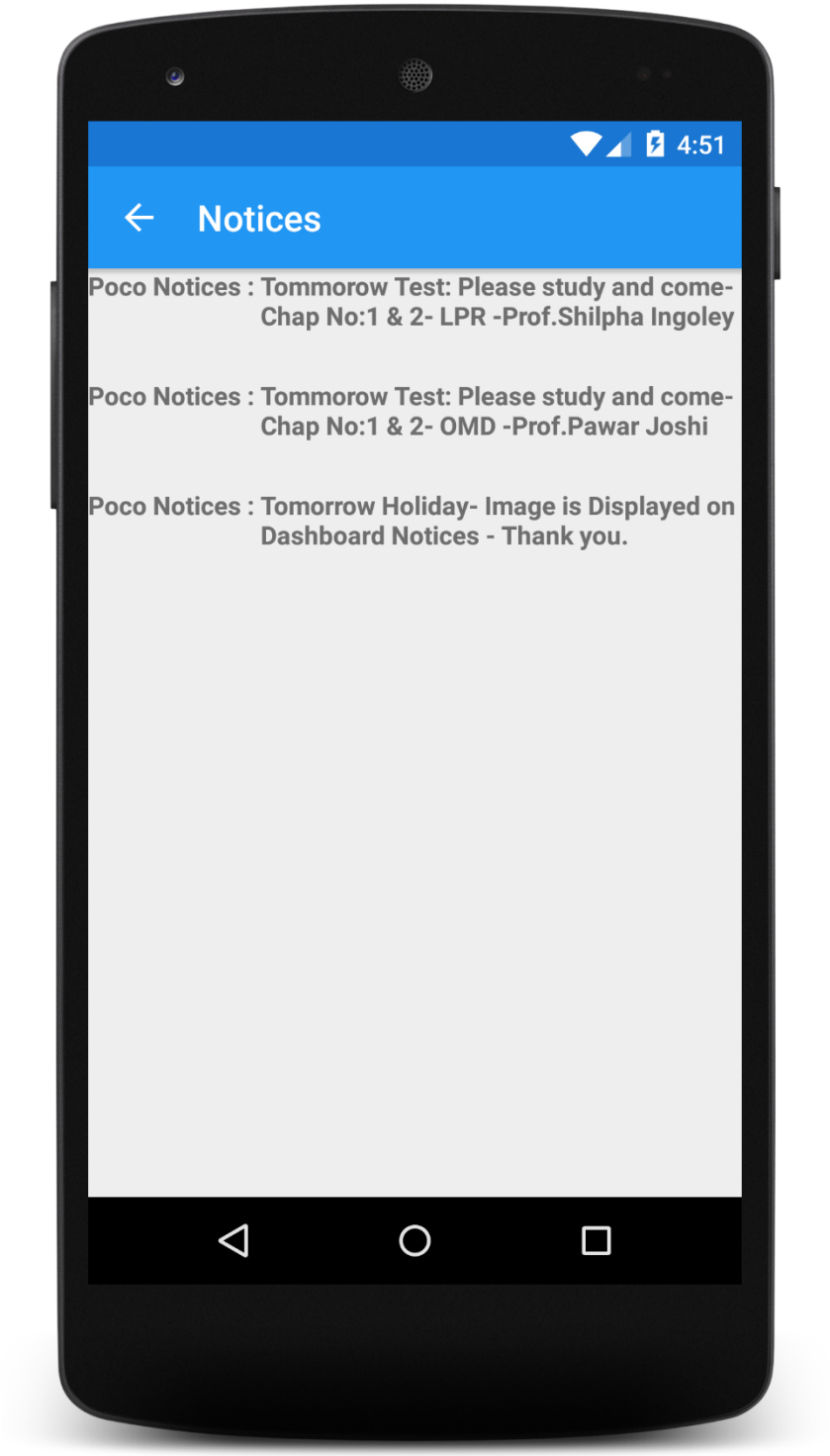
Navigation Drawer

Navigation Drawer contains **Drawer Items**:



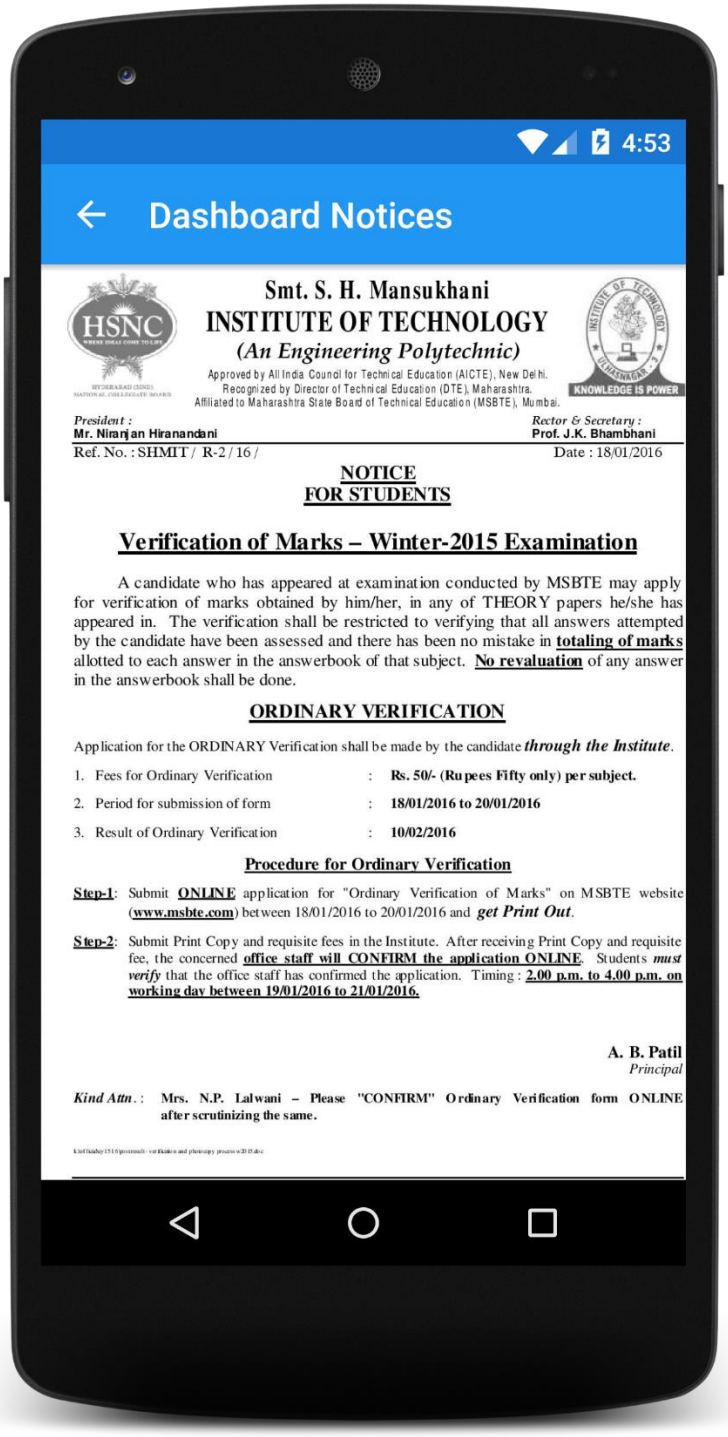
Notices Screen

Notices Screen will display **Notices** in List view **Text Format**. All data is retrieved from **Parse server**.



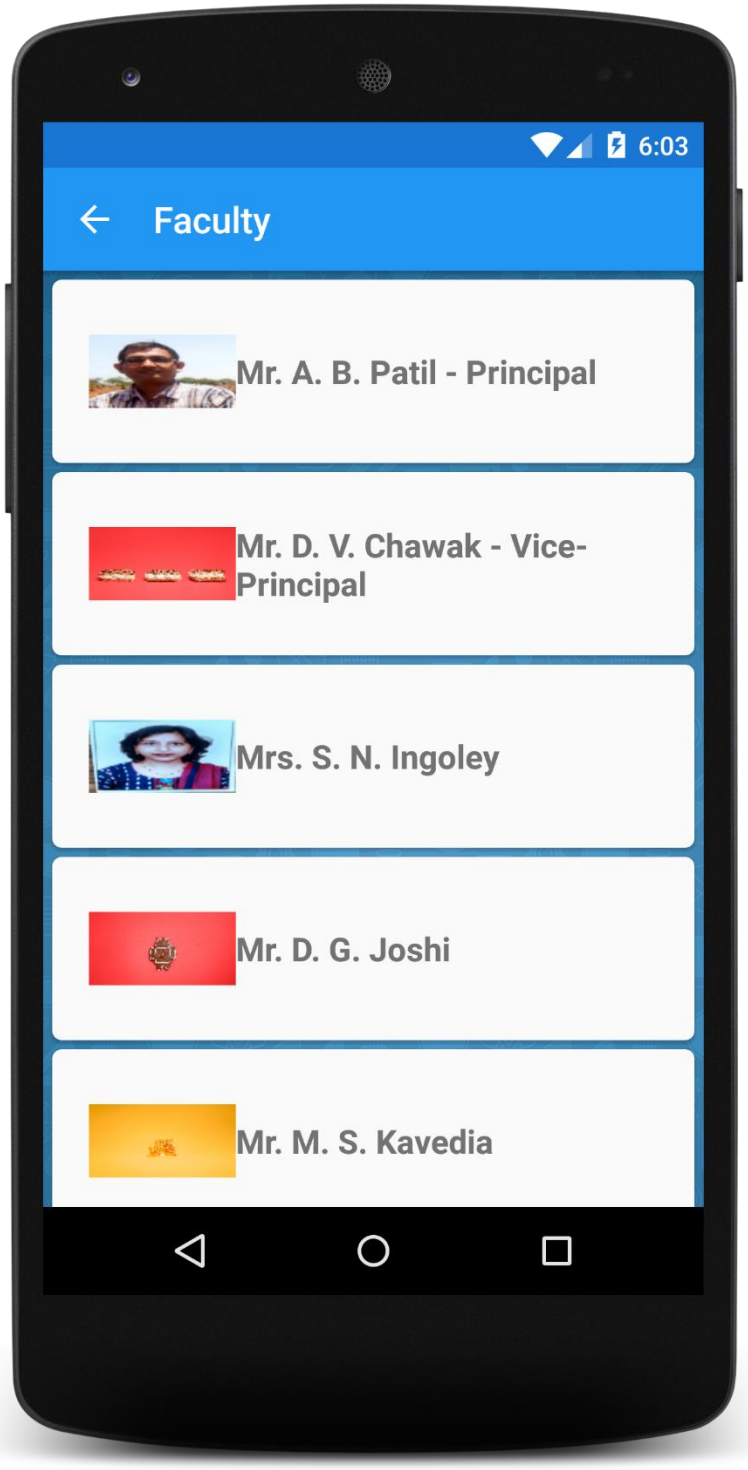
Dashboard Notices

Dashboard notices will display all the **Notices** in List View **Image Format.** All data is retrieved from **Parse server**.



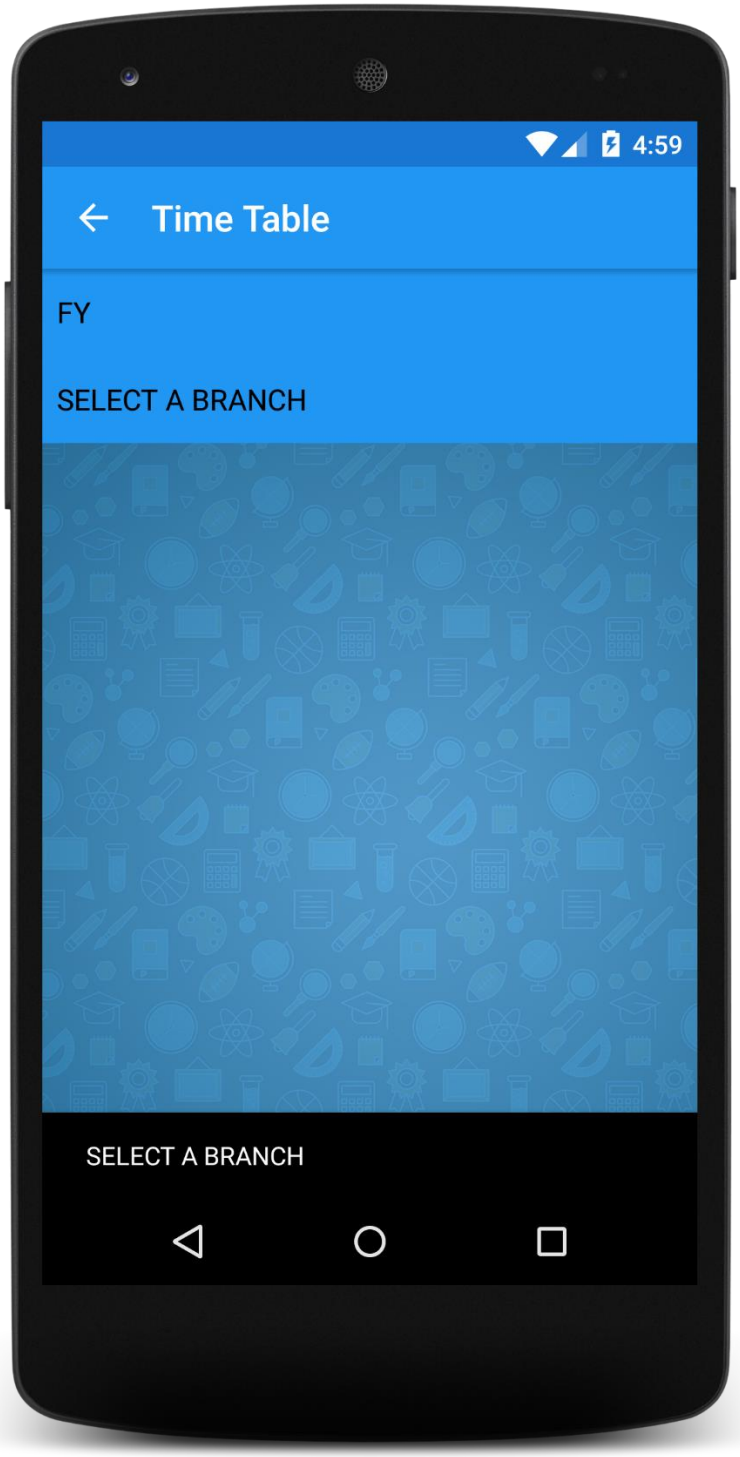
Faculty Screen

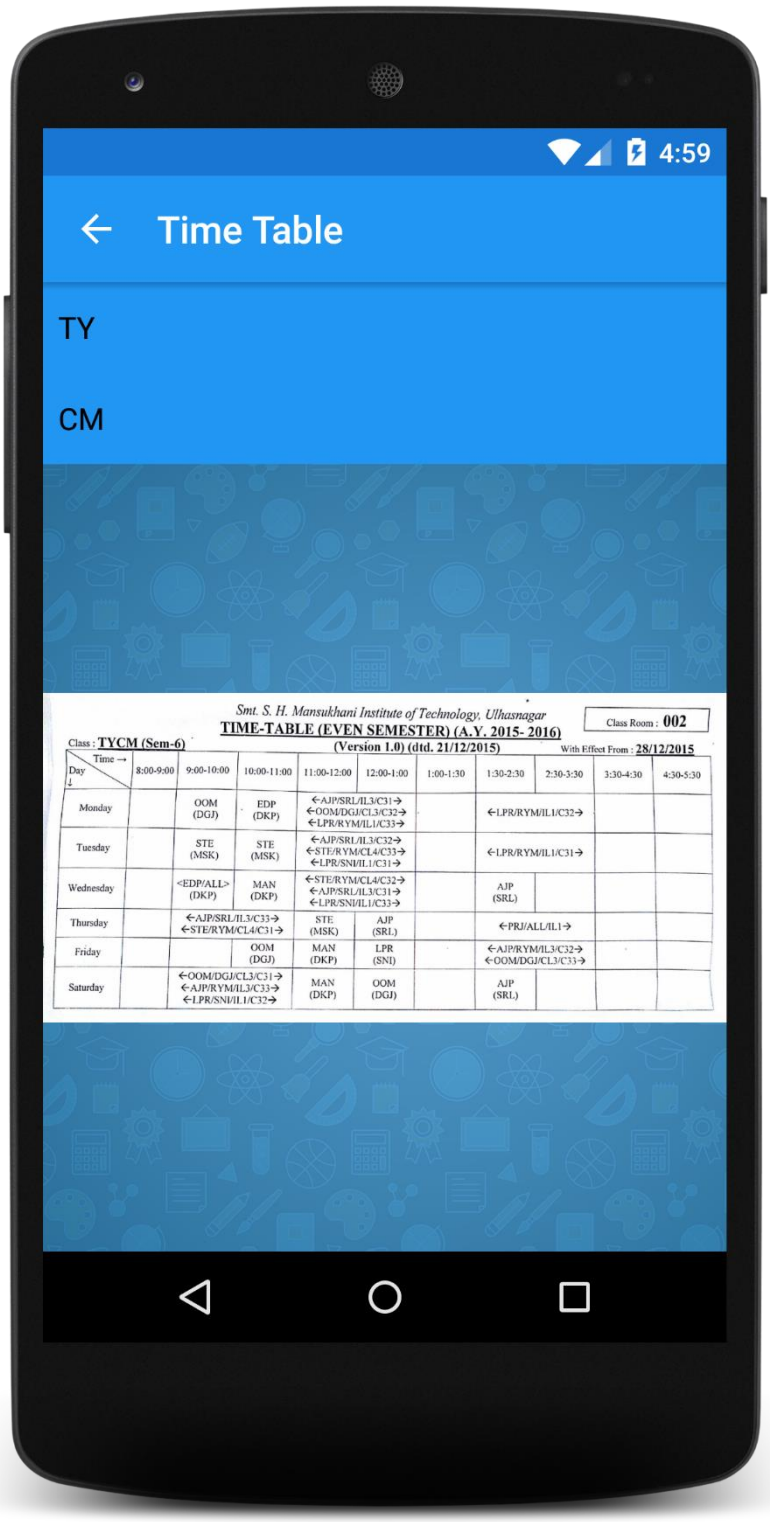
Faculty Screen will display **faculties** names in List View with **Image**.



Timetable Screen:

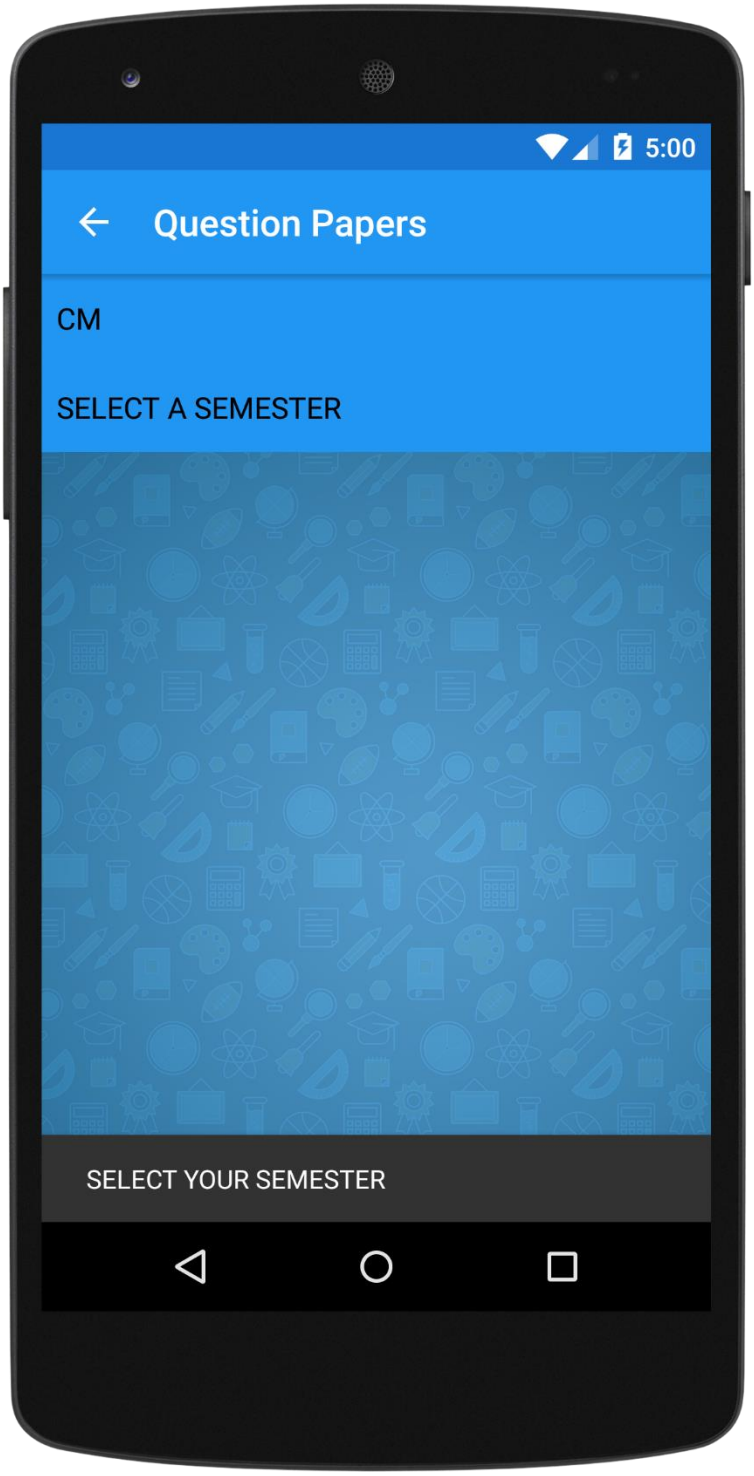
Timetable screen will display **Timetable** of specific **Year and Branch Selected**.

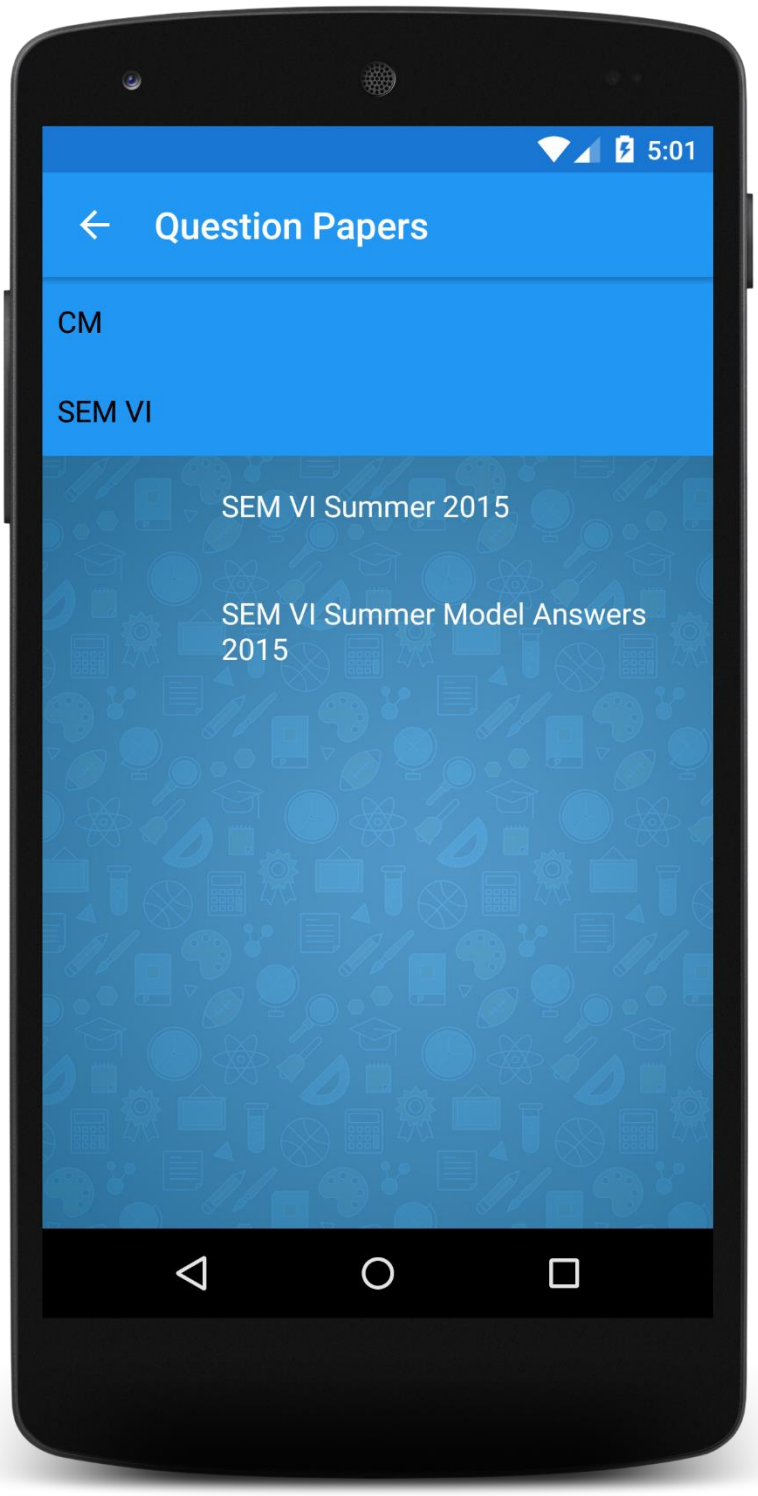


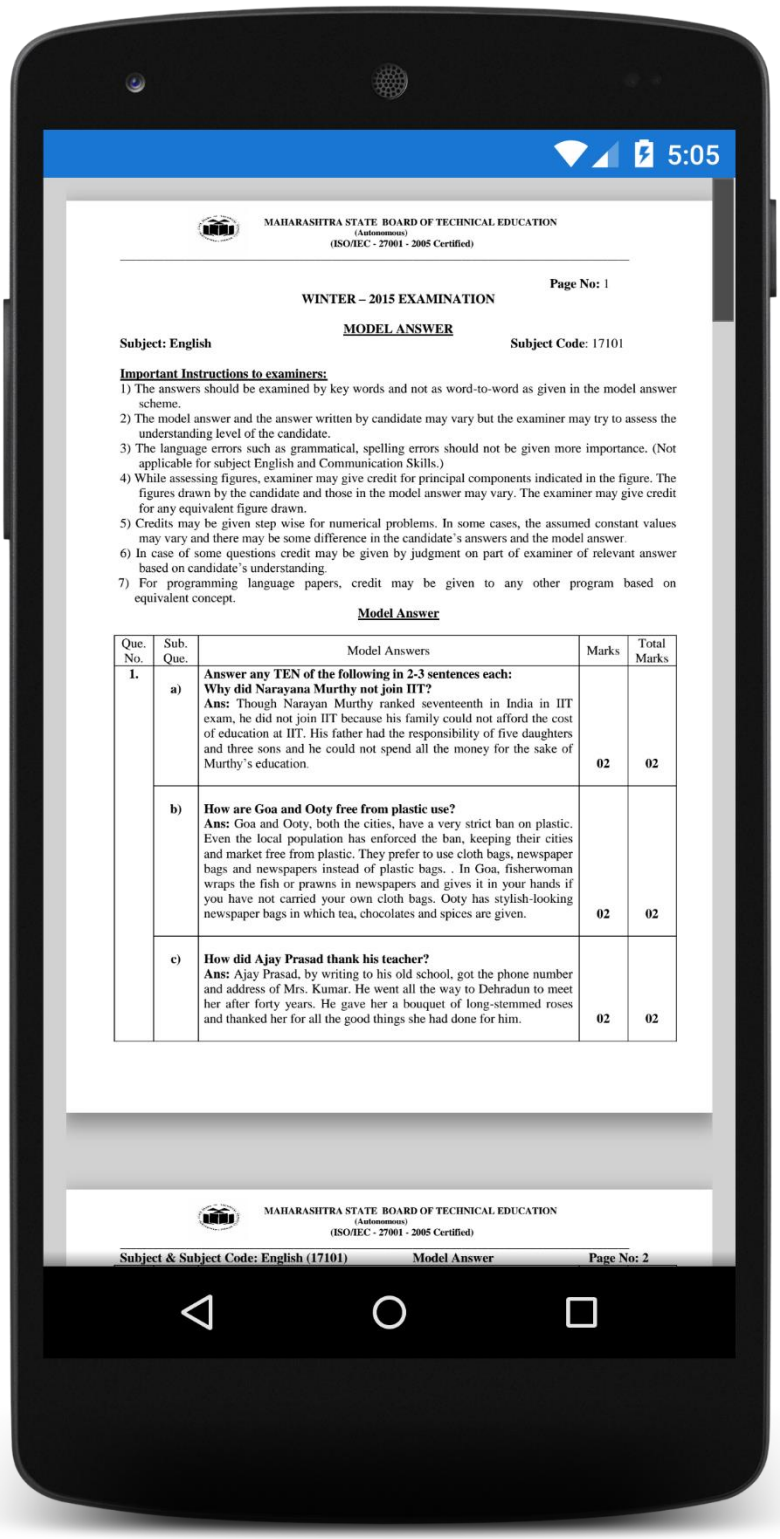

4:59
Time Table
TY
CM
Smt. S. H. Mansukhani Institute of Technology, Ulhasnagar
TIME-TABLE (EVEN SEMESTER) (A.Y. 2015- 2016)
Class Room : 002
Class : TYCM (Sem-6)
(Version 1.0) (dtd. 21/12/2015)
With Effect From : 28/12/2015
Time → / Day ↓
8:00-9:00
9:00-10:00
10:00-11:00
11:00-12:00
12:00-1:00
1:00-1:30
1:30-2:30
2:30-3:30
3:30-4:30
4:30-5:30
Monday
OOM (DGJ)
EDP (DKP)
←AJP/SRL/IL3/C31→ ←OOM/DGJ/CL3/C32→ ←LPR/RYM/IL1/C33→
←LPR/RYM/IL1/C32→
Tuesday
STE (MSK)
STE (MSK)
←AJP/SRL/IL3/C32→ ←STE/RYM/CL4/C33→ ←LPR/SNI/IL1/C31→
←LPR/RYM/IL1/C31→
Wednesday
<EDP/ALL> (DKP)
MAN (DKP)
←STE/RYM/CL4/C32→ ←AJP/SRL/IL3/C31→ ←LPR/SNI/IL1/C33→
AJP (SRL)
Thursday
←AJP/SRL/IL3/C33→ ←STE/RYM/CL4/C31→
STE (MSK)
AJP (SRL)
←PRJ/ALL/IL1→
Friday
OOM (DGJ)
MAN (DKP)
LPR (SNI)
←AJP/RYM/IL3/C32→ ←OOM/DGJ/CL3/C33→
Saturday
←OOM/DGJ/CL3/C31→ ←AJP/RYM/IL3/C33→ ←LPR/SNI/IL1/C32→
MAN (DKP)
OOM (DGJ)
AJP (SRL)

Question Paper Screen:

Question Paper screen will display **Question Paper** of specific **Branch and Semester Selected.**




5:01
Question Papers
CM
SEM VI
SEM VI Summer 2015
SEM VI Summer Model Answers 2015



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

Page No: 1

WINTER – 2015 EXAMINATION

MODEL ANSWER

Subject: English **Subject Code**: 17101

Important Instructions to examiners:

1) The answers should be examined by key words and not as word-to-word as given in the model answer scheme.
2) The model answer and the answer written by candidate may vary but the examiner may try to assess the understanding level of the candidate.
3) The language errors such as grammatical, spelling errors should not be given more importance. (Not applicable for subject English and Communication Skills.)
4) While assessing figures, examiner may give credit for principal components indicated in the figure. The figures drawn by the candidate and those in the model answer may vary. The examiner may give credit for any equivalent figure drawn.
5) Credits may be given step wise for numerical problems. In some cases, the assumed constant values may vary and there may be some difference in the candidate's answers and the model answer.
6) In case of some questions credit may be given by judgment on part of examiner of relevant answer based on candidate's understanding.
7) For programming language papers, credit may be given to any other program based on equivalent concept.

Model Answer

Que. No.	Sub. Que.	Model Answers	Marks	Total Marks
1.	a)	**Answer any TEN of the following in 2-3 sentences each:** **Why did Narayana Murthy not join IIT?** **Ans:** Though Narayan Murthy ranked seventeenth in India in IIT exam, he did not join IIT because his family could not afford the cost of education at IIT. His father had the responsibility of five daughters and three sons and he could not spend all the money for the sake of Murthy's education.	**02**	**02**
	b)	**How are Goa and Ooty free from plastic use?** **Ans:** Goa and Ooty, both the cities, have a very strict ban on plastic. Even the local population has enforced the ban, keeping their cities and market free from plastic. They prefer to use cloth bags, newspaper bags and newspapers instead of plastic bags. . In Goa, fisherwoman wraps the fish or prawns in newspapers and gives it in your hands if you have not carried your own cloth bags. Ooty has stylish-looking newspaper bags in which tea, chocolates and spices are given.	**02**	**02**
	c)	**How did Ajay Prasad thank his teacher?** **Ans:** Ajay Prasad, by writing to his old school, got the phone number and address of Mrs. Kumar. He went all the way to Dehradun to meet her after forty years. He gave her a bouquet of long-stemmed roses and thanked her for all the good things she had done for him.	**02**	**02**

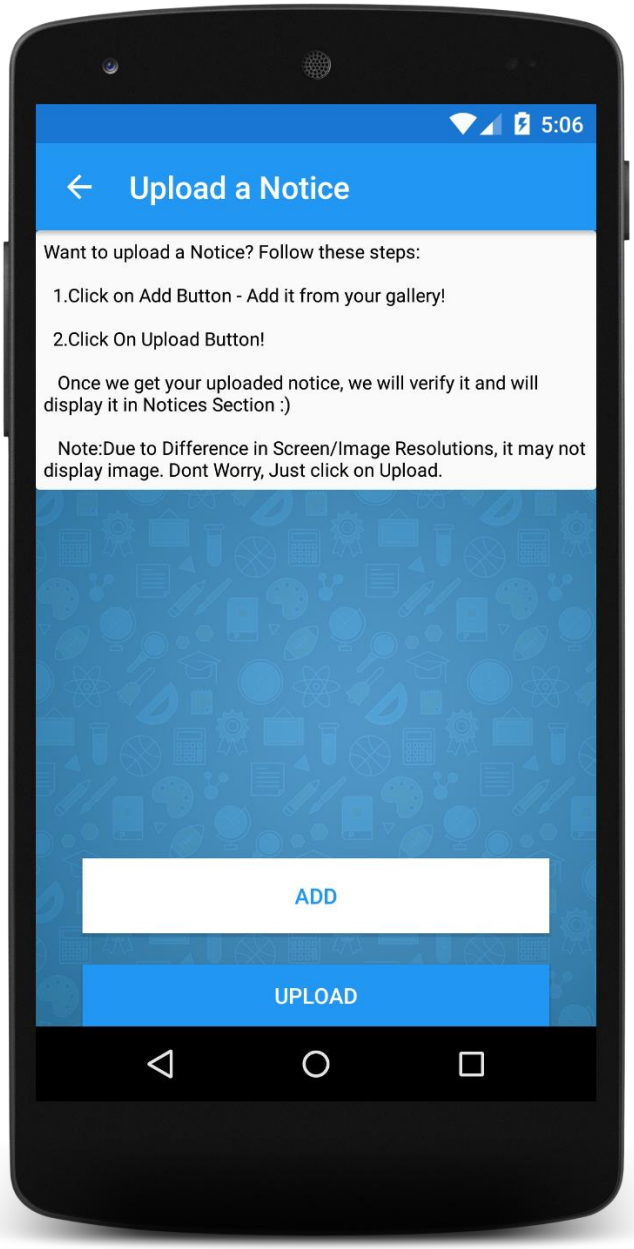
MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

Upload A Notice

Users can upload a notice directly from the app. The notice is uploaded to Parse Server. Once Admin verifies the notice:

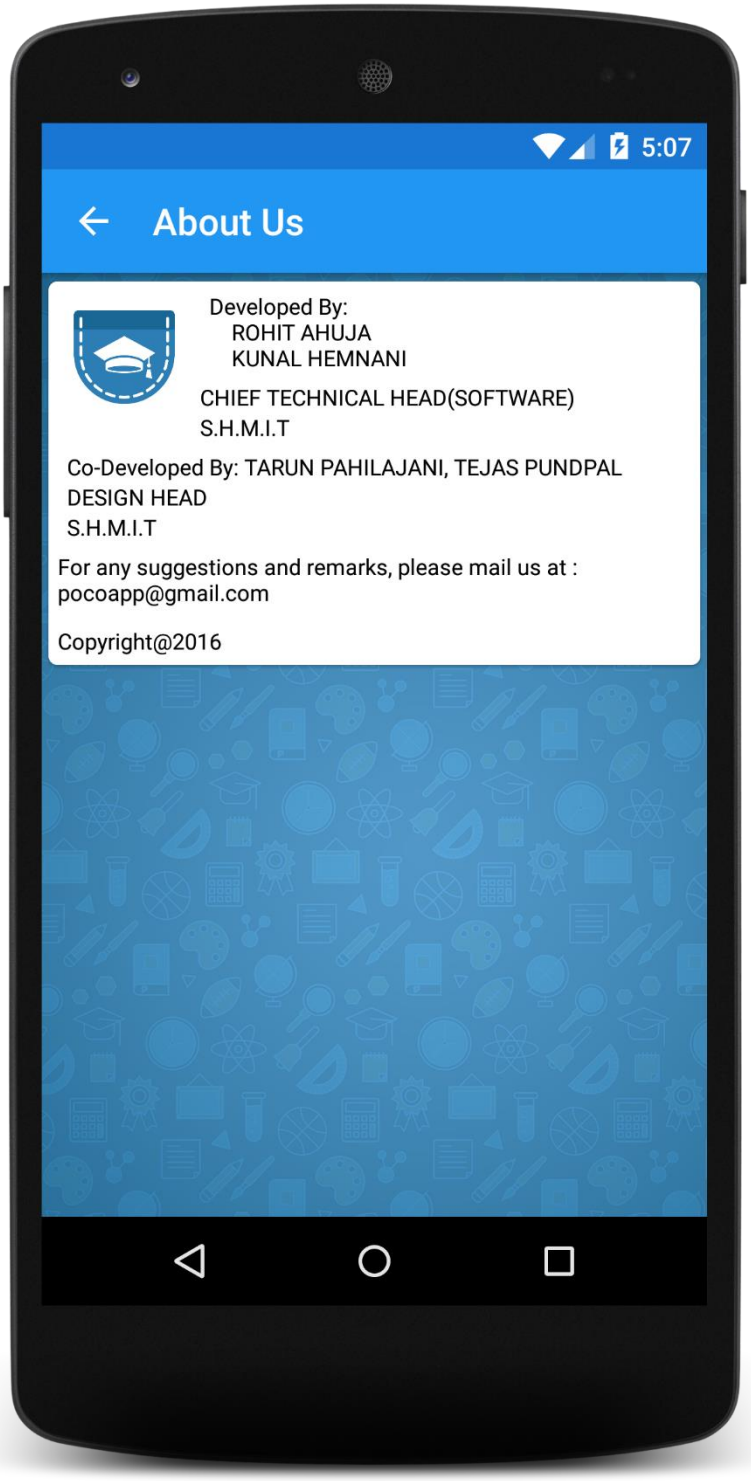
1.He/she will display it in Dashboard Notices

2.He/she will not display it in Dashboard Notices, if the Notice is not correct or its verification fails.



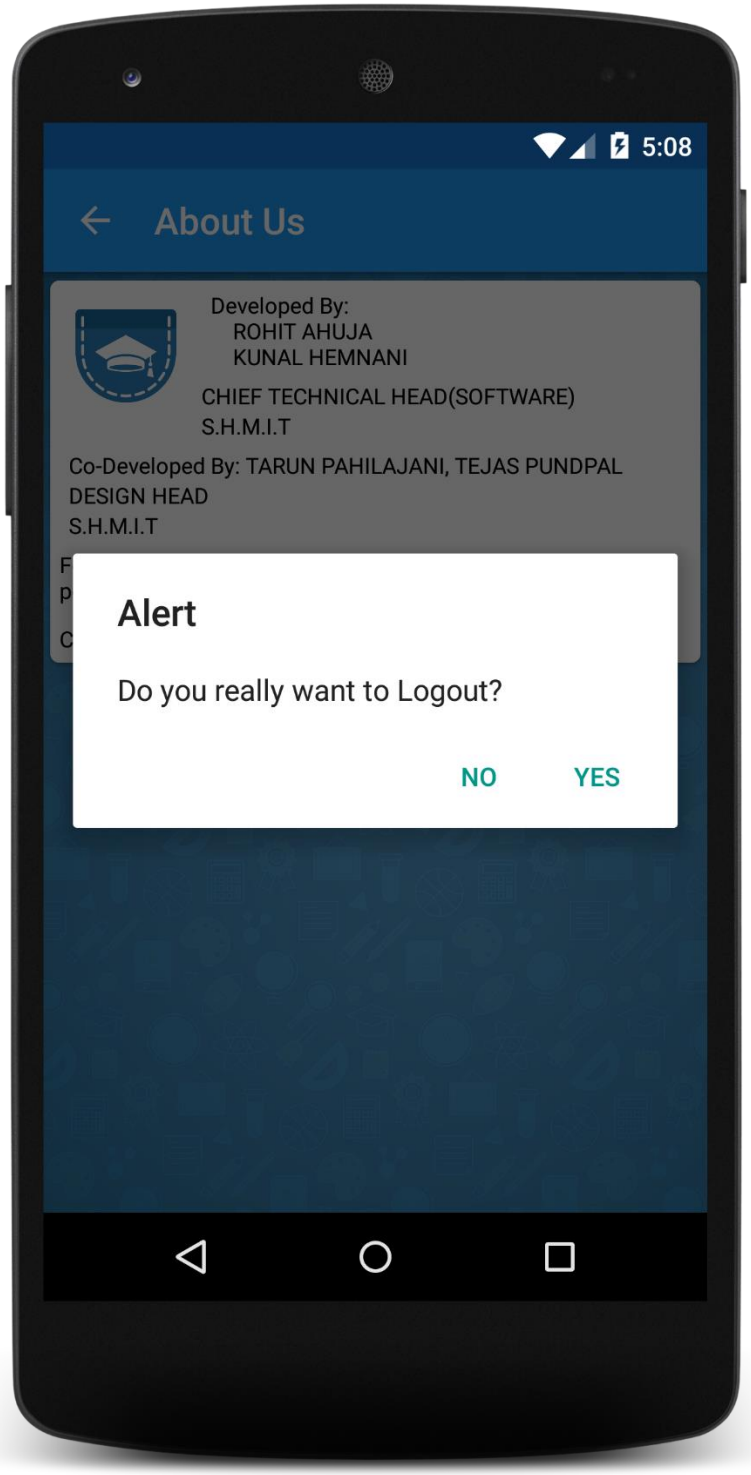
About us Screen

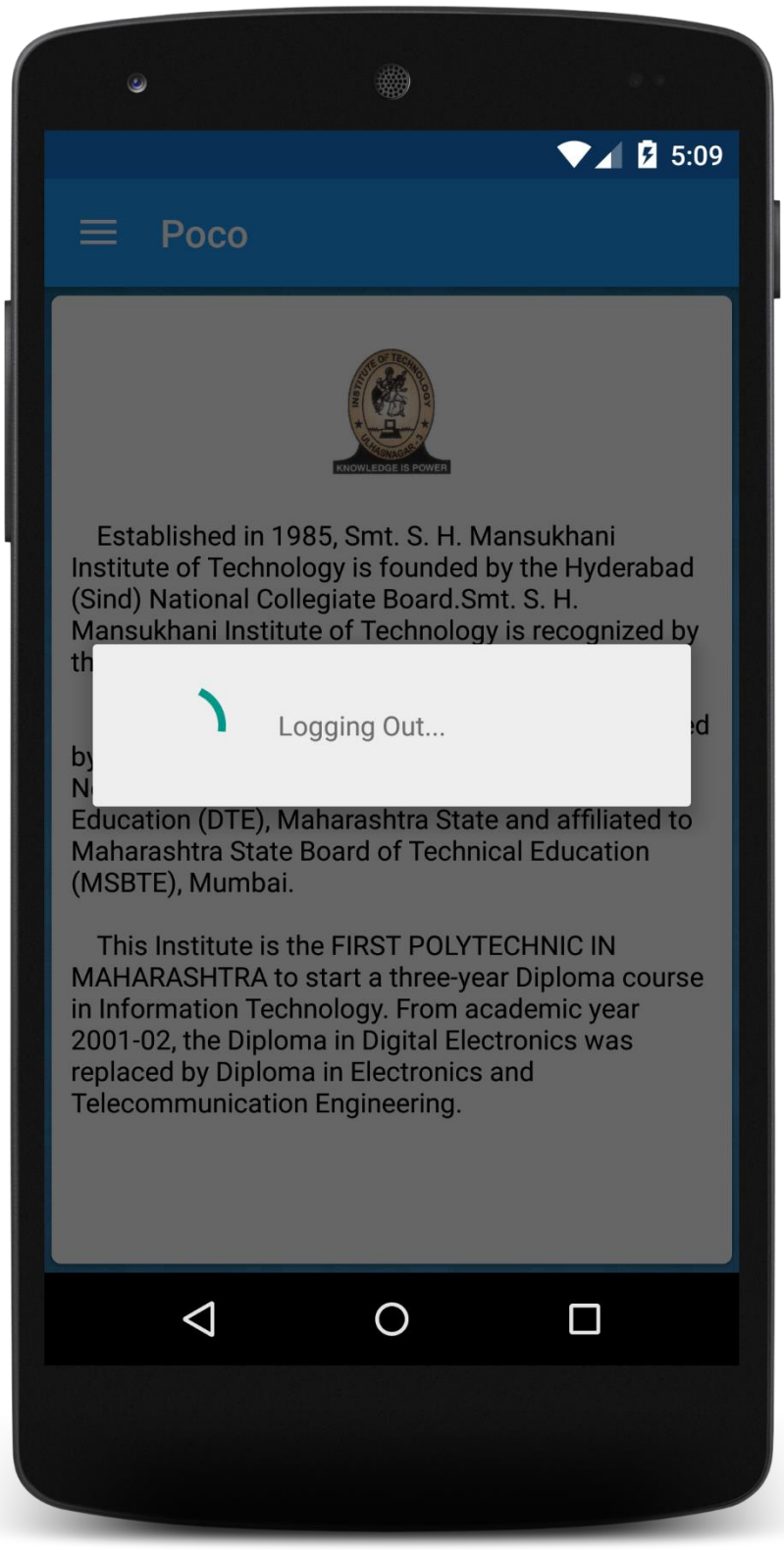
About Us Screen will display the names of the **Developers** of the App + **Contact Email ID.**

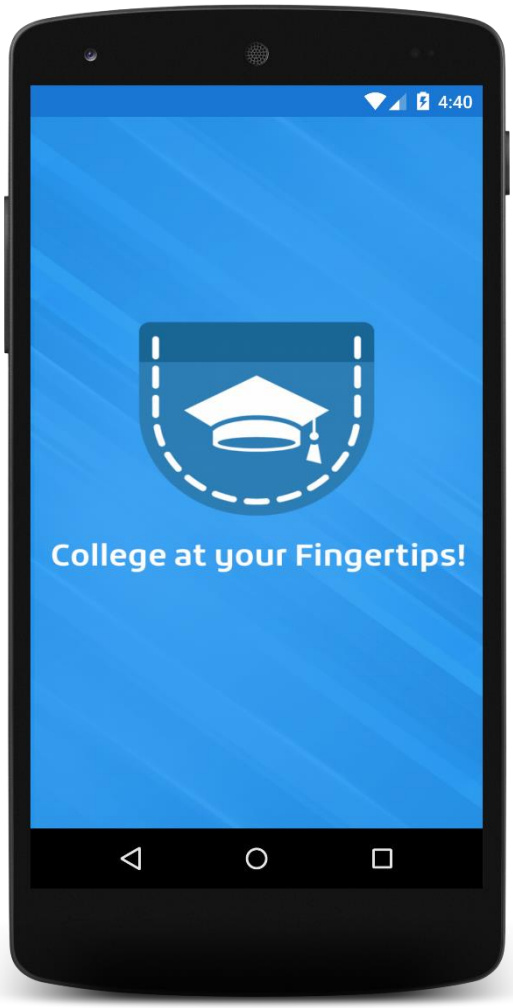


Logout Screen

Logout screen will **End session** at the App. To start again, user must **Login.**




5:09
Poco
KNOWLEDGE IS POWER
Established in 1985, Smt. S. H. Mansukhani Institute of Technology is founded by the Hyderabad (Sind) National Collegiate Board.Smt. S. H. Mansukhani Institute of Technology is recognized by th
Logging Out...
Education (DTE), Maharashtra State and affiliated to Maharashtra State Board of Technical Education (MSBTE), Mumbai.
This Institute is the FIRST POLYTECHNIC IN MAHARASHTRA to start a three-year Diploma course in Information Technology. From academic year 2001-02, the Diploma in Digital Electronics was replaced by Diploma in Electronics and Telecommunication Engineering.



PROJECT ANATOMY

After using Pocket College, if an attempt is made to anatomize the app, it can be easily partitioned into parts Stated below:

1.Creating Account – Sign Up.

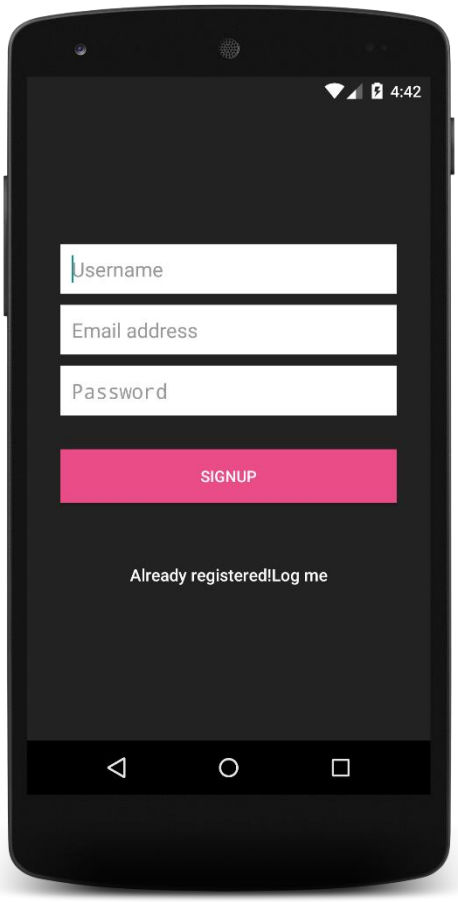
2.Logging into Account – Log In.

3.Viewing Timetable.

4.Searching for Question Papers.

5.Push Notifications, Notices, Dashboard Notices & Upload A Notice.

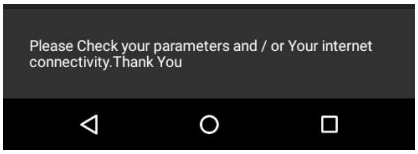
Creating Account – Sign Up



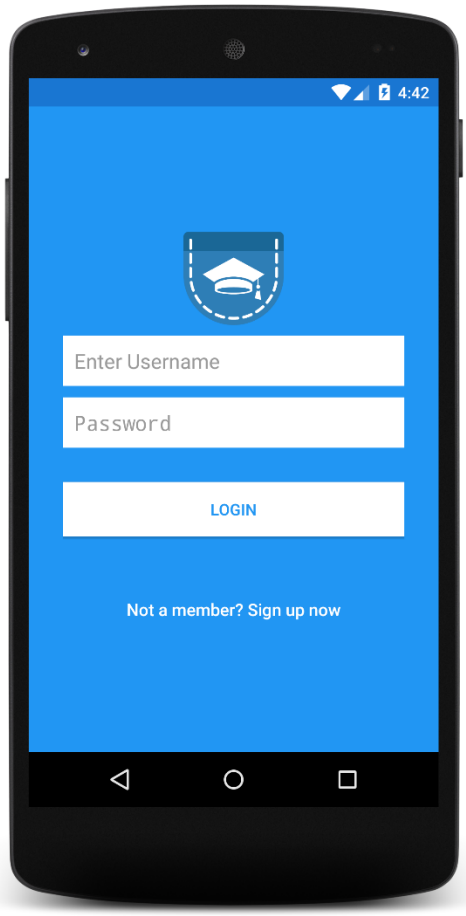
This Screen shot shows the most important part of the Pocket College app. i.e. Sign Up. It is dialog that takes in the input details of the user required by the app to start using app efficiently. It is important that the information recorded from this dialog is validated properly with all the constraints that ought to be present in the real world. All the data is saved and validated at Parse Backend Server. For instance,

- A user must enter appropriate Username, avoid using of any special Characters.
- A user must enter Email Address followed by the address specification i.e. @.
- A user must enter password recommended having special characters and symbols.

These constraints are valid in the rea world, every one follows this pattern If the user attempts to enter Email id without @; an error message will pop up at the bottom of the Screen.



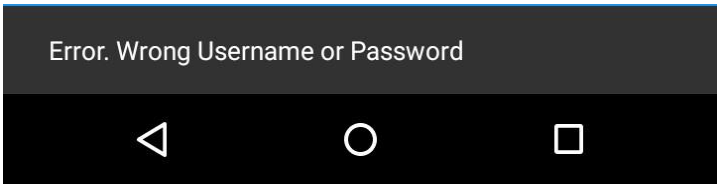
Logging into Account – Log In.



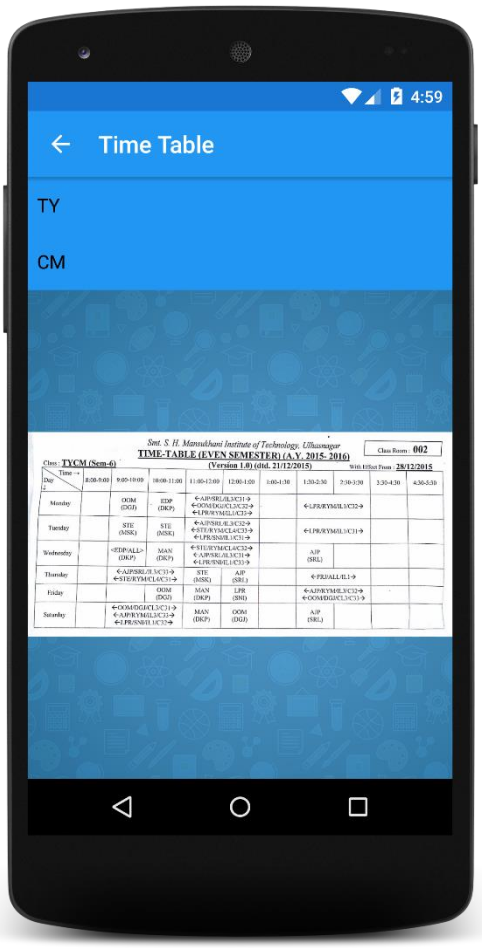
This Screen shot shows the 2nd most important part of the Pocket College app. i.e. Log In. It is dialog that takes in the input details of the user required by the app to start using app efficiently. It is important that the information recorded from this dialog is validated properly with all the constraints that ought to be present in the real world. All the data is retrieved and verified at Parse Backend Server. For instance,

- A user must enter his/her appropriate account Username
- A user must enter password of his/her appropriate Username that was entered while Signing Up.

These constraints are valid in the rea world, every one follows this pattern If the user attempts to enter wrong username or password; an error message will pop up at the bottom of the Screen.



Viewing Timetable



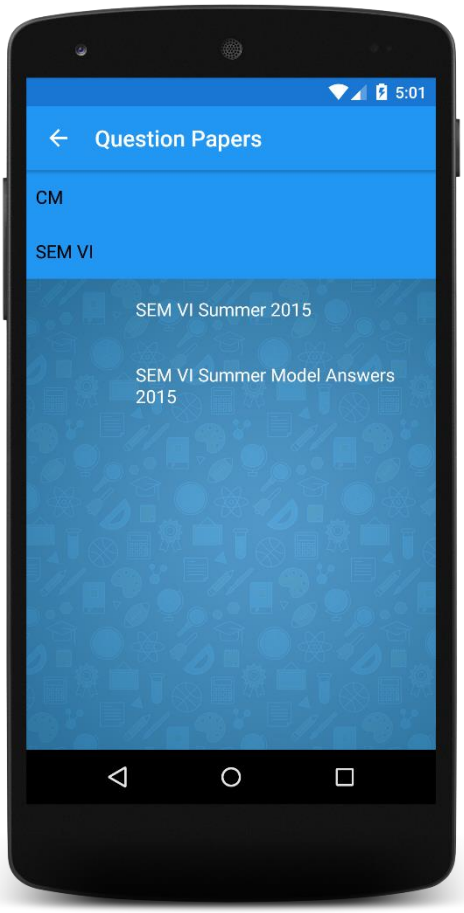
This activity shows the Timetable activity where the user can view the timetable of his/her class by selecting year and branch.

This activity is the most graphically rich activity within the app. This activity shows the timetable of user class when the user selects its year and branch; else it shows the Snack bar with tag “SELECT A BRANCH”.

The user can easily navigate between the timetable because of the Android Spinner. The user can even Zoom in or zoom out the timetable displayed with the help of the inbuilt feature of app called as Touch Image View.

If the user doesn’t select branch, app will display Snack bar:

Search for Question Papers

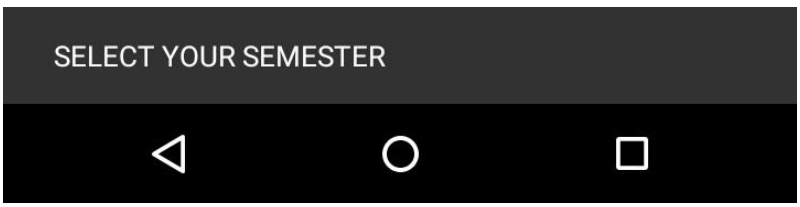


This activity shows the Timetable activity where the user can view the question papers and model answers of his/her Semester by selecting Branch and Semester.

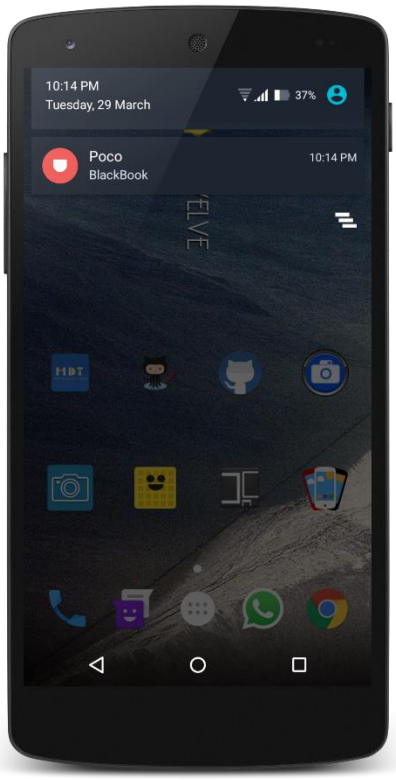
This activity is also the most graphically rich activity within the app. This activity shows the Question paper and model answers of User's semester when the user selects its branch and semester; else it shows the Snack bar with tag "SELECT YOUR SEMESTER".

The user can easily navigate between the Question papers because of the Android Spinner. The user can even Zoom in or zoom out the question paper displayed with the help of the Google Docs activity launched in Web view, Users can even print directly from app provided printer has Wireless Feature.

If the user doesn't Semester, app will show snack bar:



Notices- Push Notifications and Dashboard Notices



Push Notifications

This activity shows the Push Notifications received onto the user's phone.

1.What is Push Notification?

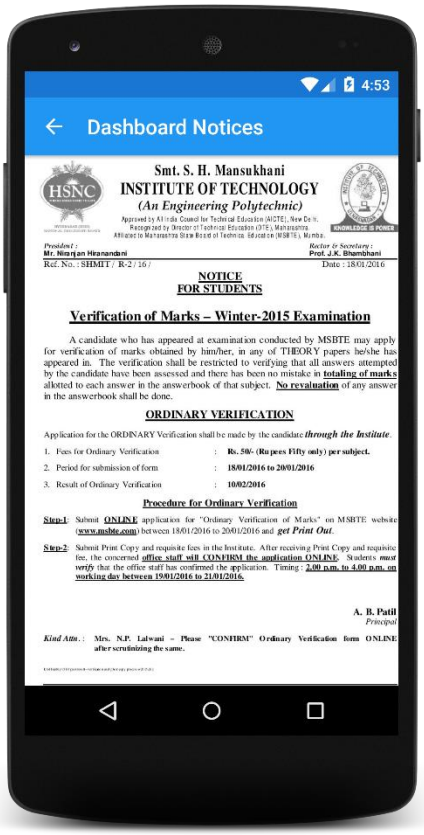
Push notifications let your application notify a user of new messages or events even when the user is not actively using your application. On Android devices, when a device receives a push notification, your application's icon and a message appear in the status bar.

Dashboard Notices

This activity shows the Notices and Dashboard Notice received onto the user's phone which are continuously synced with Parse.

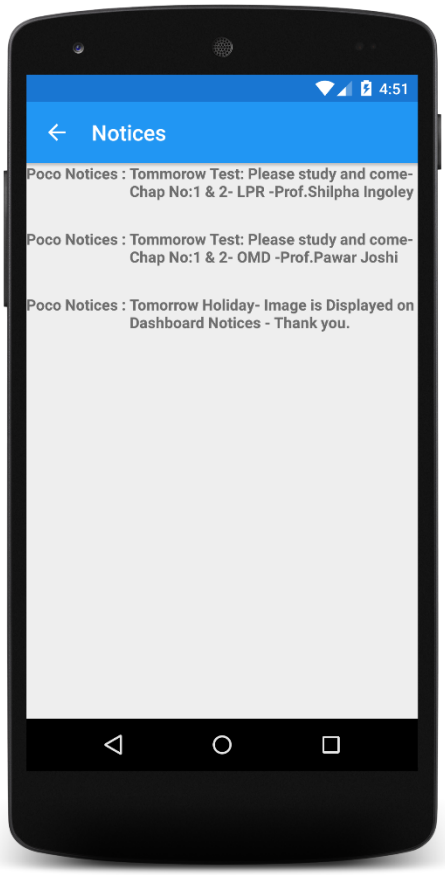
2.What are Dashboard Notices?

Image format of the Notices is Dashboard Notices. Dashboard Notices will display all the image notices E.g.: Fees structure for admission process.

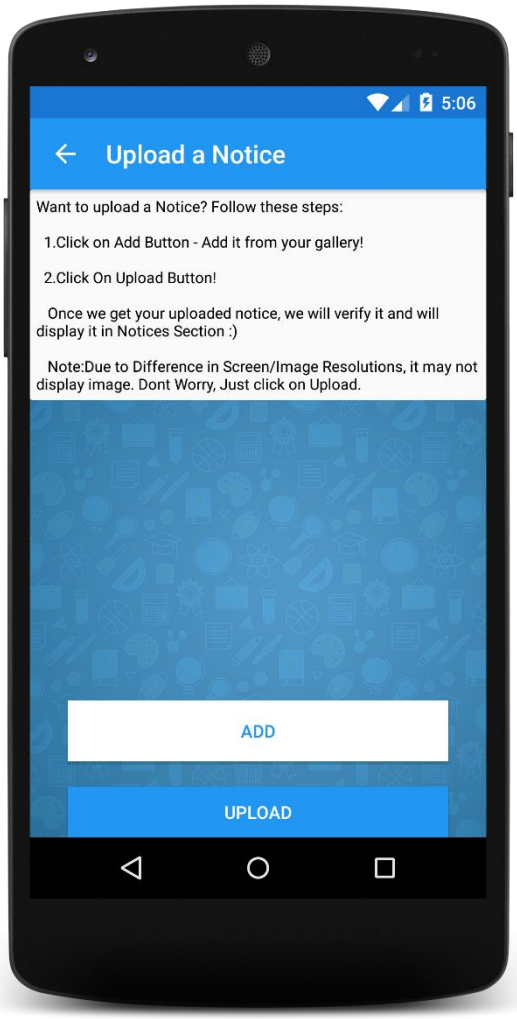


1.What is Notice?

Written or formal information, notification, or warning about something is a Notice. Notices Activity will display all the Written Notices. E.g.: Tomorrow Test.



Upload A Notice



This screenshot shows the activity in Pocket College where the user can Upload a Notice.

Steps:

1.Click on Add Button - Add it from your gallery!

2.Click On Upload Button!

Once admin gets your uploaded notice, admin will verify it and will display it in Notices Section :)

Note: Due to Difference in Screen/Image Resolutions, it may not display image. Don’t Worry, just click on Upload."

MINIMUM REQUIREMENTS

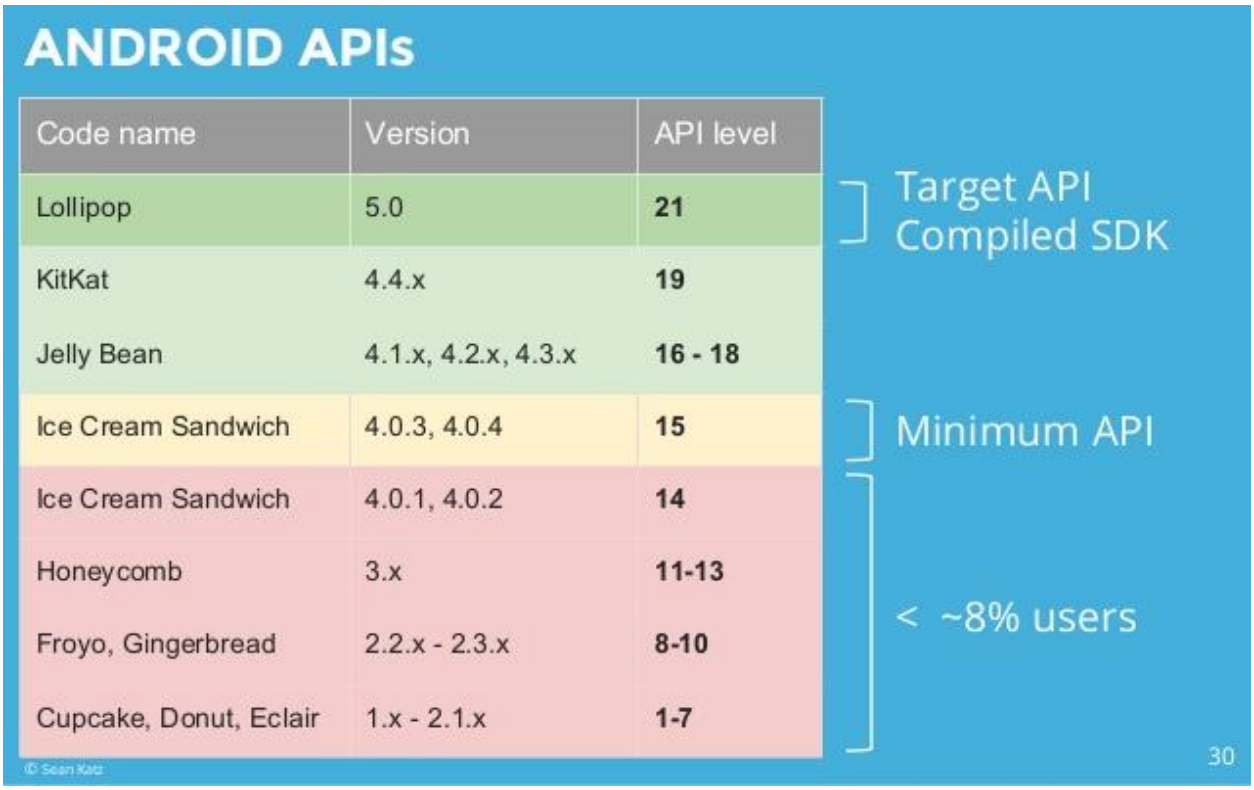
POCO Pocket College is intended to be helpful tool to a teacher and engineered so at to squeeze in the user's device without acquiring a whole lot of memory, processing power and excessive graphics rendering.

It is a fairly small app in comparison with its contemporary apps. So all that functionality is provided to the users using different Android versions at an average of 10 MB as per the size comparison table given below. This is the initial size of the app i.e. the size of the app when it is installed for the first time on user's device.

However, the size of the app varies according to the differences in the underlying hardware platform of the device and the Android version that is being used by the device viz. the Dalvik Runtime or the all new ART (Android Run Time).

Since ART is still an ongoing project as quoted by its Developers, it not might not fully optimize the size of the app to its fully optimize the size of apps installed on the user's device.

Since Android Version 5.1.1 is intended to work on devices with a minimal amount of processing capacity, its capable of optimizing the apps installed on the user's device in a better way than the other Android versions.



Pocket College requires the minimum API as 15 i.e. Jelly Bean. This means that Pocket College can work on every device being operated on Jelly Bean and above. This makes it available to a large chunk of Android users out there.

Minimum Requirements:

- Memory Space – Differs from device to device
- Other app for support – None
- Memory Card – Not Required
- RAM – 512 MB and above

The above specs give the minimum system requirements for Pocket College to be installed on a device and work properly. Whereas, this doesn't provide the user with a full-fledged user experience.

Recommended Requirements:

- Memory Space- 512MB
- Other Apps- Not Required
- Memory Card- Not Required
- Ram Recommended- 1GB

PROJECT STRUCTURE AND CLASS DESCRIPTION

As mentioned previously, the Android apps are all programmed using java and xml languages. The basic architecture of an android app goes like this: anything that is displayed on the screen at any particular instance is called as an activity. An activity consists of a java file and an optional xml file. These two files are linked together to generate one activity. There is no limit on the number of activities being created in an app, just that they finally have to be mentioned in the app's manifest file.

- **App Manifest**

Every Android app must have an AndroidManifest.xml file (with precisely that same) in its root directory. The manifest file represents essential information about your app to the Android system, information the system must have before it can run any of the app's code. Among other things, the manifest names the java package for the app.

The package name serves as unique identifier for the app. It describes the component of the app-the activities, services, broadcast receivers, and content providers that the app is composed of. It names the classes that implement each of the component and publishes their capabilities (for example, which

Intent message they can handle). These declarations let the android system know what the components are and under what conditions they can be launched. It determines which process will host app components.

It declares which permissions the app must have in order to access protected parts of the API and interact with other apps. It also declares the permissions that others are required to have in order to interact with the app's components.

It declares the minimum level of the Android API that the app requires.

- **Libraries**

A library is a collection of implementation of behavior written in terms of language, which has a well-defined interface by which the behavior is invoked. This means that as long as a higher level program uses a library to make system calls, it does not need to be re-written to implement those system calls over and over again.

In addition, the behavior is provided for reuse by multiple independent programs.

A program invokes the library-provided behavior via a mechanism of the language.

Every app is linked against the default Android library, which includes the basic packages, for building apps (With common classes such as Activity, Services, Intent, View, Button, Application, Content Provider, and so on).

- **SUPPORT LIBRARIES**

The Android Support Library package is a set of code libraries that provide backward-compatible versions of Android framework APIs as well as features that are only available through the library APIs. Each Support Library is backward-compatible to a specific Android API level.

This design means that your apps can use the libraries' feature and still be compatible with devices running Android 1.6(API level 4) and up. Including the support libraries in the Android project is considered a best practice for app developers, depending on the range of platform versions the app is targeting and the APIs that it uses.

Including the support Libraries in your Android Project is considered a best practice for app developers, depending on the range of platform versions your app is targeting and the APIs that it uses.

- **V4 SUPPORT LIBRARY**

This library is designed to be used with Android 1.6 (API level 4) and higher. It includes the largest set of APIs compared to the other libraries, including support for app components, user interface features, accessibility, data handling, network connectivity, and programming utilities. There are many others APIs included in this library.

This library is located in the <sdk>/extras/Android/support/v4/ directory after you download the android Support Libraries. This library does not contain user interface resources.

The only class that has been used in this app from this API is the Share Compact Intent Builder.

It just allows the app to send the report to the Dropbox app, Email apps and other such apps that can share that content.

SOME MORE IMPORTANT CLASSES

Android API is quite some lot rich in terms of classes. In the latest API of Android, there exist 3698 classes categorized in 197 different packages. Moreover, other libraries can be implemented in the project & many classes of java can also be used in addition.

The main classes that have been used in Pocket College are:

- Activity class
- Preference class
- View class
- Intent class
- Context class

Activity Class

The Activity class is an important part of an application's overall lifecycle, and the way activities are launched and put together is a part of the platform application model. An activity is a single, focused thing that the user can do.

Almost all activities interact with the user, so the Activity class takes care of creating a window for you in which you can place your UI with set Content View (View).

View Class

The View class represents the basic building block for user interface components. A View occupies a rectangular area on the screen and is responsible for drawing and event handling. View is the base class for widgets, which are used to create interactive UI components (buttons, text fields, etc.). All of the views in a window are arranged in a single tree.

One can add views either from code or by specifying a tree of views in one or more XML layout files. There are many specialized subclasses of views that act as controls or are capable of displaying text, images, or other content.

Context Class

The Context class acts as an interface to global information about an Android application's environment. This is an abstract class whose implementation is provided by the Android system.

It allows access to application-specific resources and classes, as well as up-calls for application-level operations such as launching activities, broadcasting and receiving intents, etc.

Preference Class

Provides classes that manage app preferences and implement the preference. Using these ensures that all

the preferences within each app are maintained in the same manner and the user experience is consistent with that of the system and other apps.

All settings made for a given Preference will be automatically saved to the app's instance of Shared Preferences. Access to the Shared Preferences is simple with get Shared Preferences.

Intent Class

An intent is an abstract description of an operation to be performed. It can be used with start Activity to launch an Activity, broad cast lntent to send it to any interested Broad Cast Receiver components, and start Service(Intent) or bind Service (Intent, Service Connection, int) to communicate with a background Service.

An Intent provides a facility for performing late runtime binding between the codes in different apps. Its most significant use is in the launching of activities, where it can be thought of as the glue between activities.

It is basically a passive data structure holding an abstract description of an action to be performed.

PROJECT LIMITATIONS

Although, it's a good idea to integrate the new technology in our day-to-day life, but that brings in a scope of risk, regarding the same. For some reason, even the Pocket College app faces few risks. Maximum risks have been tactfully handled by implementing few features in it. However, there are some risks that just are not possible to be taken care of.

- **'Clear Data' Option**

In general, one of the major risks for any app in Android is the 'Clear Data' option for that app in system settings. This option is usually used only when the data of a specific app is to be deleted permanently. In simple term, after this button is pressed, the corresponding app behaves as if it has been installed lately in the system and no one has opened it even once, after installation.

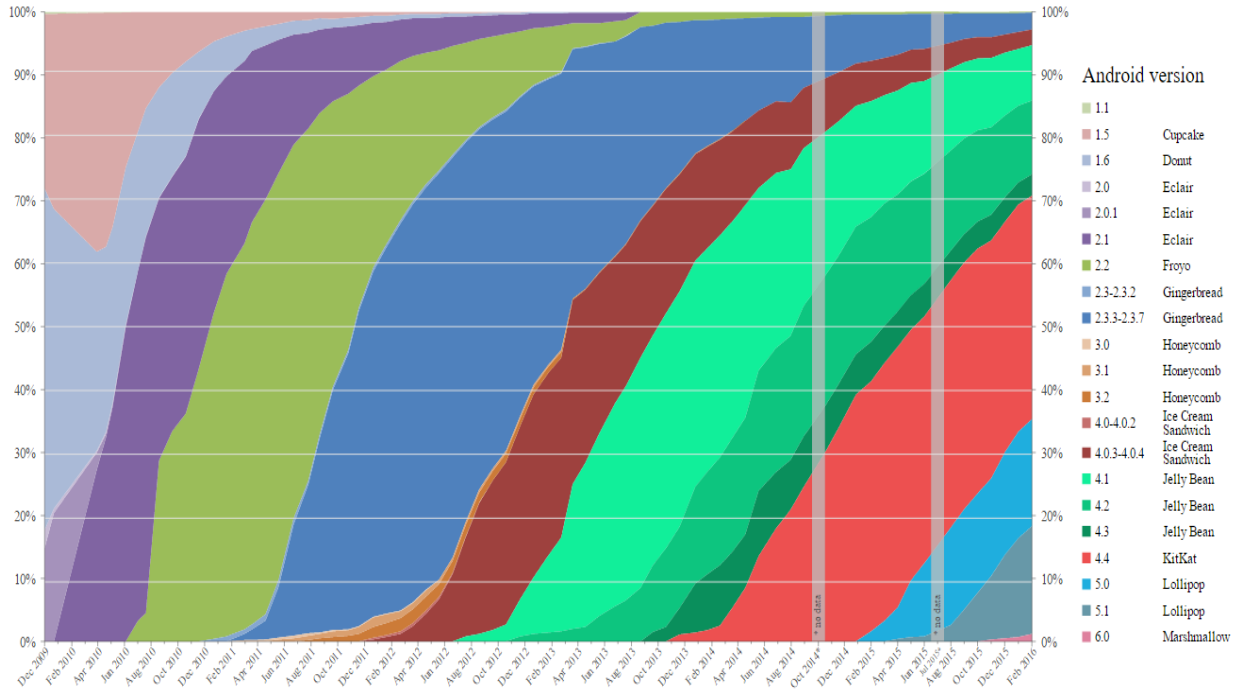
The Pocket College App faces this similar problem. Most of the times, a user is not exactly aware of what he/she is trying to attempt. If by chance, does a user click this option, the user will be logged out from his account, if user forgot Login username and password, this will create problem for the user.

But, user can contact Administrator to reset his/her password, by this problem will be solved. As, we are using Parse as a backend server there is no tension of Data loss. Data security is maintained in Parse.

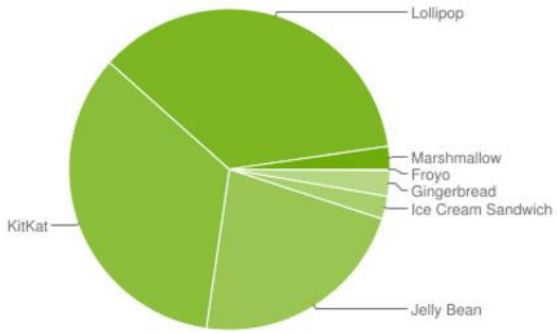
- **Minimum API Level**

Moreover, the app can only run on an Android device with an Operating System version of 4.0 (ICS) or above. The users having Honeycomb or lower versions will not be able to use this app.

However, the percentage of devices being operated on versions lower than Ice Cream Sandwich is comparatively quite less. The statistics below shows the Global Android version distribution since December 2010; as of March 2016.



Version	Codename	API	Distribution
2.2	Froyo	8	0.1%
2.3.3 - 2.3.7	Gingerbread	10	2.6%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	2.3%
4.1.x	Jelly Bean	16	8.1%
4.2.x		17	11.0%
4.3		18	3.2%
4.4	KitKat	19	34.3%
5.0	Lollipop	21	16.9%
5.1		22	19.2%
6.0	Marshmallow	23	2.3%



Hereby, it can be clearly noted that Android 4.4 KitKat is the most widely used Android version; operating on around 34% of Android devices worldwide. And that, the market share of rest of the Android versions below the Jelly Bean is merely 21.3%.

- **Miscellaneous**

Another risk that might be considered, is the risk of complete date loss, due to various problems like theft, memory corruption, OS or hardware issues, permanent phone damage, etc. Keeping all this in mind Pocket College App provides a feature letting the user to store the data is cloud (Parse**)**, as a secondary option of retrieval.

FUTURE ENHANCEMENTS

Keeping in mind the end-user review and the setbacks of the app, there is a lot of scope for it to improve. Below stated are few improvements that can be expected from it.

1. Introducing an option, of Web view activity which will display notices in card view and when user clicks on it, a web view should be launched. This will help our app to be more user-responsive module.

2. Providing a template for the data, so as the user can import the cache memory Data that was previously used by the app.

3. Providing different color themes for each Activity, to enhance the UI and for making the user know in which activity he/she is in.

4. Adding To-do List Activity, so that the user can write what they have to do. Similar to Google Notes App.

5. Adding support for iOS, as many users are using iOS.

CONCLUSION

Pocket College is an amazing app which helps student to complete their Studies hassle-free and smoothly.

From our side we tried to fulfil the Students requirements but it might be possible that some disadvantages can be present in our project. If any, we will try to overcome that disadvantages in the Future Enhancements.

Finally, we want to give the vote of thanks to Prof. Shilpa Ingoley (Our Project Guide), all teaching staff as well as my Friends who invested their valuable time for helping in our Project.

BIBLIOGRAPHY

Websites:

developer.android.com

https://developers.google.com/

http://stackoverflow.com/

http://www.vogella.com/

https://www.youtube.com/

https://en.wikipedia.org/

https://github.com/juanpabloprado/NavigationDrawerSample

http://www.appcoda.com/parse-server-migration/

https://parse.com

Tutorials:

https://www.youtube.com/user/slidenerd

http://www.101apps.co.za/

https://thenewboston.com/

http://www.androidhive.info/

Development Tools:

www.genymotion.com

http://developer.android.com/sdk/index.html

http://www.bluestacks.com/

Books:

1.Learn Android Studio: Build Android Apps Quickly and Effectively.

2.Java The Complete Reference

3.Head First Android Development