****

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

A Dissertation Report on

Pingr

Submitted by

Abhishek N 1MS15CS004

Abishek Padaki 1MS15CS005

Aravind P Anil 1MS15CS024

# *Bachelor of Engineering in Computer Science & Engineering*

Under the guidance of

S. Rajarajeswari

Assistant Professor

**RAMAIAH INSTITUTE OF TECHNOLOGY**

**(Autonomous Institute, Affiliated to VTU)**

**BANGALORE-560054**

2017-2018, [www.msrit.edu](http://www.msrit.edu),

****

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

A Dissertation Report on

Pingr

Submitted by

Abhishek N 1MS15CS004

Abishek Padaki 1MS15CS005

Aravind P Anil 1MS15CS024

# *Bachelor of Engineering in Computer Science & Engineering*

Under the guidance of

S. Rajarajeswari

Assistant Professor

**RAMAIAH INSTITUTE OF TECHNOLOGY**

**(Autonomous Institute, Affiliated to VTU)**

**BANGALORE-560054**

2017-2018, [www.msrit.edu](http://www.msrit.edu),

**BANGALORE-560054**

**Ramaiah Institute of Technology**

**(Autonomous Institute, Affiliated to VTU)**

**BANGALORE-560054**

# Department of Computer Science & Engineering

****

**CERTIFICATE**

This is to certify that the project work titled **Pingr is** a bonafide work carried out by **Abhishek N – 1MS15CS004, Abishek Padaki – 1MS15CS005 and Aravind P Anil – 1MS15CS024** in partial fulfillment for the course of Bachelor of Engineering in Computer Science and Engineering during the year 2018. The Project report has been approved as it satisfies the academic requirements with respect to the project work. To the best of our understanding the work submitted in this report has not been submitted anywhere.

### Signature of the Guide Signature of the HOD

S. Rajarajeswari Dr. Anita Kanavalli

**External Examiners**

Name of the Examiners: Signature

1. Srinidhi H

2. Hanumantha R

**DECLARATION**

I Student of sixth semester BE, Dept. of Computer Science and Engineering, Ramaiah Institute of Technology, Bangalore, hereby declare that the project entitled “**Pingr”,** thesis completed and written by me under the guidance of **S. Rajarajeswari,** Dept. of CSE, Bangalore.

Place: RIT, Mathikere, Bangalore

Date: 08-05-2018

1MS15CS004 – Abhishek Narayan

1MS15CS005 – Abishek Padaki

1MS15CS024 – Aravind P Anil

**ACKNOWLEDGEMENT**

First and foremost, my utmost gratitude to *S. Rajarajeswari* dept of CSE, MSRIT whose sincerity and encouragement we will never forget. She has been our inspiration as we overcame all the obstacles in the completion of this project work.

Dr. Anita Kanavalli, Head of the Department of Computer Science and Engineering, had kind concern and consideration regarding project work and we would like to thank him for continues support.

We would like to thank our beloved principal Dr. N.V.R Naidu for his support and encouragement.

This work would not have been possible without the guidance and help of several individuals who in one way or another contributed their valuable assistance in preparation and completion of this study.

We would like to express sincere thanks to all the teaching and non-teaching faculty of CSE Department and my dear friends who helped in all the ways while preparing the Report.

# Abstract

Students in the current timeline rely on digital media to study in colleges and schools. With easy access to internet on multiple devices, it only makes sense to make education better by using electronic media for the purpose of distribution of materials required to progress in classes. This comes with a problem where the distribution of media is fragmented across multiple platforms – Google groups, Messaging apps, Email, Cloud storages and so on. This is what we aim to solve with the project.

We aim to provide few features of email and WhatsApp under one app. The objective of this project is to make sure it is easy for students to combine various platforms and media into one section, thus reducing confusion and the need to have many forms of media for various college activities. The app is designed primarily to act as a hub for various notifications in your class. To make sure you do not miss out on anything important, the final product aims at organizing your study related electronic media in a single place so they’re easily accessible.

The target group of this project is clearly students of all ages. It is to make sure that the students are more easily informed about the important events and important days in an academic year. It includes important dates like dates of exams, public holidays, low attendances, assignment submission dates, new text books and their sources, warnings for low marks and so on. A secondary objective is to save the precious time of students. Time is a commodity that does not come by easy these days. Even more so for students who have to balance various aspects of their life in educational institutions. The service delivered by this app will save some time in looking for various materials over multiple platforms by optimizing them to appear under a single roof.

LIST OF FIGURES

|  |  |  |
| --- | --- | --- |
| NO. | FIGURE TITLE | PAGE NO. |
| 1 | SCHEDULE PLAN SPREADSHEET | 19 |
| 2 | GANTT CHART | 20 |
| 3 | SYSTEM ARCHITECTURE | 27 |
| 4 | GUI | 28 |
| 5 | CLASS DIAGRAM | 29 |
| 6 | SEQUENCE DIAGRAM | 30 |
| 7 | NOTIFICATION INTERCEPT | 33 |
| 8 | SERVER CLIENT MODULE | 34 |
| 9 | SIS MODULE | 35 |

LIST OF TABLES

|  |  |  |
| --- | --- | --- |
| NO | TABLE NAME | PAGE NUMBER |
| 1 | PROJECT DELIVERABLES | 12 |
| 2 | RISK MANAGEMENT | 21 |

#### Contents

***Declaration i***

***Acknowledgements ii***

***Abstract iii***

***List of Figures***

***List of Tables***

***Title Page No***

1 **INTRODUCTION 11**

* 1. General Introduction……………….
  2. Problem Statement…………..
  3. Objectives of the project……………
  4. Project deliverables……………
  5. Current Scope………………………
  6. Future Scope……………………….

1. **PROJECT ORGANIZATION 13** 
   1. Software Process Models
   2. Roles and Responsibilities
2. **LITERATURE SURVEY 16**

3.1….Introduction

3.2…Related Works with the citation of the References

3.3 Conclusion of Survey

1. **PROJECT MANAGEMENT PLAN 19**
   1. Schedule of the Project (Represent it using Gantt Chart)
   2. Risk Identification
2. **SOFTWARE REQUIREMENT SPECIFICATIONS 22**

5.1 Product Overview

5.2 External Interface Requirements

5.2.1 User Interfaces

5.2.2 Hardware Interfaces

5.2.3 Software Interfaces

5.2.4 Communication Interfaces

5.3 Functional Requirements

5.3.1 Functional Requirement 1.1

:

5.3.n Functional Requirement 1.n

1. **DESIGN 27**
   1. Introduction
   2. Architecture Design
   3. Graphical User Interface
   4. Class Diagram and Classes (represent Inheritance, Aggregation and Association)
   5. Sequence Diagram
   6. Data flow diagram
   7. Conclusion
2. **IMPLEMENTATION 31**
   1. Tools Introduction
   2. Technology Introduction
   3. Overall view of the project in terms of implementation
   4. Explanation of Algorithm and how it is been implemented
   5. Information about the implementation of Modules
   6. Conclusion
3. **TESTING 44**
   1. Introduction
   2. Testing Tools and Environment
   3. Test cases
4. **CONCLUSION & SCOPE FOR FUTURE WORK 45**
5. **REFERENCES 46**
6. **Appendix 47**

CHAPTER 1

INTRODUCTION

1.1 GENERAL INTRODUCTION: This app is a college notification app mainly aimed at providing timely notifications of important events such as examination fee submission dates, internals dates, upcoming public holidays, low attendance warning.

This is primarily for benefiting the students. It is to make sure that the students are more

easily informed about the important events and important days in an academic year.

1.2 PROBLEM STATEMENT: The current generation apps such as WhatsApp, Emails or Student information system app are the most widely used platforms that most colleges use for sharing information like internal examination timetable(dates), important announcements by class representative and teachers, class notes, or attendance and marks status etc. Which has a lot of drawbacks which includes wastage of time and lot more.

1.3 OBJECTIVES OF THE PROJECT: PINGR aims at reducing the drawbacks that exists in the above mentioned systems as much as possible by providing a much better interface to enable better student teacher communication. PINGR’S main purpose is to support time management amongst students as well as teachers.

1.4 PROJECT DERIVERABLES

* Notification enhancing application with dynamic application.
* Automatically converts notifications from other applications into a presentable form.
* Content based valuable information and data is delivered on time.
* It is administrator supported.
* Has integration with database for student information features.

|  |  |  |
| --- | --- | --- |
|  | EVENT | DEADLINE |
| 1. | Setting up Work Environment | March |
| 2. | Testing Notification phase | March |
| 3. | UI Designing | March |
| 4. | Firebase Integration | April |
| 5. | Web Portal development | April |
| 6. | Implement Portal to Android Environment | April |
| 7. | Web Scraping for SIS | April |
| 8. | Implementing QOL Features | May |
| 9. | Refining UI | May |
| 10. | Beta Testing | May |

Table 1

1.5 CURRENT SCOPE

The Application currently aims at providing a better interface for student teacher collaboration. The application aims at reducing the time and effort put in to send or receive valuable data. It doesn’t come as a surprise as WhatsApp and email is currently providing a similar background for data sharing, but our application tries to integrate data sharing features of the above mentioned apps. This application is a pure notification application mainly for the students and teachers.

1.6 FUTURE SCOPE

The outcome of this project may lead to a creation of a easier compatible system of data sharing and notification in a college environment. With access to important data and notifications with not much time involved will make it easier for students and teachers in achieving their desired goals. With the increase in chatting apps usage in the current generation, important data and notification may be missed or overlooked. That’s why PINGR aims at providing pure important data and relevant notifications that are related to college. It is an application purely made for college. Availability of student information and current semester progress stats becomes much easier through PINGR.

CHAPTER 2

PROJECT ORGANIZATION

2.1 Software Process Models

We use waterfall model in this project due to the nature of the project. The waterfall model is used because of two reasons – extensive documentation and scarce possibility of bugs. An iterative model is too time consuming for a project of this nature. Hence we introduce the waterfall model.

Everything is done in a sequential manner where the output of the previous step is required for the next step. Hence, we call it waterfall model.

However, in actuality, the steps overlap a lot. Design and implementation can be done parallel and integrated in a later phase. Waterfall models are rarely linear and in our case, even more so.

Abstract and Synopsis

SRS

System Design

Modules and Coding

Testing and Final Report

2.2 Roles and Responsibilities

Abstract, Synopsis and Introduction – Abhishek N

Literature Survey – Abishek Padaki

Requirement Specification Document – Aravind P Anil

System Design – Abhishek N

Project Management Plan (Gantt Chart) – Abishek Padaki

Risk Identification – Aravind P Anil

Sequence Diagram – Abhishek N

Use Case Diagram – Abhishek N

Data Flow Diagram – Abishek Padaki

Architecture Diagram – Abishek Padaki

Flowchart – Aravind P Anil

Graphical User Interface (Design Implementation) – Aravind P Anil

Class Diagram – Aravind P Anil

Admin to Client Notification System – Abishek Padaki

SIS Database and Implementation – Abhishek N

Autonotification App Creation and Integration - Aravind P Anil

Firebase Integration – Abishek Padaki

Firebase Database Structure Design – Abhishek N

Algorithmic Design – Aravind P Anil

Project Report Formatting – Abhishek N

Testing Implementation – Abishek Padaki

Implementation Tools Description – Abhishek N

Powerpoint Presentation – Aravind P Anil

Android Studio Gradle Modifications – Aravind P Anil

CHAPTER 3

LITERATURE SURVEY

3.1 INTRODUCTION

The various surveys mentioned below gave information about the currently existing system. The papers reveal how push notification services and cloud messaging are applicable in applications. The papers also show how an android based mobile framework can be used for student alert notification. The concept of Google Cloud Messaging, C2DM & Xtify is explained in brief with their applications, procedure and working. How to use GCM to give push notifications. The papers give information about MOODLE (course management system) that enhances learning.

3.2 RELATED WORKS WITH CITATION OF THE REFRENCES

1. *B.Dhivya, G.Lakshmiprabhu, P.Nivethitha and K.Kala* 2015 Cloud Messaging for Android is a Push Notification Service

Push messaging provides an important aspect of server to device communication.

This service provides a simple, lightweight mechanism that servers can use to tell mobile applications to contact the server directly, to fetch updated application or user data.

2) *Sagar Gore, Nitesh Sonawane, Sayali Pawar and Mrunal Nerkar 2017* Survey Paper on “An Android Based Mobile Framework for Student Alert Notification”

This paper deals with an android application made for academic avail of students, and teachers and staffs of the educational institution. Its features are- provide class and laboratory schedule, notice board, teachers update, notification for recently inclusive updates, CGPA (cumulative grade point average) calculation. Its goal is to provide avail in academically works by making communication more facile, provide simple and quick access to information.

*3) Jadhav Komal, Sayyad Sana, Shinde Swapnali, Bhaldar Jasmin and Prof. Mahind R. N.* 2016 Android based College Notification System

We have developing for staff add notes, experiments dates, submission and for students upload and download tutorials, experiments, question paper and notes in an easy and efficient manner. This project includes major activity entities such as: add notices, add staff record, and add student record which is handled by Admin. The Admin can add, delete, and update them simultaneously on the online system. These can then be viewed by the students.

4) *Saurabh Malgaonkar, Vivek Maurya, Mukul Kulkarni and Gurtej Singh Majithia 2014* Multipurpose Android based Mobile Notifier

This paper discusses about the various techniques and methods used for pushing or notification of messages using various Android connectivity methodologies on Android devices & applications. The concept of Google Cloud Messaging, C2DM & Xtify is explained in brief with their applications, procedure and working. The current application and usage of these technologies is also discussed with their future scope.

*5)Vineet Singh* 2015 Android Based Student Learning System

The proposed system has many learning tools, mainly the timely notification for different modules such as activities due dates, exam alert and discussion notification. Integrated with MOODLE (course management system) to enhance learning, it also has tutorial tool for students/tutors to reduce paper work and unnecessary workload on the tutors. This proposed system’s tools provide more flexibility, mobility and convenient than eLearning mode which rises routine attention to self-learning modules which might boost learner’s skills in learning.

*6) Rasha el Stohy, Nashaat el Khamesy and Haitham el Ghareeb 2016* A Proposed System for Push Messaging on Android

In this paper, we look at different push messaging alternatives available for Android. Push messaging provides an important aspect of server to device communication, and we specifically focus on the integration of cloud computing with mobile devices through the use of push-based technologies.

The paper presents a proposed notification system; ToParent, that actively provide alerts and messaging between parents and schools. Also, we investigate the performance of the two relevant push technologies for the Android devices, push notification service (PNS) and the short messaging service (SMS).

*7) Sourinda Chatterji, Srivatsa M S, Pareekshith K J, Amith Kashyap* 2014 A Mobile Application for Bus Notification System

The major talking point of this paper is the integration of VMD. This is a very clever implementation to cut down costs thus making the application more productive and cost effective. Knowledge of VMD will be helpful for Pingr. Also, the handling of multiple notifications is excellent as we require a similar implementation of the GSM for Pingr.

8) *Harshad Kale, Ganesh Rane, Sagar Shende, Swapnil Shinde* 2014 *Short* Message Service: Offline Notification System through SMS for Android Application

The project being undertaken here is an offline notification system that combines GCM and SMS – one that has not been seen before. The use of GCM to give push notifications is an excellent and clever addition. The structure is divided into three parts – One system receiving data, a third party service providing instructions and a service such as GCS or Apple Push Notification Service.

3.3 CONCLUSION OF SURVEY

The purpose of this was to identify the various fields in an android application. These papers show the different ways in which notifications can be sent and received. It shows how to avail in academically works by making communication more facile, provide simple and quick access to information. How integration of cloud computing with mobile devices through the use of push-based technologies is done is being identified. The main purpose of this survey was to identify different methods that can be undertaken to make Pingr a better application than the already existing applications in the current generation. We have tried to apply these concepts in our application. The Knowledge of VMD that is acquired from the papers will be helpful for Pingr. Also, the handling of multiple notifications is excellent as we require a similar implementation of the GSM for Pingr

CHAPTER 4

PROJECT MANAGEMENT PLAN

4.1 SCHEDULE OF THE PROJECT

The project plan has been done and followed as per the given requirements and the

deadline given to us by the institution. The commencement of the project began this semester

and has almost ended prior to our expected due date.

SCHEDULE PLAN SPREADSHEET:

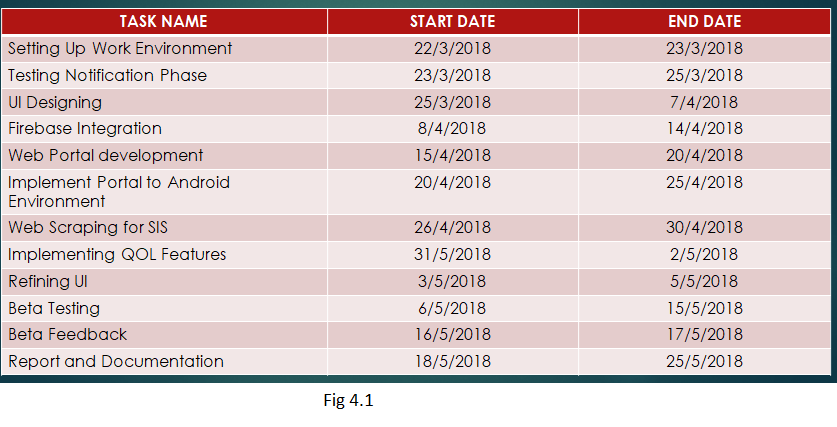


Figure 1

GANTT CHART FOR PROJECT PLANNING:

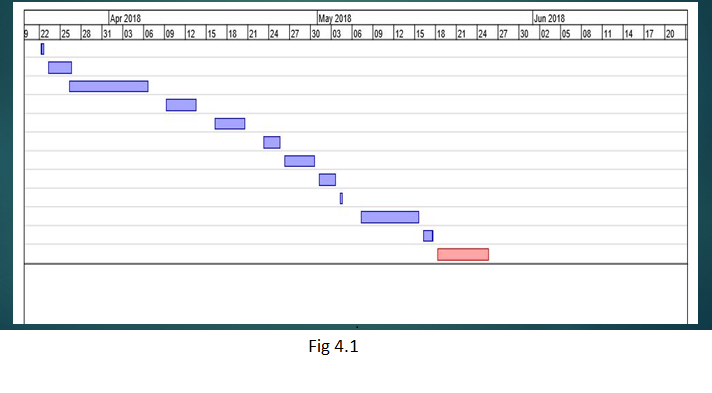


Figure 2

4.2 RISK IDENTIFICATION

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| RISK | DESCRIPTION | PROBABILITY | SEVERITY | MITIGATION |
| Scheduling shortcomings | Not able to meet deadline for various stages | Medium | High | Create accurate schedule charts. Follow multiple release cycles and agile practice |
| Requirement shortcomings | Client not happy with the final product and it is short of listed requirements | Low | Medium | Verify the SRS with client. Create two SRS and compare for client and project head. Follow prototyping and staged releases |
| Requirements Inflation | New requirements arise as the project moves forward thus causing danger in scheduling and effort estimation | Low | Low | Follow a standard model for effort estimation. Create scheduling and modify according to generated effort. Include overhead time for requirements inflation and keep aside buffer time for unforeseen circumstances |
| Lack of Resources | As the project is taken on, realization that initial resource assumption in system architecture was wrong and necessary resources cannot be obtained on short notice | Low | High | Detailed System architecture documents verified by multiple personnel. Assumption of resource scarcity and having backup for pooling resources from multiple sources |
| Design Compromises | Rushing design to start implementation causing half-baked design translating to poor framework for implementation | Low | Medium | Cooperation from design team and implementation team. Proper scheduling to ensure enough time is assigned for design and coding |
| Personnel Risks | Day to day activities of team interfering with project schedule | High | Low | Including personnel risk timing overhead in scheduling. Creating planning document according to personal calendar of teams involved |

Table 2

Chapter 5

Software Requirement Specification

1. Product Overview

This document contains a general description of what the specific requirements of the application being developed needs. It is written from a developer’s point of view so that the end users can clearly understand what to expect from this project. It gives a viewpoint of the project that shows the specific technicalities and features needed by the end users and these very needs that must be implemented by us, the developers.

At the core of the application, the main features include a software that runs in the android environment and depends on firebase integration for server side notifications. A web portal is also included for easy management of files and notifications to avoid clutter.

1. Hardware Requirements
   1. Android Ecosystem

The application will only run on android environment as it is built using android studio. It is being targeted for a minimum SDK of API Level 19. The minimum version of android is 4.4.4 codenamed Android KitKat.

* 1. Android RAM Requirements

The application is fairly lightweight and will run on any phone released after the year 2010. Most android phones have a minimum RAM of 1 GB. This app being very light on load will run on any phone with more than 512 MB of RAM.

* 1. Android Internal Storage Requirements

The application will take a maximum of 25 MB of internal storage space. Most phones come with a minimum of 8 GB of internal storage. Hence, the application can run on any modern phone with ease.

* 1. Web Portal Requirements

The web portal must run on a Web Browser. The portal web app is optimized for Google Chrome. Hence, a system – android, IOS or Computer than can run google chrome is required. Google Chrome is not the exclusive system. Safari, Mozilla can also be used. However, it is recommended to use Google Chrome.

* 1. Web Portal Ram Requirements

The Web portal being optimized for google chrome, will take up a fair amount of RAM. Google Chrome places a heavy load on RAM with the usage going up to 4 GB for 8 tabs. Our requirement is only one tab. Hence, a minimum of 2 GB RAM and a recommended specification of 4 GB RAM is required.

1. Software Requirements
   1. Android Ecosystem

The application requires android ecosystem. It will not run on apple as it has been coded using Java and XML and not Swift. The app will not be ported to the apple ecosystem or the windows ecosystem. It is exclusive to android.

* 1. Firebase

The project relies on firebase integration for software. The Server-client notification system is built solely on firebase which combines with the web portal to provide a seamless experience for clients and end users alike.

1. Functional Requirements

This section highlights the various features and functions the application provides and the modules or functionalities the end users can put to use with this project

* 1. TITLE – Download Android Application

DESC – The user must be able to download the given application from either the google play store, a github release page, a relevant open source coding base such as androidcodebase, or another trusted store release such as aptoide, bitbucket, f-droid etc.

* 1. TITLE – Updates to The Application

DESC – The user must be notified of any updates to the application as soon as possible. The update can be via a staged release to different parts of the world or a synchronized OTA at the same time to everyone.

* 1. TITLE – Student Registration

DESC – The users must be able to register to a particular class. This is done by generating a unique code that must be provided at both server and client side.

* 1. TITLE – Student Login

DESC – After registration, the student must be given the ability to login automatically each time the app is opened. The login information must be stored either encrypted or as plain text on a centralized server or the disk storage.

* 1. TITLE – Notification Log

DESC – The user must immediately view the notification log in an intuitive UI that shows all the important notifications that he has received from the server corresponding to his class. The notifications are prioritized and sorted by date.

* 1. TITLE – Notification Categories

DESC – The notifications being received are of multiple types which are described here

* + 1. TITLE – Dates for External Examinations

DESC – Dates of the final examinations taking place at the end of the semester are shown 10 days prior as well as 3 hours prior to the examination start time.

* + 1. TITLE – Dates for Internal Examinations

DESC – Dates of the internal examinations taking place 3 times during the course of the semester are notified to the students.

* + 1. TITLE – Dates for lab internals

DESC – Lab Internals follow a different schedule to the regular internals as they vary from one class to another. The appropriate date for lab internals is also notified at least a day prior to the beginning of the examination.

* + 1. TITLE – Dates for Quizzes and Multiple Choice Questionnaires

DESC – The 20 internal marks for most subjects are allotted via quizzes. The quizzes are scheduled at times that depends on the availability of the teachers and students. This is notified accordingly.

* + 1. TITLE – Dates for Submission of Projects for the internal component

DESC – The 20 marks for internal scheme may also be a report on a case study, a small activity to be performed in the lab or other projects of the small nature. These are given a deadline by the teacher who assigns the project. These deadlines are duly noted.

* + 1. TITLE – Dates for pre-determined holidays

DESC – Dates of holidays that fall in the category of government holidays, religious holidays, holidays given prior to exams are pre-determined. Their dates can be assigned at the beginning of the year. Hence, these holidays are notified.

* + 1. TITLE –Dynamic Holidays

DESC – Holidays that arise during the course of the year due to unforeseen circumstances such as strikes, bandhs, civil unrest and so on. These dates cannot be notified early because they cannot be predicted. They are notified immediately after the college announces them.

* + 1. TITLE – Text Books

DESC – New text book in pdf or epub format are notified with a link that provides a download location for these text books. Clicking on the notification will download the media and place it in /Downloads folder.

* + 1. TITLE – Hand Written Notes

DESC – Either converted pdfs or hand written notes in jpg or png format is notified preferably before the internals to aid the students. Clicking on the notification ensures a download of the given media into the Downloads folder/

* + 1. TITLE – Code Snippets

DESC – Either entire programs or algorithms or just snippets of code are given to the student. They will preferably be put in a version control system such as git or mercurial. If that is not possible, they are put in a file with the right extension such as c, py etc. and zipped. The last resort is a pdf containing the code if neither of the options are possible.

* + 1. TITLE – Parts of Text books chosen to represent the syllabus accurately

DESC – Sometimes, trimmed pdfs of very large text books are provided that represent only what is required according to prescribed syllabus. These are also notified with a click on the notification resulting in the download of a file.

* + 1. TITLE – Other forms of media such as mp4 or mp3 files

DESC – Video links for self-study or studying beyond the given scope are also linked. Download links will be provided irrespective of the size of the file.

* + 1. TITLE – Reminders for attendance warnings

DESC – Students are notified of the warnings to attendance when their attendance falls below 85%.

* + 1. TITLE – Reminders for internal marks

DESC – Students need a minimum of 20 marks out of 50 to gain eligibility to write their semester end exams. Hence students are given warnings after each internal examination if they do not meet this criterion.

1. Performance Requirements
   1. TITLE – Network Requirements

DESC – Since the app deals with internet to collect information for notifications, there must be no latency delay with the app while it connects to the internet.

* 1. TITLE – Interface of the Notifications

DESC – Notifications must be unambiguous, clear and concise. They are an integral part of the app and hence, an improvement in the notifications means a general improvement to the app itself.

* 1. TITLE – Notification channels by the android framework

DESC – Notifications that come to the user handled by android must be intuitive. The title and description must immediately give the user an idea of what the notification means to convey without having to open the app.

* 1. TITLE – Response Time

DESC – When the app is opened and a dynamic object is clicked, the application must respond within 3 seconds 80% of the time.

* 1. TITLE – Internet Delay

DESC – Given the app has a stable internet connection of at least 1 Megabit/second, the application must respond within 10 seconds while scraping information 75% of the time.

1. Design Constraints

Since this is an android application, the only design constraint that is applicable here is limitations within android itself. The app may vary between different sizes and different resolutions of different devices. Hence, making the application compatible with android screens of different sizes is a difficult constraint to overcome.

Another constraint is the availability of a steady internet connection. A connection of at least 1 Mbps is needed to ensure to proper working of the application.

CHAPTER 6

DESIGN

6.1 INTRODUCTION

The design process begins when the requirements documents for the software to be developed is available and the architecture has been developed. The design process involves refining of the architecture. The design of a system is a blueprint or a plan for a solution for the system.

6.2 SYSTEM ARCHITECTURE

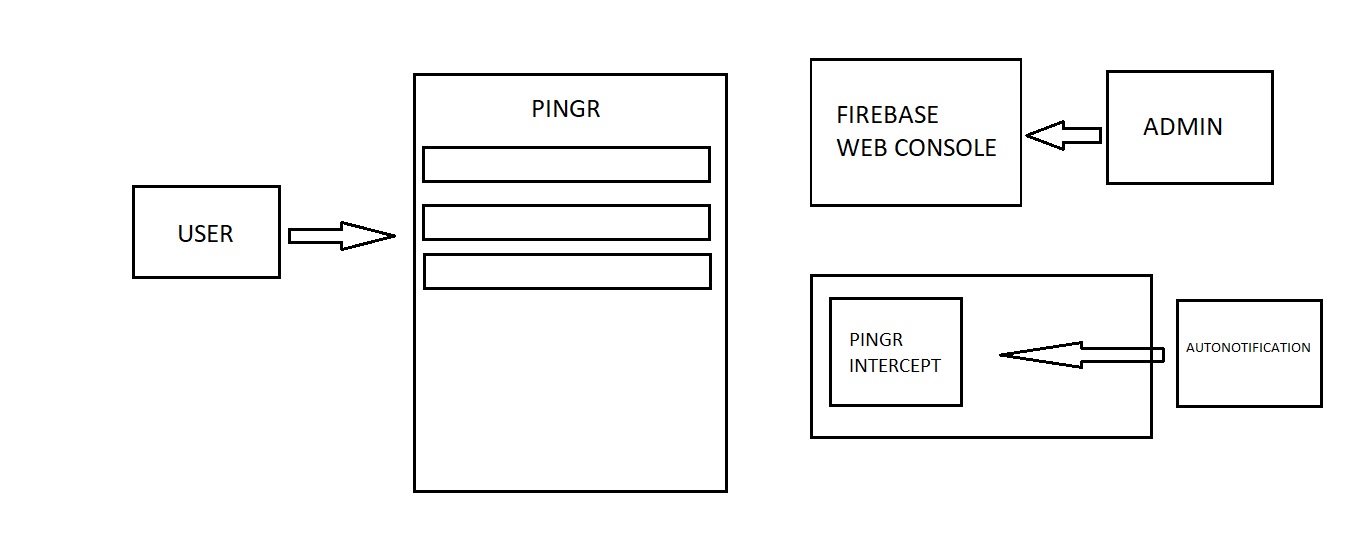


Figure 3

6.3 GRAPHICAL USER INTERFACE

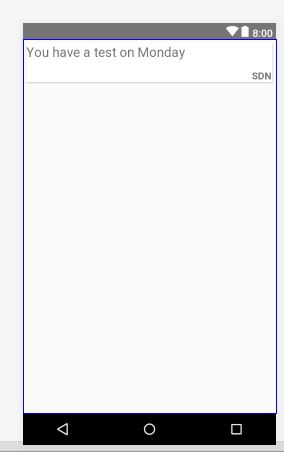


Figure 4

6.4 CLASS DIAGRAMS AND CLASSES

firebase

gradle

Card Item Class

Main Activity

Inflates

Card Adapter

Category calculate

String regex

int [][] attendance

SIS Build

int [][][] marks

int [][] total

Figure 5

6.5 SEQUENCE DIAGRAM

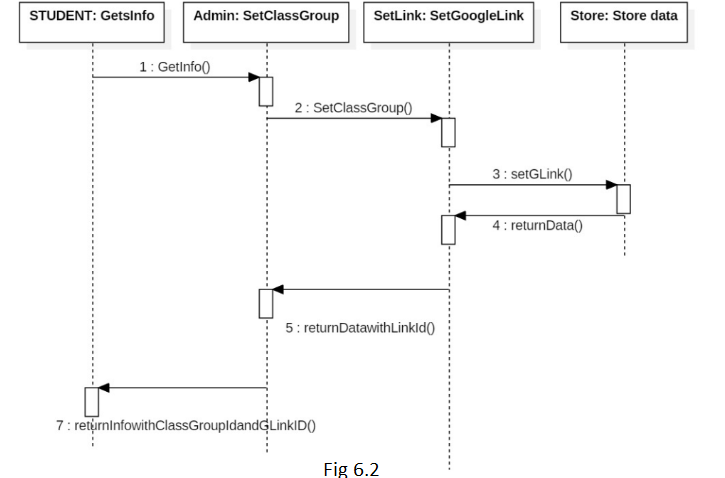


Figure 6

CHAPTER 7

Implementation

7.1 Tools Introduction

Android Studio – A fully featured IDE for developing apps. Made by jetbrains, this is the go to IDE for most developers to make android apps. Our choice was Android Studio because of the extensive support and documentation that is provided in the software. The studio was the most used tool for coding the algorithm as well as the GUI on XML.

Firebase – A free cloud service by google, Firebase started as an improvised version of GCM. However, it has come leaps and bounds now. Firebase provides authentication, Cloud messaging, storage, Database solutions and much more. We only use the Cloud messaging from console to a single instance or broadcast notification in our project.

Tasker – Tasker is a relatively unknown resource for many. It is an automation service provided on android platform. It has documentation for most of its features. It is very powerful if used correctly and combined with root permissions on android. We use tasker to provide support for its plugin – Autonotification. We also provide push notification service and toast features using tasker.

Autonotification – Used as a plugin for tasker, it enables users to intercept notifications and make them richer and more feature filled. Autonotification is used in our project for intercepting notifications from other apps based on regular expressions.

7.2 Overall View

The implementation can be divided into three categories.

The first is the automatic notification intercept. Using tasker and autonotification, we have designed an app called Pingtercept that will intercept notifications on your device with a custom trigger. The custom trigger corresponds to relevance of notifications and the intercepted notifications are sent to Pingr. Once on Pingr, it will be displayed in a card view with automatic refresh and categorized.

The second implementation is Server Client Push Notification Service. The server can access a firebase web portal. Through this portal, he/she can send push notifications as a broadcast message to all instances of Pingr. This makes it easy for users to receive notifications that can’t be displayed on their phone through normal circumstances.

Finally, the third implementation is a SIS portal. We have used a dynamically generated database inside Android. With the USN, we pull the data and verify it against some integrity constraints. Once the constraints are done, any discrepancies are notified.

7.3.1 Notification Intercept

The intercept works by using autonotification. Autonotification is a dependency for Pingtercept. Pingtercept intercepts all notifications from whatsapp and gmail. It also intercepts notifications of Pingr itself. These are notifications from the admin.

Once pingtercept gets the notifications, it matches them to a regex that verifies the relevance. If found, a named group in the regex returns category of the notification and writes to a file called “Pingr.txt”. The Pingr app reads from this file and displays notification on the Card Item created along with the category.

Pinger Intercept App

Performs push notification and intercept functions.

Also Provides toast

Autonotification Plugin

Write to Pingr.txt

Text file on sdcard

Pingr app categorizes depending on regex named groups

Displays on cardview

Read

Figure 7

7.3.2 Client Server Notification

This feature is a simple module that enables push notifications. By enabling integration with firebase into Pingr, we make a firebase project. The firebase project has a console by which we can send Pingr notification messages. Like explained before, the Pingr notifications are intercepted by Pingtercept and sent back to Pingr thus showing on the display.

Admin Firebase Console

Display notification using android notification framework

Also displays on Pingr cardview

Admin User

Receive token

Ask for token

Notification

Figure 8

7.3.3 SIS Module

The SIS module is a dynamically generated java database within Pingr. Each time the activity is refreshed, the module assigns random values. With the USN provided in “instructions.txt”, we pull the relevant data and perform checks. If any discrepancy is found, we enable it in the GUI using carditem.

Figure 9

Dynamic Database

In Java

Random Number generator

populates

Pingr Pulls Data from database

Multiple checks for marks/attendance

Flags

GUI

Display in cardview

7.4 Code

7.4.1 MainAcitivity

**package** com.pingr.pingrteam.pingr;  
  
**import** android.content.BroadcastReceiver;  
**import** android.content.Context;  
**import** android.content.Intent;  
**import** android.content.IntentFilter;  
**import** android.graphics.Color;  
**import** android.os.Build;  
**import** android.os.Environment;  
**import** android.support.annotation.RequiresApi;  
**import** android.support.v4.content.LocalBroadcastManager;  
**import** android.support.v4.widget.DrawerLayout;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.support.v7.widget.LinearLayoutManager;  
**import** android.support.v7.widget.RecyclerView;  
**import** android.text.Html;  
**import** android.util.Log;  
**import** android.view.Menu;  
**import** android.view.MenuInflater;  
**import** android.view.MenuItem;  
**import** android.widget.TableRow;  
**import** android.widget.TextView;  
**import** android.widget.Toast;  
**import** android.widget.Toolbar;  
  
**import** java.io.BufferedReader;  
**import** java.io.File;  
**import** java.io.FileInputStream;  
**import** java.io.FileNotFoundException;  
**import** java.io.FileReader;  
**import** java.io.IOException;  
**import** java.io.InputStreamReader;  
**import** java.util.ArrayList;  
**import** java.util.List;  
**import** java.util.Random;  
**import** java.util.regex.Matcher;  
**import** java.util.regex.Pattern;  
  
**public class** MainActivity **extends** AppCompatActivity {  
  
 RecyclerView **recyclerView**;  
 List<CardItem> **productList**;  
 String **title**, **category**;  
  
  
 @RequiresApi(api = Build.VERSION\_CODES.***O***)  
 @Override  
  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
 **recyclerView** = (RecyclerView) findViewById(R.id.***recycler\_view***);  
 **recyclerView**.setLayoutManager(**new** LinearLayoutManager(**this**));  
 **productList** = **new** ArrayList<>();  
 CardAdapter adapter = **new** CardAdapter(**this**, **productList**);  
 **recyclerView**.setAdapter(adapter);  
 File sdCardDir = Environment.*getExternalStorageDirectory*();  
 File txtFile = **new** File(sdCardDir, **"/pingr.txt"**);  
 File instr = **new** File(sdCardDir, **"/instructions.txt"**);  
 android.support.v7.widget.Toolbar tb1;  
 tb1 = (android.support.v7.widget.Toolbar) findViewById(R.id.***tb1***);  
 setSupportActionBar(tb1);  
 **try** {  
  
 BufferedReader reader = **new** BufferedReader(**new** FileReader(txtFile));  
  
 String line;  
  
 **while** ((line = reader.readLine()) != **null**) {  
 productListPopulate(line, categoryCalculate(line));  
 }  
 reader.close();  
 } **catch** (IOException e) {  
 Log.*e*(**"C2c"**, **"Error occured while reading text file!!"**);  
  
 }  
  
  
 **try** {  
  
 BufferedReader reader = **new** BufferedReader(**new** FileReader(instr));  
  
 String line;  
 line = reader.readLine();  
 sisBuild(line);  
  
 reader.close();  
 } **catch** (IOException e) {  
 Log.*e*(**"C2c"**, **"Error occured while reading text file!!"**);  
  
 }  
  
  
 }  
  
  
 **private void** sisBuild(String l) {  
 Random rand = **new** Random();  
 **int** i, j, k, pos = 0, sum = 0;  
 String[] sub = {**"USP"**, **"CD"**, **"S.E"**, **"WEB"**};  
 String t = **"1MS15CS0"**;  
 String[] usn = **new** String[50];  
 **int**[][][] marks = **new int**[50][4][3];  
 **int**[][] attendance = **new int**[50][4];  
 **int**[][] total = **new int**[50][4];  
 **for** (i = 0; i < 50; i++) {  
 usn[i] = t + Integer.*toString*(i);  
 **if** (l == usn[i]) {  
 pos = i;  
 Toast.*makeText*(getApplicationContext(), usn[i], Toast.***LENGTH\_SHORT***).show();  
 }  
 }  
  
 **for** (i = 0; i < 50; i++) {  
 **for** (j = 0; j < 4; j++) {  
 **for** (k = 0; k < 3; k++) {  
 marks[i][j][k] = rand.nextInt(31);  
 sum = sum + marks[i][j][k];  
 }  
 total[i][j] = sum;  
 sum = 0;  
 }  
 }  
  
 **for** (i = 0; i < 50; i++)  
 **for** (j = 0; j < 4; j++)  
 attendance[i][j] = (rand.nextInt(51) \* 2);  
  
 **for** (i = 0; i < 4; i++) {  
 **if** (attendance[pos][i] <= 60)  
 productListPopulate(**"Your attendance is "** + attendance[pos][i] + **"% in "** + sub[i], **"SIS"**);  
 }  
  
 **for** (i = 0; i < 4; i++) {  
 **if** (total[pos][i] <= 36)  
 productListPopulate(**"Your total for "** + sub[i] + **" is "** + total[pos][i] + **".You may not get eligibility."**, **"SIS"**);  
 }  
  
  
 }  
  
 @RequiresApi(api = Build.VERSION\_CODES.***O***)  
 **private** String categoryCalculate(String line) {  
 String regex = **"(?i).\*(?<category>sdn|usp|web|iot|cd|compiler|s\\.e|exam|time table)[^?]\*$"**;  
 Pattern p = Pattern.*compile*(regex);  
 Matcher m = p.matcher(line);  
 **if** (m.find())  
 **return** (m.group(**"category"**));  
 **else return "misc"**;  
 }  
  
 **void** productListPopulate(String text, String category) {  
 **productList**.add(**new** CardItem(text, category));  
 }  
  
 @Override  
 **public boolean** onCreateOptionsMenu(Menu menu) {  
 MenuInflater mi = getMenuInflater();  
 mi.inflate(R.menu.***actionbar1***, menu);  
 **return super**.onCreateOptionsMenu(menu);  
 }  
  
 @Override  
 **public boolean** onOptionsItemSelected(MenuItem item) {  
 **int** i = item.getItemId();  
 **if** (i == R.id.***tlbPlus***) {  
 Intent in = getIntent();  
 finish();  
 startActivity(in);  
 }  
 **return super**.onOptionsItemSelected(item);  
 }  
  
  
}

7.4.2 CardItem

**package** com.pingr.pingrteam.pingr;  
  
**public class** CardItem {  
  
 String **title**,**subject**;  
  
 **public** CardItem(String title, String subject) {  
 **this**.**title** = title;  
 **this**.**subject** = subject;  
 }  
  
 **public** String getTitle() {  
 **return title**;  
 }  
  
 **public** String getSubject() {  
 **return subject**;  
 }  
}

7.4.3 CardAdapter

**package** com.pingr.pingrteam.pingr;  
  
**import** android.content.Context;  
**import** android.support.v7.widget.RecyclerView;  
**import** android.view.LayoutInflater;  
**import** android.view.View;  
**import** android.view.ViewGroup;  
**import** android.widget.ImageView;  
**import** android.widget.TextView;  
  
  
**import** com.pingr.pingrteam.pingr.CardItem;  
**import** com.pingr.pingrteam.pingr.R;  
  
**import** java.util.List;  
  
*/\*\*  
 \* Created by Belal on 10/18/2017.  
 \*/***public class** CardAdapter **extends** RecyclerView.Adapter<CardAdapter.ProductViewHolder> {  
  
  
 *//this context we will use to inflate the layout* **private** Context **mCtx**;  
  
 *//we are storing all the products in a list* **private** List<CardItem> **productList**;  
  
 *//getting the context and product list with constructor* **public** CardAdapter(Context mCtx, List<CardItem> productList) {  
 **this**.**mCtx** = mCtx;  
 **this**.**productList** = productList;  
 }  
  
 @Override  
 **public** ProductViewHolder onCreateViewHolder(ViewGroup parent, **int** viewType) {  
 *//inflating and returning our view holder* LayoutInflater inflater = LayoutInflater.*from*(**mCtx**);  
 View view = inflater.inflate(R.layout.***card\_item***, **null**);  
 **return new** ProductViewHolder(view);  
 }  
  
 @Override  
 **public void** onBindViewHolder(ProductViewHolder holder, **int** position) {  
 *//getting the product of the specified position* CardItem product = **productList**.get(position);  
 *//binding the data with the viewholder views* holder.**textViewTitle**.setText(product.getTitle());  
 holder.**textViewSubject**.setText(product.getSubject());  
 }  
  
  
 @Override  
 **public int** getItemCount() {  
 **return productList**.size();  
 }  
  
  
 **class** ProductViewHolder **extends** RecyclerView.ViewHolder {  
  
 TextView **textViewTitle**, **textViewSubject**;  
  
 **public** ProductViewHolder(View itemView) {  
 **super**(itemView);  
  
 **textViewTitle** = itemView.findViewById(R.id.***title***);  
 **textViewSubject** = itemView.findViewById(R.id.***subject***);  
 }  
 }  
}

7.4.4 Activity\_main.xml

*<?***xml version="1.0" encoding="utf-8"***?>*<**RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:id="@+id/relative\_layout"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"**>  
  
 <**android.support.v7.widget.Toolbar  
 android:id="@+id/tb1"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_alignParentStart="true"  
 android:layout\_alignParentTop="true"  
 android:background="?attr/actionModeSplitBackground"  
 android:minHeight="?attr/actionBarSize"  
 android:theme="?attr/actionBarTheme"** />  
  
 <**android.support.v7.widget.RecyclerView  
 android:id="@+id/recycler\_view"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_alignParentStart="true"  
 android:layout\_alignParentTop="true"  
 android:layout\_marginTop="54dp"**>  
  
 </**android.support.v7.widget.RecyclerView**>  
  
  
</**RelativeLayout**>

7.4.5 Carditem.xml

*<?***xml version="1.0" encoding="utf-8"***?>*<**RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"**>  
  
 <**android.support.v7.widget.CardView  
 android:layout\_width="match\_parent"  
 android:layout\_height="60dp"  
 android:layout\_margin="5dp"**>  
  
  
 <**RelativeLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"**>  
  
 <**TextView  
 android:id="@+id/title"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="You have a test on Monday"  
 android:textSize="20dp"** />  
  
 <**TextView  
 android:id="@+id/subject"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="SDN"  
 android:textSize="15dp"  
 android:textStyle="bold"  
 android:layout\_alignParentBottom="true"  
 android:layout\_alignParentRight="true"  
 android:layout\_marginRight="2dp"**/>  
  
 </**RelativeLayout**>  
  
  
 </**android.support.v7.widget.CardView**>  
</**RelativeLayout**>

7.4.6 MenuItem.xml

*<?***xml version="1.0" encoding="utf-8"***?>*<**menu xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:android="http://schemas.android.com/apk/res/android"**>  
  
 <**item  
 android:id="@+id/tlbPlus"  
 android:icon="@drawable/ic\_refresh\_black\_24dp"  
 android:title="Plus"  
 app:showAsAction="always"** />  
</**menu**>

7.4.7 Pingr Gradle

*// Top-level build file where you can add configuration options common to all sub-projects/modules.*buildscript {  
   
 repositories {  
 google()  
 jcenter()  
 }  
 dependencies {  
 classpath **'com.android.tools.build:gradle:3.1.2'** classpath **'com.google.gms:google-services:3.2.0'** *// NOTE: Do not place your application dependencies here; they belong  
 // in the individual module build.gradle files* }  
}  
  
allprojects {  
 repositories {  
 google()  
 jcenter()  
 }  
}  
  
task clean(**type**: Delete) {  
 delete rootProject.buildDir  
}

7.4.8 App Gradle

*// Top-level build file where you can add configuration options common to all sub-projects/modules.*buildscript {  
   
 repositories {  
 google()  
 jcenter()  
 }  
 dependencies {  
 classpath **'com.android.tools.build:gradle:3.1.2'** classpath **'com.google.gms:google-services:3.2.0'** *// NOTE: Do not place your application dependencies here; they belong  
 // in the individual module build.gradle files* }  
}  
  
allprojects {  
 repositories {  
 google()  
 jcenter()  
 }  
}  
  
task clean(**type**: Delete) {  
 delete rootProject.buildDir  
}

7.4.9 AndroidManifest.xml

*<?***xml version="1.0" encoding="utf-8"***?>*<**manifest xmlns:android="http://schemas.android.com/apk/res/android"  
 package="com.pingr.pingrteam.pingr"**>  
  
 <**application  
 android:allowBackup="true"  
 android:icon="@mipmap/ic\_launcher"  
 android:label="@string/app\_name"  
 android:roundIcon="@mipmap/ic\_launcher\_round"  
 android:supportsRtl="true"  
 android:theme="@style/Theme.AppCompat.Light.NoActionBar"**>  
 <**activity android:name=".MainActivity"**>  
 <**intent-filter**>  
 <**action android:name="android.intent.action.MAIN"** />  
  
 <**category android:name="android.intent.category.LAUNCHER"** />  
 </**intent-filter**>  
 </**activity**>  
  
 </**application**>  
 <**uses-permission android:name="android.permission.RECORD\_AUDIO"** />  
 <**uses-permission android:name="android.permission.READ\_EXTERNAL\_STORAGE"**/>  
 <**uses-permission android:name="android.permission.INTERNET"**/>  
 <**uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE"**/>  
  
</**manifest**>

CHAPTER 8

TESTING

8.1 Testing Introduction

The testing for this project was minimal as there are not many chances for bugs to arise. Testing was mostly done in sandboxed as well as real time scenarios extensively to weed out any bugs or complications that might arise. Testing was done for

* Android app GUI such as overflow from cards etc.
* Tasker for any wrong cases caught by the regular expression
* Performance testing for tasker autonotification intercept
* Performance testing for firebase
* Latency testing
* File read testing for file not found cases

8.2 Testing Environment and Test Cases

1. Initial testing was done within android studio. Bug reports, Adb dumps, logcats were analyzed thoroughly for any mistakes or things out of place.
2. Preliminary testing also included GUI tests before inflating. This meant space reductions, complying with material design guidelines at each step and adding the right amount of backdrop and depth for each element in the UI
3. Secondary testing was done for functions written and any recursive calls or infinite loops. Such cases are strictly to be avoided as they are unethical coding practices.
4. Secondary testing was thoroughly done with timed tests to check for responses. If the app takes too many resources, it can crash Android system itself and not just Pingr.
5. Third phase of testing included individual testing for modules. Each module was sandboxed to work independently of one another.
6. The SIS module was tested for performance and latency. Since the database is dynamically generated, there is a slight delay on the OnCreate() of the main activity.
7. The Admin module was tested for firebase latency and response time. The test responses were immediate 94 out of 100 times.
8. The automatic intercept went through some very intensive testing. First phase was testing Pingr Intercept on tasker. Tests were scheduled to calculate various possible outcomes and possibilities of bugs arising.

CHAPTER 9

CONCLUSION

The result of this project is an amazing product that can be used by people in the field of education alike. Very often we find ourselves missing out on important notifications because we didn’t notice it in the huge mess of social media. This app rectifies it to a certain extent with very little human interaction.

The three modules come together and tie in very nicely. The SIS module is integrated seamlessly yet shown as an important factor by intelligent use of categories. Once you see the notifications, you will never miss another one again.

A design principle of this app was to keep it as simple as possible. Hence, there is just one activity to base this application off on. The main aim was to simplify the influx of electronic media with relevance to education. It goes against principle to complicate it further by having multiple activities for them to browse through.

In the end, this application has a myriad of uses in the industry and can be used by many.

CHAPTER 10

REFERENCES

[1] Cloud Messaging for Android Is a Push Notification Service -B. Dhivya, G. Lakshmiprabha, P. Nivethitha and K. Kala

[2] Survey Paper on “An Android Based Mobile Framework for Student Alert Notification”

-Sagar Gore, Nitesh Sonawane, Sayali Pawar and Mrunal Nerkar

[3] Android Based College Notification System -Jadhav Komal, Sayyad Sana, Shinde Swapnali, Bhaldar Jasmin and Prof. Mahind R. N.

[4] Multipurpose Android Based Mobile Notifier -Saurabh Malgaonkar, Vivek Maurya, Mukul Kulkarni and Gurtej Singh Majithia

[5] Android Based Student Learning System -Vineet Singh

[6] A Proposed System for Push Messaging on Android -Rasha el Stohy, Nashaat el Khamesy and Haitham el Ghareeb

[7] A Mobile Application for Bus Notification System -Sourinda Chatterji, Srivatsa M S, Pareekshith K J, Amith Kashyap

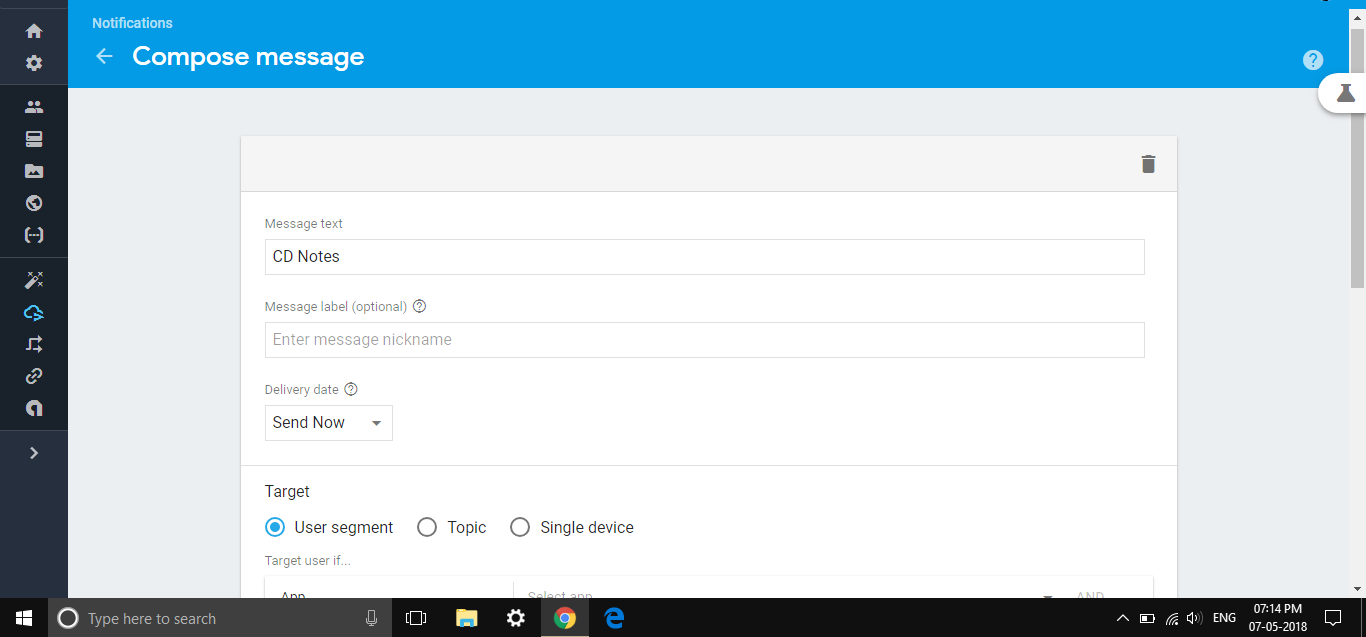
[8] Short Message Service: Offline Notification System through SMS for Android Application -Harshad Kale, Ganesh Rane, Sagar Shende, Swapnil Shinde

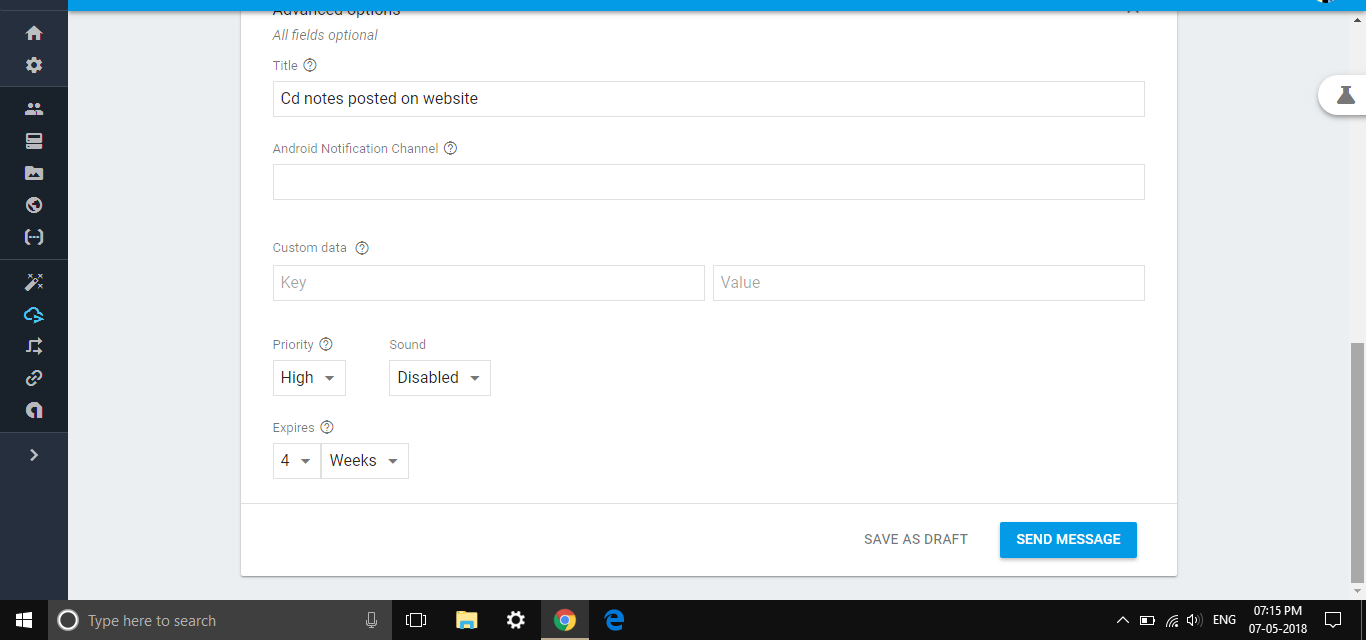
[9] Personal Emergency Notification Application for Mobile Devices -Akansha Raj, Asmita Pawar, Ganesh M. Gaikwad

CHAPTER 11

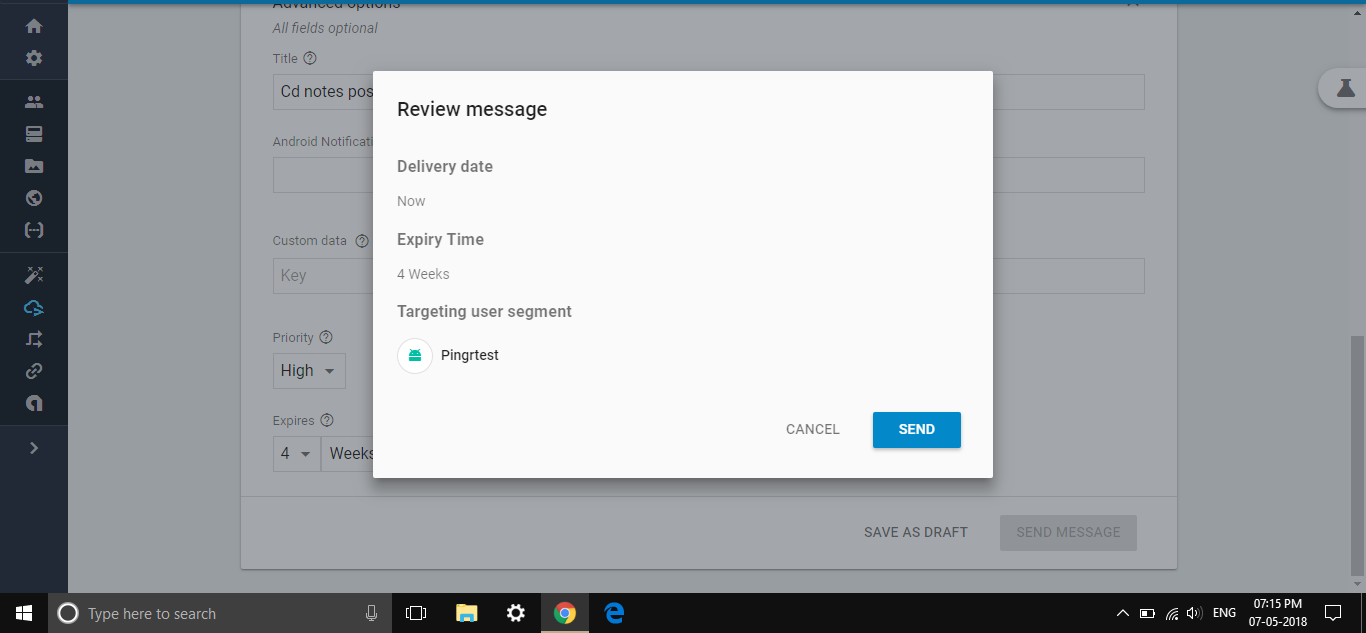
APPENDIX

**Screen Snapshots (Results)**

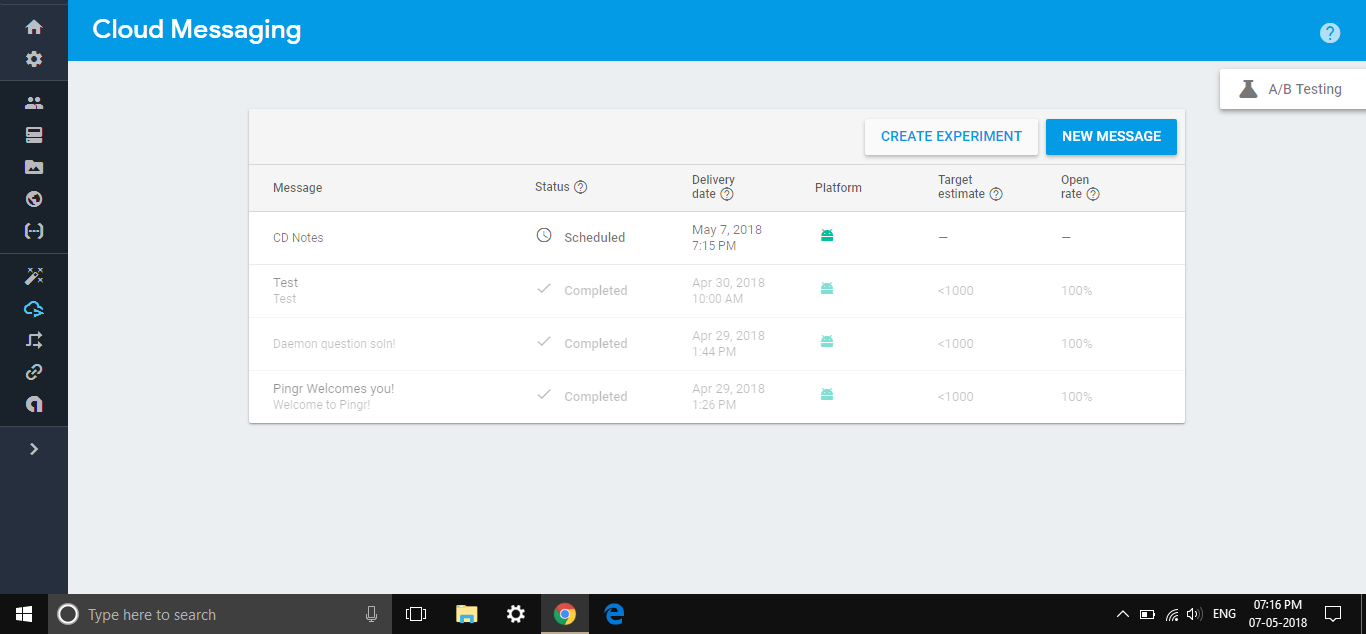




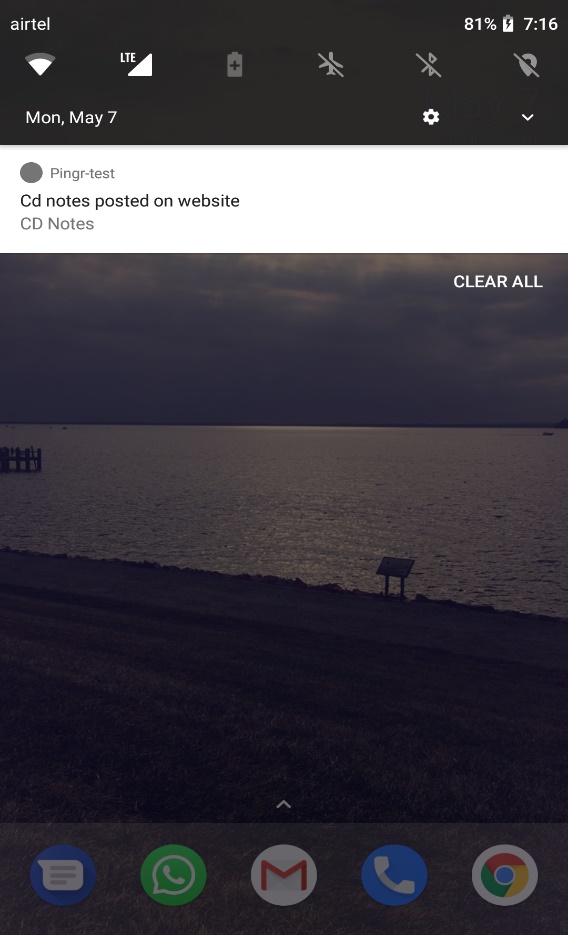
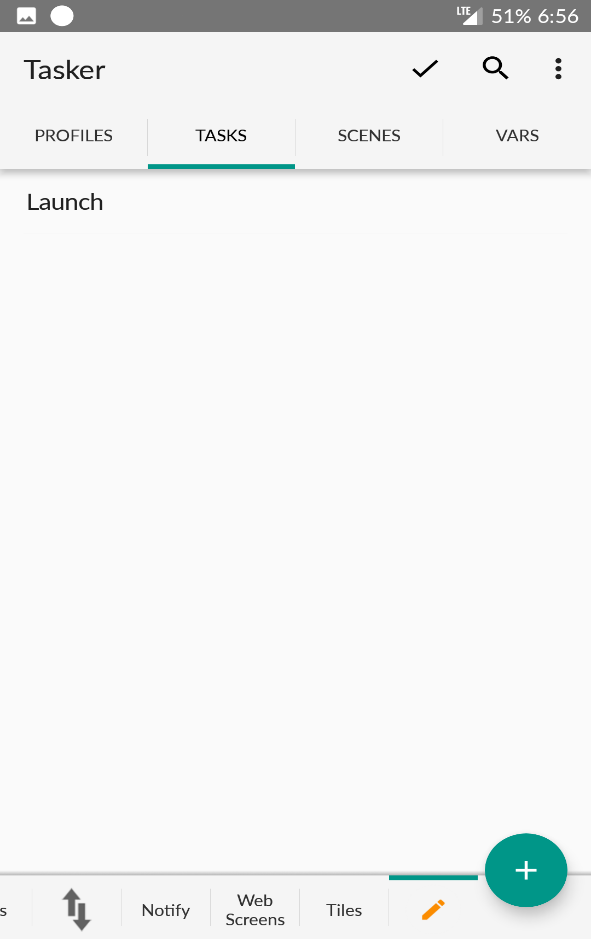
Composition of Message/Notification on Firebase



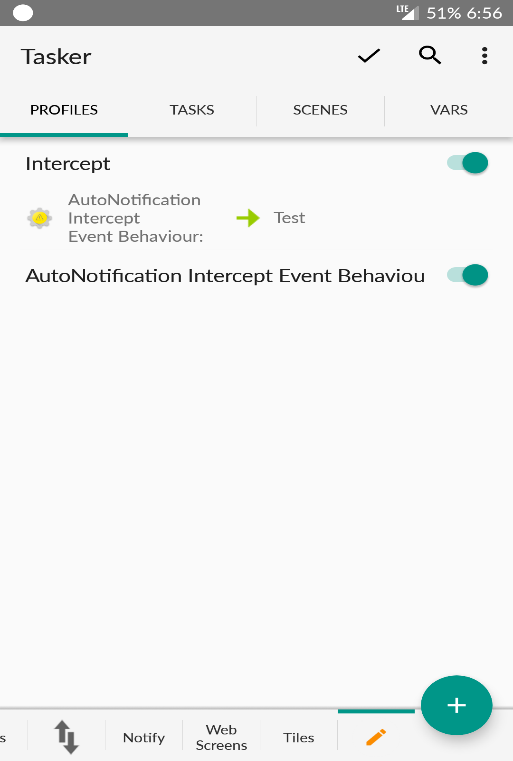
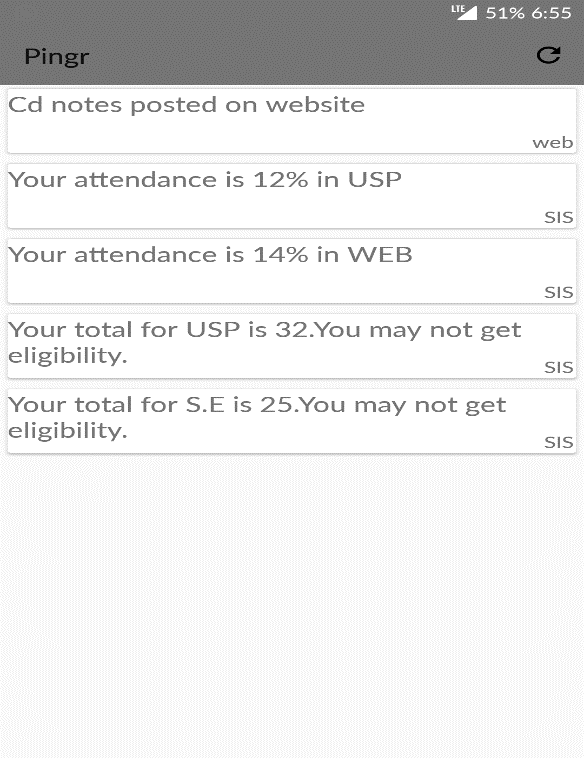
Notification Review



Overview of Sent Notifications

Notification on Mobile Launch Task

Intercept Profile Application Interface