

# **E-NOTICE APPLICATION**

**For Android Phones**

**REPORT**

SUBMITTED FOR THE PARTIAL FULFILLMENT OF THE DEGREE

at

**DigiMantra Labs, Ludhiana**  
**from January 5, 2014 to May 30, 2014**

SUBMITTED By

Priyanka Kapoor

Information Technology

106160

100371180720



**Information Technology Department**

**GURU NANAK DEV ENGINEERING COLLEGE**  
**LUDHIANA, INDIA**

## **Abstract**

e-Notice App helps you access online notices on your phone. It is an online notice board maker where a group of people can easily communicate with each other by sticking virtual notes. These notes can have text, images or include online videos such as from YouTube.

The notice board has always been the place where staff/students gathers to get their latest release of corporate news. eNotice brings the notice board to a virtual location where staff/students can not only read notices, but immediately react and respond to them - from their own desks! With this electronic notice and announcement system, notification may be sent out notifying staff that a new notice has been posted, where staff may know if it concerns him directly.

In this way, e-Notice Application also serves as a mailing list for all employees in the directory. This eliminates the need to keep a separate mailing list which is hard to maintain due to the rapid movement of staff.

## **Acknowledgment**

The author is highly grateful to the Dr. M.S. Saini (Director, Guru Nanak Dev Engineering College, Ludhiana) for providing this opportunity to carry out the six month training at Testing and Consultancy Cell, Guru Nanak Dev Engineering College, Ludhiana.

The constant guidance and encouragement received from Er. K. S. Mann (Dean Training and Placement Cell, Guru Nanak Dev Engineering College, Ludhiana) has been of great help in carrying out the project work and is acknowledged with reverential thanks.

The author would like to express a deep sense of gratitude and thanks profusely to Mr. Sachin Khosla (CEO, DigiMantra Labs, IT Park, Ludhiana). Without their wise counsel and able guidance, it would have been impossible to complete the report in this manner.

The help rendered by Mr. Rustam Singh, Associate Software Developer (DigiMantra Labs, Ludhiana) for experimentation is greatly acknowledged.

The author express gratitude to other faculty members of Information Technology department of Guru Nanak Dev Engineering College for their intellectual support throughout the course of this work.

Finally, I am indebted to all whosoever have contributed in this report work and friendly stay at Digi-Mantra Labs, Ludhiana.

**Priyanka Kapoor**

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Introduction To Organisation . . . . .	1
1.2	Introduction to Project . . . . .	3
1.3	Project Category (Mobile Application) . . . . .	5
1.4	Objective . . . . .	6
1.5	Problem Formulation . . . . .	7
1.6	Identification of Need . . . . .	8
1.7	Existing System . . . . .	9
1.8	Proposed System . . . . .	10
1.9	Unique Features of System . . . . .	12
<b>2</b>	<b>Requirement Analysis and System Specifications</b>	<b>14</b>
2.1	Feasibility Study . . . . .	14
2.2	Software Requirement Specification Document . . . . .	16
2.2.1	Data Requirements . . . . .	16
2.2.2	Functional Requirements . . . . .	16
2.2.3	Performance Requirements . . . . .	17
2.2.4	System Dependability . . . . .	18
2.2.5	Maintainability Requirements . . . . .	18
2.2.6	Security Requirements . . . . .	19
2.2.7	Look and Feel Requirements . . . . .	19
2.3	Validations . . . . .	21
2.4	Expected Hurdles . . . . .	23
2.5	SDLC Model Used . . . . .	24
<b>3</b>	<b>System Design</b>	<b>27</b>
3.1	Design Approach . . . . .	27
3.2	Detail Design . . . . .	27
3.3	System Design . . . . .	30

3.4	User Interface Design . . . . .	32
3.5	Database Design . . . . .	35
3.5.1	ER Diagrams . . . . .	36
3.5.2	Normalization . . . . .	37
3.5.3	Database Manipulation . . . . .	38
3.5.4	Database Connection Control and Strings . . . . .	38
3.6	Methodology . . . . .	40
<b>4</b>	<b>Implementation,Testing and Maintenance</b>	<b>42</b>
4.1	Introduction to Languages, IDE's,Tools and Technologies Used For Implementation . . .	42
4.1.1	Java . . . . .	42
4.1.2	Android Development Tools . . . . .	43
4.1.3	Security and Permission Concept in Android . . . . .	44
4.2	Coding Standards of Language Used . . . . .	46
4.3	Project Scheduling . . . . .	47
4.4	Test Plan and Test Activities . . . . .	50
4.4.1	Test Plan . . . . .	50
4.4.2	Test Activities . . . . .	51
<b>5</b>	<b>Results and Discussions</b>	<b>53</b>
5.1	User Interface Representation . . . . .	53
5.1.1	Brief Description of Various Modules . . . . .	53
5.2	SnapShots of System . . . . .	57
5.3	Back End Representations . . . . .	70
5.3.1	Snapshots of Database Tables . . . . .	70
<b>6</b>	<b>Conclusion and Future Scope</b>	<b>72</b>
6.1	Future Scope . . . . .	72
6.2	Conclusion . . . . .	73

# List of Figures

1.1	GNDEC . . . . .	2
2.1	Login Page . . . . .	24
3.1	Detailed Design . . . . .	29
3.2	Use Case Diagram For User . . . . .	30
3.3	Use Case Diagram For Admin . . . . .	31
3.4	Landing Page . . . . .	32
3.5	Registration Page . . . . .	32
3.6	Login Page . . . . .	33
3.7	Dashboard Of Notices . . . . .	33
3.8	Post Notice Page . . . . .	34
3.9	Reset Password Page . . . . .	34
3.10	ER Diagram . . . . .	37
3.11	Agile Software Development Methodology . . . . .	40
4.1	Graph of Commits on GitHub . . . . .	48
4.2	Code Frequency Graph on GitHub . . . . .	49
5.1	Google Cloud Messaging Working . . . . .	54
5.2	Landing Page . . . . .	57
5.3	Register Page . . . . .	58
5.4	Login Page . . . . .	59
5.5	DashBoard Page . . . . .	60
5.6	Post Notices Page . . . . .	61
5.7	Reset Password Page . . . . .	62
5.8	Intent Service Running In Application . . . . .	63
5.9	Notification Received . . . . .	64
5.10	Email Validation . . . . .	65
5.11	Mobile Number Validation . . . . .	66

5.12 Contextual Action Bar . . . . .	67
5.13 Search Box . . . . .	68
5.14 Notice Details . . . . .	69
5.15 Tables inside Database . . . . .	70
5.16 Schema of GCM User Table . . . . .	70
5.17 Schema of User Table . . . . .	71
5.18 Schema of Notices Table . . . . .	71

# Chapter 1

## Introduction

### 1.1 Introduction To Organisation

I had my six months industrial training at DigiMantra Labs, IT Park, Ludhiana. DigiMantra Labs is an IT company based in India. It was established in 2009. The CEO of this company is Mr. Sachin Khosla. DigiMantra mainly focus in development of preeminent Web, Mobile and Desktop solutions. It believes in delivering quality and cater diversified niche of industries. DigiMantra have a proven track record of delivering high performance solutions to various big to small level industries. This company is aided by Government of India.

DigiMantra Labs is a customary participant of various important events like IndiaSoft, ICT HongKong. It always look forward to build new relationships with like minded companies/ individuals. DigiMantra has a team of young dynamic professionals formed by the group of engineers who have graduated from reputed Engineering Colleges. Their team is lead by professional who has worked with big products like Limewire, Trulia etc.

Good thing about DigiMantra is that it is driven by ethical values, energy to build great apps and determination to deliver the best. They are keen to provide out of box solutions to the client keeping the cost to company aside. DigiMantra teams are trained to work under various software models such as Agile, Scrum etc. They make sure that they adhere to these standards so that they are able to deliver quality product All the activities at office are closely monitored and recorded using CCTV cameras. We just ensure that your business plan is safe with us. DigiMantra innovate and build our applications using latest technologies, which makes them fast, robust & scalable. This company contributes to open source software by organizing / participating in OSS Camps.

Their expertise is in the following fields:-



- Enterprise Level Product Development
- Internet Tools including Viral campaigns, Affiliate Marketing solutions
- Mobile Application Development; iOS & Android
- Cloud Computing Solutions SaaS & PaaS
- Website Development with Complete CMS and Social Networking Support
- User Experience & Product Designing
- Social Media Applications & Campaigning



Figure 1.1: DigiMantra Labs Logo

DigiMantra believes that 'communication' is the foremost ingredient to make a project successful. It communicate with the clients on regular basis and using the dedicated VOIP lines and various other ways of communication. This company understand different needs of different customers and hence have custom solutions based on the requirement. Their onsite team facilitation is one such example.

## 1.2 Introduction to Project

e-Notice App helps you access online notices on your phone. It is an online notice board maker where a group of people can easily communicate with each other by sticking virtual notes. These notes can have text, images or include online videos such as from YouTube.

The notice board has always been the place where staff/students gathers to get their latest release of corporate news. eNotice brings the notice board to a virtual location where staff/students can not only read notices, but immediately react and respond to them - from their own desks! With this electronic notice and announcement system, notification alerts may be sent out notifying staff and students that a new notice has been posted, where staff may know if it concerns him directly. In this way, e-Notice also serves as a mailing list for all employees in the directory. This eliminates the need to keep a separate mailing list which is hard to maintain due to the rapid movement of staff.

These are the features that an eNotice App should have:

- An electronic dashboard board for disseminating information out to staff and students.
- Notices can be posted, with response obtained instantly.
- Staff or student can be notified of new postings via notification alert.
- Notice administrator may push important notices in to selected staff's email.
- Notice administrator may create any notice category.

The interface of this application is straightforward and takes you roughly a minute to get started. Adding notes to board is easy, just click on the post notice button and enter the text. Users can view the post on the spot by having a notification alert in android phone. Here registration is must for all the users having this application in order they want to have notification and staying stuned.

This application can also acts as a market place and lets you advertise, for example, a room to rent, a car for sale, a job opportunity, a house for sale, an upcoming event, an announcement, a service, and so on.

This application creates interactive notice boards online. You can customize boards with a custom background, title and background images. It allows users to add sticky notes with text, images and videos. You can set preferences on who can edit and view the board. Facility of drag and drop notes anywhere on the board is available. Users can sign using their Google accounts / Facebook Account and can share boards with others.

### **1.3 Project Category (Mobile Application)**

”e-Notice App” is a mobile application i.e an Internet based Mobile application. A mobile application is a computer program designed to run on smartphones, tablet computers and other mobile devices. Apps are usually available through application distribution platforms, which began appearing in 2008 and are typically operated by the owner of the mobile operating system, such as the Apple App Store, Google Play, Windows Phone Store, and BlackBerry App World.

Mobile apps were originally offered for general productivity and information retrieval, including email, calendar, contacts, and stock market and weather information. However, public demand and the availability of developer tools drove rapid expansion into other categories, such as mobile games, factory automation, GPS and location-based services, banking, order-tracking, ticket purchases and recently mobile medical apps. The explosion in number and variety of apps made discovery a challenge, which in turn led to the creation of a wide range of review, recommendation, and curation sources, including blogs, magazines, and dedicated online app-discovery services.

Mobile application development is the process by which application software is developed for low-power handheld devices, such as personal digital assistants, enterprise digital assistants or mobile phones. These applications can be pre-installed on phones during manufacturing, downloaded by customers from various mobile software distribution platforms, or delivered as web applications using server-side or client-side processing (e.g. JavaScript) to provide an ”application-like” experience within a Web browser.

Application software developers also have to consider a lengthy array of screen sizes, hardware specifications and configurations because of intense competition in mobile software and changes within each of the platforms. Mobile app development has been steadily growing, both in terms of revenues and jobs created. The popularity of mobile apps has continued to rise, as their usage has become increasingly prevalent across mobile phone users.

## 1.4 Objective

The proposed system's objectives are to overcome all the limitations and drawbacks of the existing system. The online eNotice application is user-friendly android application. The main objective of the application is its simplicity of design and ease of implementation that shows and helps to collect most of the information about events going on in college premises. The interface will be very user-friendly.

The main objectives of the proposed system can be enumerated as follows:

- Faster dissemination of notices regarding education, technical events, cultural events.
- Any lost/found going out in college.
- Easy way to broadcast your message.
- Helps you to be updated with whats going on in College.
- Good way to advertise about Tuitions/ Coaching and Courses.
- User can also follow a group notice board.

## **1.5 Problem Formulation**

To develop a mobile application that will help you receiving the notices from the college, anywhere , anytime. Earlier their was problem that notices were pasted on notice board. If there is holiday on the next day, nobody will be able to read it. Moreover any update on website is also very difficult. Everyone feels lazy to go and update the website data. The more easy way is, just type in message sitting wher-ever and post by pressing a button. It will notify all the staff and students, registered with that application.

## **1.6 Identification of Need**

1. As we discussed earlier that manual maintenance of a notices is a tedious job. So to enhance the ease of working, we go for this package.
2. Giving the facility to convey messages to all students anytime and anywhere.
3. Making students updated about all the events and activities going on in the college.
4. The student will not require to stand in the crowd to see the notice. There wil be no issue of fighting in order to see the notice first. Everyone is first to see that notice inside thier own mobile phone anywhere and anytime.
5. The least but most important it saves time.
6. Utilizing less man power. As there are many persons involved in circulating the message. With this application, only one person is required to post the notice. Rest of the man power is saved in the entire process.

## 1.7 Existing System

Currently our college has manual system of putting notices on notice board. Its outdated now. As nobody has a time to stand in rush in order to read the notices on noticeboard.

### *Limitations of Existing System:*

1. **Order of Data:** Notice can get out of order in traditional notice board system. If someone accidentally puts some data in the wrong place, it can lead to lost data. Automated notice management systems allow users to quickly check whether information already exists somewhere in the system, which helps avoid problems like redundant data.
2. **Complexity:** Automated system is less complex than manual system of handling notices, which can make it easier for untrained people to access and manipulate data. Anyone having the basic knowledge of mobiles can work on the automated system.
3. **Inconsistency of data:** There will be an unavailability for future use, since notice might get misplaced during manual notices management. So notice won't be preserved properly for future use.
4. **Damage:** Manual notices stack are vulnerable to damage, destruction and theft in ways that digital databases are not. A company may back up its digital data both on site and at offsite locations, ensuring its security if the office building suffered a fire or similar disaster. A manual database, however, may only exist in one place without any copies. As a result, a manual database would be very vulnerable to a fire or other natural disaster. In addition, while access time in a manual database system, information must be found by hand rather than electronically. While a digital database will typically allow users to search the entire database for specific information in seconds, someone looking for information in a manual system may have to spend hours searching for a particular piece of data.
5. **Editing and Communication:** Manual notices do not allow users to easily edit data or information. Manual notices often cannot be edited directly, forcing users to make new copies. To circulate notice on paper, users must require peons and other staff. e-Notice app allow users to edit information fields directly, and because data is stored digitally, it is already in a form that can be easily transmitted.



## 1.8 Proposed System

Proposed System will be able to do the following:

1. ***To eliminate wastage of time and energy:***

e-Notice app will be able to save lot of paper and time. It directs both teacher and pupils energy and attention to one thing at a time by placing proper persons at their proper places at the proper time. Everything will be instanteneous.

2. ***To avoid duplication and overlapping:***

This application will help to remove the duplicacy of notices. Only one person, who is admin can post the notice. No one else would be able to do so. Soo student and staff will be given correct information all the time.

3. ***To ensure due attention of student to each and every notice:***

e-Notice App ensures that everyone has kind attention to every notice and updates going on in college. There will be a buzz at each and every notice to drive the attention of student to check it once. In this way, students will be well informed about their college activities.

4. ***To bring system into college life:***

It would be dire need of all colleges as its easy and shortcut method to inform all the students. In the absence of proper notification system will make it very difficult to inform students at right time.

5. ***Searching a particular Notice:***

This application allows you search the notice very easily through title of notice. If anyone forgots about the notice details, he can search it out very easily.

6. ***Free Service:***

It gives free service to notify all the students. There will be no cost of sending notification to all. Just have the good system implemented in college and that too free of cost.

7. ***Prevent Crowd in College:***

As you can see, there is always a crowd at notice board. As notice board is one, and people to see notice are more. With this application there will be no more crowd. Everyone will be well informed even at their homes. So they are free to do there other work.

8. ***Automatically Updated Dashboard:***

The dashboard of notice is automatically updated when a new message arrives. The user can himself refresh the dashboard to see any new notice.

9. ***Anytime Anywhere Service:***

With this application, notices will be delivered anytime and at any place. There is no restriction of time to send a notice.

10. ***Keeping Notices at one place:***

This application allow you to have notices in one place only. If there is an attachment with that, all will be placed in a separate folder dedicated to that application. So there will be no here and there of notices.

## 1.9 Unique Features of System

The unique features of this application are as follow:

1. ***GCM Notification:*** Google Cloud Messaging has been used to broadcast notices to all the students who are registered with this application. It will help you receive the notice even if you application is closed. So it implements anywhere anytime notifications. Google Cloud Messaging (GCM) is a free service for sending messages to Android devices. GCM messaging can greatly enhance the user experience. Your application can stay up to date without wasting battery power on waking up the radio and polling the server when there are no updates. Also, GCM allows you to attach up to 1,000 recipients to a single message, letting you easily contact large user bases quickly when appropriate, while minimizing the work load on your server.
2. ***Battery Saving Application:***  
This application saves your battery too. Its because, the service implemented in application is not running all the time. Whenever GCM ping the mobile, only then it makes a broadcast to phone that initiates the service. In this way, its saving your battery alot.
3. ***Automatically Updated Dashboard:***  
The dashboard of notice is automatically updated when a new message arrives. The user can himself refreah the dashboard to see any new notice.
4. ***Anytime Anywhere Service:***  
With this application, notices will be delivered anytime and at any place. There is no restriction of time to send a notice.
5. ***Keeping Notices at one place:***  
This application allow you to have notices in one place only. If there is an attachment with that, all will be placed in a separate folder dedicated to that aplication. So there will be no here and there of notices.
6. ***Free Service:***  
It gives free service to notify all the students. There will be no cost of sending notification to all. Just have the good system implemented in college and that too free of cost.

#### **7. *Case Pictures in One Folder:***

The attachments of notices goes at one place only. If you want to see all attachments, you can go in gallery and can see it. Its a good feature of this application that cares for our files organisation in phone too.

# Chapter 2

## Requirement Analysis and System Specifications

### 2.1 Feasibility Study

Depending on the results of the initial investigation, the survey is expanded to a more detailed feasibility study. Feasibility study is a test of system proposal according to its work ability, impact on the organization, ability to meet user needs, and effective use of resources. The objective for this phase is not to solve the problem but to acquire a sense of scope. During the study, the problem definition is crystallized and aspects of the problem to be included in the system are determined.

Mobile Application Development Systems are capital investments because resources are being spent currently in order to achieve benefits to be received over a period of time following completion. There should be a careful assessment of each project before it is begun in terms of economic justification, technical feasibility, operational impact and adherence to the master development plan. We started the project by listing the possible queries that the user might want to be satisfied. And on these lines we guided the project further.

The three main points, kept in mind at the time of project, are :

- Possible (To build it with the given technology and resources)
- Affordable (given the time and cost constraints of the organization)
- Acceptable (for use by the eventual users of the system)

The three major areas to be considered while determining the feasibility of a project are :

1. **Technical Feasibility:** The technical issue usually raised during the feasibility stage of the investigation includes the following :

- Does the necessary technology exist to do what is suggested?

- Do the proposed equipments have the technical capacity to hold the data required to use the new system?
- Will the proposed system provide adequate response to inquiries, regardless of the number or location of users?
- Can the system be upgraded if developed?
- Are there technical guarantees of accuracy, reliability, ease of access and data security?

Earlier no system existed to cater to the needs of Secure Infrastructure Implementation System. The current system developed is technically feasible. It is a web based user interface. Thus it provides an easy access to the users. The databases purpose is to create, establish and maintain a work-flow among various entities in order to facilitate all concerned users in their various capacities or roles. Permission to the users would be granted based on the roles specified. Therefore, it provides the technical guarantee of accuracy, reliability and security. The software and hardware requirements for the development of this project are not many and are already available as free as open source. The work for the project is done with the current equipment and existing software technology. Necessary bandwidth exists for providing a fast feedback to the users irrespective of the number of users using the system.

2. **Operational Feasibility:** Under this category of service we conduct a study to analysis and determine whether your need can be fulfilled by using a proposed solution. The result of our operational feasibility Study will clearly outline that the solution proposed for your business is operationally workable and conveniently solves your problems under consideration after the proposal is implemented. We would precisely describe how the system will interact with the systems and persons around. Our feasibility report would provide results of interest to all stakeholders. It will do as per the needs of the business requirements.
3. **Timeline Feasibility:** It is important to understand that a need must be fulfilled when it has to be. Some otherwise feasible and highly desirable projects can become non-feasible due to very restrictive timeline constraints. This fact makes it imperative that milestones are clearly linked to the timeline and projects are well conceived with safe unforeseen margins. We make sure that we strictly follow what has been stated above.

## 2.2 Software Requirement Specification Document

### 2.2.1 Data Requirements

Data requirement is meant to be the data that will be used in our application. Data required in this project is all notices, that need to be conveyed to the user. This application also require the username and passwords of persons in order to register them and sending notification about updates. So two main requirements are:

- Notice Details
- User Details

### 2.2.2 Functional Requirements

In order to make this application functional, we require the following:

- ***Download mobile application:***

A user should be able to download the mobile an application through either an application store or similar service on the mobile phone. The application should be free to download.

- ***User registration:***

Given that a user has downloaded the mobile application, then the user should be able to register through the mobile application. The user must provide user-name, password and e-mail address. The user can choose to provide a regularly used phone number.

- ***User Login:***

Given that a user has registered, then the user should be able to log in to the mobile application. The log-in information will be stored on the phone and in the future the user should be logged in automatically.

- ***Reset Password:***

Given that a user has registered, then the user should be able to retrieve his/her password by e-mail.

- ***DashBoard:***

Given that a user is logged in to the mobile application, then the first page that is shown should be the dashboard page. The user should be able to see all the college notices.

- ***Search Notice:***

The user should be able to search for a notice by its title. For example, if a user types fee, all the notices having fee in their content get displayed.

- ***Selecting a Notice:***

A user should be able to select any notice from list view. The click on particular notice will take him to notice details of that particular notice.

- ***Navigating back to Notices List:***

The user should be able to navigate back to notices list from the notice details section. This is required to give a good user experience.

- ***Deleting Notices:***

The user should have the option to delete the unnecessary notices from his phone, by ticking them one by one and then deleting them in one go. This way, user can save this phone memory from unrequired notices.

- ***Posting Notices:***

The admin of this application should be able to post the notices. He should be able to add a picture within notices. That picture can be taken either from gallery or by using the camera of the mobile phone.

- ***Notification Alert:***

All the registered users should be able to have a ping or notification on their mobile phone whenever a new notice is posted.

### **2.2.3 Performance Requirements**

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

- ***Prominent search feature:***

The search feature should be prominent and easy to find for the user.

- ***Usage of the Notice Information:***

The notice link should be prominent and it should be evident that it is a usable link. Selecting the notice link should only take one click.



- ***Response Time:***

The response time should not be more than 5 seconds if user have a proper internet connection.

- ***Fault Tolerance:***

The fault tolerance of the system should be very good. If the system loses the connection to the Internet or the system gets some strange input, the user should be informed.

## **2.2.4 System Dependability**

Following are the requirements that an application require from the device/mobile on which it is installed.

- ***Internet Permission:***

Application developed, require full internet permissions of mobile so that it can fetch notices from the server. At the same time, it should be able to receive buzz or notification tone whenever new notice is posted by admin.

- ***External SD Card Writable Permissions:***

This application would be requiring read write access to SD card. It is required in order to download the notices attachment and save in SD card of mobile phone.

- ***System Tools:***

This application require various system tools to be used. For example, it requires Camera of mobile in order to click the image and post in into notice. It also require system tool, that prevents it from sleeping.

- ***Hardware Control:***

It uses vibrator of mobile phone whenever any notification arrives.

- ***Account Info:***

It also fetches your google account information in order to get the user registered with Google Cloud Messaging.

## **2.2.5 Maintainability Requirements**

Following are the maintainability requirement of e-Notice mobile application:

- ***Application extendibility:***

The application should be easy to extend. The code should be written in a way that it favors implementation of new functions. It is requires in order for future functions to be implemented easily to the application.

- ***Application testability:***

Test environments should be built for the application to allow testing of the applications different functions.

## **2.2.6 Security Requirements**

- ***Communication Security:***

There should be security of the communication between the system and server. The messages should be encrypted for log-in communications, so others cannot get user-name and password from those messages. Every exchanged of information between client and server should be encrypted so that no one can track it.

- ***Admin Login Account Security:***

If an admin tries to log in to the web portal with a non-existing account then the admin should not be logged in. The admin should be notified about log-in failure.

- ***Admin Account Security:***

There should be security of admin accounts. An admin and IP address should not be able to log-in to the web portal for a certain time period after three times of failed log-in attempts.

- ***User Create Account Security:***

The security of creating account for users of the system should be maintained. If a user wants to create an account and the desired user name is occupied, the user should be asked to choose a different user name.

## **2.2.7 Look and Feel Requirements**

Regarding look and feel, our client is straight forward. They believe in simplicity. So these are their requirements:

- ***Simple and Light:***

The user interface should be simple and lightly colored. It should give relaxing effect on looking at its GUI. No bright colors should be used while designing the UI of this application.

- ***Easy to Use***

The application should be easy to use. If any user is doing something wrong, he/she should be informed correctly, what is going wrong behind the scene. There should be proper instructions for the user to use this application.

- ***Soft Sound Notification:***

The sound for notification should be very soft. It should not disturb the users with a loud note. Everything should be sober in this application.

## 2.3 Validations

Any application is useless without validation. There should be a way to validate the user input first before sending the user request to the server. Following are the validations implemented in proposed system:

- ***User Password Validation:***

The application should check the user and password fields before sending any request to the server. It should check whether the fields are filled or not. If fields are not filled up, user should be instructed to fill up the fields before moving further. In this way, there will be less traffic on the server.

- ***Validations During Registration:***

There are a lot of validations that need to be implemented in the application. They are as follow:

1. ***First and Last Name of User:***

The first and last name of user should be not null. Also first letter of first and last name should be in uppercase.

2. ***Username:***

The username can contain only alphabets, digits, underscore and hyphen. It should be at least 3 characters long and maximum of 15 characters.

3. ***Password:***

The password must contain one digit from 0-9, one lowercase character, one uppercase character, one special symbol in the list ”#\$%” and length of password must be at least 6 characters and maximum of 20.

4. ***Email:***

The application must validate and email address entered by the user before sending request to the server.

5. ***Mobile Number:***

The mobile number should be of only ten digits. No more, no less than that.

- ***Validating During Posting Notices:***

The application should validate the notice posting fields before posting any notice. It should check

whether title and description fields are filled or not. if not, it should tell the user to fill up the required fields while posting the notice.

- ***Reset Password Validation:***

The application should check that user has entered the username or email in the given field before pressing the reset password button.

## 2.4 Expected Hurdles

The main hurdles that can come in the notice are as follow:

1. ***GCM Notification:***

Its possible that there may be a problem with receiving GCM notifications. When to receive a broadcast signal and when to start the service. If service is not started at the proper time, there may be the case, user will not be able to receive any notification from the server.

2. ***Device Database:***

Its very much difficult to see entries going inside the device database. You have to check through coding. You can't access the applications database easily. So its bit hard to debug the database errors.

3. ***Device SD Card:***

The application will be requiring access to the SD card of the user mobile. It may be possible that SD Card is full or missing SD Card. So the user will not be able to receive the respective attachment of notices.

4. ***Google Play Services:***

Registering to the GCM Server requires to have Google Play services on device installed. Otherwise user will not be able to register to GCM server nor will be able to receive any message.

## 2.5 SDLC Model Used

This section describes the project as per the various stages of the Software Development life cycle. The model of software development life cycle used in this project is the waterfall method. The Waterfall Method is comprised of a series of very definite phases, each one run intended to be started sequentially only after the last has been completed, with one or more tangible deliverables produced at the end of each phase of the waterfall method of SDLC. Essentially, it starts with a heavy, documented, requirements-planning phase that outlines all the requirements for the project, followed by sequential phases of design, coding, test-casing, optional documentation, verification (alpha-testing), validation (beta-testing), and finally deployment/release.

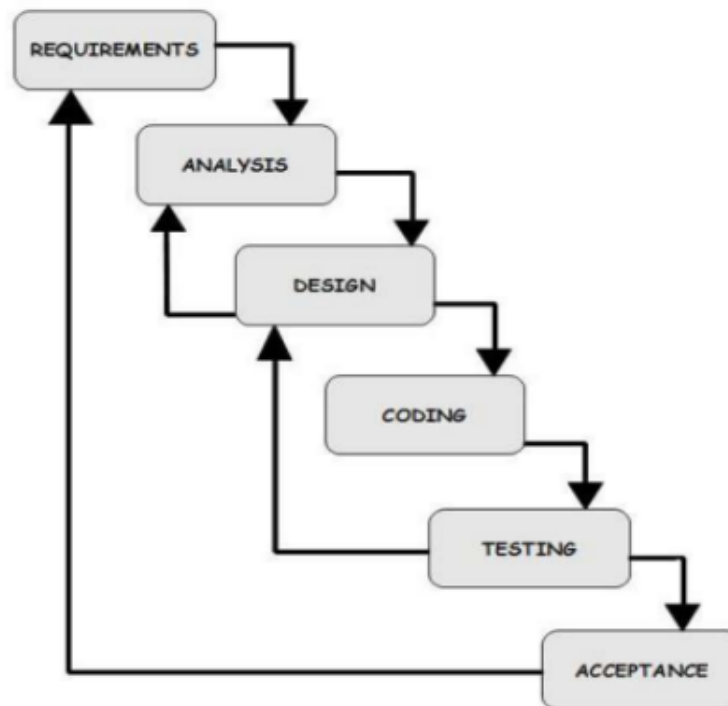


Figure 2.1: Login Page

### 1. *Requirement Analysis:*

Existing system is time consuming and it makes difficult to convey huge amount of users about any event, class or seminar almost instantly. Also there is always a big crowd in front of noticeboard. So it was hectic to read any useful instruction and information. Thus all the problems of the

existing system are summarized and proposing a new system that works as an online application. It is a value added solution to the problem. It resolves all the problems stated above. It will provide simple interface to the user to operate on and convey the intended users about events almost instantly, anytime and anywhere.

## 2. ***Design:***

It includes translation of the requirements specified in the SRS into a logical structure that can be implemented in a programming language. The output of the design phase is a design document that acts as an input for all the subsequent SDLC phases. The design of this app is simple and user friendly containing six main activities, namely:

- (a) Register
- (b) Login
- (c) Dashboard
- (d) Details of Notices
- (e) Admin Panel
- (f) Reset Password

## 3. ***Coding/Implementation:***

It includes translation of the requirements specified in the SRS into a logical structure that can be implemented in a programming language. The output of the design phase is a design document that acts as an input for all the subsequent SDLC phases. The project is implemented using the Android virtual device (AVD). This emulator helped to implement the project in a real-like environment and sketch out the details of how it will work on a real hardware. Each activity is linked with another and interconnectivity is transparent and smooth.

## 4. ***Testing:***

It includes detection of errors in the application. The testing process starts with a test plan that recognizes test-related activities, such as test case generation, testing criteria, and resource allocation for testing. The code is tested and mapped against the design document created in the design phase. The output of the testing phase is a test report containing errors that occurred while testing the application. Testing of the project has not been done on real hardware and also on the emulator or software environment. Testing has been done for each of the individual activities of the project.



#### 5. *Maintenance:*

It includes implementation of changes that software might undergo over a period of time, or implementation of new requirements after the software is deployed at the customer location. The maintenance phase also includes handling the residual errors that may exist in the software even after the testing phase. The project maintenance is low cost and efficient as user will get this application at free of cost and also this application is shared over network, therefore maintenance is little bit difficult.

# Chapter 3

## System Design

### 3.1 Design Approach

A design approach is a general philosophy that may or may not include a guide for specific methods. Some are to guide the overall goal of the design. Other approaches are to guide the tendencies of the designer. A combination of approaches may be used if they don't conflict.

*Function Oriented Design Approach:*

Function Oriented Design Approach is partitioning of a design into subsystems and modules, with each one handling one or more functions. Contrast with object-oriented design, data-structure-oriented design.

This application project uses function oriented design approach. Every module and sub modules are made, based on their functionality. These modules are designed and implemented separately and then they are integrated together to form the desired application.

### 3.2 Detail Design

The detailed design of this application is as follow:

1. ***Registering a User:***

The first step in this application is to get the users registered to both GCM Server and to Remote Web Server. For this, user will provide all the necessary details and press the register button. The request will first go to Google Cloud Messaging Server. GCM Server will provide the registration id for that device. After that, all the information along with registration id is stored on Web Server and the user gets registered.

2. ***User Login:***

After registering, the user is allowed to log in. Username and password after validating at client

side, is sent to server side to authentication. After authentication response is sent by the server to client, and then user gets logged in.

### 3. ***Viewing the Notices:***

At the first time, when you are using this application for the first time, it will fetch all the notices from server. In all the other case, all previous notices are fetched from application's own database stored inside client mobile. It then checks for new notices from the server. If there are new notices on the server, it will fetch all those notices.

### 4. ***Searching a Notice:***

The user is able to search the notice in listview depending on the title of the notice. It helps user to get the desired notice instantly.

### 5. ***Deleting a Notice:***

If the user does not want some notices, he/she can delete it from their phone. There will be no effect on server entry.

### 6. ***Posting a Notice:***

If a user is an admin, he is able to post the notice. In order to post the notices, he has three option. One option is that, he can post a simple text notice. Another option allows him to send some attachment image with the notice. In this, he has two options. Either he can pick the image from the gallery or he can click a picture on the spot by using camera. After that, press the post button to post the notice.

### 7. ***Notification Buzz:***

As soon as the admin post a notice, the script will run with which request is made by GCM Server to WebServer for all the registered IDs. After getting all the registered IDs, notification is sent to all the users registered with this application. Notification has a tune and vibration that runs whenever there is a notification received by the user from GCM Server.

### 8. ***Reset Password:***

This application also has the facility to reset the password. If one user has forgot his password, he/she can reset the password by giving his username or email address. The user will be given a page in which he can set his new password. Forgotten password will be updated with the new one on the server.

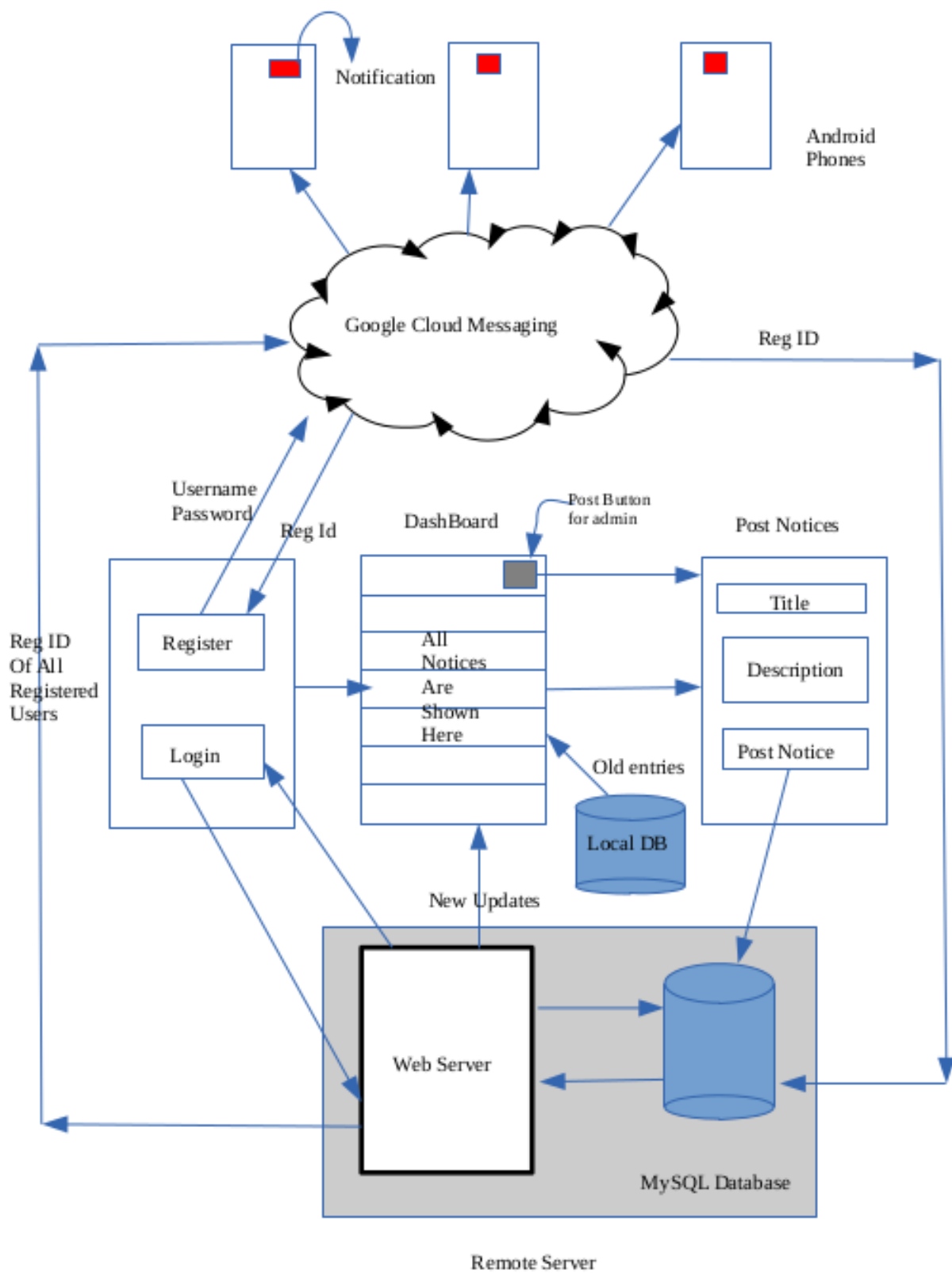


Figure 3.1: Detailed Design

### 3.3 System Design

The system design can be clearly explained from the following diagrams: *Use Case Diagram*:

A Use Case diagram at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the different types of users of a system and the various ways that they interact with the system. This type of diagram is typically used in conjunction with the textual use case and will often be accompanied by other types of diagrams as well. There are two types of user in this application, user and admin. Following depicts their use case diagram:

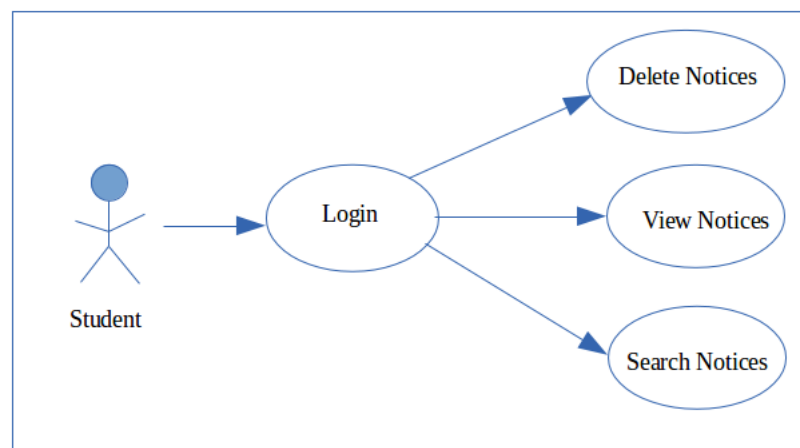


Figure 3.2: Use Case Diagram For User

This diagram is showing what a normal user can do with this application. The user can login, after that he can view the notices, delete the notices and can search for particular notices.

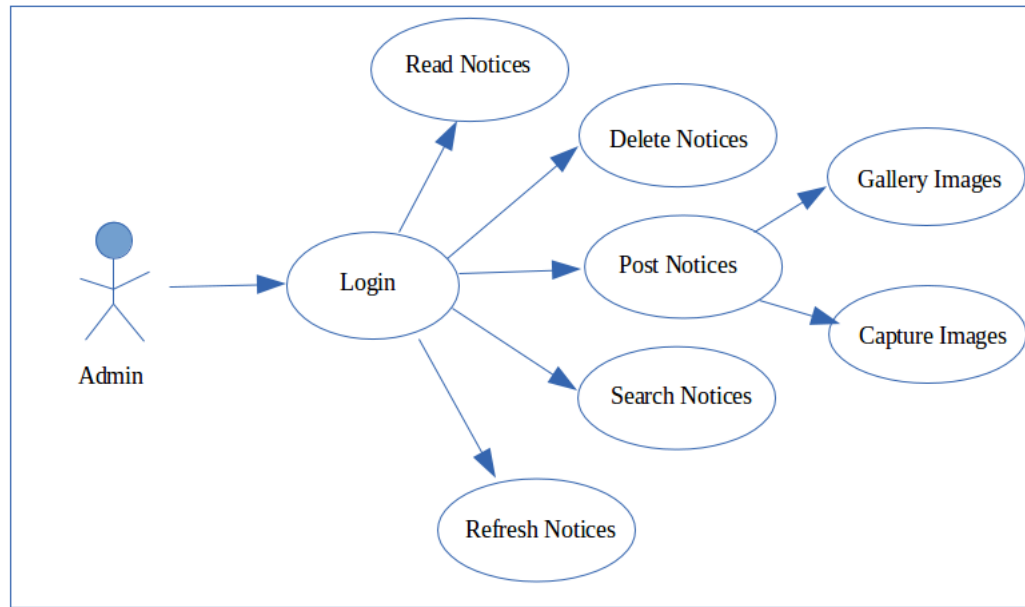


Figure 3.3: Use Case Diagram For Admin

This diagram shows the privileges of admin. An admin can post the notice in addition to viewing and deleting it. He can also post images as an attachment to the notices. Images can be chosen from the gallery or he can click the picture instantly using this application.

## 3.4 User Interface Design

User Interface Design means the design of application with which the user interacts. So it should be kept in mind that UI should be very simple and easy to use. It should be simple enough in look and feel also.

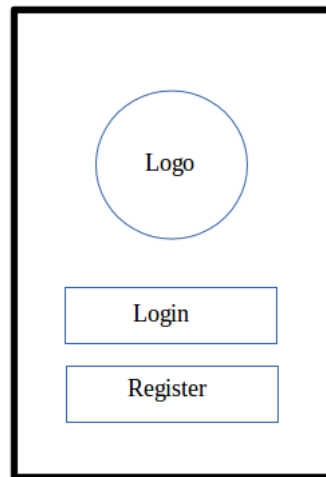


Figure 3.4: Landing Page

This page is the first page which is presented to the user. It has two buttons, to login or to register.

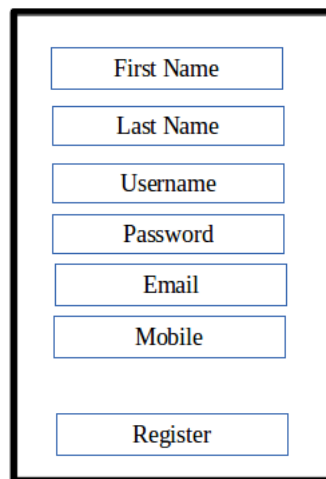
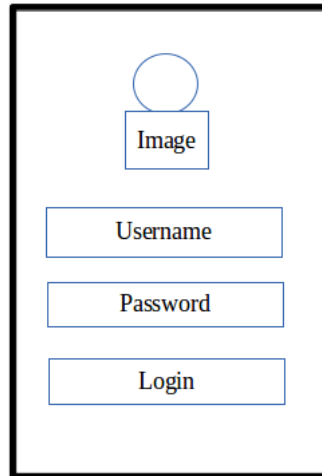
The image shows a registration form layout. It consists of a central vertical rectangle. Inside this rectangle, there are six rectangular input fields stacked vertically. The labels for these fields are "First Name", "Last Name", "Username", "Password", "Email", and "Mobile". Below these input fields is a single rectangular button labeled "Register". All elements are centered within the rectangle.

Figure 3.5: Registration Page

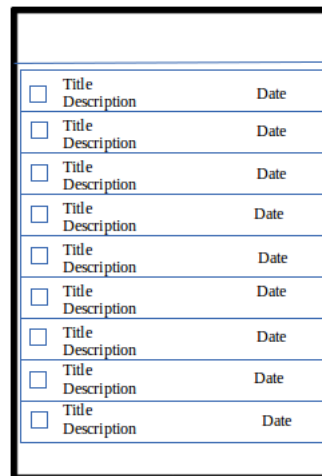
This is the registration page where the user get himself registered with the servers of this application.



A login page form with a black border. At the top is a circular placeholder for a profile picture, with the word "Image" written below it. Below this are three rectangular input fields stacked vertically, labeled "Username", "Password", and "Login".

Figure 3.6: Login Page

This is the login page where user enters his username and password in order to access the notices.

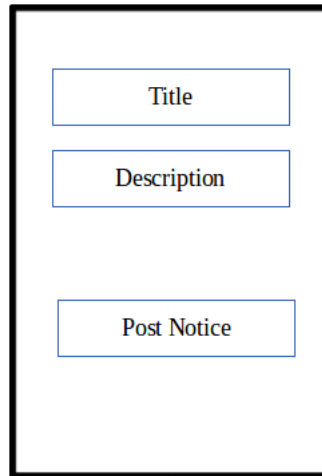


<input type="checkbox"/>	Title Description	Date
<input type="checkbox"/>	Title Description	Date
<input type="checkbox"/>	Title Description	Date
<input type="checkbox"/>	Title Description	Date
<input type="checkbox"/>	Title Description	Date
<input type="checkbox"/>	Title Description	Date
<input type="checkbox"/>	Title Description	Date
<input type="checkbox"/>	Title Description	Date
<input type="checkbox"/>	Title Description	Date
<input type="checkbox"/>	Title Description	Date

Figure 3.7: Dashboard Of Notices

This is the main dashboard, which receives all the notices from the user. The user can click on any notice, to see its details. In addition to it, user can search and delete the notices.

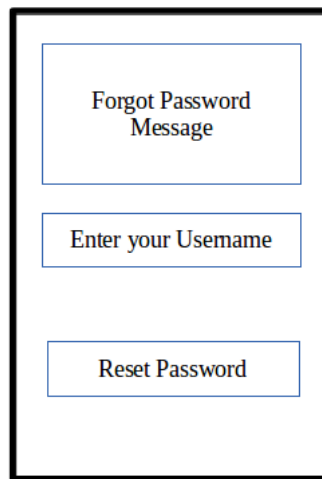




A diagram of a 'Post Notice' page. It consists of a large black rectangular frame containing three smaller, horizontally-oriented rectangular boxes stacked vertically. The top box is labeled 'Title', the middle box is labeled 'Description', and the bottom box is labeled 'Post Notice'.

Figure 3.8: Post Notice Page

This page is only for the admin. Only admin can post the notices. He can choose images as attachment to notices.



A diagram of a 'Reset Password' page. It consists of a large black rectangular frame containing three smaller, horizontally-oriented rectangular boxes stacked vertically. The top box is labeled 'Forgot Password Message', the middle box is labeled 'Enter your Uername', and the bottom box is labeled 'Reset Password'.

Figure 3.9: Reset Password Page

This is Reset Password page where the user can reset the password in order if he forgots his password.

## 3.5 Database Design

Database design is the process of producing a detailed data model of a database. This logical data model contains all the needed logical and physical design choices and physical storage parameters needed to generate a design in a data definition language, which can then be used to create a database. A fully attributed data model contains detailed attributes for each entity.

The term database design can be used to describe many different parts of the design of an overall database system. Principally, and most correctly, it can be thought of as the logical design of the base data structures used to store the data. In the relational model these are the tables and views. In an object database the entities and relationships map directly to object classes and named relationships. However, the term database design could also be used to apply to the overall process of designing, not just the base data structures, but also the forms and queries used as part of the overall database application within the database management system (DBMS).

The process of doing database design generally consists of a number of steps which will be carried out by the database designer. Usually, the designer must:

- Determine the relationships between the different data elements.
- Superimpose a logical structure upon the data on the basis of these relationships.

### *Design process*

1. Determine the purpose of the database - This helps prepare for the remaining steps.
2. Find and organize the information required - Gather all of the types of information to record in the database, such as product name and order number.
3. Divide the information into tables - Divide information items into major entities or subjects, such as Products or Orders. Each subject then becomes a table.
4. Turn information items into columns - Decide what information needs to be stored in each table. Each item becomes a field, and is displayed as a column in the table. For example, an Employees table might include fields such as Last Name and Hire Date.

5. Specify primary keys - Choose each table's primary key. The primary key is a column, or a set of columns, that is used to uniquely identify each row. An example might be Product ID or Order ID.
6. Set up the table relationships - Look at each table and decide how the data in one table is related to the data in other tables. Add fields to tables or create new tables to clarify the relationships, as necessary.
7. Refine the design - Analyze the design for errors. Create tables and add a few records of sample data. Check if results come from the tables as expected. Make adjustments to the design, as needed.
8. Apply the normalization rules - Apply the data normalization rules to see if tables are structured correctly. Make adjustments to the tables

### **3.5.1 ER Diagrams**

An ER diagram is a diagram that helps to design databases in an efficient way. Attributes in ER diagrams are usually modeled as an oval with the name of the attribute, linked to the entity or relationship that contains the attribute.

Within the relational model the final step can generally be broken down into two further steps, that of determining the grouping of information within the system, generally determining what are the basic objects about which information is being stored, and then determining the relationships between these groups of information, or objects.

An Entity Relationship Diagram (ERD) is a visual representation of different data using conventions that describe how these data are related to each other. ER diagrams are most often associated with complex databases that are used in software engineering and IT networks. In particular, ER diagrams are frequently used during the design stage of a development process in order to identify different system elements and their relationships with each other. For example, an inventory software used in a retail shop will have a database that monitors elements such as purchases, item, item type, item source and item price.

The ER diagram of database used in the project is as follow:

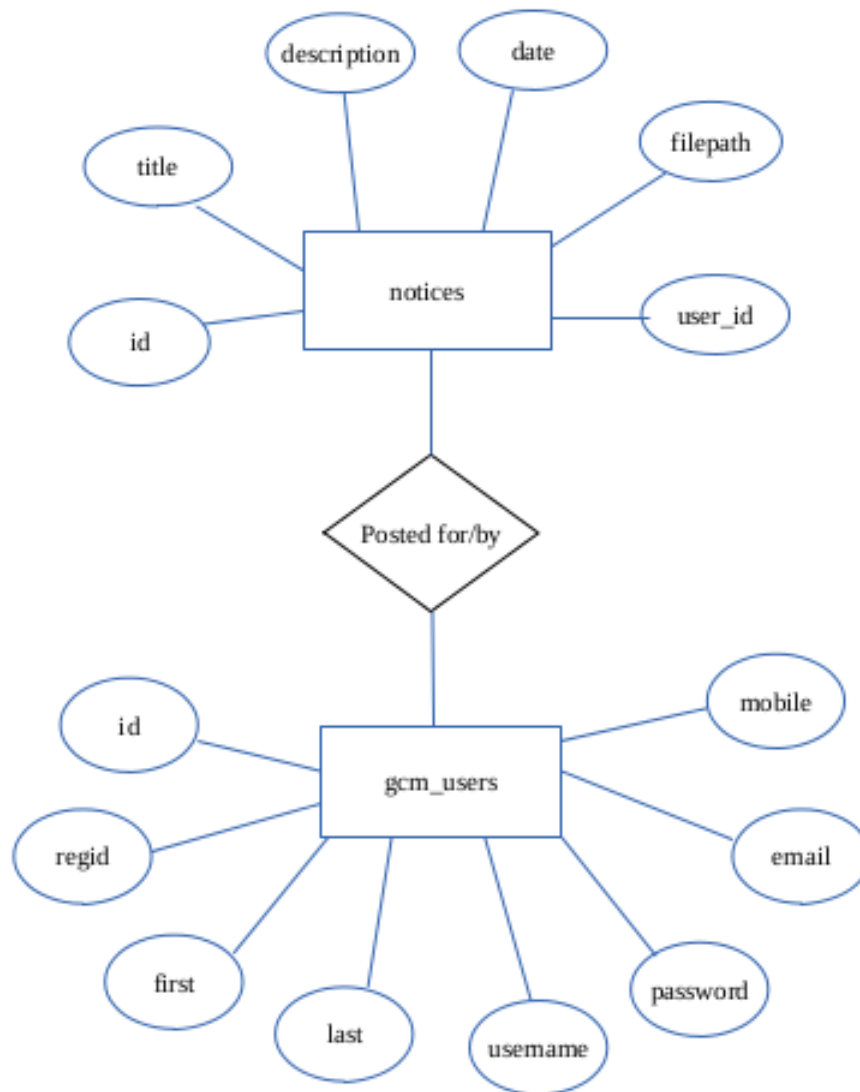


Figure 3.10: ER Diagram

### 3.5.2 Normalization

Normalization is a systematic way of ensuring that a database structure is suitable for general-purpose querying and free of certain undesirable characteristicsinsertion, update, and deletion anomalies—that could lead to a loss of data integrity.

A standard piece of database design guidance is that the designer should create a fully normalized design; selective denormalization can subsequently be performed, but only for performance reasons. However,

some modeling disciplines, such as the dimensional modeling approach to data warehouse design, explicitly recommend non-normalized designs, i.e. designs that in large part do not adhere to 3NF. Normalization consists of normal forms that are 1NF, 2NF, 3NF, BOYCE-CODD NF (3.5NF), 4NF and 5NF.

### 3.5.3 Database Manipulation

Data Manipulation is retrieval of information from the database, insertion of new information into the database, deletion of information in the database modification of information in the database. A DML is a language which enables users to access and manipulate data. The goal is to provide efficient human interaction with the system.

There are two types of DML:

1. *Procedural*: In it, the user specifies what data is needed and how to get it.
2. *Non Procedural*: In it, the user only specifies what data is needed. It is easier for user and may not generate code as efficient as that produced by procedural languages.

A query language is a portion of a DML involving information retrieval only. The terms DML and query language are often used synonymously.

### 3.5.4 Database Connection Control and Strings

1. ***Database connection control***: It is a feature which prevents users from connecting to a database. This feature is also called passive shutdown because when connection control is invoked, users who are currently connected to the database remain unaffected until they disconnect. This capability is useful if you need to acquire exclusive access to a database to perform maintenance operations, such as compacting the database, or make updates to the database's design.

When connection control is invoked, users who are currently connected to a database remain unaffected until they disconnect from the database. At that point they will be unable to reconnect to the database until the user who invoked connection control revokes it or closes his or her connection to the database.

The following scenarios provide examples of how the connection control feature works. Assume there are five users in the database and that user five has invoked connection control on the database:

#### **Scenario 1**

User six tries to connect to the database, but is denied access, and an error message is returned stating that user five is preventing the database from being opened.

#### **Scenario 2**

User one closes the database and tries to reconnect to the database, but is denied access, and an error message is returned stating that user five is preventing the database from being opened.

#### **Scenario 3**

User five closes the database. User six tries to open the database and is successful. This is because connection control only persists while the user who invoked it remains connected to the database.

#### **Scenario 4**

Users one through four exit the database. User five uses the ADO OpenSchema method to retrieve a list of users in the database and determines that no other users are in the database. User five compacts the database, closes the database, and connection control is automatically revoked.

2. **Connection String:** A connection string is a string that specifies information about a data source and the means of connecting to it. It is passed in code to an underlying driver or provider in order to initiate the connection. Whilst commonly used for a database connection, the data source could also be a spreadsheet or text file.

The connection string may include attributes such as the name of the driver, server and database, as well as security information such as user name and password.

#### *Example of MySQL Connection String*

```
Server=myServerAddress;Port=1234;Database=myDataBase;Uid=myUsername;  
Pwd=myPassword;
```

The port 3306 is the default MySql port. The value is ignored if Unix socket is used.

## 3.6 Methodology

The methodology used in project is Agile Software Development. Agile Software Development methodology is used especially for software development, that is characterized by the division of tasks into short phases of work and frequent reassessment and adaptation of plans.

It is for a project that needs extreme agility in requirements. The key features of agile are its short-termed delivery cycles (sprints), agile requirements, dynamic team culture, less restrictive project control and emphasis on real-time communication.

Agile software development is a group of software development methods based on iterative and incremental development, in which requirements and solutions evolve through collaboration between self-organizing, cross-functional teams. It promotes adaptive planning, evolutionary development and delivery, a time-boxed iterative approach, and encourages rapid and flexible response to change. It is a conceptual framework that promotes foreseen tight iterations throughout the development cycle.

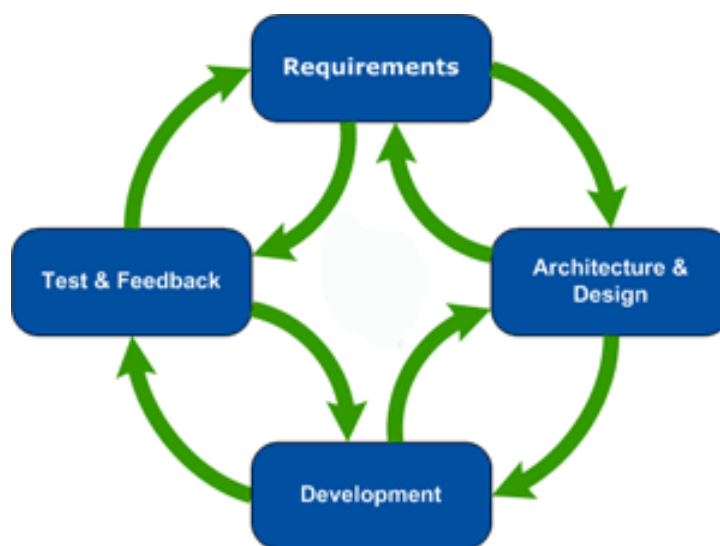


Figure 3.11: Agile Software Development Methodology

The Manifesto of Agile Software Development context are:

- *Individuals and interactions* In agile development, self-organization and motivation are important, as are interactions like co-location and pair programming.

- *Working software* Working software will be more useful and welcome than just presenting documents to clients in meetings.
- *Customer collaboration* Requirements cannot be fully collected at the beginning of the software development cycle, therefore continuous customer or stakeholder involvement is very important.
- *Responding to change* Agile development is focused on quick responses to change and continuous development.



# Chapter 4

## Implementation, Testing and Maintenance

### 4.1 Introduction to Languages, IDE's, Tools and Technologies Used For Implementation

#### 4.1.1 Java

Java is a very popular programming language developed by Sun Microsystems (now owned by Oracle). Developed long after C and C++, Java incorporates many of the powerful features of those powerful languages while addressing some of their drawbacks. Still, programming languages are only as powerful as their libraries. These libraries exist to help developers build applications.

Some of the Java's important core features are:

- Its easy to learn and understand.
- Its designed to be platform-independent and secure, using virtual machines.
- Its object-oriented.

Android relies heavily on these Java fundamentals. The Android SDK includes many standard Java libraries (data structure libraries, mathlibraries, graphics libraries, networking libraries and everything else you could want) as well as special Android libraries that will help you develop awesome Android applications.

#### ***Platform Independence Importance***

With many programming languages, you need to use a compiler to reduce your code down into machine language that the device can understand. While this is well and good, different devices use different machine languages. This means that you might need to compile your applications for each different device or machine language in other words, your code isn't very portable. This is not the case with Java. The Java

compilers convert your code from human readable Java source files to something called bytecode in the Java world. These are interpreted by a Java Virtual Machine, which operates much like a physical CPU might operate on machine code, to actually execute the compiled code. Although it might seem like this is inefficient, much effort has been put into making this process very fast and efficient. These efforts have paid off in that Java performance is generally second only to C/C++ in common language performance comparisons.

Android applications run in a special virtual machine called the Dalvik VM. While the details of this VM are unimportant to the average developer, it can be helpful to think of the Dalvik VM as a bubble in which your Android application runs, allowing you to not have to worry about whether the device is a Motorola Droid, an HTC Evo, or the latest toaster running Android. You don't care so long as the device is Dalvik VM friendly and that's the device manufacturer's job to implement, not yours.

### ***Why is Java Secure?***

Let's take this bubble idea a bit further. Because Java applications run within the bubble that is a virtual machine, they are isolated from the underlying device hardware. Therefore, a virtual machine can encapsulate, contain, and manage code execution in a safe manner compared to languages that operate in machine code directly. The Android platform takes things a step further. Each Android application runs on the (Linux-based) operating system using a different user account and in its own instance of the Dalvik VM. Android applications are closely monitored by the operating system and shut down if they don't play nice (e.g. use too much processing power, become unresponsive, waste resources, etc.).

Therefore, it's important to develop applications that are stable and responsive. Applications can communicate with one another using well-defined protocols.

## **4.1.2 Android Development Tools**

- ***Android SDK:***

The Android Software Development Kit (Android SDK) contains the necessary tools to create, compile and package Android applications. Most of these tools are command line based. The primary way to develop Android applications is based on the Java programming language.

- ***Android debug bridge (adb):***

The Android SDK contains the Android debug bridge (adb), which is a tool that allows you to connect to a virtual or real Android device, for the purpose of managing the device or debugging your application.

- ***Android Developer Tools and Android Studio:***

Google provides two integrated development environments (IDEs) to develop new applications. The Android Developer Tools (ADT) are based on the Eclipse IDE. ADT is a set of components (plugins), which extend the Eclipse IDE with Android development capabilities. Google also supports an IDE called Android Studio for creating Android applications. This IDE is based on the IntelliJ IDE.

Both IDEs contain all required functionality to create, compile, debug and deploy Android applications. They also allow the developer to create and start virtual Android devices for testing.

Both tools provide specialized editors for Android specific files. Most of Android's configuration files are based on XML. In this case these editors allow you to switch between the XML representation of the file and a structured user interface for entering the data.

**Dalvik Virtual Machine**  
The Android system uses a special virtual machine, i.e., the Dalvik Virtual Machine (Dalvik) to run Java based applications. Dalvik uses a custom bytecode format which is different from Java bytecode.

Therefore you cannot run Java class files on Android directly; they need to be converted into the Dalvik bytecode format.

- ***Android RunTime (ART):***

With Android 4.4, Google introduced the Android RunTime (ART) as optional runtime for Android 4.4. It is expected that versions after 4.4 will use ART as default runtime. ART uses Ahead Of Time compilation. During the deployment process of an application on an Android device, the application code is translated into machine code. This results in approx. 30% larger compile code, but allows faster execution from the beginning of the application.

### **4.1.3 Security and Permission Concept in Android**

- ***Security Concept In Android:***

The Android system installs every Android application with a unique user and group ID. Each application file is private to this generated user, e.g., other applications cannot access these files. In addition each Android application is started in its own process.

Therefore, by means of the underlying Linux kernel, every Android application is isolated from

other running applications. If data should be shared, the application must do this explicitly via an Android component which handles the sharing of the data, e.g., via a service or a content provider.

- ***Permission concept in Android:***

Android contains a permission system and predefines permissions for certain tasks. Every application can request required permissions and also define new permissions. For example, an application may declare that it requires access to the Internet.

Permissions have different levels. Some permissions are automatically granted by the Android system, some are automatically rejected. In most cases the requested permissions are presented to the user before installing the application. The user needs to decide if these permissions shall be given to the application.

If the user denies a required permission, the related application cannot be installed. The check of the permission is only performed during installation, permissions cannot be denied or granted after the installation.

An Android application declares the required permissions in its `AndroidManifest.xml` configuration file. It can also define additional permissions which it can use to restrict access to certain components.

## 4.2 Coding Standards of Language Used

Coding Standards of Java are used in the whole project. This standard include the following:

- No wildcard imports.
- Overloads appear sequentially.
- Braces are used even when the body is empty or contains a single statement.
- Two space indentation.
- Column limit can be 80 or 100 characters.
- No C-style array declarations.
- Default statements required in switch statements.
- Modifiers appear in the order recommended by the Java Language Specification.
- Constants use `CONSTANT_CASE`. Note that every constant is a static final field, but not all static final fields are constants.
- Class name should start with uppercase letter.
- Function name should start with lowercase letter.

## 4.3 Project Scheduling

Project scheduling is concerned with the techniques that can be employed to manage the activities that need to be undertaken during the development of a project. Scheduling is carried out in advance of the project commencing and involves:

- Identifying the tasks that need to be carried out.
- Estimating how long they will take.
- Allocating resources (mainly personnel).
- Scheduling when the tasks will occur.

Once the project is underway control needs to be exerted to ensure that the plan continues to represent the best prediction of what will occur in the future based on what occurs during the development and often necessitates revision of the plan. Effective project planning will help to ensure that the systems are delivered:

- Within cost;
- Within the time constraint;
- To a specific standard of quality.

Two project scheduling techniques will be presented, the Milestone Chart and Gantt Chart.

### 1. *Milestone Charts:*

Milestones mark significant events in the life of a project, usually critical activities which must be achieved on time to avoid delay in the project. Milestones should be truly significant and be reasonable in terms of deadlines (avoid using intermediate stages).

Examples include:

- Installation of equipment.
- Completion of phases.
- File conversion.
- Cut over to the new system

## 2. *Gantt Charts:*

A Gantt chart is a horizontal bar or line chart which will commonly include the following features:

- Activities identified on the left hand side.
- Time scale is drawn on the top (or bottom) of the chart.
- A horizontal open oblong or a line is drawn against each activity indicating estimated duration.
- Dependencies between activities are shown.
- At a review point the oblongs are shaded to represent the actual time spent (an alternative is to represent actual and estimated by 2 separate lines).
- A vertical cursor (such as a transparent ruler) placed at the review point makes it possible to establish activities which are behind or ahead of schedule.

Following is the chart of various commits on github, while developing this application:

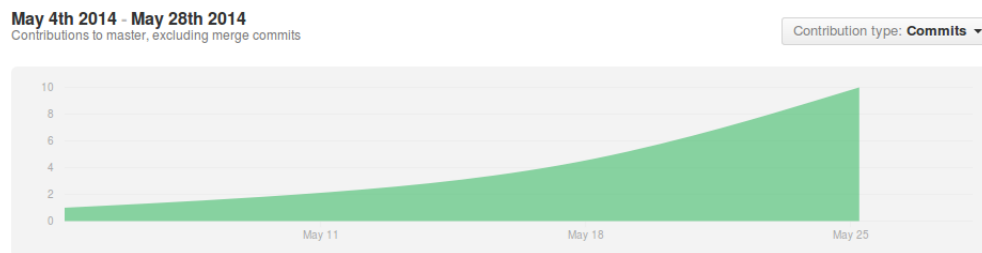


Figure 4.1: Graph of Commits on GitHub

This graph is showing commit in last week of May. It depicts how many times, changes are pushed on to github after testing.



Figure 4.2: Code Frequency Graph on GitHub

This graph depicts the frequency of code that is pushed on github on respective dates.



## 4.4 Test Plan and Test Activities

### 4.4.1 Test Plan

A test plan can be defined as a document describing the scope, approach, resources, and schedule of intended testing activities. It identifies test items, the features to be tested, the testing tasks, who will do each task, and any risks requiring contingency planning. In software testing, a test plan gives detailed testing information regarding an upcoming testing effort, including

- Scope of testing
- Schedule
- Test Deliverables
- Release Criteria
- Risks and Contingencies

It is also described as a detail of how the testing will proceed, who will do the testing, what will be tested, in how much time the test will take place, and to what quality level the test will be performed.

The process of defining a test project so that it can be properly measured and controlled. The test planning process generates a high level test plan document that identifies the software items to be tested, the degree of tester independence, the test environment, the test case design and test measurement techniques to be used, and the rationale for their choice.

A testing plan is a methodological and systematic approach to testing a system such as a machine or software. It can be effective in finding errors and flaws in a system. In order to find relevant results, the plan typically contains experiments with a range of operations and values, including an understanding of what the eventual workflow will be.

Test plan is a document which includes, introduction, assumptions, list of test cases, list of features to be tested, approach, deliverables, resources, risks and scheduling. A test plan is a systematic approach

to testing a system such as a machine or software. The plan typically contains a detailed understanding of what the eventual workflow will be. A record of the test planning process detailing the degree of tester independence, the test environment, the test case design techniques and test measurement techniques to be used, and the rationale for their choice.

#### 4.4.2 Test Activities

Various Testing Activities are as follow:

1. **Black box testing** Internal system design is not considered in this type of testing. Tests are based on requirements and functionality.
2. **White box testing** This testing is based on knowledge of the internal logic of an applications code. Also known as Glass box Testing. Internal software and code working should be known for this type of testing. Tests are based on coverage of code statements, branches, paths, conditions.
3. **Unit testing** Testing of individual software components or modules. Typically done by the programmer and not by testers, as it requires detailed knowledge of the internal program design and code. may require developing test driver modules or test harnesses.
4. **Incremental integration testing** Bottom up approach for testing i.e continuous testing of an application as new functionality is added; Application functionality and modules should be independent enough to test separately. done by programmers or by testers.
5. **Integration testing** Testing of integrated modules to verify combined functionality after integration. Modules are typically code modules, individual applications, client and server applications on a network, etc. This type of testing is especially relevant to client/server and distributed systems.
6. **Functional testing** This type of testing ignores the internal parts and focus on the output is as per requirement or not. Black-box type testing geared to functional requirements of an application.
7. **System testing** Entire system is tested as per the requirements. Black-box type testing that is based on overall requirements specifications, covers all combined parts of a system.
8. **End-to-end testing** Similar to system testing, involves testing of a complete application environment in a situation that mimics real-world use, such as interacting with a database, using network communications, or interacting with other hardware, applications, or systems if appropriate.

9. ***Acceptance testing*** - Normally this type of testing is done to verify if system meets the customer specified requirements. User or customer do this testing to determine whether to accept application.
10. ***Usability testing*** User-friendliness check. Application flow is tested, Can new user understand the application easily, Proper help documented whenever user stuck at any point. Basically system navigation is checked in this testing.

# Chapter 5

## Results and Discussions

### 5.1 User Interface Representation

In order to make user interface, many controls are used. Some of which are as follow:

1. ***TextView:***

Text View displays text to the user and optionally allows them to edit it. A TextView is a complete text editor, however the basic class is configured to not allow editing.

2. ***EditText:***

EditText is a thin veneer over TextView that configures itself to be editable.

3. ***Button:***

A button consists of text or an icon (or both text and an icon) that communicates what action occurs when the user touches it.

4. ***List View:***

ListView is a view group that displays a list of scrollable items. The list items are automatically inserted to the list using an Adapter that pulls content from a source such as an array or database query and converts each item result into a view that's placed into the list.

5. ***Check Boxes:***

Checkbox is a control or widget that can be activated with a finger touch and can be polled in the application's code for a checked and unchecked state.

#### 5.1.1 Brief Description of Various Modules

##### GCM Messaging

Google Cloud Messaging for Android (GCM) is a service that allows you to send data from your server to your users' Android-powered device, and also to receive messages from devices on the same connec-

tion. The GCM service handles all aspects of queueing of messages and delivery to the target Android application running on the target device. GCM is completely free no matter how big your messaging needs are, and there are no quotas.

### ***GCM Working***

1. Android device sends SENDER\_ID to GCM Server for registration.
2. After successful registration, GCM sends Registration ID to Android device.
3. After getting registration id, Android device sends registration id to Web Server.
4. Store registration id in our database at the server.
5. Whenever Push Notification needed, get registration ids from server, and send the request to GCM with registration id and message.
6. After push notification request, GCM sends Push Notifications to Android device.

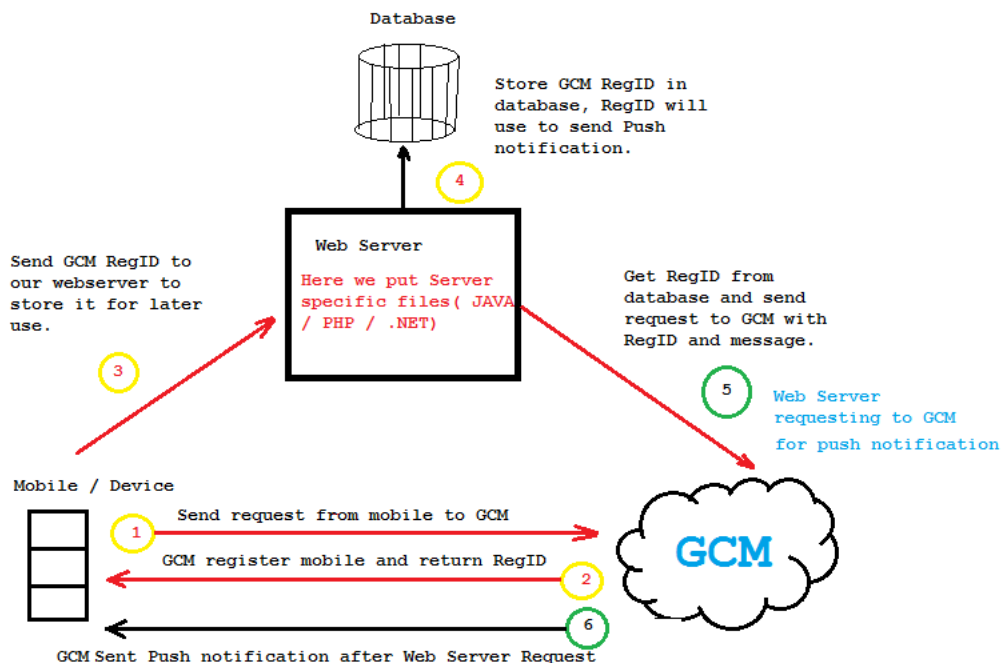


Figure 5.1: Google Cloud Messaging Working

Following are the steps to use GCM in android project:

1. Register with Google Cloud Messaging from GOOGLE API Console and get Sender ID and API key for Google Cloud Messaging.
2. Set Emulator for Google Cloud Messaging helper library.
3. Create Android project to register with Google Cloud Messaging(GCM).
4. Create server side code to save Google Cloud Messaging registration id in our database and send push notification to device.

### **HTTP Connections To Web Server**

Steps to have HTTP connection to Web Server are as follow:

1. Declare Internet permissions in the manifest by adding the following line to AndroidManifest.xml.
2. Create your HttpClient and HttpPost objects to execute the POST request.
3. Set your POST data. This is done by creating and setting a list of NameValuePairs as your HttpPost's entity. Be sure to catch the UnsupportedOperationException thrown by HttpPost.setEntity().
4. Execute the POST request. This returns an HttpResponse object, whose data can be extracted and parsed. Be sure to catch the ClientProtocolException and IOException thrown.

### **JSON Parsing**

JSON stands for JavaScript Object Notation. It is an independent data exchange format and is the best alternative for XML. Steps of JSON Parsing are:

1. For parsing a JSON object, we will create an object of class JSONObject and specify a string containing JSON data to it.
2. An JSON file consist of different object with different key/value pair e.t.c. So JSONObject has a seperate function for parsing each of the component of JSON file.

### **Broadcast Receiver**

Broadcast receiver is a component that responds to system conditions such as low battery or the screen being turned off. You can use broadcast receivers to initiate a response from a running application, such as if a picture has been taken.

## **Services**

Service is a process that runs in background to perform long term operations or work for remote processes. Services don't provide a User Interface.

## **Content Provider**

It manages persistent data on the device or external sources such as web or cloud or any other system application has access to. Android devices has on-board SQLite database management system to provide organized persistent data storage.

## 5.2 SnapShots of System

Following are the snapshots of working application:

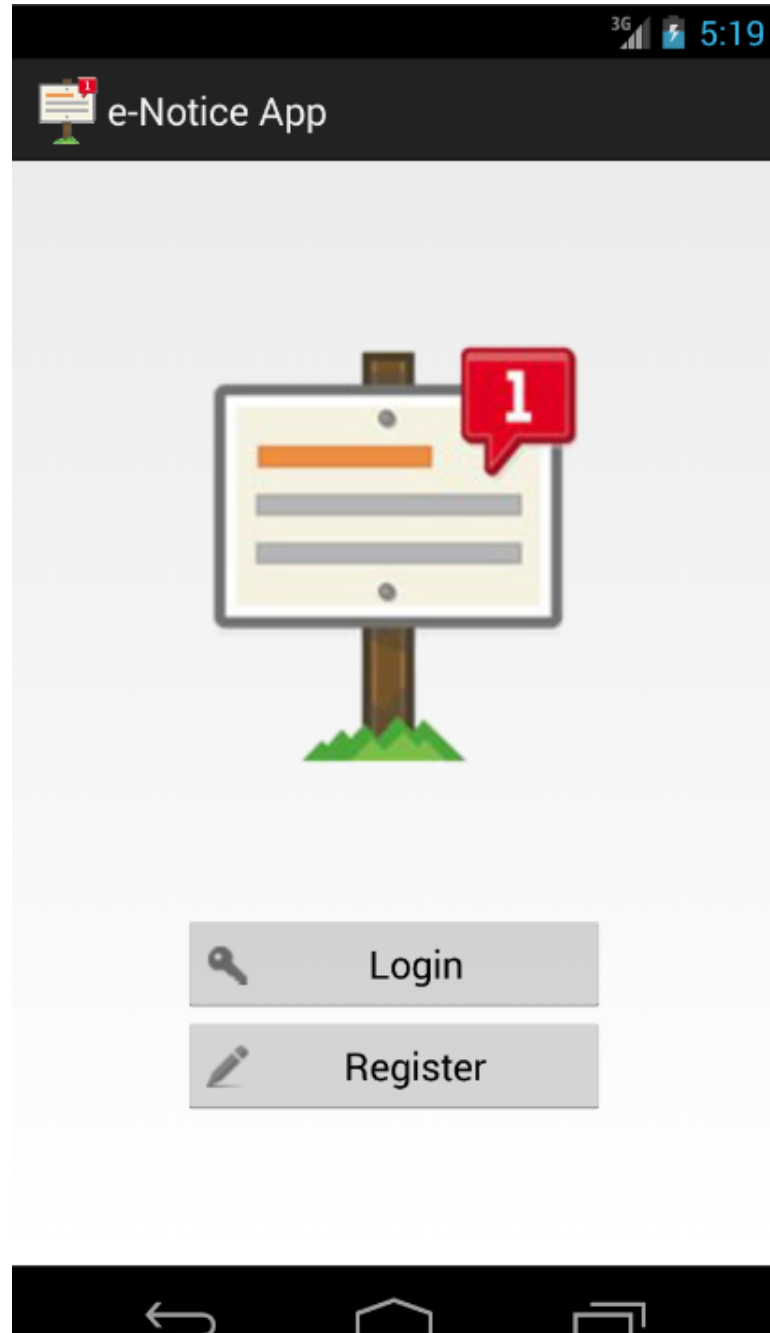
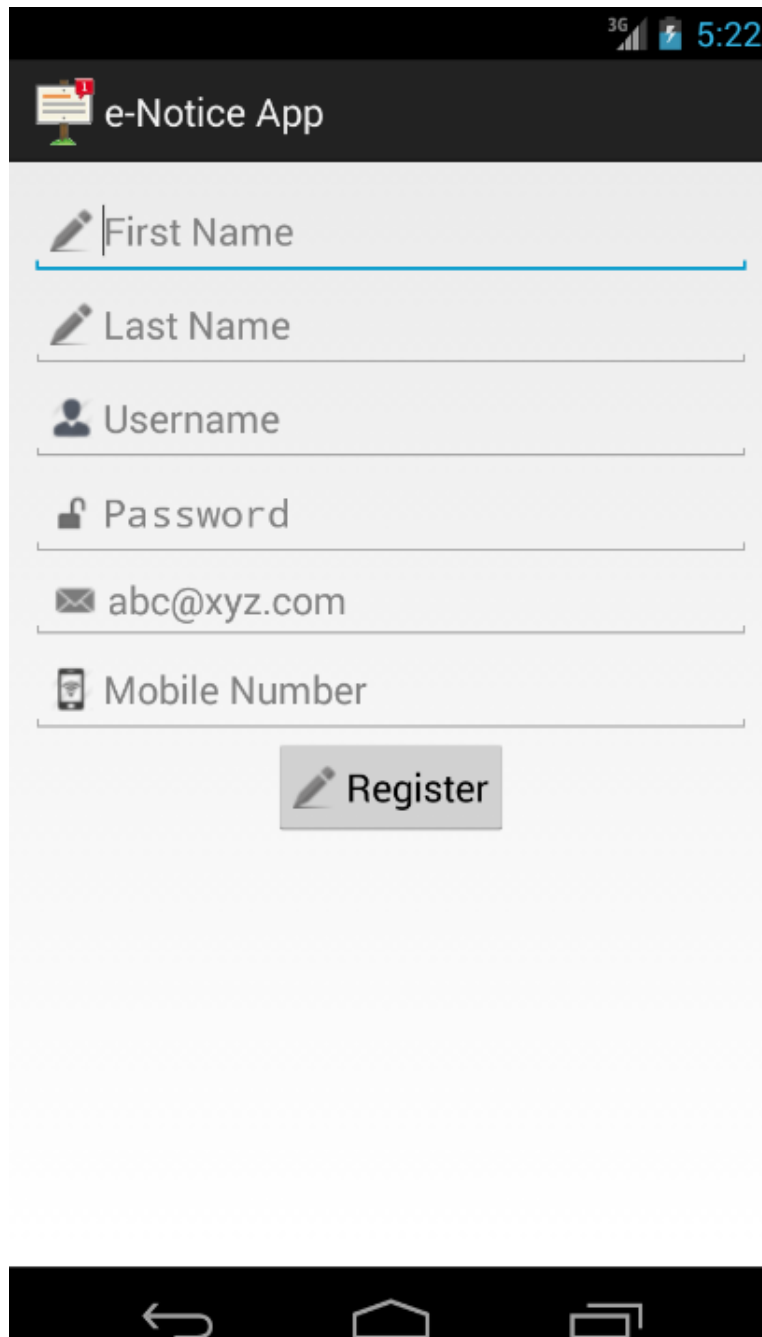


Figure 5.2: Landing Page

This is the first page, the user is presented with at the first time, when he opens the application.





The screenshot shows the 'e-Notice App' interface on a mobile device. The status bar at the top indicates 3G connectivity, battery level, and the time 5:22. The app's header is dark grey with a notification icon and the title 'e-Notice App'. The main content area is light grey and contains six input fields, each with an icon on the left: 'First Name' (pencil), 'Last Name' (pencil), 'Username' (person), 'Password' (lock), 'Email' (envelope) with the value 'abc@xyz.com', and 'Mobile Number' (phone). Below these fields is a grey 'Register' button with a pencil icon. The bottom of the screen features a black Android navigation bar with back, home, and recent apps icons.

Figure 5.3: Register Page

This is the page where user registers himself to GCM and Web Server.

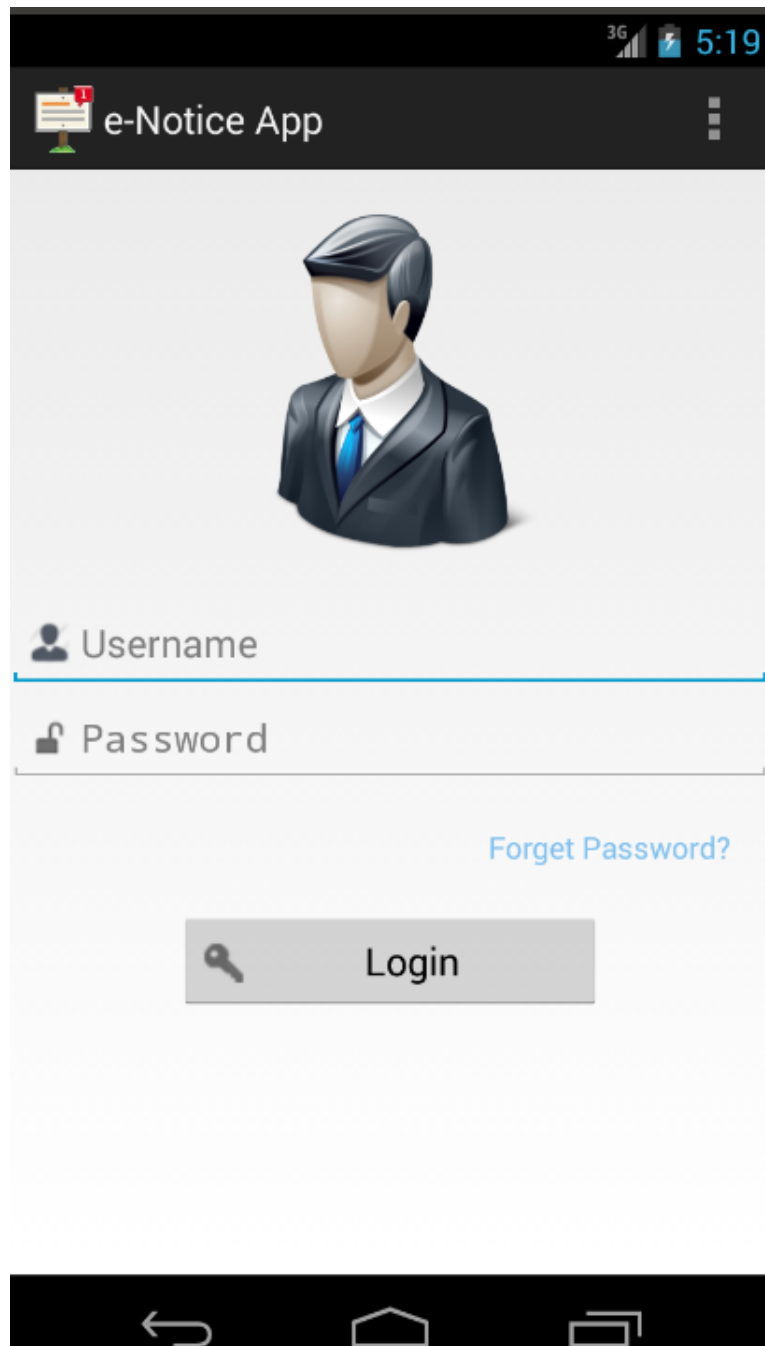


Figure 5.4: Login Page

This page is used to login to the application.

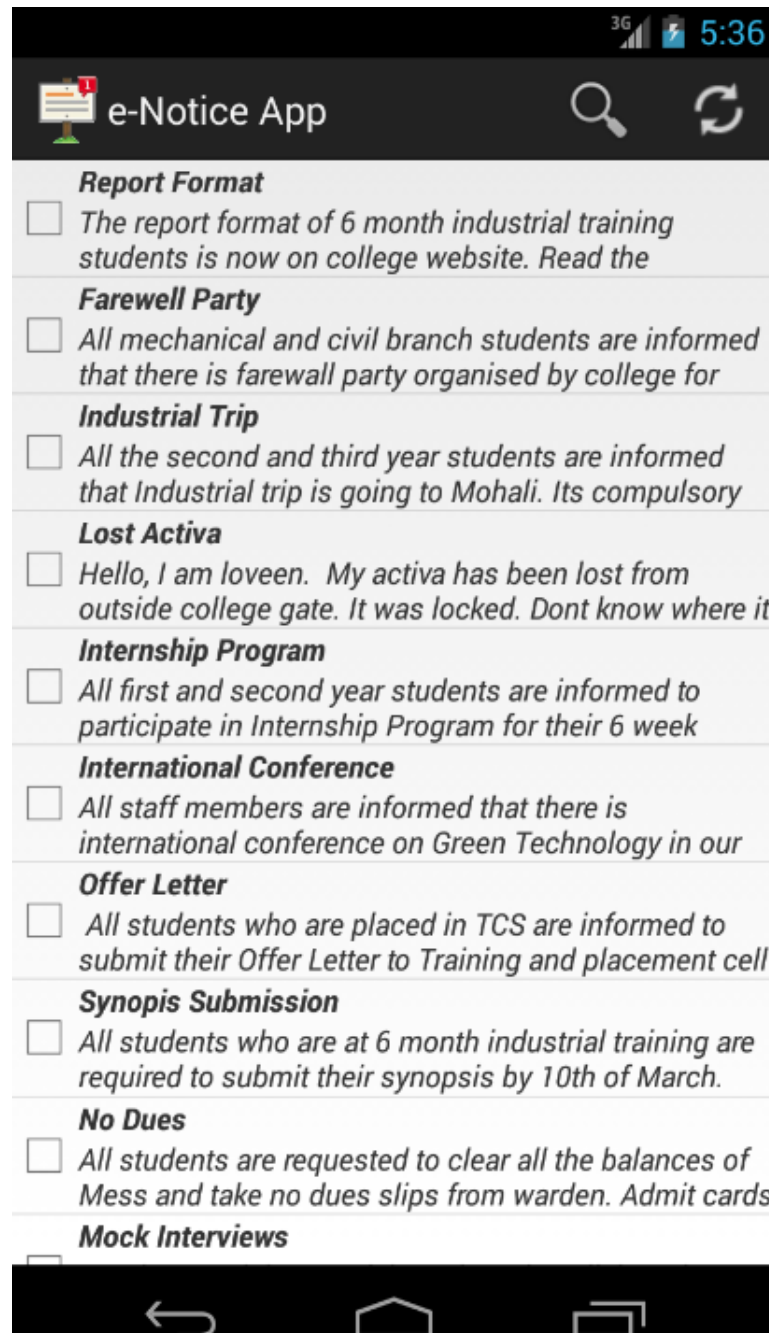


Figure 5.5: DashBoard Page

In this page, all the notices are displayed. This is the main panel that is required by the normal user of application. The user will tap on a particular notice in order to view details of notices.

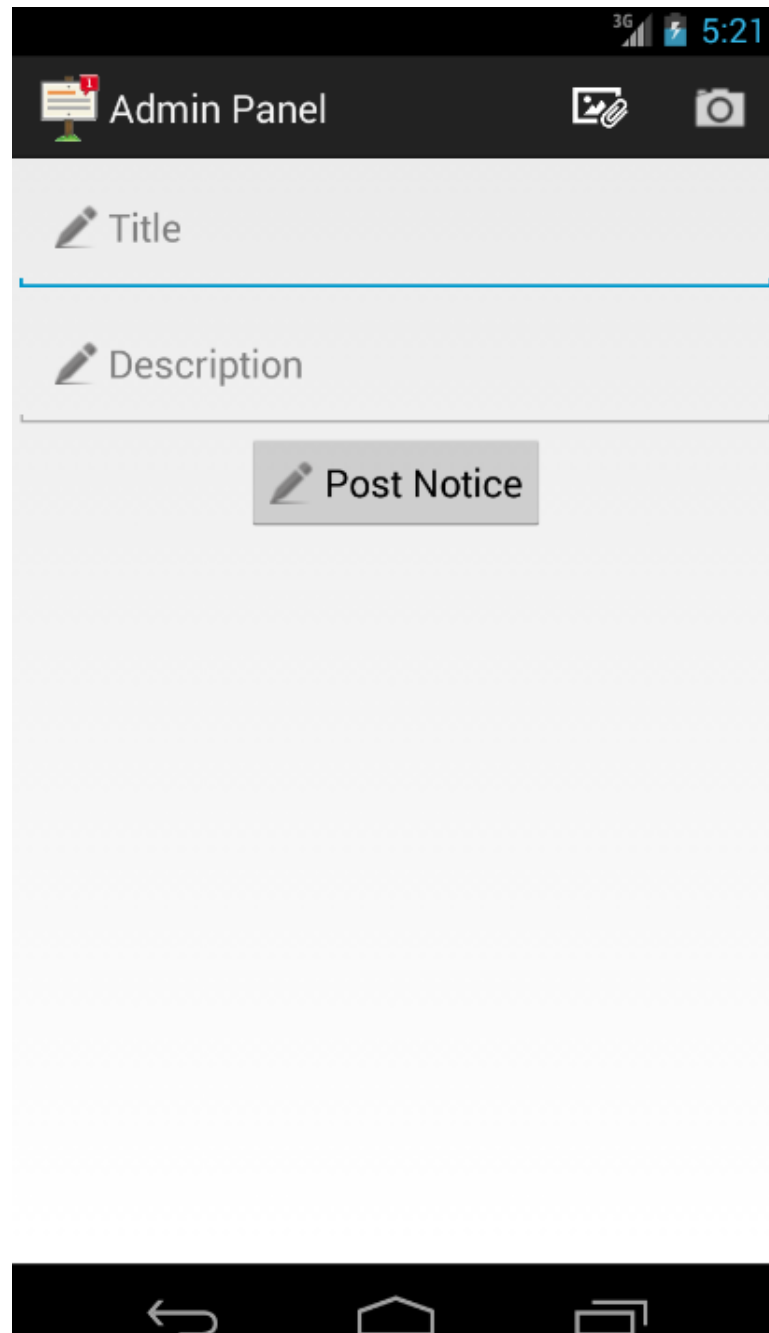


Figure 5.6: Post Notices Page

This is an Admin Panel Page where admin is allowed to post the message. Admin has two options of attaching image to notice. First is to pick up image from gallery and second is to take an image from camera instantly and post with the notice.

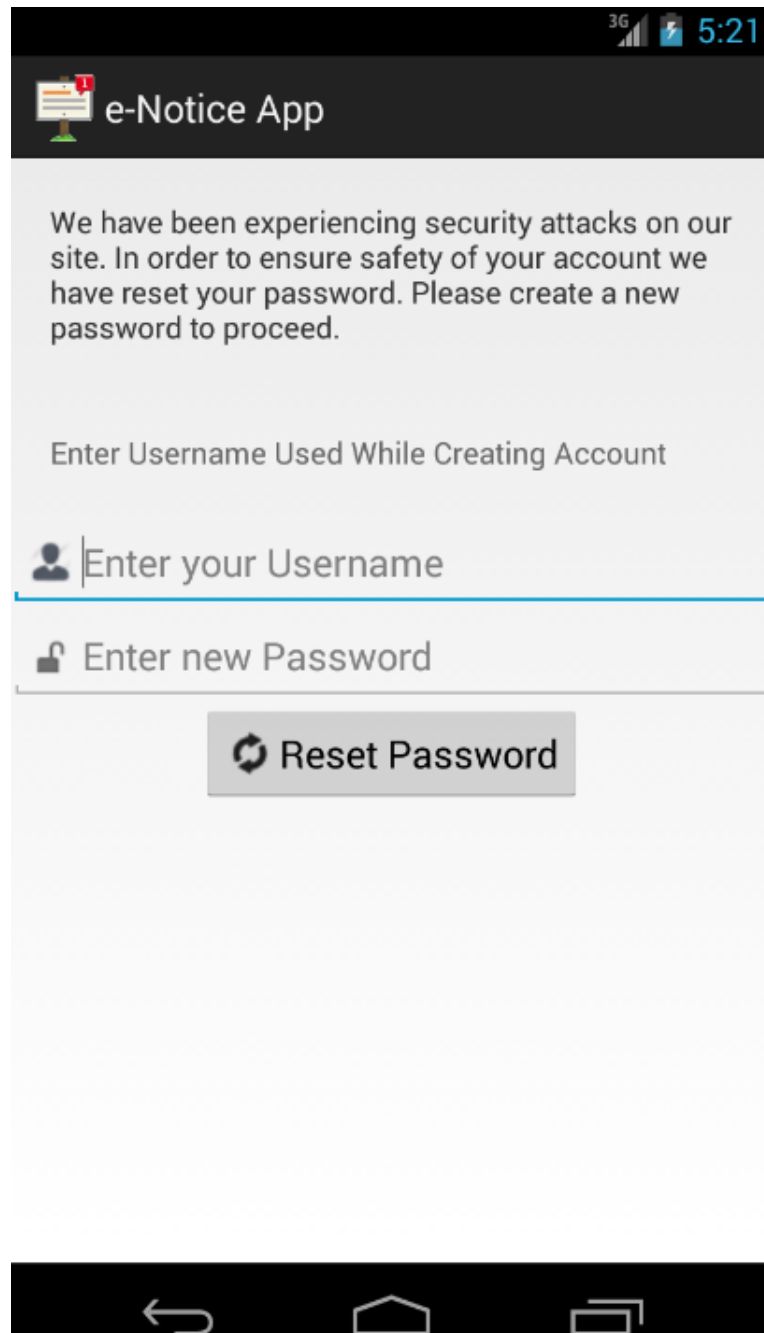


Figure 5.7: Reset Password Page

This page is presented to the user when user forgots his password and want to reset the password.

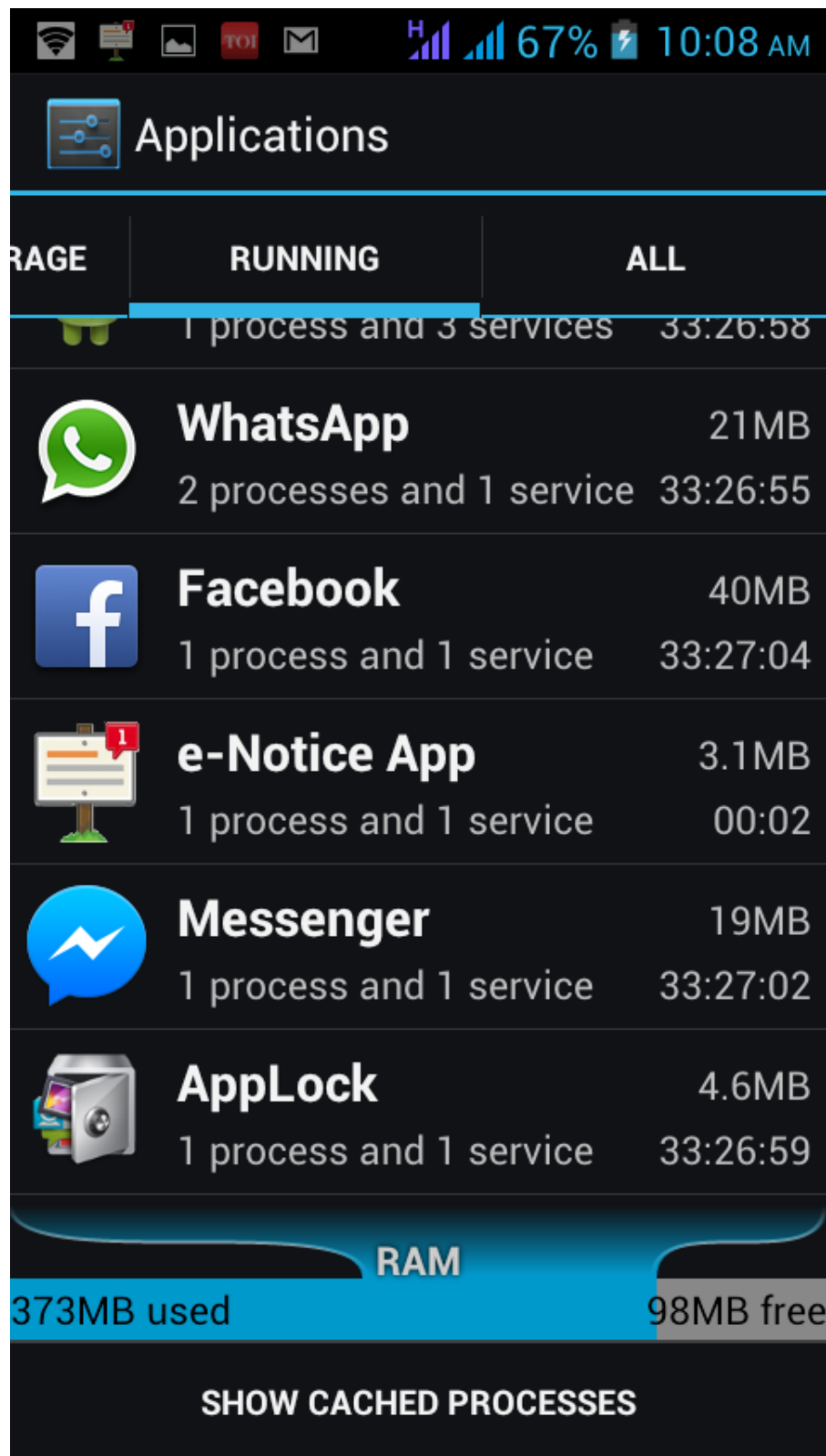


Figure 5.8: Intent Service Running In Application

This figure shows that service of application starts running, whenever any GCM notification arrives.

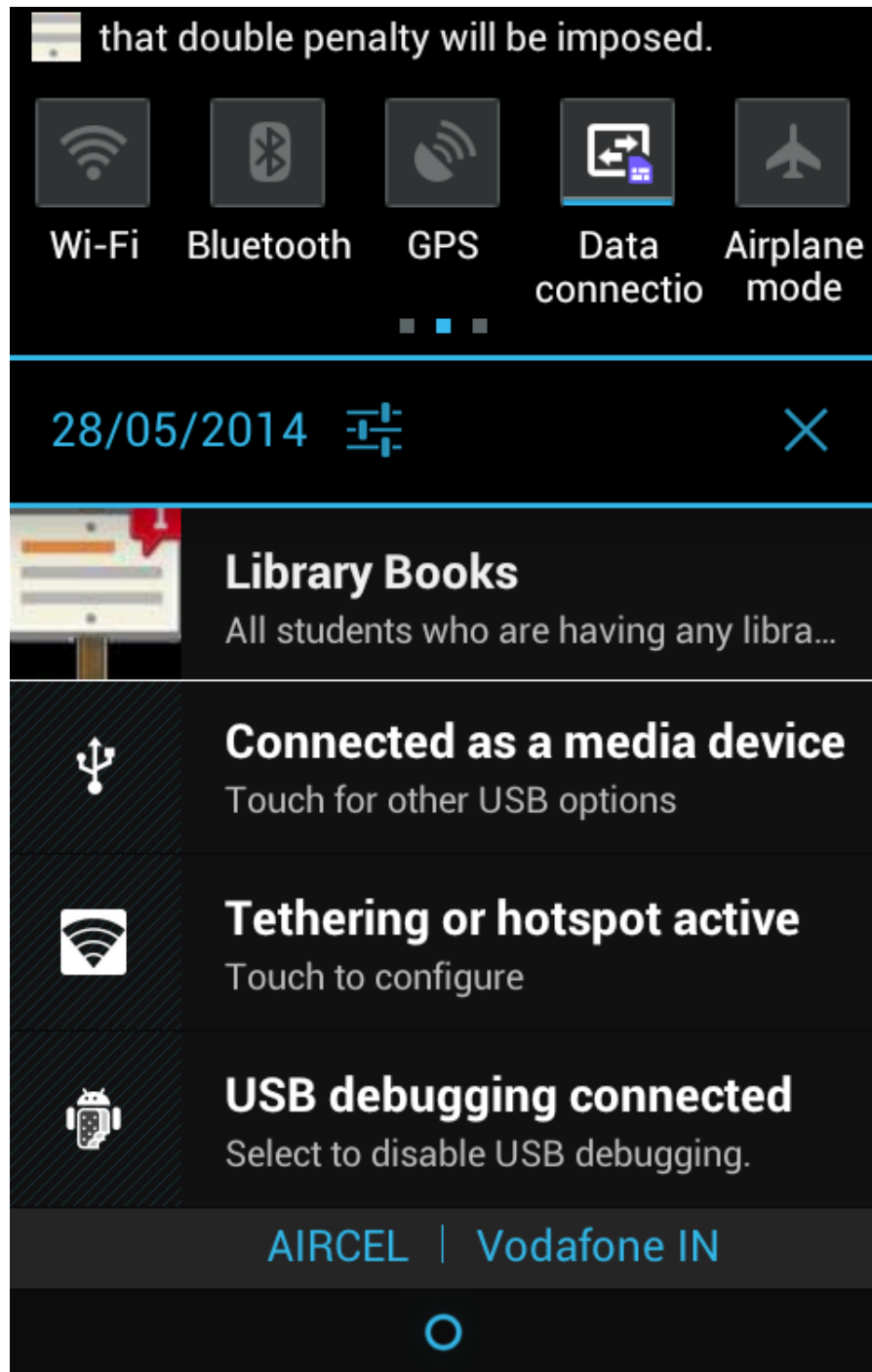


Figure 5.9: Notification Received

This figure shows the notification received by the user from GCM server, whenever any notice is posted by the admin.

The screenshot shows the 'e-Notice App' interface on a mobile device. The status bar at the top indicates 3G connectivity, battery level, and the time 6:06. The app header is dark grey with a notification icon and the title 'e-Notice App'. Below the header, there are six input fields for registration details: a name field with 'Priyanka', a last name field with 'Kapoor', a user ID field with 'priyanka', a password field with masked dots, an email field with 'abc-jkjjk@gmail..com', and a phone number field with '9890999089'. The email field is highlighted with a red border, and a grey error message box below it states 'Please enter a valid Email address.' A 'Register' button is positioned below the input fields. The bottom of the screen features a standard Android navigation bar with back, home, and recent apps icons.

3G 6:06

e-Notice App

Priyanka

Kapoor

priyanka

.....

abc-jkjjk@gmail..com

9890999089

Register

Please enter a valid Email address.

Figure 5.10: Email Validation

This figure shows the email validations implemented inside the application.



3G 6:07

e-Notice App

Priyanka

Kapoor

priyanka

.....

priyanka@gmail.com

989099908989899

Register

Please enter a valid Mobile Number.

Figure 5.11: Mobile Number Validation

This figure shows the mobile number validation implemented in the application.

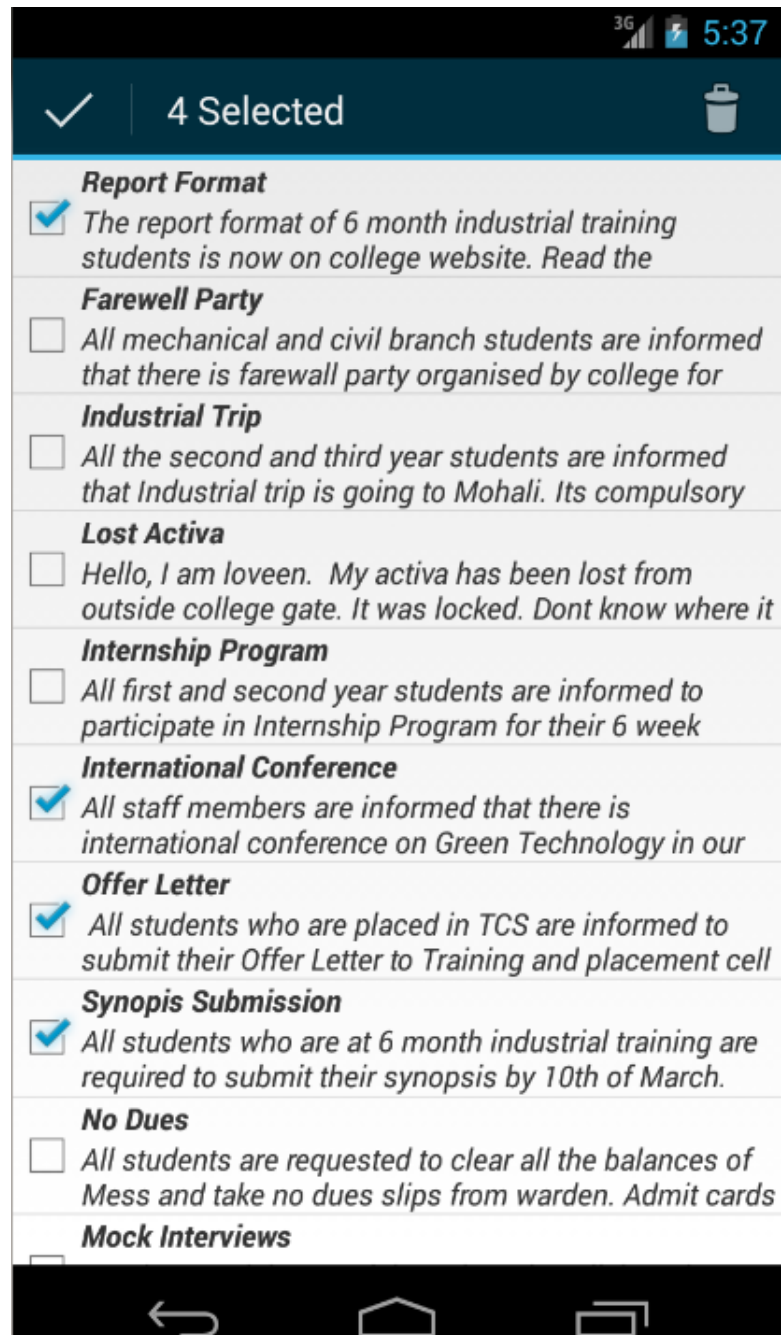


Figure 5.12: Contextual Action Bar

Contextual Action bar is implemented with the help of which, user is able to delete the notice.

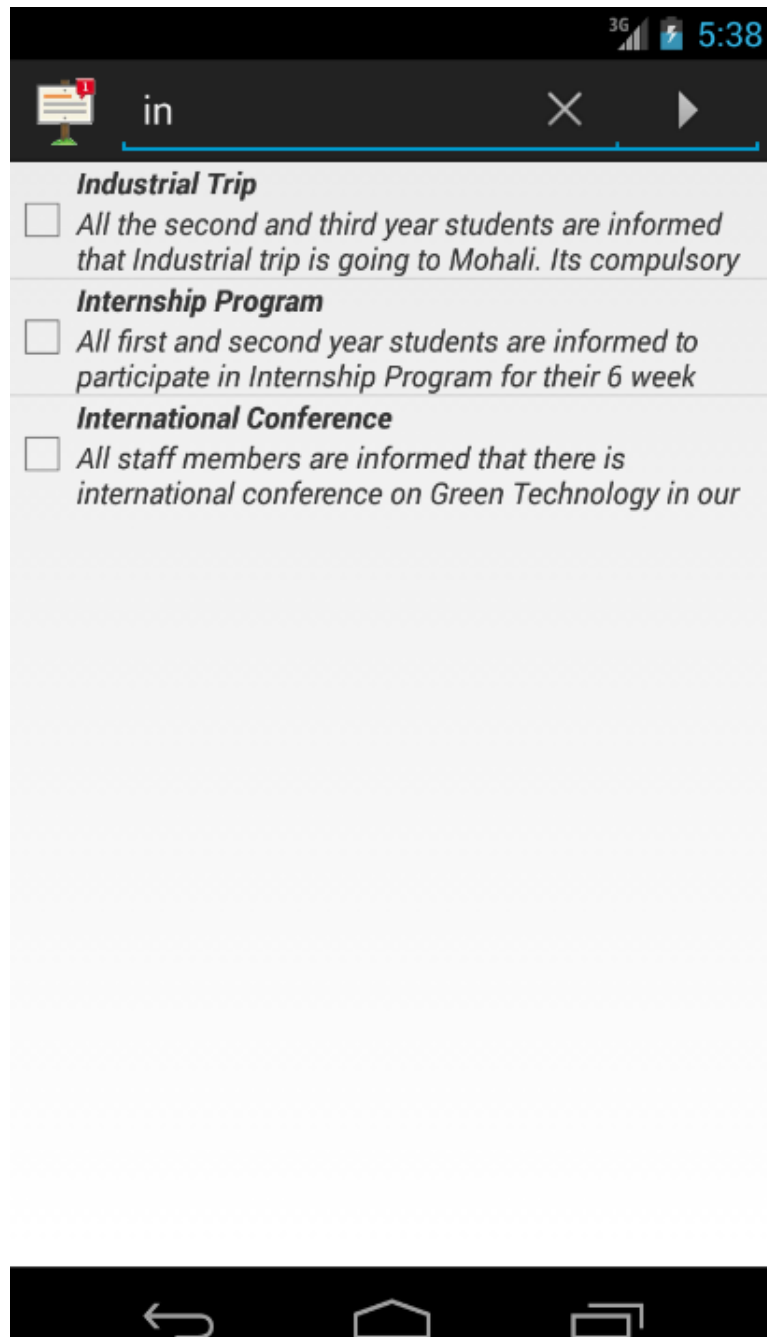


Figure 5.13: Search Box

This figure shows the search box available for the ease of user to search any notice by its title.

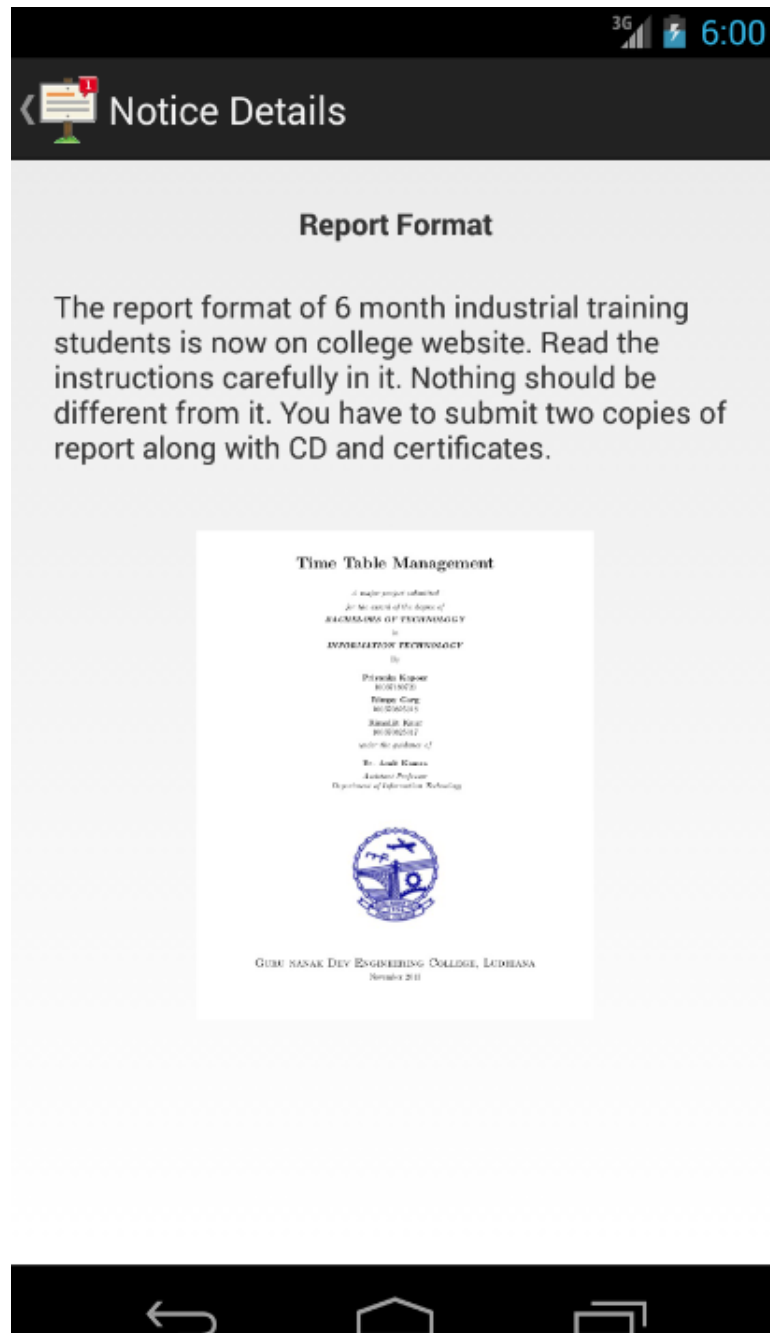


Figure 5.14: Notice Details

This figure is showing the details of a particular notice that is tapped by the user.

## 5.3 Back End Representations

The database used at the Back-end is MySQL database. Web Server used is Apache. Server side scripting language used is PHP.

Database design used at backend server has the following tables:

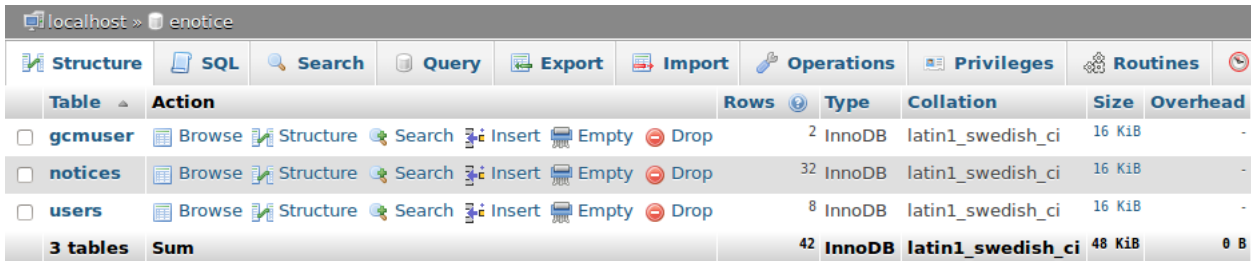


Table	Action	Rows	Type	Collation	Size	Overhead
<input type="checkbox"/> gcmuser	Browse Structure Search Insert Empty Drop	2	InnoDB	latin1_swedish_ci	16 KiB	-
<input type="checkbox"/> notices	Browse Structure Search Insert Empty Drop	32	InnoDB	latin1_swedish_ci	16 KiB	-
<input type="checkbox"/> users	Browse Structure Search Insert Empty Drop	8	InnoDB	latin1_swedish_ci	16 KiB	-
<b>3 tables</b>	<b>Sum</b>	<b>42</b>	<b>InnoDB</b>	<b>latin1_swedish_ci</b>	<b>48 KiB</b>	<b>0 B</b>

Figure 5.15: Tables inside Database

The first table is "gcmuser" which contains registration id given by GCM along with all personal information of the user. The second table is "user" which just contains the id, username and password of the user. The third table is "notices". It contains notice id, title and description of the notice, date on which the notice is posted and filepath if a notice has an attachment.

### 5.3.1 Snapshots of Database Tables

#	Name	Type	Collation	Attributes	Null	Default	Extra
<input type="checkbox"/> 1	<b>id</b>	int(11)			No	None	AUTO_INCREMENT
<input type="checkbox"/> 2	<b>title</b>	text	latin1_swedish_ci		Yes	NULL	
<input type="checkbox"/> 3	<b>description</b>	text	latin1_swedish_ci		Yes	NULL	
<input type="checkbox"/> 4	<b>date</b>	timestamp			No	CURRENT_TIMESTAMP	
<input type="checkbox"/> 5	<b>filepath</b>	text	latin1_swedish_ci		Yes	NULL	

Figure 5.16: Schema of GCM User Table

This table contains registration id given by GCM along with all personal information of the user like first name, last name, email id and phone number.

#	Name	Type	Collation	Attributes	Null	Default	Extra
<input type="checkbox"/> 1	<b>id</b>	int(11)			No	None	AUTO_INCREMENT
<input type="checkbox"/> 2	<b>regid</b>	text	latin1_swedish_ci		Yes	NULL	
<input type="checkbox"/> 3	<b>first</b>	varchar(25)	latin1_swedish_ci		Yes	NULL	
<input type="checkbox"/> 4	<b>last</b>	varchar(25)	latin1_swedish_ci		Yes	NULL	
<input type="checkbox"/> 5	<b>user</b>	varchar(25)	latin1_swedish_ci		Yes	NULL	
<input type="checkbox"/> 6	<b>pass</b>	varchar(50)	latin1_swedish_ci		Yes	NULL	
<input type="checkbox"/> 7	<b>email</b>	varchar(50)	latin1_swedish_ci		Yes	NULL	
<input type="checkbox"/> 8	<b>mobile</b>	text	latin1_swedish_ci		Yes	NULL	

Figure 5.17: Schema of User Table

This table just contains the id, username and password of the user.

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	<b>id</b>	int(11)			No	None	AUTO_INCREMENT
2	<b>username</b>	varchar(50)	latin1_swedish_ci		Yes	NULL	
3	<b>password</b>	varchar(50)	latin1_swedish_ci		Yes	NULL	

Figure 5.18: Schema of Notices Table

This table contain all the details about the notices like notice id, title, description, date of notice post and filepath if the notice has an attachment.

# Chapter 6

## Conclusion and Future Scope

### 6.1 Future Scope

The future scope of the project is that it can be used as any news giving application or it can be used to advertise your products, telling the customers about new schemes and products coming to your shop. This application of e-Notice can be further extended to include the following features:

1. ***Categorization of Notice:***

Notices can be categorized in different categories, so that its possible for user to easily manage the notices. Categorisation can also be done by making groups. Defining the notice to be circulated in a particular group can make it more secure.

2. ***Documents and PDF files:***

The attachments can be further improved to include PDF files or Doc files. Then there will not be much need to send images with the notices. A single file would serve all the purposes.

3. ***Feedback:***

Feedback on the notices can also be taken. It can increase communication among connected members and any issue can be easily sorted out on the spot.

## 6.2 Conclusion

I learned a lot by doing this project.

- Operating system: Ubuntu
- Languages used: Java, Android UI, PHP for backend
- Servers Used: LAMP Server, GCM Server
- Database: MySQL, SQLite
- Typesetting by:  $\text{\LaTeX}$

So during this project I learned all the above things. Before this project, I had no idea about Java and Android for making application. Although I had little bit knowledge of Ubuntu before. But now I learned a lot about Ubuntu and got knowledge of using Android and Java for developing mobile application and PHP for server side scripting. Now I prefer to work on command line rather than graphically. I learned how to work on shell script.

If I talk about the project, e-Notice Application has reduced lot of manual work. It has made notifying each and every user very easy and taht too with no time and place restrictions.



# References

- [1] Basic Android Tutorials, available at <http://www.tutorialspoint.com/android>
- [2] Android Development Tutorials, available at <http://www.developer.android.com>
- [3] Android Book. Wei-Meng Lee, *Android 4 Application Development*, John Wiley & Sons, Inc., 2012
- [4] L<sup>A</sup>T<sub>E</sub>XBook. Donald E. Knuth, *The TeXbook*, AddisonWesley, Boston, 1986