

CHAPTER-1

INTRODUCTION

As the digital world is growing exponentially, the need is to be able to adapt to the technology. The current educational system is a one way communication system, as the faculty will not know that whether students have understood the concepts taught.

The main aim of Android based Course Delivery Evaluation System is to improve the standard of education by continuous monitoring the students and analysing the appraisal provided by the students. The system is an Android based application system, that utilizes the current technology and helps the institution and the faculty to know the level of understanding of the students and to improve or retain the current teaching pattern or to improve it for their betterment. Here is the introduction of all these things.

1.1 Android

Android is an open source and Linux-based operating system for mobile devices such as smartphones and tablet computers. Android was developed by the Open Handset Alliance, led by Google, and other companies. Android offers a unified approach to application development for mobile devices which means developers need only develop for Android, and their applications should be able to run on different devices powered by Android. The source code for Android is available under free and open source software licenses. Google publishes most of the code under the Apache License version 2.0 and the rest, Linux kernel changes, under the GNU General Public License version 2.

1.1.1 Why Android

Android is popular with technology companies that require a ready-made, low-cost and customizable operating system for high-tech devices. Its open nature has encouraged a large community of developers and enthusiasts to use the open-source code as a foundation for

community-driven projects, which add new features for advanced users or bring Android to devices originally shipped with other operating systems. Few features of Android is shown in figure 1.1.

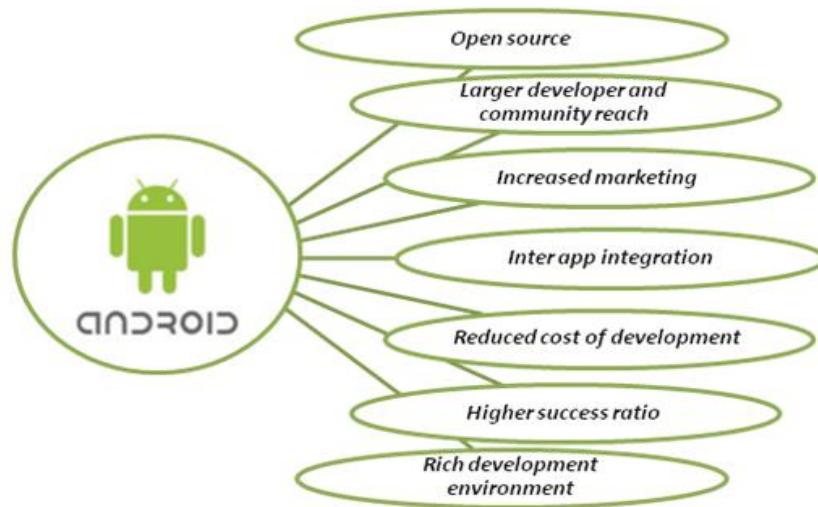


Figure 1.1: Android Features

1.1.2 Android Architecture

Android operating system is a stack of software components which is roughly divided into five sections and four main layers as shown below in the architecture diagram.

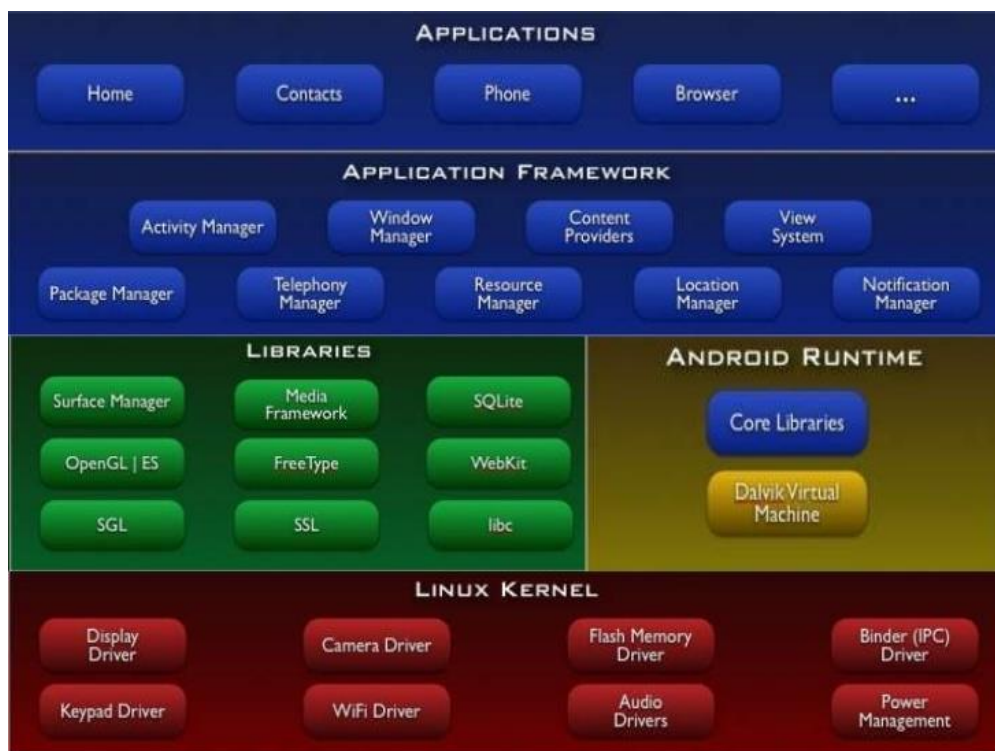


Figure 1.2: Android Architecture

The different types of layers in Android Architecture are:

- **Linux Kernel**

The basic layer is the Linux Kernel. The whole Android OS is built on top of the Linux Kernel with some further architectural changes. The term Kernel means the core of any Operating System. By saying Android is based upon Linux Kernel, it doesn't mean that it is another Linux distribution. It simply means Android at its core is Linux. It is a totally different OS. It is this Linux kernel that interacts with the hardware and it contains all the essential hardware drivers. Drivers are programs that control and communicate with the hardware. For example, consider the Bluetooth function. All devices have Bluetooth hardware in it. Therefore the kernel must include a Bluetooth driver to communicate with the Bluetooth hardware. The Linux kernel also acts as an abstraction layer between the hardware and other software layers. As Android is built on a most popular and proven foundation, porting of Android to variety of hardware became a relatively painless task.

- **Libraries**

The next layer is the Android's native libraries. It is this layer that enables the device to handle different types of data. These libraries are written in C or C++ language and are specific for a particular hardware. Some of the important native libraries include the following:

- **Surface Manager:** It is used for compositing window manager with off-screen buffering. Off-screen buffering means the apps can't directly draw into the screen; instead the drawings go to the off-screen buffer. There it is combined with other drawings and form the final screen the user will see. This off screen buffer is the reason behind the transparency of windows.
- **Media framework:** Media framework provides different media codecs allowing the recording and playback of different media formats.
- **SQLite:** SQLite is the database engine used in android for data storage purposes.
- **Web Kit:** It is the browser engine used to display HTML content.
- **OpenGL:** Used to render 2D or 3D graphics content to the screen.

- **Android Runtime**

Android Runtime consists of Dalvik Virtual machine and Core Java libraries.

- **Dalvik Virtual Machine:** It is a type of JVM used in android devices to run apps and is optimized for low processing power and low memory environments. Unlike the JVM, the

Dalvik Virtual Machine doesn't run .class files, instead it runs .dex files. .dex files are built from .class file at the time of compilation and provide higher efficiency in low resource environments. The Dalvik VM allows multiple instance of Virtual machine to be created simultaneously providing security, isolation, memory management and threading support.

- **ART:** Google has introduced a new virtual machine known as ART (Android Runtime) in their newer releases of Android. In Lollipop, the Dalvik Virtual Machine is completely replaced by ART. ART has many advantages over Dalvik VM such as AOT (Ahead of Time) compilation and improved garbage collection which boost the performance of apps significantly.
- **Core Java Libraries:** These are different from Java SE and Java ME libraries. However these libraries provide most of the functionalities defined in the Java SE libraries.

- **Application Framework**

These are the blocks that our application directly interacts with. These programs manage the basic functions of phone like resource management, voice call management etc. As a developer, you just consider these are some basic tools with which we are building our applications. Important blocks of Application framework are:

- **Activity Manager:** Manages the activity life cycle of applications
- **Content Providers:** Manage the data sharing between applications
- **Telephony Manager:** Manages all voice calls. We use telephony manager if we want to access voice calls in our application.
- **Location Manager:** Location management, using GPS or cell tower
- **Resource Manager:** Manage the various types of resources we use in our Application

- **Applications**

Applications are the top layer in the Android architecture and this is where our applications are going to fit into. Several standard applications come pre-installed with every device, such as:

- SMS client app
- Dialler
- Web browser
- Contact manager

1.2 Google Cloud Messaging

Google Cloud Messaging (GCM) is a free service that enables developers to send messages between servers and client apps. This includes downstream messages from servers to client apps, and upstream messages from client apps to servers.

For example, a lightweight downstream message could inform a client app that there is new data to be fetched from the server, as in the case of a "new email" notification. For use cases such as instant messaging, a GCM message can transfer up to 4kb of payload to the client app. The GCM service handles all aspects of queuing of messages and delivery to and from the target client app.

1.2.1 GCM Architecture

A GCM implementation includes a Google connection server, an app server in your environment that interacts with the connection server via HTTP or XMPP protocol, and a client app.

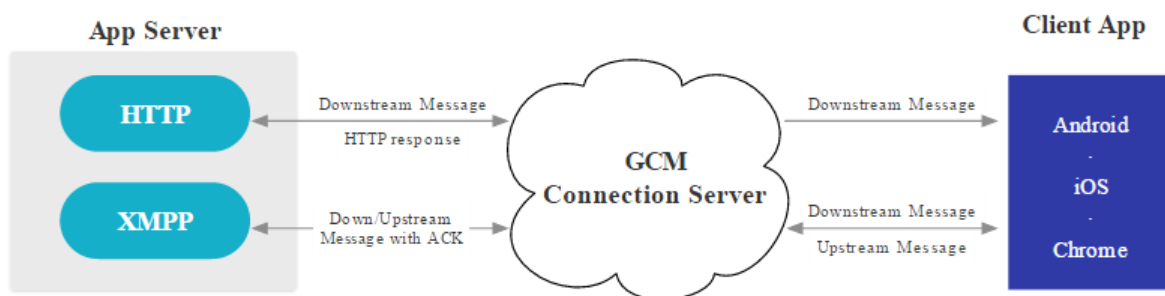


Figure 1.3: GCM Architecture

Here's how these components interact:

- Google GCM Connection Servers accept downstream messages from your app server and send them to a client app. The XMPP connection server can also accept messages sent upstream from the client app and forward them to your app server. For more information, see [About GCM Connection Server](#).
- On your App Server, you implement the HTTP and/or XMPP protocol to communicate with the GCM connection server(s). App servers send downstream messages to a GCM connection

server; the connection server enquirers and stores the message, and then sends it to the client app. If you implement XMPP, your app server can receive messages sent from the client app.

- The Client App is a GCM-enabled client app. To receive and send GCM messages, this app must register with GCM and get a unique identifier called a registration token. For more information on how to implement the client app, see the documentation for your platform.

1.2.2 GCM Lifecycle

GCM lifecycle has 3 main steps which include registration, send and receive downstream and upstream messages. The steps are:

- **Register to enable GCM:** An instance of a client app registers to receive messages.
- **Send and receive downstream messages:**
 - Send a message. The app server sends messages to the client app:
 - The app server sends a message to GCM connection servers.
 - The GCM connection server enquirers and stores the message if the device is offline.
 - When the device is online, the GCM connection server sends the message to the device.
 - On the device, the client app receives the message according to the platform-specific implementation. See your platform-specific documentation for details.
 - Receive a message. A client app receives a message from a GCM connection server. See your platform-specific documentation for details on how a client app in that environment processes the messages it receives.
- **Send and receive upstream messages.** This feature is only available if you're using the XMPP connection server.
 - Send a message. A client app sends messages to the app server:
 - On the device, the client app sends messages to the XMPP connection server. See your platform-specific documentation for details on how a client app can send a message via XMPP.
 - The XMPP connection server enquirers and stores the message if the server is disconnected.
 - When the app server is re-connected, the XMPP connection server sends the message to the app server.

- Receive a message. An app server receives a message from the XMPP connection server and then does the following:
 - Parses the message header to verify client app sender information.
 - Sends "ACK" to the XMPP connection server to acknowledge receiving the message.
 - Optionally parses the message payload, as defined by the client app.

1.3 Organization of Report

The technical aspect, system requirements and organization of project report are discussed as follows. The project report mainly consists of eight chapters.

CHAPTER 1: INTRODUCTION

This chapter includes the overall information about the project. The background knowledge needed for the project such as brief description on Android and its Architecture and Google Cloud Messaging and its Life cycle

CHAPTER 2: LITERATURE SURVEY

This chapter mainly discusses about the papers that are referred for implementing the project. All papers provide information about taking feedback and recording attendance based on NFC, Biometrics, and online feedback.

CHAPTER 3: PROBLEM DEFINITION

It includes purpose, motivation and objective of the project and the detailed study of background related to the project, objective, existing solution and proposed solution.

CHAPTER 4: SYSTEM REQUIREMENTS AND SPECIFICATION

Here we look into different kinds of requirements needed to successfully complete the project. It also states the functional and qualitative requirements.

CHAPTER 5: SYSTEM DESIGN

This chapter includes some basic design concepts, methodology, and some necessary diagram to outline the project easily.

CHAPTER 6: IMPLEMENTATION AND RESULTS

This describes the implementation of different phases and snapshots depicting different pages. Introduces about the technology and methodology used to implement the project.

CHAPTER 7: TESTING

This chapter defines various testing methods used for testing the different phases in the project; they are unit testing, integration testing and system testing mainly.

CHAPTER 8: CONCLUSION AND FUTURE ENHANCEMENT

This chapter gives the summary of the work carried out and provides the conclusion and the future enhancement of the project.

CHAPTER-2

LITERATURE SURVEY

2.1 Improve Student Learning Using Online Formative Assessment System

This paper aims to explain the key factors of an online assessment system that can effectively improve student learning. Some of the key literatures with regard to the style of online assessment and feedback were reviewed. The goal of the study is to conduct a survey questionnaire in order to collect data for analysis to address the research question. The research results show that the key factors of an online assessment system that can effectively improve students learning are providing instant and detailed feedback, using formative style assessment, and providing a communication tool between students and teachers.

The process of collecting the data used a specifically designed questionnaire in accordance with the research objectives. Firstly, 40 respondents were chosen randomly from students who are studying at WIT and IT Carlow, 20 from WIT and 20 from IT Carlow. The reason for selecting the respondents from the local county is to save cost and time and also for the ease of collecting data. The questionnaires were given to various respondents who were expected to fill the questionnaire and returned back to the researcher.

The questionnaire requires each of the respondents to grade a perceived/expected requirement on 1-5 Likert scale. A Likert scale consists of several declarative items that express a view point on a topic. Respondents were asked to indicate the degree to which they agree or disagree with the opinion expressed by the statement [23]. The statements are related to students' level of agreement on instant and detailed feedback, preference on continue assessment and final exam, and level of importance on different functions in an online assessment system.

2.2 Student Satisfaction as a Quality Management Technique in Higher Education

The importance of students' feedback within universities is gradually increasing, against the dynamic background of the demand and offer of higher education institutions. The present paper is focused upon the outcomes obtained by the implementation of a student satisfaction survey as a quality management technique in a Romanian university. The survey instrument followed three dimensions (teaching-learning activities, material base, facilities and services) and it was filled in by 997 students. The students' responses show that the most important aspects are the teaching and learning activities, followed by material base, facilities and services. For all the aspects evaluated, the students rate higher the importance they render to each of the dimensions than their satisfaction level with these dimensions.

Although the scientific literature often points out the weaknesses of student opinion surveys, such mechanisms prove to be extremely useful for educational institutions to gather feedback from its customers.

2.3 TouchIn: An NFC Supported Attendance System in a University Environment

This work proposes a web based attendance system using Near Field Communication (NFC) technology. The NFC technology is now integrated into mobile devices which can be used for online payment, access control, user identification, transfer of personal and private information, etc.

NFC is a new, short range, high frequency, low bandwidth, and wireless communication technology. NFC communication is activated by touching two NFC enabled devices together, or bringing them into close range. The range is usually few centimetres, and it operates at the frequency of 13.56 MHz. The maximum data transfer rate is 424 kbit/s. NFC is based on Radio frequency Identification (RFID) thus its communication involves initiator and a target, the initiator actively generates a Radio Frequency (RF) field that can be used as a signal to power a passive target. The initiator (active) has its own internal power that can be used to power the ICs that

generate the outgoing signal; while the target (passive) has only ICs with no internal power, which makes it to be in different forms like tags, stickers or cards.

NFC support three modes of operation they are: reader/writer mode, card emulation mode, and peer to peer mode. The communication in reader/writer mode is between NFC device and a tag in which device either read from a tag or write to a tag. Peer-to-Peer mode involves exchange of data between two NFC devices. While in card emulation mode the NFC device acts as a tag which will appear to an NFC reader as a contact-less smart card. One popular application of NFC is Smart Poster. The concept of Smart Poster is to keep information like URL, phone number, SMS into a tag and attached the tag to a physical object. This information can be accessed by touching the tag with NFC enabled device. The Smart Poster has some actions that can initiate a phone call, can launch a URL, or can send an SMS. In our proposed system two modes of operation will be used: Reader/writer mode (like smart poster) and Peer-to-Peer mode (like android beam). This paper presents our works in developing an NFC supported attendance system.

2.4 POS Terminal using Biometrics for Attendance Management System

This paper proposes an automatic attendance management system using biometrics that would provide the needed solution for automating attendance. An attendance management system is software developed for daily employee attendance in organizations. It facilitates access to the attendance of a particular employee in a particular department with a particular time. This system will also help in automatic generating reports and evaluating the attendance eligibility of an employee. Rather than signing an attendance sheet, individuals will pass their thumb over the fingerprint scanner, the finger print is compared against a list of pre-registered users, and once a match is made, the individual will be registered as having attended that particular day as many times he/she login/logout.

During enrolment, the biometrics of the user is captured (using a fingerprint reader, which are likely to be an optical, solid state or an ultrasound sensor or other suitable device) and the unique features are extracted and stored in a database as a template for the subject along with the student ID. The objective of the enrolment module is to admit an employee using his/her ID and

fingerprints into a database after feature extraction. These features form a template that is used to determine the identity of the employee, formulating the process of authentication. The enrolment process is carried out by an administrator in the attendance system. During authentication, the biometrics of the user is captured again and the extracted features are compared (using a matching algorithm) with the ones already existing in the database to determine a match.

Summary

This chapter mainly discusses about the papers that are referred for implementing the project. All papers provide information about taking feedback and recording attendance based on NFC, Biometrics, and online feedback. The proposed project is deals with collecting instant feedback and recording attendance to the students who give feedback, as by collecting instant feedback we are ensuring that the students do not have to go back and login to give feedback to all the sessions conducted on the day, rather give feedback to every class then and there.

CHAPTER-3

PROBLEM DEFINITION

In this section, we present detailed information about this thesis. First, we start discussing problem purpose and motivation, problem statement, then project objective, existing system that worked against the problem, finally the proposed system.

3.1 Purpose of the Project

In the traditional way of education system, the lecturer covers the topics that are scheduled for the day. The lecturer moves on to further topics unaware of the level of understanding of the previous topics covered, by the students. This cumulates to huge gap between the topics being covered and that have been understood by the students. This reduces the standard of education as the students are not in par with the topics being covered.

The main purpose of the proposed system is to bridge the gap between lecturers and students, by collecting instant appraisal from the students as to how well they have understood the covered topics. This cumulative appraisal from all the students is given to the lecturer, who can decide as to continue teaching further topics or re-visit the previous topics.

3.2 Motivation for the Project

The Android Based Course Delivery and Evaluation System is an automated way of collecting appraisals from the students regarding the performance in each class and at the same time recording the attendances of the students. As the digital world is growing exponentially, the need is to be able to adapt to the technology. The current educational system is a one way communication system, as the faculty will not know that whether students have understood the concepts taught.

The main aim of Android based Course Delivery Evaluation System is to improve the standard of education by continuous monitoring the students and analysing the appraisal provided by the students. The system is an Android based application system, that utilizes the current technology and helps the institution and the faculty to know the level of understanding of the students and to improve or retain the current teaching pattern or to improve it for their betterment.

The goal of this system is to make sure that the students who belong to such educational institutions have good knowledge and good core competencies.

3.3 Problem Statement

The project aims at building an Android application to records student's feedback about understanding of the session. The system provides real-time data to faculties about effectiveness and how well the student has understood the topic.

3.4 Project Objective

The objective of the project is to improve the standard of higher education by using ICT (Information Communication Technology). The proposed system will collect real time appraisals from students through the Android application. Gathered appraisal will be sent to the faculty with consolidated grade, this will assist the faculty to know the level of understanding of concepts by students. This will help the faculty for better planning and delivery of topics.

3.5 Existing Systems

Previous research in the automated attendance and appraisal systems make use of the following technologies to take attendance and collect appraisals.

- Questionnaire [1].
- Online feedback [2].
- NFC [3].
- Biometric [4].

3.5.1 Disadvantage of Existing Systems

The above existing system deals with taking feedback and recording attendance based on NFC, Biometrics, and online feedback. They do not deal with collecting instant feedback which is an integral part that helps in improving the quality of education through continuous evaluation. They mainly focus on collecting attendance and appraisal after the session is completed, which makes it difficult for students as they have to give feedback to all the sessions they have attended on that particular day.

3.6 Proposed System

The proposed system will automate appraisal and attendance. This minimizes wastage of student and faculty's productive time. It will improve the student's academic performance and understanding of the subjects through real-time collection of appraisals. The proposed system can be used in any educational institution for uplifting the quality of education and to ensure students with strong fundamental knowledge.

3.6.1 Advantages of Proposed System

In the proposed system, we are collecting instant feedback by the students and also recording the attendance for the session held. This system minimizes the overhead of recording attendance of all the students by the faculty manually. The system also helps the faculty to know the student's level of understanding of the topics covered. The faculty by knowing this can decide what corrective measures needed to be taken, thus helping in improving the overall standard of education in educational institution.

Summary

This chapter basically includes purpose, motivation and objectives of the project and detailed study of background related to the project, objective, existing solution and proposed solution.

CHAPTER-4

SYSTEM REQUIRMENTS SPECIFICATION

4.1 Introduction to System Requirements Specification

A software requirements specification (SRS) is a comprehensive description of the intended purpose and environment for software under development. The software requirements specification fully describes what the software will do and how it will be expected to perform.

A software requirements specification minimizes the time and effort required by developers to achieve desired goals and also minimizes the development cost. A good software requirements specification defines how an application will interact with system hardware, other programs and human users in a wide variety of real-world situations. Parameters such as operating speed, response time, availability, portability, maintainability, footprint, security and speed of recovery from adverse events are evaluated.

4.2 Functional Requirements

Most requirements definition focuses mainly on functional requirements, which are based upon the expected functioning of the product or system to be created. Functioning typically is equated with product/system features for which you might have a menu or button choice.

All things considered, requirements definers probably are best at identifying functional requirements, although they often overlook and get wrong more of the functional requirements than they ordinarily recognize. On hindsight reflection, they frequently do realize that many of the problems which surface later, and thus are harder and more expensive to fix, are attributable to inadequately addressed non-functional requirements.

4.3 Non-functional Requirements

Non-functional requirements refer to a whole slew (I've identified more than 30) of attributes including performance levels, security, and the various "ilities," such as usability, reliability, and availability. Invariably, requirements definers get wrapped up in how the product/system is expected to function and lose sight of these added elements.

When such factors are not addressed adequately, seemingly proper product/system functioning in fact fails to function. For example, a system may identify customers in such a slow, insecure, and difficult to use manner that it can cause mistakes which make data unreliable, provoke frustration-based attempted work-around that can create further problems, and ultimately lead to abandonment. That's the recognized way in which non-functional requirements impact product/system success. Other often unrecognized issues also need to be appreciated.

4.4 Software Requirements

Software Requirements is a field within software engineering that deals with establishing the needs of stakeholders that are to be solved by software. The IEEE Standard Glossary of Software Engineering Terminology defines a requirement as:

- A condition or capability needed by a user to solve a problem or achieve an objective.
- A condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed document.

Some of the important Software requirements required in our project are:

- **XAMPP Server**

XAMPP is a free and open source cross-platform web server solution stack package developed by Apache Friends, consisting mainly of the Apache HTTP Server, MariaDB database, and interpreters for scripts written in the PHP and Perl programming languages. XAMPP stands for Cross-Platform (X), Apache (A), MariaDB (M), PHP (P) and Perl (P). It is a simple, lightweight Apache distribution that makes it extremely easy for developers to create a local web server for testing and deployment purposes. Everything needed to set up a web server – server application (Apache), database (MariaDB), and scripting language (PHP) – is included in an extractable file. XAMPP is also cross-platform, which means it works equally well on Linux, Mac and Windows.

Since most actual web server deployments use the same components as XAMPP, it makes transitioning from a local test server to a live server extremely easy as well.

- **MySQL Database**

MySQL is a popular choice of database for use in web applications, and is a central component of the widely used XAMPP open-source web application software stack (and other "AMP" stacks). Free-software open-source projects that require a full-featured database management system often use MySQL.

- **Apache Server**

The Apache HTTP Server, colloquially called Apache, is the world's most used web server software. Apache is developed and maintained by an open community of developers under the auspices of the Apache Software Foundation. Most commonly used on a Unix-like system (usually Linux), the software is available for a wide variety of operating systems besides Unix, including eComStation, Microsoft Windows, NetWare, OpenVMS, OS/2, and TPF. Released under the Apache License, Apache is free and open-source software.

- **Android OS 4.4 or higher**

Android is a mobile operating system (OS) currently developed by Google, based on the Linux kernel and designed primarily for touchscreen mobile devices such as smartphones and tablets. Android's user interface is mainly based on direct manipulation, using touch gestures that loosely correspond to real-world actions, such as swiping, tapping and pinching, to manipulate on-screen objects, along with a virtual keyboard for text input. Variants of Android are also used on notebooks, game consoles, digital cameras, and other electronics.

- **SQLite database**

SQLite is a relational database management system contained in a C programming library. In contrast to many other database management systems, SQLite is not a client–server database engine. Rather, it is embedded into the end program. SQLite is ACID-compliant and implements most of the SQL standard, using a dynamically and weakly typed SQL syntax that does not guarantee the domain integrity. SQLite is a popular choice as embedded database software for local/client storage in application software such as web browsers. It is arguably the most widely

deployed database engine, as it is used today by several widespread browsers, operating systems, and embedded systems (such as mobile phones), among others. SQLite has bindings to many programming languages.

- **Android Studio**

Android Studio is the official integrated development environment (IDE) for Android platform development. Based on JetBrains' IntelliJ IDEA software, Android Studio is designed specifically for Android development. It is available for download on Windows, Mac OS X and Linux, and replaced Eclipse Android Development Tools (ADT) as Google's primary IDE for native Android application development.

- **Java Development Kit**

The Java Development Kit (JDK) is an implementation of either one of the Java Platform, Standard Edition, Java Platform, Enterprise Edition or Java Platform, Micro Edition platforms released by Oracle Corporation in the form of a binary product aimed at Java developers on Solaris, Linux, Mac OS X or Windows. The JDK includes a private JVM and a few other resources to finish the development of a Java Application. Since the introduction of the Java platform, it has been by far the most widely used Software Development Kit (SDK).

- **PHP (Hypertext Pre-processor)**

PHP is a server-side scripting language designed for web development but also used as a general-purpose programming language. PHP originally stood for Personal Home Page, but it now stands for the recursive backronym PHP: Hypertext Pre-processor. PHP code may be embedded into HTML code, or it can be used in combination with various web template systems, web content management system and web frameworks. PHP code is usually processed by a PHP interpreter implemented as a module in the web server or as a Common Gateway Interface (CGI) executable. The web server combines the results of the interpreted and executed PHP code, which may be any type of data, including images, with the generated web page. PHP code may also be executed with a command-line interface (CLI) and can be used to implement standalone graphical applications.

4.5 Hardware Requirements

The most common set of requirements defined by any operating system or software application is the physical computer resources, also known as hardware, a hardware requirements list is often accompanied by a hardware compatibility list (HCL), especially in case of operating systems. An HCL lists tested, compatible, and sometimes incompatible hardware devices for a particular operating system or application. The requirements of the system are:

- **Android supported smartphone**

The most common set of requirements defined by any operating system or software application is the physical computer resources, also known as hardware, a hardware requirements list is often accompanied by a hardware compatibility list (HCL), especially in case of operating systems. An HCL lists tested, compatible, and sometimes incompatible hardware devices for a particular operating system or application. The following sub-sections discuss the various aspects of hardware requirements.

- **Wi-Fi Network**

Wi-Fi is a technology that allows electronic devices to connect to a wireless LAN (WLAN) network, mainly using the 2.4 gigahertz (12 cm) UHF and 5 gigahertz (6 cm) SHF ISM radio bands. A WLAN is usually password protected, but may be open, which allows any device within its range to access the resources of the WLAN network. Devices which can use Wi-Fi technology include personal computers, video-game consoles, smartphones, digital cameras, tablet computers and digital audio players. Wi-Fi compatible devices can connect to the Internet via a WLAN network and a wireless access point. Such an access point (or hotspot) has a range of about 20 meters (66 feet) indoors and a greater range outdoors. Hotspot coverage can be as small as a single room with walls that block radio waves, or as large as many square kilometres achieved by using multiple overlapping access points.

- **Internal Memory**

The Application requires about 15-20Mb of internal memory to be installed. The phone's internal memory should be at least 20Mb of internal storage.

- **Random Access Memory (RAM)**

The Application requires about 512Mb or more inbuilt RAM phones. The phone's RAM must be at least 512Mb.

Summary

This chapter gives the details of the functional requirements, non-functional requirements, hardware and software requirements.

CHAPTER-5

SYSTEM DESIGN

Once the requirements document for the software to be developed is available, the software design phase begins. While the requirement specification activity deals entirely with the problem domain, design is the first phase of transforming the problem into a solution. In the design phase, the customer and business requirements and technical considerations all come together to formulate a product or a system.

The design process comprises a set of principles, concepts and practices, which allow a software engineer to model the system or product that is to be built. This model, known as design model, is assessed for quality and reviewed before a code is generated and tests are conducted. The design model provides details about software data structures, architecture, interfaces and components which are required to implement the system. This chapter discusses the design elements that are required to develop a software design model. It also discusses the design patterns and various software design notations used to represent a software design.

5.1 Principles of Design

Developing design is a cumbersome process as most expansive errors are often introduced in this phase. Moreover, if these errors get unnoticed till later phases, it becomes more difficult to correct them. Therefore, a number of principles are followed while designing the software. These principles act as a framework for the designers to follow a good design practice.

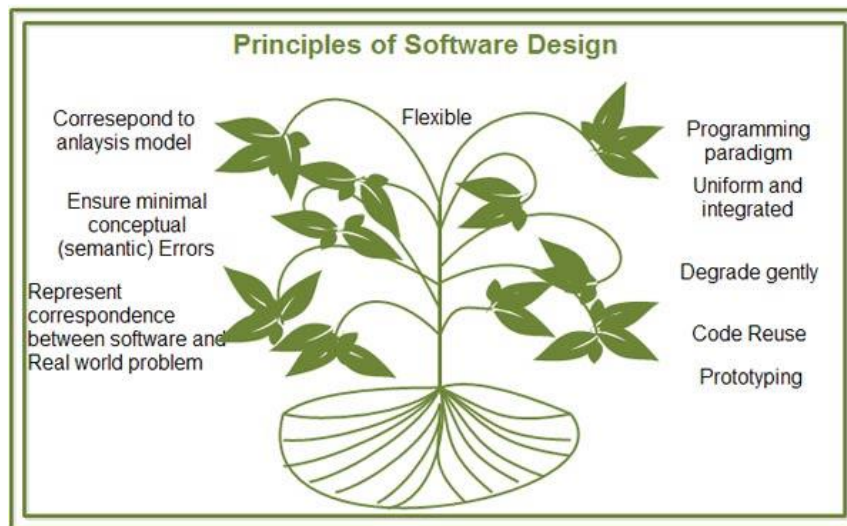


Figure 5.1: Principles of Software Design

Some of the commonly followed design principles are as following.

- **Software design should correspond to the analysis model:** Often a design element corresponds to, many requirements; therefore, we must know how the design model satisfies all the requirements represented by the analysis model.
- **Choose the right programming paradigm:** A programming paradigm describes the structure of the software system. Depending on the nature and type of application, different programming paradigms such as procedure oriented, object-oriented, and prototyping paradigms can be used. The paradigm should be chosen keeping constraints in mind such as time, availability of resources and nature of user's requirements.
- **Software design should be uniform and integrated:** Software design is considered uniform and integrated, if the interfaces are properly defined among the design components. For this, rules, format, and styles are established before the design team starts designing the software.
- **Software design should be flexible:** Software design should be flexible enough to adapt changes easily. To achieve the flexibility, the basic design concepts such as abstraction, refinement, and modularity should be applied effectively.
- **Software design should ensure minimal conceptual (semantic) errors:** The design team must ensure that major conceptual errors of design such as ambiguousness and inconsistency are addressed in advance before dealing with the syntactical errors present in the design model.
- **Software design should be structured to degrade gently:** Software should be designed to handle unusual changes and circumstances, and if the need arises for termination, it must do so in a proper manner so that functionality of the software is not affected.

- **Software design should represent correspondence between the software and real-world problem:** The software design should be structured in such a way that it always relates with the real-world problem.
- **Software reuse:** Software engineers believe on the phrase: 'do not reinvent the wheel'. Therefore, software components should be designed in such a way that they can be effectively reused to increase the productivity.
- **Designing for testability:** A common practice that has been followed is to keep the testing phase separate from the design and implementation phases. That is, first the software is developed (designed and implemented) and then handed over to the testers who subsequently determine whether the software is fit for distribution and subsequent use by the customer. However, it has become apparent that the process of separating testing is seriously flawed, as if any type of design or implementation errors are found after implementation, then the entire or a substantial part of the software requires to be redone. Thus, the test engineers should be involved from the initial stages. For example, they should be involved with analysts to prepare tests for determining whether the user requirements are being met.
- **Prototyping:** Prototyping should be used when the requirements are not completely defined in the beginning. The user interacts with the developer to expand and refine the requirements as the development proceeds. Using prototyping, a quick 'mock-up' of the system can be developed. This mock-up can be used as an effective means to give the users a feel of what the system will look like and demonstrate functions that will be included in the developed system. Prototyping also helps in reducing risks of designing software that is not in accordance with the customer's requirements.

5.2 High Level Design

High-level design (HLD) explains the architecture that would be used for developing a software product. The diagram provides an overview of an entire system, identifying the main components that would be developed for the product and their interfaces. The HLD uses possibly nontechnical to mildly technical terms that should be understandable to the administrators of the system. In contrast, low-level design further exposes the logical detailed design of each of these elements for programmers.

5.2.1 Use Case Diagram

A use case diagram is a graphic depiction of the interactions among the elements of a system. A use case is a methodology used in system analysis to identify, clarify, and organize system requirements.

- **Admin Use Case Diagram**

The below Use Case diagram depicts the role of Administrator, whose responsibilities are to add/update students and faculties details. He is also responsible for maintaining the system.

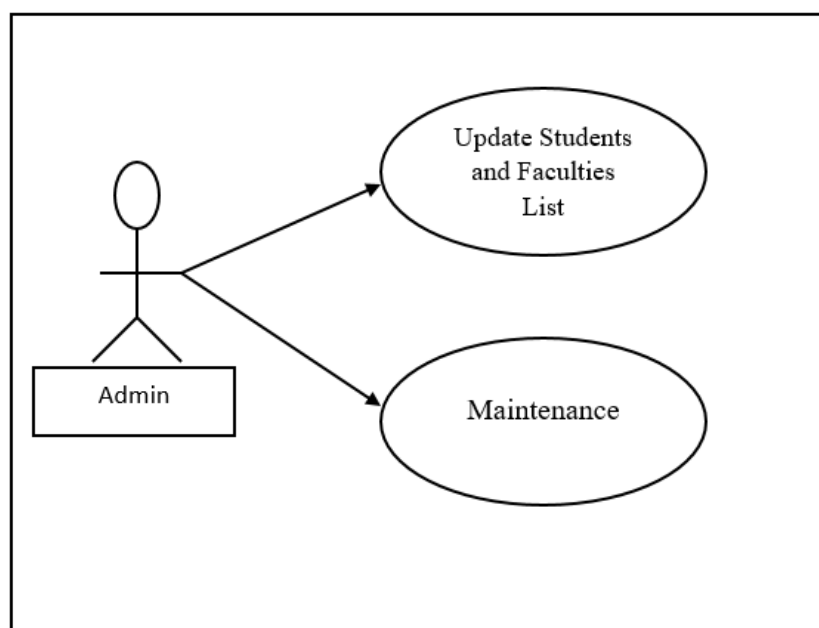


Figure 5.2: Admin Use Case Diagram

- **Faculty Use Case Diagram**

The user in the below Use Case diagram is the Faculty who is responsible for requesting students to give appraisal by sending the notification. Faculty can also view the appraisal given by the students for the class conducted.

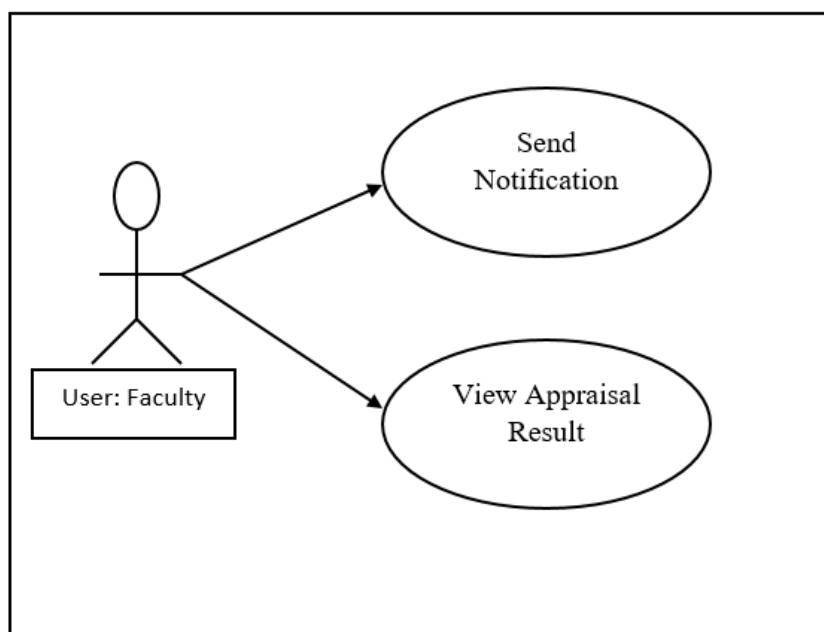


Figure 5.3: Faculty Use Diagram

- **Student Use Case Diagram**

The below Use Case diagram depicts the responsibilities of the user, student. Here the student who is the user is responsible for giving his/her instant feedback to the current session that is being held in the class. The student can also view his/her current attendance status in the application provided.

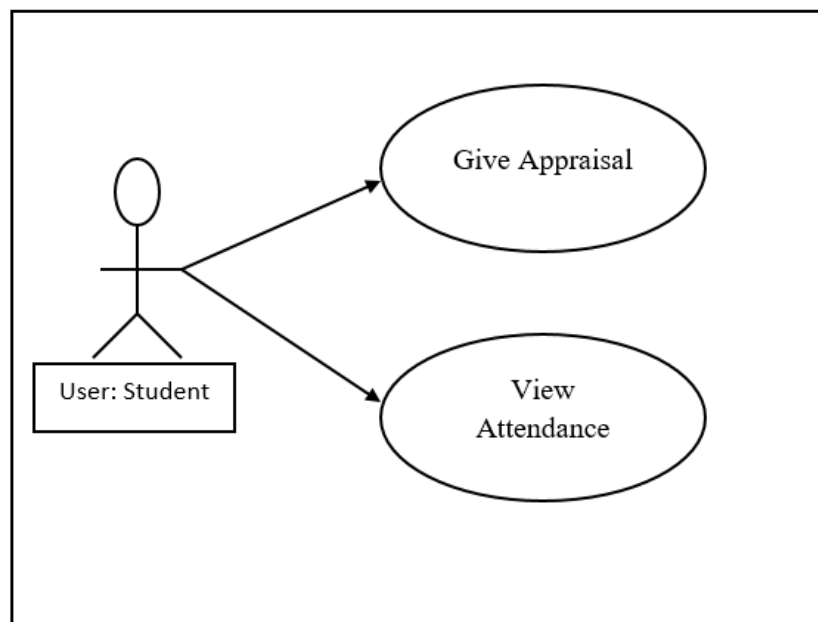


Figure 5.4: Student Use Case Diagram

5.2.2 Sequence Diagram

A Sequence diagram is an interaction diagram that shows how objects operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence.

- **Login Phase**

The below sequence diagram depicts the login phase for both students and faculties. The user, either student or faculty, has to open the application. On opening the application the user is redirected to the login page where the user is asked to enter the UserID in the provided space. On entering the UserID and submitting it, the details are forwarded to the server where it checks if the UserID exists or not. If the UserID exists, then the user is redirected to the verification phase and a OTP is sent to the user's registered mobile number. The user has to enter the OTP and this is forwarded to the server to check if it is correct.

If the OTP entered is correct then the user is redirected to the home page or if the OTP entered is incorrect then a toast message is displayed to user to re-enter the OTP correctly. The home pages are different for students and faculties.

The users can only Login to their account only once, and if they try to login from another device, then an error message is displayed. Once the user logs into the application the user need not login again.

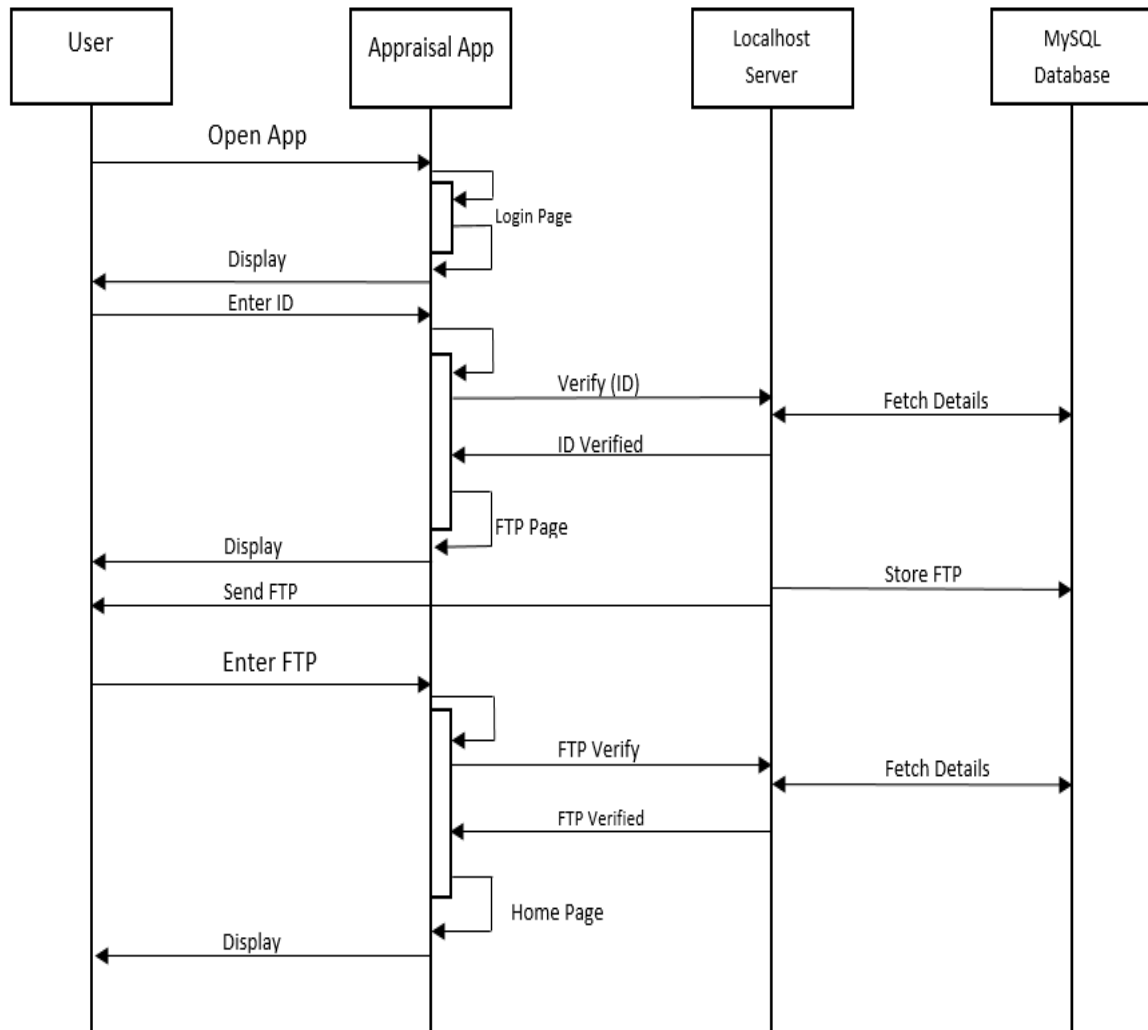


Figure 5.5: Login Phase Sequence Diagram

- **Faculty Appraisal Request phase**

The below sequence diagram mainly talks about the faculty requesting the students to give feedback of the current session conducted. The faculty, on opening the app, has two tabs, first one to request for appraisal, second to view the average appraisal given to them by the students for their previous sessions.

The faculty wants to request for appraisal, has to enter the subject code, semester, and the room number in which the current class is being held. Once the faculty fills in all details can click on send. The request is sent to the server where the server fetches all the registration ids of the students present in the class and request the Google Cloud Messaging to send Push Notification to the corresponding students.

After the students have given their feedback the faculty can ask for the headcount in the application, which returns the number of students that have given their feedback. If the number of students present in the class match the number of students who have given the feedback, then the faculty can accept and store the appraisals given, or if the numbers do not match then the faculty can reject the appraisal given as this corresponds to proxies in the appraisal.

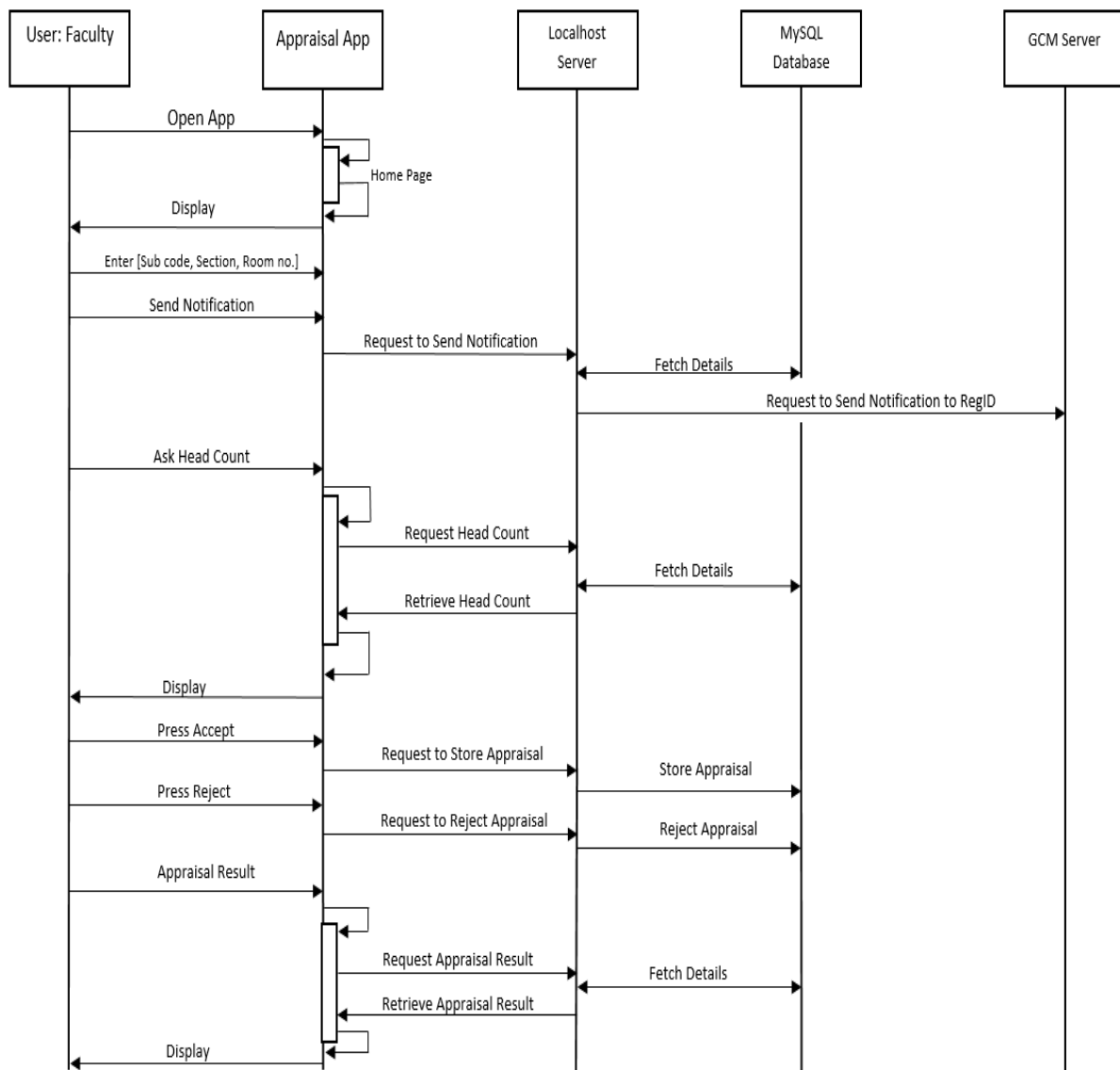


Figure 5.6: Faculty Phase Sequence Diagram

- **Student Response Phase**

The below sequence diagram depicts the how the students interact with the system. Once the student logs into the system the student's phone is registered with the Google Cloud Messaging for sending Push Notification. When the faculty requests for appraisal a Push Notification is sent to the respective students present in the class room.

The student on receiving the Push Notification has to click on the notification and then is redirected to the appraisal page. Here the student is prompted with a dialog box which consists of the Rating Bar. The student has to select number of stars based on their level of understanding of the topic that was taught. The rating is submitted to the server where the student feedback along with the timestamp is stored. Along with this the student's attendance is also incremented.

An average of cumulative feedback of all the students is sent to the faculty along with the head count, which is used by the faculty.

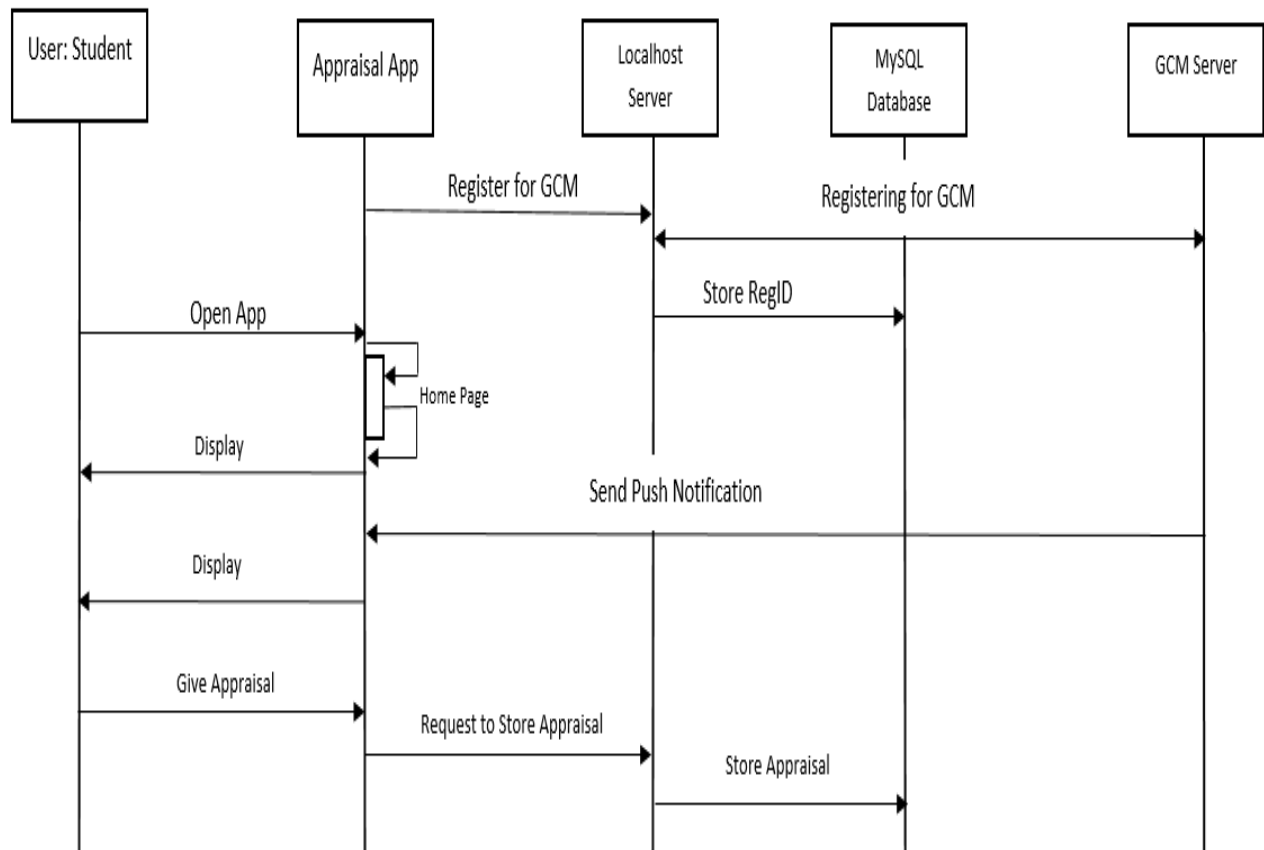


Figure 5.7: Student Phase Sequence Diagram

5.2.3 Flow Chart Diagram

A flowchart is a type of diagram that represents an algorithm, workflow or process, showing the steps as boxes of various kinds, and their order by connecting them with arrows. This diagrammatic representation illustrates a solution model to a given problem. Flowcharts are used in analysing, designing, documenting and managing processes and programs in various fields.

- **Login Phase**

The below Flow Chart depicts the login phase for both the users, students and faculty.

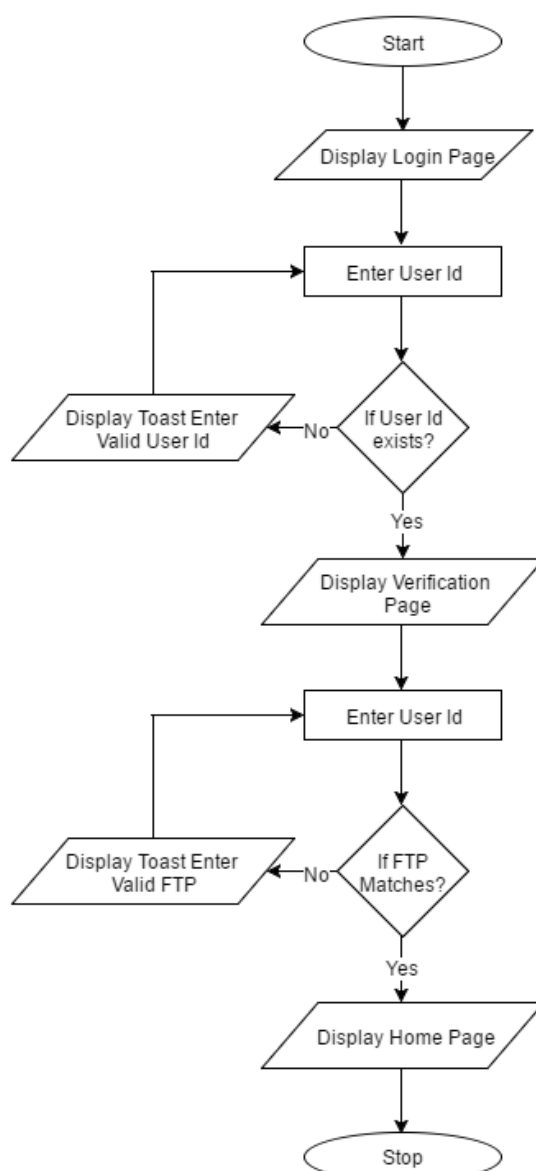


Figure 5.8: Login Phase Flow Chart Diagram

- **Faculty Appraisal Request phase**

The below Flow Chart depicts how the faculty initiates the appraisal request.

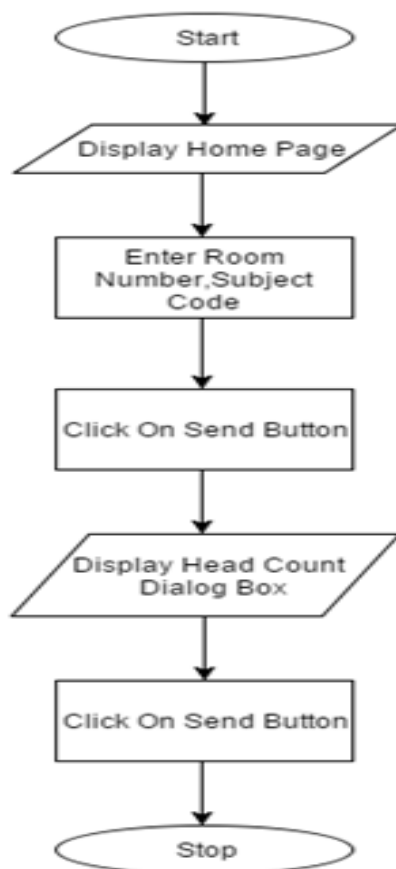


Figure 5.9: Faculty Phase Flow Chart Diagram

- **Students Response Phase**

The below Flow Chart depicts how the students give their feedback to the request initiated by the faculty.

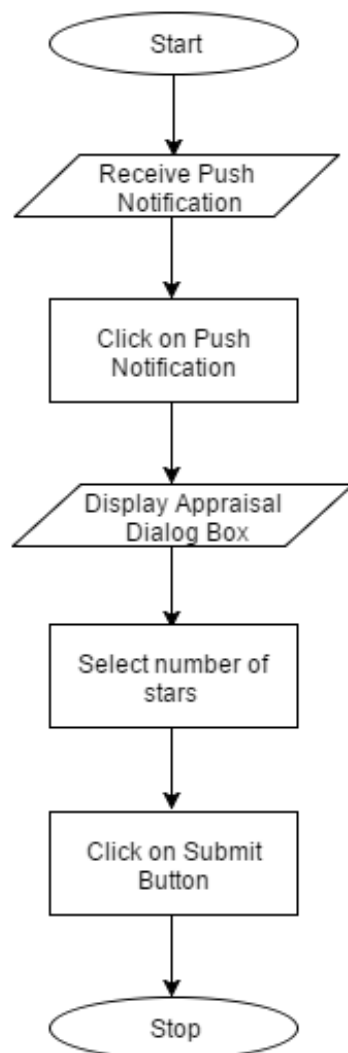


Figure 5.10: Student Phase Flow Chart Diagram

Summary

This chapter includes some basic design concepts, methodology, and some necessary diagram to outline the project easily.

CHAPTER-6

IMPLEMENTATION AND RESULTS

This chapter gives insight about how the whole application is running. The whole application can be classified into following modules

- Login
- Google Cloud Messaging
- Appraisal Request and Verification
- Evaluation and Attendance

There are two separate applications i.e. Student Application and Faculty Application. Both the applications start by displaying the welcome screen. In the below figure 6.1(a) displays the welcome screen for student and figure 6.1(b) shows the welcome screen for faculty.

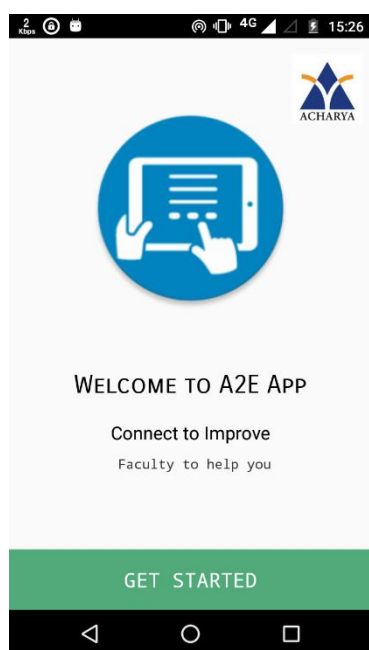


Figure 6.1(a) Student: Welcome Page

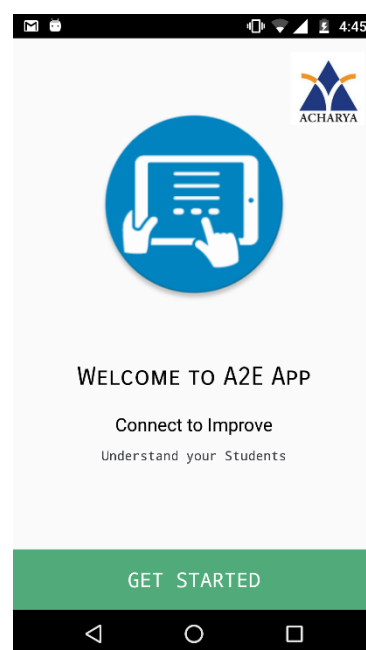


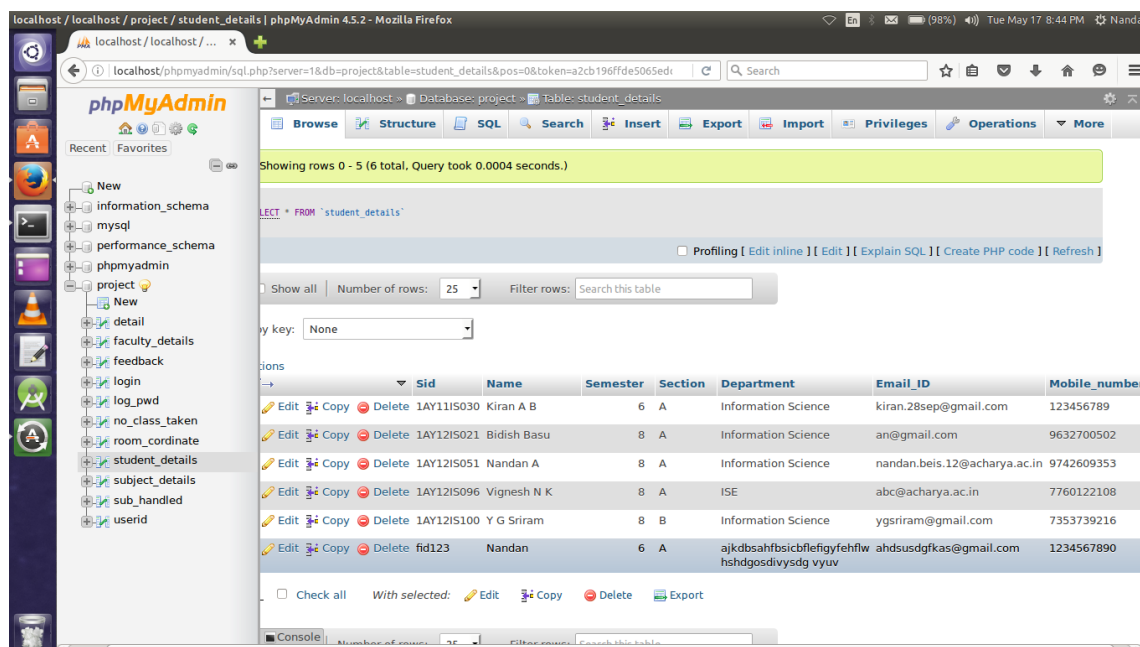
Figure 6.1(b) Faculty: Welcome Page

6.1 Login

The user has to go through two steps to successfully register with the server and login to the application. The two phases are

• ID Validation

In this step the user has to enter the user Id. The user details are entered into database by the admin. It is shown in figure 6.2(a) and figure 6.2(b).



Showing rows 0 - 5 (6 total, Query took 0.0004 seconds.)

SELECT * FROM 'student_details'

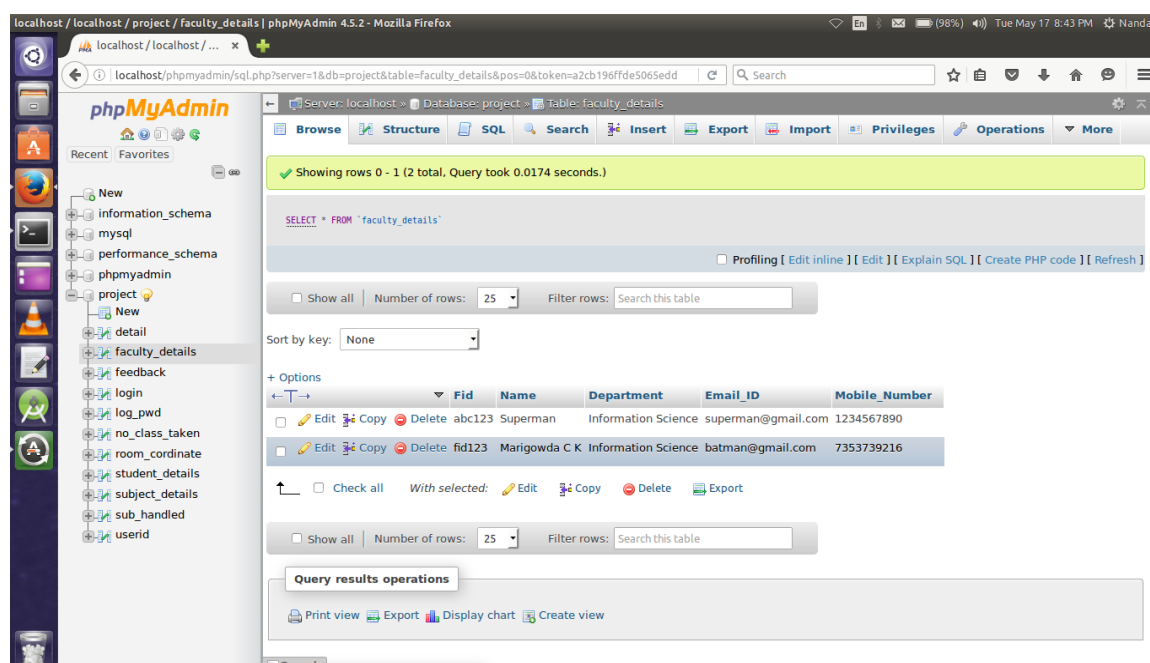
Number of rows: 25 Filter rows: Search this table

Sort by key: None

	Sid	Name	Semester	Section	Department	Email_ID	Mobile_number
Edit Copy Delete	1AY11IS030	Kiran A B	6	A	Information Science	kiran.28sep@gmail.com	123456789
Edit Copy Delete	1AY12IS021	Bidish Basu	8	A	Information Science	an@gmail.com	9632700502
Edit Copy Delete	1AY12IS051	Nandan A	8	A	Information Science	nandan.beis.12@acharya.ac.in	9742609353
Edit Copy Delete	1AY12IS096	Vignesh N K	8	A	ISE	abc@acharya.ac.in	7760122108
Edit Copy Delete	1AY12IS100	Y G Sriram	8	B	Information Science	ygsriram@gmail.com	7353739216
Edit Copy Delete	fid123	Nandan	6	A	Information Science	ahdsusdgfkas@gmail.com	1234567890

Check all With selected: Edit Copy Delete Export

Figure 6.2(a): Student Details



Showing rows 0 - 1 (2 total, Query took 0.0174 seconds.)

SELECT * FROM 'faculty_details'

Number of rows: 25 Filter rows: Search this table

Sort by key: None

	Fid	Name	Department	Email_ID	Mobile_Number
Edit Copy Delete	abc123	Superman	Information Science	superman@gmail.com	1234567890
Edit Copy Delete	fid123	Marigowda C K	Information Science	batman@gmail.com	7353739216

Check all With selected: Edit Copy Delete Export

Query results operations

Print view Export Display chart Create view

Figure 6.2(b): Faculty Details

Once the user clicks the next button the application send's a request to the server to generate a password which is sent to the user's registered Mail ID. In the below figure 6.3(a) is the ID Validation screen for the student and figure 6.3(b) is for the faculty.

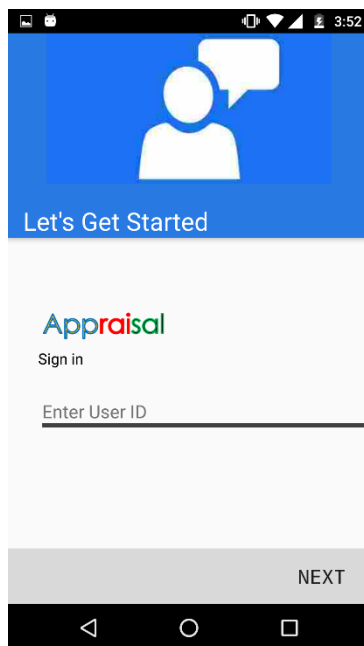


Figure 6.3(a): Student: ID Validation

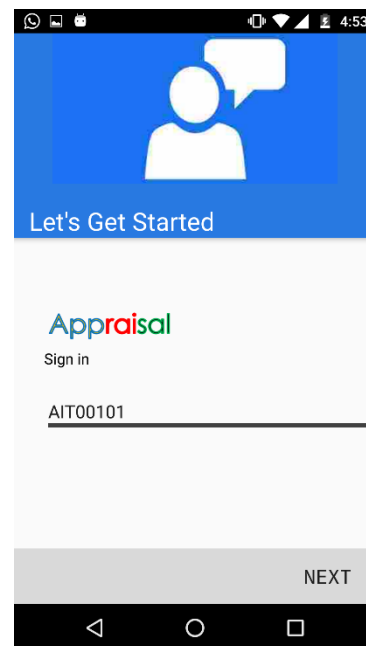


Figure 6.3(b): Faculty: ID Validation

- **First Time Password**

In this step the user has to enter the password send to him through to the registered Mail ID. The First time password also acts like OTP which when not entered within the timeframe the user is not able to Login successfully. Once the user clicks the Login button the application send's a request along with device ID. If the password is correct the server sends ID, Name, and Mobile Number, Email ID common to both student and faculty. In case of student it sends the subject details which the student has to study in that particular semester. In case of Faculty it sends the subject handled by him in that current semester. Figure 6.4 shows the First Time Password entry screen. The user is welcomed to the application using as shown in figure 6.5. The application also register's the user with the GCM server.

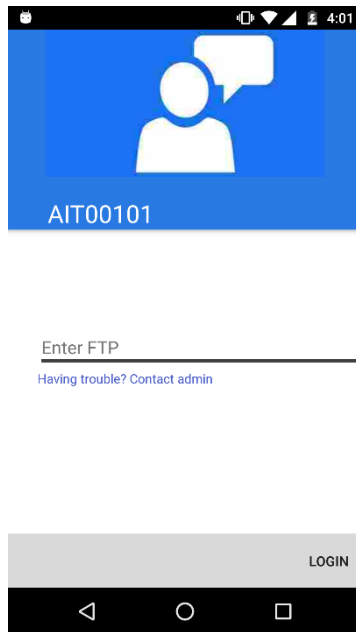


Figure 6.4: First Time Password Validation

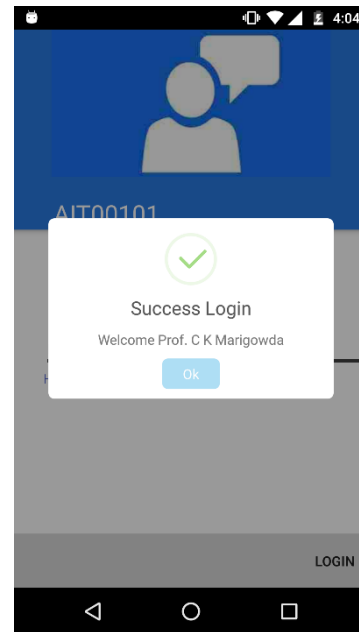


Figure 6.5: Successful Login Page

6.2 Google Cloud Messaging (GCM)

In order to get appraisal request notification from the faculty every student who login's into the system has to get him register himself with the GCM server. The registration is automatically handled by the application where the API fetches the device ID and registers the user with GCM server. On successful registration with the GCM server, the application sends the GCM registration ID and the student ID to the server. The server stores the USN and GCM registration ID as shown in figure 6.6.

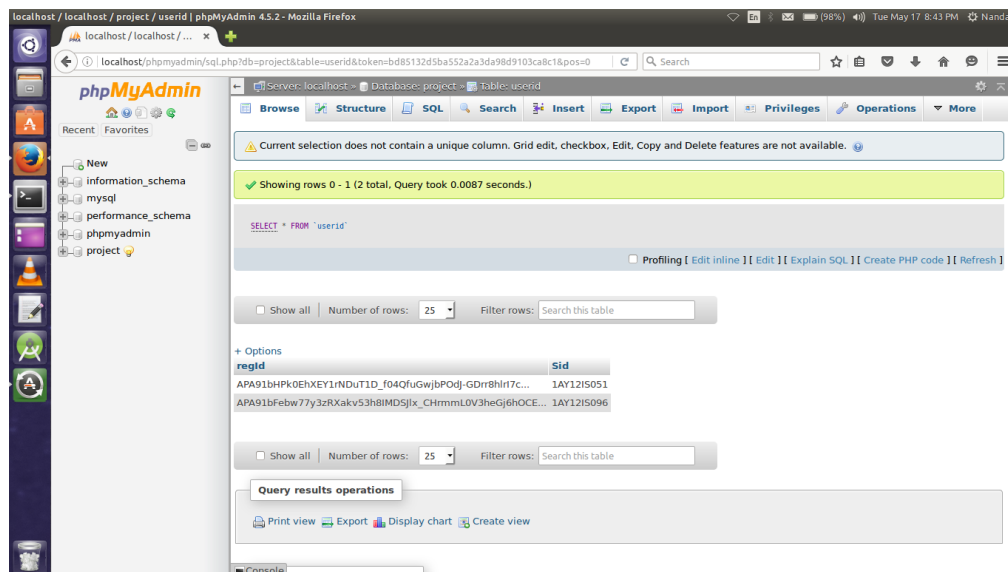


Figure 6.6: Student GCM registration

6.3 Appraisal Request and Verification

• Appraisal Request

Once the faculty has logged in to the application he is displayed the Appraisal request page which is by default the homepage of the application. The faculty has to enter the subject code, room no and Section in which class is being taken. The application sends the semester and faculty ID along with other parameters. Once the server has received the request it sends the request through the GCM to the registered Students. Figure 6.7(a) shows the entry page for subject code, room no and section. On successfully sending the request it displays notification send as a toast message as shown in figure 6.7(b).

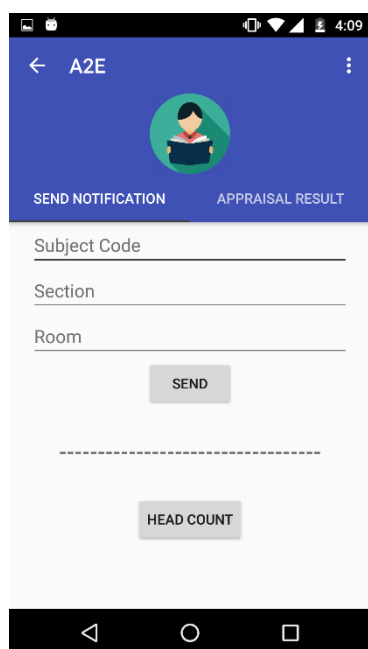


Figure 6.7(a): Appraisal Request Page

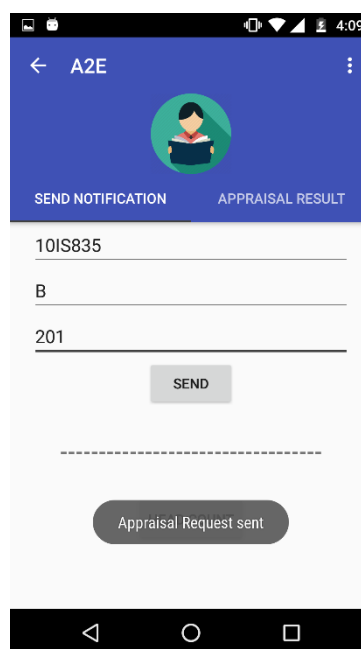


Figure 6.7(b): Appraisal Request Sent

• Appraisal Collection

Once the faculty has sent a request to give feedback, all the student receives a notification as shown in figure 6.8(a). On clicking the notification the student gets the appraisal dialog where the student has to enter his response as show in figure 6.8(b). On clicking submit the appraisal is recorded in the database and a success message is displayed to the student as Shown in figure 6.9.

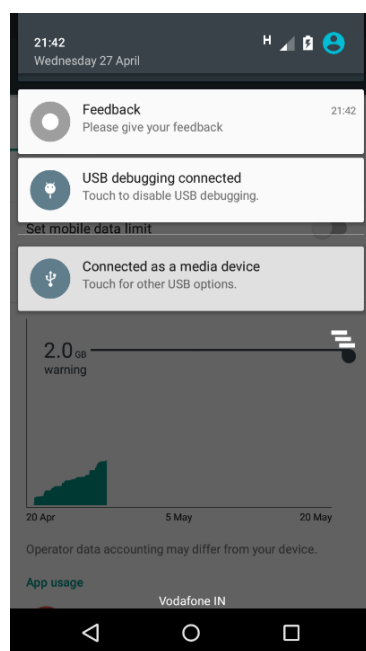


Figure 6.8(a): Appraisal Notification

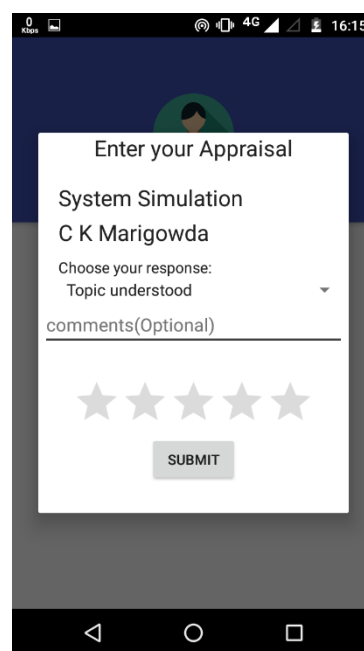


Figure 6.8(b): Appraisal Entry Page

Sid	Rating	time	Fid	subject
1AY12IS051	3	2016-05-15	201	10IS835
1AY12IS051	2	2016-05-15	fid123	10IS835
1AY12IS051	2	2016-05-15	fid123	10IS835
1AY12IS051	3	2016-05-15	fid123	10IS835
1AY12IS051	1.5	2016-05-15	fid123	10IS835
1AY12IS051	3	2016-05-15	fid123	10IS835
1AY12IS051	3	2016-05-15	fid123	10IS835
1AY12IS051	3	2016-05-15	fid123	10IS835
1AY12IS051	3	2016-05-15	fid123	10CS82
1AY12IS051	3	2016-05-15	fid123	10CS82
1AY12IS051	3	2016-05-15	fid123	10CS82
Console	1	2016-05-15	fid123	10CS82

Figure 6.9: Appraisal Record

• Appraisal Verification

In order to avoid false collection of appraisal, this step is followed which verifies that whether the appraisal of the student in the class is only recorded. Once all the students have

provided their responses, the faculty asks whether all the students present have given their Appraisal.

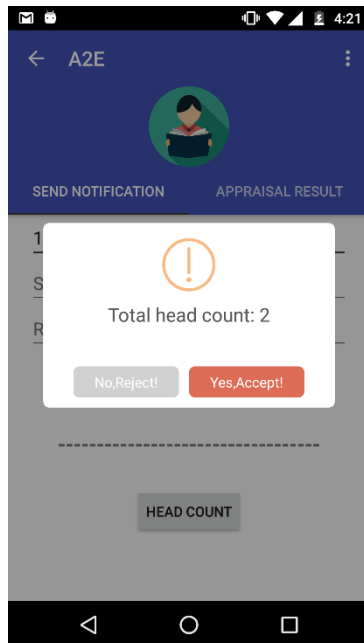


Figure 6.10: Head Count Dialogue Box

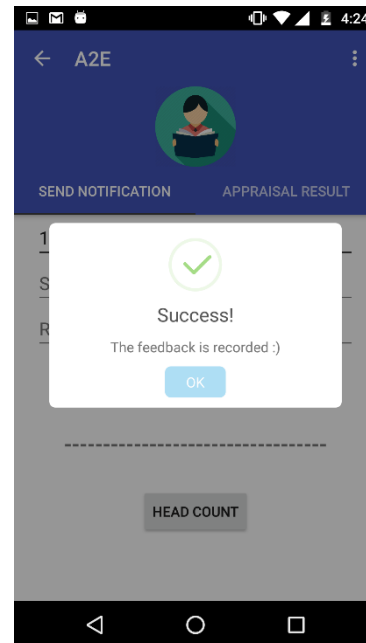


Figure 6.11: Appraisal Stored

When all the students has given their response the faculty presses stop button which stops all the incoming appraisal and provides a count of no of student present in the class as shown in figure 6.10. The faculty has to manually take a headcount of no of student and has to verify it with it server provided value. If there is no mismatch the faculty presses accept which stores the value in the database as shown in figure 6.11. If the faculty find's a mismatch in count he reject's all the appraisal as shown in figure 6.12(a) and 6.12(b).

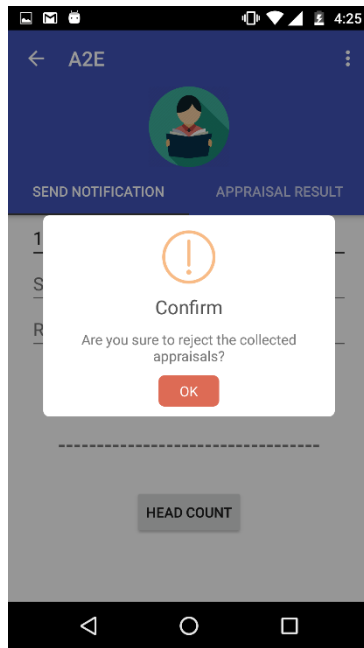


Figure 6.12(a): Rejection Confirmation

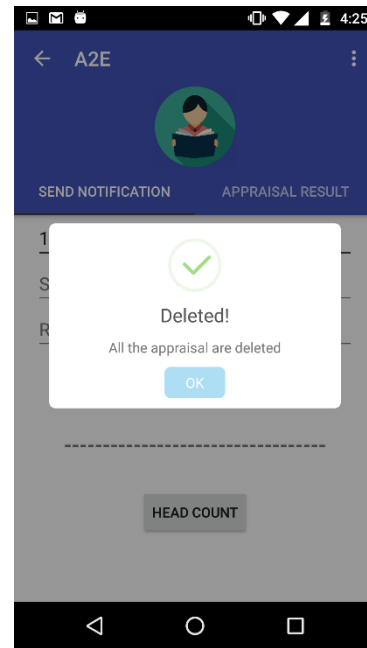


Figure 6.12(b): Appraisal Rejected

6.4 Evaluation and Attendance

• Appraisal Evaluation

The lecturer can check the appraisal collected for the class whenever he wants. The application requests for appraisal. The server finds the average of all the appraisals collected and sends it to the application. After receiving the average the application displays it with subject code, date when appraisal was taken and the appraisal in form of no of stars as shown in figure 6.13.

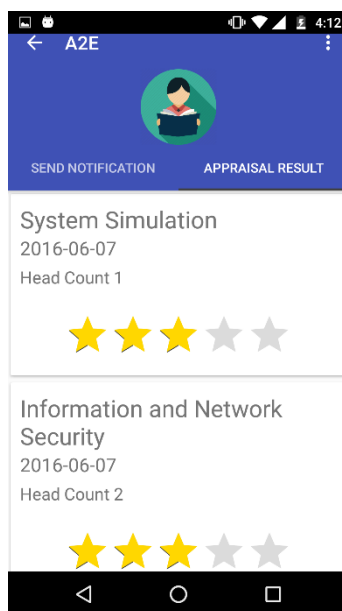


Figure 6.13: Appraisal Results

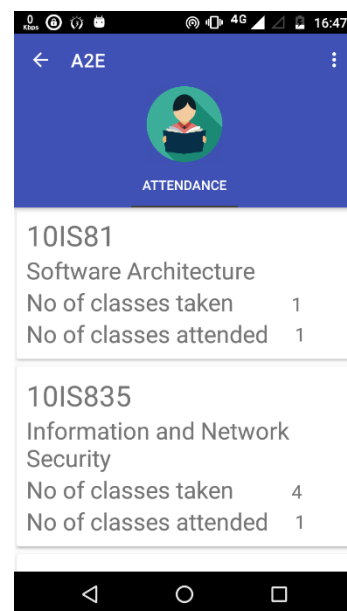


Figure 6.14: Attendance Page

- **Attendance**

Since feedback is taken for each class, the application also provides the student with updated value of their attendance. The student can check their attendance for all the subjects they attend after each appraisal. The attendance page is shown in figure 6.14.

Summary

This describes the implementation of different phases and snapshots depicting different pages. Introduces about the technology and methodology used to implement the project.

CHAPTER-7

TESTING

Testing is the process of evaluating a system or its component(s) with the intent to find whether it satisfies the specified requirements or not. Testing is executing a system in order to identify any gaps, errors, or missing requirements in contrary to the actual requirements.

7.1 Testing Principle

Before applying methods to design effective test cases, a software engineer must understand the basic principle that guides software testing. All the tests should be traceable to customer requirements.

7.2 Testing Methods

There are different methods that can be used for software testing. They are

- **Black-Box Testing**

The technique of testing without having any knowledge of the interior workings of the application is called black-box testing. The tester is oblivious to the system architecture and does not have access to the source code. Typically, while performing a black-box test, a tester will interact with the system's user interface by providing inputs and examining outputs without knowing how and where the inputs are worked upon.

- **White-Box Testing**

White-box testing is the detailed investigation of internal logic and structure of the code. White-box testing is also called glass testing or open-box testing. In order to perform white-box testing on an application, a tester needs to know the internal workings of the code. The tester needs to have a look inside the source code and find out which unit/chunk of the code is behaving inappropriately.

7.3 Levels of Testing

There are different levels during the process of testing. Levels of testing include different methodologies that can be used while conducting software testing. The main levels of software testing are:

- **Functional Testing:**

This is a type of black-box testing that is based on the specifications of the software that is to be tested. The application is tested by providing input and then the results are examined that need to conform to the functionality it was intended for. Functional testing of software is conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. There are five steps that are involved while testing an application for functionality.

- The determination of the functionality that the intended application is meant to perform.
- The creation of test data based on the specifications of the application.
- The output based on the test data and the specifications of the application.
- The writing of test scenarios and the execution of test cases.
- The comparison of actual and expected results based on the executed test cases.

- **Non-functional Testing**

This section is based upon testing an application from its non-functional attributes. Non-functional testing involves testing software from the requirements which are non-functional in nature but important such as performance, security, user interface, etc. Testing can be done in different levels of SDLC. Few of them are

7.3.1 Unit Testing

Unit testing is a software development process in which the smallest testable parts of an application, called units, are individually and independently scrutinized for proper operation. Unit testing is often automated but it can also be done manually. The goal of unit testing is to isolate each part of the program and show that individual parts are correct in terms of requirements and functionality. Test cases and results are shown in the below table.

Serial Number	Components Tested	Description	Status	Remarks
1	Login Page	Check if the login details are submitted to the server and if the server responds by sending the FTP	Pass	Verification takes 2-3 seconds
2	Verification Page	Check if the FTP generated by the server and the FTP received by the user are the same and the FTP generated to each user is distinct	Pass	FTP generated is Distinct
3	Student Detail Page	Check whether the details sent matches the user	Pass	Application fetches details correctly
4	Faculty Home Page	Check if the faculty is able to Select the semester, section and class room using drop down menu	Pass	Faculty is given less overhead by providing Drop down Menu.
5	GCM Registration	Check if the GCM registration ID is generated for all the users who have logged in	Pass	GCM server is able to generate the ID for all users
6	Push Notifications	Check if the push notifications are sent or not	Pass	Push notifications are sent by the GCM server to the user
7	Rating Page	Check if the rating along with the recording of the attendances is handled by the server or not	Pass	The server is able to store the rating and is able to record the attendances

Table 7.1 Test Cases for Unit Testing

7.3.2 Integration Testing

Integration testing is defined as the testing of combined parts of an application to determine if they function correctly. It occurs after unit testing and before validation testing. Integration testing can be done in two ways: Bottom-up integration testing and Top-down integration testing.

- **Bottom-up integration**

This testing begins with unit testing, followed by tests of progressively higher-level combinations of units called modules or builds.

- **Top-down integration**

In this testing, the highest-level modules are tested first and progressively, lower-level modules are tested thereafter.

In a comprehensive software development environment, bottom-up testing is usually done first, followed by top-down testing. The process concludes with multiple tests of the complete application, preferably in scenarios designed to mimic actual situations. The below table shows the test cases for integration testing and their results.

Serial Number	Components Tested	Description	Status	Remark
1	Login Phase	Check if the user is able to login or not	Pass	Server verifies the user and is able to send the FTP through the e-mail
2	Rating Phase	Check if the rating is sent to all concerned students by the server	Pass	Local server is able to request the GCM server to send push notifications to all concerned students
3	Appraisal Gathering Phase	Check if all student rating is handled by the server	Pass	The server is able to handle request and record the data

Table 7.2: Test Cases for Integration Testing

7.3.3 Component Interface Testing

Testing a module or component independently to verify its expected output is called component testing. Generally component testing is done to verify the functionality and/or usability of a component but not restricted to only these. A component can be of anything which can take inputs and delivers some output. Like module of code, web page, screens and even a system inside a bigger system is a component to it.

7.3.4 System Testing

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black-box testing, and as such, should require no knowledge of the inner design of the code or logic. System testing is important because of the following reasons:

- System testing is the first step in the Software Development Life Cycle, where the application is tested as a whole.
- The application is tested thoroughly to verify that it meets the functional and technical specifications.

- The application is tested in an environment that is very close to the production environment where the application will be deployed.
- System testing enables us to test, verify, and validate both the business requirements as well as the application architecture.

The below table shows the test cases for System testing with their corresponding results.

Serial Number	Components Tested	Description	Status	Remark
1	Installation	Check whether the application is installing as desired	Pass	Application is installing as desired
2	Uninstallation	Check whether the application is uninstalling without leaving any data behind	Pass	Application is uninstalling as desired
3	Student Application	Check whether the student application is working as desired	Pass	Student application is working as desired
4	Faculty Application	Check whether the faculty application is working as desired	Pass	Faculty application is working as desired

Table 7.3: Test Cases for System Testing

7.3.5 Acceptance Testing

This is arguably the most important type of testing, as it is conducted by the Quality Assurance Team who will gauge whether the application meets the intended specifications and satisfies the client's requirement. The QA team will have a set of pre-written scenarios and test cases that will be used to test the application.

More ideas will be shared about the application and more tests can be performed on it to gauge its accuracy and the reasons why the project was initiated. Acceptance tests are not only intended to point out simple spelling mistakes, cosmetic errors, or interface gaps, but also to point out any bugs in the application that will result in system crashes or major errors in the application. By performing acceptance tests on an application, the testing team will deduce how the application

will perform in production. There are also legal and contractual requirements for acceptance of the system.

7.3.6 Alpha Testing

This test is the first stage of testing and will be performed amongst the teams (developer and QA teams). Unit testing, integration testing and system testing when combined together are known as alpha testing. During this phase, the following aspects will be tested in the application:

- Spelling Mistakes
- Broken Links
- Cloudy Directions
- The Application will be tested on machines with the lowest specification to test loading times and any latency problems.

7.3.7 Beta Testing

This test is performed after alpha testing has been successfully performed. In beta testing, a sample of the intended audience tests the application. Beta testing is also known as pre-release testing. Beta test versions of software are ideally distributed to a wide audience on the Web, partly to give the program a "real-world" test and partly to provide a preview of the next release. In this phase, the audience will be testing the following:

- Users will install, run the application and send their feedback to the project team.
- Typographical errors, confusing application flow, and even crashes.
- Getting the feedback, the project team can fix the problems before releasing the software to the actual users.
- The more issues you fix that solve real user problems, the higher the quality of your application will be.
- Having a higher-quality application when you release it to the general public will increase customer satisfaction.

Summary

This chapter defines various testing methods used for testing the different phases in the project; they are unit testing, integration testing and system testing mainly.

CHAPTER-8

CONCLUSION AND FUTURE ENHANCEMENT

8.1 Conclusion

The Android Based Course Delivery and Evaluation system is a newer approach to collect reviews from the students for each and every class taken by the faculties. It proposes a different approach to collecting and storing the appraisals which the faculties can view and make the necessary improvements that would help the students understand the topics better and gain knowledge.

8.2 Future Enhancement

The developed application covers instant appraisal and also recording attendance of the respective students who have given feedback.

The application can be improvised by providing Geo-Fencing for sending the Push Notification to the students who are present only in that particular class room, by providing separate views for Proctor, Head Of The Department and Principal, by providing additional security enhancements, alert faculty when his/her performance based on the feedback results reduces. We have listed only few future enhancements of the project.

REFERENCES

- [1] Shanshan Hu and Yonghua Xie , “Improve Student Learning Using Online Formative Assessment System”, IEEE, pp. V1-235 – V1-257, 2010
- [2] Andrei Mărcuş, Monica Zaharie, Codruța Osoian, “Student Satisfaction as a Quality Management Technique in Higher Education”, IEEE, pp. 388-391, 2009
- [3] Media Anugerah Ayu, Member, IACSIT and Barroon Ismaeel Ahmad, “TouchIn: An NFC Supported Attendance System in a University Environment”, IJJET, Vol 4, Vol 5, 2014
- [4] D. Madhuri, Padmaja .T, Pasp Naidu, “POS Terminal using Biometrics for Attendance Management System”, IJARCSSE, Vol 4, 2014
- [5] www.developers.android.com
- [6] www.stackoverflow.com
- [7] www.androidhive.com
- [8] www.javapapers.com
- [9] www.tutorialspoint.com
- [10] www.vogella.com
- [11] developers.google.com/cloud-messaging.com
- [12] www.javatechig.com/android/android-cardview-example.com