#### 

#### **E:\ShareLink\Download\26_03_15_093820_logo.jpgE:\ShareLink\Download\5_Menu_Untitled.png** A PROJECT REPORT **ON**

**HOUSING MANAGEMENT SYSTEM**

**SUBMITTED BY**

#### MR. RAHUL GUPTA

#### AND

#### MR. DEEPAK JAIN

***Under Guidance of PROF. SMRUTI NANAVATY***

***Submitted in partial fulfilment of the requirements for***

***qualifying B.Sc. – (I.T.),***

***Semester – VI Examination***

#### USHA PRAVIN GANDHI COLLEGE OF MANAGEMENT

###### VILE PARLE (WEST)

###### MUMBAI - 400056

###### 2017 - 2018

College Certificate

**ACKNOWLEDGEMENT**

We would like to take this opportunity to express my profound gratitude to all those who helped us in the development of my project.

We thank the Principal of Usha Pravin Gandhi College of Management,

**Dr. (Mrs.) Anju Kapoor** for her support that inspired us to successfully conclude this project.

We would like to thank out professors and staff-coordinators for their constant support, encouragement and guidance without which the successful completion of this project would be distant reality.

We would like to especially thank our stream coordinator **Prof. Swapnali Lotlikar** along with **Prof. Smruti Nanavaty**, **Prof. Wahid Kapadia, Prof. Sunita Gupta, Prof.**

**Sohrab Vakharia** for always being supportive and inspiring us throughout the completion of our project.

We express our sincere thanks and extreme gratitude to **USHA**

**PRAVIN GANDHI COLLEGE OF MANAGEMENT** for providing us with valuable and necessary infrastructure to carry out my project work.

It is something of a cliché to say that –“All our teachers from

Kindergarten till date have touched and shaped us well”. But like most clichés this one has more than an ounce of truth.

We would like to thank **Jagdish** Sir, our lab assistant **Sandeep** Sir and all of our friends whose kindness and helping nature knows no bounds.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **SR. NO** | **TOPICS** | **PG NO.** |
| **1** | **Scope Of The System** | 6 |
| 1.1 | Introduction | 7 |
| 1.2 | Existing System and It’s Limitation | 8- 9 |
| 1.3 | Proposed System and It’s Advantages | 10 - 11 |
| 1.4 | Objective Of Proposed System | 12 |
| 1.5 | Fact Finding Techniques | 13 - 14 |
| **2** | **Technologies Used** | 15 |
| 2.1 | Technologies Used | 16 - 18 |
| 2.2 | Software Used | 19 - 21 |
| **3** | **Feasibility Study** | 22 - 23 |
| **4** | **Methodology Adopted** | 24 |
| 4.1 | Model Used | 25 - 29 |
| **5** | **Diagrams** | 30 |
| 5.1 | Gantt Chart | 31 |
| 5.2 | Use Case Diagram | 32 - 33 |
| 5.3 | E-R Diagram | 34 |
| 5.4 | Activity Diagram | 35 |
| **6** | **Screenshots** | 36 - 43 |
| **7** | **Code** | 44 - 59 |
| **8** | **Testing** | 60 - 67 |
| **9** | **System Maintenance and User Manual** | 68 - 70 |
| 10 | **Conclusion** | 71 |
| 10.1 | Limitation | 72 |
| 10.2 | Future Scope | 73 |
| 10.3 | Bibliography | 74 |
| 10.4 | References | 75 |

**CHAPTER 1:**

**SCOPE OF THE PROJECT**



**1.1 Introduction**

* A housing society management and billing project that effectively manages and handles all the functioning of a cooperative housing society.
* The software system can store the data of various flat owners and their family members.
* The system also maintains and calculates the society maintenance as well as parking, cultural funds, emergency funds and other charges and adds them automatically in individual flat bill.
* The system needs an administrator to update various flat owner data and billing amounts into it.
* The rest of the work is done by the system on its own. The system consists of automatic bill generation facility. It calculates various associated costs, adds them up and provides a bill accordingly.
* This application also has a notice board for any news that needs to be circulated amongst the society members.
* This application also provides a user to make complaints to Admin about society related issues. For eg. Complaint about late night party music.

**1.2 Existing System and It’s Drawbacks**

* Society members maintenance records are stored in excel or physical copies which isn’t organized properly.
* Society members are informed about the due dates of fees by personally reminding them by the secretary.
* Records cannot be added/updated on the go.
* There’s no secured storage/backup of the excel sheets.
* Every detail about the society needs to be shared personally with every member of society.
* Society members cannot keep a check of their maintenance records and notices. (lack of transparency).
* Human errors are possible at many places which cannot be avoided.
* Answering to every member about every query is very difficult for the secretary..
* The records maintained in excel sheets are not in the same file, they are spread randomly, which is difficult to manage and does not allow to fetch all the details of any particular member easily.

**1.3 Proposed System & It’s Advantages**

* A housing society management and billing project that effectively manages and handles all the functioning of a cooperative housing society.
* The software system can store the data of various flat owners and their family members.
* The system also maintains and calculates the society maintenance as well as parking, cultural funds, emergency funds and other charges and adds them automatically in individual flat bill.
* The system would be user-friendly so as to make it easier for people from almost all age groups to access their account and make use of all the features provided to them.
* The system needs an administrator to update various flat owner data and billing amounts into it.
* The rest of the work is done by the system on its own. The system consists of automatic bill generation facility. It calculates various associated costs, adds them up and provides a bill accordingly.
* This application also has a notice board for any news that needs to be circulated amongst the society members.
* This application also provides a user to make complaints to admin about society related issues. For eg. Complaint about late night party music.
* Admin has the rights to update or delete a member if any member is not living in the society.

**1.4 Objective of Proposed System**

* To reduce paperwork.
* To reduce operational time.
* Increased accuracy and reliability.
* Increased operational efficiency.
* Data security.
* Data availability.
* Ease for accessing of information (mobility).
* Data Backup & Storage facility.
* Improving transparency for society members and society officials.

**1.5 Fact Finding Techniques**

Fact finding technique is an important activity in system investigation. In this stage, the functioning of the system is to be understood by the system analyst to design the proposed system. Various methods are used for this and these are known as fact - finding techniques. The analyst needs to fully understand the current system.

The analyst needs data about the requirements and demands of the project undertaken. The techniques employed to gather this data are known as fact-finding techniques.

We made use of the following fact finding techniques for collecting the data:-

* **Observation:** This is a technique wherein the system analyst either participates in or watches a person perform activities to learn about the system. So we decided to visit few societies to understand the working of their society which helped us to understand their requirements in much better way.
* **Sample of Existing System:** We requested our societies’ secretary to allow us to go through their few of existing documents that they use for maintaining the records of society members. We used those registers and excel sheets to understand their facts and manner of storing data.
* **Interviews:** Interview is the most commonly used technique to collect information from the face-to-face interviews. The purpose of interview is to find, verify, clarify facts, motivate end-users involved, identify requirements and gather ideas and opinions. The role of interview includes interviewer who is system analyst and an interviewee who is the system owner or user. Interviewing technique needs good communication skills for interaction between system analyst and user. For our project, We used unstructured format of interview wherein open-ended questions were asked to our neighbours and classmates.

**CHAPTER 2:**

**TECHNOLOGIES AND FEASIBILITY**

**STUDY**



**2.1 Technologies Used**

* **XML:** Extensible Markup Language (XML) is a [markup language](https://en.wikipedia.org/wiki/Markup_language" \o "Markup language) that defines a set of rules for encoding [documents](https://en.wikipedia.org/wiki/Electronic_document" \o "Electronic document) in a [format](https://en.wikipedia.org/wiki/File_format" \o "File format) that is both [human-readable](https://en.wikipedia.org/wiki/Human-readable_medium" \o "Human-readable medium) and [machine-readable](https://en.wikipedia.org/wiki/Machine-readable_data" \o "Machine-readable data). The [W3C](https://en.wikipedia.org/wiki/World_Wide_Web_Consortium" \o "World Wide Web Consortium)'s XML 1.0 Specification and several other related specifications, all of them are free [open standards](https://en.wikipedia.org/wiki/Open_standard" \o "Open standard) that define XML. The design goals of XML emphasize simplicity, generality, and usability across the [Internet](https://en.wikipedia.org/wiki/Internet" \o "Internet). It is a textual data format with strong support via [Unicode](https://en.wikipedia.org/wiki/Unicode" \o "Unicode) for different [human languages](https://en.wikipedia.org/wiki/Language" \o "Language). Although the design of XML focuses on documents, the language is widely used for the representation of arbitrary [data structures](https://en.wikipedia.org/wiki/Data_structure" \o "Data structure) such as those used in [web services](https://en.wikipedia.org/wiki/Web_service" \o "Web service). Several [schema systems](https://en.wikipedia.org/wiki/XML_schema" \o "XML schema) exist to aid in the definition of XML-based languages, while programmers have developed many [application programming interfaces](https://en.wikipedia.org/wiki/Application_programming_interface" \o "Application programming interface) (APIs) to aid the processing of XML data.
* **HTML5:** It is a markup language used for structuring and presenting content on the World Wide Web. It is the fifth and current version of the HTML standard. It was published in October 2014 by the World Wide Web Consortium (W3C) to improve the language with support for the latest multimedia, while keeping it both easily readable by humans and consistently understood by computers and devices such as web browsers, parsers, etc. HTML5 includes detailed processing models to encourage more interoperable implementations; it extends, improves and rationalizes the markup available for documents, and introduces markup and application programming interfaces (APIs) for complex web applications. For the same reasons, HTML5 is also a candidate for cross-platform mobile applications, because it includes features designed with low-powered devices in mind.
* **JAVA:** Java is a general-purpose [computer-programming language](https://en.wikipedia.org/wiki/Programming_language" \o "Programming language) that is [concurrent](https://en.wikipedia.org/wiki/Concurrent_computing" \o "Concurrent computing), [class-based](https://en.wikipedia.org/wiki/Class-based_programming" \o "Class-based programming), [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming" \o "Object-oriented programming) and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "[write once, run anywhere](https://en.wikipedia.org/wiki/Write_once,_run_anywhere" \o "Write once, run anywhere)" (WORA), meaning that [compiled](https://en.wikipedia.org/wiki/Compiler" \o "Compiler) Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to [bytecode](https://en.wikipedia.org/wiki/Java_bytecode" \o "Java bytecode) that can run on any [Java virtual machine](https://en.wikipedia.org/wiki/Java_virtual_machine" \o "Java virtual machine) (JVM) regardless of [computer architecture](https://en.wikipedia.org/wiki/Computer_architecture" \o "Computer architecture). As of 2016, Java is one of the most [popular programming languages in use](https://en.wikipedia.org/wiki/Measuring_programming_language_popularity" \o "Measuring programming language popularity), particularly for client-server web applications, with a reported 9 million developers. Java was originally developed by [James Gosling](https://en.wikipedia.org/wiki/James_Gosling" \o "James Gosling) at [Sun Microsystems](https://en.wikipedia.org/wiki/Sun_Microsystems" \o "Sun Microsystems) (which has since been [acquired by Oracle Corporation](https://en.wikipedia.org/wiki/Sun_acquisition_by_Oracle" \o "Sun acquisition by Oracle)) and released in 1995 as a core component of Sun Microsystems' [Java platform](https://en.wikipedia.org/wiki/Java_(software_platform)" \o "Java (software platform)). The language derives much of its [syntax](https://en.wikipedia.org/wiki/Syntax_(programming_languages)" \o "Syntax (programming languages)) from [C](https://en.wikipedia.org/wiki/C_(programming_language)" \o "C (programming language)) and [C++](https://en.wikipedia.org/wiki/C%2B%2B" \o "C++), but it has fewer [low-level](https://en.wikipedia.org/wiki/Low-level_programming_language" \o "Low-level programming language) facilities than either of them. The original and [reference implementation](https://en.wikipedia.org/wiki/Reference_implementation" \o "Reference implementation) Java [compilers](https://en.wikipedia.org/wiki/Compiler" \o "Compiler), virtual machines, and [class libraries](https://en.wikipedia.org/wiki/Library_(computing)" \o "Library (computing)) were originally released by Sun under proprietary licenses.
* **CSS:** Cascading Style Sheets (CSS) is a [style sheet language](https://en.wikipedia.org/wiki/Style_sheet_language" \o "Style sheet language) used for describing the [presentation](https://en.wikipedia.org/wiki/Presentation_semantics" \o "Presentation semantics) of a document written in a [markup language](https://en.wikipedia.org/wiki/Markup_language" \o "Markup language). Although most often used to set the visual style of [web pages](https://en.wikipedia.org/wiki/Web_page" \o "Web page) and user interfaces written in [HTML](https://en.wikipedia.org/wiki/HTML" \o "HTML) and [XHTML](https://en.wikipedia.org/wiki/XHTML" \o "XHTML), the language can be applied to any [XML](https://en.wikipedia.org/wiki/XML" \o "XML) document and is applicable to rendering in [speech](https://en.wikipedia.org/wiki/Speech_synthesis" \o "Speech synthesis), or on other media. Along with HTML and [JavaScript](https://en.wikipedia.org/wiki/JavaScript" \o "JavaScript), CSS is a cornerstone technology used by most websites to create visually engaging webpages, user interfaces for [web applications](https://en.wikipedia.org/wiki/Web_applications" \o "Web applications), and user interfaces for many mobile applications.
* **JSON:** JavaScript Object Notation or JSON is an [open-standard](https://en.wikipedia.org/wiki/Open_standard" \o "Open standard) [file format](https://en.wikipedia.org/wiki/File_format" \o "File format) that uses [human-readable](https://en.wikipedia.org/wiki/Human-readable_medium" \o "Human-readable medium) text to transmit data objects consisting of [attribute–value pairs](https://en.wikipedia.org/wiki/Attribute%E2%80%93value_pair" \o "Attribute–value pair) and [array data types](https://en.wikipedia.org/wiki/Array_data_type" \o "Array data type) (or any other [serializable](https://en.wikipedia.org/wiki/Serialization" \o "Serialization) value). It is a very common data format used for [asynchronous](https://en.wikipedia.org/wiki/Asynchronous_I/O" \o "Asynchronous I/O) browser–server communication, including as a replacement for [XML](https://en.wikipedia.org/wiki/XML" \o "XML) in some [AJAX](https://en.wikipedia.org/wiki/Ajax_(programming)" \o "Ajax (programming))-style systems.

JSON is a [language-independent](https://en.wikipedia.org/wiki/Language-independent_specification" \o "Language-independent specification) data format. It was derived from [JavaScript](https://en.wikipedia.org/wiki/JavaScript" \o "JavaScript), but as of 2017 many [programming languages](https://en.wikipedia.org/wiki/Programming_language" \o "Programming language) include code to generate and [parse](https://en.wikipedia.org/wiki/Parsing" \o "Parsing) JSON-format data. The official Internet [media type](https://en.wikipedia.org/wiki/Media_type" \o "Media type) for JSON is application/json. JSON filenames use the extension .json.

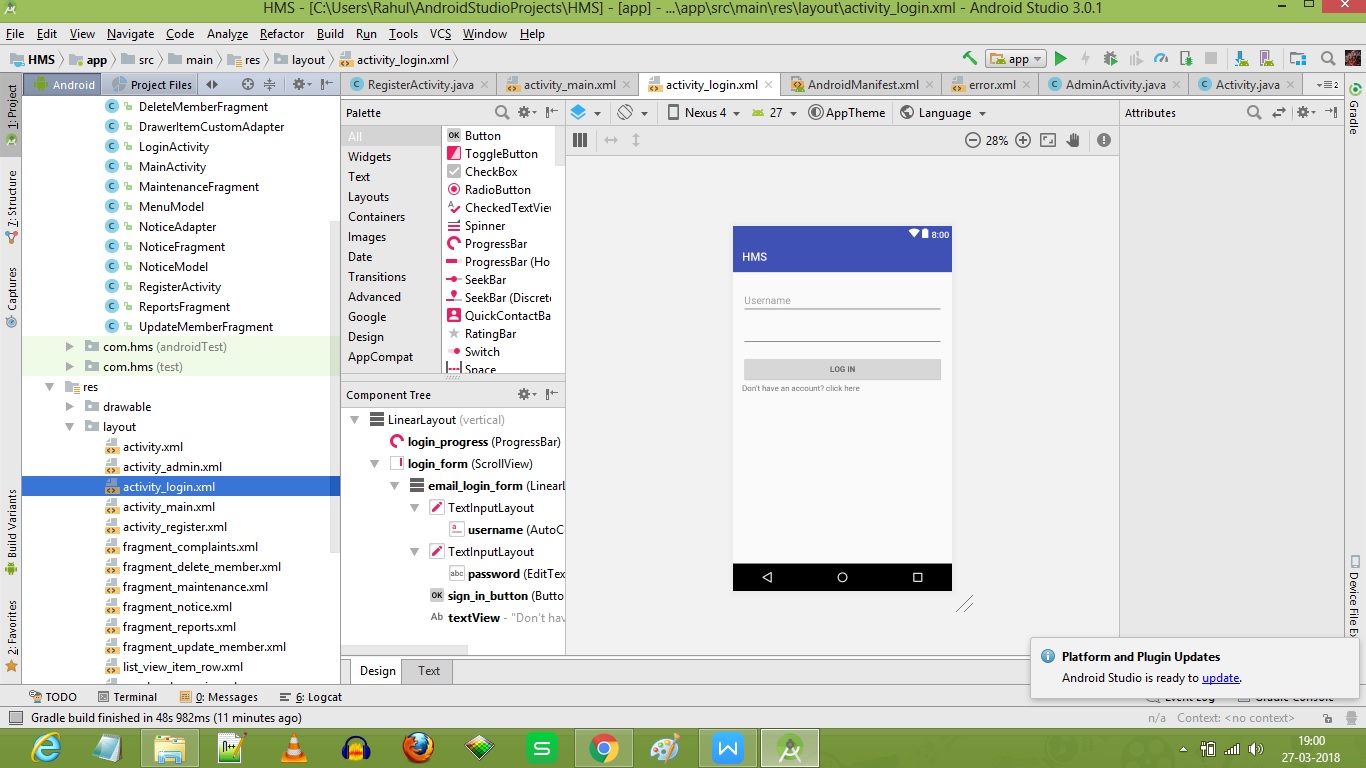
**2.2 Software Used**

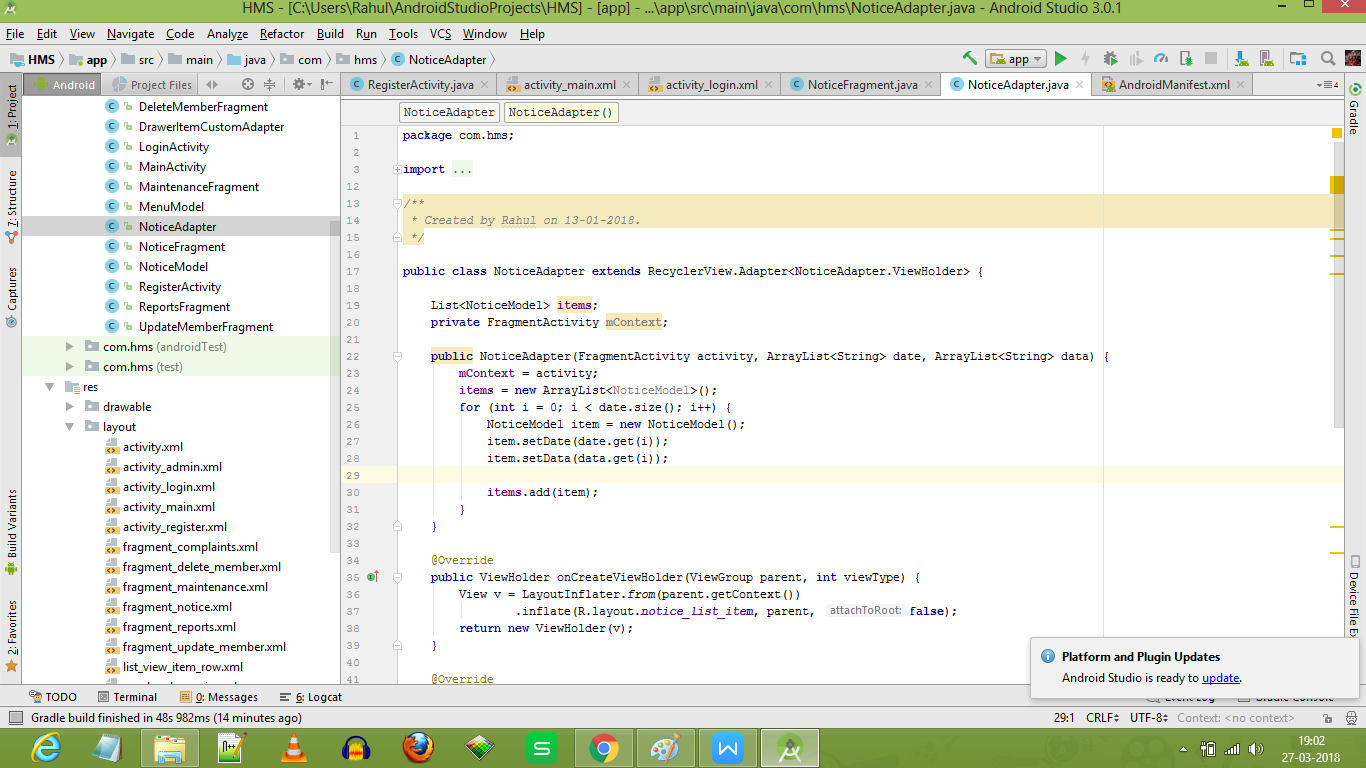
* **Android Studio 1.0.1:** Android Studio is the official Integrated Development Environment for Android application development, based on [IntelliJ IDEA.](https://www.jetbrains.com/idea/) On top of the capabilities expected from IntelliJ, Android Studio also offers:

1. Flexible Gradle-based build system
2. Build variants and multiple APK file generation.
3. Code templates to help you build common app features
4. Rich layout editor with support for drag and drop them editing

* Lint tools to catch performance, version compatibility, usability and other problems
* ProGuard and app-signing capabilities
* Built-in support for [Google Cloud Platform,](http://developers.google.com/cloud/devtools/android_studio_templates/) making it easy to integrate Google Cloud Messaging and App Engine.

**Graphical User Interface (XML): Android studio**





**Graphical User Interface (JAVA): Android studio**

**3 Feasibility Study**

* Feasibility study aims to objectively and rationally uncover the strengths and weakness of an existing business or proposed venture, opportunities and threats present in the environment, the resources required to carry through, and ultimately the prospects for success. In its simplest term the two criteria to judge feasibility are cost required and value to be attained.
* This project on maintaining a society through an Android application is highly feasible.
* During feasibility analysis for this project, following primary areas of interest are to be considered:
* **Operational feasibility :**
* Here, the admin and other members can easily access the application. The proposed application is completely feasible in terms of its operations. The system is self-explanatory and does not need any extra sophisticated training.
* The application is easy and efficient to use as it has a simple layout which helps user to access the application using any android device having version above 4.4.
* **Economical feasibility :**
* This evaluation handles the cost benefit analysis. It also checks whether the organization has the funds to develop the proposed system.
* This android application does not require enormous amount of money to be developed.
* **Technical feasibility :**
* The technical feasibility is frequently the most difficult area encountered. It centers on the existing system and to what extent it can support the proposed system. Selection of a suitable platform according to the organization’s need is very important. Hence the front end & back end technology is chosen based on:
* GUI.
* Flexibility.
* Platform independence.
* Easy and effective data handling.
* Security.
* Efficient data retrieval and maintenance.

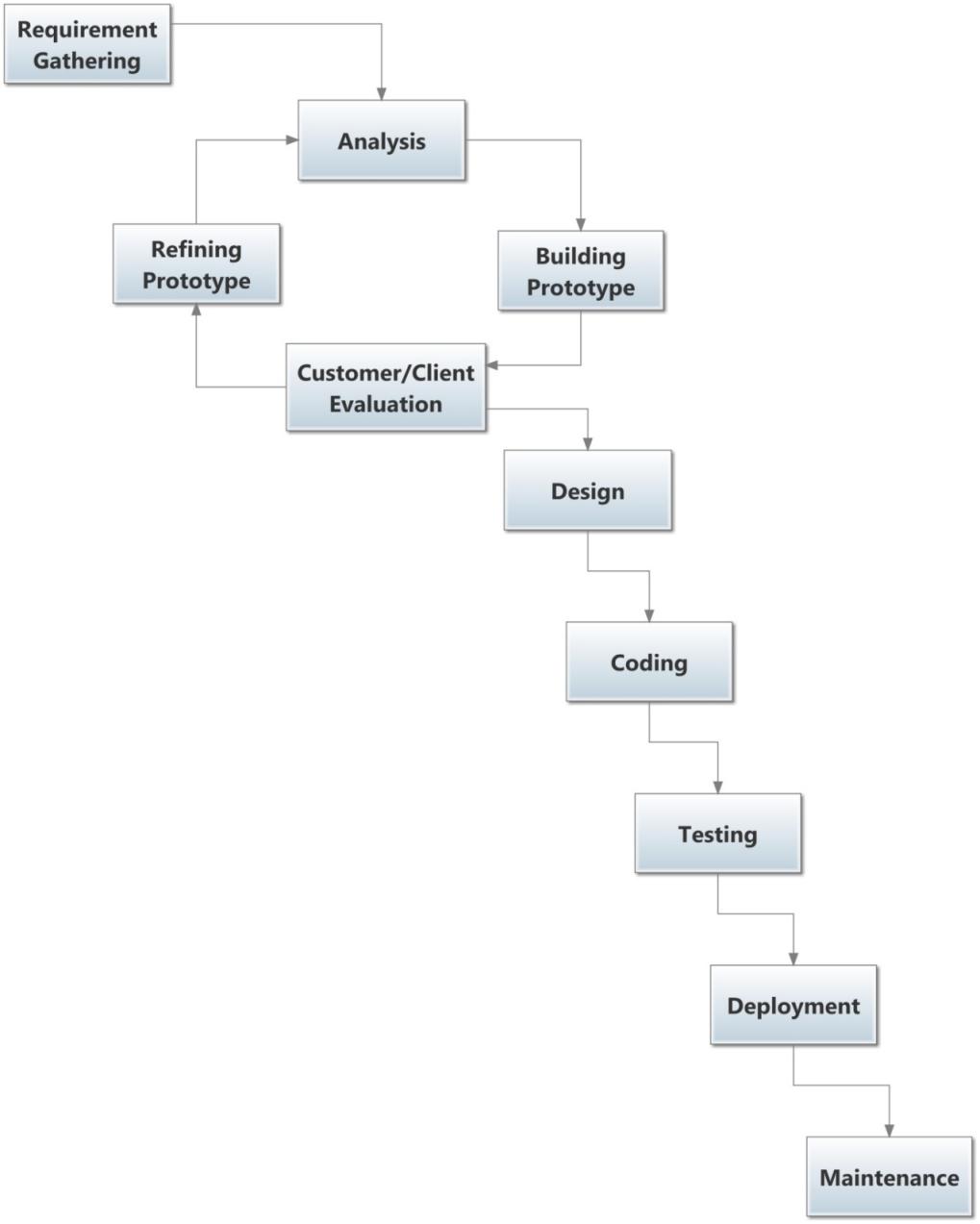
**CHAPTER 4:**

**METHODOLOGY USED**



**4.1 Model Used**

**Prototype Model**

****

**Prototype Model (Incremental prototyping model)**

The Software Prototyping refers to building software application prototypes which display the functionality of the product under development but may not actually hold the exact logic of the original software.

Software prototyping is becoming very popular as a software development model, as it enables to understand customer requirements at an early stage of development. It helps get valuable feedback from the customer and helps software designers and developers understand about what exactly is expected from the product under development.

* **What is Prototyping?**
* Prototype is a working model of software with some limited functionality.
* The prototype does not always hold the exact logic used in the actual software application and is an extra effort to be considered under effort estimation.
* Prototyping is used to allow the users evaluate developer proposals and try them out before implementation.
* It also helps to understand the requirements which are user specific and may not have been considered by the developer during product design.
* **WHAT IS INCREMENTAL PROTOTYPING :**
* Final product is built as separate components one at a time. There is one overall design for the system. It is partitioned into independent and smaller components. Final product is released as a series of products.
* **ADVANTAGES OF INCREMENTAL PROTOTYPING**
* Allows large systems to be installed in phases.
* Helps to avoid the delays between specification and implementation.
* Core system features are provided early. Users are not overwhelmed with a complex level of functionality in one go.
* Suitability and appropriateness of key requirements can be checked.
* Less essential features can be added later.
* **When to use Prototype model:**
* Prototype model should be used when the desired system needs to have a lot of interaction with the end users.
* Typically, online systems, web interfaces have a very high amount of interaction with end users, are best suited for Prototype model. It might take a while for a system to be built that allows ease of use and needs minimal training for the end user.
* Prototyping ensures that the end users constantly work with the system and provide a feedback which is incorporated in the prototype to result in a use-able system. They are excellent for designing good human computer interface systems.
* **Following is the stepwise approach to design a software prototype:**
* **Basic Requirement Identification:** This step involves understanding the very basic product requirements especially in terms of user interface. The more intricate details of the internal design and external aspects like performance and security can be ignored at this stage.

This phase has been completed with the help of fact finding techniques discussed in previous chapter.

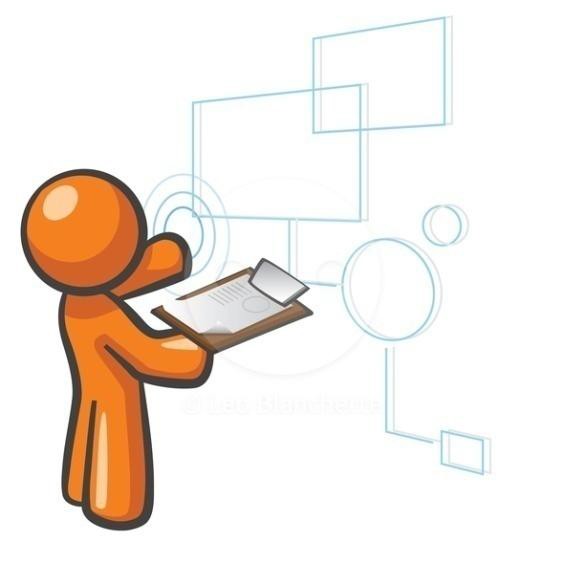
* **Developing the initial Prototype:** The initial Prototype is developed in this stage, where the very basic requirements are showcased and user interfaces are provided. These features may not exactly work in the same manner internally in the actual software developed and the workarounds are used to give the same look and feel to the customer in the prototype developed.

This involves graphical representation, wireframes, basic application built with Java and Firebase to help users in better understanding of what the final application would look like.

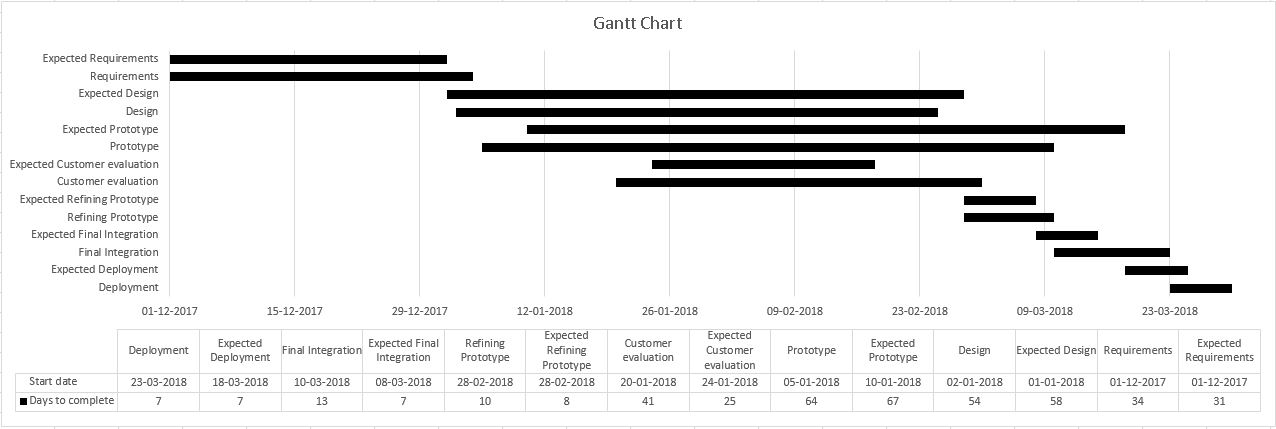
* **Review of the Prototype:** The prototype developed is then presented to the customer and other important stakeholders in the project. The feedback is collected in an organized manner and used for further enhancements in the product under development.
* **Revise and enhance the Prototype:** The feedback and the review comments are discussed during this stage and some negotiations happen with the customer based on factors like, time and budget constraints and technical feasibility of actual implementation. The changes accepted are again incorporated in the new Prototype developed and the cycle repeats until customer expectations are met.

**Chapter 5:**

**DIAGRAMS**

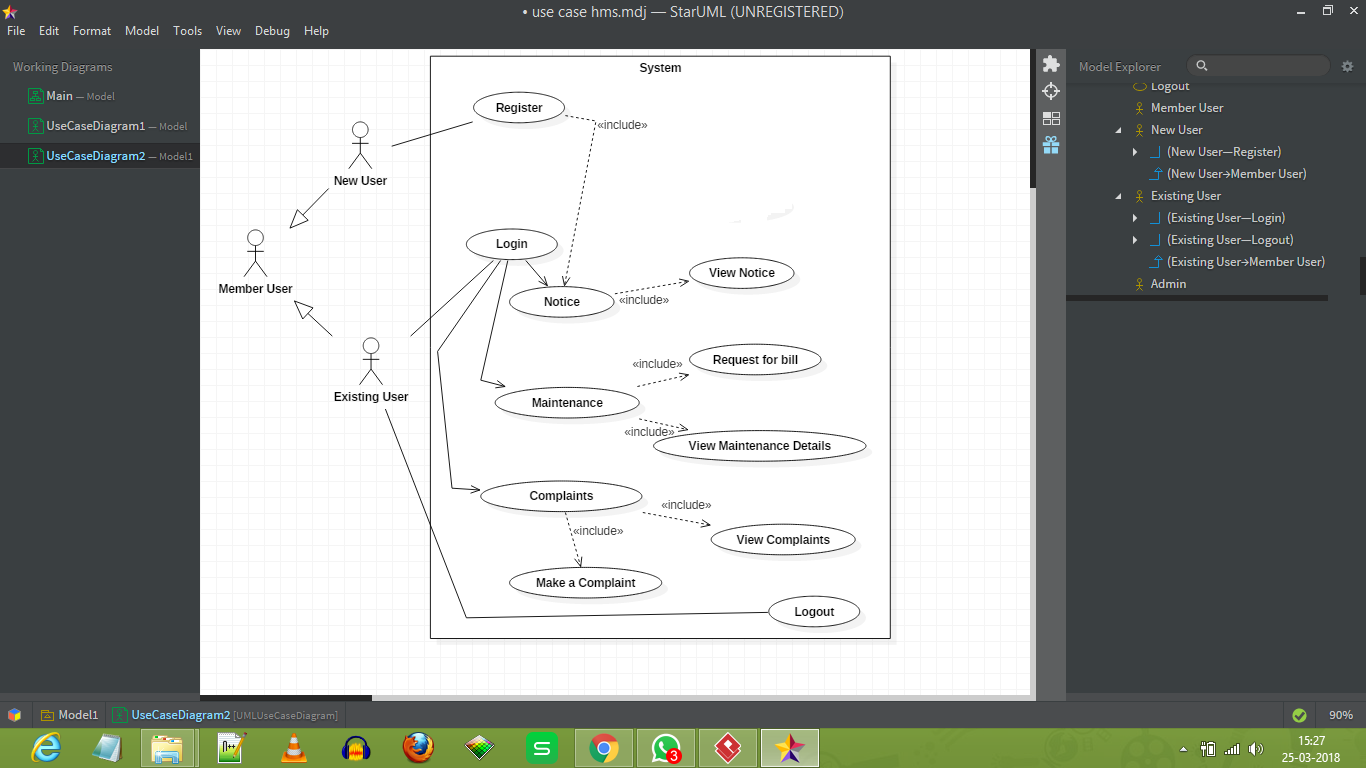


**5.1 Gantt chart**

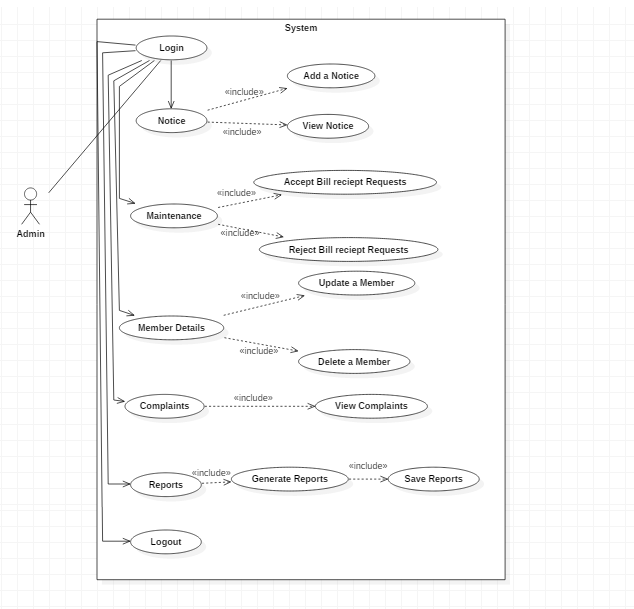
****

**5.2 Use Case Diagram**

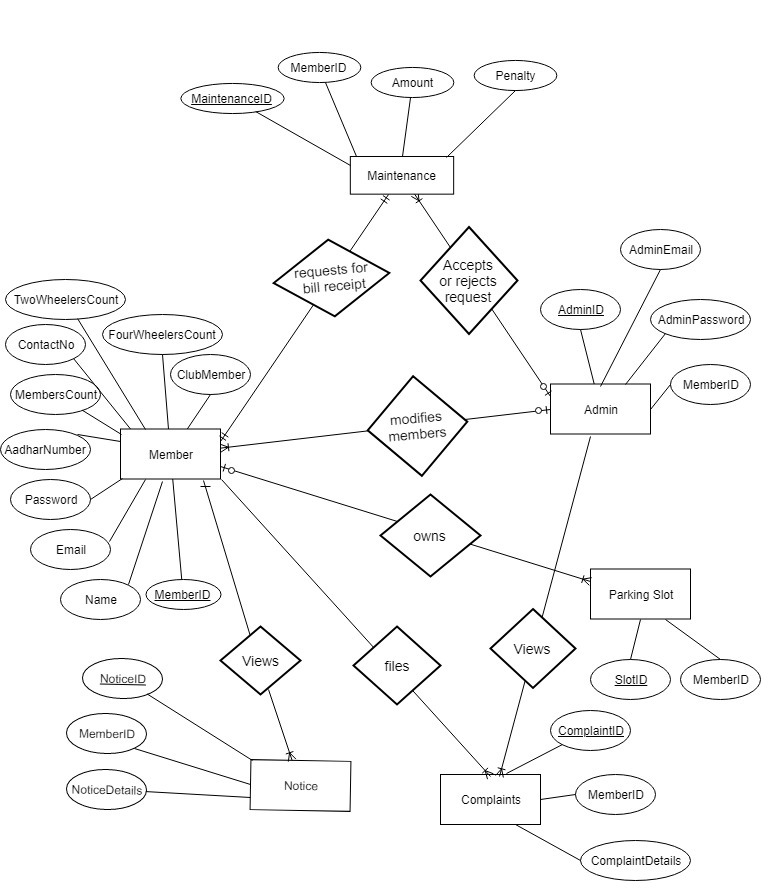
5.2.1 - User Module



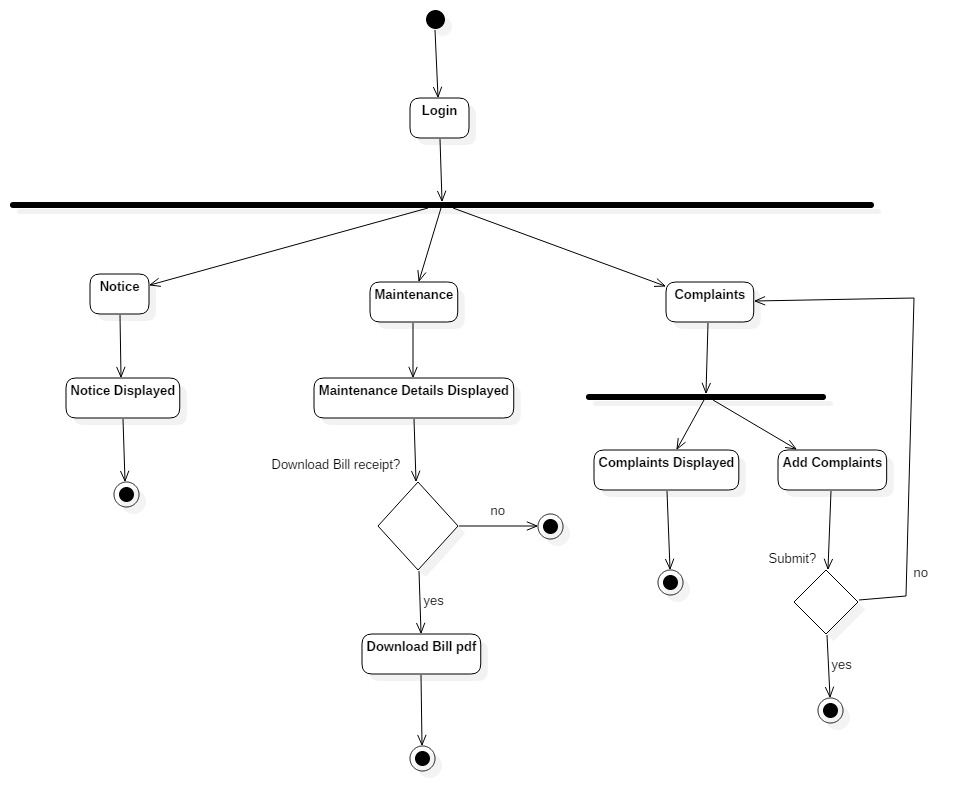
5.2.2 - Admin Module



**5.3 E-R Diagram**

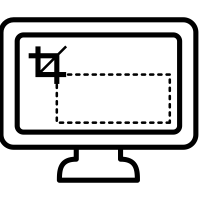
****

**5.4 Activity Diagram**

****

**CHAPTER 6:**

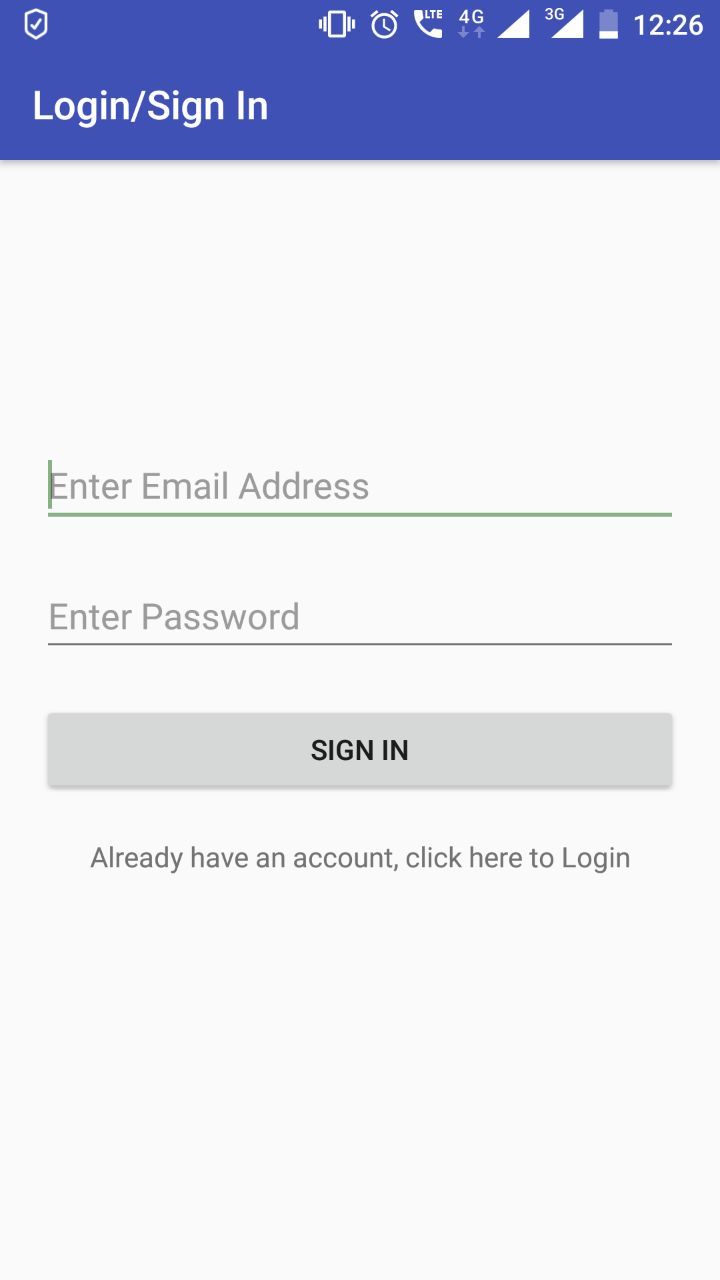
**SCREENSHOTS**

****

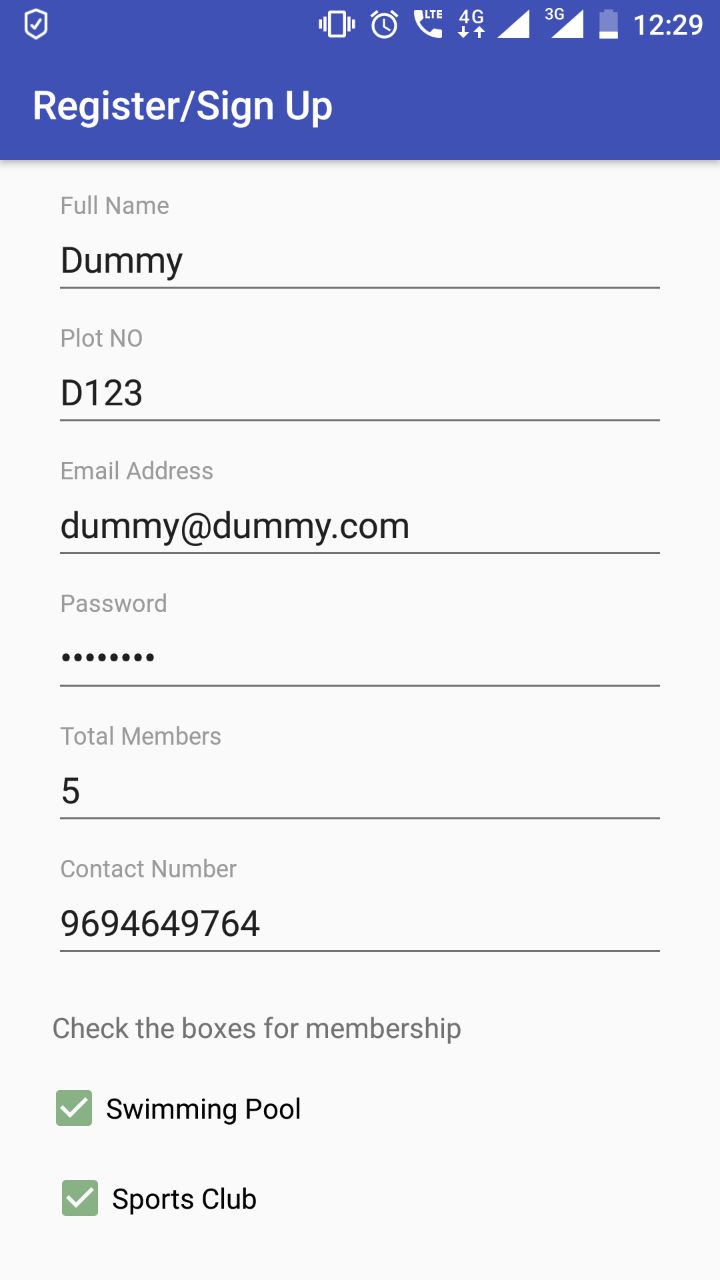
**Splashscreen:**



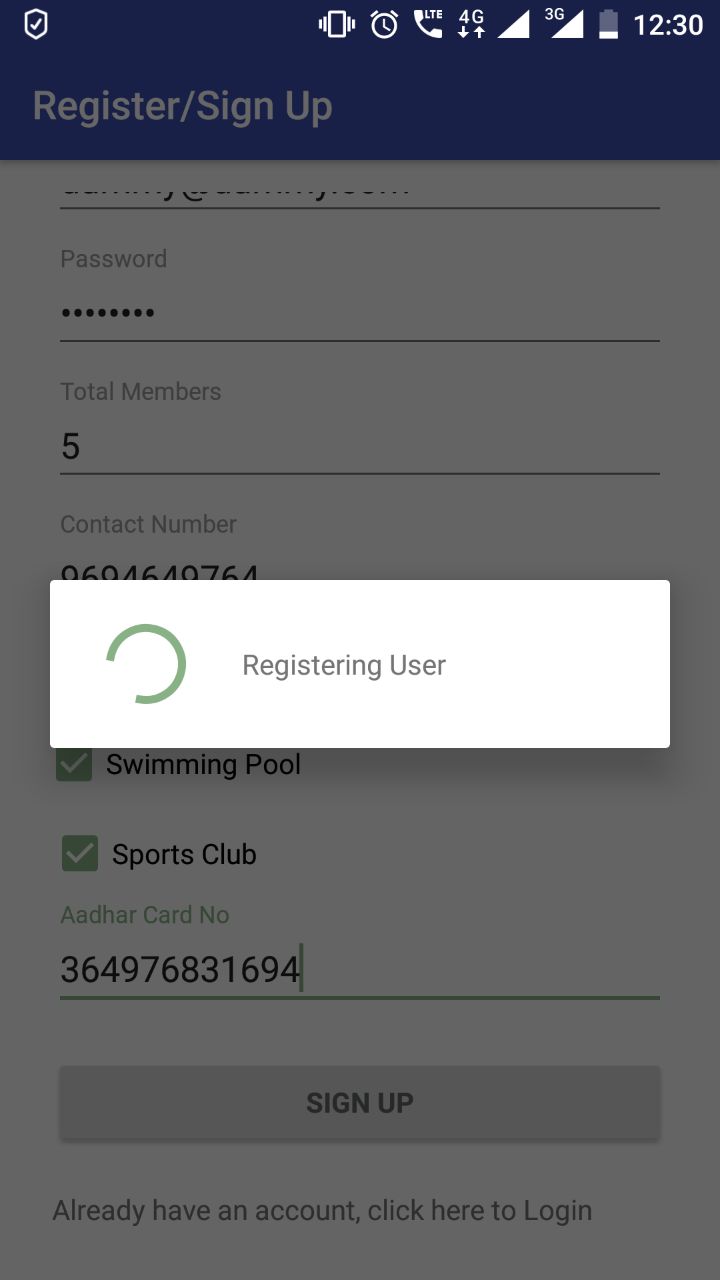
**Login**



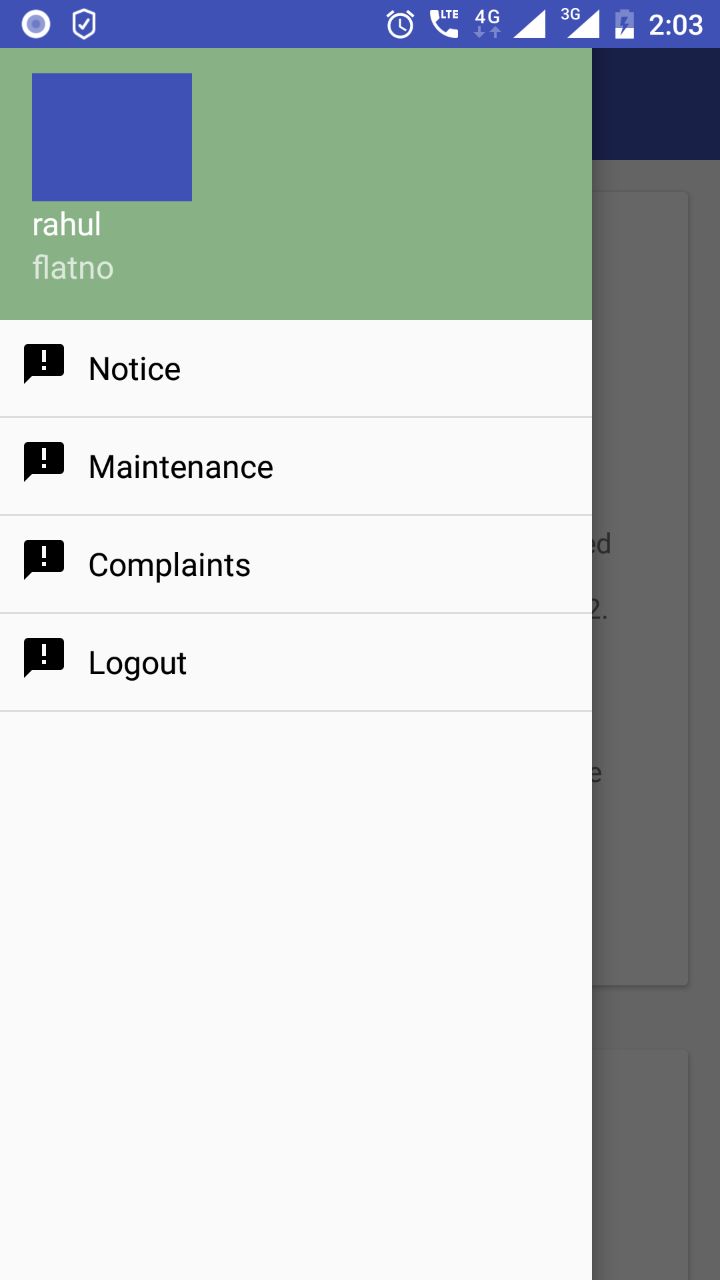
**Register**

****

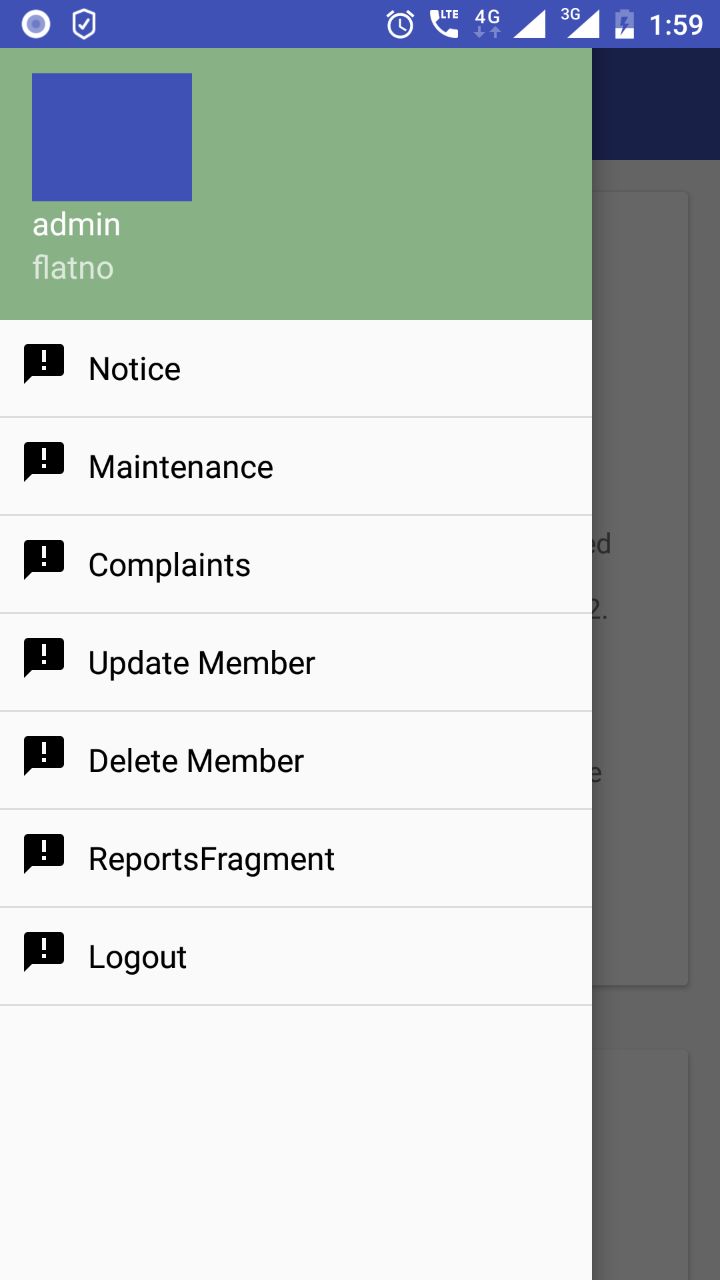
**Registering**

****

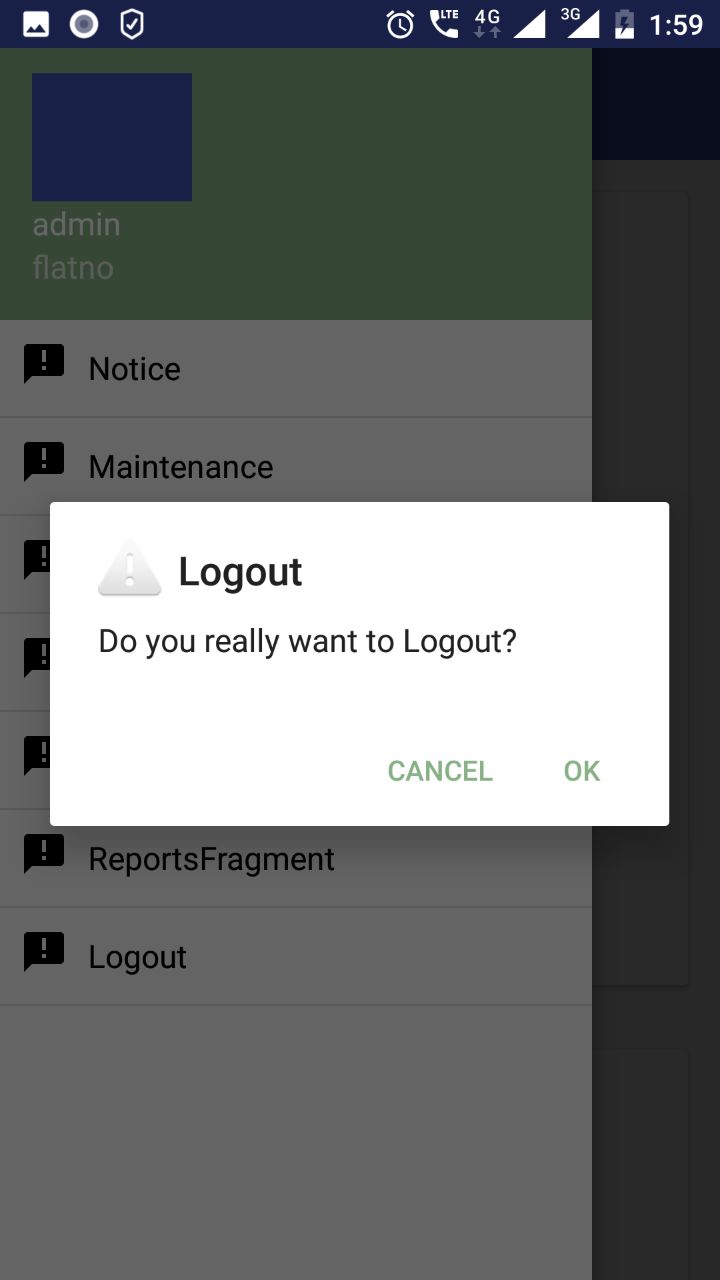
**Member User Home navigation drawer**

****

**Admin’s Home navigation drawer**

****

**Logout**

****

**CHAPTER 7:**

**CODE**



* **Login.xml**

*<?***xml version="1.0" encoding="utf-8"***?>*<**LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:paddingTop="130dp"  
 android:paddingLeft="10dp"  
 android:paddingBottom="10dp"  
 android:paddingRight="10dp"  
 android:orientation="vertical"  
 tools:context="com.hms.Login"**>  
  
 <**EditText  
 android:layout\_margin="10dp"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:id="@+id/emailUser"  
 android:inputType="textEmailAddress"  
 android:hint="@string/emailHint"** />  
  
 <**EditText  
 android:layout\_margin="10dp"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:id="@+id/passwordUser"  
 android:inputType="textPassword"  
 android:hint="@string/passwordHint"** />

<**Button  
 android:layout\_margin="10dp"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="Sign in"  
 android:id="@+id/buttonLogin"** />  
 <**TextView  
 android:layout\_margin="10dp"  
 android:textAlignment="center"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:id="@+id/switchToRegister"  
 android:text="@string/already\_have\_an\_account\_click\_here\_to\_login"** />  
  
</**LinearLayout**>

* **Register.xml**

*<?***xml version="1.0" encoding="utf-8"***?>*<**LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:gravity="center\_horizontal"  
 android:orientation="vertical"  
 android:paddingBottom="@dimen/activity\_vertical\_margin"  
 android:paddingLeft="@dimen/activity\_horizontal\_margin"  
 android:paddingRight="@dimen/activity\_horizontal\_margin"  
 android:paddingTop="@dimen/activity\_vertical\_margin"  
 tools:context="com.hms.RegisterActivity"**>  
  
 *<!-- Login progress -->* <**ProgressBar  
 android:id="@+id/register\_progress"  
 style="?android:attr/progressBarStyleLarge"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginBottom="8dp"  
 android:visibility="gone"** />  
  
 <**ScrollView  
 android:id="@+id/register\_form"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"**>  
  
 <**LinearLayout  
 android:id="@+id/email\_register\_form"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical"**>  
  
 <**android.support.design.widget.TextInputLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"**>  
  
 <**AutoCompleteTextView  
 android:layout\_margin="10dp"  
 android:id="@+id/fullname"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="@string/prompt\_fullname"  
 android:maxLines="1"  
 android:singleLine="true"** />  
  
 </**android.support.design.widget.TextInputLayout**>  
  
  
 <**android.support.design.widget.TextInputLayout  
 android:layout\_height="wrap\_content"  
 android:layout\_width="match\_parent"**>  
  
 <**AutoCompleteTextView  
 android:layout\_margin="10dp"  
 android:id="@+id/plotno"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="@string/prompt\_plotno"  
 android:maxLines="1"  
 android:singleLine="true"** />  
  
 </**android.support.design.widget.TextInputLayout**>  
  
 <**android.support.design.widget.TextInputLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"**>  
  
 <**AutoCompleteTextView  
 android:layout\_margin="10dp"  
 android:id="@+id/emailRegister"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="@string/prompt\_email"  
 android:maxLines="1"  
 android:singleLine="true"** />  
  
 </**android.support.design.widget.TextInputLayout**>  
  
 <**android.support.design.widget.TextInputLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"**>  
  
 <**EditText  
 android:layout\_margin="10dp"  
 android:id="@+id/password"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="@string/prompt\_password"  
 android:imeActionId="6"  
 android:imeActionLabel="@string/action\_sign\_in\_short"  
 android:imeOptions="actionUnspecified"  
 android:inputType="textPassword"  
 android:maxLines="1"  
 android:singleLine="true"** />  
  
 </**android.support.design.widget.TextInputLayout**>  
  
 <**android.support.design.widget.TextInputLayout  
 android:layout\_height="wrap\_content"  
 android:layout\_width="match\_parent"**>  
  
 <**AutoCompleteTextView  
 android:layout\_margin="10dp"  
 android:id="@+id/totalmembers"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="@string/prompt\_Total\_members"  
 android:inputType="number"  
 android:maxLines="1"  
 android:singleLine="true"  
 android:maxLength="2"** />  
  
 </**android.support.design.widget.TextInputLayout**>  
  
 <**android.support.design.widget.TextInputLayout  
 android:layout\_height="wrap\_content"  
 android:layout\_width="match\_parent"**>  
  
 <**AutoCompleteTextView  
 android:layout\_margin="10dp"  
 android:id="@+id/contactno"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="@string/prompt\_contactno"  
 android:inputType="phone"  
 android:maxLines="1"  
 android:singleLine="true"  
 android:maxLength="10"** />  
  
 </**android.support.design.widget.TextInputLayout**>  
  
  
 <**TextView  
 android:layout\_margin="10dp"  
 android:id="@+id/cbTextView"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="@string/check\_the\_boxes\_for\_membership"** />  
  
 <**CheckBox  
 android:layout\_margin="5dp"  
 android:id="@+id/swimmingpoolcb"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="@string/swimming\_pool"** />  
  
 <**CheckBox  
 android:layout\_margin="8dp"  
 android:id="@+id/sportsclubcb"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="@string/sports\_club"** />  
  
 <**android.support.design.widget.TextInputLayout  
 android:layout\_height="wrap\_content"  
 android:layout\_width="match\_parent"**>  
  
 <**AutoCompleteTextView  
 android:layout\_margin="10dp"  
 android:id="@+id/aadharno"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="@string/prompt\_Aadhar\_details"  
 android:inputType="number"  
 android:maxLines="1"  
 android:singleLine="true"  
 android:maxLength="16"**/>  
  
 </**android.support.design.widget.TextInputLayout**>  
  
 <**Button  
 android:layout\_margin="10dp"  
 android:id="@+id/sign\_up\_register"  
 style="?android:textAppearanceSmall"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="16dp"  
 android:text="@string/action\_Sign\_up"  
 android:textStyle="bold"** />  
  
 <**TextView  
 android:layout\_margin="10dp"  
 android:id="@+id/labeltextview"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="@string/already\_have\_an\_account\_click\_here\_to\_login"** />  
  
  
 </**LinearLayout**>  
  
 </**ScrollView**>  
</**LinearLayout**>

* **Activity\_Main.xml**

*<?***xml version="1.0" encoding="utf-8"***?>*<**android.support.v4.widget.DrawerLayout  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 android:id="@+id/drawer\_layout"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:fitsSystemWindows="true"  
 tools:openDrawer="start"**>  
  
 <**LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical"**>  
  
 <**android.support.v7.widget.Toolbar  
 android:id="@+id/toolbar"  
 android:layout\_width="match\_parent"  
 android:layout\_height="?attr/actionBarSize"  
 android:background="@color/colorPrimary"  
 android:popupTheme="@style/AppTheme.PopupOverlay"  
 android:theme="@style/ThemeOverlay.AppCompat.Dark"**/>  
  
 <**FrameLayout  
 android:id="@+id/activity\_main"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"**>  
  
 </**FrameLayout**>  
  
 </**LinearLayout**>  
  
 <**android.support.design.widget.NavigationView  
 android:id="@+id/nav\_view"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:layout\_gravity="start"  
 android:fitsSystemWindows="true"  
 app:headerLayout="@layout/nav\_header\_main"**>  
  
 <**ListView  
 android:id="@+id/menu\_list"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:layout\_marginTop="@dimen/nav\_header\_height"**>  
  
 </**ListView**>  
 </**android.support.design.widget.NavigationView**>  
  
</**android.support.v4.widget.DrawerLayout**>

* **Login.java - LoginUser Method**

**public void** LoginUser() {  
 **final** String email = **emailUser**.getText().toString().trim();  
 **final** String password = **passwordUser**.getText().toString().trim();  
  
 **if** (TextUtils.*isEmpty*(email)) {  
 Toast.*makeText*(**this**, **"Email cannot be blank"**, Toast.***LENGTH\_LONG***).show();  
 *//It will stop function execution* **emailUser**.setError(**"Please enter email"**);  
 **return**;  
 }  
 **if** (TextUtils.*isEmpty*(password)) {  
 Toast.*makeText*(**this**, **"Password cannot be blank"**, Toast.***LENGTH\_LONG***).show();  
 **passwordUser**.setError(**"Please enter password"**);  
 **return**;  
 }  
 **progressDialog**.setMessage(**"Logging in"**);  
 **progressDialog**.show();  
 **if** (CommonMethods.*isOnline*(getBaseContext())) {  
 *//validates user* **mAuth**.signInWithEmailAndPassword(email, password).addOnCompleteListener(**this**, **new** OnCompleteListener<AuthResult>() {  
 @Override  
 **public void** onComplete(@NonNull Task<AuthResult> task) {  
  
 **if** (task.isSuccessful() == **true**) {  
  
 **if**(email.equals(**"deepak.jain186@gmail.com"**) && (password.equals(**"qwertyuiop"**)))  
 {  
 **progressDialog**.dismiss();  
 finish();  
 startActivity(**new** Intent(Login.**this**,MaintenanceFragment.**class**));  
 }  
 **else** {  
 **progressDialog**.dismiss();  
 finish();  
 startActivity(**new** Intent(Login.**this**, MainActivity.**class**));  
 }  
 } **else if** (task.isSuccessful() == **false**) {  
 **progressDialog**.dismiss();  
 Toast.*makeText*(Login.**this**, **"Invalid emailid or password."**, Toast.***LENGTH\_SHORT***).show();  
 **passwordUser**.getText().clear();  
 **emailUser**.getText().clear();  
 **emailUser**.requestFocus();  
 **return**;  
 }  
 }  
 });  
 }  
 **else** {  
 **progressDialog**.dismiss();  
 Toast.*makeText*(**this**, **"Please check your internet connection"**, Toast.***LENGTH\_LONG***).show();  
 }  
}

* **Register.java - OnClick method**

**public void** onClick(View view) {  
 **if** (view == **sign\_up**) {  
 *//fetching data* **final** String Name = **name**.getText().toString().trim();  
 **final** String emailId = **email**.getText().toString().trim();  
 **final** String pass = **password**.getText().toString().trim();  
 **final** String plot = **plot\_no**.getText().toString().trim();  
 **final** String countMembers = **number\_of\_members**.getText().toString().trim();  
 **final** String contact = **contact\_no**.getText().toString();  
 **final** String aadhar = **aadhar\_card\_no**.getText().toString().trim();  
 **final** String checkSwim, checkSports;  
 **if** (**swimming\_pool**.isChecked()) {  
 checkSwim = **"Yes"**;  
 } **else** {  
 checkSwim = **"No"**;  
 }  
 **if** (**sports\_club**.isChecked()) {  
 checkSports = **"Yes"**;  
 } **else** {  
 checkSports = **"No"**;  
 }  
 **if** (TextUtils.*isEmpty*(Name)) {  
 Toast.*makeText*(**this**, **"Name cannot be blank"**, Toast.***LENGTH\_LONG***).show();  
 **return**;  
 } **else if** (TextUtils.*isEmpty*(plot)) {  
 Toast.*makeText*(**this**, **"Plot Number cannot be blank"**, Toast.***LENGTH\_LONG***).show();  
 **return**;  
 } **else if** (TextUtils.*isEmpty*(emailId)) {  
 Toast.*makeText*(**this**, **"Email ID cannot be blank"**, Toast.***LENGTH\_LONG***).show();  
 **return**;  
 } **else if** (TextUtils.*isEmpty*(pass)) {  
 Toast.*makeText*(**this**, **"Password cannot be blank"**, Toast.***LENGTH\_LONG***).show();  
 **return**;  
 } **else if** (TextUtils.*isEmpty*(countMembers)) {  
 Toast.*makeText*(**this**, **"Number of members cannot be blank"**, Toast.***LENGTH\_LONG***).show();  
 **return**;  
 } **else if** (TextUtils.*isEmpty*(contact)) {  
 Toast.*makeText*(**this**, **"Contact Number cannot be blank"**, Toast.***LENGTH\_LONG***).show();  
 **return**;  
 } **else if** (TextUtils.*isEmpty*(aadhar)) {  
 Toast.*makeText*(**this**, **"Aadhar cannot be blank"**, Toast.***LENGTH\_LONG***).show();  
 **return**;  
 }  
 **progressDialog**.setMessage(**"Registering User"**);  
 **progressDialog**.show();  
 **if** (CommonMethods.*isOnline*(getBaseContext())) {  
 **mAuth**.createUserWithEmailAndPassword(emailId, pass).addOnCompleteListener(**this**, **new** OnCompleteListener<AuthResult>() {  
 @Override  
 **public void** onComplete(@NonNull Task<AuthResult> task) {  
 **if** (task.isSuccessful() == **true**) {  
 **progressDialog**.dismiss();  
 DatabaseQueries dq = **new** DatabaseQueries();  
 dq.*updateRegister*(Name, emailId, pass, plot, countMembers, contact, aadhar, checkSwim, checkSports);  
 finish();  
 startActivity(**new** Intent(RegisterActivity.**this**, MainActivity.**class**));  
 }  
 **else if** (task.isSuccessful() == **false**)  
 {  
 **progressDialog**.dismiss();  
 Toast.*makeText*(RegisterActivity.**this**, **"PLease enter valid data."**, Toast.***LENGTH\_LONG***).show();  
 **name**.getText().clear();  
 **plot\_no**.getText().clear();  
 **email**.getText().clear();  
 **password**.getText().clear();  
 **number\_of\_members**.getText().clear();  
 **contact\_no**.getText().clear();  
 **aadhar\_card\_no**.getText().clear();  
 **name**.requestFocus();  
 **return**;  
 }  
 }  
 });  
  
 } **else if** (view == **loginLink**) {  
 finish();  
 startActivity(**new** Intent(RegisterActivity.**this**, Login.**class**));  
 }  
 } **else** {  
 **progressDialog**.dismiss();  
 Toast.*makeText*(**this**, **"Please check your internet connection"**, Toast.***LENGTH\_LONG***).show();  
 }  
}

**Chapter 8:**

**TESTING**



**Introduction**

The code is tested at various levels in software testing. Unit, system and user acceptance testing are often performed. This is a grey area as many different opinions exist as to what the stages of testing are and how much if any iteration occurs.

**Types of testing:**

* + Data set testing.
  + Unit testing.
  + System testing.
  + Integration testing.
  + Black Box testing.
* White Box testing
* Regression testing.
* Automation testing.
* User acceptance testing.
* Performance testing.

###### The two basic approaches to testing are:

* + - Black Box or Functional Testing.
    - White Box or Structural Testing.

### B l a c k B o x Testing

* + BBT is related with input and output and not related with internal structure of the program.
  + In BBT it is checked if some input is given than whether specific output is produce by the program or not.
  + The various sets of input test cases are prepared and applied on a program corresponding output are verified.
  + This type of testing is done by Test Engineers.
* **Types of White Box testing conducted are:**
* **Statement Coverage:** Is a method of validating that each and every line of code is executed at least once.
* **Branch Coverage:** Branch Coverage is a testing method which when executed ensures that each branch from each decision point is executed.
* **Path Coverage:** Path coverage is used to test the complex code snippets, which basically involves loop statements or combination of loops and decision statements.

**Black Box Testing:**

Technique: Boundary Value Analysis (BVA)

Member user Sign up Activity

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sr. No | Test Case | Input | Expected Output | Actual output | Result |
| 1 | Enter Name | Rahul | It should accept | It accepted | Pass |
| 2 | Enter Name | Deepak@123 | It should not accept | It did not accept | Pass |
| 3 | Enter Email Id | Dieepak@gmail.com | It should accept | It accepted | Pass |
| 4 | Enter Email Id | Rahulgmail.com | It should not accept | It did not accept | Pass |
| 5 | Enter Contact number | RDE@123 | It should not accept | It did not accept | Pass |
| 6 | Enter Contact number | 8882634523 | It should accept | It accepted | Pass |
| 7 | Enter Aadhar number | 459515235566 | It should not accept | It did not accept | Pass |

**Member user Module:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sr. No | Test Case | Input | Expected Output | Actual Output | Result |
| 1 | Login | Email and password | Checks with database and logs  in | Logs in successfully | Pass |
| 2 | View Notice | Click on notice in navigation drawer | Notices should be viewed | Notices are viewed | Pass |
| 3 | View maintenance details | Click on Maintenance in navigation drawer | Maintenance details should appear | Maintenance details are appeared | Pass |
| 4 | Request for maintenance bill receipt | Click on “Ask for receipt” toggle button | Request should be sent to admin | Request sent | Pass |
| 5 | Add a complaint | Click on Complaints in navigation drawer and type a complain and click on send | Complaint should be sent and seen in the complaints section | Complaint sent and displayed | Pass |
| 6 | View Complaints | Click on Complaints in navigation drawer | Complaints should be viewed | Complaints are viewed | Pass |
| 7 | Logout | Click on Logout | Checks with database and logs  out | Logs out successfully | Pass |

**Admin Module:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sr. No | Test Case | Input | Expected Output | Actual Output | Result |
| 1 | Login | Email and password | Checks with database and logs  in | Logs in successfully | Pass |
| 2 | View Notice | Click on notice in navigation drawer | Notices should be viewed | Notices are viewed | Pass |
| 3 | Add a notice | Click on Notice in navigation drawer and type a notice and click on send | Notice should be sent and seen on the Notice Fragment | Notice sent and displayed | Pass |
| 3 | View maintenance details | Click on Maintenance in navigation drawer | Maintenance details should appear | Maintenance details are appeared | Pass |
| 4 | Accept or reject request for maintenance bill receipt | Click on accept or reject button for each entry | Request should be accepted or rejected accordingly | Response sent | Pass |
| 6 | View Complaints | Click on Complaints in navigation drawer | Complaints should be viewed | Complaints are viewed | Pass |
| 7 | Generate Reports | Enter Filters for reports and click on generate | Reports should be generated | Reports are generated | Pass |
| 8 | View members | Click on Members in navigation drawer | Members details should appear | Members details appeared | Pass |
| 9 | Logout | Click on Logout | Checks with database and logs  out | Logs out successfully | Pass |

**White Box Testing:**

**Technique: Cyclomatic Complexity**

**RegistrationActivity.java**

Cyclomatic Complexity=E-N+P

Where,

E=number of edges in the flow graph.

N=number of nodes in the flow graph.

P=number of nodes that have exit points.

Cyclomatic Complexity for user validate function.

Here, E=36, N=35, p=2

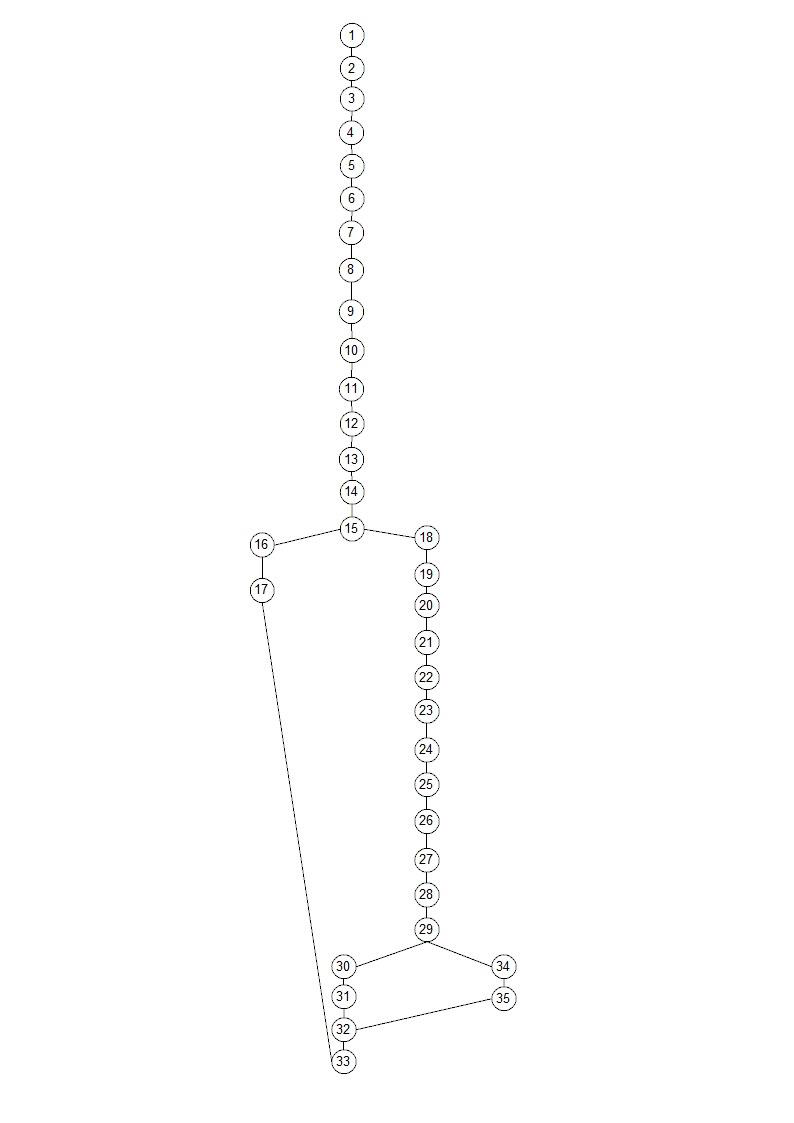
Therefore, cyclomatic complexity = 36-35+2 = 3

Complexity=3

Basic paths to be tested are

Path 1: 1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-33

Path 2: 1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-18-19-20-21-22-23-24-25-26-27-28-29-34-35- 32-33

Path 3: 1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-18-19-20-21-22-23-24-25-26-27-28-29-30-31-32-33 

**Figure: Control Flow Diagram/Graph**

**Chapter 9:**

**SYSTEM MAINTENANCE**

**AND USER MANUAL**



**9.1 System Maintenance:**

Once the application is deployed, then maintenance phase starts. Application requires maintenance because there are some residual errors remaining in the system that must be removed as they are discovered.

In Maintenance, we will be doing the following:

* Fixing bugs if at all anything found during actual working.
* Any minor changes that is required when the user working with it will be done.
* Periodic checking of software at regular intervals.
* Make better use of existing tools and techniques.
* Online technical support at any instance of time.

**9.2 User Manual**

1. User (Society Member) can register his or her self by filling registration form.

2. User (Admin) can put Notices after logging into application.

3. User (Society Member) can file complaints after logging into application under complaints section.

3. Admin and society member users both can view all notices and complains.

4. User (Society Member) can view his or her maintenance details.

5. User (admin) can generate reports.

6. User (Society Member) can download or print maintenance bill receipt(iff the bill is paid).

**CHAPTER 10:**

**CONCLUSIONS**



* 1. **10.1 Limitations**
* The limitations and pitfalls of such systems are their online presence. High security is needed to protect the data. Personal information protection is a serious responsibility.
* Other limitation is the availability of internet connection. As the data managing part isn’t offline the user needs a stable internet connection to connect to the server properly and to be able to modify/see his/her data.
* Online Payment service could not be implemented as it would not be economically a feasible feature.
* SMS feature to the members for reminder about the maintenance due dates could not be added as of now considering the economic feasibility.
  1. **10.2 Future Scope**

We believe in the one liner “There is always an inch more to Build”. This means, that the application can be enhanced further more. We were bound by certain criteria such as time, cost, syllabus & other things like professional talent and work experiences.

But positively, without these bounding criteria, We would like to enhance this software project and empower it with many more functionalities.

My project is a sincere effort to balance all the current limiting criteria but we hope that we can add below functionalities to my current project at the earliest.

* Some few improvements over the look and feel could be added in future.
* Extending to Apple iOS.
* High security is needed to protect the data. We look forward to adding security features.
* Online payment service along with message service could be added in future.
* Supporting video calls to discuss the problems with doctors.

**10.3 Bibliography**

* 1. Websites:
  + <https://www.w3schools.com/>
  + <https://www.androidhive.info/category/app/>
  + <http://stackoverflow.com/>
* <https://www.androidhive.info/category/firebase/>

Videos:

* Lynda - Bootstrap 3 Essential Training
* Prudhvi Raj Kumar - Android Tutorials
  1. **10.4 References**
  2. Websites:
* <https://www.tutorialspoint.com/index.htm>
* [https://www.androidhive.info](https://www.androidhive.info/category/firebase/)
* <https://en.wikipedia.org/wiki/Main_Page>