Table of Contents

CONTENTS **PAGE NO**

ABSTRACT

Smart scheduling for project management to Schedule and manage task/work/events quickly by checking status in a single view. The proposed smart scheduling can create the dynamic/multiple task with start, end date and alter every task that has start and completion dates that take the task interdependencies into account. A milestone is accomplished when one or more work products from an engineering task have passed quality review. The software implemented in mobile app. It takes care of the activities that are scheduled and helps in the management of daily tasks of the employees in the organization. In a company where the hierarchy of employees spans over thousand managing the work with them is a difficult job. And in an environment where number of jobs is done simultaneously picking the right person for the job is also difficult task, as you are not aware of their availability. This application is designed for such an environment where the work is divided into group of employees and during the course of division the employees are selected to be part of the work in hand.

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

1.1 OVERVIEW:

The Online Task Management System is used to automate the process of admin and user management and user task. The work is under observation of the higher authorities. The project provides online platform to accomplish day to day task in an organization. The proposed software will help the employee and admin to communicate with each other. The system easily assigns tasks so as to avoid all the time-consuming and unnecessary meetings. Admin can periodically share all the details regarding the task with the employee. The management of assignment or task is easy from both the ends . The admin is able to assign task to the employee. This software provides facilities to assign task, send message, send and view notification to the users of this software.

1.2 PROBLEM DEFINITION

The problem definition for designing this system is to preserve the data of employee, to have an easy controlling of employees, to divide tasks and access control of employees, to use technology for accurate and timely processing by providing privacy and full authority access.

1.3 PURPOSE

In a company where the hierarchy of employees spans over thousand managing the work with them is a difficult job. And in an environment where number of jobs is done simultaneously picking the right person for the job is also difficult task, as you are not aware of their availability.

This application is designed for such an environment where the work is divided into group of employees and during the course of division the employees are selected to be part of the work in hand.

1.4 SCOPE

This system will be focusing on developing of a web-based employee management system that would suit the organisation. This project is helpful in maintaining the employee's record, calculating the status for each employee and it also focuses on each employee's task. There is also a possibility of checking status report at any time so that it doesn't lead to any miscalculation.

CHAPTER 2

LITERATURE SURVEY

A literature Survey is a text of a scholarly paper, which includes the current knowledge including substantive findings as well as theoretical and methodological contributions to a particular topic. Literature reviews are secondary sources and do not report new or original experimental work. Most often associated with academic-oriented literature, such reviews are found in academic journals, and are not to be confused with books reviews C.A narrow-scope literature review may be included as a part of a peer-reviewed journal article presenting new research, serving to situate the current study within the body of the relevant literature and to provide context for the reader .In such a case ,the review usually precedes the methodology and results sections of the work. Producing a literature review may also be part of graduate and postgraduate student work, including in the preparation of a thesis, dissertation, or a journal article.

2.1 REVIEW OF RELATED RESEARCH PAPERS

TITLE : Using a Risk Breakdown Structure in project management

AUTHOR : D. A. Hillson

YEAR : 2013

DESCRIPTION

Risk identification often produces nothing more than a long list of risks, which can be hard to understand or manage. The list can be prioritised to determine which risks should be addressed first, but this does not provide any insight into the structure of risk on the project. Traditional qualitative assessment cannot indicate those areas of the project which require special attention, or expose recurring themes, concentrations of risk, or 'hot-spots' of risk exposure. The best way to deal with a large amount of data is to structure the information to aid comprehension. For risk management, this can be achieved with a Risk Breakdown Structure (RBS) a hierarchical structuring of risks on the project. The RBS can assist in understanding the distribution of risk on a project or across a business, aiding effective risk management. Just as the Work Breakdown Structure (WBS) is an important tool for projects because it scopes and defines the work, so the RBS can be an invaluable aid in understanding risk. The WBS forms the basis for many aspects of the project management process; similarly, the RBS can be used to structure and guide the risk management process. This paper presents the concept of the RBS, and gives a number of examples drawn from different project types and industries. Although not necessarily based in FM, the concepts and experience can be applied to any project. The benefits of using the RBS are then outlined,

including as an aid for risk identification or risk assessment, comparison of projects, providing a framework for cross-project risk reporting, and structuring lessons to be learned for future projects. This paper shows how to use the RBS to gain these benefits.

TITLE : Organisational Complexity and Perceptions of Task

AUTHOR : S. McKenna

YEAR : 2013

DESCRIPTION

This paper explores the idea that risk, rather than being a linear, objective and manageable concept, is actually created, re-created and constructed. The construction of risk takes place within the constant sense-making that occurs in organizational life. Risk is, therefore, a complex phenomenon, and one that cannot be fully subjected to objective and rational analysis. Indeed, because risk is created through the ways in which reality is constructed by members of the organization, such risks are not 'real' but rather based on assumptions and perceptions. This paper, then, investigates the non-linearity and non-rationality of risk creation, and thus the difficulty of managing some risks in a linear and rational manner. The linearity of risk management techniques, as proposed in much of the literature, often creates a facade of order and control that do not exist in an essentially constructed and complex organizational world.

TITLE : Time and Risk Management of International Projects

AUTHOR : A. Aleshin

YEAR : 2014

DESCRIPTION

This paper proposes a mathematical model supporting the management of project risk. The model distinguishes between risks which have to be accepted and risks which can be eliminated at some cost, helping to decide which risks should be eliminated so that the customer requirements with respect to project completion time can be satisfied at minimal cost. The model is based on a modification of the PERT method and can be reduced to a mixed linear programming problem. The model is illustrated by means of a real world case concerning a construction project.

CHAPTER 3

SYSTEM REQUIREMENTS SPECIFICATIONS

3.1 EXISTING SYSTEM:

- In Existing System is that it can only keep track of tasks Manually.
- Should visit every person to check the task is completed or not.
- It consumes time.
- It uses Many resources
- Hard to maintain the records of Task manually

3.1.1 EXISTING SYSTEM DISADVANTAGES:

- No proper work report of employees.
- No proper maintenance of employees.
- Paper work is more.
- Team leader cannot have proper information of employees

3.2 PROPOSED SYSTEM

- Task management system commonly offers communication tools that can assist teams in discussing issues in real time. The benefit is that each team member can be kept up to date, quickly dealing with issues as they arise.
- For projects that require the use of significant documentation, document sharing tools allow individuals to edit, update the status of reports and create systems that allow for transparency and communication**.** Controlling costs is one of the most important benefits of project management. Project management software generally includes tools that can assist in managing project costs.

3.2.1 PROPOSED SYSTEM ADVANTAGES

- All employees are structured properly
- Work progress of each employee is calculated
- Work progress of each employee is monitored

3.3 REQUIREMENTS

3.3.1 HARDWARE REQUIREMENTS

The hardware requirements may serve as the basis for a contract for the implementation of the system and should therefore be a complete and consistent specification of the whole system. They are used by software engineers as the starting point for the system design. It should what the system do and not how it should be implemented.

HARDWARES

- Processor : Pentium Dual Core 2.00 GHz
- Speed : 1.1 GHz
- Hard disk : 160 GB
- RAM : 4GB (minimum)

3.3.2 SOFTWARE REQUIREMENTS

The software requirements document is the specification of the system. It should include both a definition and a specification of requirements. It is a set of what the system should do rather than how it should do it. The software requirements provide a basis for creating the software requirements specification. It is useful in estimating cost, planning team activities, performing tasks and tracking the teams and tracking the team's progress throughout the development activity.

SOFTWARES

- Operating system : Windows 7 or Higher Version
- Platform : Android
- IDE : ANDROID STUDIO
- Front End : ANDROID
- Server : XAMP/Apache
- Backend : Java, H2CONSOLE 5.7

3.4 FEASIBILITY ANALYSIS

Feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis, the feasibility study of the proposed system to be carried out. This is to ensure that the purpose of the system is not burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

3.4.1 TECHNICAL FEASIBILITY

This study is carried out to check the feasibility that is the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the clients. The developed system must have modest requirements as only minimal or null ranges are required for implementing this system.

3.4.2 OPERATION FEASIBILITY

The operation staff in the organization feasibility. The employees of the concerned organization are supportive enough to implement the proposed system. Hence, it is operationally feasible. Therefore, the proposed system is feasible in all aspects. Hence, it is encouraging to undertaking a detailed system analysis.

3.4.3 ECONOMIC FEASIBILITY

The study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is omitted. The expenditure must be justified. Thus, the development system as well as within the budget and this was achieved because most of the technologies used are freely available. Only customized product to be purchased.

3.5 FUNCTIONAL REQUIREMENTS

The GUI is designed to be as user-friendly as possible, without interference to the user's regular mobile activities. For projects that require the use of significant documentation, document sharing tools allow individuals to edit, update the status of reports and create systems that allow for transparency and communication. Controlling costs is one of the most important benefits of project management. Project management software generally includes tools that can assist in managing project costs.

3.6 NON-FUNCTIONAL REQUIREMENTS

- **Portability:** It should run on specified platforms successfully. To achieve this we should test the product on all platforms before launching the product. If our project runs successfully on different platforms then our system is portable in nature.
- **Reliability:** The system should perform its intended functions under specified conditions. If our system satisfies all the specified conditions then it is Reliable in nature.
- **Reusability:** The system should be extremely reusable as a whole or part. Make the system modularize and make sure that modules are loosely coupled. This project is having reusability nature because we can reuse whole or part of this project on other systems.
- **Robustness:** The system on the whole should be robust enough to perform well under different circumstances without any inconsistencies.
- **Testability:** The product of a given development phase should satisfy the conditions imposed at the start of that phase.
- **Usability:** It should be perfect and comfortable for users to work.
- **Security:** The system is completely based on the security. This system will providesecurity base on the password.

CHAPTER 4

SYSTEM ANALYSIS AND DESIGN

4.1 DESIGNING PART

The act, process, or profession of studying an activity (such as a procedure, a business, or a physiological function) typically by mathematical means in order to define its goals or purposes and to discover operations and procedures for accomplishing them most efficiently. Design Engineering deals with the various UML [Unified Modelling language] diagrams for the implementation of project. Design is a meaningful engineering representation of a thing that is to be built. Software design is a process through which the requirements are translated into representation of the software. Design is the place where quality is rendered in software engineering. Design is the means to accurately translate customer requirements into finished product. The concept of design as a way of making sense of things has been the subject of many studies as has the design process itself. Since “design” can be used to express intention as opposed to the actual materials, forms, processes and markets, it is often used to describe the driving force of the creative thought itself. When attempting to characterize the major movements which operate within operate within the world of design today, three in particular seem to each be characterized by specific discourses and values and to be practiced by large numbers of designers and other professionals. Technology driven design, sustainable design and human created design are major movements which usually lead to distinguishably different results despite operating within the same legal, regulatory, contextual and economic constraints. Human centred design has its roots in semi-scientific fields such as ergonomics, computer science and artificial intelligence. The toolbox of human centred design techniques grows continuously, sometimes by borrowing from fields such as psychology or sociology and sometimes by defining new approaches which emerge from design practice. The most basic form of tool consists of facts about people such as anthropometric, biomechanical, cognitive, emotional, psychophysical, psychological and sociological data and model. This design is also well aligned with the corporate branding frameworks which many businesses use to prevent themselves to the world and to position themselves with respect to their competitors.

Design Process:

Designing a product is a challenging process when the purpose is to create a pleasure and a convincing experience for the user. A well-designed and a clear process is a solution for all the cases and it will remove all confusion and doubts. Documenting the design process would be helpful in providing an estimated delivery time of the product and the required effort for the project. User Experience Design is the process of enhancing user satisfaction with a product by improving the usability, accessibility and pleasure provided in the interaction with the product. In the process we go through different stages repeatedly. Each stage involves relevant stakeholders in the organization that take part in the process of making products highly efficient and usable.

The design process involves the following 6 stages:

1. Understand:

Before beginning the design work, let the design team understand the requirements clearly. Outcomes of this stage are User Personas, User Stories, Use Cases, user Flows.

2.Research:

Design team does their research work to explore how the outer world is working on the particular feature. Outcomes of this stage are a bunch of ideas and material on which we can build the actual design work.

3.Sketch:

This stage involves UI definition of required feature. Designing is not something that we just create and start using it. Draw and draft and redraw and redraft, thus creating an unmatched experience. Outcomes of this stage are sketches, wireframes, Mockups, User Flows. 15

4.Design:

Turn the initial mock ups and wireframes to great-looking images with theme and styles applied to them. Outcomes of this stage are Design images, detailed design specs like colours, theme, styles, guidelines, Icons.

5.Implement:

Development team builds backend functionality first and connects it with UI when they get design artifacts. While implementing, it is possible to raise the need of minor

changes in design. Outcome of this stage is developed UI with complete functionality and experience following the designed theme and style

6. Evaluate:

When products features are implemented, the end product are evaluated based on few factors like whether the system is usable or not, is it easy to use for end user. The outcomes of this. stage are User feedback, UI audit reports, areas marked where improvement is required. After this stage, the process will iterate itself and depending on the required changes, you may go to stage 2, 3 or 4. The process goes on until the desired experience and customer satisfaction is achieved.

4.2 MODULES AND THEIR DESCRIPTION

- User Interface
- Admin
- Developer

4.2.1 User Interface:

In this module we design the Activities for the project. These Activities are used for secure login for all users. To connect with server user must give their username and password then only they can able to connect the server. If the user already exits directly can login into the server else user must register their details such as username, password and Email id, into the server. Server will create the account for the entire user to maintain upload and download rate. Name will be set as user id. Logging in is usually used to enter a specific page.

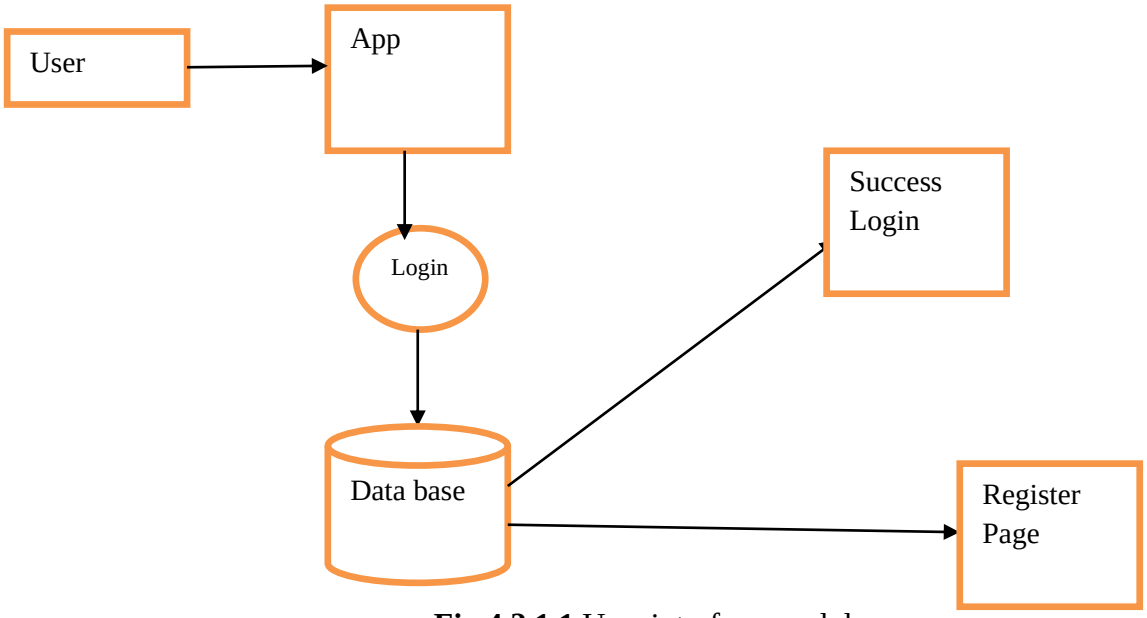


Fig 4.2.1.1 User interface module

4.2.2 Admin:

In these module admin adds the developers to the application. He can see the complaints raised by developers. He can track the bugs raised by the developers. He maintains all the activities of the application. He can track ever user in the application. He can send suggestions to the developers to solve the problems. Admin can send apis, json problems which are not accessable by developers . he can send previous solutions for same bugs

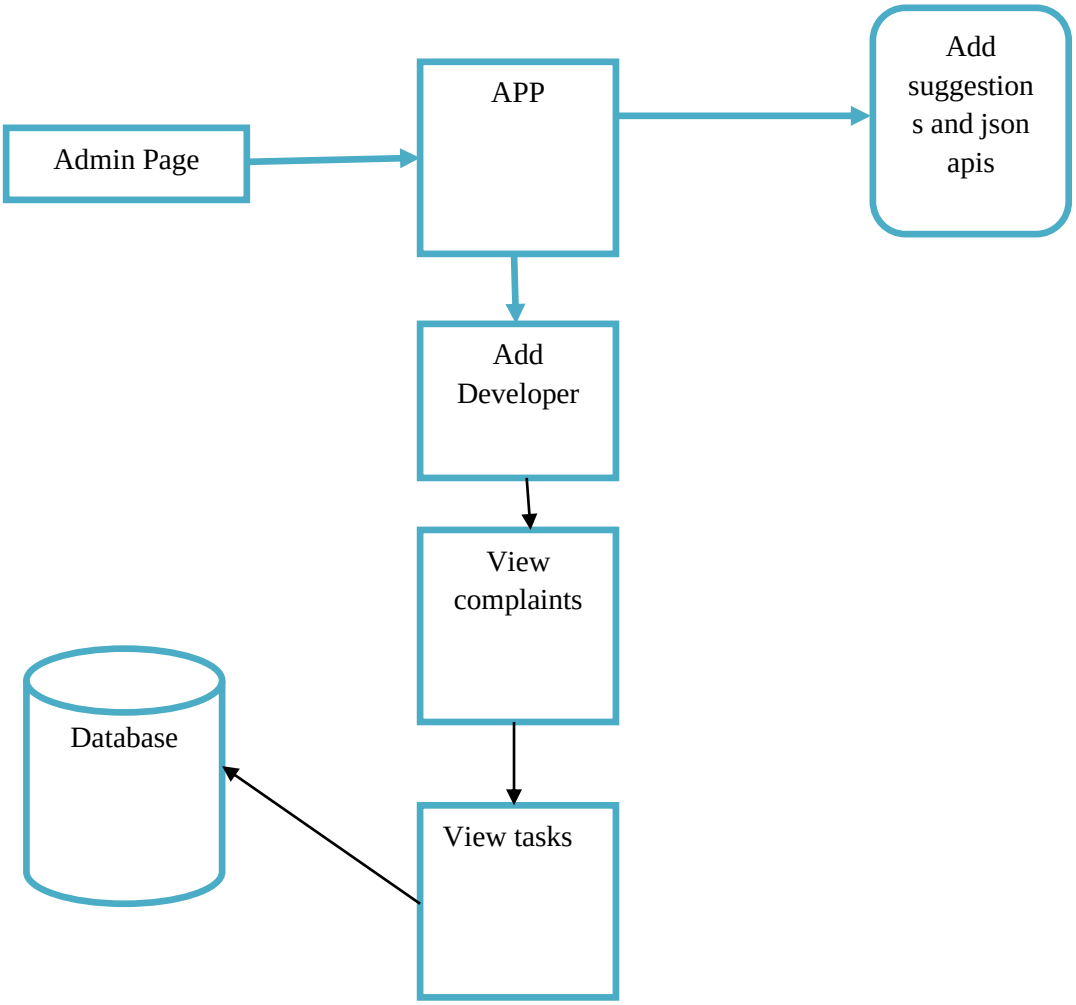


Fig 4.2.2.1 Admin module

4.2.3 Developer :

In these module developer has to register with given employee id and login to application. He can raise the complaints regarding the applications provided to him. Developer can check the bugs and get solution or suggestions form admin to solve the bugs. He can access previously solved bugs . he can access the previously asked questions regarding the bug. He can send image screen shots to the admin.

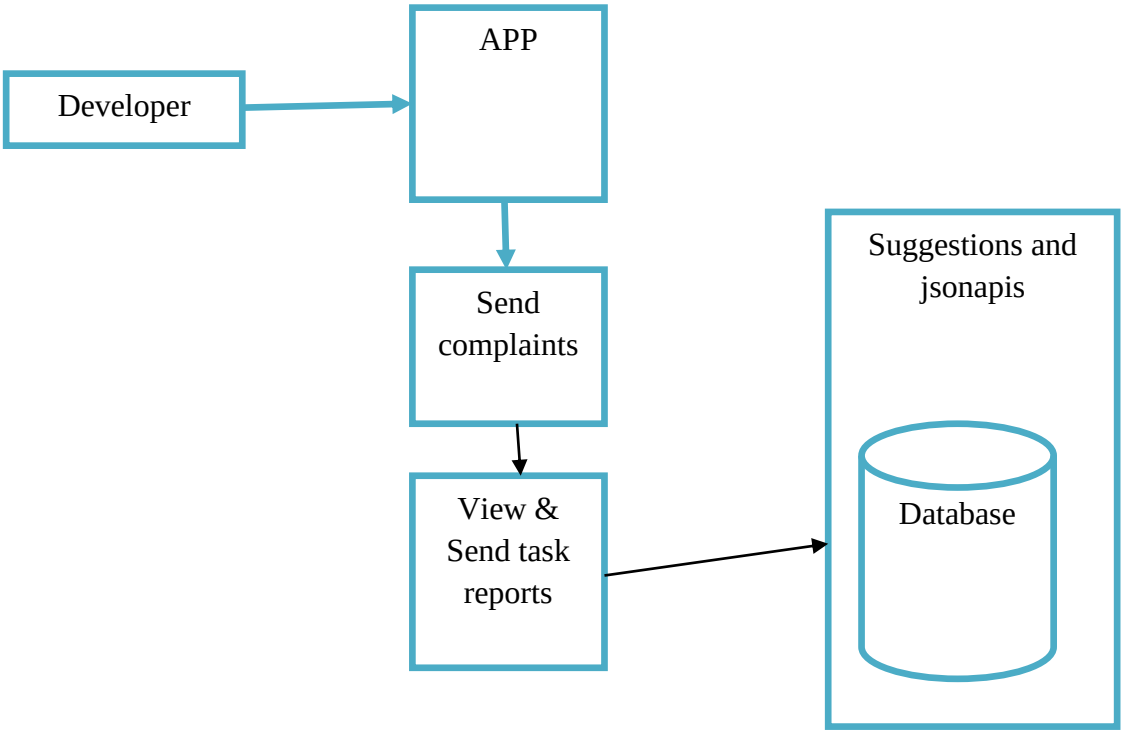


Fig 4.2.3.1 Developer module

4.3 SYSTEM ARCHITECTURE

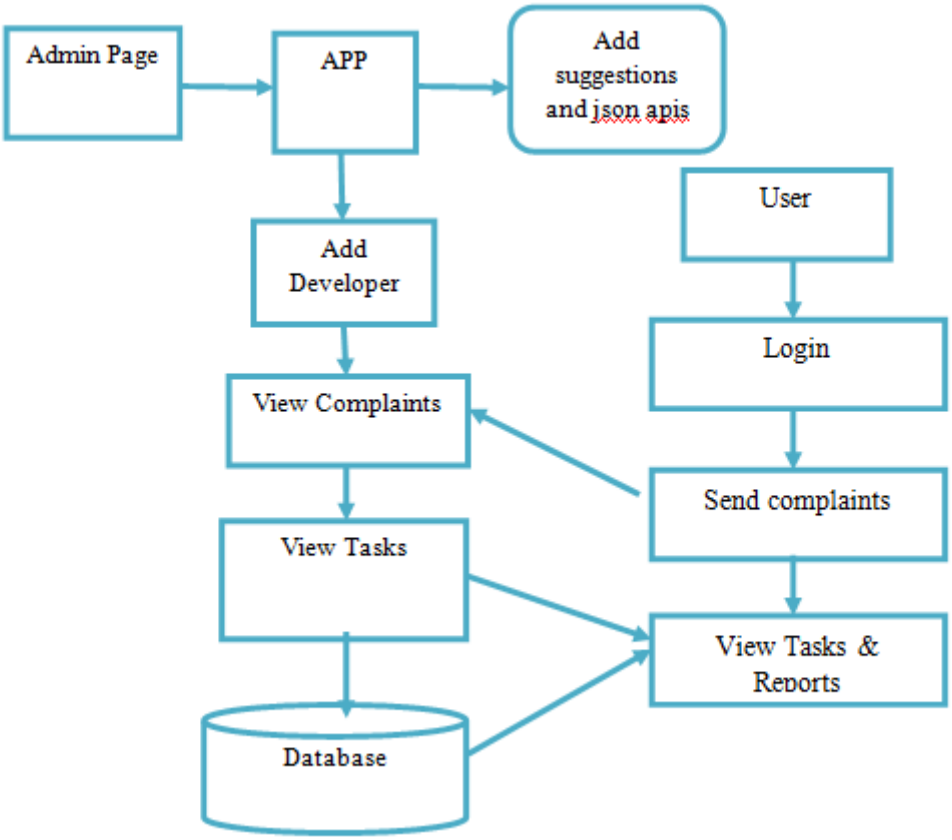


Fig 4.3 System Architecture

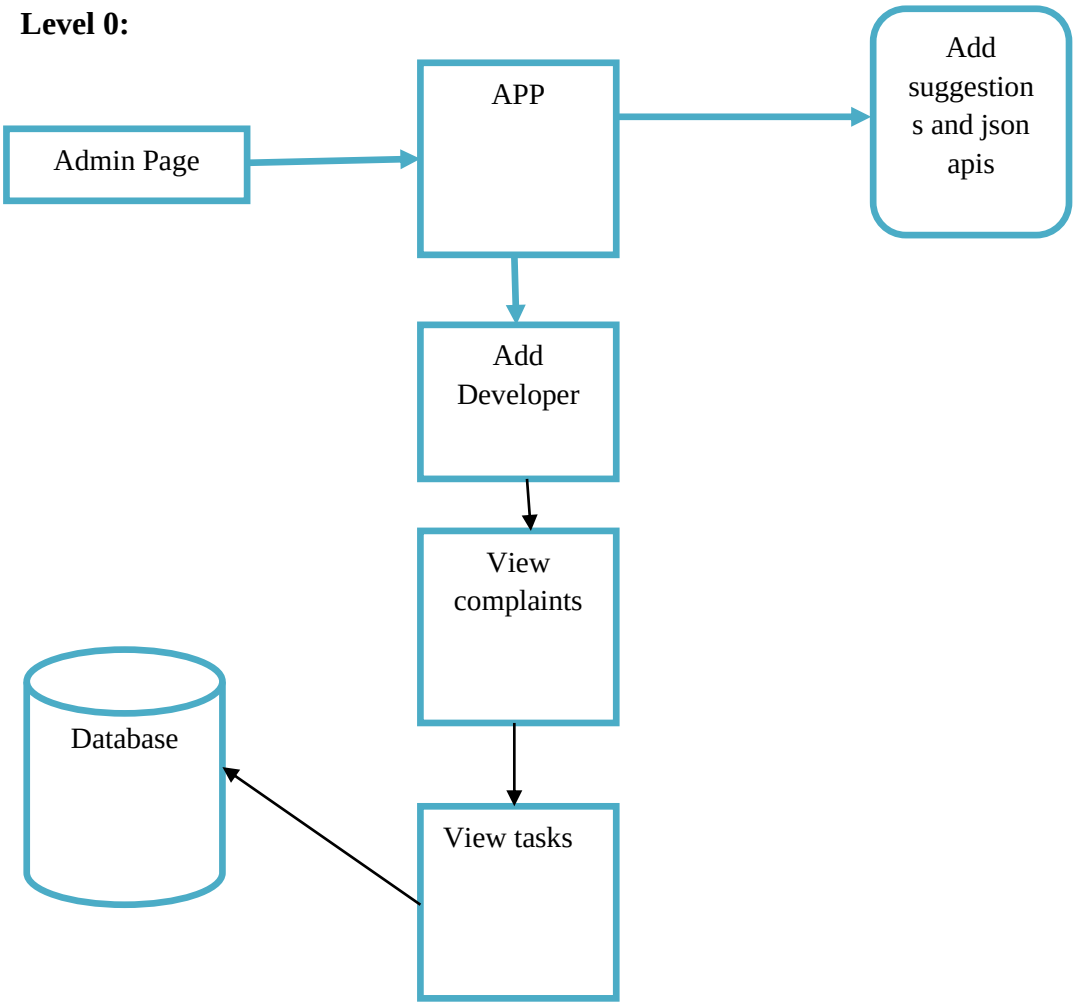
EXPLANATION:

The motivation for drivers to offer car rides can be achieving rating points (which will grant discounts and offers). Payment from passengers is another option (although not entirely allowed everywhere, because of restrictions of insurance policies). Businesses may encourage their employees (financially) to use this system, which will reduce the number of cars some companies have to provide to their workers, and reduce their expenses on gas and transportation

4.4 DATA FLOW DIAGRAM:

- The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out this data, and the output data is generated by this system.
- The data flow diagram (DFD) is one of the most important modelling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
- DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow the transformations that are applied as data moves from input to output.
- DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.

Level 0:



Level 1:

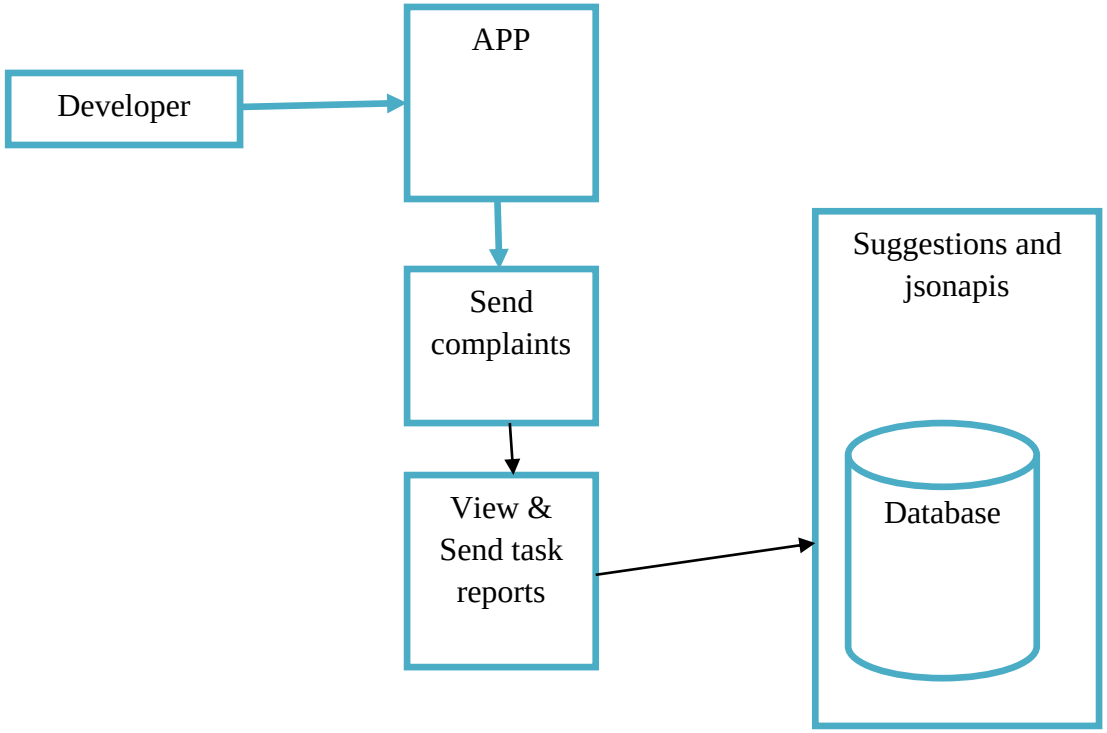


Fig 4.4 Data Flow diagram

4.5 UML DIAGRAMS:

UML stands for Unified Modelling Language. UML is a standardized general-purpose modelling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.The Unified Modelling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modelling and other non software systems.The UML represents a collection of best engineering practices that have proven successful in the modelling of large and complex systems.The UML is a very important part of developing object oriented software and the software development process. The UML uses mostly graphical notations to express the design of software project

4.5.1 GOALS:

The Primary goals in the design of the UML are as follows:

- Provide users a ready-to-use, expressive visual modelling Language so that they can develop and exchange meaningful models.
- Provide extendibility and specialization mechanisms to extend the core concepts.
- Be independent of particular programming languages and development process.
- Encourage the growth of OO tools market.
- Provide a formal basis for understanding the modelling language.
- Support higher level development concepts such as collaborations, frameworks, patterns and components.
- Integrate best practices.

4.5.2 Use Case Diagram

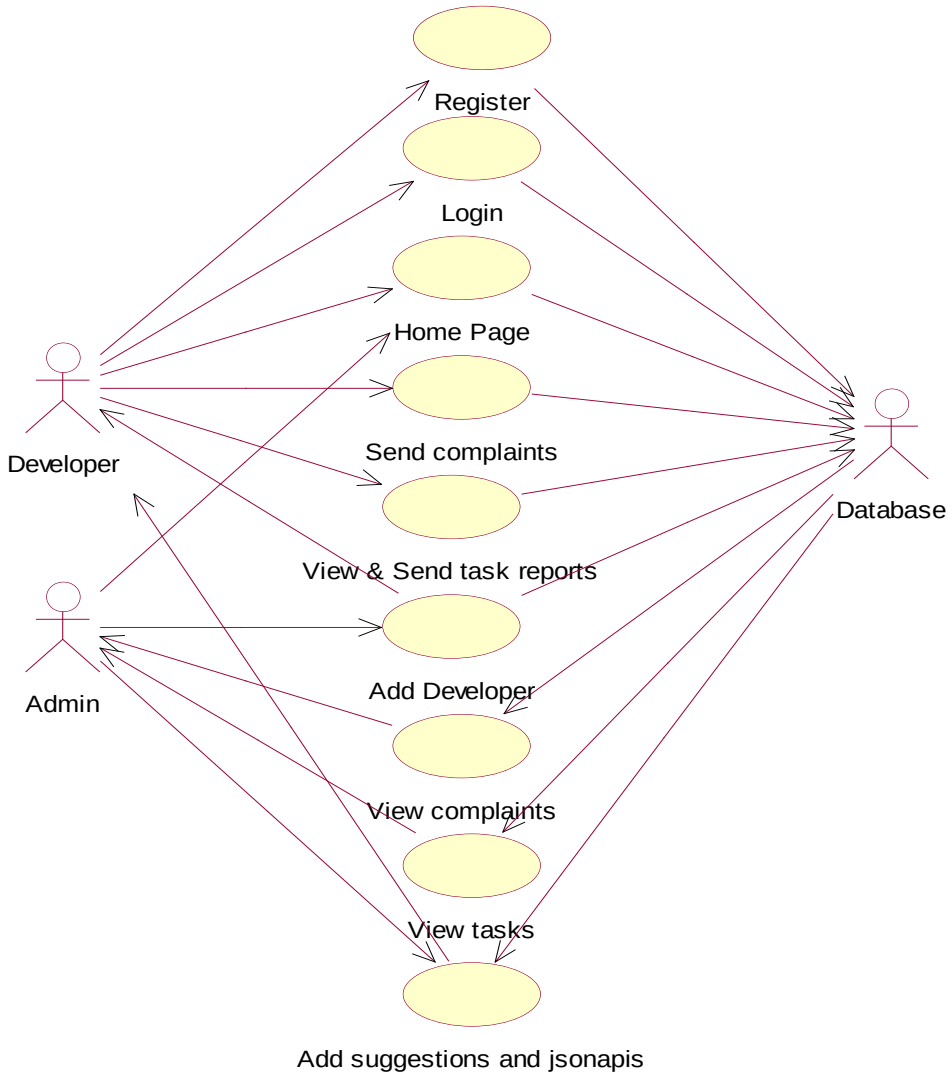


Fig 4.5.2 Use case diagram

EXPLANATION:

The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted. The above diagram consists of user as actor. Each will play a certain role to achieve the concept.

4.5.3 Class Diagram

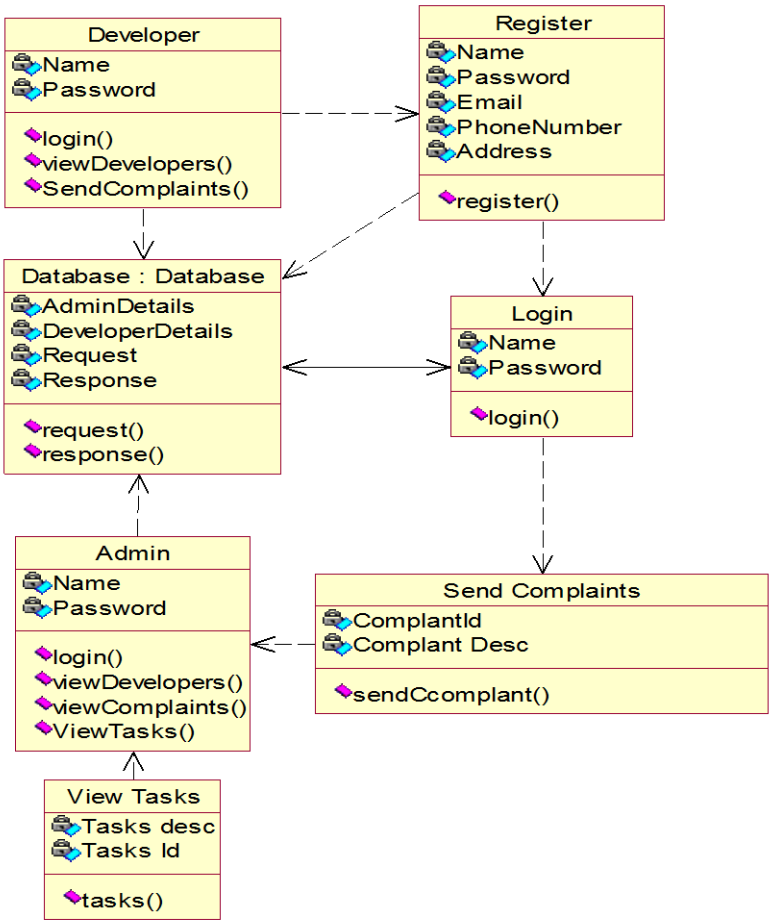


Fig 4.5.3 Class diagram

EXPLANATION

In this class diagram represents how the classes with attributes and methods are linked together to perform the verification with security. From the above diagram shown the various classes involved in our project

4.5.4 Object Diagram

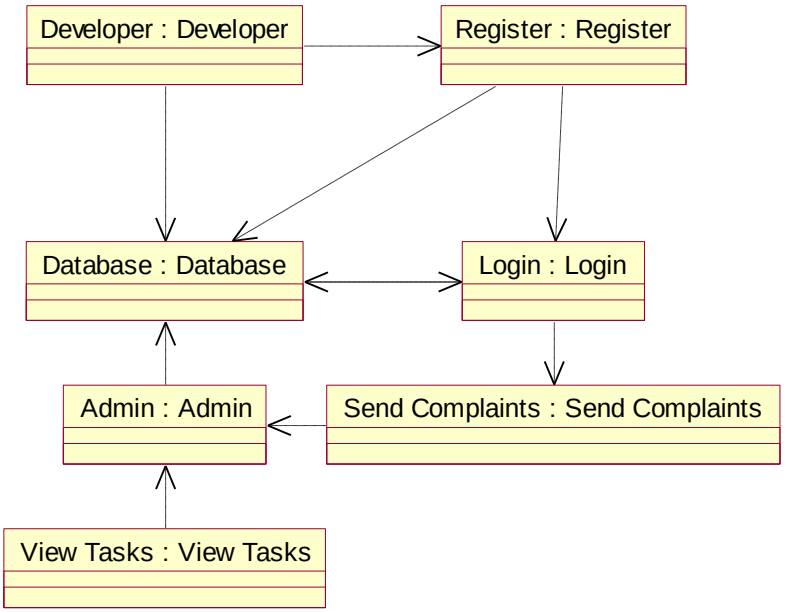


Fig 4.5.4 Object diagram

EXPLANATION:

In the above digram tells about the flow of objects between the classes. It is a diagram that shows a complete or partial view of the structure of a modelled system. In this object diagram represents how the classes with attributes and methods are linked together to perform the verification with security.

4.5.5 State Chart Diagram

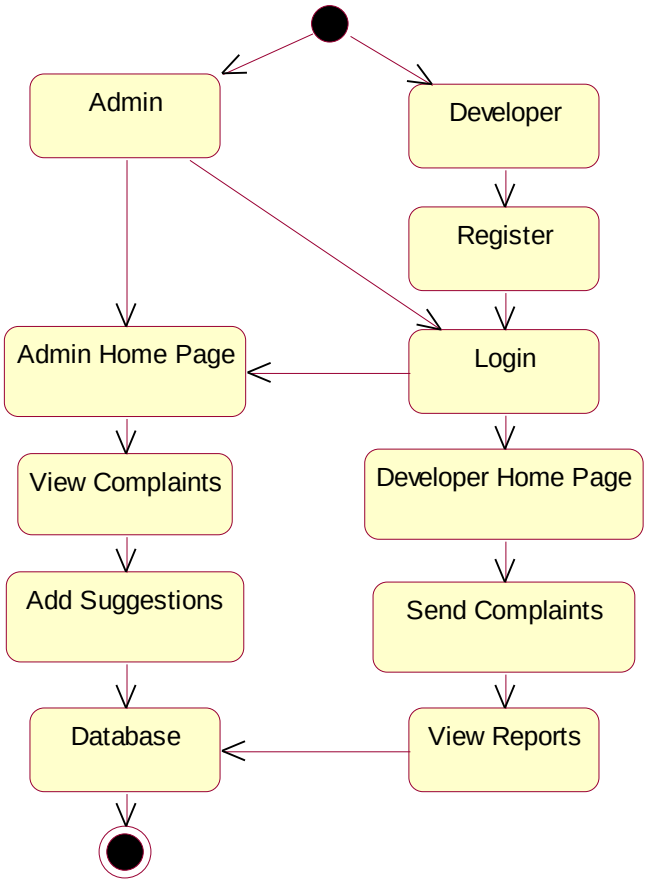


Fig 4.5.5 State Chart diagram

EXPLANATION:

State diagram are a loosely defined diagram to show workflows of stepwise activities and actions, with support for choice, iteration and concurrency. State diagrams require that the system described is composed of a finite number of states; sometimes, this is indeed the case, while at other times this is a reasonable abstraction. Many forms of state diagrams exist, which differ slightly and have different semantics.

4.5.6 Sequence Diagram

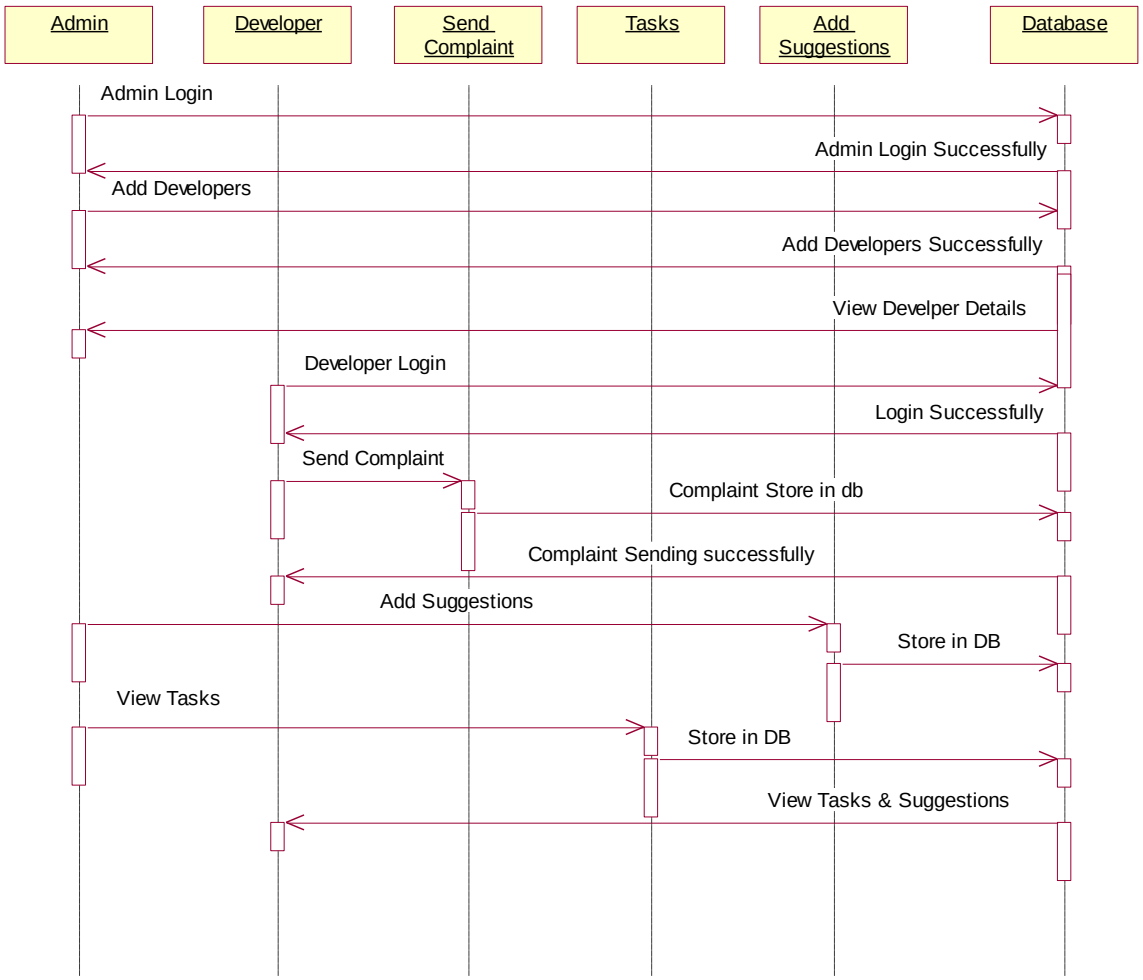


Fig 4.5.6 Sequence diagram

EXPLANATION:

A sequence diagram in Unified Modelling Language (UML) is a kind of interaction diagram that shows how processes 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. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario.

4.5.7 Collaboration Diagram

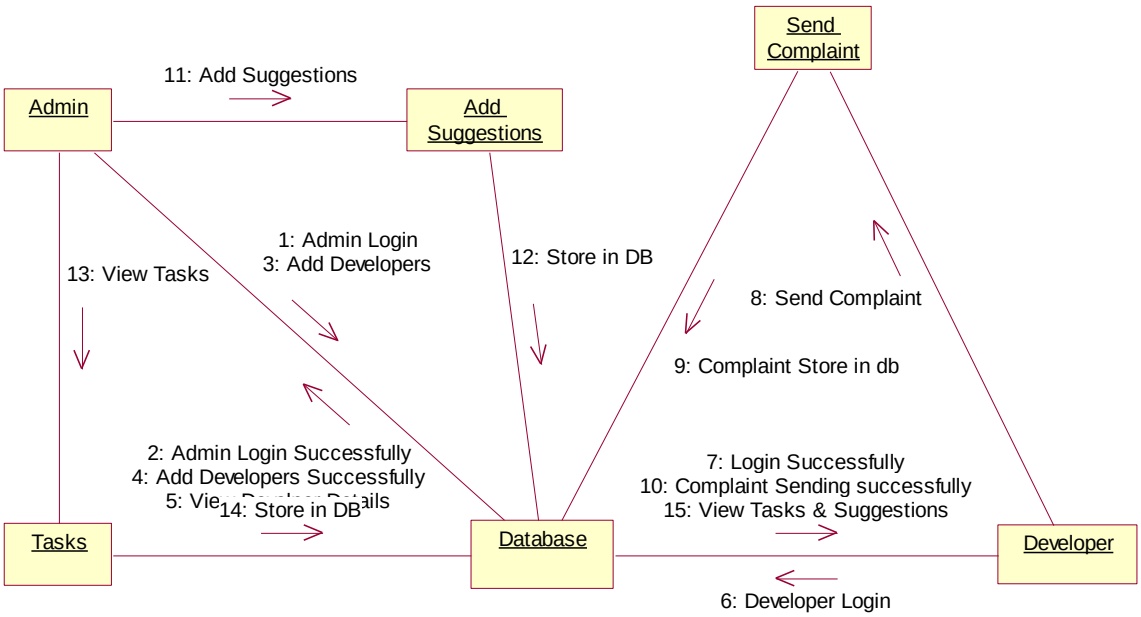


Fig 4.5.7 Collaboration diagram

EXPLANATION:

A collaboration diagram, also called a communication diagram or interaction diagram, is an illustration of the relationships and interactions among software objects in the Unified Modelling Language (UML). The concept is more than a decade old although it has been refined as modelling paradigms have evolved.

4.5.8 Activity Diagram

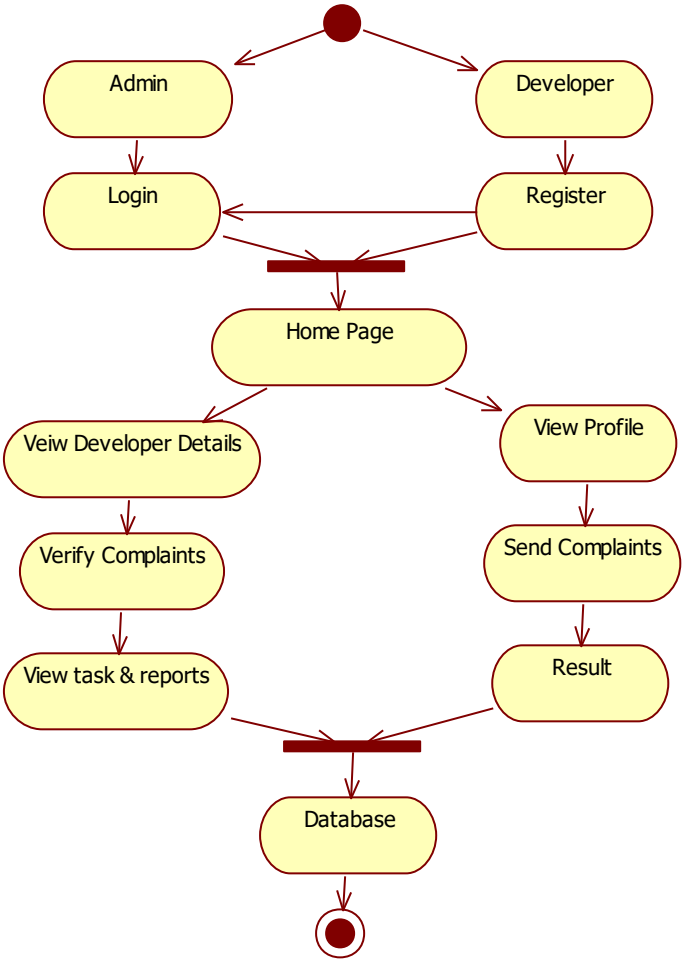


Fig 4.5.8 Activity diagram

EXPLANATION:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modelling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

4.5.9 Component Diagram

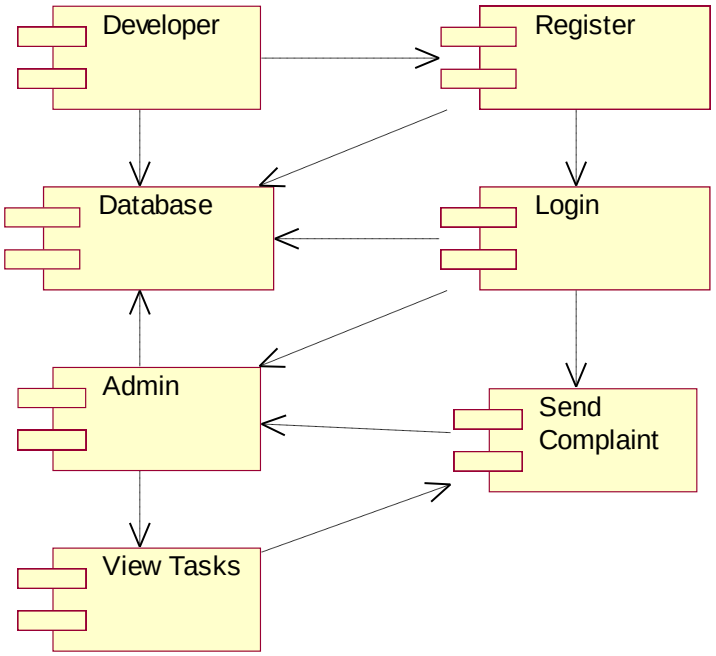


Fig 4.5.9 Component diagram

EXPLANATION:

In the Unified Modelling Language, a component diagram depicts how components are wired together to form larger components and or software systems. They are used to illustrate the structure of arbitrarily complex systems. User gives main query and it converted into sub queries and sends through data dissemination to data aggregators. Results are to be showed to user by data aggregators. All boxes are components and arrow indicates dependencies.

4.5.10 E-R Diagram:

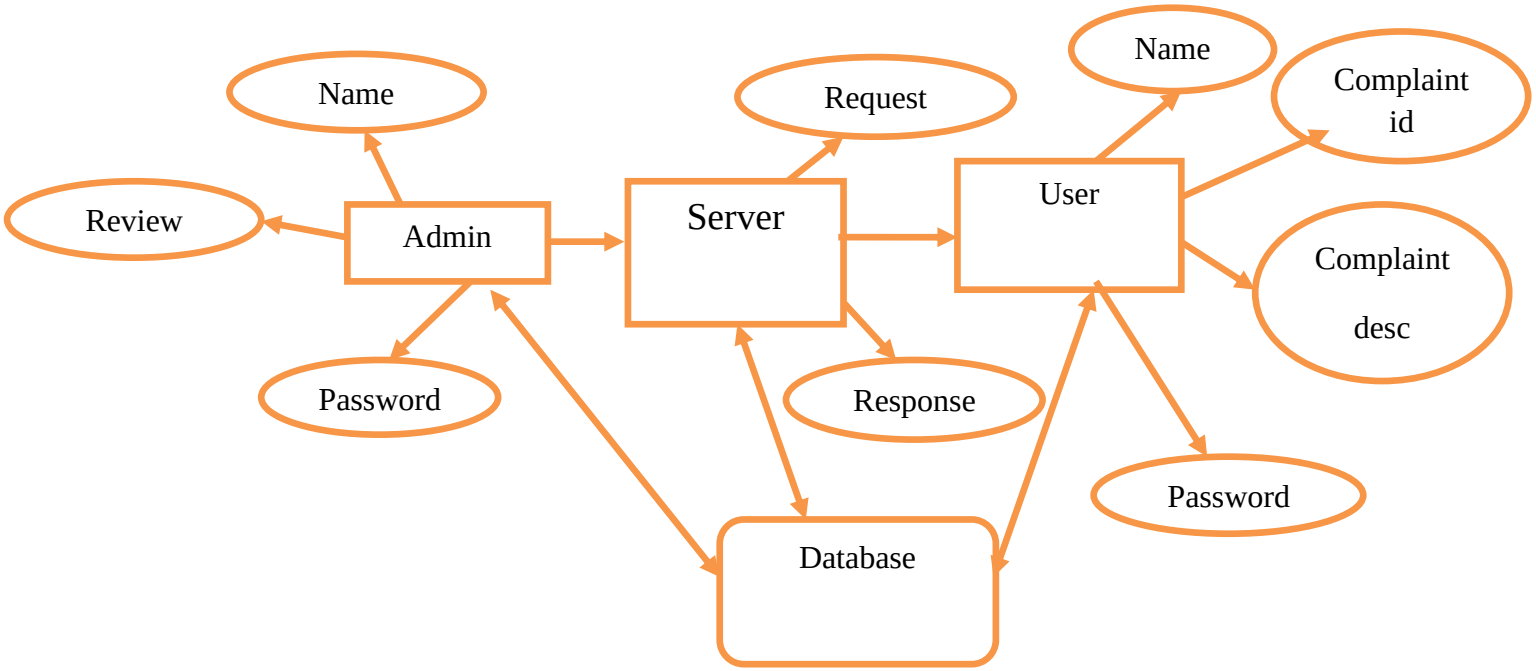


Fig 4.5.10 E R diagram

EXPLANATION:

Entity-Relationship Model (ERM) is an abstract and conceptual representation of data. Entity-relationship modelling is a database modelling method, used to produce a type of conceptual schema or semantic data model of a system, often a relational database.

4.5.11 Deployment Diagram:

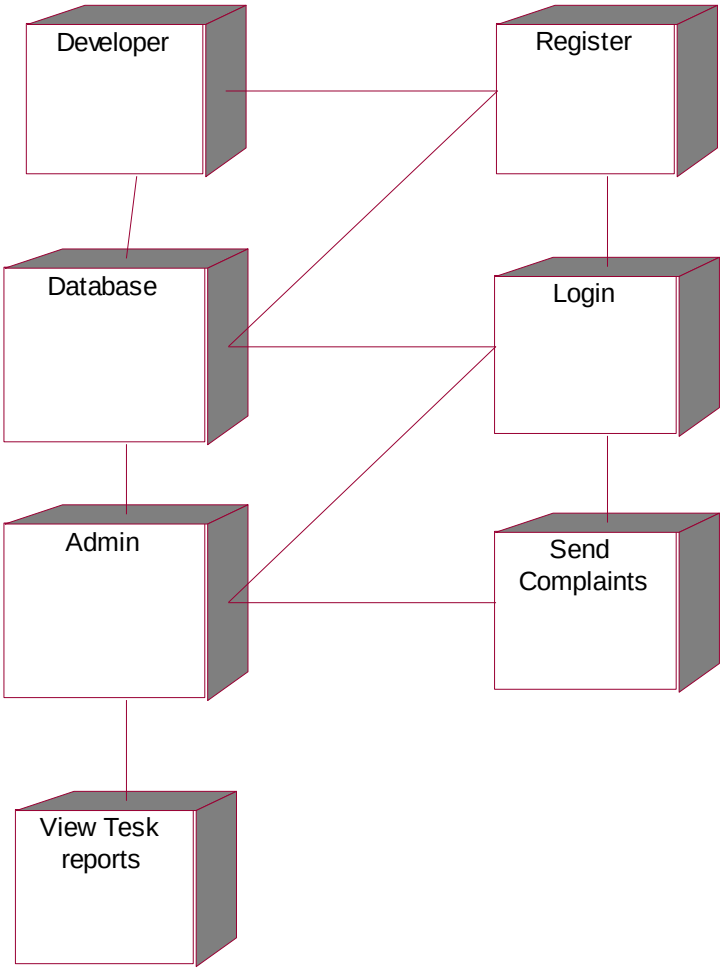


Fig 4.5.11 Deployment diagram

EXPLANATION:

In the Unified Modelling Language, a deployment diagram depicts how deploys are wired together to form larger deployment and or software systems. They are used to illustrate the structure of arbitrarily complex systems. User gives main query and it converted into sub queries and sends through data dissemination to data aggregators. Results are to be showed to user by data aggregators. All boxes are components and arrow indicates dependencies.

CHAPTER 5
IMPLEMENTATION

5.1TECHNOLOGY USED:

5.1.1 About android

Android is a complete set of software for mobile devices such as tablet computers, notebooks, smart phones, electronic book readers, set-top boxes. It contains a Linux-based Operating System, middleware and key mobile applications. It can be thought of as a mobile operating system. But it is not limited to mobile only. It is currently used in various devices such as mobiles, tablets, televisions etc.

5.1.2 What is android?

Android is a software package and Linux based operating system for mobile devices such as tablet computers and smart phones. It is developed by Google and later the OHA (Open Handset Alliance). Java language is mainly used to write the android code even though other languages can be used. The goal of android project is to create a successful real-world products that improves the mobile experience for end users. There are many code names of android such as Lollipop, Kitkat, Jelly Bean, Ice cream Sandwich, Oreo, Eucliar, Donut etc which is covered in next page.

5.1.3 About Open Handset Alliance (OHA)

It's a consortium of 84 companies such as Google, Samsung, AKM, synaptics, KDDI, Garmin, Teleca, Ebay, Intel etc. It was established on 5th November, 2007, led by Google. It is committed to advance open standards, provide services and deploy handsets using the Android Platform.

5.1.4 Features of Android

1) It is open-source.

2) Anyone can customize the Android Platform.

3) There are a lot of mobile applications that can be chosen by the consumer.

4) It provides many interesting features like weather details, opening screen, live RSS (Really Simple Syndication) feeds etc.

5) It provides support for messaging services(SMS and MMS), web browser, storage (SQLite), connectivity (GSM, CDMA, Blue Tooth, Wi-Fi etc.), media, handset layout etc.

5.1.5 Categories of Android applications

- Entertainment
- Tools
- Communication
- Productivity
- Personalization
- Music and Audio
- Social
- Media and Video
- Travel and Local etc.

5.1.6 History of Android

1) Initially, Andy Rubin founded Android Incorporation in Palo Alto, California, United States in October, 2003.

2) In 17th August 2005, Google acquired android Incorporation. Since then, it is in the subsidiary of Google Incorporation.

3) The key employees of Android Incorporation are Andy Rubin, Rich Miner, Chris White and Nick Sears.

4) Originally intended for camera but shifted to smart phones later because of low market for camera only.

5) Android is the nick name of Andy Rubin given by coworkers because of his love to robots.

6) In 2007, Google announces the development of android OS. In 2008, HTC launched the first android mobile.

5.1.7 Android Architecture

1. Linux kernel: It is the heart of android architecture that exists at the root of android architecture. Linux kernel is responsible for device drivers, power management, memory management, device management and resource access.

2. Native libraries (middleware): On the top of linux kernel, their are native libraries such as WebKit, OpenGL, FreeType, SQLite, Media, C runtime library (libc) etc. The WebKit library is responsible for browser support , SQLite is for database, FreeType for font support, Media for playing and recording audio and video formats.

3. Android Runtime: In android runtime, there are core libraries and DVM (Dalvik Virtual Machine) which is responsible to run android application. DVM is like JVM but it is optimized for mobile devices. It consumes less memory and provides fast performance.

4. Application Framework: On the top of Native libraries and android runtime, there is android framework. Android framework includes Android API's such as UI (User Interface), telephony, resources, locations, Content Providers (data) and package managers. It provides a lot of classes and interfaces for android application development.

5. Applications: On the top of android framework, there are applications. All applications such as home, contact, settings, games, browsers are using android framework that uses android runtime and libraries. Android runtime and native libraries are using linux kernal.

5.1.8 Android Core Building Blocks

An android component is simply a piece of code that has a well defined life cycle e.g. Activity, Receiver, Service etc.The core building blocks or fundamental components of android are activities, views, intents, services, content providers, fragments and AndroidManifest.xml.

Activity

An activity is a class that represents a single screen. It is like a Frame in AWT.

View

A view is the UI element such as button, label, text field etc. Anything that you see is a view.

Intent

Intent is used to invoke components. It is mainly used to:

- Start the service
- Launch an activity
- Display a web page
- Display a list of contacts
- Broadcast a message
- Dial a phone call etc.
- For example, you may write the following code to view the webpage.

Example :

```
Intent intent=new Intent(Intent.ACTION_VIEW);
intent.setData(Uri.parse("http://www.gurunanak.com"));
```

startActivity(intent);

Service

Service is a background process that can run for a long time. There are two types of services local and remote. Local service is accessed from within the application whereas remote service is accessed remotely from other applications running on the same device.

Content Provider

Content Providers are used to share data between the applications.

Fragment

Fragments are like parts of activity. An activity can display one or more fragments on the screen at the same time.

AndroidManifest.xml

It contains information about activities, content providers, permissions etc. It is like the web.xml file in Java EE.

Android Virtual Device (AVD)

It is used to test the android application without the need for mobile or tablet etc. It can be created in different configurations to emulate different types of real devices.

APK File

An APK file is created by the framework automatically. If you want to run the android application on the mobile, transfer and install it.

Resources

It contains resource files including activity_main, strings, styles etc.

Manifest file

It contains information about package including components such as activities, services, content providers etc.

5.1.9 Android Activity Lifecycle

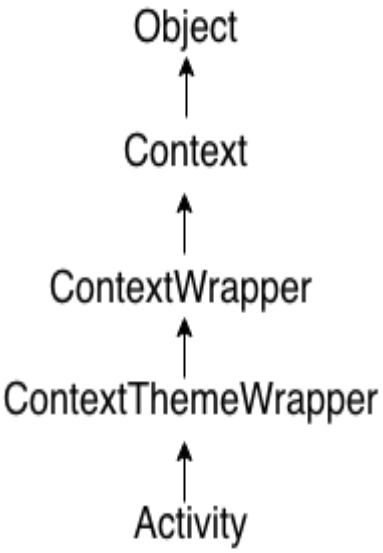
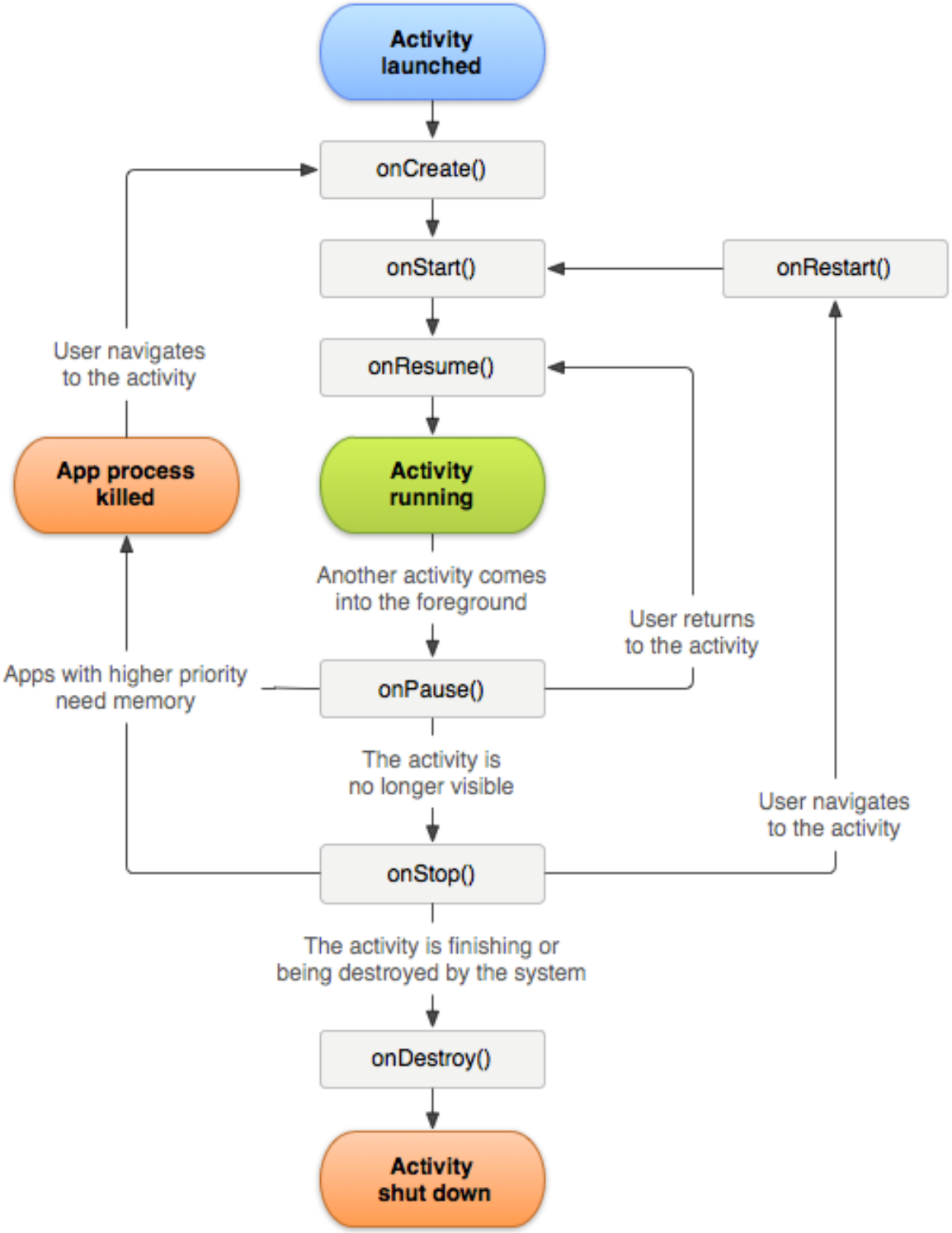


Fig 5.9 Android Activity Lifecycle

Android Activity Lifecycle is controlled by 7 methods of android.app.Activity class. The android Activity is the subclass of ContextThemeWrapper class. An activity is the single screen in android. It is like window or frame of Java. By the help of activity, you can place all your UI components or widgets in a single screen. The 7 lifecycle method of Activity describes how activity will behave at different states.



ANDROID LIFE CYCLE

5.2 SAMPLE CODE:

Constants.java:

```
  package com.example.abhi.taskmanagement.helper;
  public class Constants {
  //there should be no spaces between the ip address and port number
  //http://ip:portno/
   public static String ip="http://192.168.0.6:8096/";
   public static String userLogin=ip+"userLogin";
   public static String userreg=ip+"userReg";
   public static String getUserDetails=ip+"getUserDetails";
   public  static String getProjects=ip+"getProjects";
   public static String getDevelopers=ip+"getDevelopers";
   public static String storeWork=ip+"storeWork";
   public static String getWorks=ip+"getWorks";
   public static String updateStatus=ip+"updateStatus";
   public static String getdevWorks=ip+"getdevWorks";
   public static String profile=ip+"profile";
}
```

Store.java:

```
package com.example.abhi.taskmanagement.helper;
import android.content.Context;
import android.content.SharedPreferences;
import java.util.HashMap;
public class Store {
static SharedPreferences sharedPreferences;
static SharedPreferences.Editor editor;
public static  void userDetails(Context context, int userid, String username,String role){
sharedPreferences=context.getSharedPreferences("railway",Context.MODE_PRIVATE);
editor=sharedPreferences.edit();
editor.putInt("userid",userid);
    editor.putString("developername",username);
    editor.putString("role",role);
    editor.commit();
   }
    public static HashMap<String,String> getUserDetails(Context context){
    sharedPreferences=context.getSharedPreferences("railway",Context.MODE_PRIVATE);
    HashMap map=new HashMap();
    map.put("developername",sharedPreferences.getString("developername",""));
    map.put("userid",sharedPreferences.getInt("userid",0));
    map.put("role",sharedPreferences.getString("role",""));
    return map;
   }
   public static  void logout(Context context){
   sharedPreferences=context.getSharedPreferences("railway",Context.MODE_PRIVATE);
   editor=sharedPreferences.edit();
   editor.clear();
```

```
    editor.commit();
}}
```

ToastHelper.java:

```
package com.example.abhi.taskmanagement.helper;
import android.content.Context;
import android.widget.Toast;
public class ToastHelper {
static Context context;
static  String msg;
public static void toastMsg(Context context,String msg){
Toast.makeText(context,msg,Toast.LENGTH_SHORT).show();

    }}
```

AssignProjects.java:

```
package com.example.abhi.taskmanagement;
import android.app.Activity;
import android.app.DatePickerDialog;
import android.app.Dialog;
import android.os.Bundle;
import android.support.annotation.Nullable;
import android.util.Log;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.DatePicker;
import android.widget.EditText;
import android.widget.Spinner;
import android.widget.Toast;
import com.android.volley.AuthFailureError;
import com.android.volley.Request;
import com.android.volley.RequestQueue;
import com.android.volley.Response;
import com.android.volley.VolleyError;
import com.android.volley.toolbox.StringRequest;
import com.android.volley.toolbox.Volley;
import com.example.abhi.taskmanagement.helper.Constants;
import com.example.abhi.taskmanagement.helper.ToastHelper;
import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;
import java.util.ArrayList;
import java.util.Calendar;
import java.util.HashMap;import java.util.Map;
public class AssignProjects extends Activity {
Spinner spprojects;
Spinner spdevelopers;
EditText etdate;
```

```
private DatePicker datePicker;
private Calendar calendar;
private int year, month, day;
EditText etwork;Button btsubmit;
@Override
protected void onCreate(@Nullable Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_assign_projects);
spprojects=findViewById(R.id.spprojects);
spdevelopers=findViewById(R.id.spdevelopers);
etdate=findViewById(R.id.etdate);
etwork=findViewById(R.id.etwork);
btsubmit=findViewById(R.id.btsubmit);
getProjects();
getDevelopers();
calendar = Calendar.getInstance();
        year = calendar.get(Calendar.YEAR);
        month = calendar.get(Calendar.MONTH);
        day = calendar.get(Calendar.DAY_OF_MONTH);
        showDate(year, month+1, day);
        etdate.setKeyListener(null);
        etdate.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
        showDialog(999);
        Toast.makeText(getApplicationContext(), "ca",
        Toast.LENGTH_SHORT)

                .show();

            }

        });

            btsubmit.setOnClickListener(new View.OnClickListener() {

            @Override
            public void onClick(View view) {
                String project=spprojects.getSelectedItem().toString();
                String developer=spdevelopers.getSelectedItem().toString();
                String work=etwork.getText().toString();
                String s[]=developer.split(",");
                String date=etdate.getText().toString();
                storeworkDetails(project,work,s[0],date);
}});
}
   });

private void storeworkDetails(final String project, final String work, final String s, final String date) {
```

```
        StringRequest stringRequest=new StringRequest(Request.Method.POST, Constants.storeWork, new
Response.Listener<String>() {
 @Override
            public void onResponse(String response) {
                Log.e("response",response);
                ToastHelper.toastMsg(getApplicationContext(),"work submitted successfully");
                etwork.setText("");
            }
        }, new Response.ErrorListener() {
            @Override
            public void onErrorResponse(VolleyError error) {
                Toast.makeText(getApplicationContext(),error.toString(),Toast.LENGTH_SHORT).show();
}
        }){
            @Override
            protected Map<String, String> getParams() throws AuthFailureError {
                HashMap map=new HashMap();
                map.put("projectname",project);
                map.put("work",work);
                map.put("developername",s);
                map.put("date",date);
                return map;

            }
        };
        RequestQueue requestQueue= Volley.newRequestQueue(this);
        requestQueue.add(stringRequest);}

      @Override

      protected Dialog onCreateDialog(int id) {

     // TODO Auto-generated method stub

        if (id == 999) {
            return new DatePickerDialog(this,
                    myDateListener, year, month, day);
        }
        return null;
    }

    private DatePickerDialog.OnDateSetListener myDateListener = new
            DatePickerDialog.OnDateSetListener() {
                @Override
                public void onDateSet(DatePicker arg0,
                                      int arg1, int arg2, int arg3) {
                    // TODO Auto-generated method stub
                    // arg1 = year
                    // arg2 = month
```

```
                // arg3 = day
                showDate(arg1, arg2+1, arg3);
            }
        };
    private void getDevelopers() {
    StringRequest stringRequest=new StringRequest(Request.Method.GET,
Constants.getDevelopers, new Response.Listener<String>() {
        @Override
        public void onResponse(String response) {
            try {
                JSONArray jsonArray=new JSONArray(response);
                ArrayList al1=new ArrayList();
                for(int i=0;i<jsonArray.length();i++){
                    JSONObject jsonObject=jsonArray.getJSONObject(i);
                    int pid=jsonObject.getInt("id");
                    String pname=jsonObject.getString("developername");
                    String role=jsonObject.getString("role");
                    String s=pname+",\t"+ role;
                    al1.add(s);
                }
                ArrayAdapterarrayAdapter1=new
ArrayAdapter(getApplicationContext(),android.R.layout.simple_list_item_1,al1);
                spdevelopers.setAdapter(arrayAdapter1);


            } catch (JSONException e) {
                e.printStackTrace();
            }

        }
    }, new Response.ErrorListener() {
        @Override
        public void onErrorResponse(VolleyError error) {
            Toast.makeText(getApplicationContext(),error.toString(),Toast.LENGTH_SHORT).show();

        }
    });
    RequestQueue requestQueue= Volley.newRequestQueue(this);
    requestQueue.add(stringRequest);
    }

private void getProjects() {
StringRequeststringRequest=new StringRequest(Request.Method.GET, Constants.getProjects, new
Response.Listener<String>() {
@Override
public void onResponse(String response) {
try {
JSONArray jsonArray=new JSONArray(response);
ArrayList al=new ArrayList();
```

```
for(int i=0;i<jsonArray.length();i++){
JSONObject jsonObject=jsonArray.getJSONObject(i);
int pid=jsonObject.getInt("id");
String pname=jsonObject.getString("projectname");
String s=pname;
al.add(s);

}

                ArrayAdapterarrayAdapter=new
ArrayAdapter(getApplicationContext(),android.R.layout.simple_list_item_1,al);

                spprojects.setAdapter(arrayAdapter);
            } catch (JSONException e) {
                e.printStackTrace();
            }
        }
    }, new Response.ErrorListener() {
        @Override
        public void onErrorResponse(VolleyError error)
 {
    Toast.makeText(getApplicationContext(),error.toString(),Toast.LENGTH_SHORT).show();
        }
    });
    RequestQueue requestQueue= Volley.newRequestQueue(this);
   requestQueue.add(stringRequest);
  }
  private void showDate(int year, int month, int day) {
    etdate.setText(new StringBuilder().append(day).append("/")
            .append(month).append("/").append(year));
  }
}
Developer_success:
package com.example.abhi.taskmanagement;
import android.app.Activity;
import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import android.support.annotation.Nullable;
import android.support.v7.app.AppCompatActivity;
import android.util.Log;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.ListView;
import com.android.volley.Request;
import com.android.volley.RequestQueue;
import com.android.volley.Response;
```

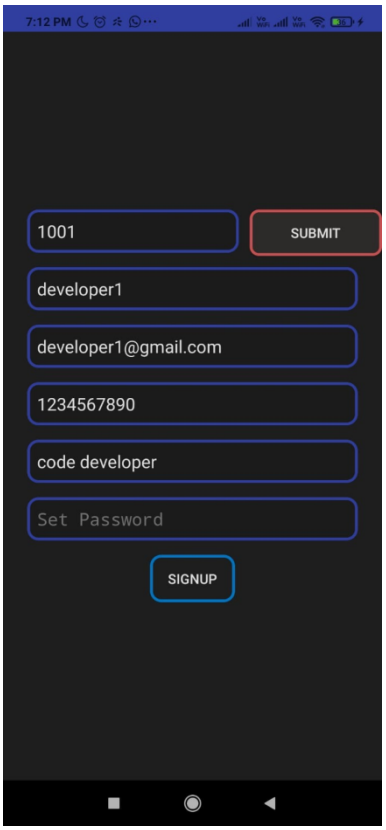
```
import com.android.volley.VolleyError;
import com.android.volley.toolbox.StringRequest;
import com.android.volley.toolbox.Volley;
import com.example.abhi.taskmanagement.helper.Constants;
import com.example.abhi.taskmanagement.helper.Store;
import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
public class Developer_success extends AppCompatActivity {
Button btview,btprofile;
ListView lvlist;
@Override
protected void onCreate(@Nullable Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_developer_page);
        btview=findViewById(R.id.btview);
        btprofile=findViewById(R.id.btprofile);
        btview.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent i=new Intent(getApplicationContext(),View_developer_Status.class);
               startActivity(i);
              //  HashMap map= Store.getUserDetails(getApplicationContext());
              //  getStatus(map.get("developername").toString());
              //  setContentView(R.layout.activity_view_status);
              //  lvlist=findViewById(R.id.lvlist);

            }

        });



        btprofile.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent i=new Intent(getApplicationContext(),Profile.class);

                startActivity(i);

            }

        });

}
```

```
    private void getStatus(String developername) {
        StringRequest stringRequest=new StringRequest(Request.Method.GET, Constants.getWorks, new
Response.Listener<String>() {
            @Override
            public void onResponse(String response) {
                Log.e("response",response);
                List<WorkModel> pm=new ArrayList();
                try {
                    JSONArray jsonArray=new JSONArray(response);
                    for(int i=0;i<jsonArray.length();i++){
                        WorkModel pms=new WorkModel();
                        JSONObject jsonObject=jsonArray.getJSONObject(i);
                        pms.setId(jsonObject.getInt("id"));
                        pms.setProjectname(jsonObject.getString("projectname"));
                        pms.setDevelopername(jsonObject.getString("developername"));
                        pms.setWork(jsonObject.getString("work"));
                        pms.setStatus(jsonObject.getString("status"));
                        pms.setDate(jsonObject.getString("date"));
                        pm.add(pms);
                    }
                    addToDisplay(getApplicationContext(),pm);
                } catch (JSONException e) {
                    e.printStackTrace();

                }

            }
        }, new Response.ErrorListener() {
            @Override
            public void onErrorResponse(VolleyError error) {
            }
        });
        RequestQueue requestQueue= Volley.newRequestQueue(this);
        requestQueue.add(stringRequest);
    }
    private void addToDisplay(Context context, List<WorkModel> pm) {
        StatusAdapter projectAdapter=new StatusAdapter(context,pm);
        lvlist.setAdapter(projectAdapter);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.user_success, menu);
        return true;

    } @Override

public boolean onOptionsItemSelected(MenuItem item) {
```

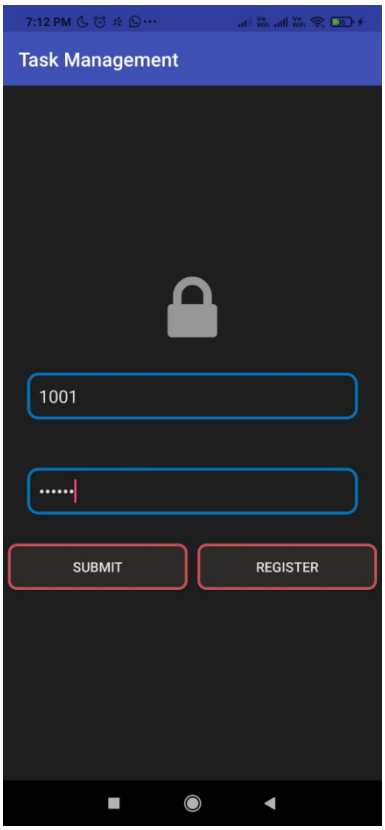
```
            switch (item.getItemId()){
            case R.id.action_settings:{
                Store.logout(this);
                Intent i=new Intent(this,MainActivity.class);
                finish();
                startActivity(i);
            }
        }
        return true;
    }
}
```

5.3 SCREENSHOTS

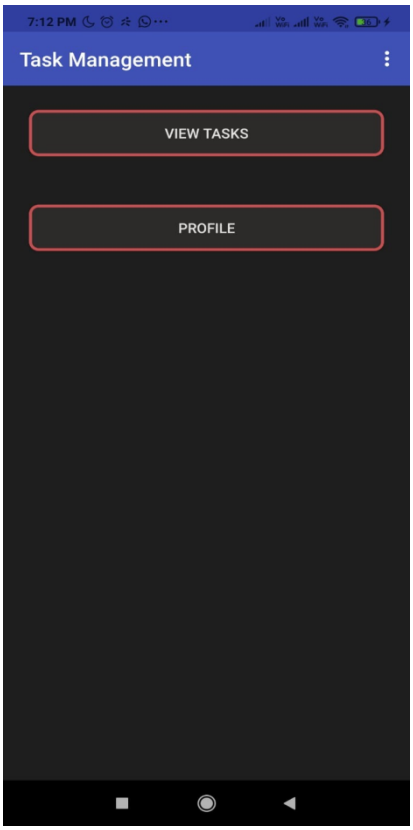
5.3.1 Signup Page



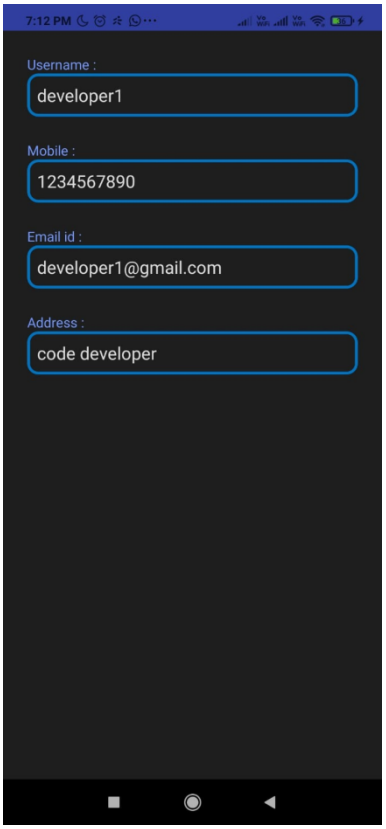
5.3.2 Submit or Register Page



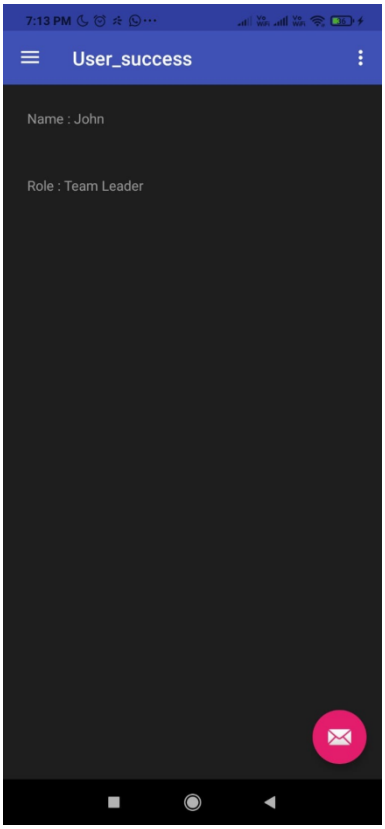
5.3.3 Employee Login Success



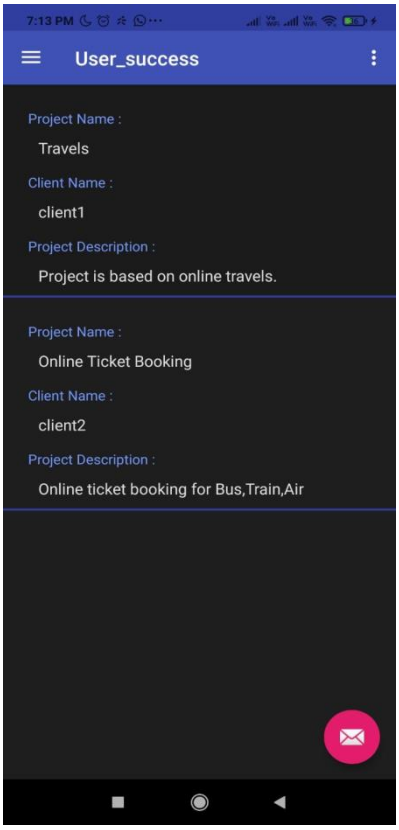
5.3.4 Employee Profile



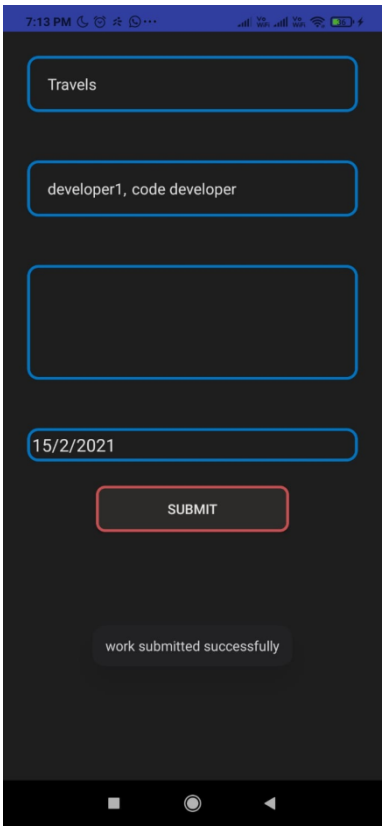
5.3.5 User Success Page



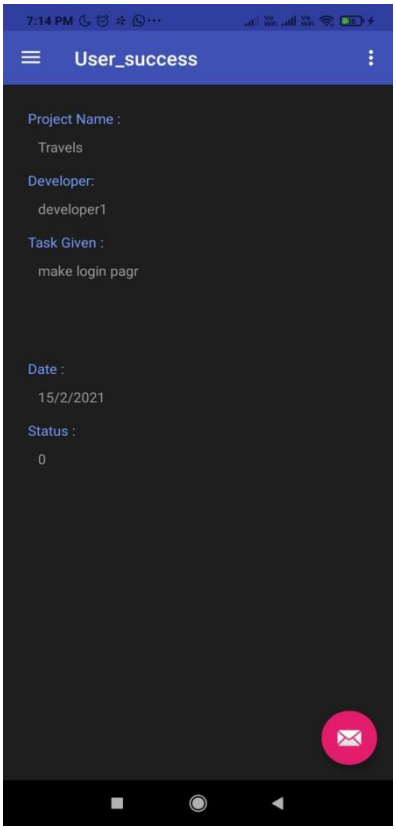
5.3.6 Projects Page

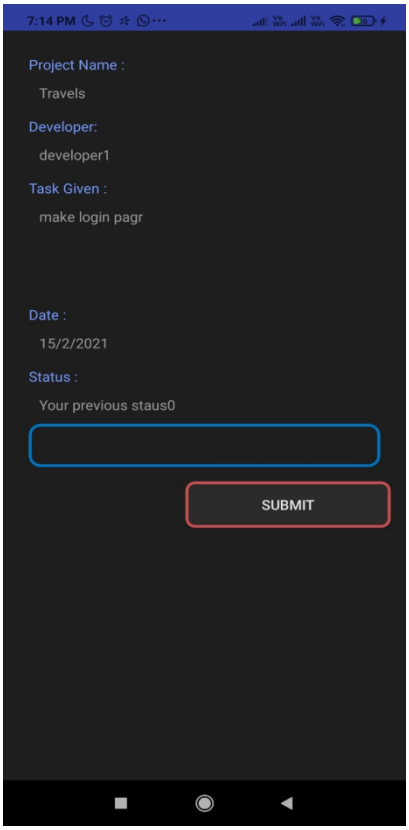


5.3.7 Assign Projects Page

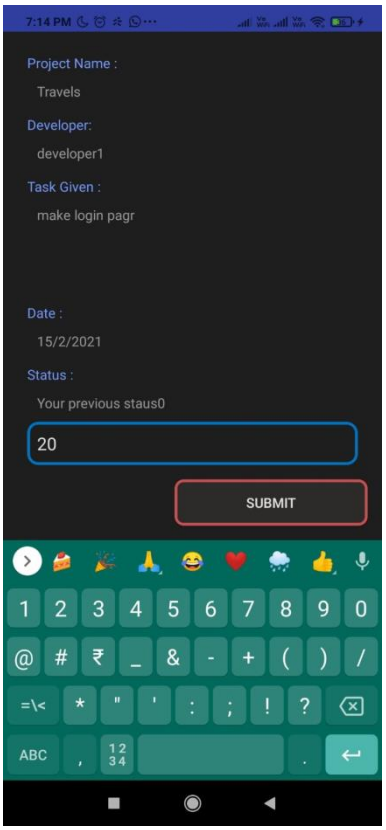


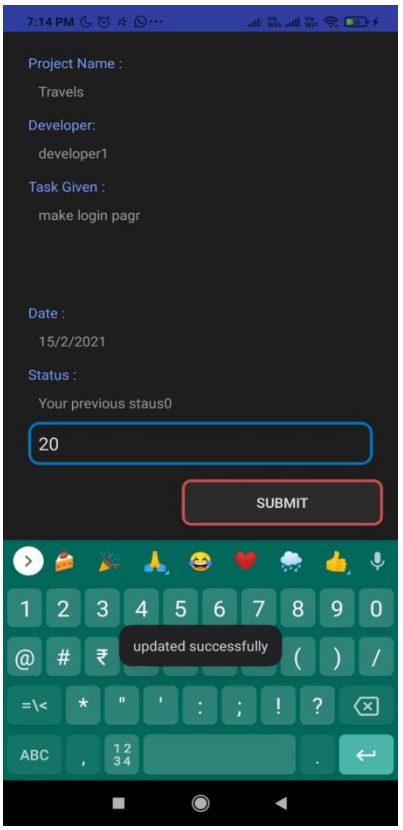
5.3.8 View Status Pages



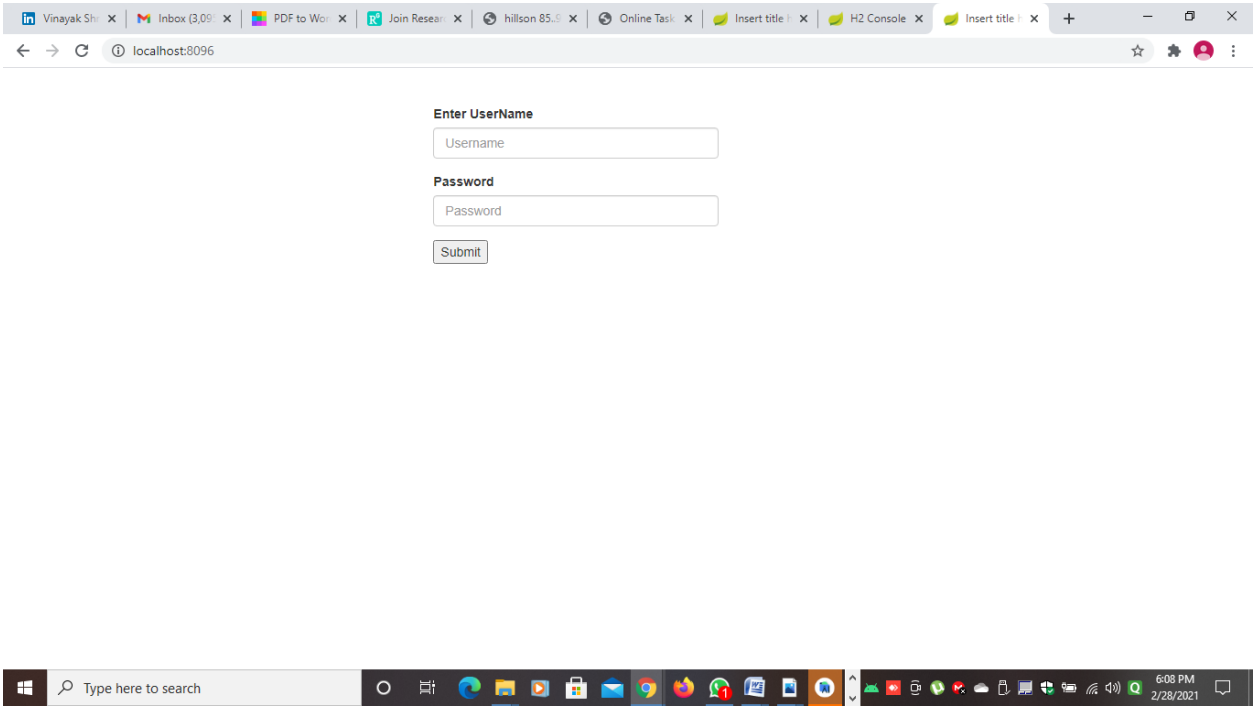


5.3.9 Updating Status Pages

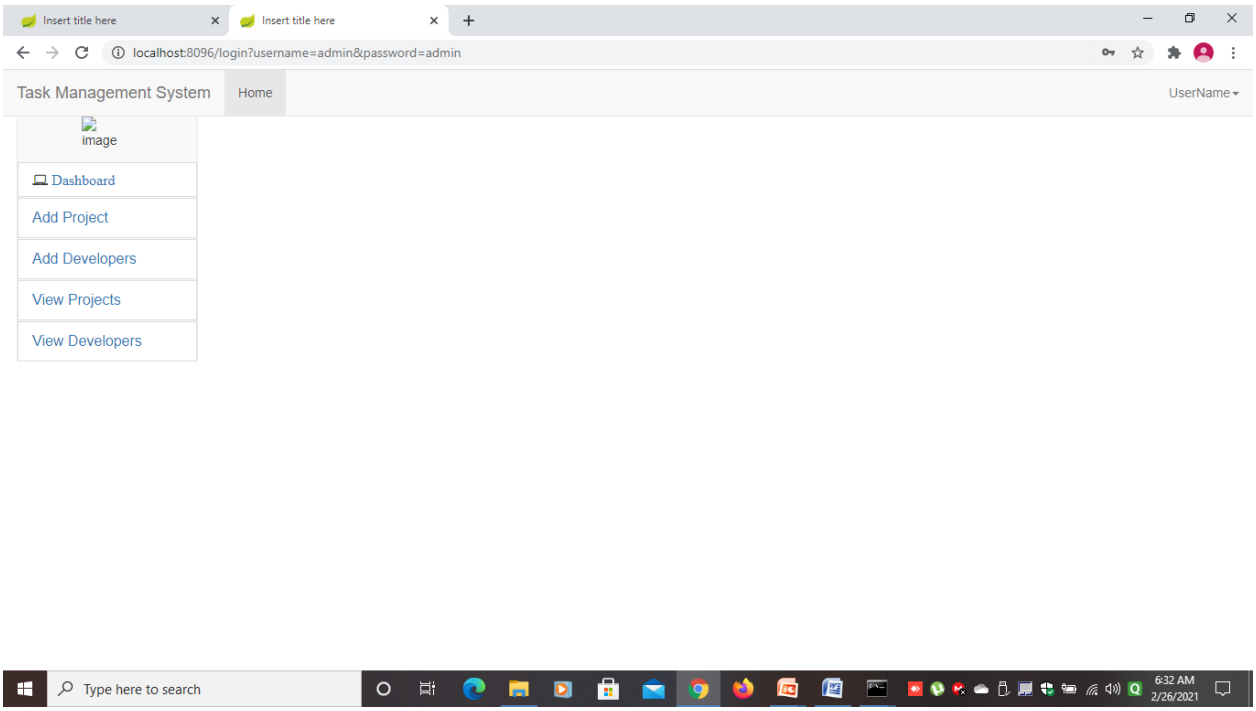




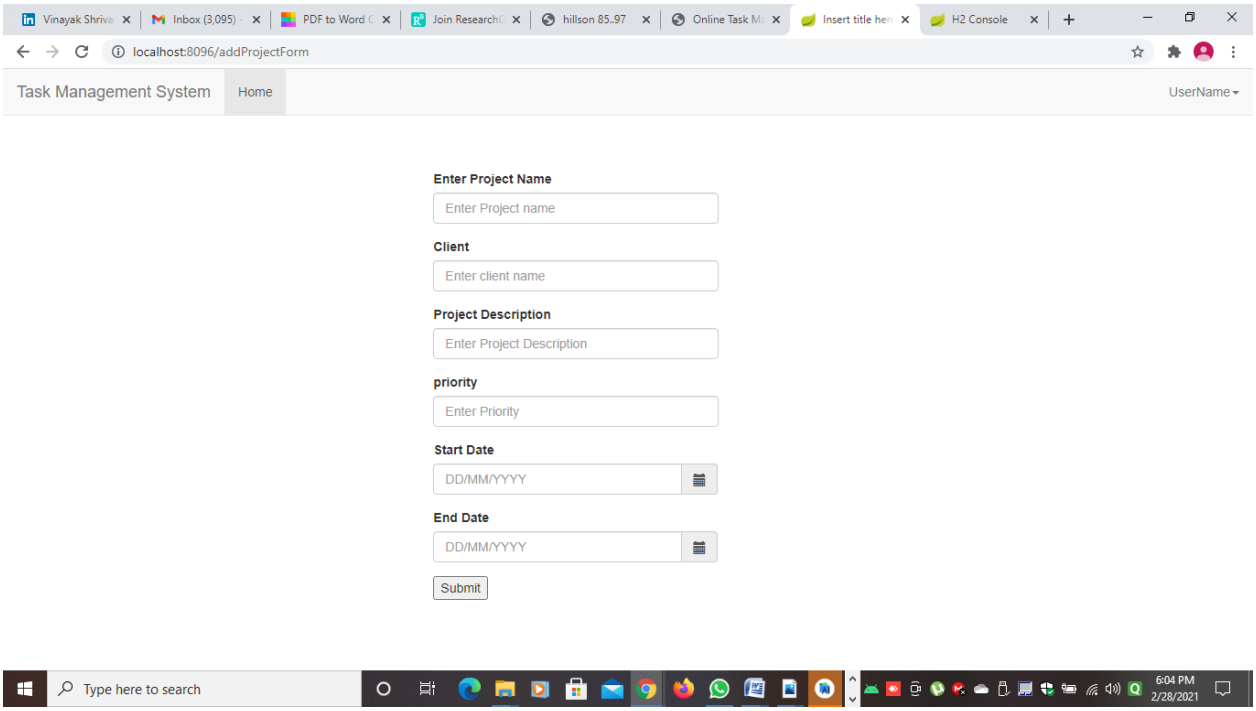
5.3.10 Admin Login Page



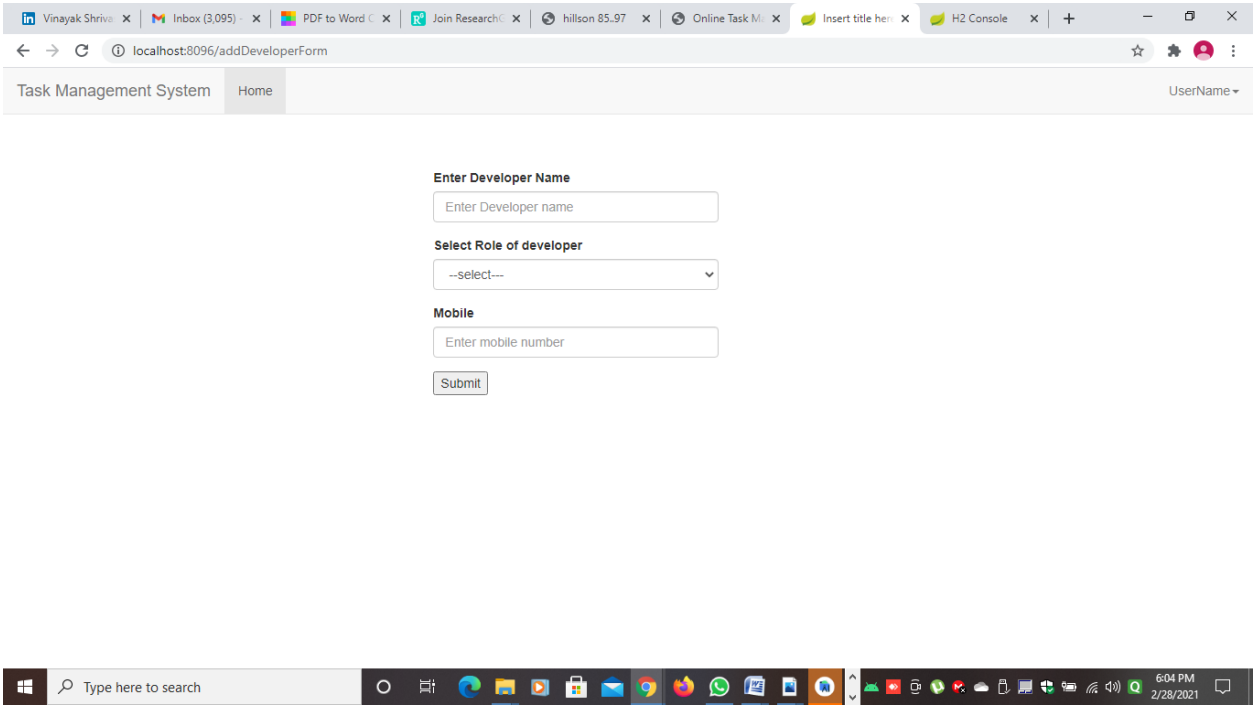
5.3.11 Admin Login Success



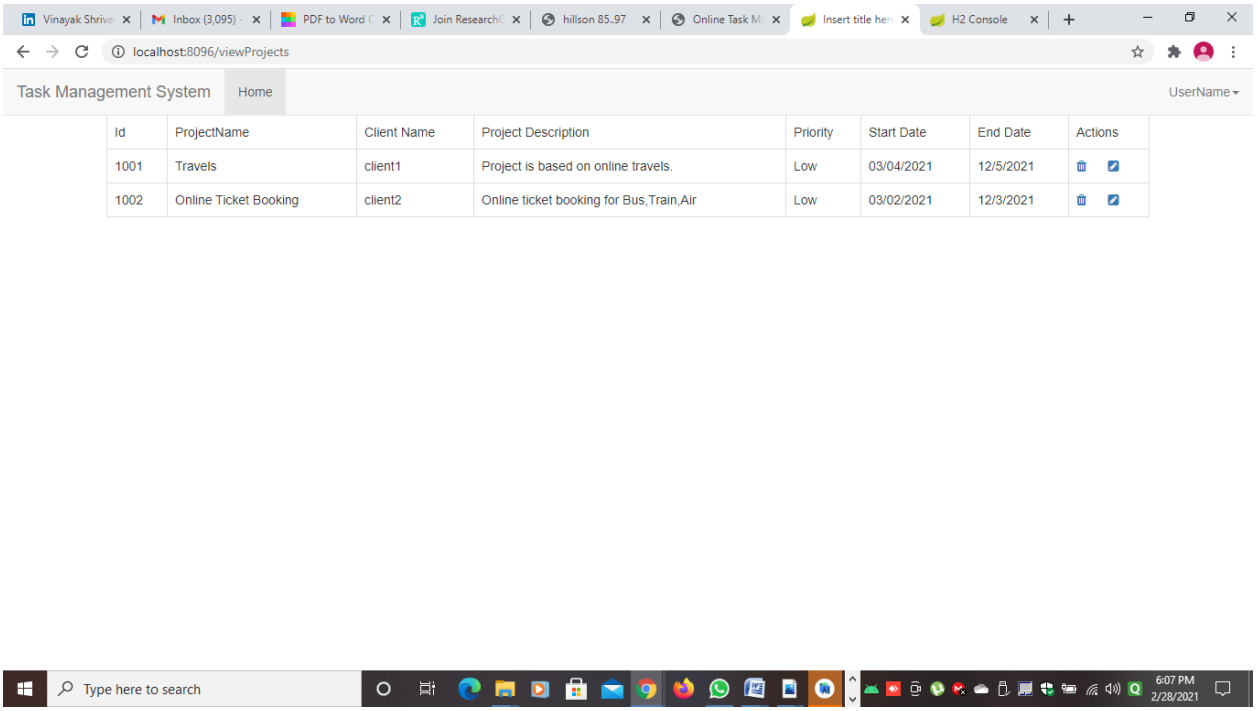
5.3.12 Add Projects Page



5.1.13 Add Developers Page

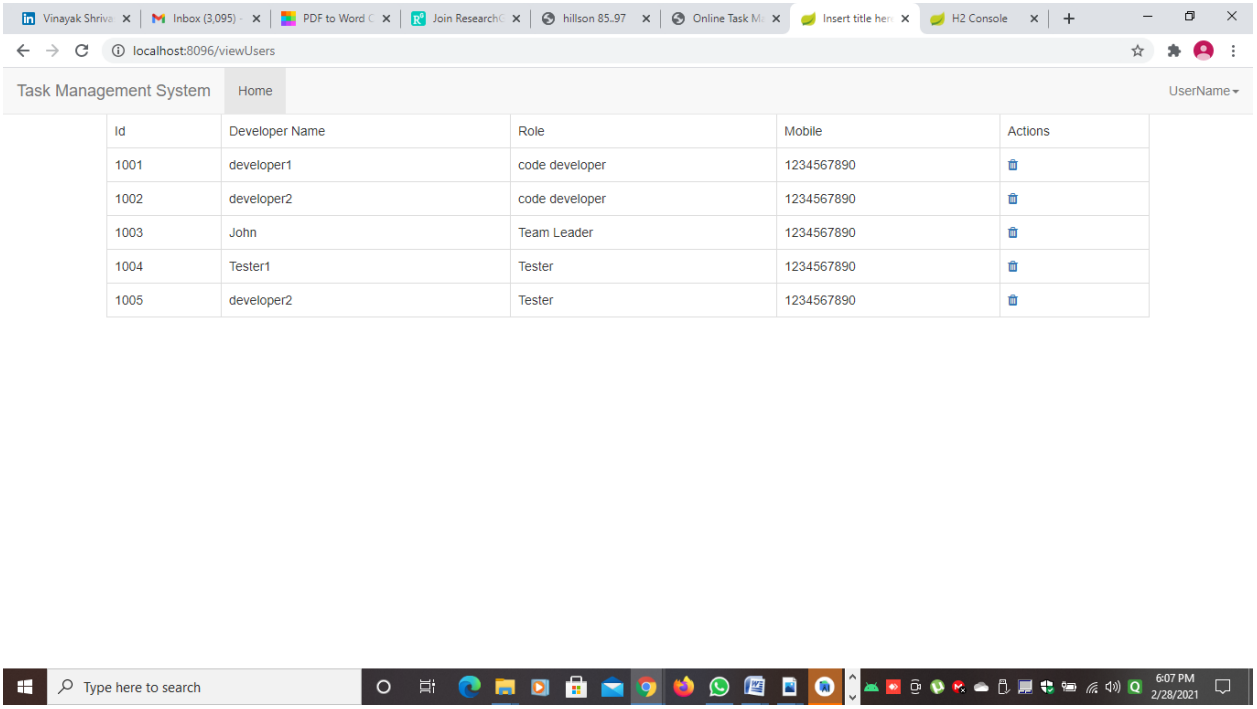


5.3.14 View Projects Page



Id	ProjectName	Client Name	Project Description	Priority	Start Date	End Date	Actions
1001	Travels	client1	Project is based on online travels.	Low	03/04/2021	12/5/2021	
1002	Online Ticket Booking	client2	Online ticket booking for Bus,Train,Air	Low	03/02/2021	12/3/2021	

5.3.15 View Developers Page



CHAPTER 6
SOFTWARE TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product.

6.1 DEFINITION

A process of executing a program with the explicit intention of finding errors, that is making the program fail.

6.2 SOFTWARE TESTING

It is the process of testing the functionality and correctness of software by running it. Process of executing a program with the intent of finding an error.

Software Testing is usually performed for one of two reasons:

- Defect detection
- Reliability estimation

6.3 BLACK BOX TESTING

Applies to software system or module, tests functionality in terms of inputs and outputs at interfaces. Test reveals if the software function is fully operational with reference to requirements specification.

6.4 WHITE BOX TESTING

Knowing the internal workings i.e., to test if all internal operations are performed according to program structures and data structures. To test if all internal components have been adequately exercised.

6.5 SOFTWARE TESTING STRATEGIES

A strategy for software testing will begin in the following order:

- Unit testing
- Integration testing
- System testing
- Validation testing

6.6 UNIT TESTING

It concentrates on each unit of the software as implemented in source code and is a white box oriented. Using the component level design description as a guide, important control paths are tested to uncover errors within the boundary of the module. In the unit testing, the step can be conducted in parallel for multiple components.

6.7 INTEGRATION TESTING

Here focus is on design and construction of the software architecture. Integration testing is a systematic technique for constructing the program structure while at the same time conducting tests to uncover errors associated with interfacing.

The objective is to take unit tested components and build a program structure that has been dictated by design.

6.8 VALIDATION TESTING

In this, requirements established as part of software requirements analysis are validated against the software that has been constructed i.e., validation succeeds when software functions in a manner that can reasonably expected by the customer.

6.9 SYSTEM TESTING:

In this software and other system elements are tested as a whole.

6.9.1 TEST CASE-1:

Test Case 1: User Login	**Priority (H, L): High**
Test Objective: To access the data into the server. **Test Description:** The user can login using credentials display User Home page and can view available data.	
Requirements verified: YES	
Test Environment: Android Application, Server	
Test Setup/Pre-condition: Valid username and password & connectivity to database is essential for this process	
Actions:	**Expected:**
User logins in	Display user home page and can view available data
Pass: Yes **Condition:** NO	**Fail:** NO
Problems/Issues: NIL	
Notes: Successfully executed	

6.9.2 TESTCASE-2:

Test Case 2: Add projects to database	**Priority (H, L): High**
Test Objective: To add projects **Test Description:** This test will ensure addition of project.	
Requirements verified: YES	
Test Environment: Server	
Test Setup/Pre-condition: Valid login and permissions to create a project	
Actions:	**Expected:**
Admin logins in	owner can access the details of the user and the database
Pass: Yes **Condition:** NO **Fail:** NO	
Problems/Issues: NIL	
Notes: Successfully executed	

6.9.3 TESTCASE-3:

Test Case 3: Add tasks to the database	**Priority (H, L): High**
Test Objective: To update the database by adding tasks. **Test Description:** This test will ensure addition of task.	
Requirements verified: YES	
Test Environment: Server	
Test Setup/Pre-condition: Valid login and permissions to create tasks	
Actions:	**Expected:**
Admin logins in	Addition of tasks
Pass: Yes **Condition:** NO **Fail:** NO	

6.9.4 TESTCASE-4:

Test Case 4: Define tasks scheduled in database	**Priority (H, L): High**
Test Objective: To update the database by adding tasks. **Test Description:** This test will ensure addition of task schedule	
Requirements verified: YES	
Test Environment: Server	
Test Setup/Pre-condition: Connectivity to database and task already created in database	
Actions:	**Expected:**
Admin logins in	Defining of tasks
Pass: Yes **Condition:** NO **Fail:** NO	

6.9.5 TESTCASE-5:

Test Case 5: Assigning tasks to employees	**Priority (H, L): High**
Test Objective : Assigning tasks to employees **Test Description:** This test will ensure that tasks can be assigned to the employees	
Requirements verified: YES	
Test Environment: Server	
Test Setup/Pre-condition: Connectivity to database and task and employees created in database	
Actions:	**Expected:**
Admin logins in	Assigning of tasks
Pass: Yes **Condition:** NO **Fail:** NO	

6.9.6 TESTCASE-6:

Test Case 6: Updating Tasks Status	**Priority (H, L): High**
Test Objective : Updating Task Status **Test Description:** This test will ensure validation of data while changing task status	
Requirements verified: YES	
Test Environment: Server	
Test Setup/Pre-condition: Connectivity to database and task already created in the database	
Actions:	**Expected:**
Admin logins in	Updating of tasks
Pass: Yes **Condition:** NO **Fail:** NO	

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENT

7.1 CONCLUSION:

The developed project is a tasks reminder app, with AI-powered Chatbot that will make user enjoy productivity. Whether your goal is to make good habits or get rid of bad ones. This application helps you to make sure that the tasks you set actually get done, with the help of its AI assistant. It monitors your android phone, pings you at times you're likely to see a notification and makes sure you don't forget about items on your to-do list. With this, which is one of the bot characters, this app brings you not only your tasks reminder but also a free productivity assistant that engages you throughout the day to create a more productive version of yourself.

7.2 FUTURE ENHANCEMENT:

Each and every day the government of India requires a new system of updating to update peoples welfare, these sorts of welfare updates will be succeeded by providing a new system of service, a service will be perfected by systematical procedures in this project a new scope of a mobile system has been introduced, although the project has been perfected still all department has to be bring under one shelter like hospital law and civil departments and these system will be brought up by the future development taking time under concern my proposal will stop under particular point

REFERENCES:

[1] Android Studio 2 Development Essentials, Book by Neil Smyth.

[2] PHP and MySQL Web Development, Book by Luke Welling, 2001

[3] D. A. Hillson, "Using a Risk Breakdown Structure in project management", Journal Of Facilities Management, vol. 2, no. 1, pp. 85-97, 2013.

[4] S. McKenna, "Organisational Complexity and Perceptions of Task", Task Management: An International Journal, vol. 3, no. 2, pp. 53-64, 2013.

[5] A. Aleshin, "Time and Risk Management of International Projects", International Journal of Project Management, no. 19, pp. 207-222, 2014

[6] J. Ellis, L. Kvavilashvili, "Prioritization of tasks in 2000: Past present and future directions", Task Management, vol. 14, pp. 1-9, 2000.

[7] Drake Baer, "Dwight Eisenhower Nailed A Major Insight About Productivity", Business Insider, 2014.