

GUJARAT TECHNOLOGICAL UNIVERSITY

Chandkheda, Ahmedabad

Affiliated



GYANMANJARI INSTITUTE OF TECHNOLOGY BHAVNAGAR



A Project report On **Procker**

Under Subject Of
Internship
B.E, Semester – VIII
(Information Technology Branch)

Submitted By
Team Id: 300145

Sr.No.	Name of Student	Enrolment No.
1.	Jyoti Maurya	201290116514

Prof. Vipul Bambhaniya
(Faculty Guide)

Prof. Dhaval R. Chandarana
(Head of Department)

Academic Year
2022-2023

GYANMANJARI INSTITUTE OF TECHNOLOGY
BHAVNAGAR



DEPARTMENT OF I.T. ENGINEERING

CERTIFICATE

This is to certify that the basic process and framework has been satisfactorily carried out by **Ms. Jyoti Maurya** under my guidance in the fulfillment of the course internship during the academic year 2022-23.

Date of submission:

Faculty In-Charge

Internal Examiner

External Examiner

Student Information

Name Of Student:	Jyoti Maurya
Enrollment NO.:	201290116514
Contact No.:	+91 9925059687
Email Id:	<u>jyoti.maurya.it@gmail.com</u>
College Name/Code:	Gyanmanjari Institute of Technology [129]
Branch/Sem.:	Information Technology / 8th
Student's sign:	

Acknowledgement

The internship opportunity I had with Midnight Digital for the duration of 3 months was a great chance for learning and professional development. Therefore, I consider myself fortuitous as I was given a chance to be part of them. I am very grateful for having an opportunity to meet so many amazing people and professionals who led me through this internship.

My sincere thanks to **Ms. Vidhi Patel** for giving me the chance to work in Company for my professional growth. I would like to thank and show my deepest gratitude to **Prof. Vipul Bambhaniya** (Internal Guide) for his careful and valuable guidance which I treasured both theoretically and practically. I also heartily thanks to my other friends pursuing internship in other companies who greatly helped me in my work when I get confused.

Finally, at last but not least, we would like to acknowledge and thanks in large measures to all our fellow friend & guides for their support.

Thank you.

Abstract

This Application is for tracking the projects of the company. On this Application, you can get easy track and check the progress of any project of the company In this website we can check the involvement of a particular employee in a project and also check progress of any employee. Also, this website has functionalities to add, edit and delete users, projects. Also, you can add, edit and delete modules and master modules. Also, it has functionality where you can add project wise modules. This website also contains timesheet functionality where you can add your daily report of work done by you.

List of Table

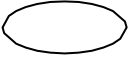
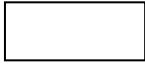




Table Number	Description	Page Number
Table 4.1.1	User Db	17
Table 4.1.2	Project Db	17
Table 4.1.3	Module Db	18
Table 4.1.4	Timesheet Db	18
Table 6.4.1	User registration	42
Table 6.4.2	User Login	42
Table 6.4.3	Add Timesheet	42
Table 6.4.4	Add Project	43
Table 6.4.5	Cart details	43

List of Figures



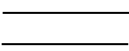
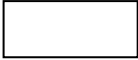
Figure Number	Description	Page Number
Figure 1.1	Midnight Digital	1
Figure 1.2	Snapshot of OrangeHRM Interface	5
Figure 2.1	Incremental Model	9
Figure 3.1	Entity-Relationship Diagram	12
Figure 3.2.1	Context (Level - 0) Diagram	13
Figure 3.2.2	Level-1 DFD	13
Figure 3.3	Use Case Diagram	14
Figure 3.4	Activity Diagram	15
Figure 5.1	Coding Standard	22
Figure 5.1.1	Splash Screen	24
Figure 5.1.2	Login Screen	24
Figure 5.1.3	Projects Screen (admin)	25
Figure 5.1.4	Add Project Screen (admin)	25
Figure 5.1.5	Add Users Screen (admin)	26
Figure 5.1.6	Users Screen	26
Figure 5.1.7	Projects Screen	27
Figure 5.1.8	Add timesheet Screen	27
Figure 5.1.9	Timesheet Added Screen	28
Figure 5.1.10	Projects Screen (admin)	28
Figure 5.1.11	Modules Screen (admin)	29
Figure 5.1.12	Assigned to users (admin)	29
Figure 5.1.13	Profile Screen (admin)	30
Figure 5.1.14	Home Screen	30
Figure 5.1.15	Profile Screen	31
Figure 5.1.16	Logout Screen	31
Figure 6.1	Testing Objective	33
Figure 6.2	Resource Planning	35
Figure 6.3	Testing Automation	36
Figure 6.4	Testing Strategy	37

Symbols and Abbreviations



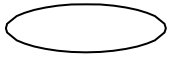

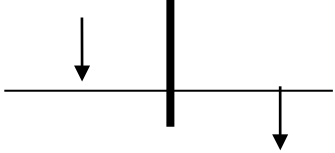
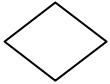
➤ ERD(Entity Relationship diagram):

No.	Symbol	Description
1.		This symbol represents characteristics or attributes of entity
2.		This symbol represents the object or concept that represents important data.
3.		This symbol represents the relationship among entities.
4.		This symbol provides link between attributes and entities.
5.		This symbol represents the multivalued attributes.
6.		This symbol represents the primary key attribute.

➤ DFD (Data flow diagram):

No.	Symbol	Description
1.		This symbol represents the direction of flow of data.
2.		This symbol represents function or process of the data.
3.		This symbol represents the database which stores data.
4.		This symbol represents the living entity of the system.

➤ **Activity diagram:**

No.	Symbol	Description
1.		This symbol represents the START of the activity diagram.
2.		This symbol represents the END of the activity diagram.
3.		This symbol is used to show the activity.
4.		This symbol defines the control flow.
5.		Swim lane defines the way to group activities performed by the same actor in activity diagram.
6.		This symbol defines the decision-making statement or condition in the activity diagram.

Symbol	Description
Procker	Project Tracker

Contents

Chapter 1	Introduction			1
	1.1	Organization Profile		2
	1.2	Project Detail		3
		1.2.1	Project Profile	3
		1.2.2	Project Definition	3
	1.3	Purpose		3
	1.4	Scope		4
	1.5	Objective		4
	1.6	Technology And Literature Review		4
Chapter 2	About The System			6
	2.1	System Requirement Specification		7
	2.2	Feasibility Study		7
	2.3	Project Planning		9
		2.3.1	Project Development Approach	9
		2.3.2	Project Plan	10
Chapter 3	Analysis			11
	3.1	E-R Diagram		12
	3.2	Data Flow Diagram		13
Chapter 4	Design			16
	4.2	Data Dictionary		17

Chapter 5		Implementation	19
	5.1	Implementation Environment	20
	5.2	Security Feature	21
	5.3	Coding Standard	22
	5.4	Results	23
	5.5	Application Screenshots	24
Chapter 6		Testing	32
	6.1	Testing Plan	33
	6.2	Testing Strategy	37
	6.3	Testing Methods	39
	6.4	Test Cases	42
		References	44

CHAPTER-1

INTRODUCTION

1.1 Organization Profile:



Figure 1.1: Midnight Digital

- **Name of the organization:** Midnight Digital
- **External Guide:** Ms. Vidhi Patel
- **Mobile Number:** 9879393936
- **Email:** vidhi@midnight.in
- **Work:** Software Development Company
- **Address:** Swara Park Square, Sanskar mandal Rd, Bhavnagar

1.2 Project Detail

1.2.1 Project Profile

- PROCKER stands for Project Tracker system, is an flutter application.
- The project management application is a software tool designed to assist project managers in organizing, planning, executing, and monitoring project tasks and resources. The application will provide a central location for project information, including timelines, team members, tasks, and milestones. The goal of the application is to help project managers streamline their work, minimize errors, and improve project outcomes. It is aimed to be working as an application that develops the profile for an employee.

1.2.2 Project Definition

- PROCKER is a software tool designed to help project managers and their teams plan, execute, and monitor projects from start to finish.
- The application should provide a centralized platform for collaboration, communication, and data management, allowing project stakeholders to track progress, identify issues, and make informed decisions in real-time.

1.3 Purpose:

- The purpose of a project management application is to provide project stakeholders with the tools and resources they need to successfully manage projects, meet project goals, and deliver high-quality results.
- The application should help simplify and automate project management processes, reducing the time and effort required to complete project tasks.
- The application helps project managers to plan, organize, execute, and monitor project tasks and resources more efficiently. It also helps to ensure that projects are completed within the allocated budget and time frame while meeting quality standards. By providing real-time visibility into project progress, risks, and issues, project managers can make informed decisions and take corrective action when necessary.

1.4 Scope

The scope of a project management application should cover the key areas of project management, including planning, execution, monitoring, and reporting, and provide a comprehensive solution for managing projects from start to finish

1.5 Objective

- The objective of a project management application is to provide project stakeholders with the tools and resources they need to successfully manage projects, meet project goals, and deliver high-quality results.
- The application should help simplify and automate project management processes, reducing the time and effort required to complete project tasks.
- The application should provide a platform for team members to communicate, share files, and collaborate on tasks. It should facilitate teamwork and encourage transparency among project stakeholders.
- By automating repetitive tasks and providing real-time visibility into project progress, the application can help improve overall project efficiency and productivity.
- The application should provide accurate and timely data, analytics, and reports, allowing project managers to make informed decisions and adjust project plans as needed.

1.6 Technology and Literature Review

1.6.1 Literature Review

- Management is not only about managing resources and controlling expenses.
- Although these are basic functions of management, there's more to management than just managing resources and controlling expenses.
- Another extremely important function of management is the ability to manage employees – especially since they are lifeline of any business.
- Given the downturn in the economy, many businesses have not only been forced to lay off employees, but they've also been forced to close their doors due to lack in demand.
- However, there are also businesses that have capitalized on the downturn of the economy to reduce overhead and increase employee productivity even if the need does not exist.
- This can certainly be a temporary fix, but squeezing everything out of employees does not appear to be the solution for long term business success.
- With that in mind, this project will dive into some of the common management issues that businesses and leaders face today such as,
 - The effects of poor leadership,
 - Not motivating employees effectively,
 - Not being able to manage conflict appropriately.
- Based on the study of these management issues, the study concludes that poor leadership can result in not providing proper direction and/or guidance to your employees to meet company goals, not motivating employees can result in decrease in employee productivity, and not managing conflict can lead to low morale.

1.6.1 Review of Related Technology

OrangeHRM

- This is a powerhouse human resources tool that any small or midsize business can benefit from using.
- With OrangeHRM, you have options: You can download and install the system on your own hardware, or you can purchase a hosted solution. To get prices for the hosted solution, you have to contact them from their “Request a Quote page”.
- OrangeHRM's features include: fully modular, addons (e.g., benefits, employee self-service, training, budget, job and salary history, etc.) for purchase, all standard HR functions (employees, leave, benefits, performance, etc.), and more.
- The installation is fairly straight-forward.
- With a self-extracting Windows installer or full-source installations for Windows, Mac, and Linux, you can get OrangeHRM up and running on nearly every platform.
- If you don't have the hardware or the skills to set up Orange onsite, you can request a quote for a hosted instance of OrangeHRM. You can also purchase support plans and customizations.

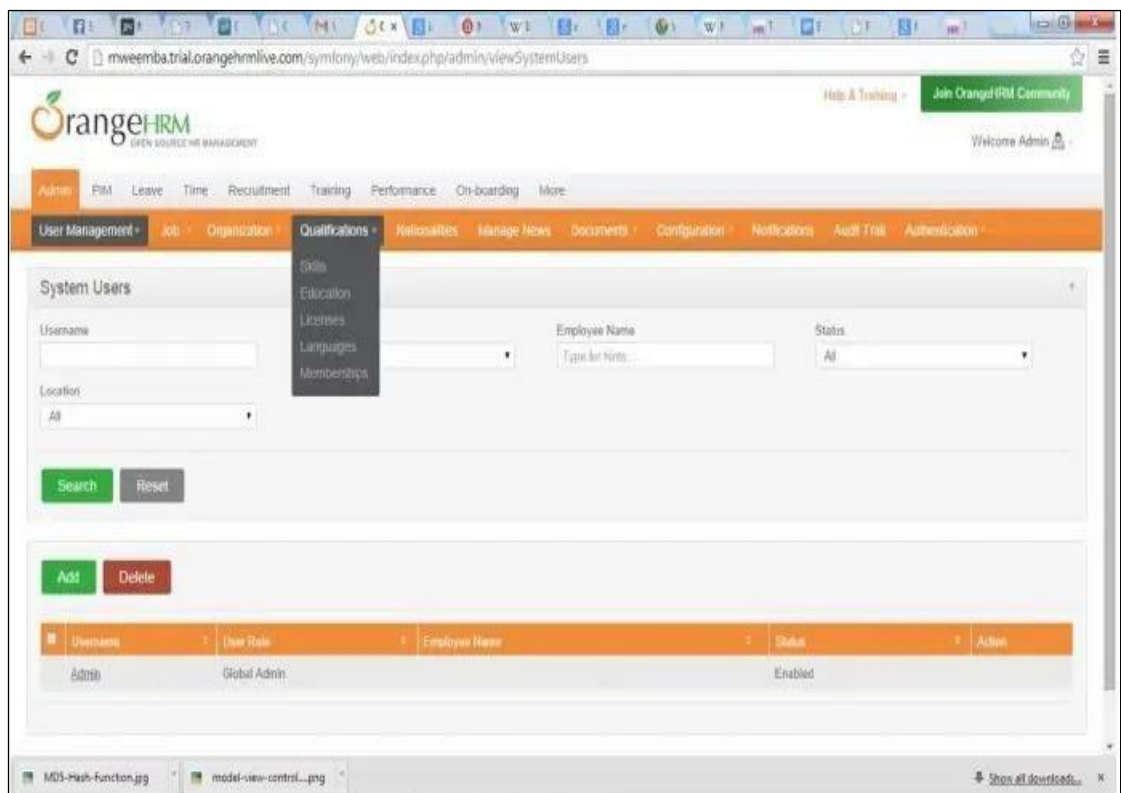


Figure 1.2: Snapshot of OrangeHRM Interface

CHAPTER-2

About The System

2.1 System Requirement Specification:

- **Software Requirements: -**
 - Operating System: Windows 7, Windows 10
 - Front-End: Flutter
 - Back-End: MySQL 8.0
 - IDE: Android Studio
 - Version Control: Git
- **Hardware Requirements: -**
 - CPU Type: Intel Pentium 4
 - RAM Size: 512 MB
 - Hard disk capacity: 40 GB
 - Monitor type: 15 Inch color monitor
 - Mobile: Android/IOS Mobile

2.2 Feasibility Study:

- A feasibility study is an evaluation and analysis of the potential of the proposed project which is based on extensive investigation and research to give full comfort to the decisions makers.
- A feasibility studies main goal is to assess the economic viability of the proposed business.
- Feasibility studies aim to objectively and rationally uncover the strengths and weaknesses of an existing business or proposed venture, opportunities and threats as presented by the environment, the resources required to carry through, and ultimately the prospects for success.
- In its simplest terms, the two criteria to judge feasibility are cost required and value to be attained. Generally, feasibility studies precede technical development and project implementation.
- The system feasibility can be divided into the following sections-
 - **Economic Feasibility:**
 - Economic feasibility means Analysis of a project's costs and revenues in an effort to determine whether or not it is logical and possible to complete.
 - The project is economically feasible as the only cost involved is having a computer with the minimum requirements mentioned earlier.
 - For the users to access the application, the only cost involved will be in getting access to the Internet.

▪ **Technical Feasibility:**

- The technical issues usually raised during the feasibility stage of the investigation includes the following-
 - Does the necessary technology exist to do what is suggested?
 - Do the proposed equipment's have the technical capacity to hold the data required to use the new system?
 - Will the proposed system provide adequate response to inquiries, regardless of the number or location of users?
 - Can the system be upgraded if developed?
 - Are there technical guarantees of accuracy, reliability, ease of access and data security?
- A large part of determining resources has to do with assessing technical feasibility. It considers the technical requirements of the proposed project. The technical requirements are then compared to the technical capability of the organization. The systems project is considered technically feasible if the internal technical capability is sufficient to support the project requirements.
- To deploy the application, the only technical aspects needed are mentioned below:
 - Operating System: Windows 7, Windows 10
 - Front-End: Flutter
 - Back-End: MySQL
- For Users:
 - Internet Connection
 - Mobile Phone
- Time is one of the critical factors in the development of any system and hence proper scheduling is very essential for the timely completion of a project.

▪ **Operational Feasibility:**

- Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization's operating requirements.
- Operational Feasibility aspects of the project are to be taken as an important part of the project implementation.
- Some of the important issues raised are to test the operational feasibility of a project includes the following-
 - Is there sufficient support for the management from the users?
 - Will the system be used and work properly if it is being developed and implemented?
 - Will there be any resistance from the user that will undermine the possible application benefits?
- This system is targeted to be in accordance with the above-mentioned issues.
- Beforehand, the management issues and user requirements have been taken into consideration.
- So, there is no question of resistance from the users that can undermine the possible application benefits.
- The well-planned design would ensure the optimal utilization of the computer resources and would help in the improvement of performance status.

2.3 Project Planning:

2.3.1 Project Development Approach

- Development Approach acts a system of practices, techniques, procedures, and rules used by those who work in a discipline. It allows us to create and evolve the product, service or result during the project life cycle, such as waterfall model, Iterative model, Prototyping model, Spiral model, Agile model, etc.

- **Incremental Model:**

Incremental Model: In this Model, each module passes through the requirements, design, implementation and testing phases.

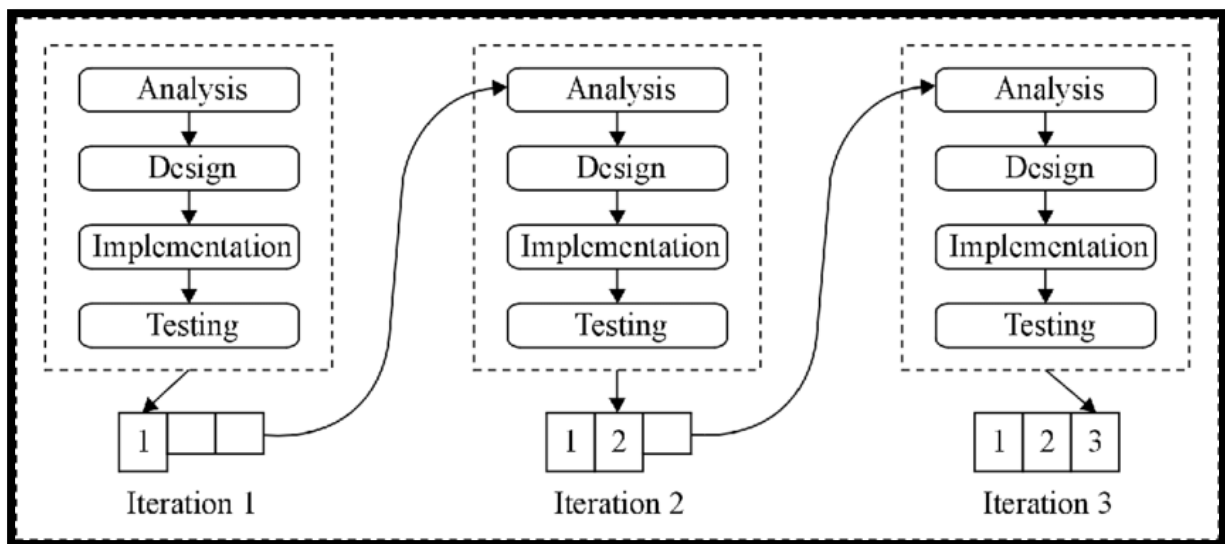


Figure 2.3.1 (Incremental Model)

- In the above figure when we incrementally adding each phase and expect that each phase is fully finished thus keep on adding the phase until it's complete.
- Due to the below benefits we use the “Incremental Model” for our system, reasons are listed below:
 - This Model is more flexible.
 - Less Costly to change Scope and requirements.
 - Using this model customer can respond to each built.
 - It is easier test and debug during a smaller iteration.

2.3.2 Project Plan

- At the very commencement, we proceeded to a decision to carry out the development of my task into the following steps:
 - Exploring the available development environments and techniques.
 - Database Analyzing.
 - Database design and Implementation.
 - Program's Structure Analyzing.
 - GUI (Graphical User Interface) constructing———
 - Bringing all the stuff together (controls data binding and functions implementation).
 - Tests.
- Each one of these steps could be explained in some brief details as follows:
 - Exploring the available development environments and techniques-

There is a lot of programming environments available to be used for such kind of elaborations. The point is to choose such an environment that we will be able to operate with in a convenient and easy way. This is more or less optional and individual process, that depends on the developer's experience as well.
 - Database Analyzing-

It concerns all of the demands, put upon the database content and its functionality. The database should be designed and implemented in a way that the user would expect it to be.
 - Database design and Implementation-

This step is tightly related with the previous one as it is completely determined by the requirements, analyzed and discussed in step2.
 - Program's Structure Analyzing-

The application program as an interface between the users and the database should be an accurate "reflection" of the database on the screen; hence a well analyzed and defined structure is needed.
 - GUI Constructing-

After analyzing the program's structure and defining what it should consist of, a graphical representation of this stuff is needed in order to enable the user to interact with the data.
 - Bringing all the stuff together-

The next step that should be taken is connecting the program with the database and performing the necessary functionality upon all of the controls.
 - Tests-

To ensure that everything works properly and as it has been expected, test performance has to be done upon the system's functionality.

CHAPTER-3

Analysis

3.1 E-R Diagram:

- Upper mentioned diagram depicts entity relationship diagram that shows association between different entities of the system

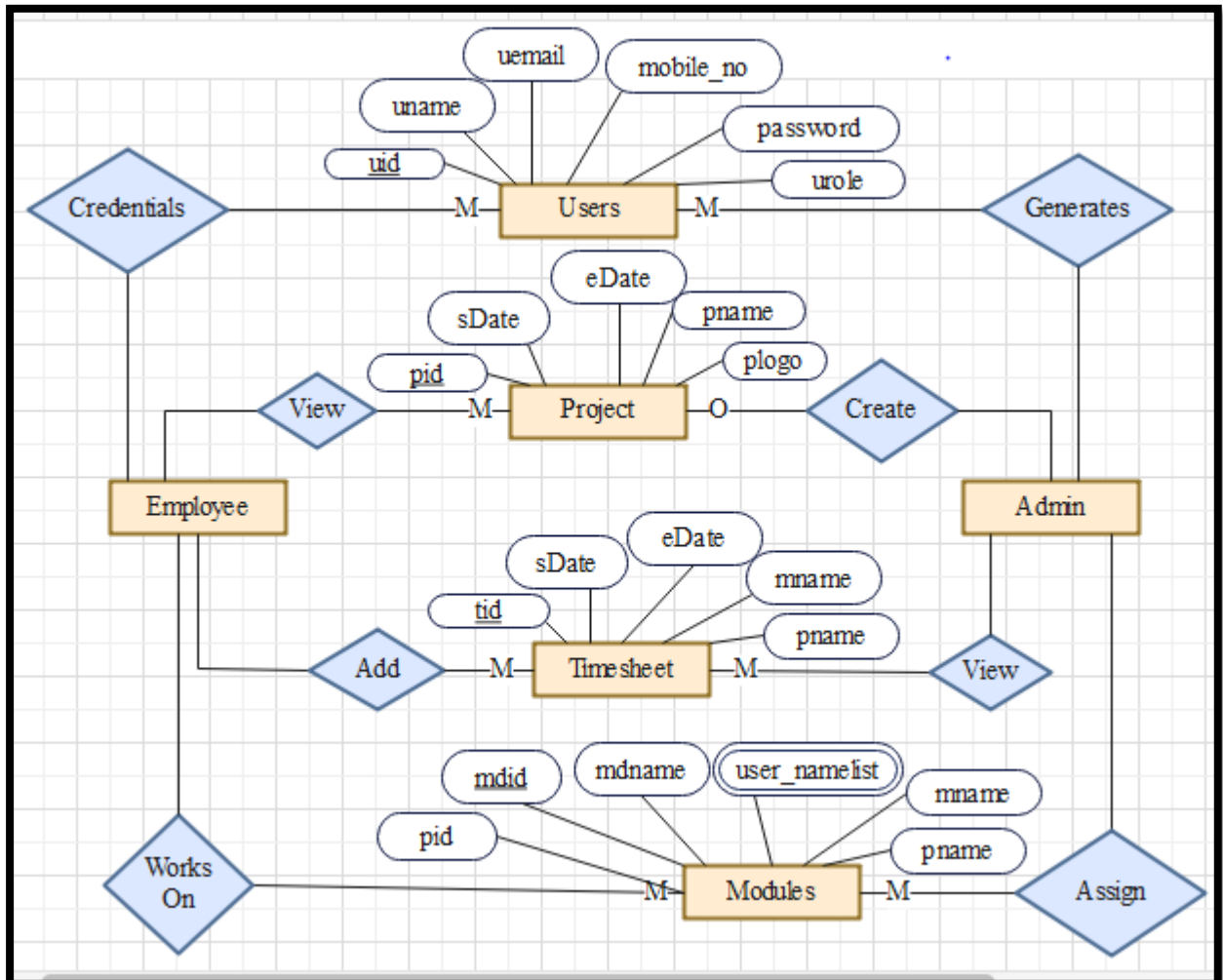


Figure 3.1: Entity-Relationship Diagram

3.2 Data Flow Diagram:

- **Context (Level-0) Diagram:**

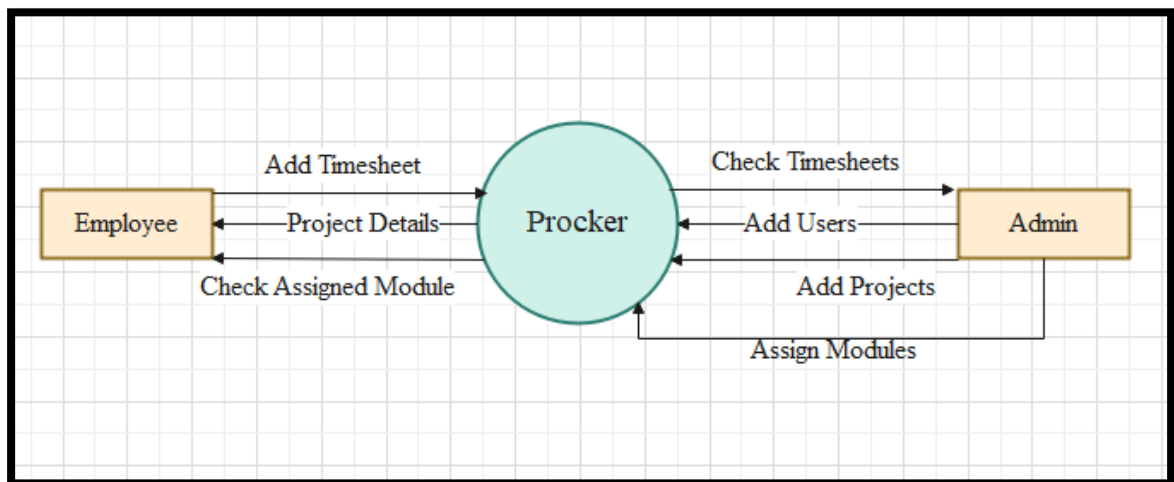


Figure 3.2.1: Context (Level-0) Diagram

- **Level-1 Diagram of Organization/Admin:**

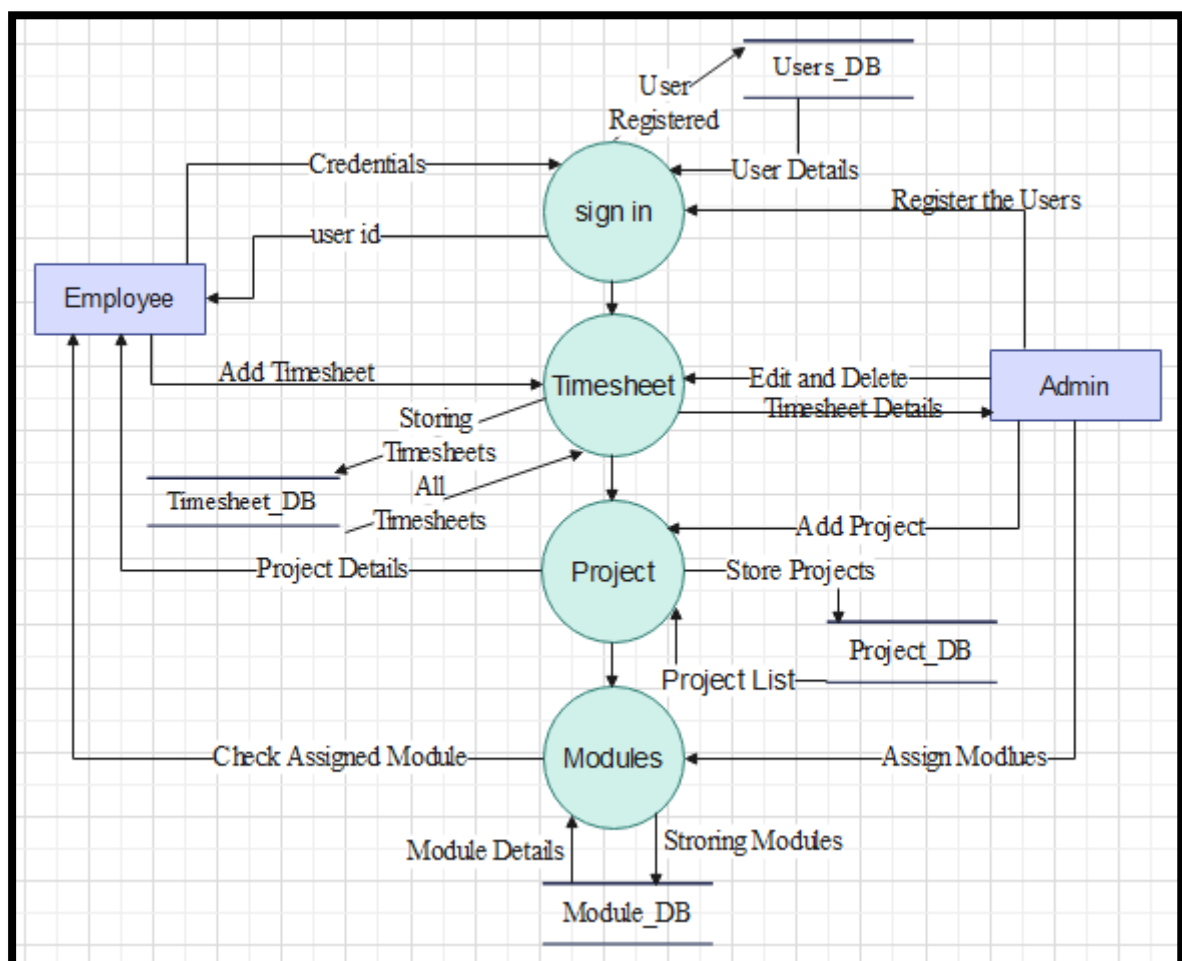


Figure 3.2.2: Level-1 DFD

3.3 Use Case Diagram:

- It is the graphical depiction of the admin and user's possible interaction with the system.

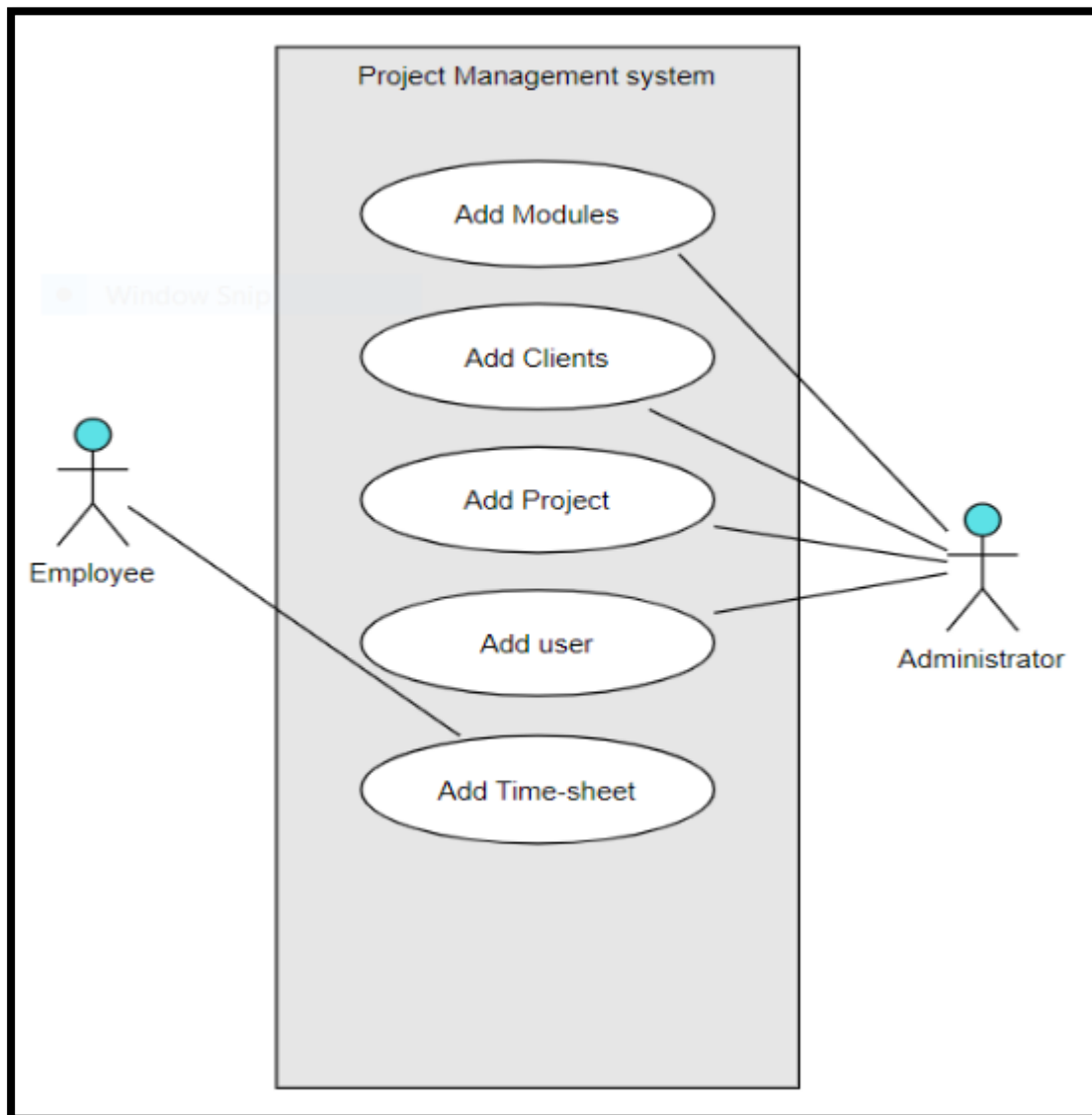


Figure 3.3: Use Case Diagram of System

3.4 Activity Diagram:

- Admin/User will initiate the activity through Login.
- If Login details are valid, Admin/User will redirect to their respective main page.

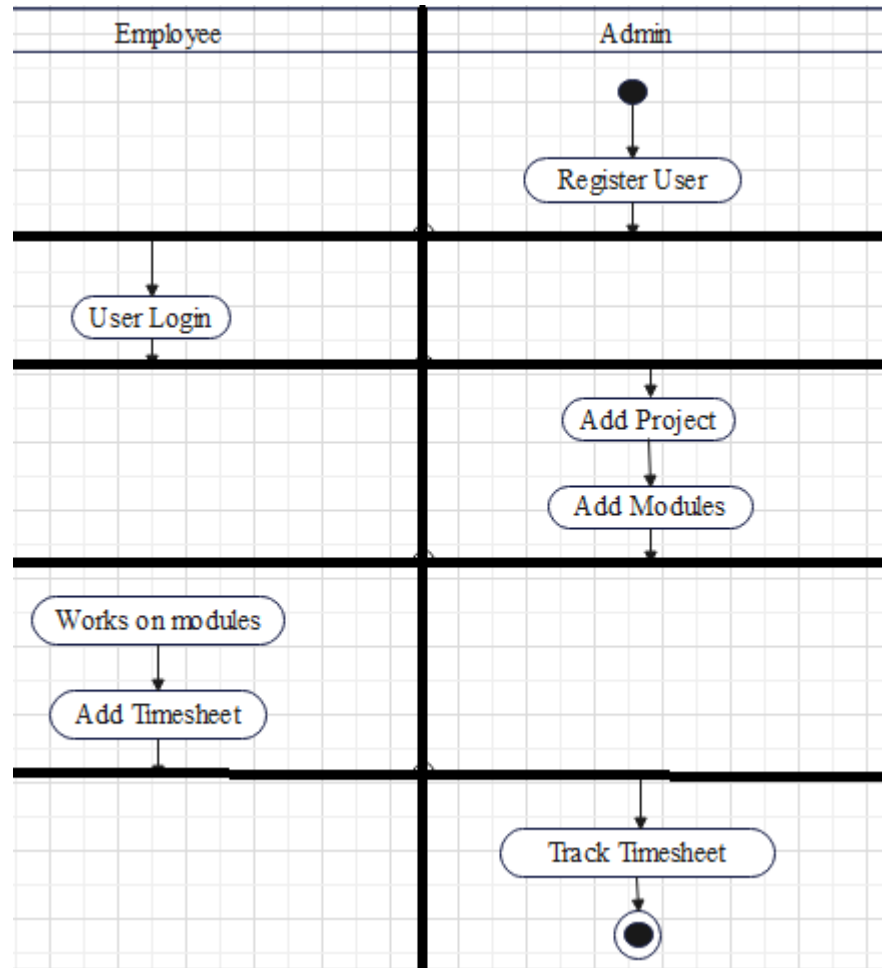


Figure 3.4: Activity Diagram of System

CHAPTER-4

Design

4.1 Data Dictionary:

1. user_db:

This database is used to store the data about the users at the time of registration and fetch details from this at the time of Log- in.

Sr_no	Field_name	Datatype	Field_length	Constraint	Description
1.	u_id	Integer	11	Primary key, Autoincrement	Unique ID User
2.	uname	Varchar	20	Not Null	User Name
3.	u_email	Varchar	20	Not Null	User Email
4.	u_mobile	Integer	10	Not Null	Mobile Number
5.	u_role	Varchar	12	Not Null	User Role

Table 4.1.1 user_db

2. project_db:

This database is used to store the data about the projects at the time of project creation and fetch details the from user side to view project details.

Sr_no	Field_name	Datatype	Field_length	Constraints	Description
1.	p_id	Integer	11	Primary key, Autoincrement	Unique ID Project
2.	sdate	Date	-	Not Null	Start Date
3.	edate	Date	-	Not Null	End Date
4.	pname	Varchar	50	Not Null	Project Name
5.	plogo	longblob	-	Not Null	Project Logo

Table 4.1.2 project_db

3. module_db:

module_db is used to store details of modules.

Sr_no	Field_name	Datatype	Field_length	Constraints	Description
1.	md_id	Integer	11	Primary key	ID of module
2.	pid	Integer	10	Foreign Key	ID of Project
3.	starttime	Date	-	Not Null	Starting Date of module
4.	endtime	Date	-	Not Null	Ending Date of module
5.	user_namelist	Varchar	50	Not Null	List Of Users
6.	pname	Varchar	30	Not Null	Project Name
7.	mname	Varchar	30	Not Null	Module Name

Table 4.1.3 module_db

4. timesheet_db:

timesheet_db is used to store details of modules's timelist.

Sr_no	Field_name	Datatype	Field_length	Constrains	Description
1.	ts_id	Integer	11	Primary key, AutoIncremet	Unique ID Of timesheet
2.	p_id	Integer	11	Foreign key	Unique ID of Project
3.	m_id	Integer	11	Foreign key	Unique ID of module
4.	ts_date	Date	-	Not Null	Date of timesheet
5.	sdate	Date	-	Not Null	Start Date of timesheet
6.	edate	Date	-	Not Null	End Date of timesheet
7.	pname	Varchar	50	Not Null	Project Name
8.	mname	Varchar	50	Not Null	Module Name

Table 4.1.4 timesheet_db

CHAPTER-5

Implementation

5.1 Implementation Environment:

- The **Systems Development Life Cycle (SDLC)** is a software engineering framework that is used to describe the various phases used to develop an information system.
- These phases include planning, analysis, design, development, testing, and implementation.
- SDLC environments describe the activities and tools required to perform a particular process within the SDLC.
- They are also defined as controlled points where software engineers can carry out activities related to development, testing, installation, and configuration.
- These environments are associated with the different phases that make up the SDLC.
- The main SDLC environments include:
 - The analysis and design environment
 - The development environment
 - The common build environment
 - The testing environment, which has two components:
 - The systems integration testing environment
 - The user acceptance testing environment
 - The production environment
- Analysis & Design Environment-

The analysis and design environment is aligned to the **planning and analysis phases** of the SDLC. In this environment, the main processes that take place include carrying out an in-depth examination of the current system and the proposed system. The system architecture is also defined and includes developing the design of the hardware, software, and network requirements for the system. Within this environment, systems and business analysts work closely with software engineers.

- Development & Common Build Environment-

The **development** environment is aligned to the development phase of the SDLC. This is where processes related to software development are carried out. The development environment contains a set of different processes and tools for programming. These are used to develop the final software.

The development environment can also be a physical space where development takes place and where software engineers interact. Another example of the development environment is the **integrated development environment (IDE)**. The IDE provides a platform where tools and development processes are coordinated in order to provide software engineers a convenient way of accessing the resources they require during the development process.

The **common build environment** is closely aligned to the development phase of the SDLC. In this environment, software engineers merge the work done in the development environment. Within this environment, software engineers build systems. These are used to automate the process of software compilation.

- Testing Environment-

The **testing** environment is closely aligned to the testing phase of the SDLC. The testing environment comprises the following components: the System Integration Testing Environment and the User Acceptance Testing Environment.

- Production Environment-

Production is the final environment in your software development process. It is the work that it ready to be publicly available, and only the most thoroughly tested code should end up here.

While most people associate the production environment with the product being live, this is not necessarily the case. Until the URL is actually shared with the public, your production site can instead act as a platform where the public-ready code is kept.

5.2 Security Feature:

- Secure software development includes enabling software security (security requirements planning, designing a software architecture from a security perspective, adding security features, etc.) and maintaining the security of software and the underlying infrastructure (source code review, penetration testing).
- Application security is the process of developing, adding, and testing security features within applications to prevent security vulnerabilities against threats such as unauthorized access and modification.
- Application security is important because today's applications are often available over various networks and connected to the cloud, increasing vulnerabilities to security threats and breaches. There is increasing pressure and incentive to not only ensure security at the network level but also within applications themselves. One reason for this is because hackers are going after apps with their attacks more today than in the past. Application security testing can reveal weaknesses at the application level, helping to prevent these attacks.
- Different types of application security features include authentication, authorization, encryption, logging, and application security testing. Developers can also code applications to reduce security vulnerabilities.

- **Authentication:**

When software developers build procedures into an application to ensure that only authorized users gain access to it. Authentication procedures ensure that a user is who they say they are. This can be accomplished by requiring the user to provide a user name and password when logging in to an application. Multi-factor authentication requires more than one form of authentication—the factors might include something you know (a password), something you have (a mobile device), and something you are (a thumb print or facial recognition).

- **Authorization:**

After a user has been authenticated, the user may be authorized to access and use the application. The system can validate that a user has permission to access the application by comparing the user's identity with a list of authorized users. Authentication must happen before authorization so that the application matches only validated user credentials to the authorized user list.

- **Encryption:**

After a user has been authenticated and is using the application, other security measures can protect sensitive data from being seen or even used by acybercriminal. In cloud-based applications, where traffic containing sensitive data travels between the end user and the cloud, that traffic can be encrypted tokeep the data safe.

- **Logging:**

If there is a security breach in an application, logging can help identify who got access to the data and how. Application log files provide a time- stamped record of which aspects of the application were accessed and by whom.

- **Application security testing:**

A necessary process to ensure that all of these security controls work properly.

5.3 Coding Standard:

- A coding standard gives a uniform appearance to the codes written by different engineers.
- It improves readability, and maintainability of the code and it reduces complexity also.
- It helps in code reuse and helps to detect error easily.
- It promotes sound programming practices and increases efficiency of the programmers.
- The following are some representative coding standards:

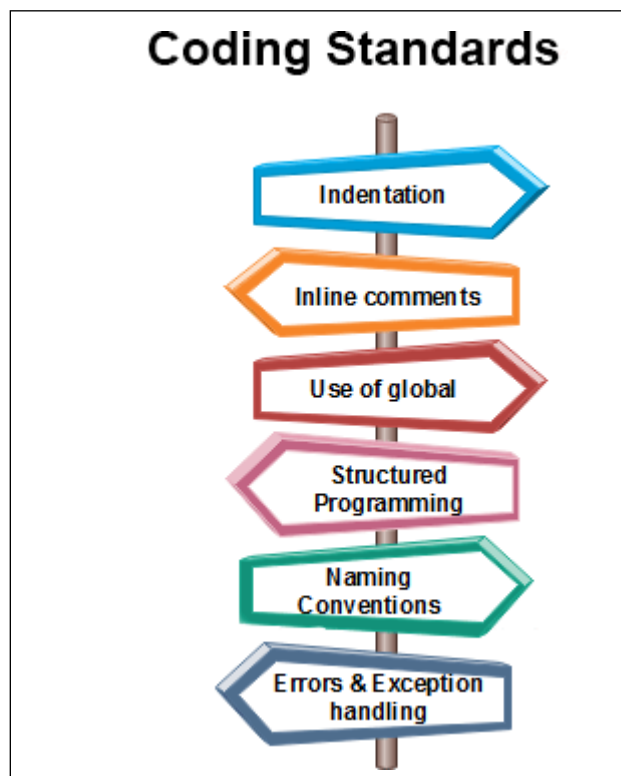


Figure 5.1: Coding Standard

- **Indentation:**

Proper and consistent indentation is essential in producing easy to read and maintainable programs.

Indentation should be used to:

- Emphasize the body of a control structure such as a loop or a select statement.
- Emphasize the body of a conditional statement
- Emphasize a new scope block

- **Inline comments:**

Inline comments analyze the functioning of the subroutine, or key aspects of the algorithm shall be frequently used.

- **Rules for limiting the use of global:**

These rules file what types of data can be declared global and what cannot.

- **Structured Programming:**

Structured (or Modular) Programming methods shall be used. "GOTO" statements shall not be used as they lead to "spaghetti" code, which is hard to read and maintain, except as outlined line in the FORTRAN Standards and Guidelines.

- **Naming conventions for global variables, local variables, and constant identifiers:**

A possible naming convention can be that global variable names always begin with a capital letter, local variable names are made of small letters, and constant names are always capital letters.

- **Error return conventions and exception handling system:**

Different functions in a program report the way error conditions are handled should be standard within an organization. For example, different tasks while encountering an error condition should either return a 0 or 1 consistently.

5.4 Results:

- The project takes shape during the implementation phase. This phase involves the construction of the actual project result. Programmers are occupied with encoding, designers are involved in developing graphic material, contractors are building, the actual reorganisation takes place. It is during this phase that the project becomes visible to outsiders, to whom it may appear that the project has just begun. The implementation phase is the doing phase, and it is important to maintain the momentum.
- At the end of the implementation phase, the result is evaluated according to the list of requirements that was created in the definition phase. It is also evaluated according to the designs. For example, tests may be conducted to determine whether the web application does indeed support Explorer 5 and Firefox 1.0 and higher. It may be determined whether the trim on the building has been made according to the agreement, or whether the materials that were used were indeed those that had been specified in the definition phase.
- This phase is complete when all of the requirements have been met and when the result corresponds to the design.

5.5 Application Screenshots



Figure 5.5.1 Splash Screen

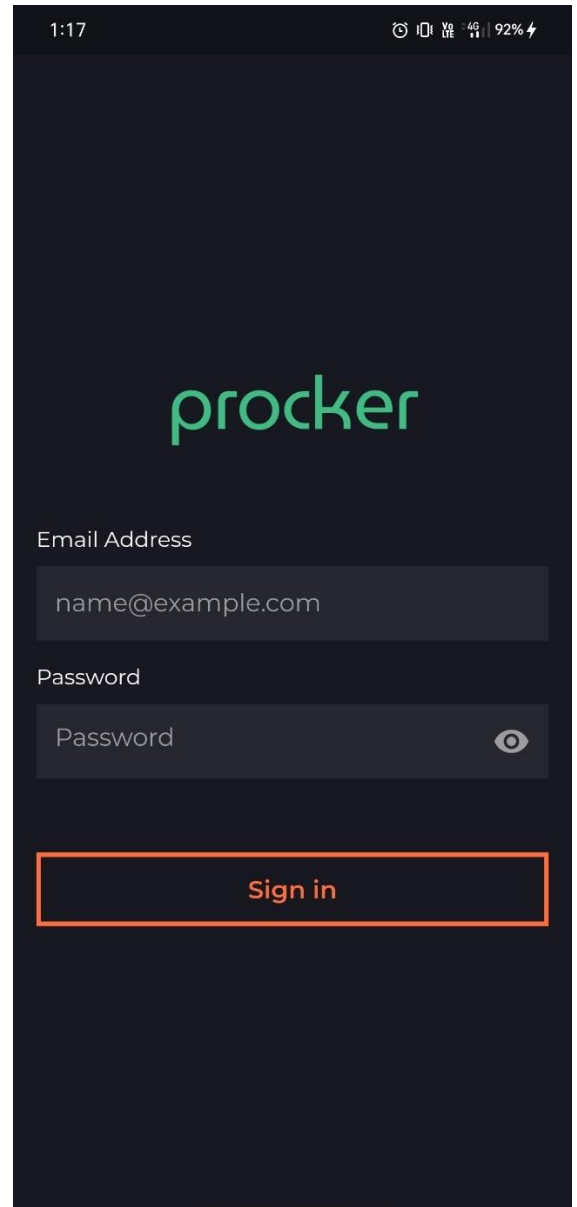


Figure 5.5.2 Login Screen

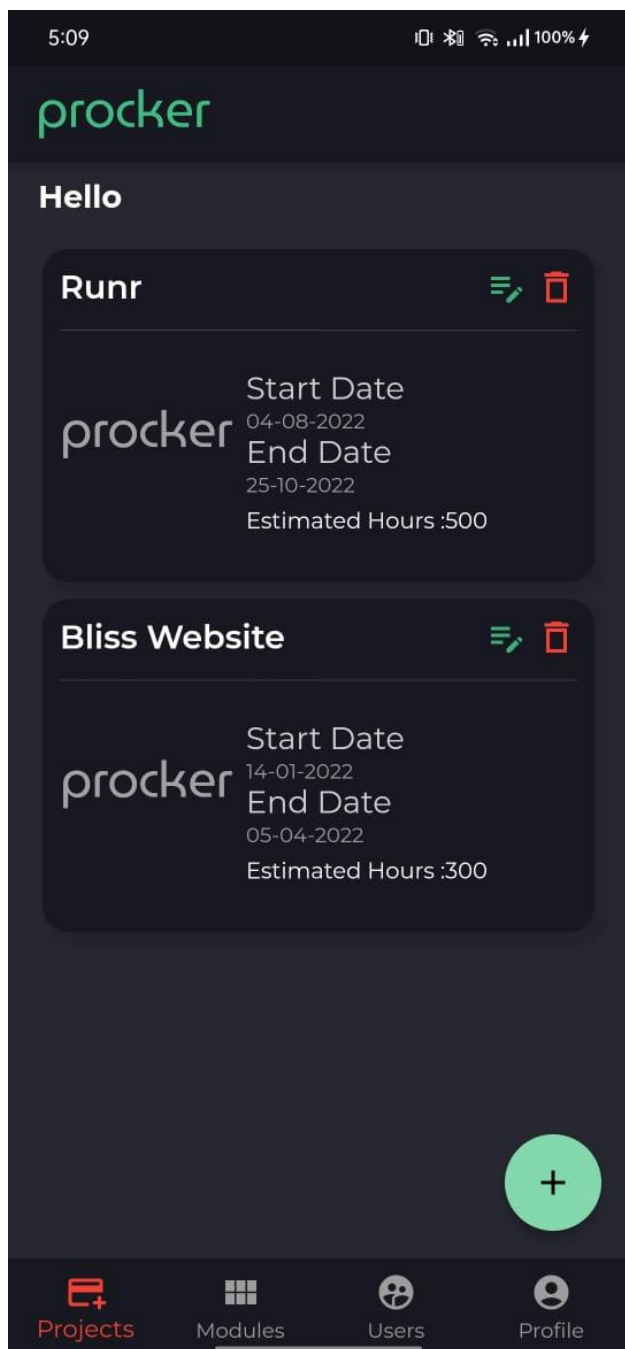


Figure 5.5.3 Projects Screen (admin)

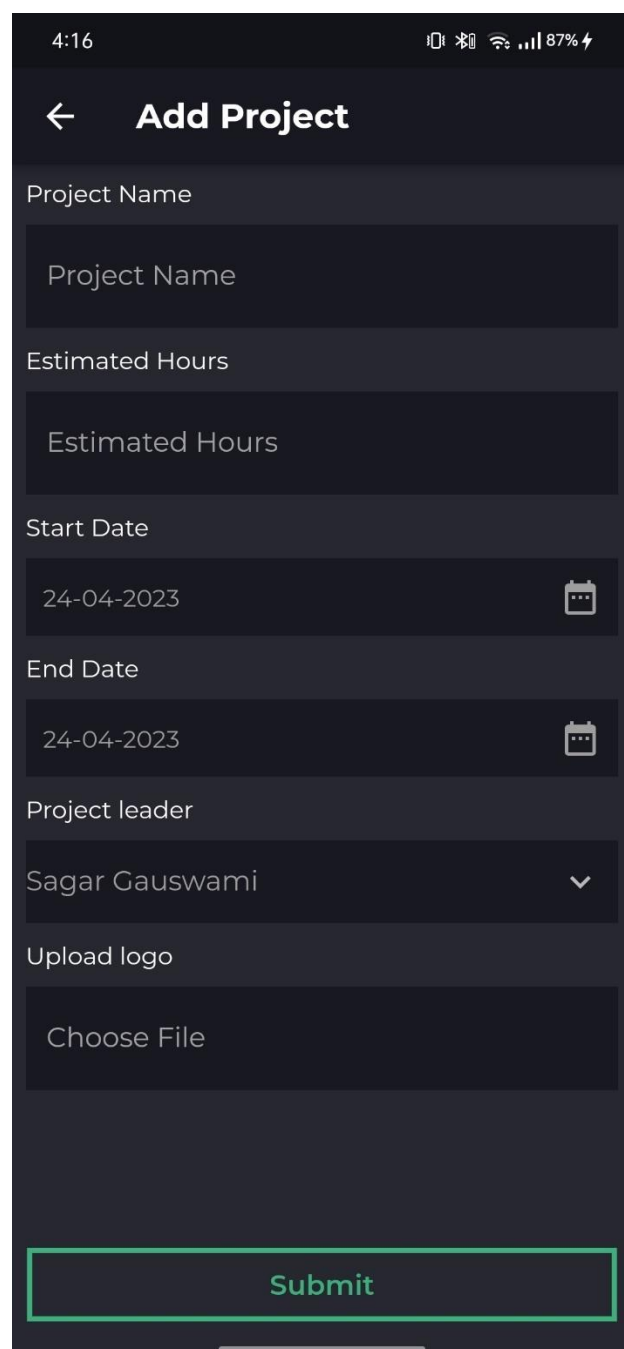


Figure 5.5.4 Add project Screen (admin)

4:22 88%

← Add User

Name

Name

Mobile No.

Mobile No.

Email

Email

Role

Super Admin

Submit

Figure 5.5.5 Add users Screen (admin)

4:22 88%

procker

1	Midnight Admin Super Admin admin@gmail.com	▼
2	Trushit Gadhvi Developer 1234@gmail.com	▼
3	Tushar Dabhi Developer tushar@themidnight.in	▼
4	Prithvirajsinh Gohil Developer prithviraj.midnight@gmail.com	▲

Edit Delete

+

Projects Modules Users Profile

Figure 5.5.6 Users Screen (admin)

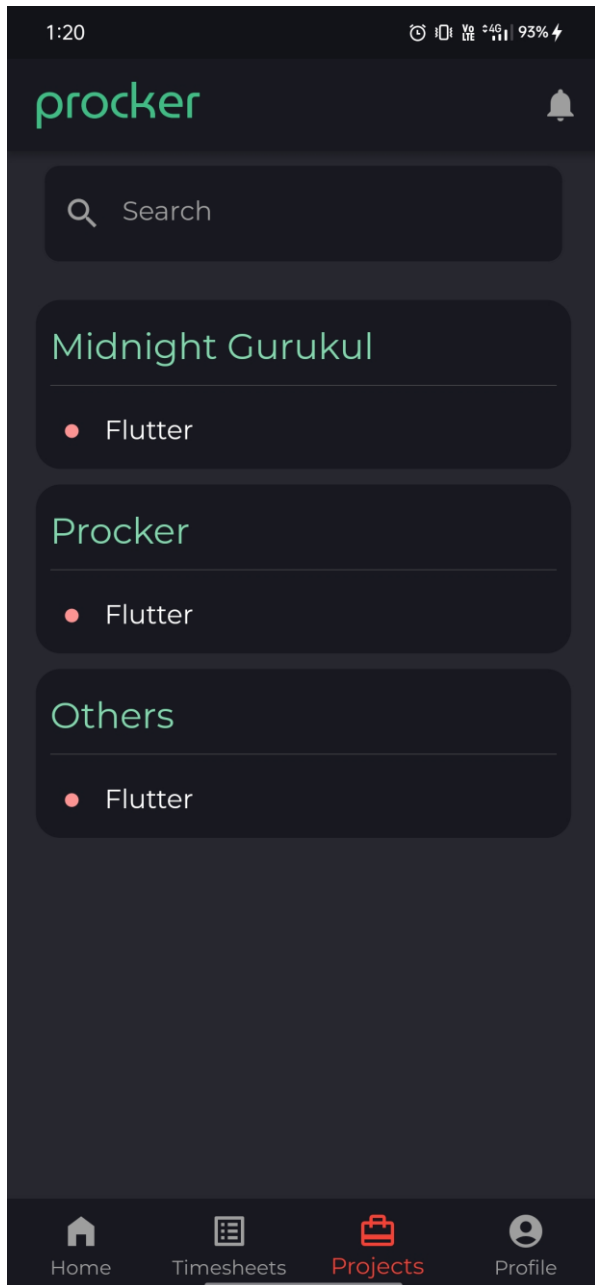


Figure 5.5.7 Projects Screen

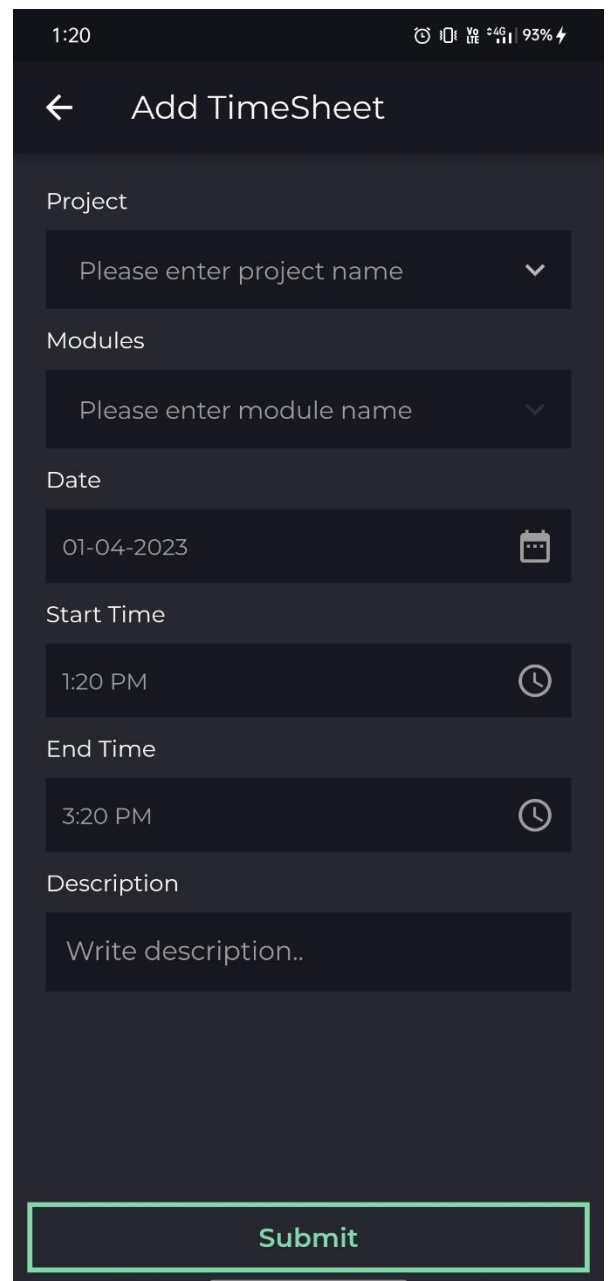


Figure 5.5.8 Add timesheet Screen

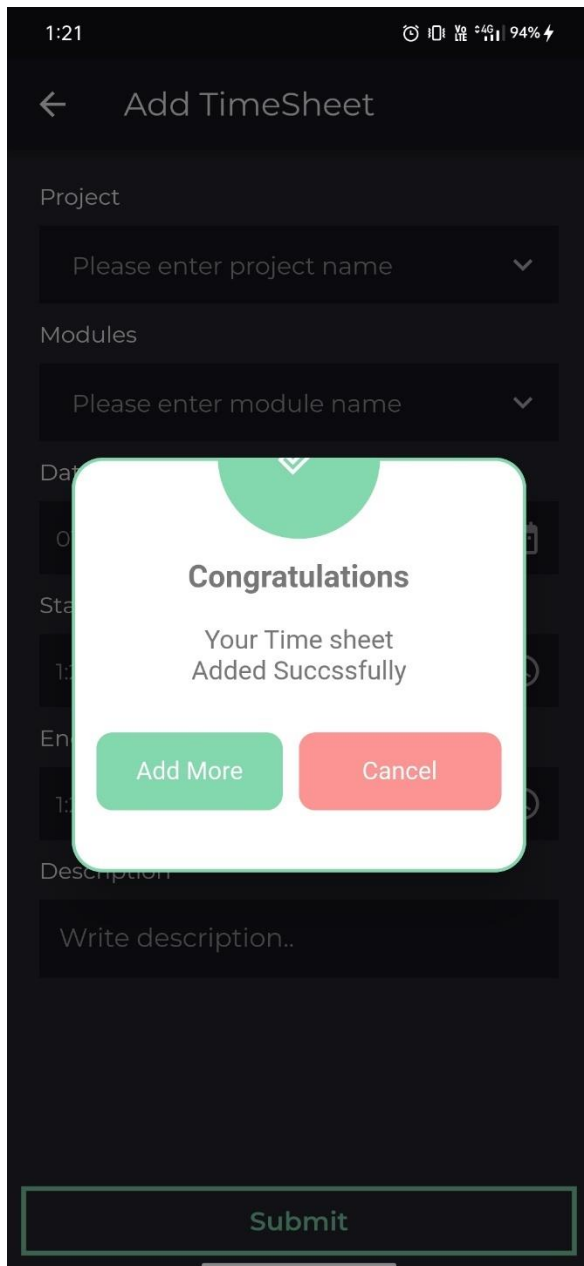


Figure 5.5.9 Timesheet added Screen

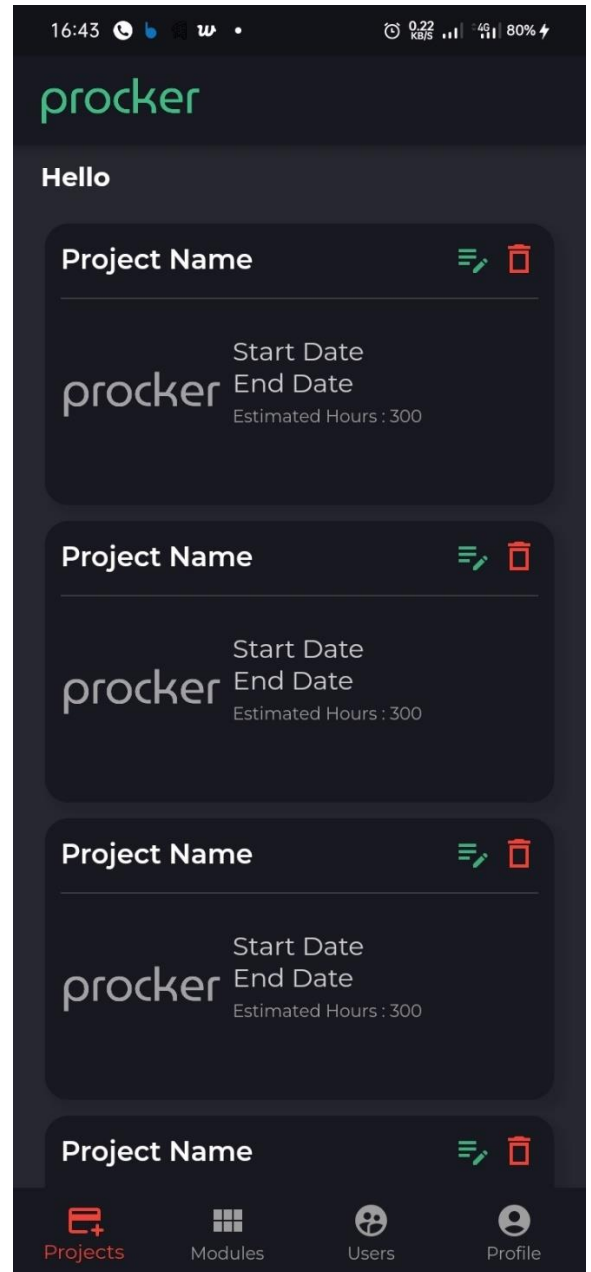


Figure 5.5.10 Projects Screen (admin)

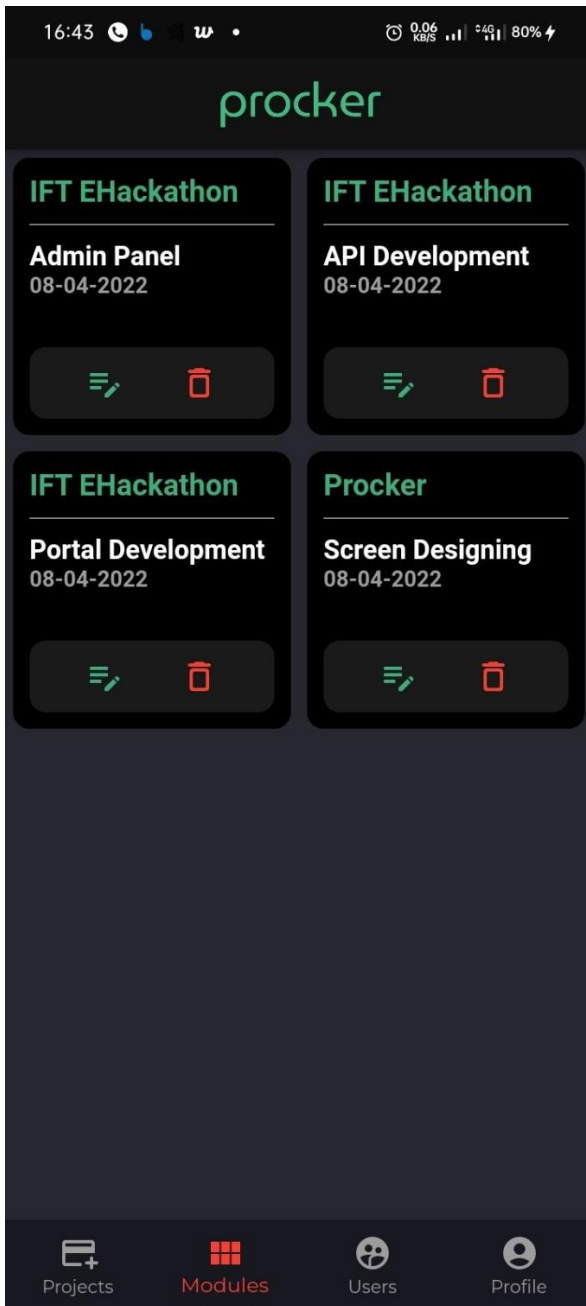


Figure 5.5.11 Modules Screen (admin)

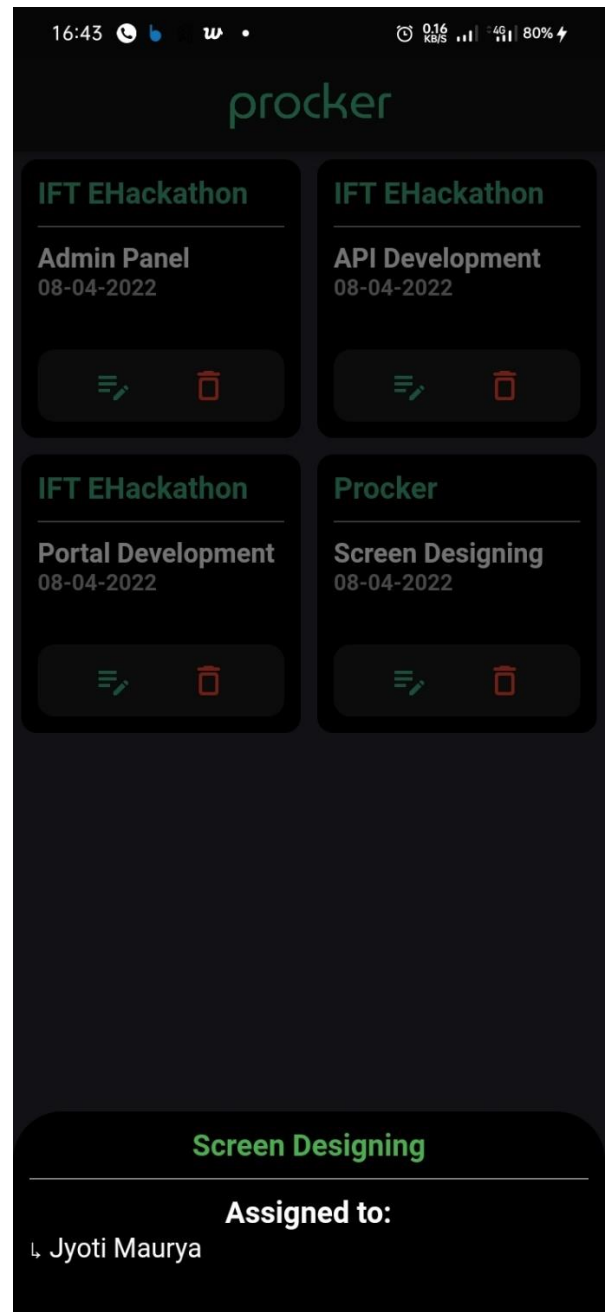


Figure 5.5.12 Assigned to users (admin)

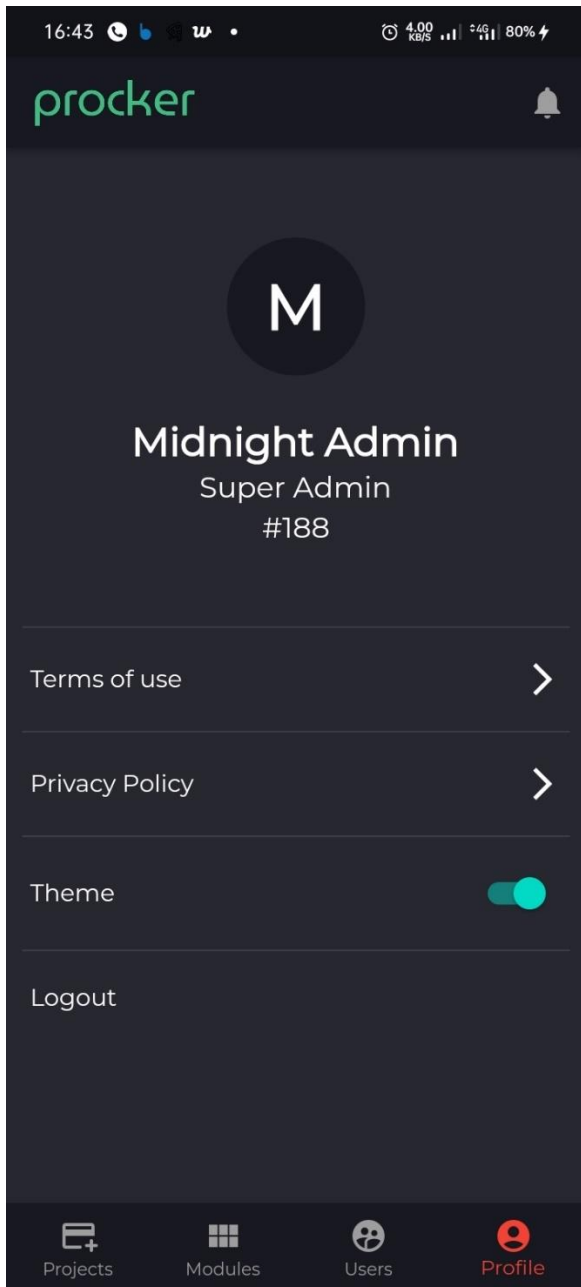


Figure 5.5.13 Profile Screen (admin)

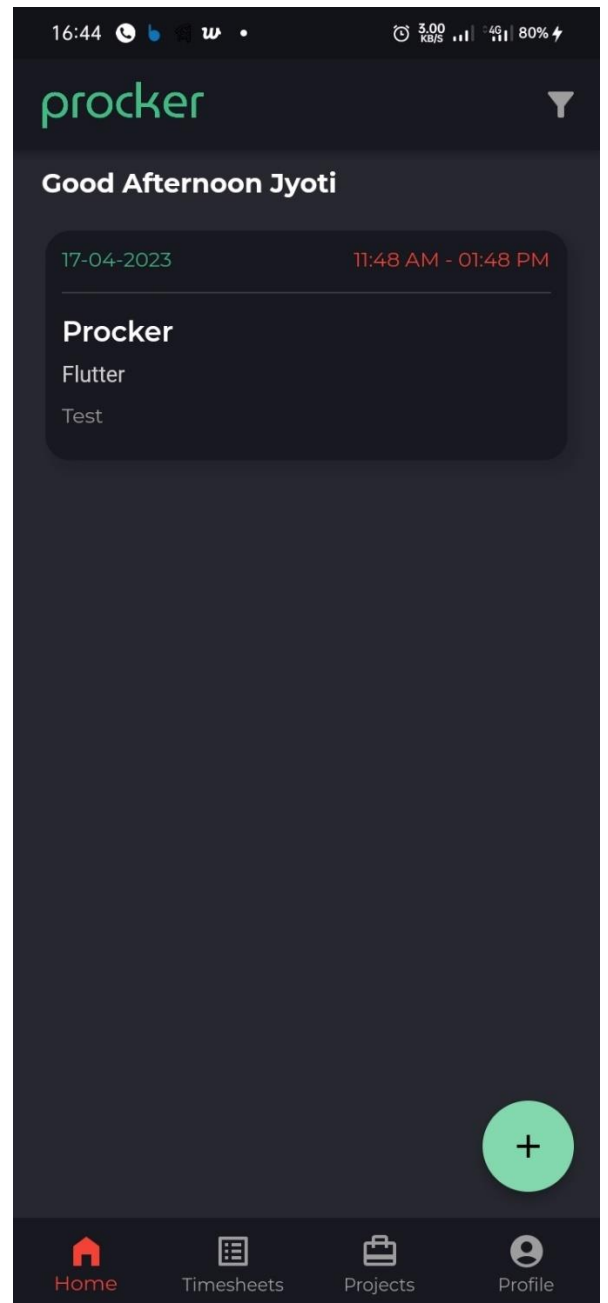


Figure 5.5.14 Home Screen

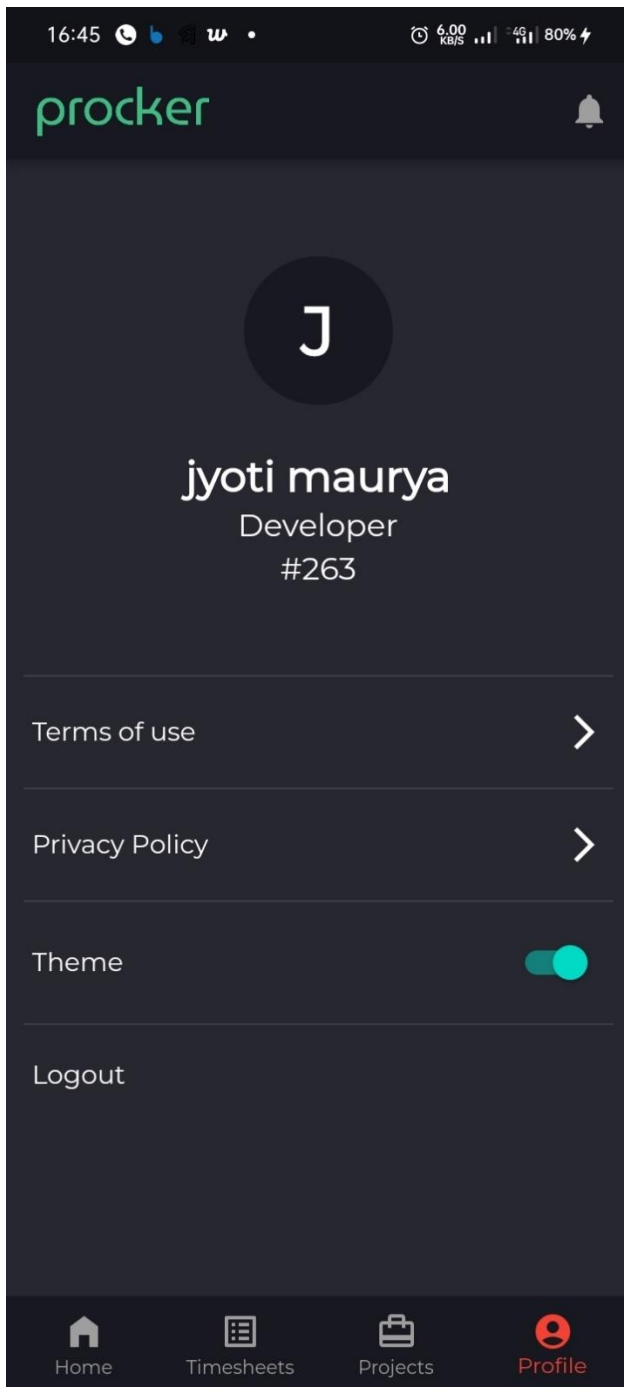


Figure 5.5.15 Profile Screen

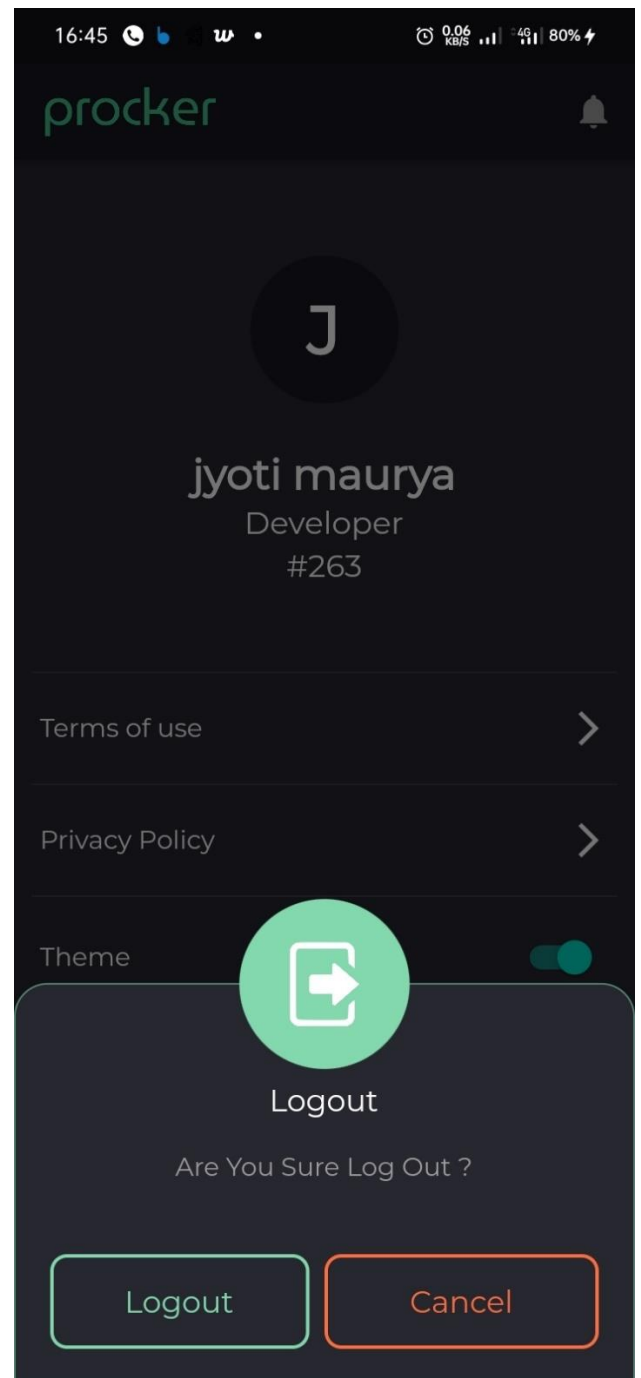


Figure 5.5.16 Logout Screen

CHAPTER-6

Testing

6.1 Testing Plan:

- A test plan is a document that sets out the scope, approach, and schedule of intended testing activities. The test plan may also list the resources the software tester needs to function effectively.
- The test plan usually includes the following information:
 - The overall objective of the testing effort.
 - A detailed outline of how testing will be conducted (the test approach).
 - The features, applications, or components to be tested.
 - Detailed scheduling and resource allocation plans for testers and developers throughout all stages of testing.
- **What are the objectives of a software test plan?**

The primary objective for a test plan is to produce documentation that describes how the tester will verify that the system works as intended. The document should describe what needs to be tested, how it will be tested, and who's responsible for doing so.

By writing up a test plan, all team members can work in unison and communicate their roles to one another. You should consider creating some SMART objectives for your test plan.



Figure 6.1: Testing Objective

■ **How to write a Test Plan?**

You already know that making a **Test Plan** is the most important task of Test Management Process

Following are the 14 essential things to include in your software test plan as part of the QA process.

➤ **Learn about the software-**

Before testing starts, it's important to learn everything you can about the software. Ask questions about how it was developed to learn about its intended purpose, how it works, and to garner information that might help you understand its functionality.

By understanding your software properly, you can create test cases that are relevant and useful for testing your product.

➤ **Define the scope of testing-**

There's no point in creating testing documents that are longer than the product itself. Before anything else, establish what exactly will be tested during the process, which modules or functions need to be covered in detail, and any other essential aspects you should know about.

➤ **Create Test Cases-**

One of the main tasks when developing a software testing document is creating test cases. A test case is a document that describes the steps taken to carry out your testing. It should include information such as:

What needs to be tested

- How it will be tested
- Who will do the testing
- Expected results

➤ **Develop a Test Strategy-**

The test strategy defines how you plan to implement testing. Your testers should all be working off the same game plan, so make sure every member of the team is aware of what they're supposed to be doing at any given time.

➤ **Define the Test Objective-**

Each test case should be linked to a test objective. The objective ensures every action is relevant and contributes toward making your software more valuable for customers. Test objectives can include things like:

- Testing known features
- Testing newly implemented features
- Performing exploratory tests
- Ensuring stability throughout the product lifecycle

➤ **Choose Testing tools-**

You'll need to make sure you have the right software testing solution to perform your testing activities. Some of these tools might be software-based, while others may require physical resources like test machines. It's important to choose appropriate tools for each specific job and not to rely on a one-size-fits-all solution.

➤ **Find bugs early-**

Leave time in your planning document for 'bug fixing' sessions. These allow you to identify problems with the software early on before they become too problematic or expensive to fix. This makes them easier and cheaper to tackle. Check out any app security measures, use every feature, and seek out what doesn't work well.

➤ **Define your Test Criteria-**

This should be part of the test case, but it's good to break it down separately. Test criteria are essentially your objectives broken down into smaller parts. They include specific information about how each objective will be met, which helps you track your testing progress.

Suspension criteria are criteria that need to be met before testing can stop. For example, you may want to suspend testing if a certain number of bugs have been found or if the software is unable to run due to performance issues.

Exit criteria are criteria that need to be met before testing can finish. For example, the test case should finish once each objective has been met and all bugs have been resolved.

➤ **Resource Planning-**

In your software testing document, include a resource plan that lists the number of people required for the testing process. This should detail what each person's role is and any training they'll require to fulfil it effectively.



Figure 6.2: Resource Planning

➤ **Plan your Test Environment-**

In your test plan, include information about the environment where testing will take place, such as:

- Test hardware required for product testing.
- Sizing requirements for software and servers.
- Platforms supported by the product.
- Other essential information related to the environment that might affect your testing process.

➤ **Plan test team logistics-**

Test management is one of the most important parts of implementing process. If you're not able to communicate with your testers effectively, their progress will suffer and your testing document won't be as useful as it could be.

➤ **Schedule & Estimation-**

In your test plan, include a schedule that allows you to outline specific testing milestones and deadlines. Milestones may include the initial release of the product, internal testing sessions, public beta tests, or any other key points in time where your team needs to focus their efforts on testing.

➤ **Test Deliverables-**

Your testing document should include a list of all the deliverables required for testing. These should be linked to the steps in your schedule so everyone knows exactly when they need to be ready for action.

➤ **Test Automation -**

If your software is particularly complex and requires a vast number of test cases, you may want to consider software test automation.

Automating the process means testers can accomplish more in less time, which boosts productivity and significantly reduces the overall cost of testing. You might even be able to utilize a mobile bot to speed up testing activities.

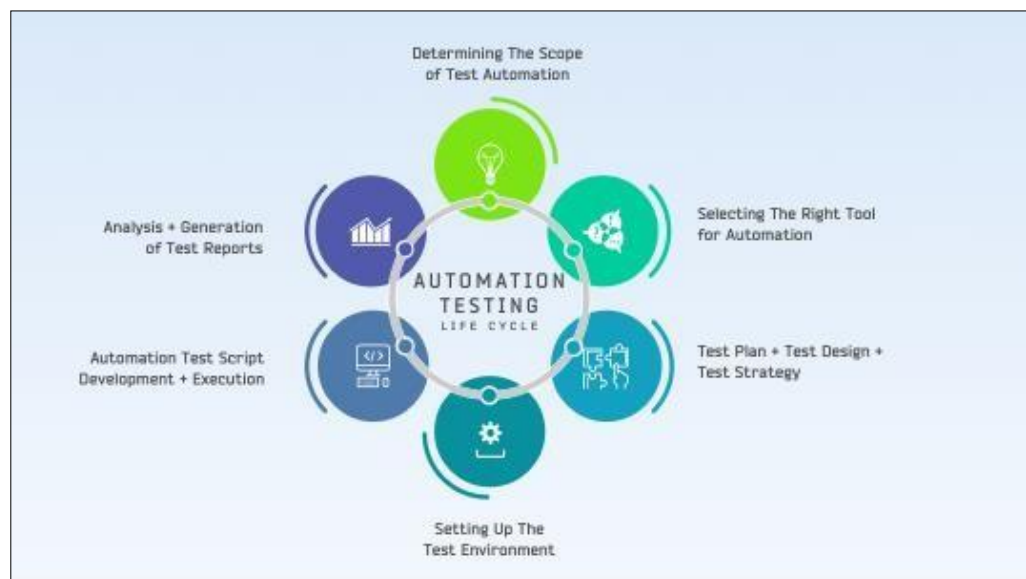


Figure 6.3: Testing Automation

6.2 Testing Strategy:

- Software Testing is a type of investigation to find out if there is any default or error present in the software so that the errors can be reduced or removed to increase the quality of the software and to check whether it fulfils the specifies requirements or not.
- The overall strategy for testing software includes:

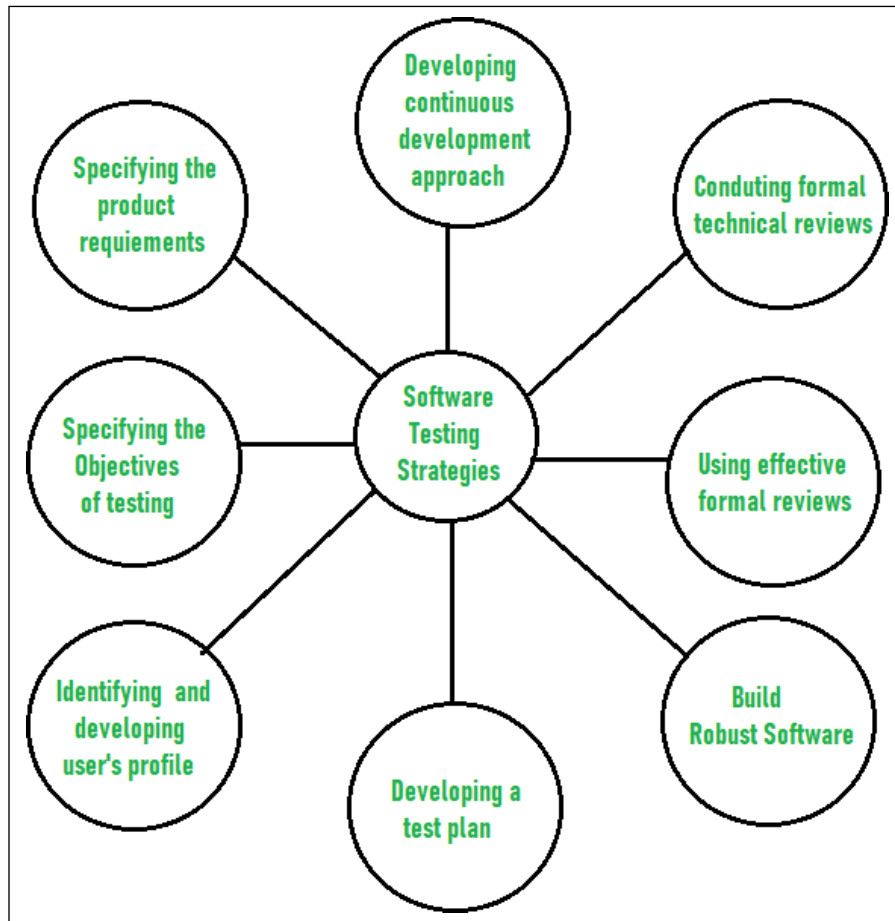


Figure 6.4: Testing Strategy

- **Before testing starts, it's necessary to identify and specify the requirements of the product in a quantifiable manner.**

Different characteristics quality of the software is there such as maintainability that means the ability to update and modify, the probability that means to find and estimate any risk, and usability that means how it can easily be used by the customers or end-users. All these characteristic qualities should be specified in a particular order to obtain clear test results without any error.

- **Specifying the objectives of testing in a clear and detailed manner.**

Several objectives of testing are there such as effectiveness that means how effectively the software can achieve the target, any failure that means inability to fulfil the requirements and perform functions, and the cost of defects or errors that mean the cost required to fix the error. All these objectives should be clearly mentioned in the test plan.

- **For the software, identifying the user's category and developing a profile for each user.**
Use cases describe the interactions and communication among different classes of users and the system to achieve the target. So as to identify the actual requirement of the users and then testing the actual use of the product.
- **Developing a test plan to give value and focus on rapid-cycle testing.**
Rapid Cycle Testing is a type of test that improves quality by identifying and measuring the any changes that need to be required for improving the process of software. Therefore, a test plan is an important and effective document that helps the tester to perform rapid cycle testing.
- **Robust software is developed that is designed to test itself.**
The software should be capable of detecting or identifying different classes of errors. Moreover, software design should allow automated and regression testing which tests the software to find out if there is any adverse or side effect on the features of software due to any change in code or program.
- **Before testing, using effective formal reviews as a filter.**
Formal technical reviews are the technique to identify the errors that are not discovered yet. The effective technical reviews conducted before testing reduces a significant amount of testing efforts and time duration required for testing software so that the overall development time of software is reduced.
- **Conduct formal technical reviews to evaluate the nature, quality or ability of the test strategy and test cases.**
The formal technical review helps in detecting any unfilled gap in the testing approach. Hence, it is necessary to evaluate the ability and quality of the test strategy and test cases by technical reviewers to improve the quality of software.
- **For the testing process, developing a approach for the continuous development.**
As a part of a statistical process control approach, a test strategy that is already measured should be used for software testing to measure and control the quality during the development of software.

6.3 Testing Methods:

- Software Testing Methodologies are the various strategies or approaches used to test an application to ensure it behaves and looks as expected. These encompass everything from front to back-end testing, including unit and system testing.
- **Functional vs. Non-functional Testing:**
 - The goal of utilizing numerous testing methodologies in your development process is to make sure your software can successfully operate in multiple environments and across different platforms.
 - These can typically be broken down between functional and non-functional testing.
 - Functional testing involves testing the application against the business requirements.
 - It incorporates all test types designed to guarantee each part of a piece of software behaves as expected by using use cases provided by the design team or business analyst.
 - These testing methods are usually conducted in order and include:
 - Unit testing
 - Integration testing
 - System testing
 - Acceptance testing
 - Non-functional testing methods incorporate all test types focused on the operational aspects of a piece of software. These include:
 - Performance testing
 - Security testing
 - Usability testing
 - Compatibility testing
 - The key to releasing high quality software that can be easily adopted by your end users is to build a robust testing framework that implements both functional and non-functional software testing methodologies.
 - **Unit Testing-**

Unit testing is the first level of testing and is often performed by the developers themselves. It is the process of ensuring individual components of a piece of software at the code level are functional and work as they were designed to. Developers in a test-driven environment will typically write and run the tests prior to the software or feature being passed over to the test team. Unit testing can be conducted manually, but automating the process will speed up delivery cycles and expand test coverage. Unit testing will also make debugging easier because finding issues earlier means they take less time to fix than if they were discovered later in the testing process. TestLeft is a tool that allows advanced testers and developers to shift left with the fastest test automation tool embedded in any IDE.
 - **Integration Testing-**

After each unit is thoroughly tested, it is integrated with other units to create modules or components that are designed to perform specific tasks or

activities. These are then tested as group through integration testing to ensure whole segments of an application behave as expected (i.e, the interactions between units are seamless). These tests are often framed by user scenarios, such as logging into an application or opening files. Integrated tests can be conducted by either developers or independent testers and are usually comprised of a combination of automated functional and manual tests.

➤ **System Testing-**

System testing is a black box testing method used to evaluate the completed and integrated system, as a whole, to ensure it meets specified requirements. The functionality of the software is tested from end-to-end and is typically conducted by a separate testing team than the development team before the product is pushed into production.

➤ **Acceptance Testing-**

Acceptance testing is the last phase of functional testing and is used to assess whether or not the final piece of software is ready for delivery. It involves ensuring that the product is in compliance with all of the original business criteria and that it meets the end user's needs. This requires the product be tested both internally and externally, meaning you'll need to get it into the hands of your end users for beta testing along with those of your QA team. Beta testing is key to getting real feedback from potential customers and can address any final usability concerns.

➤ **Performance Testing-**

Performance testing is a non-functional testing technique used to determine how an application will behave under various conditions. The goal is to test its responsiveness and stability in real user situations. Performance testing can be broken down into four types:

- **Load testing** is the process of putting increasing amounts of simulated demand on your software, application, or website to verify whether or not it can handle what it's designed to handle.
- **Stress testing** takes this a step further and is used to gauge how your software will respond at or beyond its peak load. The goal of stress testing is to overload the application on purpose until it breaks by applying both realistic and unrealistic load scenarios. With stress testing, you'll be able to find the failure point of your piece of software.
- **Endurance testing**, also known as soak testing, is used to analyze the behaviour of an application under a specific amount of simulated load over longer amounts of time. The goal is to understand how your system will behave under sustained use, making it a longer process than load or stress testing (which are designed to end after a few hours). A critical piece of endurance testing is that it helps uncover memory leaks.
- **Spike testing** is a type of load test used to determine how your software will respond to substantially larger bursts of concurrent user or system activity over varying amounts of time. Ideally, this will help you understand what will happen when the load is suddenly and drastically increased.

➤ **Security Testing-**

With the rise of cloud-based testing platforms and cyber-attacks, there is a growing concern and need for the security of data being used and stored in software. Security testing is a non-functional software testing technique used to determine if the information and data in a system is protected. The goal is to purposefully find loopholes and security risks in the system that could result in unauthorized access to or the loss of information by probing the application for weaknesses. There are multiple types of this testing method, each of which aimed at verifying six basic principles of security:

- Integrity
- Confidentiality
- Authentication
- Authorization
- Availability
- Non-repudiation

➤ **Usability Testing-**

Usability testing is a testing method that measures an application's ease-of-use from the end-user perspective and is often performed during the system or acceptance testing stages. The goal is to determine whether or not the visible design and aesthetics of an application meet the intended workflow for various processes, such as logging into an application. Usability testing is a great way for teams to review separate functions, or the system as a whole, is intuitive to use.

➤ **Compatibility Testing-**

Compatibility testing is used to gauge how an application or piece of software will work in different environments. It is used to check that your product is compatible with multiple operating systems, platforms, browsers, or resolution configurations. The goal is to ensure that your software's functionality is consistently supported across any environment you expect your end users to be using.

6.4 Test Cases:

Test Case of User Registration:

Test case ID	Test Scenario	Test Steps	Expected Results	Actual Results	Pass/Fail
T101	Register User to the system.	i. Open app □ User's page of admin application ii. Click on add User. iii. Enter Details iv. Submit	User details must be store in the database and should have the message 'User Added Successfully'.	As expected,	Pass

Table 6.4.1 User Registration

Test case of User Login

Test case ID	Test Scenario	Test Steps	Expected Results	Actual Results	Pass/Fail
T201	Check User Login with valid Data	i. Open app ii. Enter valid user mail and password. iii. Click on Login	User should login into application.	As expected	Pass
T202	Check User Login with invalid Data	i. Open app ii. Enter valid user mail and password. iii. Click on login.	User should not Login into application on empty or invalid details..	As expected	Pass

Table 6.4.2 User Login

Test case of Add Timesheet

Test case ID	Test Scenario	Test Steps	Expected Results	Actual Results	Pass/Fail
T301	Add timesheet	iv. Login with email id v. Add task sheet vi. Click on submit	New Screen should appear to add task sheet date with project name and module name.	As expected	Pass

Table 6.4.3 Add Timesheet

Test case of Add Project

Test case ID	Test Scenario	Test Steps	Expected Results	Actual Results	Pass/Fail
T401	Add Project	i. Login with valid id. ii. Add Project iii. Click on submit.	New Screen should appear to add project.	As expected	Pass

Table 6.4.4 Add Project

Test case of Add Module

Test case ID	Test Scenario	Test Steps	Expected Results	Actual Results	Pass/Fail
T401	Add Module	i. Login with valid id. ii. Add Module iii. Click on submit.	New Screen should appear to add module.	As expected	Pass

Table 6.4.5 Cart Details

References

- <https://studentprojectguide.com/project-report/software-testing/test-cases-for-employee-payroll/>
- <https://www.guru99.com>
- <https://www.tutorialspoint.com>
- <https://www.softwaretestinghelp.com/>
- <https://www.inflectra.com/>
- www.geeksforgeeks.org
- www.w3schools.com
- www.projectmanagement-training.net
- www.javatpoint.com
- <https://codebots.com/>
- www.freeprojectz.com
- www.scfibd.com
- www.academia.edu
- <https://www.w3schools.com/mysql>
- <https://en.ubie.app/>
- <http://softwaretestingfundamentals.com/software-testing-methods/>
- <https://youtu.be/IJgMHowYAGo>
- <https://www.binarytides.com/create-foreign-key-phpmyadmin/>
- <https://gist.github.com/sheharyarn/20f171e900eff32bf38fd8be1d30911d>
- <https://www.youtube.com/watch?v=4e8be8xseqE>
- <https://geteasyqa.com/qa/best-test-case-templates-examples/>