LAB REPORT

Submitted by

Harshvardhan Tewani (RA2111027010032)

Keshav Kishan (RA2111027010055)

Akash Raheja(RA2111027010040)

Sarvesh Sreejesh(RA2111027010047)

Under the Guidance of

Dr Arthi K

Professor, DSBS

In partial satisfaction of the requirements for the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE ENGINEERING

with specialization in Big Data Analytics



SCHOOL OF COMPUTING

COLLEGE OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR - 603203

MAY 2023

|  |
| --- |
| COLLEGE OF ENGINEERING & TECHNOLOGY  SRM INSTITUTE OF SCIENCE & TECHNOLOGY  S.R.M. NAGAR, KATTANKULATHUR 603 203  Chengalpattu District  BONAFIDE CERTIFICATE  Register No. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Certified to be the bonafide work done by \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_of II Year/IV Sem B.Tech Degree Course in the Practical Software Software Engineering and Project Management 18CSC206J in SRM INSTITUTE OF SCIENCE AND TECHNOLOGY, Kattankulathur during the academic year 2022 2023.  LAB INCHARGE Head of the Department  Name of Faculty:  Designation:  Department of Computing Technologies  SRMIST KTR. |

TABLE OF CONTENTS

CHAPTER TITLE PAGE NO NO

ABSTRACT

1. PROBLEM STATEMENT
2. STAKEHOLDERS & PROCESS MODELS
3. IDENTIFYING REQUIREMENTS
4. PROJECT PLAN & EFFORT
5. WORK BREAKDOWN STRUCTURE & RISK ANALYSIS
6. SYSTEM ARCHITECTURE, USE CASE & CLASS DIAGRAM
7. ENTITY RELATIONSHIP DIAGRAM
8. DATA FLOW DIAGRAM
9. SEQUENCE & COLLABORATION DIAGRAM
10. DEVELOPMENT OF TESTING FRAMEWORK/USER INTERFACE
11. TEST CASES & REPORTING
12. ARCHITECTURE/DESIGN/FRAMEWORK/IMPLE -MENTATION

CONCLUSION

REFERENCES

APPENDIX (CODE)

**ABSTRACT**

A Personal Task Management System is a system for individuals to use in their personal and professional accounts. This system helps in ensuring that the work/Task is completed on time. It is not easily hackable, and helps user in ensuring better results

Project management software is designed to help businesses and individuals track projects, tasks and schedules. It's a great way to stay organized and ensure that your team remains on task.

# Personal Task Management System is a set of software designed to help project teams to plan a project, track & manage the projects to achieve the defined project goals within the time. It also helps team members to collaborate effectively and accelerate the projects to meet the specified constraints

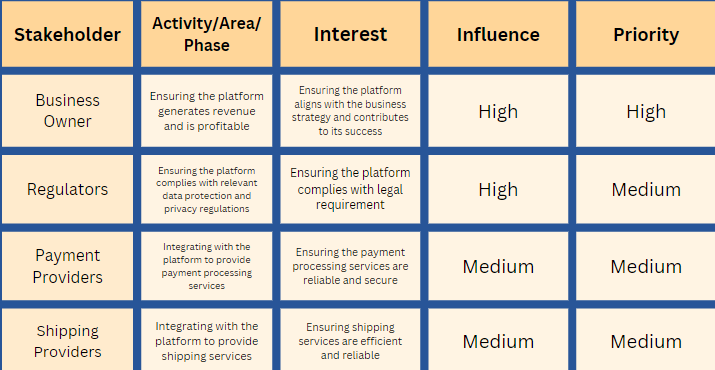
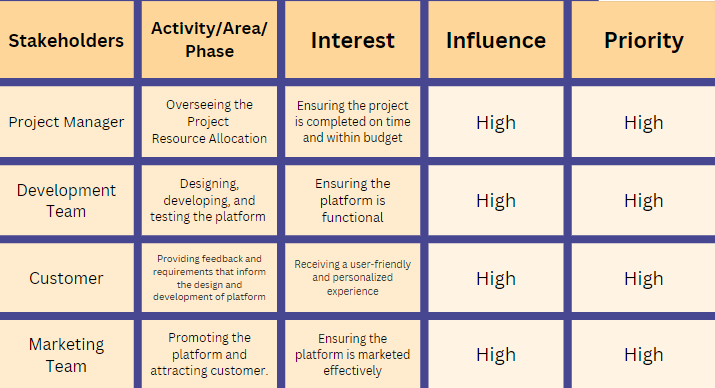
# PROBLEM STATEMENT

# [Task management](https://taimer.com/?page_id=67252#projectmgmt) is the process of managing a task through its life cycle: from planning, testing and tracking, to reporting on the outcome. It involves managing all aspects of a task, from its status and priority to the time spent, people involved, and finally, financial resources needed.

# Task management methods, systems, and techniques give you and your team a detailed and real-time view of all the moving parts of a project. It helps teams to stay productive and on schedule.

# STAKEHOLDERS & PROCESS MODELS

OUR STAKEHOLDERS: -



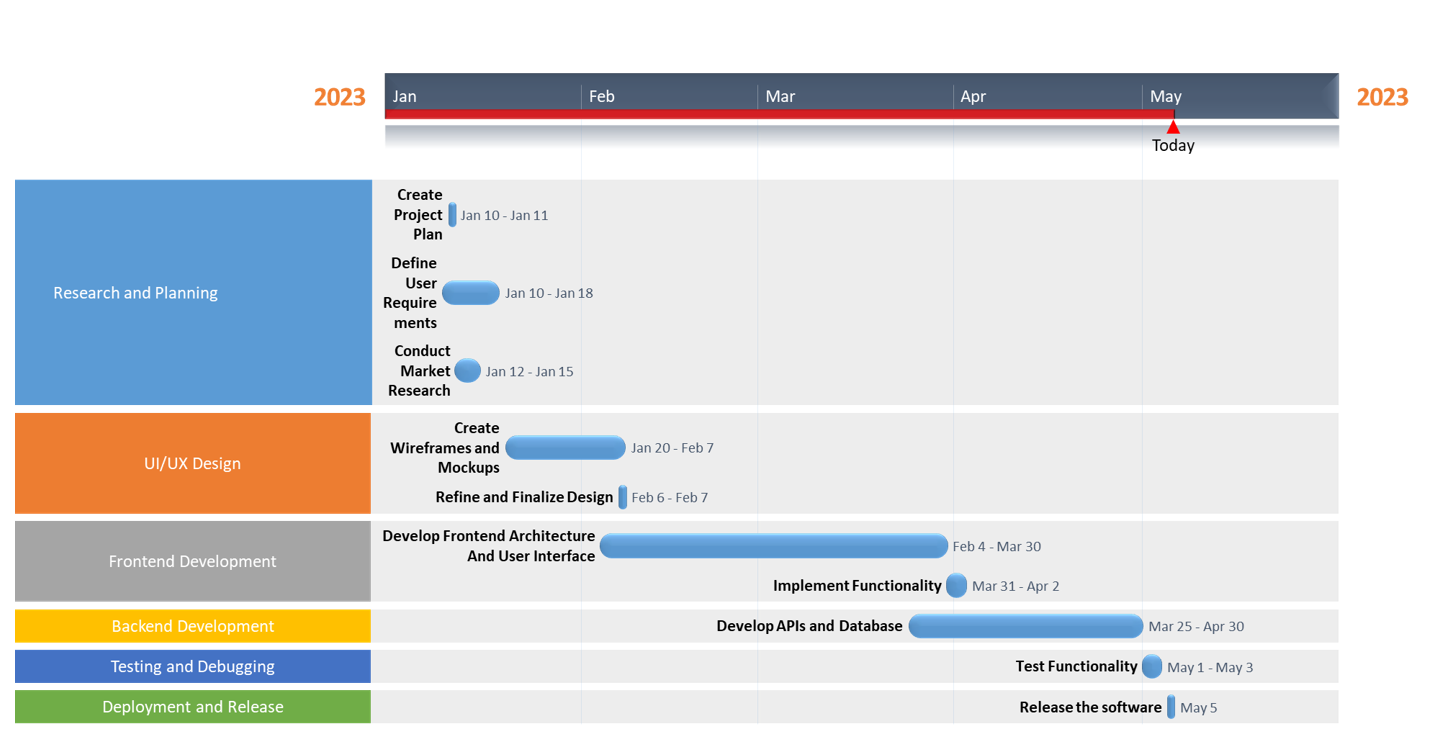
**OUR PROCESS MODEL: -**

One of the most important task management skills is risk management: no project ever goes exactly like planned. Situations change. Someone might get sick or have an unfortunate accident. A person might quit. In every case, you can use your task management system to take stock of the situation, adjust, and get back on track.  You should also:

* Have a full view of the project.
* Be proactive about anticipating situations and prioritizing tasks. The importance of prioritization can’t be overstated. However, be proactive about looking for potential priority shifts. The better you can prioritize, the more productive your team will be.
* Always be able to connect tasks to project goals. Connecting your tasks to project goals is a checklist to make sure you’re still going the right way. Make sure tasks align with project goals as much as possible.

Remove tasks that aren’t being actively worked on from immediate attention. Task management is as much about what’s not there as what is. If a task is no longer relevant due to a change in priority, just get rid of it. Too many tasks add an unnecessary cognitive load to your team.

**TIMELINE CHART**

****

**SYSTEM REQUIREMENTS:**

The system requirements specify the hardware and software components that are required for

the system to function properly. These include:

**- Server:** A server is required to host the application and ensure that it is accessible to users.

- **Hosting site:** The application will be hosted on django database or any online sevices.

**- Quad-core CPU:** The system requires an Intel Xeon quad-core CPU for efficient processing of data.

- **Platform:** The generator should be compatible with the platform(s) on which it will be used, such as Windows, Mac, Linux, or web browsers.

- **Performance:** The generator should be able to generate tasks quickly and efficiently, even for large numbers of users or high volumes of add task requests.

**FUNCTIONAL REQUIREMENTS**

**Customization:** The system should allow users to customize the length and complexity of the tasks according to their needs.

**User-friendly interface:** The system should have a user-friendly interface that is easy to use and understand, even for non-technical users.

**Unique Layout:** The system should ensure that each task is unique and not similar to any previously generated tasks.

**Security:** The system should be designed with security in mind and should not store or transmit the login information in plain text.

**Accessibility:** The system should be accessible to users across different platforms, including desktop, mobile, and web-based applications.

**Speed:** The system should generate tasks quickly and efficiently, without any noticeable delay.

**Non-Functional Requirements:**

**- Scalability:** The system should be able to handle an increase in the number of users.

**- Availability:** The system should be available and accessible to users 24/7, with minimum downtime for maintenance and upgrades.

**- Reliability:** The system should be reliable and consistent, with minimal errors and failures.

**- Maintainability:** The system should be easy to maintain and upgrade, with clear documentation and well-structured code.

**- Performance:** The system should provide fast and efficient performance, with quick response times and minimal latency.

**- Interoperability:** The system should be able to integrate with other systems and applications, allowing for seamless exchange of data and information.

**- Compliance:** The system should comply with relevant laws, regulations, and standards, such as privacy and data protection laws.

**PROJECT PLAN & EFFORT**

**Project Title:** Personal Task Management System

**Project Description:**

The project aims to develop a Personal Task Management System that can add and delete tasks for online accounts. The system will be user-friendly, efficient, and capable of creating tasks with various characters, including symbols, letters, and numbers. It will ensure that the tasks generated are given different priorities.

**Project Plan:**

**1.Requirements Gathering:**

Identify the requirements for the system by researching the best practices for task management.

Identify the necessary features, such as add tasks, delete tasks, completed task and character types.

**2. Design:**

Develop a design for the system, including the user interface.

Ensure that the design is user-friendly and efficient.

**3. Development:**

Develop the system using the selected programming language(s) and frameworks.

Test the system for functionality, usability, and security.

**4. Deployment:**

Deploy the system on a secure server and test its functionality in a real-world environment.

Ensure that the system meets the security standards and does not pose any security risks.

**5. Maintenance:**

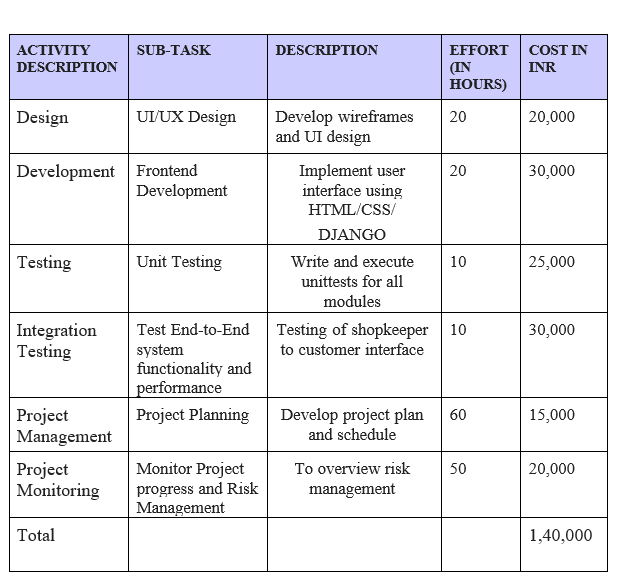
Provide ongoing maintenance and support for the system.

Address any issues and bugs that may arise and implement updates to improve its performance and security.

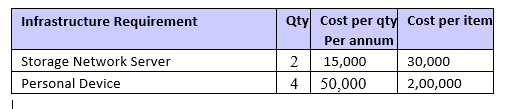
**PROJECT MANAGEMENT PLAN**

****

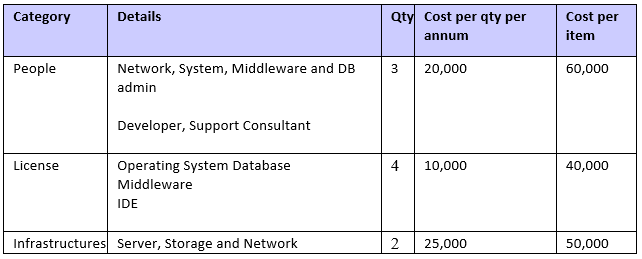
**COST AND EFFORT ESTIMATE:**

****

**INFRASTRUCTURE / RESOURCE COST: -**

****

**MAINTENANCE & SUPPORT COST: -**

****

**WORK BREAKDOWN STRUCTURE & RISK ANALYSIS:-**

Here is an example of a work breakdown structure (WBS) for a Personal task management system

1. Planning

* Define requirements
* Identify stakeholders
* Develop project plan

2. Design

* Define the personal task management algorithm
* Design user interface
* Determine data structures and storage requirements

3. Development

* Create personal task management algorithm
* Implement user interface
* Develop back-end data storage and retrieval mechanisms

4. Testing

* Develop test cases
* Perform unit testing
* Conduct integration testing

5. Deployment

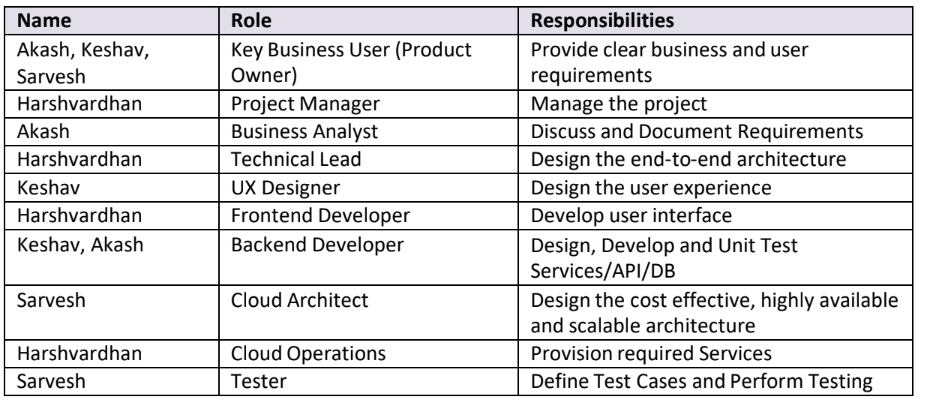
* Deploy the application to production environment
* Conduct user acceptance testing
* Train users and administrators

6. Maintenance

* Monitor system performance
* Address bug reports and user feedback

Perform regular system updates and maintenance

**TEAM MEMBER IDENTIFICATION: -**

****

**RESPONSIBILITY ASSIGNMENT MATRIX:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| RACI MATRIX |  | TEAM MEMBERS | |  |
| ACTIVITY | Akash Raheja  (BA) | Harshvardhan Tewani  (DEVELOPER) | Keshav Kishan  (PROJECT  MANAGER) | Sarvesh Sreejesh  (KEY BUSINESS  USER) |
| Requirement Gathering | R | A | C | I |
| Functional Design | R | A | C | I |
| Technical Design | R | A | C | I |
| Coding | I | R | A | - |
| Testing | A | R | C | I |
| Deployment | I | R | C | - |
| Maintenance | A | R | C | I |

Risk assessment is a crucial component of project management, as it helps to identify and analyze potential risks that may impact the project's success. By conducting a risk assessment, the project team can proactively identify potential issues and develop strategies to mitigate or eliminate them.

**SWOT ANALYSIS: -**

SWOT analysis is a strategic planning tool used to evaluate the Strengths, Weaknesses, Opportunities, and Threats of a project or organization. It involves analyzing the internal and external factors that may impact the project's success and identifying ways to capitalize on strengths, address weaknesses, exploit opportunities, and mitigate threats.



**RMMM PLAN: -**

|  |  |  |
| --- | --- | --- |
| STRATEGY | DESCRIPTION | AVOIDANCE TECHNIQUE |
| Avoidance | Reducing the likelihood of the occurrence of risk | Bringing in a consulting expert with experience in the field |
| Mitigation | Reducing the impact of a risk if it occurs | Creation of exception processing to deal with unexpected situations  Preparing an emergency Technical  Maintenance Team |
| Transferring | Moving the impact of a risk(if it occurs) to an external party | Set up a low-cost-per-hour Help Desk to provide support |
| Accepting | Deciding to accept the consequences of an occurring risk and documentation of it for  future use | Set up a Disaster Analysis and Review Team to prevent similar problems from occurring in the future |

# **SYSTEM ARCHITECTURE, USE CASE & CLASS DIAGRAM**

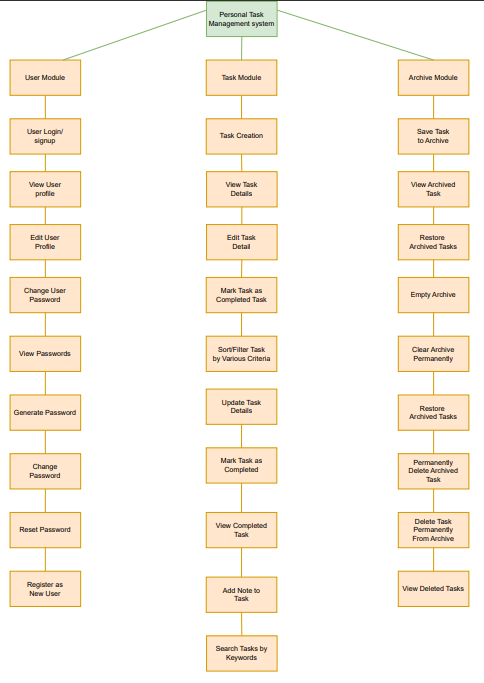
SYSTEM ARCHITECTURE: -

The system architecture of a personal task management system typically consists of several components working together. Here is an example architecture:

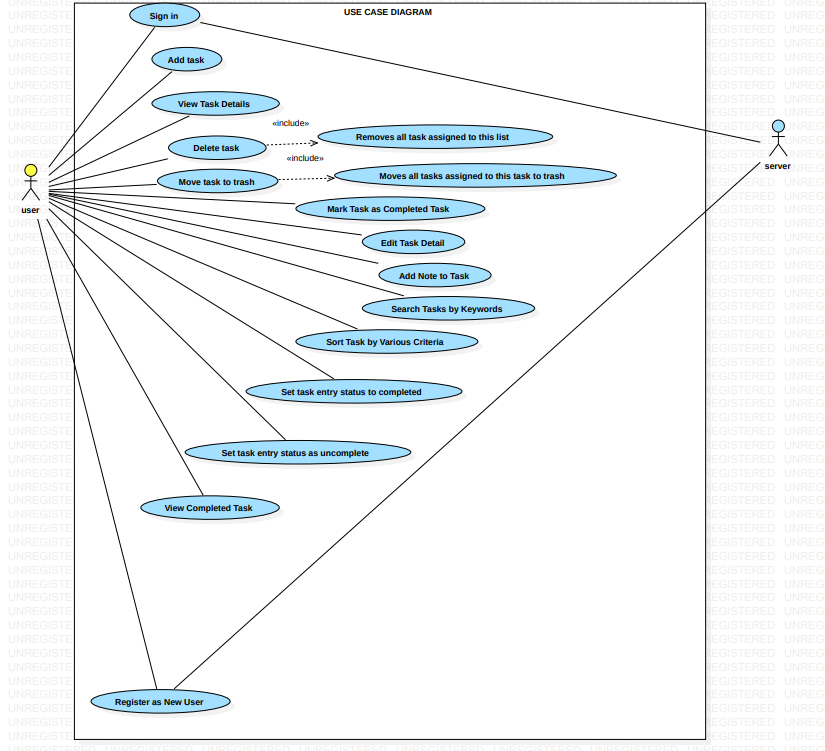
1. User Interface: The user interface component handles user interaction with the system. It presents the user with options to select length, complexity requirements, and any other relevant settings.
2. Personal Task Management System: This is the core component of the system that generates tasks based on the user's selected criteria.
3. Data Storage: The system may store user preferences and any other relevant data. This component could be a database or file storage.
4. Encryption: The system should encrypt the generated passwords before storing them in the database or displaying them to the user.
5. Security: The system should incorporate various security features, such asaccess controls, authentication mechanisms, and intrusion detection systems to protect against cyber attacks and data breaches.
6. API: The system may provide an API that allows other applications to accessits task management system capabilities.
7. Integration: The system may integrate with other applications, such as project creation models to provide a seamless user experience.

Overall, the architecture of a personal task management should prioritize security, performance, and scalability. The components of the system should be loosely coupled and modular to allow for easy maintenance and updates.

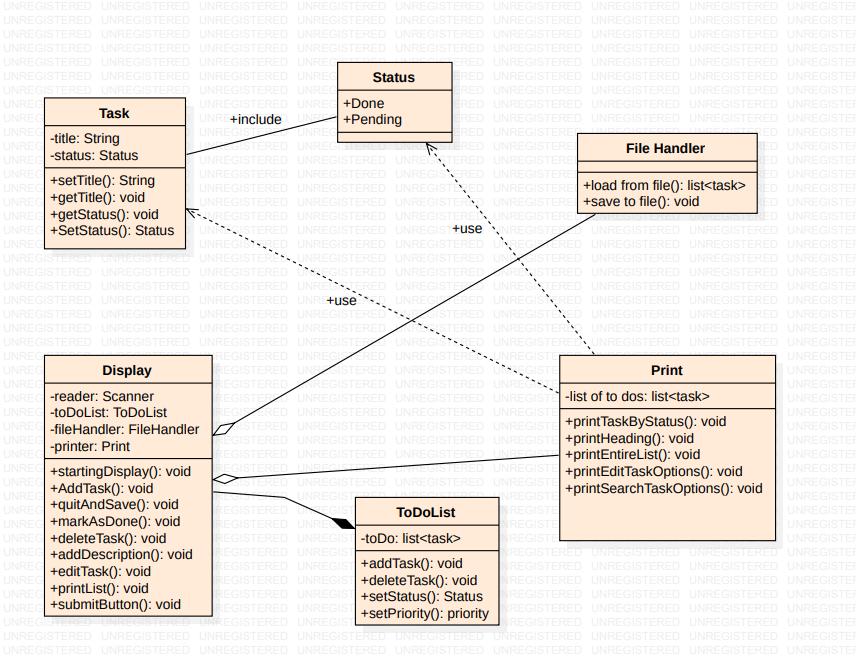
**WORK FLOW DIAGRAM: -**

****

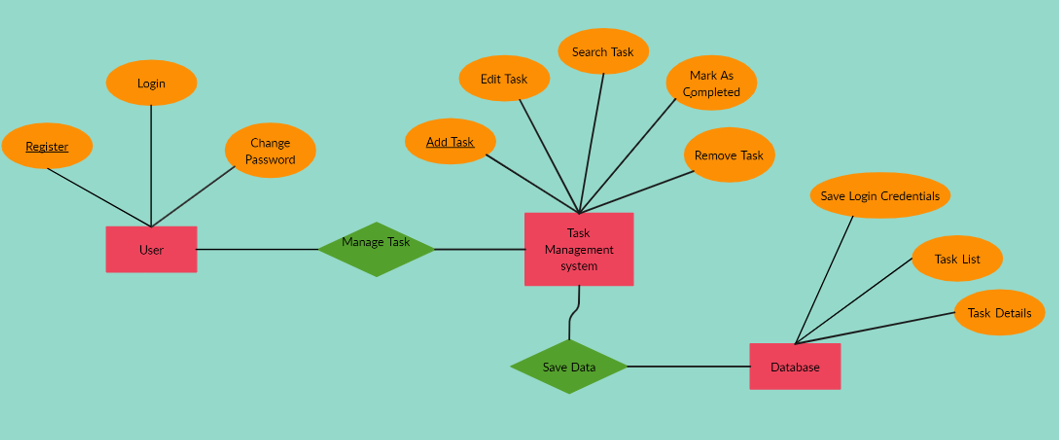
**USE CASE DIAGRAM: -**

****

**CLASS DIAGRAM: -**

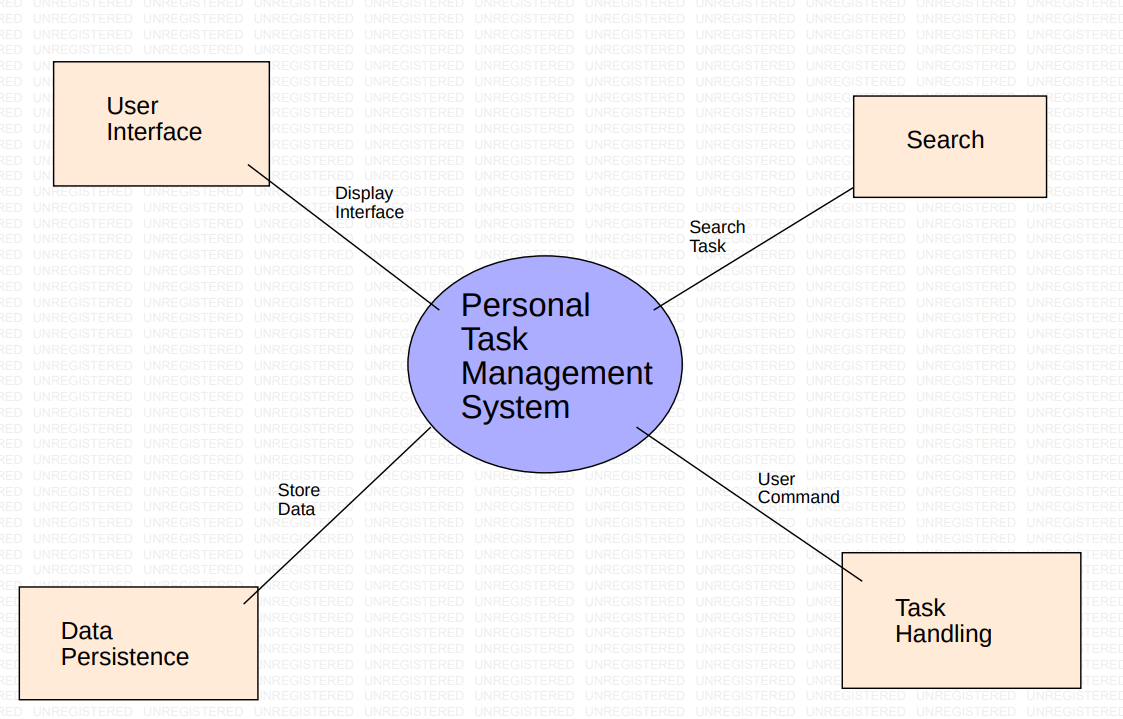
****

**ENTITY RELATIONSHIP DIAGRAM**

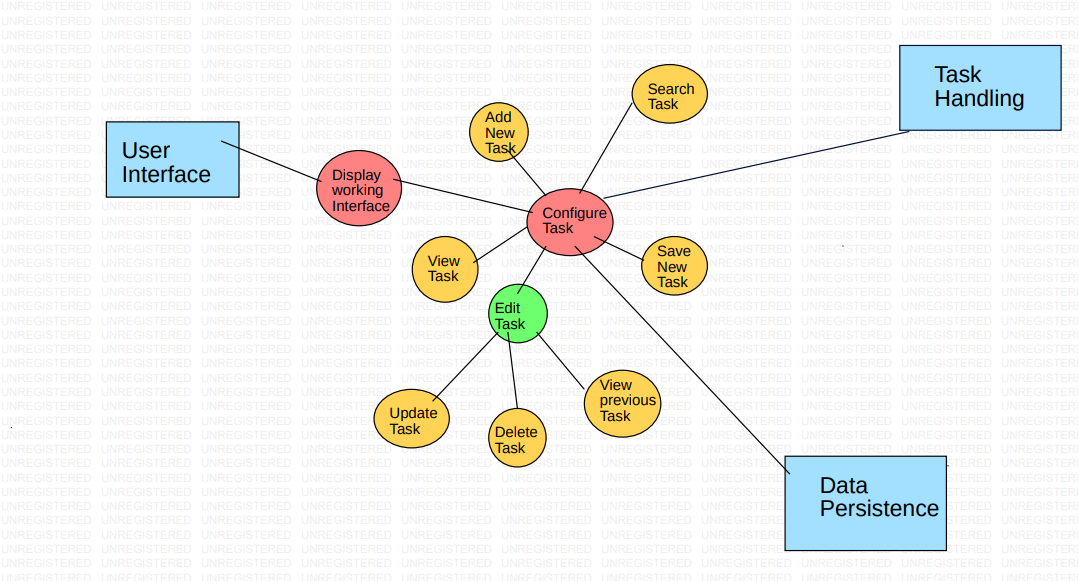
****

**DATA FLOW DIAGRAM**

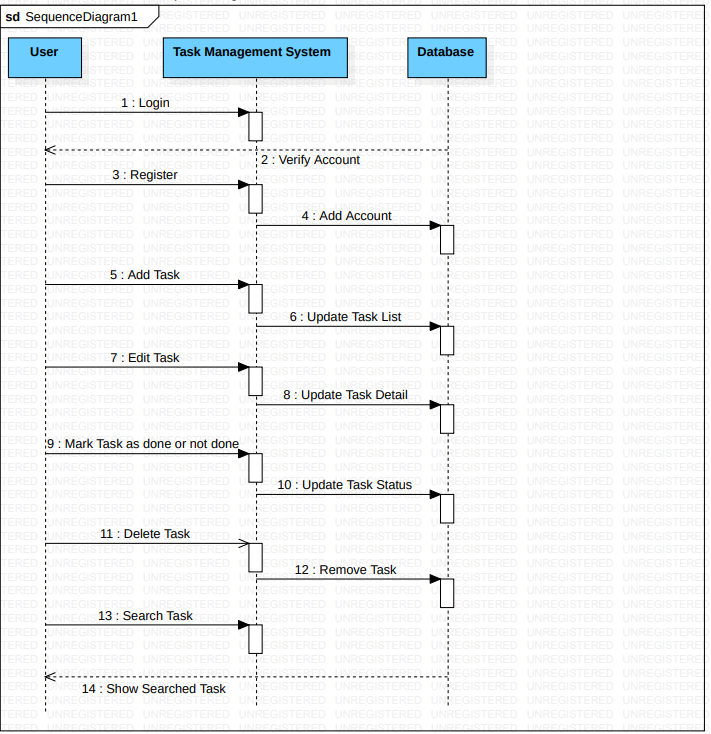
**DFD Level 0**

****

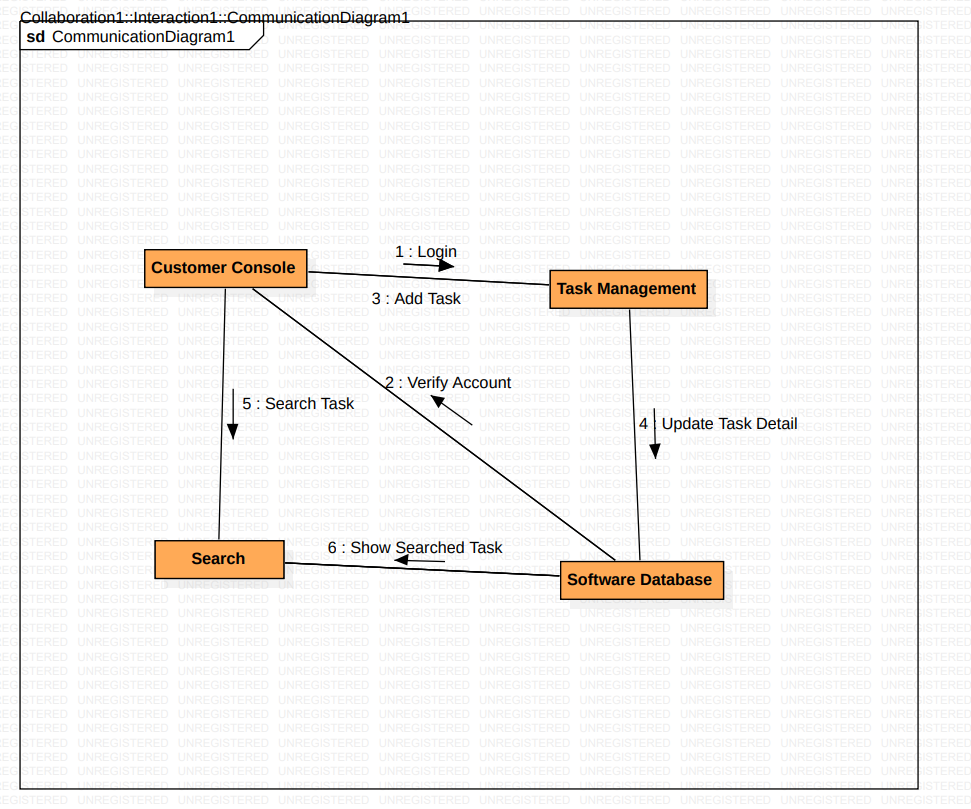
**DFD Level 1**

****

**SEQUENCE DIAGRAM**

****

**COLLABORATION DIAGRAM:**

****

**Scope of Testing**

**Functional:**

All functional test cases are covered via automation testing Unit testing, integration testing,

and functional testing are all covered. As a result, it is the best solution for functional and

critical route test cases, which are extremely important in a project. In addition, testing all test

cases take less time.

**Non-Functional:**

All non-functional modules testing are covered.

**Types of Testing, Methodology, Tools**

**Black Box Testing:** The Black Box Test is a test that only considers the external behavior of the

system; the internal workings of the software are not considered. A tester provides

input and observes the output generated by the system under test. Black box testing can be

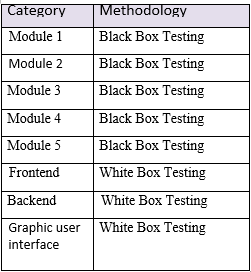
applied to three main types of tests: functional, non-functional, and regression testing.

**White Box Testing:** The White Box Test is a method used to test a software

taking into consideration its internal functioning. It is carried out by testers.

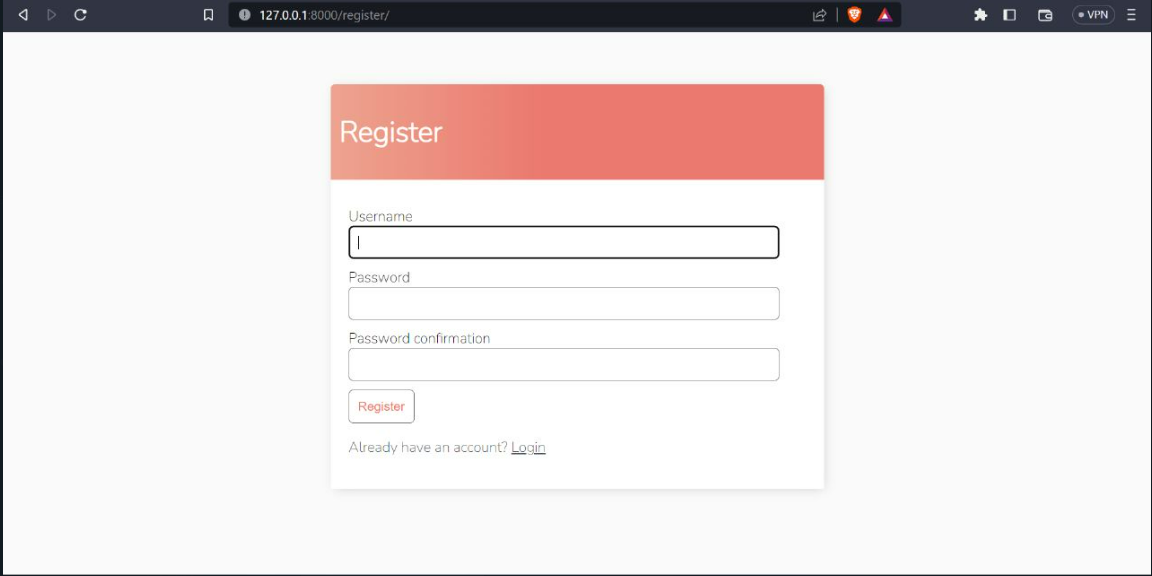
Test cases for white box testing are derived from the design phase of the

software development lifecycle.

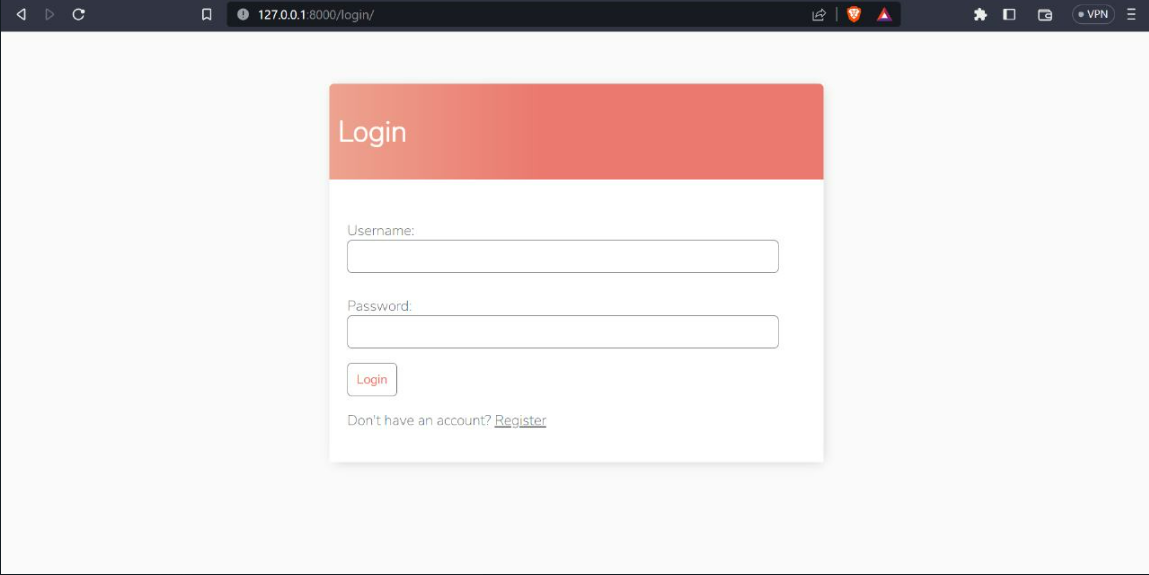


**Modules include both functional and non-functional**

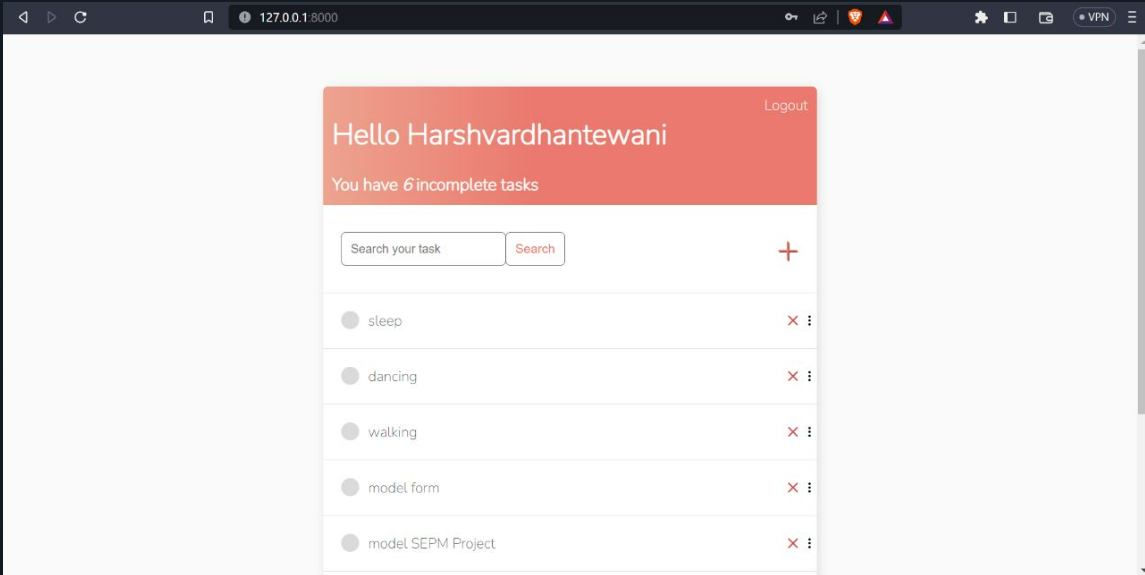
**Module 1:** consists of sign-up page test cases



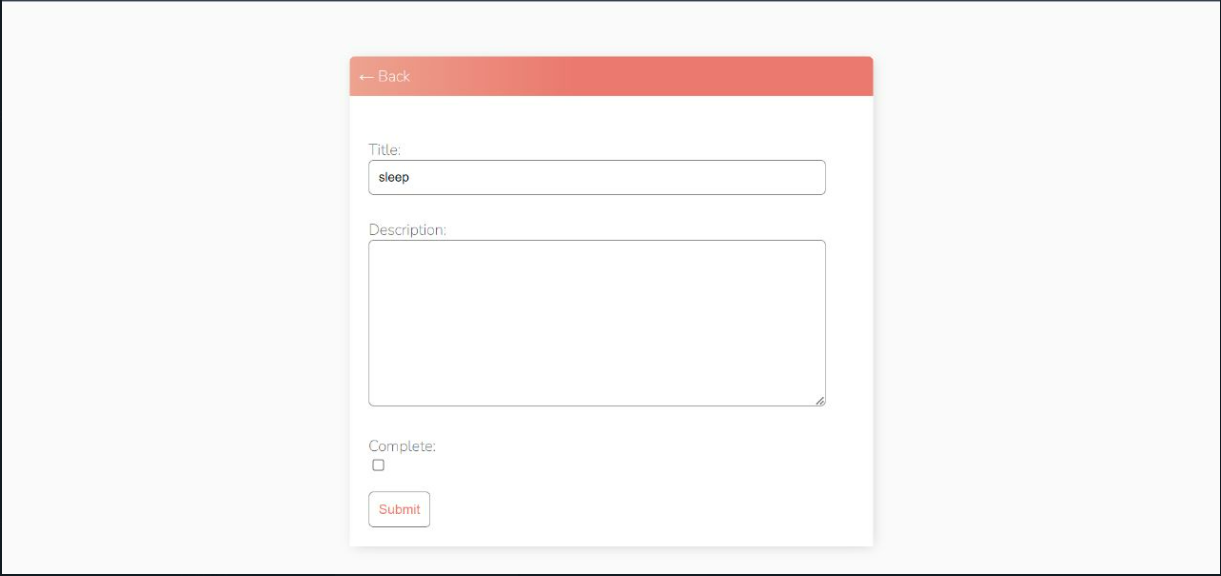
**Module 2:** consist of Login page test cases



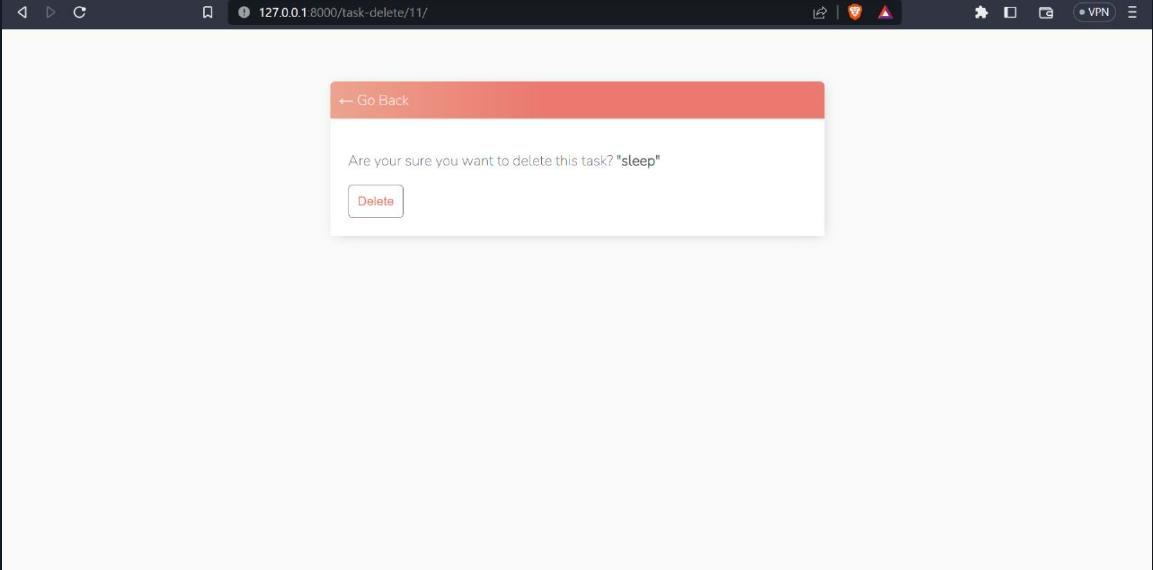
**Module 3:** consists of home page test cases



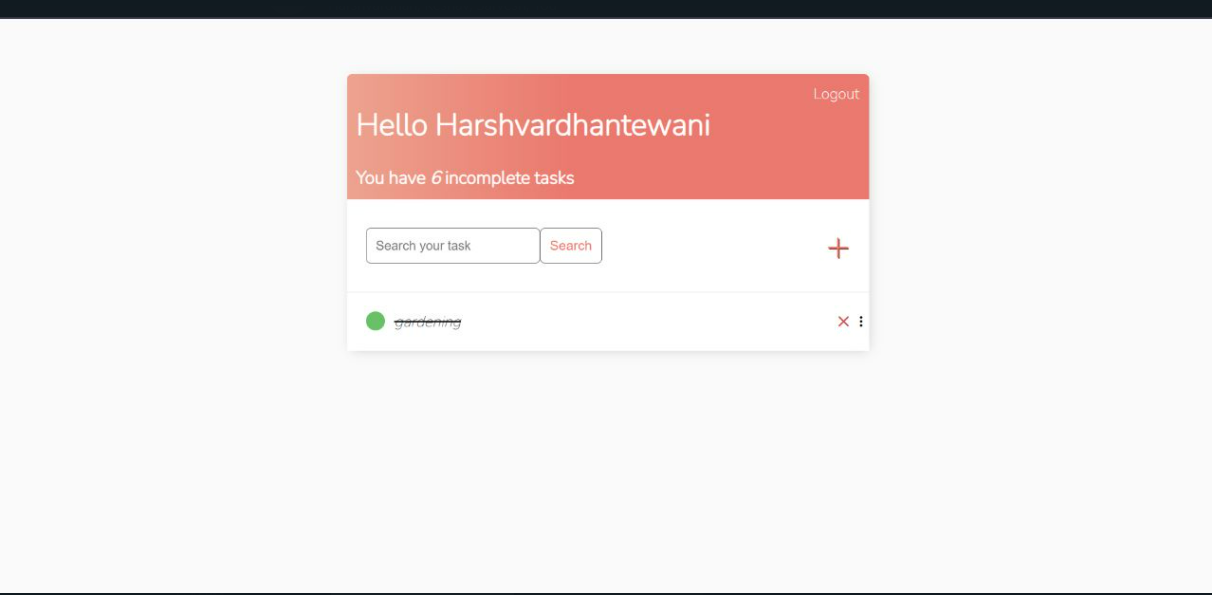
**Module 4:** consists of New Task Creation test cases



**Module 5:** consists of Deleting task



**Module 6:** consists of Updated List

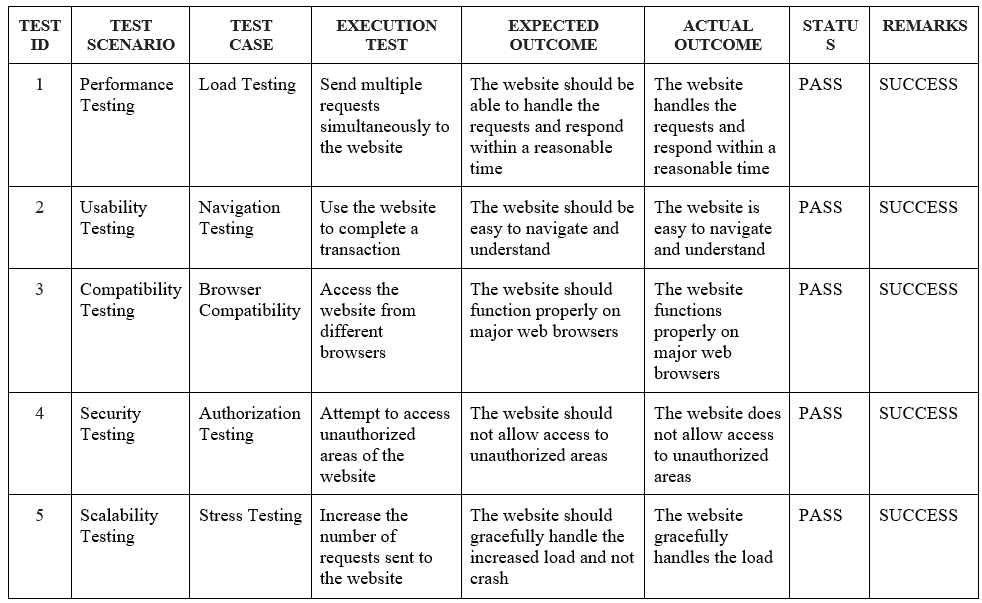


**Test Case**

**Functional Test Cases**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test**  **ID**  **(#)** | | **Test**  **Scenario** | **Test Case** | **Status** | **Remarks** |
| 1 | Check if username has alphabets | Accept only alphabets | pass | success |  |
| 2 | Check if the username has numerals | Accept max 4 numbers | pass | success |  |
| 3 | Check if the username has space | Do not accept space in between | pass | success |  |
| 4 | Check if username is in lowercase | Accept usernames with lowercase | pass | success |  |
| 5 | Check if password has alphabets | Only accept if there are at least 8 alphabets | pass | success |  |
| 6 | Check if the password has numerals | At least one numeral is necessary | pass | success |  |
| 7 | Check if password has space in it | Do not accept space in between | pass | success |  |
| 8 | Check if there is a login button | Check the link/button | pass | success |  |
| 9 | Check if there a register button | Check the link/button | pass | success |  |
| 10 | Verify that a new task can be added to the list. | Check the Addition functionality | pass | success |  |
| 11 | Verify that the task has a name and a description | At least one word name and description | pass | success |  |
| 12 | Verify that the added task is displayed on the list. | Check the list after new task added | pass | success |  |
| 13 | Verify that a task can be edited and updated with new information, such as name and description. | Check the editing functionality | pass | success |  |
| 14 | Verify that the edited task is displayed correctly on the list. | Check the updated task list | pass | success |  |
| 15 | Verify that a task can be deleted from the list. | Deletion functionality | pass | success |  |
| 17 | Verify that the deleted task is no longer displayed on the list. | Checks the updated task list | pass | success |  |
| 18 | Verify that a task can be marked as completed. | Checks the status | pass | success |  |
| 19 | Verify that the completed task is moved to the end of the list. | Checks the updated list | pass | success |  |
| 20 | Verify that the user can sort tasks based on priority. | Checks the prioritizing functionality | pass | success |  |
| 21 | Verify that the user can search for tasks based on keywords or other criteria. | Checks the search functionality | pass | success |  |
| 22 | Verify that the user can archive completed tasks. | Checks the archive functionality | pass | success |  |
| 23 | Verify that the archived tasks are moved to a separate list or section. | Checks the updated storage sever. | pass | success |  |

**Non-Functional Test Cases**

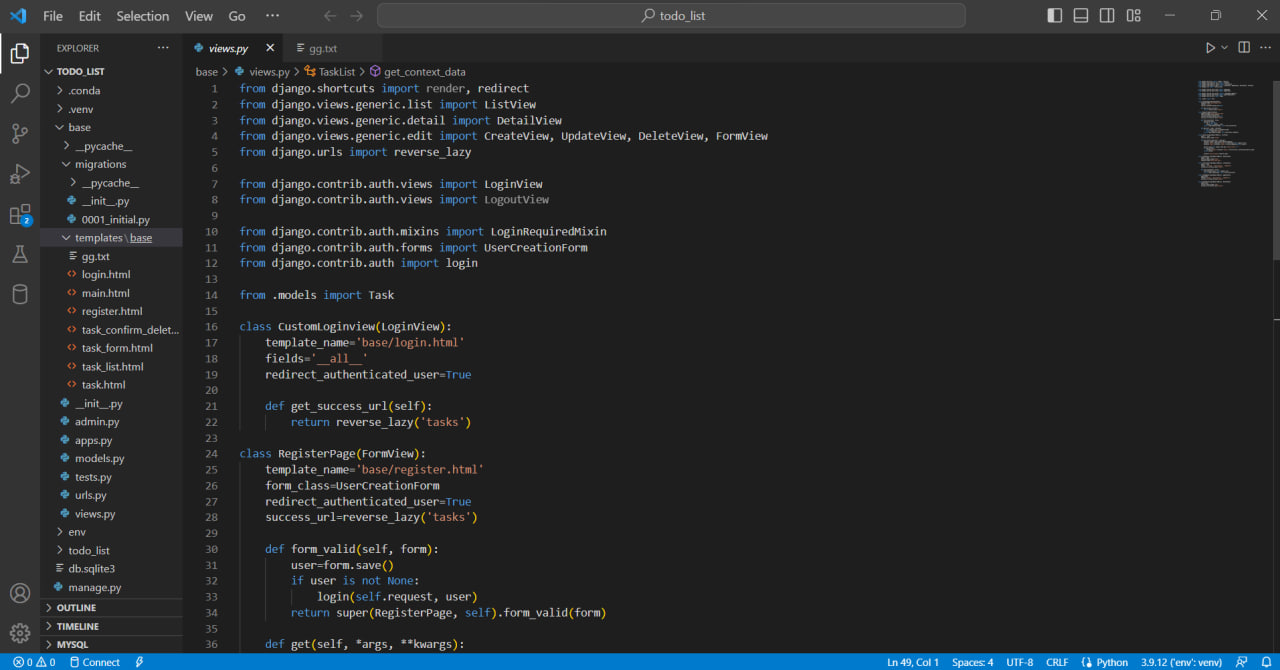


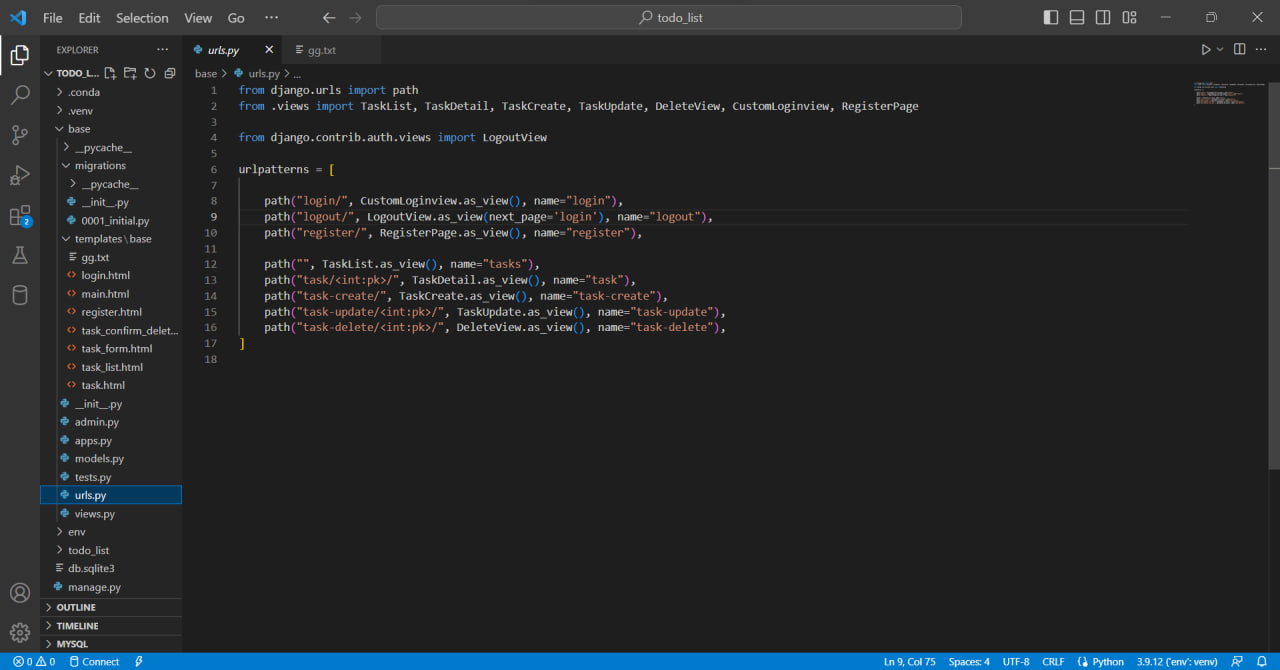
**the manual test case report**

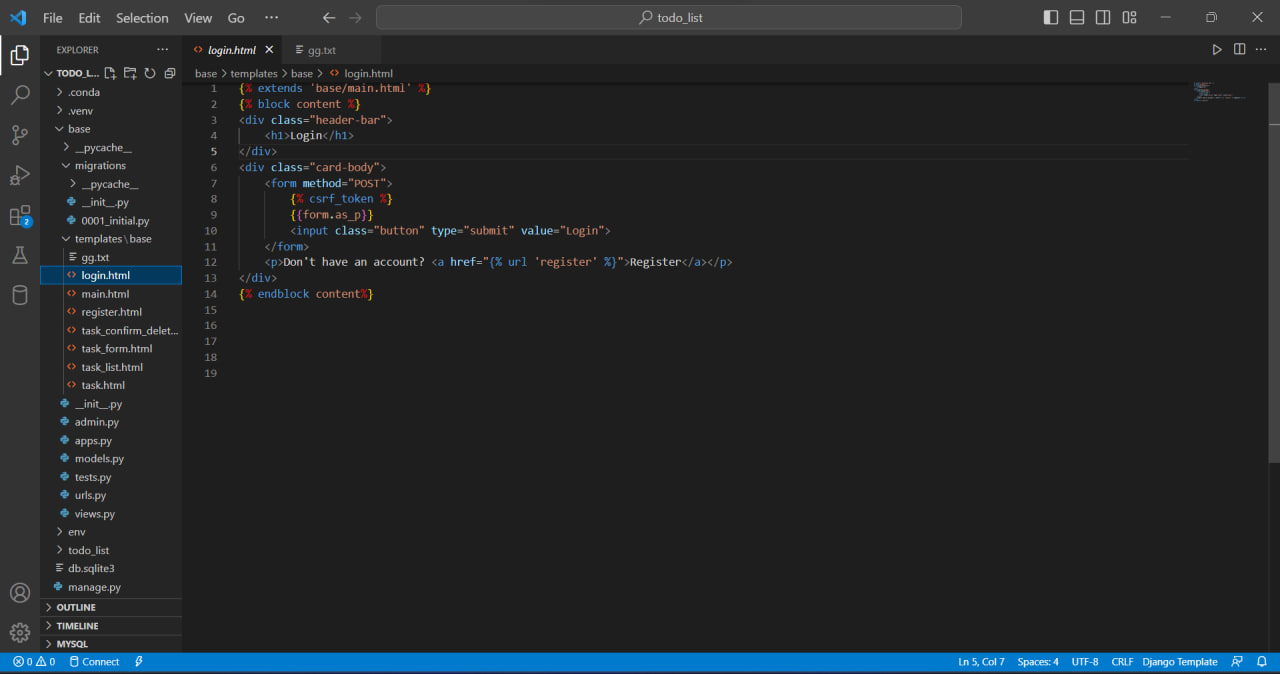
|  |  |  |
| --- | --- | --- |
| **Category** | **Progress Against Plan** | **Status** |
| Functional Testing | Green / Amber / Red | Not-Started / In-Progress / Completed |
| Check if username has alphabets |  | Completed |
| Check if the username has numerals |  | Completed |
| Check if the username has space |  | Completed |
| Check if username is in lowercase |  | Completed |
| Check if password has alphabets |  | Completed |
| Check if the password has numerals |  | Completed |
| Check if password has space in it |  | Completed |
| Check if there is a login button |  | Completed |
| Check if there a register button |  | Completed |
| Verify that a new task can be added to the list. |  | Completed |
| Verify that the task has a name and a description |  | Completed |
| Verify that the added task is displayed on the list. |  | Completed |
| Verify that a task can be edited and updated with new information, such as name and description. |  | Completed |
| Verify that the edited task is displayed correctly on the list. |  | Completed |
| Verify that a task can be deleted from the list. |  | Completed |
| Verify that the deleted task is no longer displayed on the list. |  | Completed |
| Verify that a task can be marked as completed. |  | Completed |
| Verify that the completed task is moved to the end of the list. |  | Completed |
| Verify that the user can sort tasks based on priority. |  | Not Started |
| Add color according to Priority |  | Not Started |
| Verify that the user can search for tasks based on keywords or other criteria. |  | Completed |
| Verify that the user can archive completed tasks. |  | Completed |
| Verify that the archived tasks are moved to a separate list or section. |  | Completed |
| Feedback Page |  | In- Progress |

|  |  |  |
| --- | --- | --- |
| **Functional** | **Test Case Coverage (%)** | **Status** |
| Module ID | 30% | Not-Started / In-  Progress / Completed |
| Login | 100% | Completed |
| Database | 100% | Completed |
| Delete confirmation message box | 100% | Completed |
| Feedback | 30% | In-Progress |

**FINAL CODE:**







CONCLUSION:

Hence, the documentation for the project Random Password Generator has been

implemented successfully.

This documentation will help the stakeholders and creators of the application

better understand the project and give them in-depth knowledge about the topic

and analysis for future projects.

REFRENCES:

**Geeks for geeks**: for learning technical concept.

**Draw.io:** for creating uml diagram.

**Github:** for reference code.

**Kaggle**: for sample codes.

**Django, HTML, CSS:** Creating & Styling the website.