****

**Tribhuvan University**

**Faculty of Humanities and Social Science**

**TaskEase**

**A PROJECT REPORT**

**Submitted to**

**School of Science and Technology**

**Madan Bhandari Memorial College**

***In partial fulfillment of the requirements for the Bachelors in Computer Application***

Submitted by

Rahul Khanal (6-2-432-134-2020)

Saira Shrestha (6-2-432-138-2020)

February 3

Under the supervision of

**Phul Babu Jha**

****

**Tribhuvan University**

**Faculty of Humanities and Social Science**

**Madan Bhandari Memorial College**

**Supervisor’s Recommendation**

I hereby recommend that this project prepared under our supervision by Saira Shrestha & Rahul Khanal entitled “**TaskEase**” in partial fulfillment of the requirements for the degree of Bachelor of Computer Application is recommended for the final evaluation.

**SIGNATURE**

Phul Babu Jha

**SUPERVISOR**

School of Science and Technology

Madan Bhandari Memorial College

Anamnagar, Kathmandu



**Tribhuvan University**

**Faculty of Humanities and Social Science**

**Madan Bhandari Memorial College**

**LETTER OF APPROVAL**

This is to certify that this project prepared by Saira Shrestha & Rahul Khanal entitled “**TaskEase**” in partial fulfillment of the requirements for the degree of Bachelor in Computer Application has been evaluated. In our opinion it is satisfactory in the scope and quality as a project for the required degree.

|  |  |
| --- | --- |
| ………………………..……………….. **Signature of Supervisor**  **Phul Babu Jha**  Lecturer  Madan Bhandari Memorial College  Anamnagar, Kathmandu | …………………………………………  **Signature of Coordinator**  **Phul Babu Jha**  Coordinator  Madan Bhandari Memorial College  Anamnagar, Kathmandu |
| ………………………………………… **Signature of Internal Examiner** | …………………………………………  **Signature of External Examiner** |

**TABLE OF CONTENTS**

[ABSTRACT i](#_Toc157900452)

[Acknowledgement: ii](#_Toc157900453)

[List of Abbreviations iii](#_Toc157900454)

[Table of Figures iv](#_Toc157900455)

[List of Tables v](#_Toc157900456)

[CHAPTER 1: INTRODUCTION OF PROJECT 1](#_Toc157900457)

[1.1 Introduction 1](#_Toc157900458)

[1.2 Problem Statement: 2](#_Toc157900459)

[1.3 Objectives: 2](#_Toc157900460)

[1.4 Scope and Limitation 2](#_Toc157900461)

[1.4.1 Scope 2](#_Toc157900462)

[1.4.2 Limitations of existing system 3](#_Toc157900463)

[1.5 Report Organization 3](#_Toc157900464)

[CHAPTER 2: BACKGROUND STUDY AND LITERATURE REVIEW 4](#_Toc157900465)

[2.1 Background study 4](#_Toc157900466)

[2.2 Study of existing system 4](#_Toc157900467)

[CHAPTER 3: SYSTEM ANALYSIS AND DESIGN 5](#_Toc157900468)

[3.1 System Analysis 5](#_Toc157900469)

[3.1.1 Requirement Analysis: 5](#_Toc157900470)

[i. Functional requirements: 5](#_Toc157900471)

[6](#_Toc157900472)

[ii. Non-functi3.1.4. Process Modeling (DFD)onal requirements: 7](#_Toc157900473)

[3.1.2 Feasibility study: 7](#_Toc157900474)

[i. Technical Feasibility: 7](#_Toc157900475)

[ii. Economical Feasibility: 7](#_Toc157900476)

[iii. Operational Feasibility: 8](#_Toc157900477)

[iv. Schedule Feasibility: 8](#_Toc157900478)

[8](#_Toc157900479)

[3.1.3. Data Modeling (ER-Diagram) 9](#_Toc157900480)

[9](#_Toc157900481)

[3.1.4. Process Modeling (DFD) 10](#_Toc157900482)

[3.2. System Design 13](#_Toc157900483)

[3.2.1. Architectural Design 13](#_Toc157900484)

[3.2.2. Database Schema Design 14](#_Toc157900485)

[3.2.3 Interface Design (UI Interface) 17](#_Toc157900486)

[3.2.4. Physical DFD 19](#_Toc157900487)

[CHAPTER 4: IMPLEMENTATION AND TESTING 21](#_Toc157900488)

[4.1 Implementation 21](#_Toc157900489)

[4.1.1 Tools Used (CASE tools, Programming languages, Database platform) 21](#_Toc157900490)

[4.1.2. Implementation Details of Modules( Description of procedures/functions) 22](#_Toc157900491)

[4.2. Testing 23](#_Toc157900492)

[4.2.1. Test Cases for Unit Testing 23](#_Toc157900493)

[4.2.2. Test Cases for System Testing 25](#_Toc157900494)

[CHAPTER 5: CONCLUSION AND FUTURE RECOMMENDATION 27](#_Toc157900495)

[5.1. Lesson Learnt/ Outcome 27](#_Toc157900496)

[5.2. Conclusion 27](#_Toc157900497)

[5.3. Future Recommendations 27](#_Toc157900498)

# 

# 

# ABSTRACT

This report presents the development process and outcomes of TaskEase, a task management system designed to streamline task organization and enhance productivity. TaskEase was developed using a combination of HTML, CSS, JavaScript, Node.js, and MySQL, with the aim of efficiently allocating tasks, projects, managing users and project deadlines. The report outlines the objectives, methodology, and key features of TaskEase, including its user-friendly interface.

It also discusses the challenges encountered during development and the strategies employed to overcome them, such as iterative design, user feedback, and continuous testing. Furthermore, the report highlights the future recommendations for TaskEase, including mobile optimization, integration with third-party tools, and enhanced security measures. Overall, TaskEase represents a significant achievement in task management technology, offering users a powerful and intuitive platform to effectively manage their tasks and projects.

# 

# Acknowledgement:

We express our gratitude to our professors and TU University for providing us with the opportunity to undertake this project. We believe that completing this project will serve as a valuable insight into real-world projects, enabling us to enhance our skills and knowledge.

First and foremost, I would like to express my special thanks to our supervisor **Mr. Phul Babu Jha**, who helped us a lot during the development of the system. I am thankful for his continuous feedback and encouragement throughout the project.

We are grateful to our classmates who have provided valuable insights and feedback during the proposal preparation phase. Their constructive suggestions and brainstorming sessions have enriched the overall quality of our work.

Furthermore, we acknowledge the contributions of various professionals and experts who willingly shared their knowledge and expertise during discussions, and research. Their inputs have greatly enriched our understanding of the Task Management System and influenced the proposal's effectiveness.

# List of Abbreviations

HTML: Hypertext Markup Language

CSS: Cascading Style Sheet

CASE tools: Computer Aided Software Engineering tools

DFD: Data Flow Diagram

SSR: Server Side Rendering

ER: Entity Relationship

UI: User Interface

JS: Javascript

# Table of Figures

[**Fig 1: Use Case Diagram** 6](#_Toc157901685)

[Fig 2: Gantt Chart 8](#_Toc157901686)

[Fig 3: ER-Diagram 9](#_Toc157901687)

[Fig 4: Context or 0 Level DFD 10](#_Toc157901688)

[Fig 5: 1 Level DFD 10](#_Toc157901689)

[Fig 6: 2 Level DFD 11](#_Toc157901690)

[Fig 7: Traditional SSR Architecture Design 12](#_Toc157901691)

[Fig 8: Database Schema Design 14](#_Toc157901692)

[Fig 9: Project Table 15](#_Toc157901693)

[Fig 10: User Login Table 15](#_Toc157901694)

[Fig 11: Task Table 15](#_Toc157901695)

[Fig 12: Log-in page design of TaskEase 16](#_Toc157901696)

[Fig 12: Home page design of TaskEase 16](#_Toc157901697)

[Fig 14: Context or 0 Level Physical DFD 17](#_Toc157901698)

[Fig 15: 1 Level Physical DFD 18](#_Toc157901699)

[Fig 16: 2 Level Physical DFD 19](#_Toc157901700)

[Fig: System Function Module 20](#_Toc157901701)

# List of Tables

[1. Add users Test Case: 22](#_Toc157902019)

[2. Remove users Test Case: 22](#_Toc157902020)

[3. Add projects Test Case: 22](#_Toc157902021)

[4. Delete projects Test Case: 23](#_Toc157902022)

[5. Add tasks Test Case 23](#_Toc157902023)

# 

# CHAPTER 1: INTRODUCTION OF PROJECT

## 1.1 Introduction

We propose the development and implementation of a task management system to streamline and enhance task allocation and tracking within an organization. Task management system is a digital platform that helps individuals and teams manage their tasks. These are more than just simple to-do-lists. Task management tools allow teams to collaborate digitally by organizing, prioritizing, and assigning tasks to each other.

The purpose of this proposal is to outline the problem statement, objectives, methodology, deliverables, and timeline for the project.

In today's dynamic and fast-paced work environments, effective task management is pivotal to achieving organizational success. As businesses grow and tasks become more diverse and complex, the need for a streamlined and efficient approach to task allocation, monitoring, and collaboration becomes increasingly evident. This is where a robust Task Management System comes into play, providing a comprehensive solution to enhance productivity, optimize resource allocation, and foster seamless teamwork.

TaskEase is an innovative task management software. It provides a centralized platform to efficiently manage tasks, improve communication, and enhance productivity. With its user-friendly interface and robust features, TaskEase empowers team members to focus on their work, collaborate seamlessly, and meet project deadlines.

Taskease serves as a centralized platform that empowers individuals and teams to efficiently manage tasks, track their progress, and collaborate in real time. By leveraging technology and innovative features, this system aims to overcome the challenges posed by traditional task management methods, such as manual tracking, scattered communication, and missed deadlines. It not only simplifies the task management process but also empowers organizations to make data-driven decisions for improved performance and project outcomes.

# 

## 1.2 Problem Statement:

**Problems in existing task management softwares:**

While existing task management software can greatly improve productivity and collaboration, there are some common problems that users may encounter.

* Complexity and Overwhelming features: Some task management software can be overly complex, with a multitude of features that may not be relevant to all users.
* Lack of Intuitive User Interface: If the user interface of a task management software is not intuitive and user-friendly, it can hinder adoption and lead to confusion among users.
* Insufficient Task Allocation: Task assignment and tracking features may not be robust enough in certain software, making it challenging to allocate tasks effectively and monitor their progress.

## 1.3 Objectives:

The main objectives of TaskEase are:

* To optimize task distribution.
* To help users stay organized, meet deadlines, and allocate their time efficiently across multiple tasks.
* To enable individuals and teams to focus on critical tasks.

## 1.4 Scope and Limitation

### 1.4.1 Scope

TaskEase has great future scope. This project is developed to to efficiently manage tasks, improve communication, and enhance productivity. It has a simple and friendly user interface to get around and use. This project also provides security with login id and password so that unauthorized users cannot access it. The current system is confined to optimize task distribution. It can be extended to have more features in the future like progress tracking, notifications and reminders, customization options, etc.

### 1.4.2 Limitations of existing system

* Lack of communication and collaboration tools.
* No reporting and analytics features.

## 1.5 Report Organization

The first part of the report contains a summarized introduction of the whole report. It contains problem statements, objectives, scope and limitations of the project.

The second chapter includes background story i.e description of fundamental theories, general concepts and terminologies related to the project. It also contains literature review i.e review of the similar projects, theories done by other researchers.

The third chapter includes system analysis and design where functional and non-functional requirements are stated and illustrated using use case diagrams. It includes Technical, Operational, Economic and Schedule feasibility of the project development phase. The explanation of designing the system is also done in this chapter. It includes data modeling and process modeling which is explained by using ER-Diagram and DFD. Architectural design, Database design and Interface design are also listed in this chapter.

The fourth chapter includes the implementation and testing phase of the proposed system. It contains CASE tools, programming languages and database platforms used in the system. Unit testing and System testing are also included in this chapter.

The fifth chapter includes conclusion and future recommendations. It contains outcomes of the project and lessons learnt during the developing phase.

# CHAPTER 2: BACKGROUND STUDY AND LITERATURE REVIEW

## 2.1 Background study

A background study of task management systems reveals a dynamic landscape characterized by a plethora of tools catering to diverse user needs and preferences. From established platforms like Trello, Asana, and Jira to emerging solutions, the market offers a range of features including task creation, assignment, prioritization, and tracking, often integrated with collaboration and communication functionalities. User feedback underscores the importance of intuitive interfaces, robust mobile accessibility, and seamless integration with existing workflows. Key considerations such as regulatory compliance, technological advancements, and evolving user expectations drive ongoing innovation in this space, shaping the development of task management systems to enhance productivity and streamline workflow management.

## 2.2 Study of existing system

a. Jira:

Jira is a versatile task management system primarily focused on agile project management. It allows users to create and manage tasks, track progress, and collaborate on software development projects.Some of the organizations that have used Jira at some point in time for bug-tracking and project management include Fedora Commons,Hibernate, and the Apache Software Foundation, which uses both Jira and Bugzilla.Jira is written in Java and uses the Pico inversion of control container, Apache OFBiz entity engine, and WebWork 1 technology stack.

b. Trello:

Trello is a web-based, kanban-style, list-making application and is developed by Trello Enterprise, a subsidiary of Atlassian created in 2011 by Fog Creek Software. Users can create their task boards with different columns and move the tasks between them. Typically columns include task statuses such as To Do, In Progress, Done. The tool can be used for personal and business purposes including real estate management, software project management, school bulletin boards, lesson planning, accounting, web design, gaming, and law office case management.

c. Todoist:

Todoist is a popular task management system that offers a simple and intuitive interface. It allows users to create tasks, assign priorities, set due dates and reminders, and organize tasks into projects or labels. Todoist supports collaboration by enabling task sharing and commenting. It also provides features like task recurrence, productivity tracking, and integration with various platforms and tools.

# CHAPTER 3: SYSTEM ANALYSIS AND DESIGN

## 3.1 System Analysis

### 3.1.1 Requirement Analysis:

For this Task Management System, requirements are basically identified through functional and non-functional requirements.

#### i. Functional requirements:

* User Registration and Login:

The system should allow users to register for an account and login to access its features. The new users should be able to register an account and existing users should be able to login into Taskease to enjoy its features.

* Task Creation:

Users should be able to create new tasks and assign them to specific individuals or teams within the office. The system should allow for the assignment of due dates, priorities, and task descriptions.The system allows the assignment of task dependencies if needed.

* Task Prioritization and Sorting:

The system should provide options for sorting tasks based on different attributes such as due date, priority, or assignee. The system should provide options for sorting tasks based on different attributes such as due date, priority, or assignee.Users can prioritize tasks based on urgency, importance, or other criteria.

* Deadline Management:

The system should inform users about task deadlines. The system should send notifications or reminders to task assignees and supervisors as task deadlines approach or when they are overdue. Users should be able to easily identify tasks with upcoming or missed deadlines.

Functional requirements can be expressed in Use Case diagrams as they exhibit externally visible functional behavior.

# 

##### **Fig 1: Use Case Diagram**

#### ii. Non-functi3.1.4. Process Modeling (DFD)onal requirements:

* Usability and User Experience:

The system should have an intuitive and user-friendly interface, providing a positive user experience.Navigation is straightforward, and actions are easily understandable.

* Performance:

The system should perform efficiently, even with a large number of tasks and users, ensuring quick response times and minimal downtime.The system must handle a large number of tasks and users without significant performance degradation.Response times should be quick even during peak usage.

* Security and Data Privacy:

The system should implement robust security measures to protect user data , including secure user authentication, data encryption, and access controls.Data encryption is implemented to protect sensitive information.

* Maintenance and Support:

The system should be maintainable, with regular updates, bug fixes, and technical support provided to users. Technical support is provided to address user queries and issues.Comprehensive user documentation, tutorials, and support resources are available.

### 3.1.2 Feasibility study:

#### i. Technical Feasibility:

The task management software is being developed using modern web technologies for frontend such as HTML5, CSS3, and JavaScript, for backend such as Nodejs. The required infrastructure, including servers, databases, and hosting services, is readily available.Compatibility testing will be conducted to ensure the software works seamlessly on major web browsers and mobile devices.

#### ii. Economical Feasibility:

This task management system is being developed with a focus on cost-efficiency, ensuring that it can be implemented at a minimal expense. It is designed to be accessible nationwide, providing a budget-friendly solution for efficient task management.

### 

#### iii. Operational Feasibility:

The task management software aligns with existing task management processes and workflows within the organization.The system is designed to be user-friendly and intuitive, minimizing resistance to change.Training and support resources will be provided to ensure a smooth transition and adoption by the users.

#### iv. Schedule Feasibility:

This includes the project schedule and time allocated for the completion.

## 

##### Fig 2: Gantt Chart

## 

### 3.1.3. Data Modeling (ER-Diagram)

An ER (Entity-Relationship) diagram is a visual representation of the structure of a database, showing entities, their attributes, and the relationships between them. Entities are represented as rectangles, attributes as ovals connected to their respective entities, and relationships as lines connecting entities. The diagram helps stakeholders understand the data model, including how different pieces of information relate to each other and the constraints on those relationships.

## 

##### Fig 3: ER-Diagram

# 

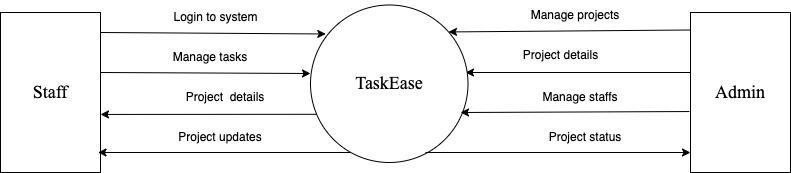
# 

# 

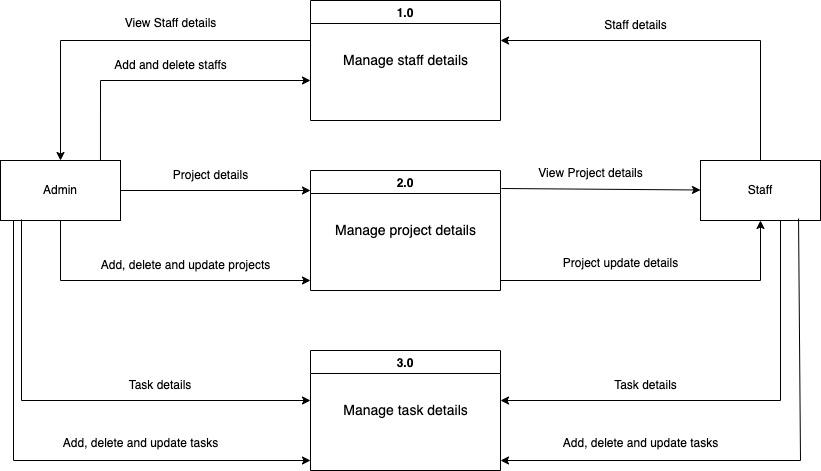
### 3.1.4. Process Modeling (DFD)

Process modeling is a method used to visually represent the flow of activities, decisions, and interactions within a system or business process. It involves creating diagrams or models that illustrate how work is performed, including the sequence of tasks, decision points, inputs, outputs, and the roles of participants.

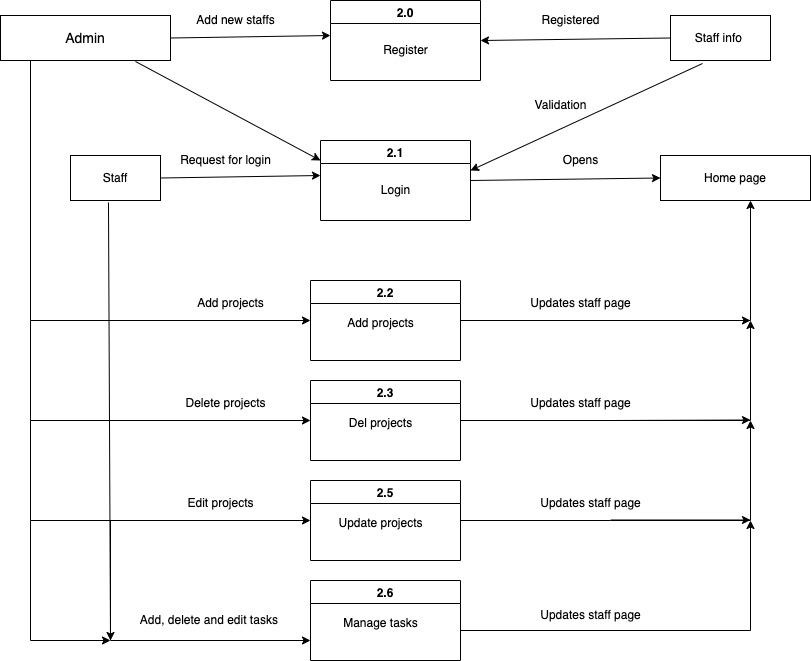
A Data Flow Diagram (DFD) is a graphical representation of the flow of data within a system, illustrating how data moves between processes, data stores, and external entities. In a DFD, processes are represented by circles or rectangles, data stores are represented by open-ended rectangles, and external entities are represented by squares. Arrows indicate the flow of data between these elements.



##### Fig 4: Context or 0 Level DFD



##### Fig 5: 1 Level DFD



##### Fig 6: 2 Level DFD

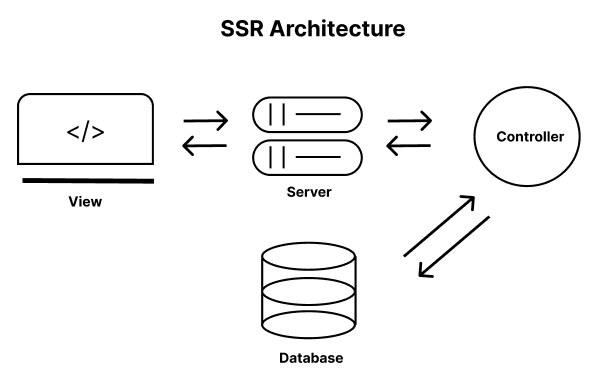
# 

## 3.2. System Design

### 3.2.1. Architectural Design

Architectural design refers to the process of defining the overall structure, components, relationships, and behavior of a software system or application. It involves making high-level design decisions that determine how the system will be organized and how its various components will interact with each other to achieve the desired functionality.

SSR stands for Server-Side Rendering, which is an architectural pattern used in web development to render web pages on the server before sending them to the client's browser. In Traditional SSR architecture, the server processes the request for a web page, retrieves the necessary data, and generates the HTML content for the page dynamically. The server then sends the fully rendered HTML page to the client's browser, which can display it to the user immediately.

**￼**

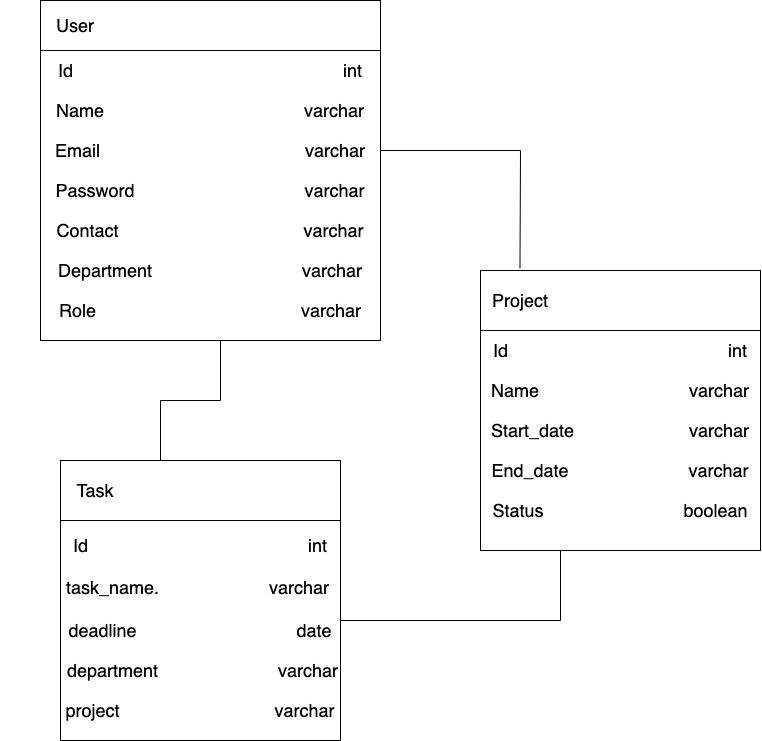
##### Fig 7: Traditional SSR Architecture Design

### 3.2.2. Database Schema Design

Database schema design refers to the process of creating a logical structure or blueprint for organizing and representing data in a database. It involves defining the tables, columns, relationships, constraints, and other elements that comprise the database schema, which serves as the foundation for storing and managing data in the database.

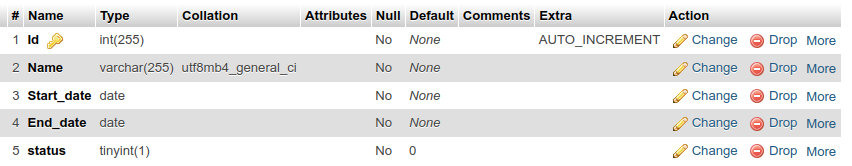
Database named TaskEase was created to do all the operations of our Task Management System. We have 3 tables:

* User
* Project
* Task



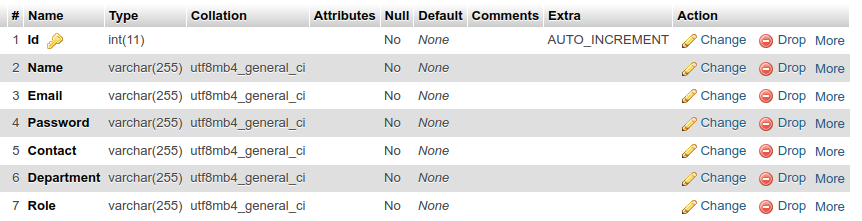
##### Fig 8: Database Schema Design

A table name “Project” is created to add new projects. Admin can add different projects with its name, start and end date and can update the project status.

****

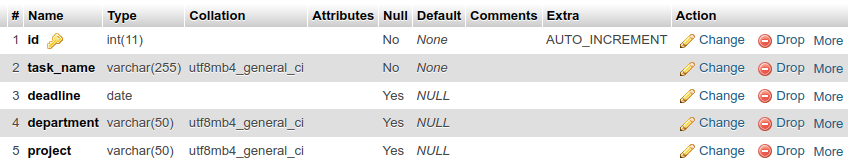
##### Fig 9: Project Table

A table name “User” is created which is both an admin and staff login panel where he/she has to login into the system with email and password. Admin can add staff in to the system.

****

##### Fig 10: User Login Table

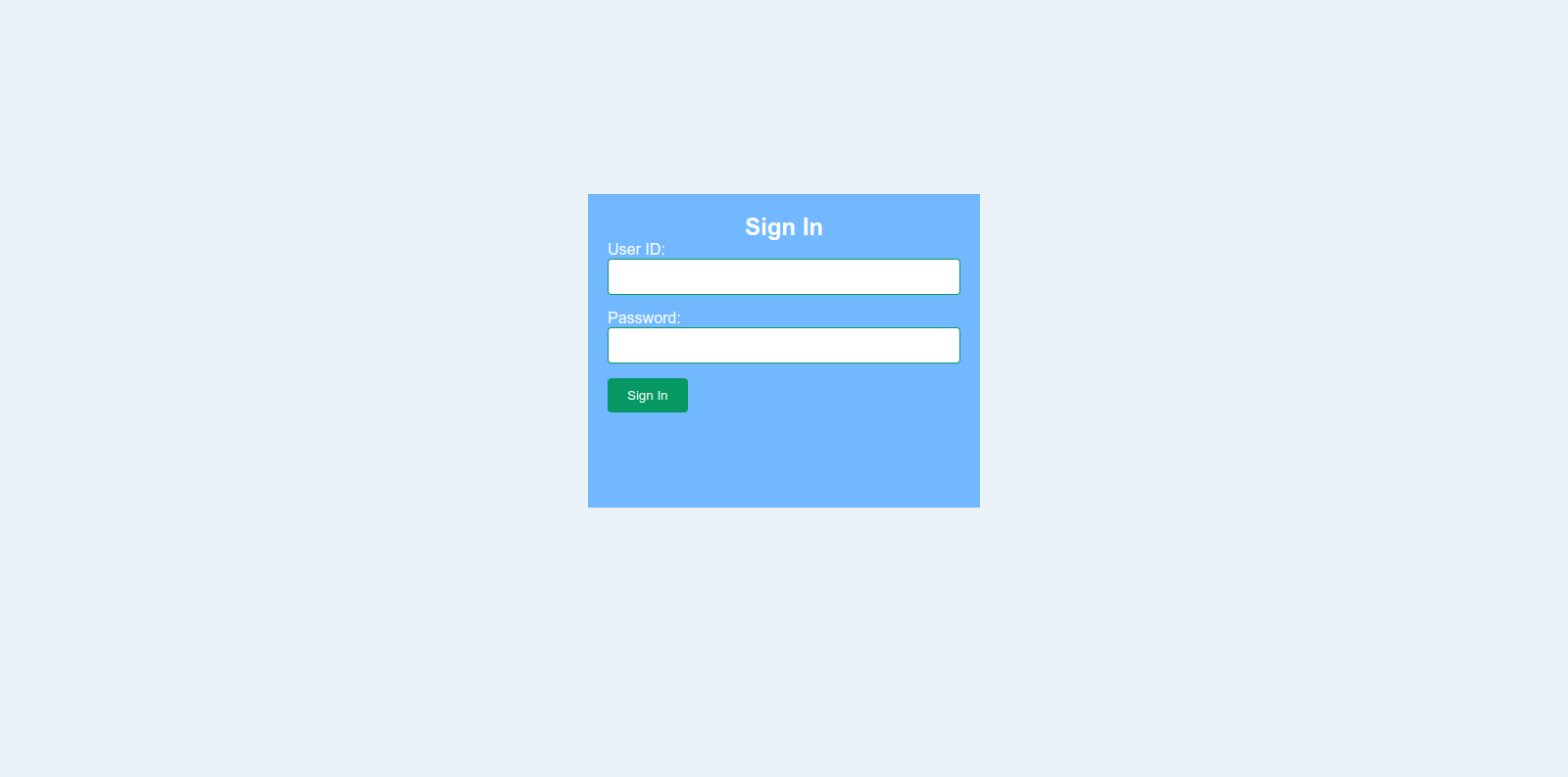
A table named “Task” is created to add new tasks. Both admin and staff can add and manage tasks.

****

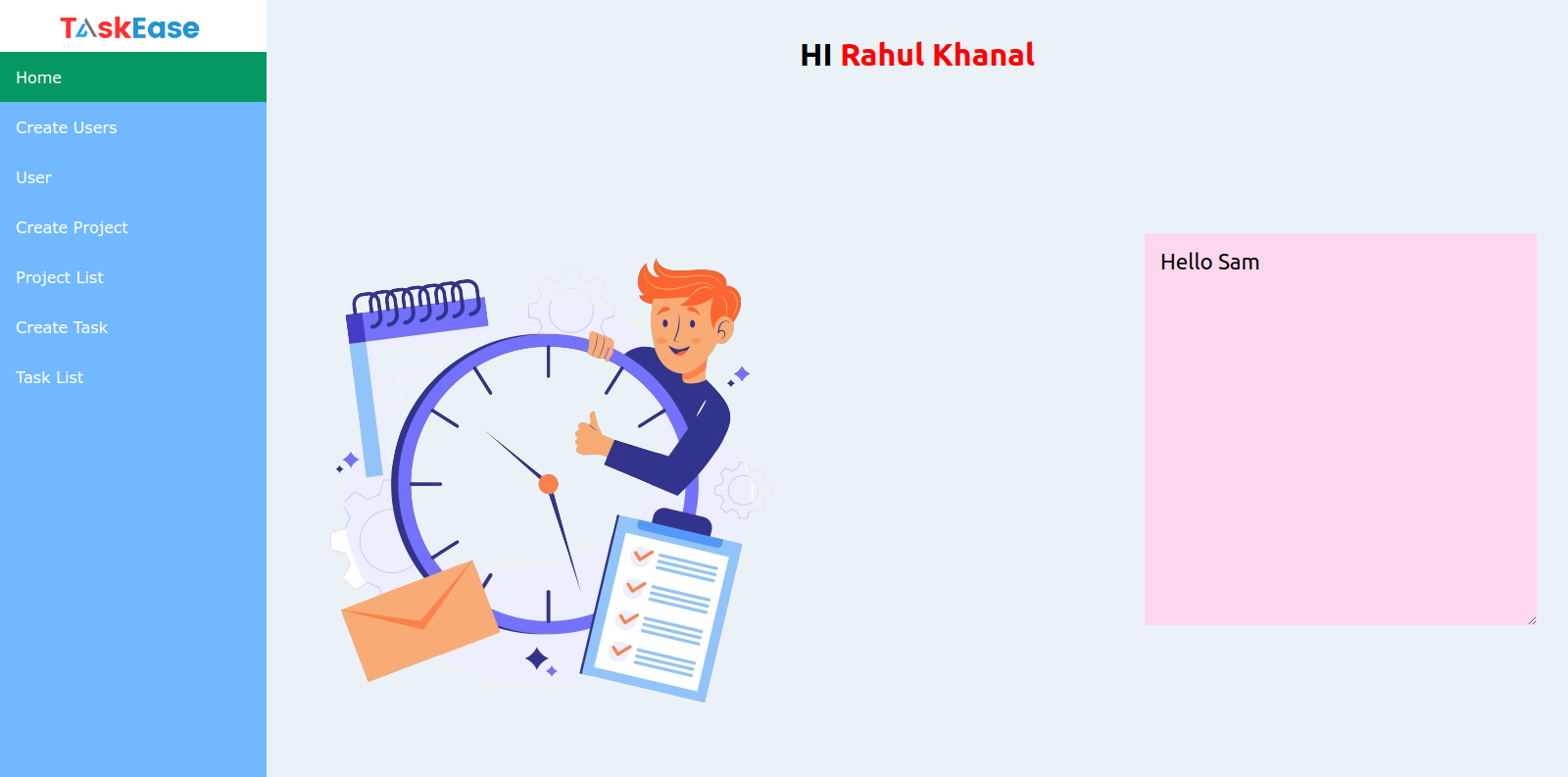
##### Fig 11: Task Table

### 3.2.3 Interface Design (UI Interface)

Interface design refers to the process of designing the user interface (UI) of a software application or system. It involves creating visually appealing, intuitive, and user-friendly interfaces that enable users to interact with the system effectively and efficiently. Interface design encompasses various elements, including layout, navigation, controls, graphics, and typography, all aimed at enhancing the user experience (UX) and usability of the application.



##### Fig 12: Log-in page design of TaskEase

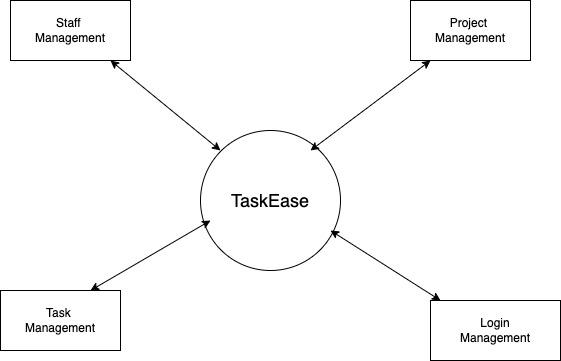
****

##### Fig 12: Home page design of TaskEase

### 3.2.4. Physical DFD

A Physical Data Flow Diagram (DFD) is a representation of the data flow within a system that emphasizes the physical aspects of the data flow, including the specific processes, data stores, and data flows involved in the system's implementation. Unlike a logical DFD, which focuses on the logical flow of data independent of implementation details, a physical DFD provides a more detailed view of how data moves through the system and interacts with physical components.

All of the below diagrams have been visualized for data processing and structured design of TaskEase.

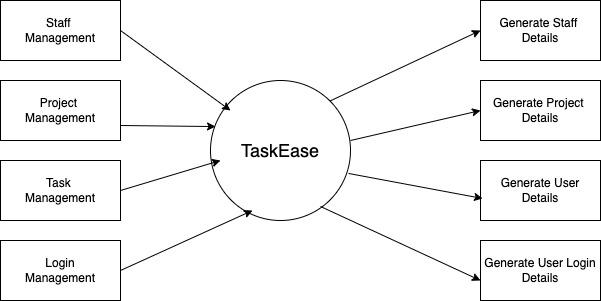


##### Fig 14: Context or 0 Level Physical DFD

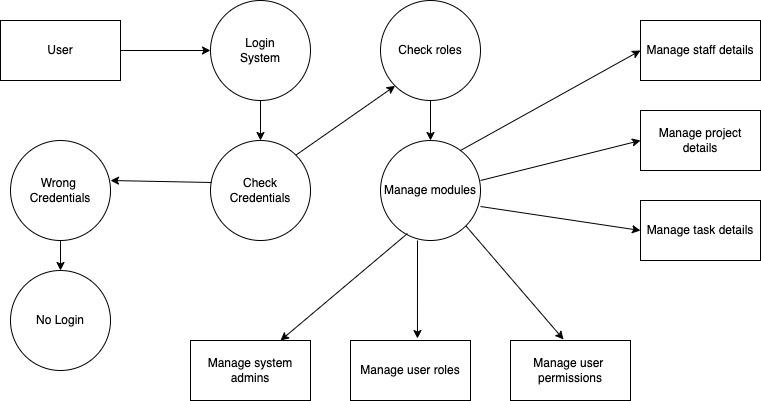
A 0-level Physical Data Flow Diagram (DFD) provides an overview of the entire system from a physical perspective, showing the main components and data flows without the internal details of processes or data stores.

A 1-level Physical Data Flow Diagram (DFD) provides a more detailed view of the system compared to the 0-level DFD.

The 2-level DFD provides a deeper insight into how data is processed and flows through the system, making it valuable for system analysis, design, and implementation.

****

##### Fig 15: 1 Level Physical DFD

****

##### Fig 16: 2 Level Physical DFD

# CHAPTER 4: IMPLEMENTATION AND TESTING

## 4.1 Implementation

### 4.1.1 Tools Used (CASE tools, Programming languages, Database platform)

Frontend:

* HBS(Handlebars.js):

In TaskEase, we are using HBS. HBS is a templating engine for creating dynamic HTML templates in web development, often used with Node.js. It allows developers to create templates with placeholders for data and then render those templates with actual data to generate HTML content.

* CSS(Cascading Style Sheets):

In TaskEase,we are using CSS to define the visual presentation of a web page. It allows you to control the layout, colors, fonts, spacing, and other stylistic aspects of HTML elements.By separating content from presentation, CSS enables greater control over the visual appearance of a website, making it easier to create responsive and visually appealing designs across different devices and screen sizes.

* JavaScript:

In TaskEase, we are using JavaScript. JavaScript is a scripting language that adds interactivity and dynamic behavior to web pages. It enables you to manipulate HTML and CSS, handle user interactions, perform calculations, make AJAX requests, and more. It is commonly employed to add behavior, interactivity, and functionality to websites, such as form validation, DOM manipulation, and event handling.

Backend:

* NodeJS:

We are using Node.js for server-side scripting. Node.js is a runtime environment that allows developers to run JavaScript code outside of a web browser, typically on the server side.Node.js is well-suited for building scalable and real-time web applications, handling API requests, and connecting frontend and backend. It has a rich set of built-in modules and libraries that facilitate tasks such as handling HTTP requests and working with databases.

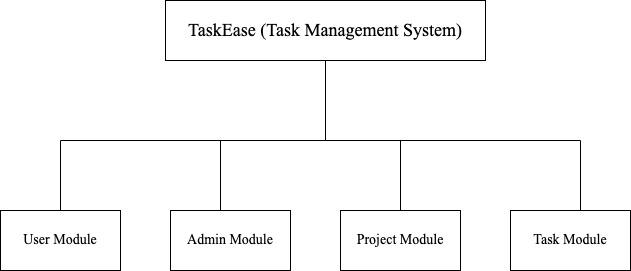
### 

Database:

* MySQL: MySQL is an open-source relational database management system (RDBMS) that provides a robust and scalable solution for storing, managing, and retrieving data. In our Task Management System we are using MySQL to store and retrieve data for the backend API built with NodeJS. MySQL supports SQL (Structured Query Language) for querying and manipulating data, making it a popular choice for organizations of all sizes seeking a robust and cost-effective database solution.

### 4.1.2. Implementation Details of Modules( Description of procedures/functions)

A task management system typically consists of several modules that work together to facilitate the creation, organization, tracking, and completion of tasks. Here are some modules found in our task management system:

****

##### Fig: System Function Module

* User module:

This module handles user authentication, authorization, and user profile management. It allows users to create accounts, log in securely, and manage their personal information.

* Admin module:

The admin module can do all the internal functionalities of the system adding new projects with due dates, assigning them to particular teams editing and updating projects, adding users and their information, etc. This is a role based module where admin can perform every operation on data of our system.

* Project module:

This module enables the user to view the project with its details. They can view the project title, start and end date, and the team assigned for it.

* Task module:

This module enables users to create new tasks within the projects, specify task details such as title, description, due date, assignee, and edit existing tasks as needed.

## 4.2. Testing

Testing is a critical phase in the software development lifecycle aimed at ensuring that the developed software meets the specified requirements, functions correctly, and is free from defects. It involves executing the software system or application under controlled conditions to identify errors, bugs, or other issues that may affect its performance, reliability, or usability.

### 4.2.1. Test Cases for Unit Testing

Unit testing is a fundamental aspect of software development aimed at verifying the correctness and functionality of individual units or components of a software system. These units can be functions, methods, classes, or modules. The main objective of unit testing is to isolate and test each unit in isolation, ensuring that it behaves as expected and meets its specified requirements.

###### Add users Test Case:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test case no. | Test Scenario | Test Steps | Test Data | Expected Result | Pass/Fail |
|  | Adding users using invalid data | Fill Create User Form | Samrajya Rana  [samrana@yahoo.com](mailto:samrana@yahoo.com)  89611176855  Designing  Admin  54/+$$$7$/ | Validation failed | Fail |
|  | Adding users using valid data | Fill Create User Form | SAMRAJYA RANA  [samrana@yahoo.com](mailto:samrana@yahoo.com)  98611176855  Designing  Admin  54/+$$$7$/ | Successfully created a user | Pass |

###### Remove users Test Case:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test case no. | Test Scenario | Test Steps | Test Data | Expected Result | Pass/Fail |
|  | Deleting user | Clicking delete button | SAMRAJYA RANA | Delete user data and reload page | Pass |

###### Add projects Test Case:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test case no. | Test Scenario | Test Steps | Test Data | Expected Result | Pass/Fail |
|  | Creating project with invalid detail | Submit form from Create Project | Empty data(NULL) | Validation failed | Fail |
|  | Creating project with valid detail | Submit form from Create Project | Medicine Inventory System  02/08/2024  01/12/2024 | Successfully created a project | Pass |

# 

###### Delete projects Test Case:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test case no. | Test Scenario | Test Steps | Test Data | Expected Result | Pass/Fail |
|  | Deleting projects | Clicking delete button | Medicine Inventory System | Delete project and reload the page | Pass |

###### Add tasks Test Case

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test case no. | Test Scenario | Test Steps | Test Data | Expected Result | Pass/Fail |
|  | Creating tasks with empty data | Fill the form of Create Task | NULL | Validation Failed | Fail |
|  | Creating tasks with valid data | Fill the form of Create Task | UI/UX Design | Successfully added task | Pass |

### 4.2.2. Test Cases for System Testing

System testing is a crucial phase in the software testing process that evaluates the functionality, performance, reliability, and overall behavior of a complete software system or application. Unlike unit testing, which focuses on testing individual units or components in isolation, system testing examines the system as a whole, including its interactions with external dependencies and components.

1. Test Case of the System:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test case no. | Test Scenario | Test Steps | Test Data | Expected Result | Pass/Fail |
|  | Admin Login | Login credentials | rahul39[khanal@proton.com](mailto:khanal@proton.com)  QAZwsx@321 | Go to Admin home page | Pass |
|  | Creating Projects | Submit form from Create Project | Medicine Inventory System  02/08/2024  01/12/2024 | Successfully created a project | Pass |
|  | Adding users | Fill Create User Form | SAMRAJYA RANA  [samrana@yahoo.com](mailto:samrana@yahoo.com)  98611176855  Designing  Admin  54/+$$$7$/ | Successfully created a user | Pass |
|  | Creating tasks with valid data | Fill the form of Create Task | UI/UX Design | Successfully added task | Pass |

# 

# 

# 

# CHAPTER 5: CONCLUSION AND FUTURE RECOMMENDATION

## 5.1. Lesson Learnt/ Outcome

Developing a task management system has yielded valuable lessons and outcomes, including a deeper understanding of user needs and preferences for organizing tasks efficiently. We learned that making the system easy to use and helpful is super important. We also had to figure out technical stuff like making sure all the data stays up-to-date and making the system work well even when lots of people are using it. Collaboration among team members, including developers, designers, and stakeholders, is essential for successful development. Thorough testing and quality assurance processes are vital for ensuring the reliability, functionality, and performance of the task management system. Developing a task management system offers opportunities for continuous learning and adaptation. By analyzing user feedback, monitoring system usage, and staying informed about emerging technologies and trends, developers can continuously improve the system to better serve user needs and preferences.

## 5.2. Conclusion

In conclusion, the development journey of TaskEase, our task management system built using HTML, CSS, JavaScript, Node.js, and MySQL, has been a fulfilling and rewarding experience. Through careful planning and diligent implementation, we have successfully achieved all our objectives, including efficient task allocation and management, timely project deadlines, and seamless user interaction. Leveraging the power of modern web technologies, we have created a user-friendly and intuitive platform that simplifies task organization and enhances productivity.

## 5.3. Future Recommendations

This project has been developed in such a way that it can accept modifications and further changes. TaskEase has a promising future with several recommendations aimed at further enhancing its functionality and user experience. Customization options, collaboration features, and enhanced security measures will be essential for tailoring TaskEase to users' specific needs and ensuring data privacy. Furthermore, the implementation of reporting and analytics capabilities, along with comprehensive user training and support resources, will empower users to optimize their task management processes and achieve greater productivity. Looking ahead, there are several future recommendations to further enhance the functionality and usability of TaskEase:

* Generate analytic reports and progress tracking
* Clockin/ Clockout feature to track the staff working hour
* Kanban for task prioritization
* Communication and collaboration tools
* Leave management