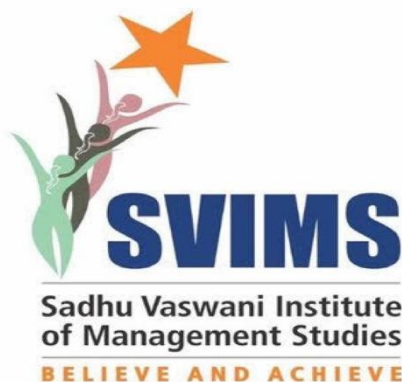


**A**  
**PROJECT REPORT ON**  
**TASK MANAGEMENT SYSTEM**

**SUBMITTED BY**  
**Ms. Chinmayee Uday Daithankar**

**SUBMITTED TO**  
**SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE**  
**IN FULFILLMENT OF DEGREE**  
**MASTER OF COMPUTER APPLICATION(SEM-1)**

**UNDER THE GUIDANCE OF**  
**Dr . Shveti. Chandan**  
**Through,**



**Sadhu Vaswani Institute of Management Studies for Girls,**  
**Koregaon Park, Pune-411001**  
**2024-2025**

## DECLARATION BY STUDENT

To,  
The Director,  
SVIMS, Koregaon-Park, Pune

I , undersigned hereby declare that this project titled, “Task Management System” written and submitted by me to SPPU, Pune , in partial fulfilment of the requirement of the award of the degree of MASTER OF COMPUTER APPLICATION(MCA-1) under the guidance of , Dr . Shveti. Chandan, is my original work.

I further declare that to the best of my knowledge and belief, this project has not been submitted to this or any University or Institution for the award of any degree.

Place: Pune

Date:

(Chinmayee Uday Daithankar)

## ACKNOWLEDGEMENT

I extend my sincere gratitude to Dr. B. H. Nanwani , Dr. Neeta Raskar , and Dr. Shveti. Chandan for allowing me to carry out the study and for their constant encouragement, valuable, suggestion, and guidance during the research work.

I extend my special thanks to my parents and friends for supporting me for the research and inspiration.

I extend my special thanks to the guidance I received from the websites, videos, and books improve my knowledge to over come the challenges while developing the project.

Place: Pune

Date: (Chinmayee Uday Daithankar)

## **CHAPTER 1: INTRODUCTION**

### **1.1 Client/Organization Profile**

Name : Syntax Tech Company Pvt

Location : Pune

About

Organization

Syntax Tech Company has achieved considerable growth over the past few years, expanding its product offerings and client base. However, with this growth, managing tasks, tracking progress, and ensuring timely completion has become increasingly complex. Currently, teams rely on traditional methods such as emails, spreadsheets, and verbal communication, which have led to inefficiencies and miscommunication. The adoption of a comprehensive task management system would help streamline operations, improve accountability, and foster collaboration.

This report outlines the critical need for implementing a task management system at Syntax Tech Company. As the company continues to grow, it faces increasing challenges in coordinating and tracking tasks across teams and departments. A task management system would significantly improve operational efficiency, enhance communication, and provide a clearer overview of project statuses.

### **1.2 Need of the System**

#### **Improved Task Visibility and Tracking:**

A centralized system allows for real-time updates, enabling employees and managers to see which tasks are in progress, who is responsible, and when they are due. Automated notifications and reminders help ensure deadlines are met and tasks do not fall through the cracks.

#### **Enhanced Collaboration:**

Teams can collaborate more effectively by sharing updates, documents, and feedback within the system. Communication will be streamlined, reducing reliance on email chains and improving the exchange of information.

### **Clearer Accountability:**

With a task management system, every task is assigned to a specific person, creating clear accountability. Manager can track performance and ensure that responsibilities are clearly outlined.

### **Efficient Project Monitoring and Reporting:**

A task management system provides built-in reporting features that make it easier for project managers to track progress, identify bottlenecks, and allocate resources accordingly. Reports on task completion and project timelines can be generated automatically, saving time and effort.

### **Scalability and Flexibility:**

As the company continues to grow, a task management system can scale with the organization's needs. It can be adapted to accommodate various teams, departments, and project types, ensuring long-term relevance.

## **1.3 Scope and Feasibility of Work**

### **User Requirement**

Collecting the details of the user and enable them to manage their task as such that there is an report generation of their work collecting detailed requirements from all departments (development, marketing, sales, operations, etc.) to understand the unique needs and challenges faced by each.

### **System Features**

The system will support task creation, deletion, editing, email and reporting functionalities. It will also include collaboration features such as task file attachments, and graph generation of the progress report.

**User Roles :** Defining user roles (Manager and Employee) can easily use the system to create task and track their progress. Both the roles have the same features.

### **Technical Feasibility**

- **System Requirements:** The system should be compatible with the company's current hardware and software environment. It is an company requirement solution , it must meet the technical specifications .
- **Scalability:** The task management system must be scalable to accommodate future growth, both in terms of users (adding more teams or departments) and functionality (supporting larger projects or more complex task tracking).

- **Security Considerations:** The system must meet industry security standards, including data encryption, user authentication, and secure access to ensure that sensitive company information is protected.

#### **Economical Feasibility**

**Cost of Acquisition:** The financial investment required will depend on whether the company chooses a commercial off-the-shelf solution or a custom-built system. The costs will include licensing fees, customizations, and development (if applicable).

**Training and Implementation Costs:** The company will need to budget for training programs, onboarding sessions, and potential consulting fees for system customization. By improving task efficiency, reducing project delays, and enhancing team collaboration, the task management system will ultimately provide a positive return on investment through increased productivity and reduced operational bottlenecks.

### **1.4 Operating Environment – H/w & S/w**

#### **Client Side System Specification**

<b>Device</b>	<b>Specification</b>
<b>Laptop / Desktop</b>	<ul style="list-style-type: none"> <li>• Minimum AMD Ryzen 5 5500U with Radeon Graphics 2.10 GHz</li> <li>• Minimum 8.00 GB (5.85 GB usable)</li> <li>• Minimum Windows 8 and above</li> </ul>

#### **Software Specification**

<b>Particular</b>	<b>Specification</b>
<b>Operating System</b>	<ul style="list-style-type: none"> <li>• Windows 7 and above</li> <li>• Linux 6.1 or above</li> <li>• Mac OS 10.1 or above</li> </ul>
<b>Browser</b>	<ul style="list-style-type: none"> <li>• Google Chrome</li> <li>• Microsoft Edge</li> <li>• Higher Internet Explorer</li> </ul>

#### **Server Side Specification**

<b>Database</b>	<ul style="list-style-type: none"> <li>• MySQL 8.0.35 or MySQL Workbench</li> </ul>
<b>Browser</b>	<ul style="list-style-type: none"> <li>• Google Chrome</li> </ul>
<b>Server</b>	<ul style="list-style-type: none"> <li>• Django Built-In Server</li> </ul>

## Developer Side System Specification

Device	Specification
Laptop / Desktop	<ul style="list-style-type: none"><li>• Minimum AMD Ryzen 5 5500U with Radeon Graphics 2.10 GHz</li><li>• Minimum 8.00 GB (5.85 GB usable)</li><li>• Minimum Windows 11</li></ul>

## Software Specification

Particular	Specification
Django Framework	<ul style="list-style-type: none"><li>• Django-admin –version (5.1.2)</li></ul>
MySQL	<ul style="list-style-type: none"><li>• MySQL Workbench 8.0 CE</li></ul>
Visual Studio	<ul style="list-style-type: none"><li>• Vs Studio Frame Work 1.86</li></ul>
Browser	<ul style="list-style-type: none"><li>• Google Chrome</li></ul>

## 1.5 Architecture of the System

### 1. High-Level Architecture Overview

A Task Management System (TMS) generally follows a layered architecture that can be broken down into the following components:

- **Frontend (User Interface):** The interface where users interact with the system.
- **Backend (Business Logic Layer):** The layer that handles business logic and communicates with data storage and other services.
- **Database:** A storage system to persist data such as tasks, users, statuses, etc.
- **API Layer (RESTful ):** Provides a communication layer between the frontend and backend.
- **Authentication & Authorization:** Handles user login, roles, and permissions.
- **Notification System:** Sends out alerts and updates (via email, SMS, or in-app).
- **Analytics and Reporting:** Generates reports and visualizations for task performance and team productivity.

### 2. Key Components and Their Role

### a. Frontend (UI/UX)

The frontend layer provides a user-friendly interface where users can interact with the task management system. This includes web that allow users to:

- Create, update, and delete tasks.
- Assign tasks to users or teams.
- Track the status and progress of tasks.
- Set due dates, priorities, and dependencies.
- View dashboards or reports.

#### Technologies:

- **Web:** Django Framework, MySQL, HTM, JavaScript , CSS

### b. Backend (Business Logic Layer)

The backend is responsible for the core business logic and performing actions based on requests from the frontend. It handles:

- **Task creation, editing, deletion:** Ensures data consistency and integrity.
- **User management:** Handling user roles.
- **Task status management:** Managing task statuses such as "Not Complete" and "Completed".

#### Technologies:

- **Languages:** Python, HTML, JavaScript, CSS
- **Frameworks:** Django (Python), Visual Studio, MySQL Workbench

### c. Database Layer

The database is where all task-related data is stored, including tasks, users, comments, and any other associated data like labels or attachments.

- **Relational Database (SQL):** Use a relational database if you need complex queries and structured relationships, e.g., MySQL
- **NoSQL Database:** If you need flexibility in storing varied data types and scalability, consider using MongoDB, Firebase, or similar.

#### Key Tables/Entities:

- **Users:** Information about users, roles, and permissions.
- **Tasks:** Task attributes like title, description, due date, priority, status, etc.
- **Assignments:** Mapping of tasks to users.



- **Attachments:** Files attached to tasks.

#### **d. API Layer**

A RESTful API allows the frontend to communicate with the backend. It exposes endpoints for:

- Creating, updating, and deleting tasks.
- Managing users and their roles.
- Fetching tasks based on filters (e.g., due date, priority).
- Sending notifications, and more.

#### **e. Authentication & Authorization**

This layer manages user authentication (who is accessing the system) and authorization (what resources a user can access). Key features include:

- **Login and Registration:** Using email/password, SSO, or OAuth.
- **Role-based Access Control (RBAC):** Defining user roles such as Employee or Manager.
- **Permission management:** Granting permissions to perform specific actions (e.g., create tasks, modify tasks).

#### **f. Notification System**

- **Email notifications:** Notifying users of task updates or upcoming deadlines.

#### **g. Analytics and Reporting**

This component provides insights into task progress, user performance, and team productivity. Dashboards and reports can show:

- Number of tasks completed.
- Progress over time.

### **3. Task Management System Workflow**

Here is a simplified flow of how the task management system works:

1. **Task Creation:** A user (Admin/Manager) creates a task with required details such as title, description, due date, and priority.
2. **Task Assignment:** The task is assigned to one or more team members.
3. **Task Completion:** Once the task is finished, it is marked as "Completed".
4. **Analytics:** Managers can access real-time analytics on team performance and task status

## 1.6 Detail Description of Technology Used

### Django Frame-Work

**Django** is a high-level, open-source web framework for building robust and scalable web applications. It is written in Python and follows the **Model-View-Template (MVT)** architectural pattern, which is similar to the popular **Model-View-Controller (MVC)** pattern. Django was created to make web development easier and faster by providing a clean, pragmatic design and a set of tools that simplify common web development tasks.

Django is maintained by the Django Software Foundation (DSF), and its primary goal is to reduce the amount of code developers need to write while following best practices for security and scalability.

#### Advantages

Ease of Use and Rapid Development, Security Features , Built-in Admin Interface , Robust ORM (Object-Relational Mapping) , Clean and Readable Code

### HTML (Hyper Text Markup Language)

HTML (Hyper Text Markup Language) is the standard markup language used to create and structure content on the web. It defines the structure of web pages using a system of elements (also called tags) that describe different parts of the content, such as headings, paragraphs, images, links, tables, and more.

HTML is essential for building any web page, as it provides the foundational structure that browsers can interpret and display. HTML documents are composed of various elements enclosed in angle brackets (< >) like <html>, <head>, <body>, <p>, etc. HTML is often combined with other technologies like CSS (for styling) and JavaScript (for interactivity) to create modern web applications.

#### Advantage

Foundation of Web Development , Ease of Learning and Use Compatibility with All Browser, Platform Independence, Rich Media Integration, Responsiveness and Mobile-Friendliness, Accessibility ,Integration with Other Technologies.

### Python

**Python** is a high-level, interpreted programming language that is known for its simplicity, readability, and versatility. It was created by **Guido van Rossum** and first released in **1991**. Python emphasizes code readability and enables developers to write clean, maintainable code with fewer lines. It supports multiple programming paradigms, including procedural, object-oriented, and functional programming.

Python's syntax is designed to be intuitive and straightforward, making it an excellent choice for beginners and experienced programmers alike. It has an extensive standard library and a large ecosystem of third-party packages, which allows it to be used in a wide range of fields, from web development to data science and automation.

#### Advantages

Easy to Learn and Use, Versatility and Wide Range of Applications, **Large and Active Community** , Cross-Platform Development , Rich Ecosystem of Libraries and Frameworks , Rich Ecosystem of Libraries and Frameworks and High Productivity and Readability

#### JavaScript

**JavaScript** is a high-level, interpreted programming language primarily used to add interactivity, dynamic , and client-side functionality to websites. It is a core technology of the **web**, alongside **HTML** (for structure) and **CSS** (for styling). JavaScript is commonly used in combination with these technologies to create modern web applications that are responsive, interactive, and dynamic.

JavaScript can be run directly in the browser, making it a powerful tool for creating client-side scripts that respond to user input, modify the content of a web page, and handle events without requiring a page reload.

#### Advantages

Interactivity and Rich User Experience , Client-Side Scripting , Versatility (Full-Stack Development) , Large Ecosystem of Libraries and Frameworks , Cross-Browser Compatibility, Improved Performance and Speed and Real-Time Web Applications.

#### CSS

**CSS** (Cascading Style Sheets) is a styling language used to define the presentation of a web page written in **HTML** or **XML**. While HTML is used to structure the content of a webpage, CSS is used to control the layout, design, and look of the web elements such as fonts, colors, spacing, positioning, and responsive behavior. CSS allows web developers to separate the content (HTML) from its presentation (style), making it easier to maintain and update the look and feel of a website.

#### Advantages

Maintainability , Reusability , Improved Page Load Speed , Consistent Styling , Responsive Web Design , Rich Animations and Transitions , Cross-Browser Compatibility , Customizable and Scalable, Easy Integration with Other Technologies and Easy Integration with Other Technologies

## MySQL Workbench

**MySQL** is an open-source relational database management system (RDBMS) that uses Structured Query Language (SQL) to manage and manipulate data. It is one of the most popular database systems used for storing and retrieving data in web applications and software. MySQL is known for its speed, reliability, and flexibility, making it a top choice for many developers and businesses. It supports a wide variety of platforms, including Windows, Linux, and macOS, and is frequently used in conjunction with other technologies such as **PHP**, **Python**, **Java**, and **Node.js**.

MySQL uses a client-server architecture, where the MySQL server manages databases and client applications (such as web servers or desktop applications) interact with the server to execute queries and retrieve results. It supports standard SQL commands to manage databases, tables, and records.

## Advantages

Relational Database SQL Support, ACID Compliance (Atomicity, Consistency, Isolation, Durability), Cross-Platform, High Performance , Scalability ,Replication, Security Backup and Recovery ,Storage Engines

## **CHAPTER 2 : PROPOSED SYSTEM**

### **Proposed System**

The proposed of Task Management System in Python will offer an automated and efficient solution it will include the following features.

- **User-friendly Interface** : An intuitive and easy-to-use system of task management and report creation.
- **Tracking** : The interface can track the task if completed or not and can make an graphical representation of it.
- **Graph Representation** : The “Progress” feature shows column chart of the task which are completed of the month
- **Authentication** : The system has the basic authentication feature with user name , designation , password elements to the templates.
- **Mobile Compatibility**: The system should be responsive or have a mobile version for on-the-go task management.
- **Integrations**: If applicable, integration with other tools (e.g., email, calendar, project management software) to enhance functionality.

### **Objectives of System**

The main objective of the Task Management System are as follows:

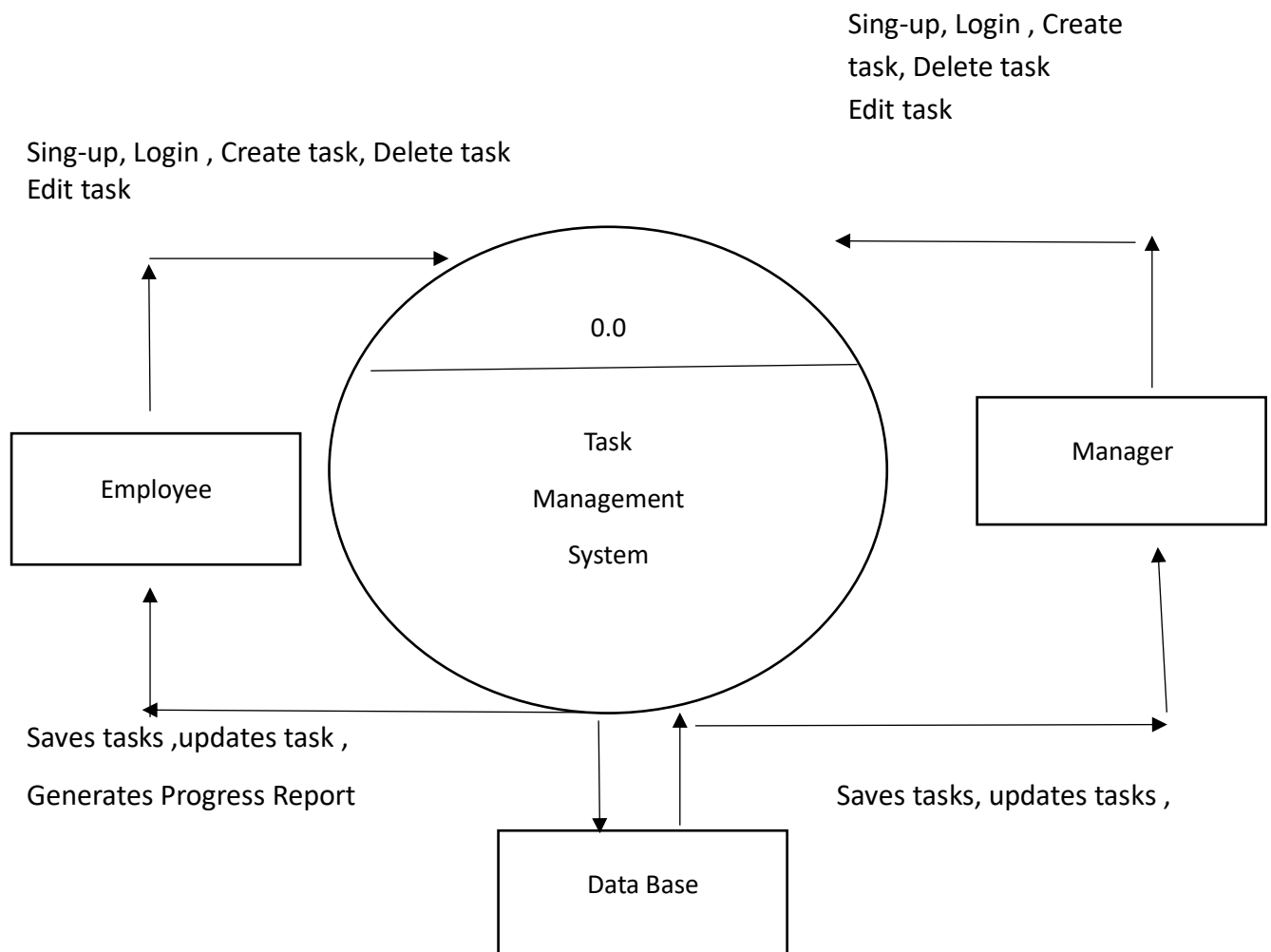
- **Efficiency** : Making task more efficient and easy to follow them to complete the goal makes this system more flexible for user
- **Accuracy** : Maintain the record of the users and their goal to the progress report
- **Improved Decision Making** : Provide valuable insights through report for the better decision-making.
- **Enhance Personal Growth** : As the user are the company’s employees and managers they both can analyse their performance for individual as well as team growth

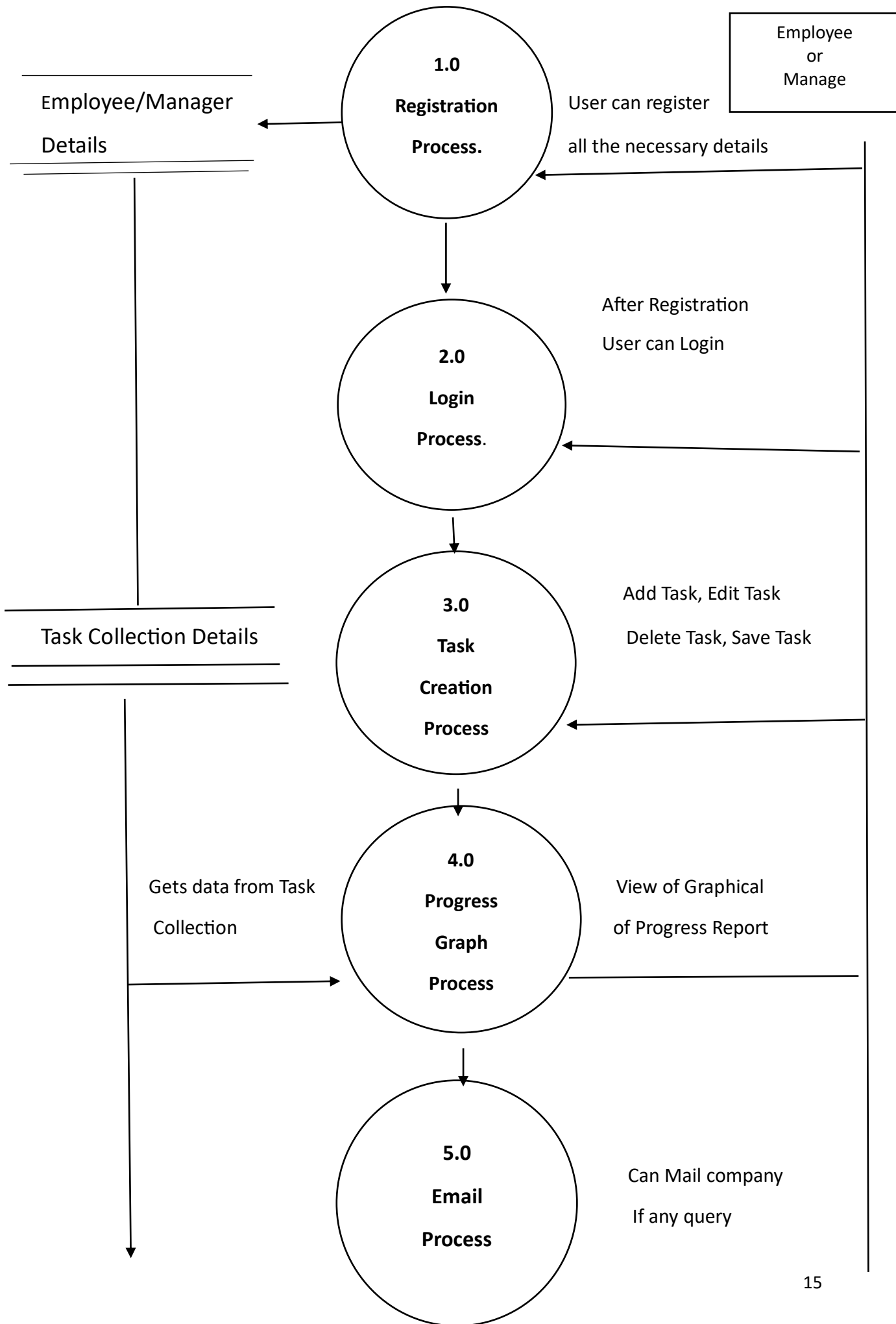
### **User Requirement**

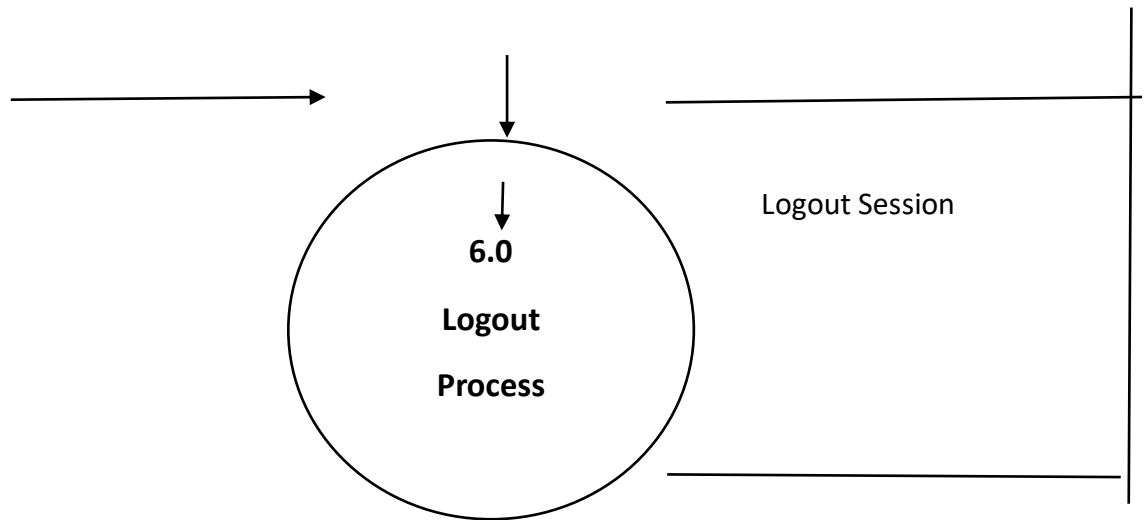
- **Login/Sign Up**: Users should be able to create an account and log into the system securely.
- **The User should be able to be** create account , create task, edit task , delete task , view progress of the task , mark complete or incomplete.
- **Task Overview**: The system should provide a clear overview of all tasks with key details like title, due date, status, and priority.
- **Fast Load Time**: The system should load quickly and handle large numbers of tasks without performance degradation.
- **Scalability**: The system should be scalable to accommodate a growing number of users, tasks, and teams as the organization expands.
- **Customer Support**: Users should have access to customer support (email) if they encounter issues or need assistance

## CHAPTER 3 : ANALYSIS & DESIGN

### 3.1 DFD







### 3.2 Table specifications (Database)

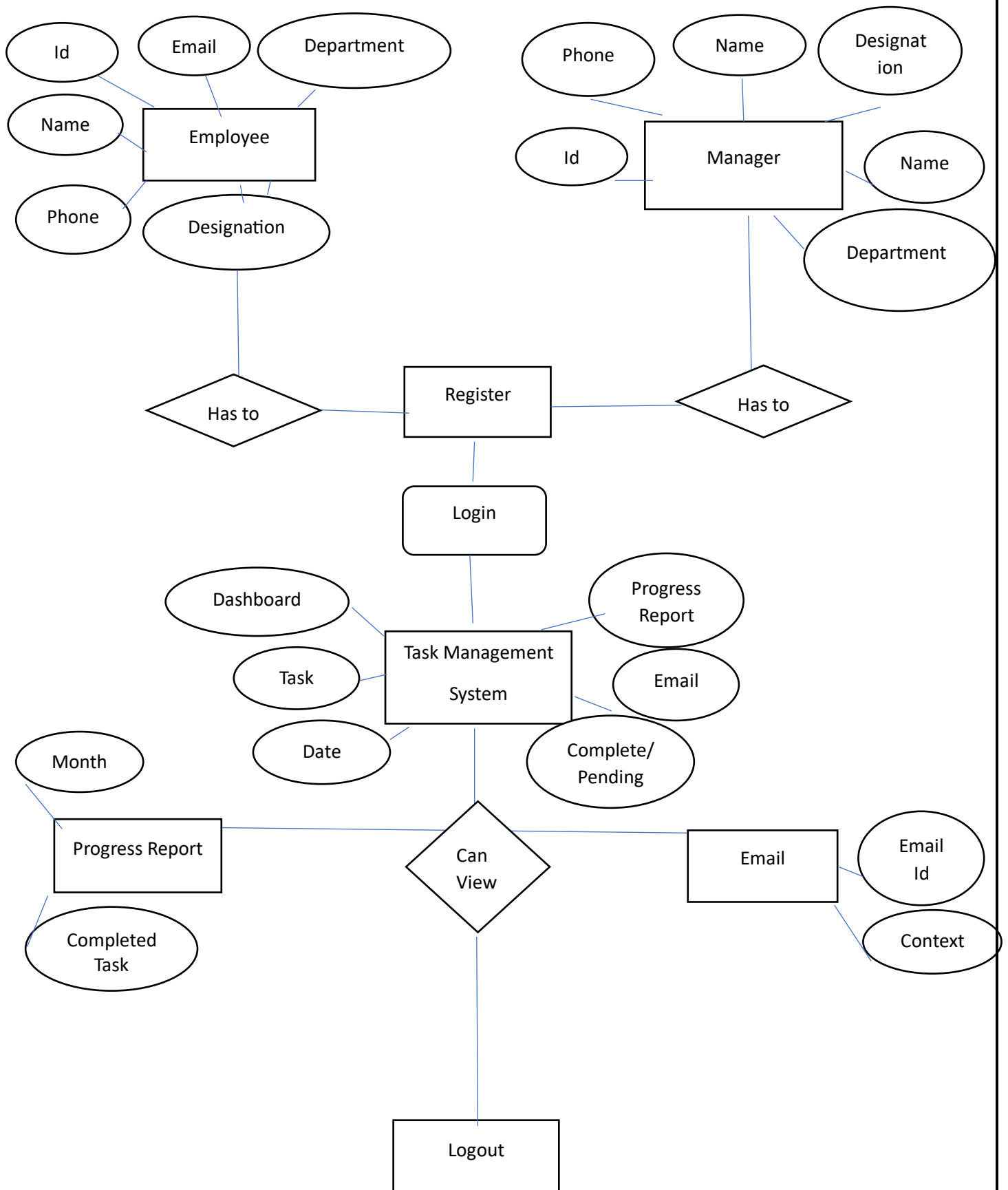
#### User (Employee/Manager)

<u>Table Name</u>	task management app user			
<u>Primary Key</u>	<u>ofid</u>			
<u>Description</u>	<u>Registration Details of the User</u>			
<u>Sr No.</u>	<u>Field Name</u>	<u>Data type</u>	<u>Constraint</u>	<u>Description</u>
<u>1</u>	<u>Id</u>	<u>int</u>	<u>Auto Increment</u>	<u>To store Id</u>
<u>2</u>	<u>nm</u>	<u>char</u>	<u>Not Null</u>	<u>To store Name</u>
<u>3</u>	<u>em</u>	<u>char</u>	<u>Not Null</u>	<u>To store Email</u>
<u>4</u>	<u>ph</u>	<u>Int</u>	<u>Not Null</u>	<u>To store Phone Number</u>
<u>5</u>	<u>dg</u>	<u>char</u>	<u>Not Null</u>	<u>To store Designation</u>

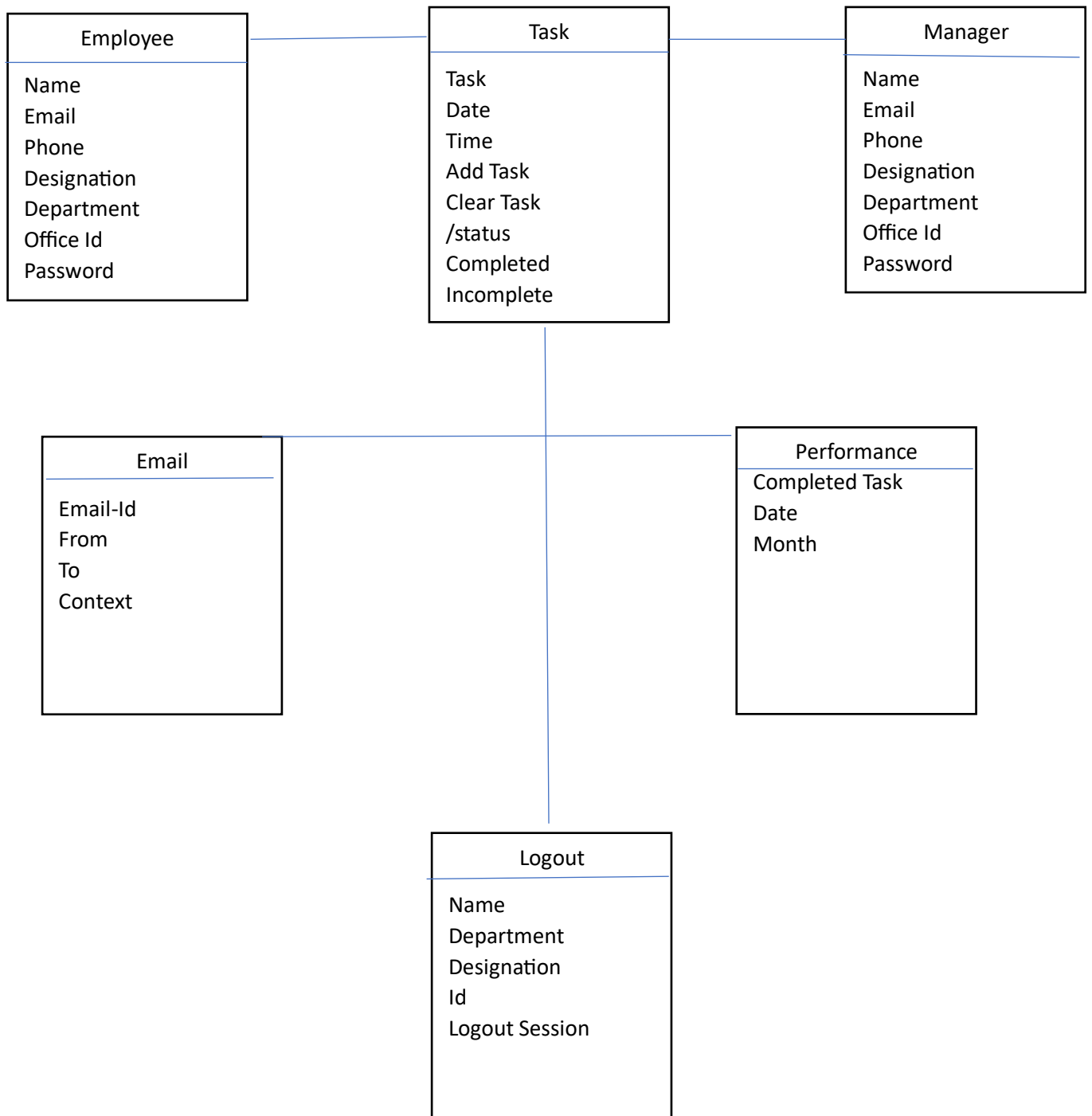


<u>6</u>	<u>ofid</u>	<u>int</u>	<u>(Primary Key)</u>	<u>Unique Office Id</u>
<u>7</u>	<u>Pas</u>	<u>int</u>	<u>Not Null</u>	<u>Password</u>

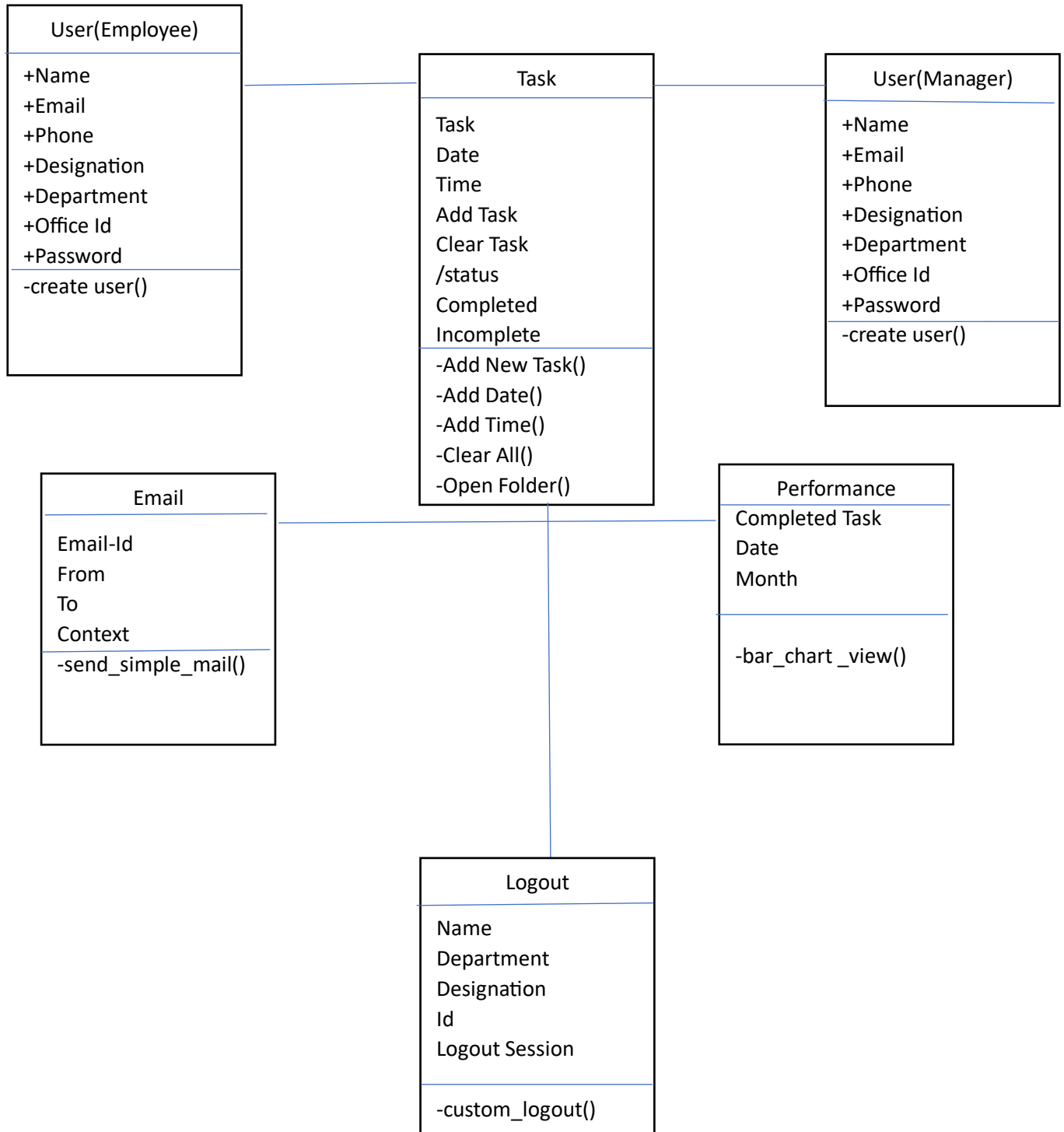
### 3.3 ERD (Entity Relationship Diagram)



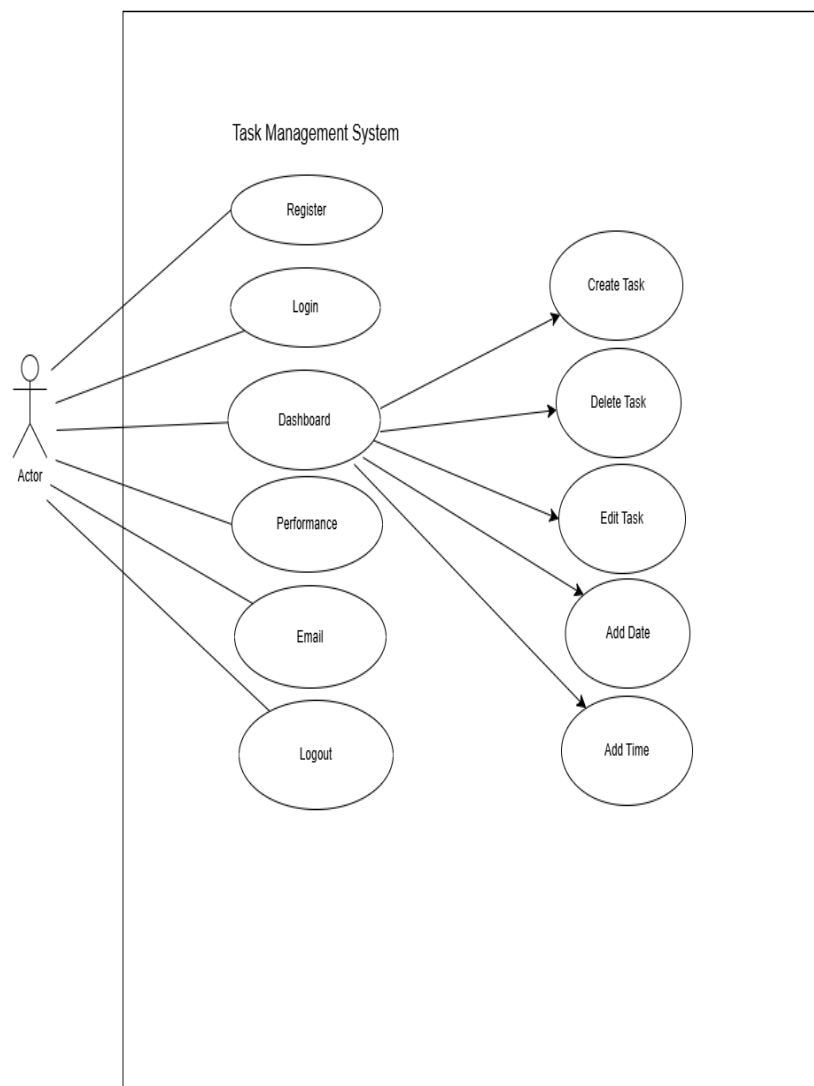
### 3.4 Object Diagram



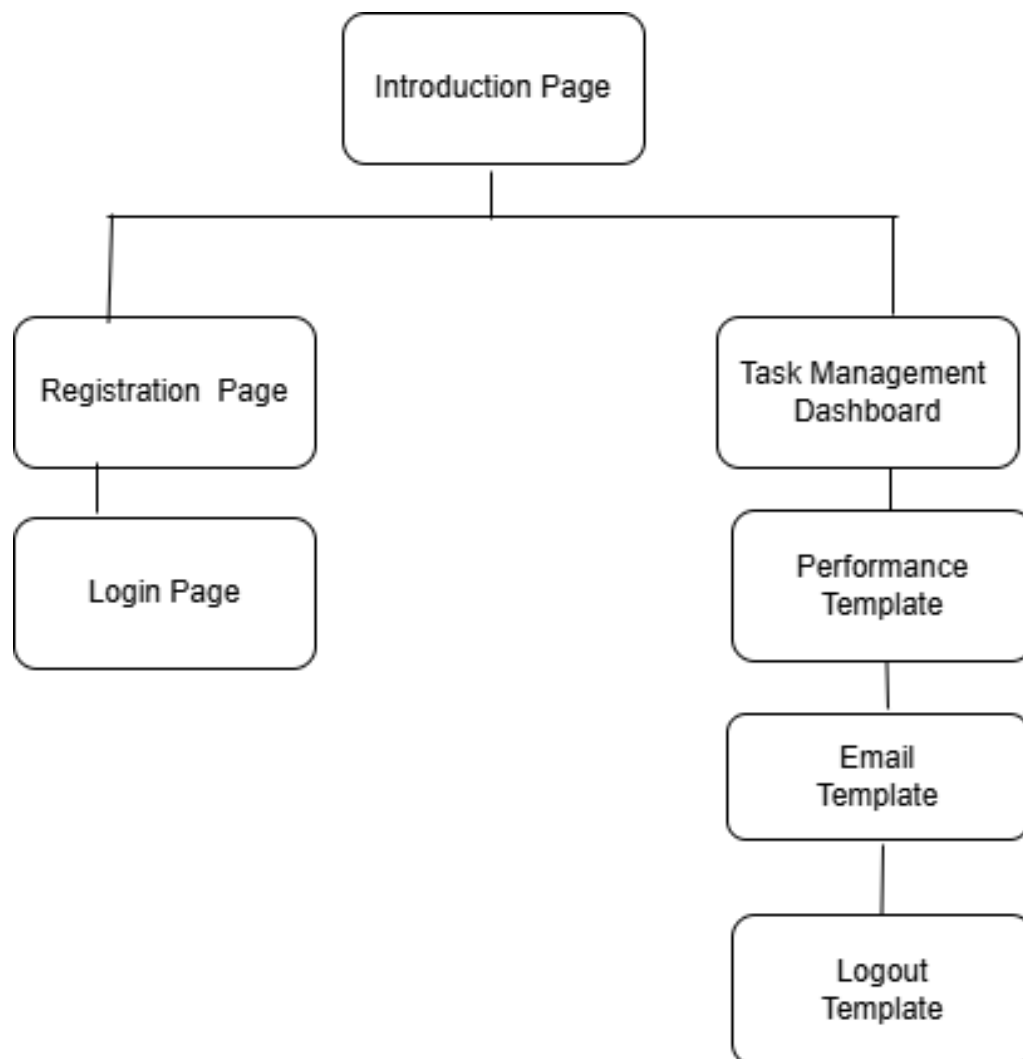
### 3.5 Class Diagram



### 3.6 Use Case Diagram



### 3.7 Web site Map Diagram



## Chapter 4 User Manul


### 4.1 User Interface Design



**Task Management System**

**Register**

Name

Email   Please fill out this field.

Phone

Designation ☐ Manager ☐ Employee

Department

Office Id

Password

Already Register? [Login!](#)

# Task Management System

## Login

Email

Chinu@gmail

Office Id

Password

! Please fill out this field.

Login



Hello!!!  
Chinu  
QA-MA[987]

Dashboard

Performance

Email

Logout

TASK	DATE	TIME
<input type="text" value="Enter task"/>	<input type="text" value="dd-mm-yyyy"/>	<input type="text" value="--:--"/>

ADD

Clear

Your Work :

TASK	DATE	TIME

Completed: 0 | Uncompleted: 0



## Email To Task Management Sytem Support Admin Team

To:

taskmanagementsystem40@gmail.com

Subject:

Technical Issue Resolve

Message:

Here is the demo message from  
the user for Technical trouble

Submit

Email Send Successfuly



### Technical Issue Resolve Inbox x



**taskmanagementsystem40@gmail.com**

to me ▼

Here is the demo message from the user for Technical trouble.

↩ Reply

➡ Forward



## 4.2 Limitation

Task management systems are powerful tools for organizing and managing projects, teams, and personal tasks. However, they also have certain limitations that users should be aware of. Here are some common limitations:

### 1. Complexity for Small Teams or Simple Tasks

- **Overhead for Simple Projects:** Task management systems can be too complex for small teams or simple tasks, as they often come with a steep learning curve and many features that are unnecessary for basic needs.
- **Too Much Functionality:** For individual users or small teams, the vast array of features can become overwhelming and may slow down productivity rather than enhance it.

### 2. Lack of Flexibility

- **Rigid Structures:** Some task management systems enforce specific workflows or structures that may not fit every team or project. This rigidity can make it difficult to adapt the system to unique needs or processes.
- **Customization Limitations:** Although many systems allow some customization, there may still be limitations in adjusting features or user interfaces to suit the team's preferences.

### 3. Dependence on Technology

- **System Downtime:** Task management systems are typically cloud-based, which means they rely on internet access and servers. Any downtime or service outages can disrupt work and cause delays.
- **Learning Curve:** Transitioning to a new system or tool can take time, and employees may resist adopting a new system, especially if they are accustomed to a different way of working.

### 4. Over-Tracking and Micromanagement

- **Micromanagement Risk:** Task management systems may lead to micromanagement if managers use them to constantly monitor every detail, leading to frustration and a lack of autonomy among team members.
- **Too Much Data:** Tracking every task in minute detail can be overwhelming and counterproductive, especially when it leads to over-analysis or constant updates without tangible results.

### 5. Limited Collaboration Features

- **Communication Gaps:** While task management tools allow task assignment and tracking, they often don't provide sufficient communication features (like chat or video calls) that are essential for collaboration.

- **Collaboration Barriers:** Some systems do not integrate well with other tools like email or communication platforms, creating silos or barriers to efficient collaboration.

## 6. Scalability Challenges

- **Managing Large Projects:** As the scope of a project grows, task management systems can become cluttered, making it difficult to track tasks effectively and maintain an overview of the project's progress.
- **Team Size:** In larger teams, task delegation and coordination can become complex, leading to confusion and difficulty in maintaining accountability.

## 7. Limited Integration with Other Tools

- **Poor Integration:** Some task management systems lack robust integration with other tools, such as CRM, email, or calendar apps, leading to inefficiencies and the need for manual data transfer between platforms.
- **Synchronization Issues:** Data synchronization can be problematic across different devices, and inconsistencies between systems can lead to lost information or missed deadlines.

## 8. Privacy and Security Concerns

- **Data Security Risks:** Storing sensitive task and project data in the cloud introduces privacy risks. Some organizations may prefer to keep certain information offline or within their private systems.
- **User Access Control:** Some systems offer inadequate control over user permissions, potentially exposing sensitive information to people who shouldn't have access.

## 9. Lack of Context

- **Missing Context in Tasks:** Task management systems often focus on the task itself, without providing much context about the larger goals, the team's strategic direction, or the reasons behind certain tasks, which can reduce motivation and clarity.
- **No Visual Overview:** Some systems lack features like Gantt charts or visual roadmaps, which makes it hard to see how tasks relate to the overall project timeline.

## 10. Task Dependency and Workflow Management

- **Task Dependencies:** Managing complex tasks with dependencies can become cumbersome in some task management systems, as not all tools provide clear visualizations for how tasks interrelate.

- **Ineffective Workflow Automation:** While some systems offer automation features, they may not be advanced enough to handle complex workflows, which can lead to additional manual effort.

## 4.3 Future Enhancement

### 1. Artificial Intelligence (AI) and Automation

- **Smart Task Prioritization:** AI will help automatically prioritize tasks based on deadlines, dependencies, and importance. It will learn from past behavior and patterns, making recommendations for optimal task management.
- **Automated Task Assignment:** AI could automatically assign tasks to the best-suited team members based on workload, skill set, and availability, optimizing team productivity.
- **Predictive Analytics:** AI could predict delays, identify potential roadblocks, or suggest adjustments to timelines based on past performance and real-time data.
- **Intelligent Task Scheduling:** Task systems will use machine learning to adjust schedules dynamically, accommodating changes in priority, team availability, or external factors in real time.

### 2. Advanced Collaboration Features

- **Integrated Communication Tools:** Future task management systems could fully integrate with communication platforms like chat, video calls, and email, allowing users to discuss tasks, share updates, and make decisions directly within the system.
- **Real-Time Collaboration:** Seamless, real-time collaboration on tasks will become more intuitive. Team members could work on documents or presentations within the system, with live updates and the ability to resolve conflicts in real time.
- **Collaboration on Task Dependencies:** Teams could visualize and collaborate on tasks that are dependent on each other, with tools to easily manage changes in shared tasks and deadlines.

### 3. Better Integration with Other Tools

- **Cross-Platform Integration:** Task management systems will likely offer deeper integrations with other popular tools (like CRMs, ERPs, financial tools, or email clients), creating a unified workflow across platforms.
- **Seamless API Integrations:** Open API systems will allow for better integration with other business tools, enabling companies to customize their workflows and automate processes across different platforms without manual intervention.
- **Calendar & Time Management Integration:** Task management systems will integrate more deeply with calendar tools (Google Calendar, Outlook) to allow for easier scheduling of tasks, automatic updates, and reminders.

#### 4. Enhanced User Experience (UX)

- **Adaptive Interfaces:** Future systems could feature adaptive, AI-driven user interfaces that adjust to the user's preferences, workflow, and role. For instance, project managers might see more advanced dashboards, while individual contributors could have simplified interfaces.
- **Voice-Activated Commands:** Voice-based task management could become commonplace, enabling users to add tasks, set reminders, or check status updates simply by speaking, improving hands-free productivity.
- **Natural Language Processing (NLP):** NLP will allow users to input tasks and deadlines more naturally using conversational language, rather than requiring strict formatting or structured input.

#### 5. Mobile and Remote Work Enhancements

- **Enhanced Mobile Experience:** As remote work continues to rise, task management tools will optimize for mobile devices, offering full functionality and ease of use for on-the-go task management.
- **Offline Task Management:** Improved offline capabilities will allow users to manage tasks, update progress, and access key information without requiring an internet connection, syncing once back online.
- **Geolocation and Time Zone Support:** For global teams, future systems will offer better support for managing tasks across time zones, incorporating geolocation features to better schedule tasks based on team members' locations and work hours.

#### 6. Task Visualization and Reporting Tools

- **Interactive Dashboards:** More interactive and customizable dashboards will allow users to visualize tasks, progress, timelines, and dependencies in a way that suits their specific needs (e.g., Gantt charts, Kanban boards, and burndown charts).
- **Real-Time Project Tracking:** Real-time tracking of project and task statuses, with automated reports that update dynamically, will help project managers stay on top of team progress.
- **Advanced Data Analytics:** Integration with analytics tools will allow users to track productivity trends, identify bottlenecks, and measure performance over time, providing actionable insights for better decision-making.

#### 7. Improved Security and Privacy

- **Advanced Encryption:** As data privacy becomes increasingly important, task management systems will incorporate more robust encryption protocols, ensuring that sensitive information remains secure.

- **Granular Access Controls:** Better user permission controls will allow managers to assign and restrict access at the task or project level, ensuring that only the relevant team members can view, edit, or comment on tasks.
- **Data Compliance:** Future systems will focus on meeting global data privacy regulations (e.g., GDPR, CCPA), ensuring organizations remain compliant with data protection laws.
- .

## 8 . Cross-Team Collaboration and Global Workflows

- **Cross-Departmental Collaboration:** Task management systems will allow seamless collaboration between different teams or departments, offering visibility and shared responsibility for interdependent tasks across organizations.
- **Multilingual Support:** With the rise of global teams, task management systems will provide better multilingual support, enabling seamless communication and task tracking in multiple languages.

## ANNEXURE: Sample program code

Show.html

```
{% load static %}

<!DOCTYPE html>

<head>

    <link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css">

    <link rel="stylesheet" href="{% static 'paint.css' %}">

    <script src="{% static 'logic.js' %}"></script>

</head>

<body>

    <!--

    <div class="sidenav">

        <center></center>

        <h2><center>Hello!!!<br>{{request.session.Name}} <br>
{{request.session.Department}}-
{{request.session.Desgination}}[{{request.session.Officeld}}]</cente
r></h2>

        <a href="#about">Dashboard</a>

        <br><br>

        <a href="#services">Performance</a>
```

```

<br><br>
<a href="#clients">Email</a>
<br><br>
<a href="#contact">Logout</a>
</div>
<center>
<table style="width:50%">
  <tr>
    <th>TASK</th>
    <th>DATE</th>
    <th>TIME</th>
  </tr>
  <tr >
    <td><input type="text" placeholder="Enter task"
id="task"> <span><span> <class="btn" id="folder"
onclick="OpenFolder()"> <i class="fa fa-folder">
</i></span></span></td>
    <td><input type="date" id="date"></td>
    <td><input type="time" id="time"></td>
  </tr>
</table>
<!--This button takes three function to add task,date,time-->
<br>

```



```
<button id="Add" onclick
="AddnewTask();AddDate();AddTime()">ADD</button>

<button id="Clear" onclick="ClearAll()"> Clear </button>
```

```
<h3>Your Work : </h3>
```

```
<table id="Allocations">
```

```
<tr>
```

```
<th>TASK</th>
```

```
<th>DATE</th>
```

```
<th>TIME</th>
```

```
</tr>
```

```
<!--Data Coloum and Un-ordered List for Tasks with ID -->
```

```
<td>
```

```
<ul id="myUL-Task">
```

```
</ul>
```

```
</td>
```

```
<!--Data Coloum and Un-ordered List for Tasks with ID -->
```

```
<td>
```

```
<ul id="myUL-Date">
```

```
</ul>
```

```
</td>
```

```
<td>
```

```
<ul id="myUL-Time">
```

```
</ul>
```

```
</td>
```

```
</table>
```

```
<div class="counter-container">
```

```
  <div id="task-counters">
```

```
    Completed: <span id="completed-counter">0</span> |
```

```
    Uncompleted:
```

```
      <span id="uncompleted-counter">0</span>
```

```
    </div>
```

```
  </div>
```

```
</center>
```

```
</head>
```

```
</body>
```

```
</html>
```

## Views .py

```
from django.shortcuts import render

from .models import User


# Create your views here.

def IndexView(request):

    return render(request,"Task_Management_App/index.html") #Index.html View

#register html page view

def RegisterView(request):

    return render(request,"Task_Management_App/register.html") #Register.html View

#insert user view

def insertuser(request):

    if request.method=="POST":

        vnm=request.POST['tname']

        vem=request.POST['temail']

        vph=request.POST['tphone']

        vdeg=request.POST['tdesignation']

        vdep=request.POST['tdepartment']

        vid=request.POST['tid']

        vpas=request.POST['tpass']

        us=User(nm=vnm,em=vem,ph=vph,dg=vdeg,dp=vdep,ofid=vid,pas=vpas)

        us.save()

        return render(request, 'Task_Management_App/register.html', {'msg': 'User created successfully!'})

    else:

        return render(request,'Task_Management_App/register.html',{})
```

**#login html view**

**def LoginView(request):**

**return render(request,"Task\_Management\_App/login.html")**

**def ShowView(request):**

**return render(request,"Task\_Management\_App/show.html")**

**#login function view**

**def loginuser(request):**

**if request.method=="POST":**

**email=request.POST['temail']**

**password=request.POST['tpass']**

**user=User.objects.get(em=email)**

**if user:**

**if user.pas == password:**

**request.session['Name']=user.nm**

**request.session['Desgination']=user.dg**

**request.session['Department']=user.dp**

**request.session['OfficeId']=user.ofid**

**return render(request,"Task\_Management\_App/show.html")**

**# return render(request,"Task\_Management\_App/login.html",{ 'msg':message})**

**else:**

**message="User does not exist"**

**return render(request,"Task\_Management\_App/login.html",{ 'msg':message})**

## **Bibliography**

I have been successfully completed by project through the following resources  
For enhancing my knowledge on Django Framework.

Links :

<https://www.learnvern.com/python-tutorial-django>

<https://docs.djangoproject.com/en/5.1/>

From Books :

Django Frame Works

Restful Django API