SOFTWARE PROJECT FINAL REPORT

Priya Bhavana

Anuradha

Sai Srinivas

12/06/2023

Table of Contents

Contents

List of Figures

List of Tables

1. Introduction

1.1. Purpose and Scope

This application is a Task and project management tool designed to streamline tasks, projects, user interactions, and document management. The application will have various functionalities, allowing users to register, create, and track projects, manage tasks, and handle file attachments.

The purpose of this document is to provide a comprehensive understanding of the requirements and specifications for developing a web-based Task and Management Application.

The scope of this application encompasses:

Streamlining project management

Task Management processes.

Improving overall team productivity and communication.

1.2. Product Overview (including capabilities, scenarios for using the product, etc.)

User Registration:

Users will be required to provide essential information such as first name, last name, email, UserID, and password to create accounts.

Project Management Module:

Project creation with details such as name, description, start date, and end date.

Project archiving for completed projects.

Task Management:

Task creation within projects.

Progress tracking and deadline management.

Collaboration Tools:

Task comments for discussions.

File attachments for resource sharing.

Scenarios for Using the Product:

Scenario 1: Project Creation

A manager logs in, creates a new project, and defines its parameters.

Scenario 2: Task Management

A team leader assigns tasks to team members, monitors progress, and updates task statuses.

Scenario 3: Collaboration

Team members utilize task comments and file attachments for effective communication.

1.3. Structure of the Document

1.4. Terms, Acronyms, and A**bbreviations**

2. Project Management Plan

2.1. Project Organization

Team Members:
Priya Bhavana Rajampalli
Anuradha
Sai Srinivas

a) **Priya Bhavana Rajampalli**

Role: Involved in developing, planning, and documentation.

Responsibilities:

Actively participate in the development process, including coding, designing, and implementing features.

Contribute to the overall planning and strategy for the project.

Collaborate with team members to ensure that development efforts align with project goals.

Lead and contribute to the documentation process, including requirements gathering, planning documents, and any other relevant project documentation.

b)Anuradha:

Role: Involved in documentation and testing.

Responsibilities:

Work closely with documentation, including gathering requirements and planning documents.

Create and maintain project documentation, including design specifications and user manuals.

Conduct testing activities to ensure the quality and reliability of the developed software.

c)Srinivas:

Role: Involved in the database and server connections.

Responsibilities:

Design, implement, and maintain the database architecture for the project.

Establish and manage server connections to ensure smooth communication between the application and the database.

2.2. Lifecycle Model Used

Process Model Selection: **Waterfall Model**

Activities:

Project Phases:

Requirements Analysis

System Design

Implementation

Testing

Deployment

Maintenance

Activities in Each Phase:

Introduction:

Project Plan :

A brief overview of the project and its objectives.

Software Requirements Specification (SRS):

Documented user requirements and project specifications.

Software Design Specification (SDS):

Detailed system architecture, wireframes, and database design.

Initial Software Version Demo:

Showcased core functionalities of the initial software.

Test Plan:

Outlined testing strategy, including unit testing and user acceptance testing.

Project Deliverables:

Emphasized the compilation of all project documents and artifacts.

2.3. Risk Analysis

Table 1

Risk Name	Probability	Impact	RM3 Pointer
Resource Availability	High	High	12
Technology Dependencies	Moderate	High	9
User Adoption	Moderate	Moderate	6
Incomplete Task Creation	Moderate	High	6
Real-time Collaboration Failure	Moderate	High	6
Missed Deadlines	Low	High	4

2.4. Hardware and Software Resource Requirements

Frameworks and Technologies:

Frontend: HTML, CSS, JavaScript

Backend: Python- Flask

Database: SQLite

Version control: GitHub

2.5 Deliverables and Schedule

Table 2

Task	Start Date	End Date
Project Plan	9/15/2023	9/21/2023
Spec (SRS)	10/20/2023	10/5/2023
Software Design Spec (SDS)	10/04/2023	10/19/2023
Initial Software Version Demo	11/18/2023	11/2/2023
Test Plan	11/01/2023	11/9/2023
Project Deliverables	12/08/2023	12/7/2023
Project Presentation & Demo	12/7/2023	12/7/2023

3. Requirement Specifications

3.1 Project Task Set

Table 3

Task	Start Date	End Date	Description
Project Plan	9/15/2023	9/21/2023	Define project scope, objectives, and constraints. Identify stakeholders and establish communication.
Spec (SRS)	10/20/2023	10/5/2023	Conduct interviews and workshops to gather requirements. Document the Software Requirements Specification (SRS).
Software Design Spec (SDS)	10/04/2023	10/19/2023	Create a system architecture. Develop a detailed Software Design Specification (SDS) based on the architecture.
Initial Software	11/18/2023	11/2/2023	Develop an initial version of the software

Task	Start Date	End Date	Description
Version Demo			for demonstration purposes.
Test Plan	11/01/2023	11/9/2023	Develop a comprehensive plan outlining the testing approach, resources, and schedule.
Project Deliverables	12/08/2023	12/7/2023	Prepare and deliver the final project deliverables according to specifications.
Project Presentation & Demo	12/7/2023	12/7/2023	Conduct a presentation and demo showcasing the completed project.

3.2. Use cases

Use Case 1: User Registration

Use Case 2: User Profile Modification

Use Case 3: Account Deletion

User Role: Registered User

Use Case 4: Project Creation

Use Case 5: Tracking Project

Use Case 6: Task Creation

Use Case 7: Tracking Task

Use Case 8: Project Archiving

User Role: Manager

3.2.1. Graphic use case model

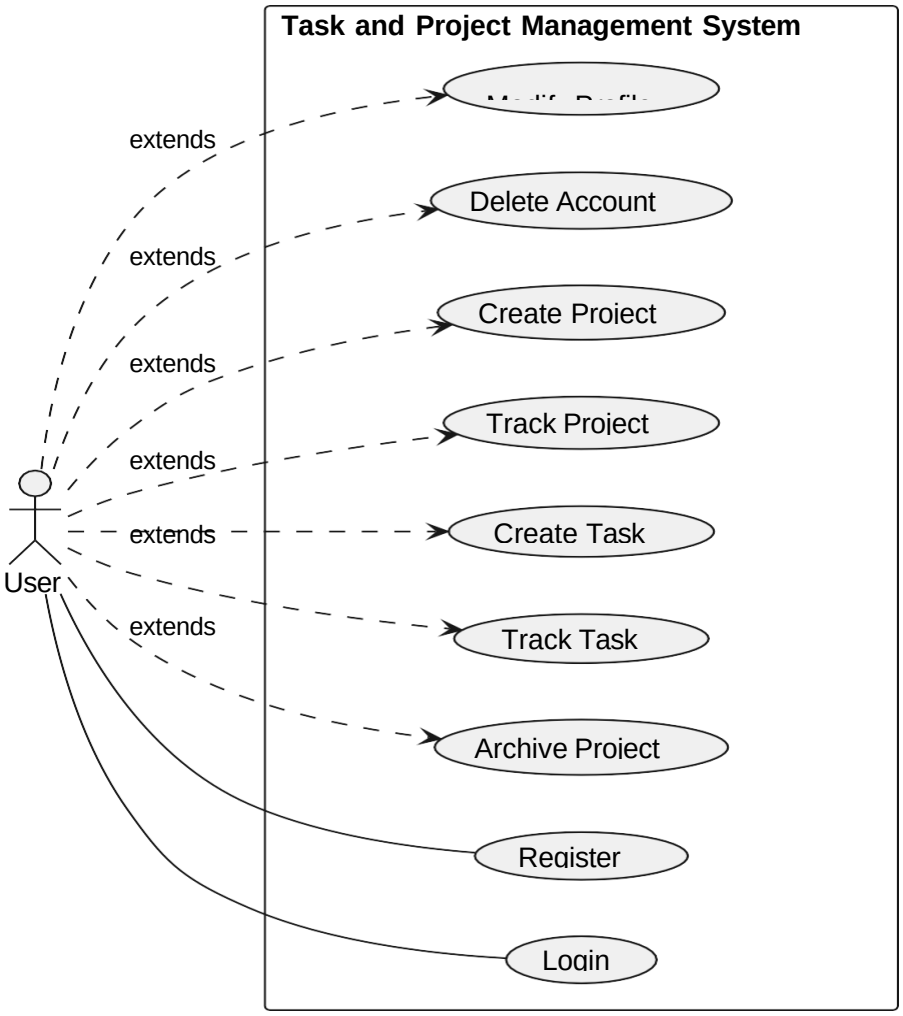


Figure 1

3.2.2. **Textual Description for each use case**

Use Case 1: User Registration

User Role: New User

Description: This use case describes the process of a new user registering an account in the system.

Flow:

1. The user accesses the registration page.

2. The user provides their first name, last name, email address, unique UserID, and a secure password.

3. The system validates the provided information.

4. If the information is valid, the system creates a new user account.

5. The user is now registered and can log in to the system.

Alternate Flows:

If the email address or UserID is already in use, the system displays an error message.

If the password does not meet the security requirements, the system displays an error message.

Use Case 2: User Profile Modification

User Role: Registered User

Description: This use case describes how a registered user can modify their profile information.

Basic Flow:

1. The user logs in to their account using their UserID and password.

2. The user accesses the profile modification page.

3. The user updates their first name, last name, and email address.

4. The system validates the changes made.

5. If the changes are valid, the system updates the user's profile information.

6. The user's profile information is now modified.

Alternate Flows:

If the provided UserID or password is incorrect, the system displays an error message.

If the email address is updated to one that already exists in the system, the system displays an error

message.

Use Case 3: Account Deletion

User Role: Registered User

Description: This use case describes how a registered user can delete their account.

Basic Flow:

1. The user logs in to their account using their UserID and password.

2. The user accesses the account deletion page.

3. The user provides their UserID, email address, and password.

4. The system validates the provided information.

5. If the information is valid, the system deletes the user's account.

6. The user's account is now deleted, and they are logged out.

Alternate Flows:

If the provided UserID, email address, or password is incorrect, the system displays an error message.

Use Case 4: Project Creation

User Role: Manager

Description: This use case describes how a manager can create a new project.

Basic Flow:

1. The manager logs in to their account using their UserID and password.

2. The manager accesses the project creation page.

3. The manager provides the project name, description, start date, and end date.

4. The system validates the provided information.

5. If the information is valid, the system creates a new project.

6. The new project is now created and visible in the system.

Alternate Flows:

If the provided project name already exists in the system, the system displays an error message.

If the start date or end date is in an invalid format, the system displays an error message.

Use Case 5: Tracking Project

User Role: Manager, Team Member, Team Lead

Description: This use case describes how a user can track the progress of a project.

Basic Flow:

1. The user logs in to their account using their UserID and password.

2. The user accesses the project tracking page.

3. The user provides the ProjectID.

4. The system validates the provided ProjectID.

5. If the ProjectID is valid, the system displays project details and progress.

Alternate Flows:

If the provided ProjectID is incorrect or does not exist, the system displays an error message.

Use Case 6: Task Creation

User Role: Manager, Team Leader

Description: This use case describes how a user can create a new task within a project.

Basic Flow:

1. The user logs in to their account using their UserID and password.

2. The user accesses the task creation page within a specific project.

3. The user provides the task title, description, assignee, and deadline.

4. The system validates the provided information.

5. If the information is valid, the system creates a new task within the project.

6. The new task is now created and associated with the project.

Alternate Flows:

If the provided deadline is in an invalid format, the system displays an error message.

Use Case 7: Tracking Task

User Role: Manager, Team Member, Team Leader

Description: This use case describes how a user can track the tasks assigned to them and monitor task

progress.

Basic Flow:

1. The user logs in to their account using their UserID and password.

2. The user accesses the task tracking page.

3. The user provides the TaskID.

4. The system validates the provided TaskID.

5. If the TaskID is valid, the system displays task details and progress.

Alternate Flows:

If the provided TaskID is incorrect or does not exist, the system displays an error message.

Use Case 8: Project Archiving

User Role: Manager

Description: This use case describes how a manager can archive a project that is completed.

Basic Flow:

1. The manager logs in to their account using their UserID and password.

2. The manager accesses the project archiving page.

3. The manager provides the ProjectID of the completed project.

4. The system validates the provided ProjectID.

5. If the ProjectID is valid, the system archives the project.

Alternate Flow:

If the provided ProjectID is incorrect or does not correspond to a completed project, the system displays an error messages

3.4. Non-functional requirements

Performance:

Response Time: This application will respond to user actions within 3 seconds for optimal user experience.

Usability:

User Interface Consistency: The user interface should have a consistent design and layout across all pages for ease of use.

Compatibility:

Browser Compatibility: This application is compatible with popular browsers such as Chrome, Firefox, and Safari.

4. Architecture

4.1. Architectural style(s) used

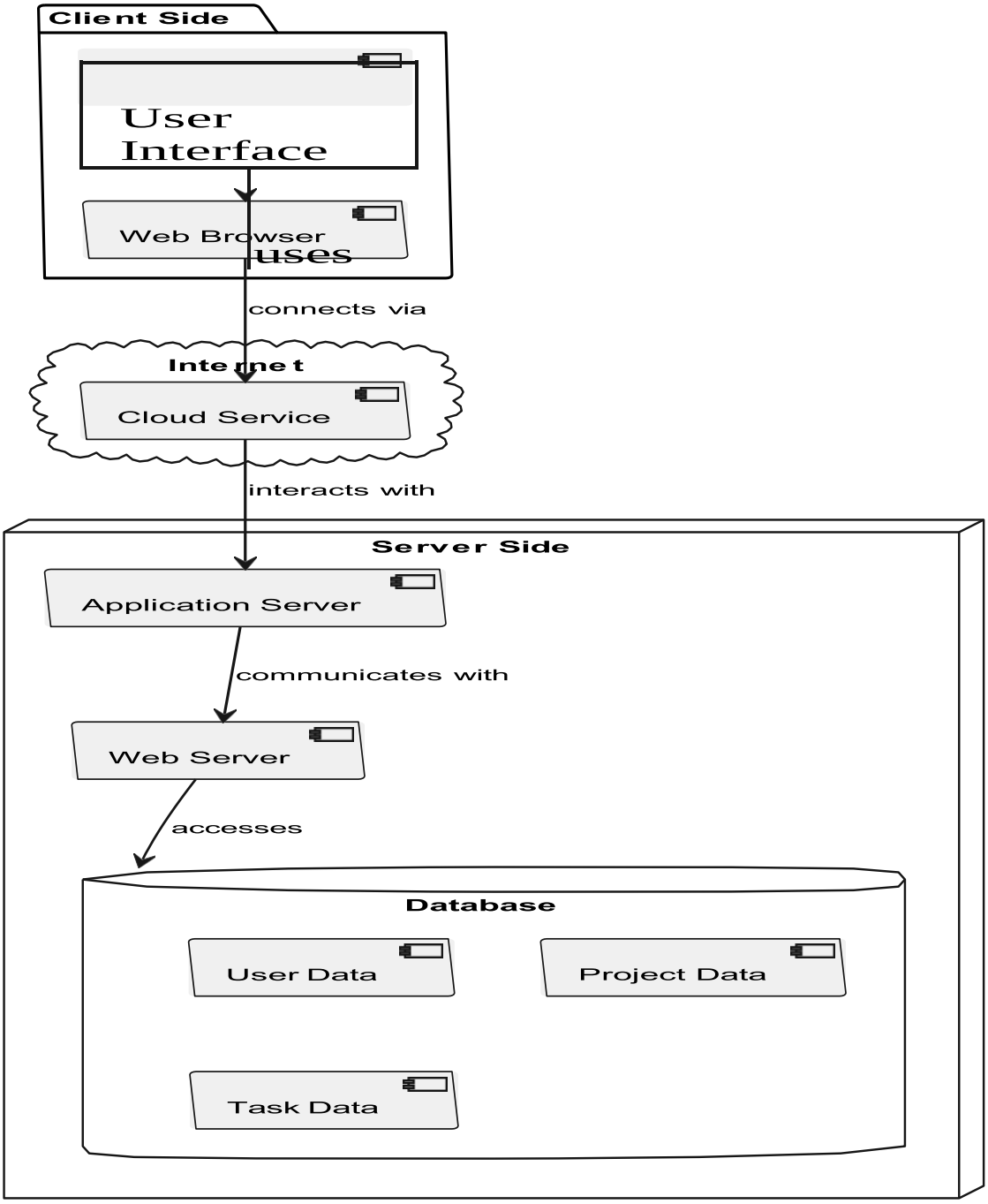


Figure 2

Communication: Users can create comments**.**

File Management: Allows users to upload attachments and subsequently update them.

Task Management: Users can create new tasks and track their progress or status.

Project Management: Facilitates the creation of new projects, monitors their progress, and offers archiving options.

User Management: New users can register, existing profiles can be edited, and accounts can be deleted if needed.

4.2. Architectural model (includes components and their interactions)

Roles & System: Different user roles (Manager, Team Leader, Team Member) interact with various system pages for project and task management.

Users: New and registered users navigate through pages for registration, profile modification, and account deletion.

Projects & Tasks: This provides distinct pages for creating, tracking, and archiving projects, as well as managing individual tasks.

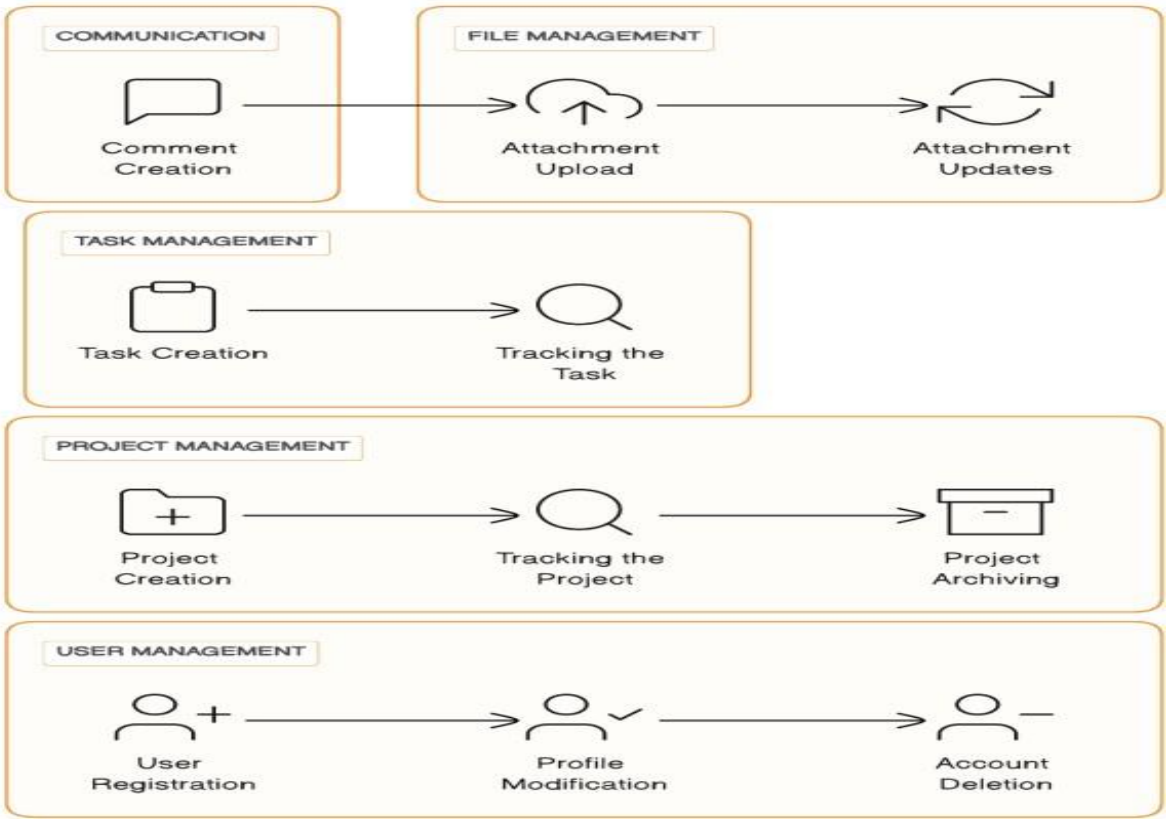

COMMUNICATION
Comment Creation
FILE MANAGEMENT
Attachment Upload
Attachment Updates
TASK MANAGEMENT
Task Creation
Tracking the Task
PROJECT MANAGEMENT
Project Creation
Tracking the Project
Project Archiving
USER MANAGEMENT
User Registration
Profile Modification
Account Deletion

Figure 3

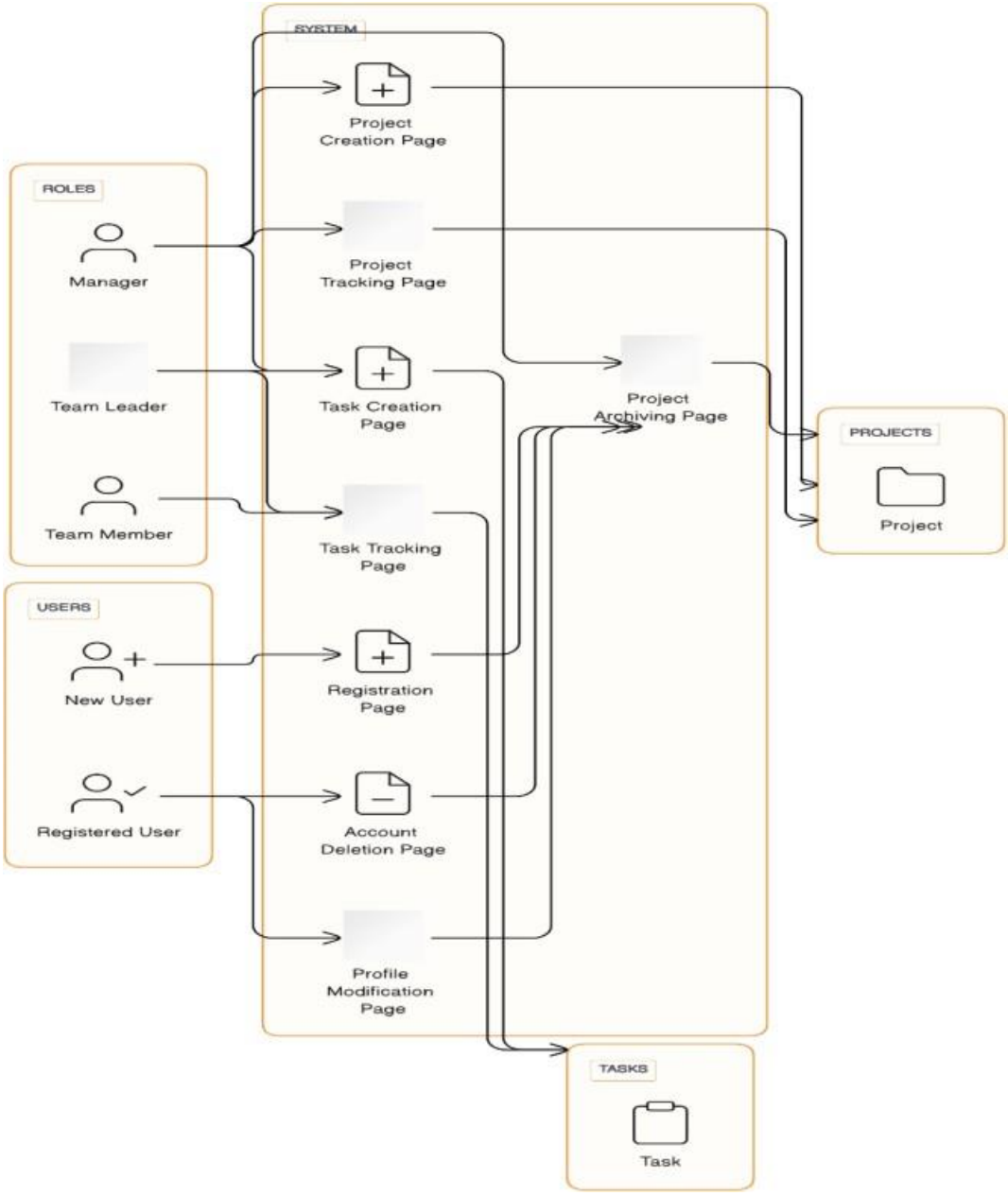

SYSTEM
Project Creation Page
ROLES
Manager
Project Tracking Page
Team Leader
Task Creation Page
Project Archiving Page
PROJECTS
Project
Team Member
Task Tracking Page
USERS
New User
Registration Page
Registered User
Account Deletion Page
Profile Modification Page
TASKS
Task

Figure 4

4.3. Technology, software, and hardware used

Frontend: HTML, CSS, JavaScript

Backend: PHP

Database: MySQL

Version control: Git/GitHub

Real-time functionality: WebSocket

4.4. Rationale for your architectural style and model

5. Design

5.1. User Interface Design

In our Task and Project Management System, the User Interface Design is meticulously crafted to ensure a seamless and intuitive user experience. Emphasizing consistency, the interface maintains a uniform design and layout across all pages, enhancing ease of use and navigability. To cater to a wide range of users, the application is engineered for compatibility with major web browsers, including Chrome, Firefox, and Safari.

The frontend development leverages the robust capabilities of HTML, CSS, and JavaScript, ensuring a responsive and dynamic interface. This approach not only facilitates efficient user interactions but also aligns with modern web standards.

Our focus on user interface consistency and browser compatibility underscores our commitment to delivering a high-quality, accessible, and user-friendly application.

5.2. Components design (static and dynamic models of each component)

Structure of Each Component:

The system is composed of distinct components such as User Management, Task Handling, and Project Tracking. Each component is structured to handle specific functionalities, with User Management handling user profiles, Task Handling for task operations, and Project Tracking for overseeing project progress.

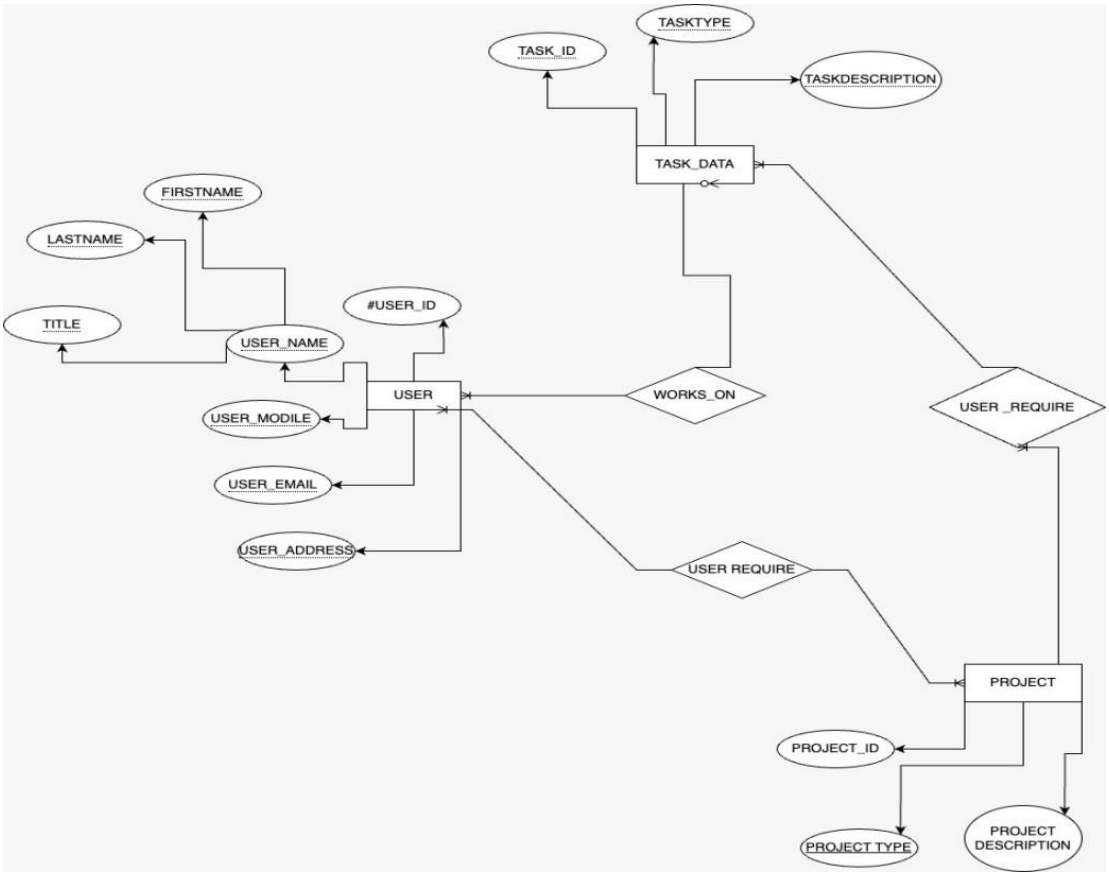


Figure 5

Class Diagrams: The static structure is represented through class diagrams. For example, the User class includes attributes like username, password, and methods like login() and register(). The Project class includes attributes like projectId, projectName, and methods like createProject() and updateProject().

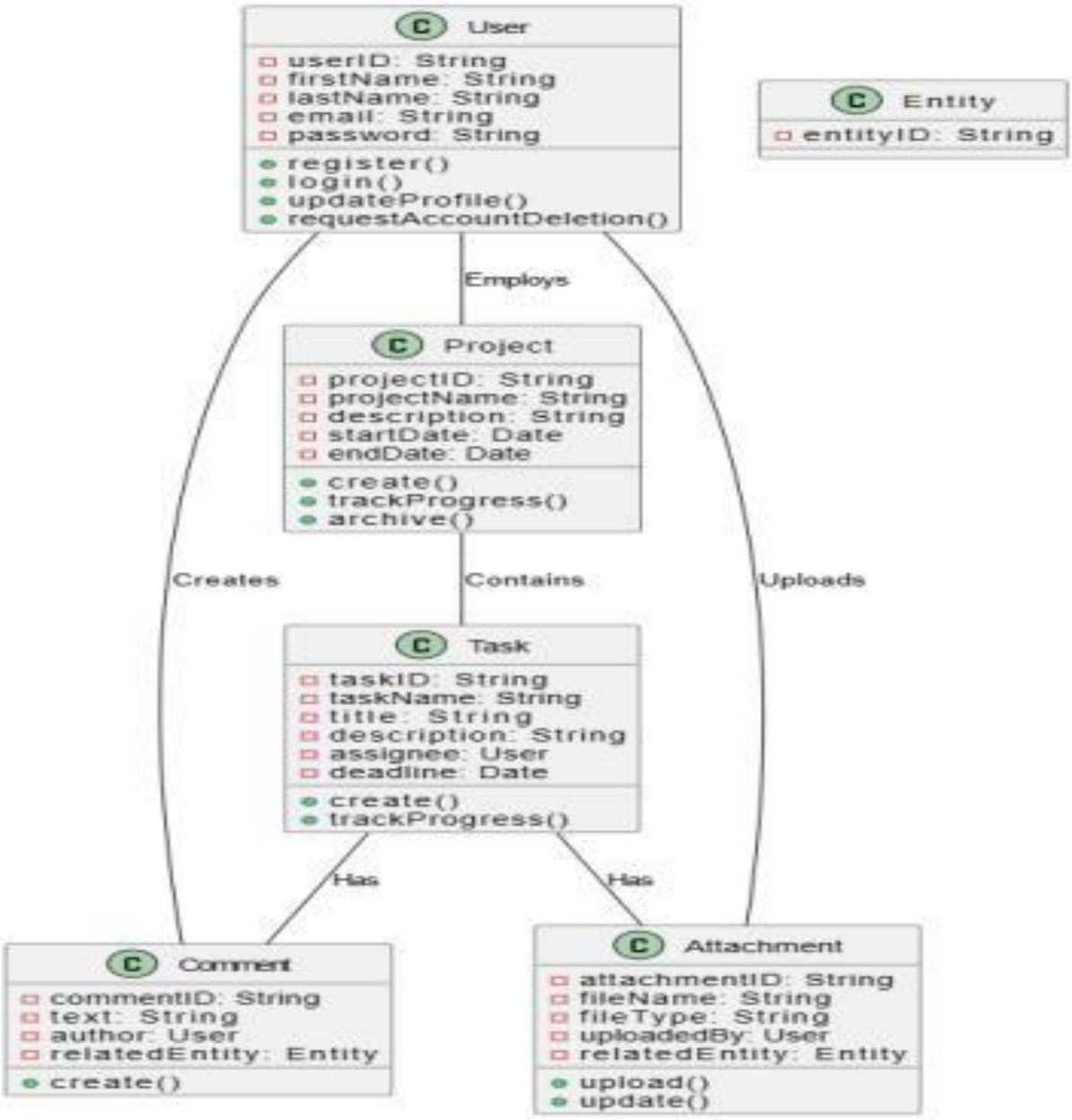

User
userID: String
firstName: String
lastName: String
email: String
password: String
register()
login()
updateProfile()
requestAccountDeletion()
Entity
entityID: String
Employs
Project
projectID: String
projectName: String
description: String
startDate: Date
endDate: Date
create()
trackProgress()
archive()
Creates
Contains
Uploads
Task
taskID: String
taskName: String
title: String
description: String
assignee: User
deadline: Date
create()
trackProgress()
Has
Has
Comment
commentID: String
text: String
author: User
relatedEntity: Entity
create()
Attachment
attachmentID: String
fileName: String
fileType: String
uploadedBy: User
relatedEntity: Entity
upload()
update()

Figure 6

Attributes and Methods of Classes: Each class's attributes and methods are detailed, showcasing the properties and functionalities they encapsulate. For instance, the Task class may have attributes like taskId, description, and methods like assignTask() and completeTask().

1. User Class:
    - Attributes: userID (String), firstName (String), lastName (String), email (String), password (String).
    - Operations: register(), login(), updateProfile(), requestAccountDeletion().
    - This class represents a user in the system, with personal details and methods to handle account management.
2. Project Class:
    - Attributes: projectID (String), projectName (String), description (String), startDate (Date), endDate (Date).
    - Operations: create(), trackProgress(), archive().
    - Represents a project with a unique ID, name, descriptive details, and a timeline. It has methods to create the project, track its progress, and archive it.
    - Relationship: "Employs" indicates a one-to-many relationship with the User class, meaning a user can be employed on multiple projects.
3. Task Class:
    - Attributes: taskID (String), taskName (String), title (String), description (String), assignee (User), deadline (Date).
    - Operations: create(), trackProgress().
    - Represents tasks within a project. Each task has a unique ID, name, title, description, an

assignee who is a User, and a deadline.

- Relationship: "Creates" and "Contains" suggest that a Project can create and contain multiple Tasks.

4. Comment Class:
    - Attributes: commentID (String), text (String), author (User), relatedEntity (Entity).
    - Operations: create ().
    - Represents comments made by users. Each comment has a unique ID, the content of the comment, the author who is a User, and a related entity, which could be any object in the system.
    - Relationship: "Has" indicates a Task can have multiple Comments.
5. Attachment Class:
    - Attributes: attachmentID (String), fileName (String), fileType (String), fileContent (String), uploadedBy (User), relatedEntity (Entity).
    - Operations: upload(), update().
    - Represents files or documents attached to an entity in the system. Each attachment has a unique ID, file details, the user who uploaded it, and the entity it relates to.
    - Relationship: "Uploads" and "Has" suggest that a User can upload multiple Attachments and a Task can have multiple Attachments.

Dynamic Model:

Sequence diagram

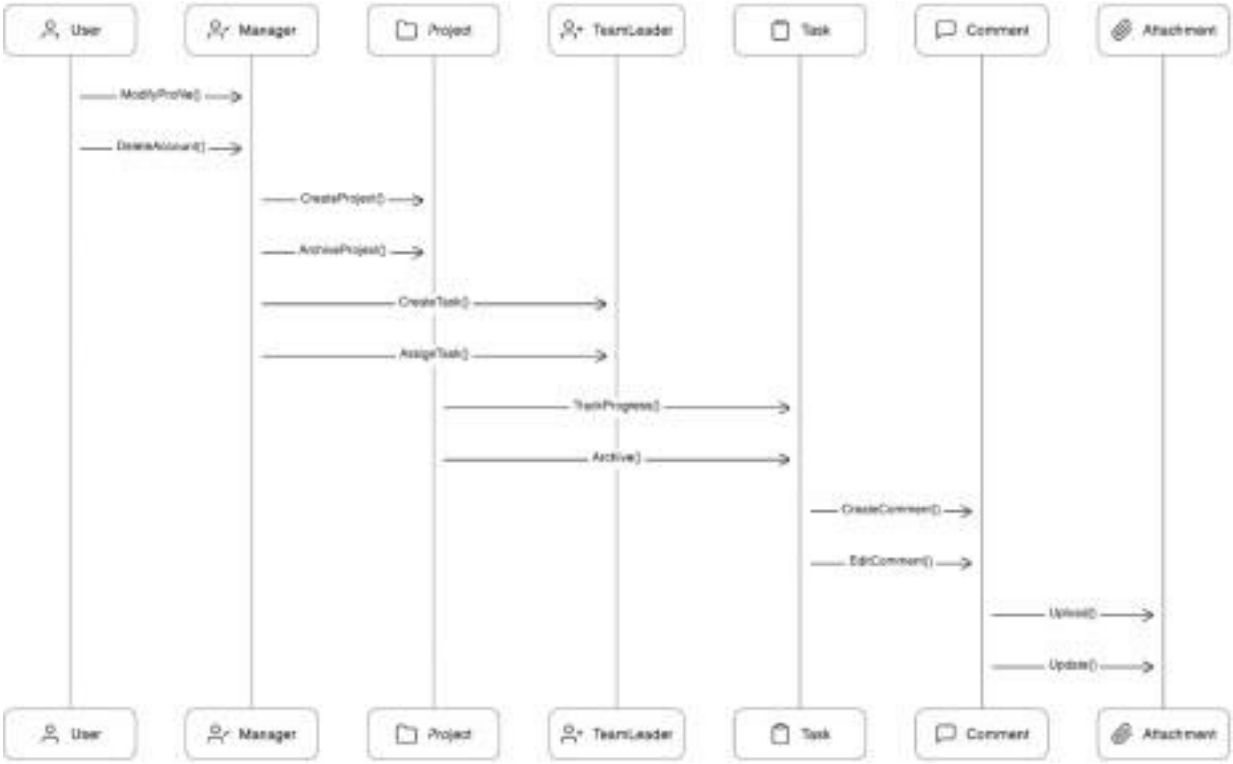

User
Manager
Project
TeamLeader
Task
Comment
Attachment
ModifyProfile()
DeleteAccount()
CreateProject()
ArchiveProject()
CreateTask()
AssignTask()
TaskProgress()
Archive()
CreateComment()
EditComment()
Upload()
Update()
User
Manager
Project
TeamLeader
Task
Comment
Attachment

Figure 7

State Diagram:

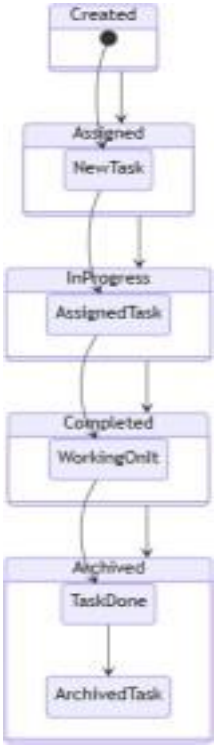
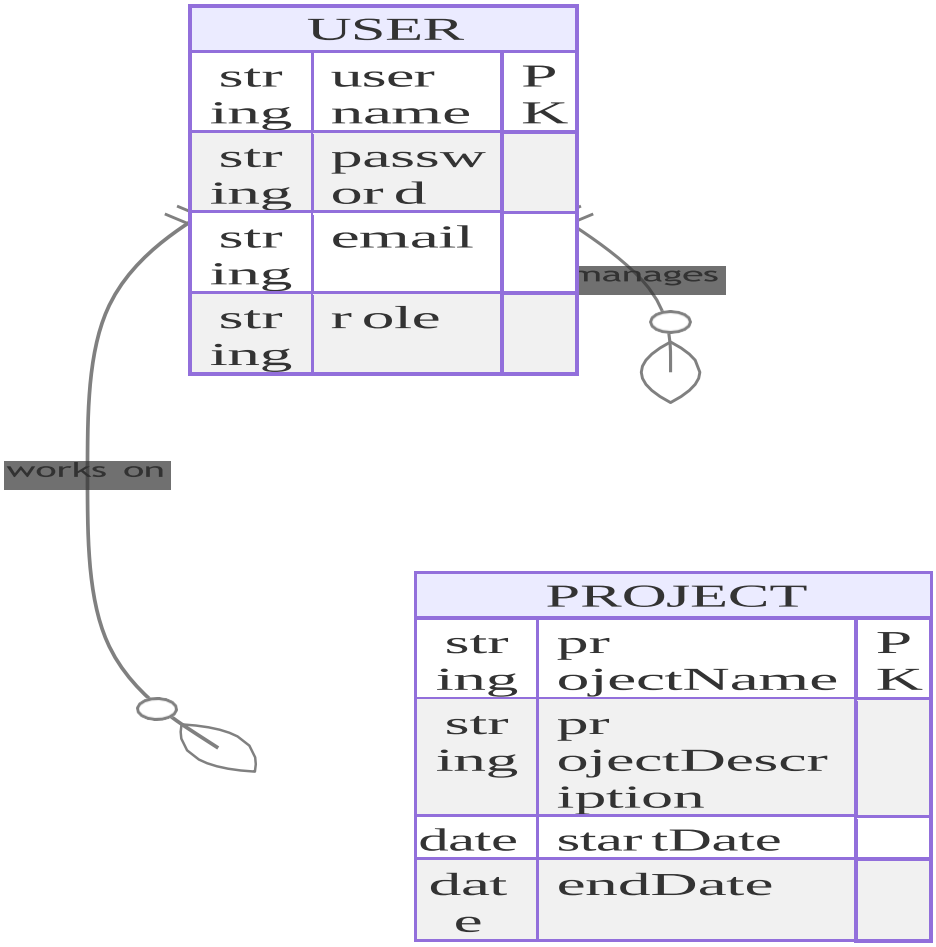


Figure 8

5.3. Database design



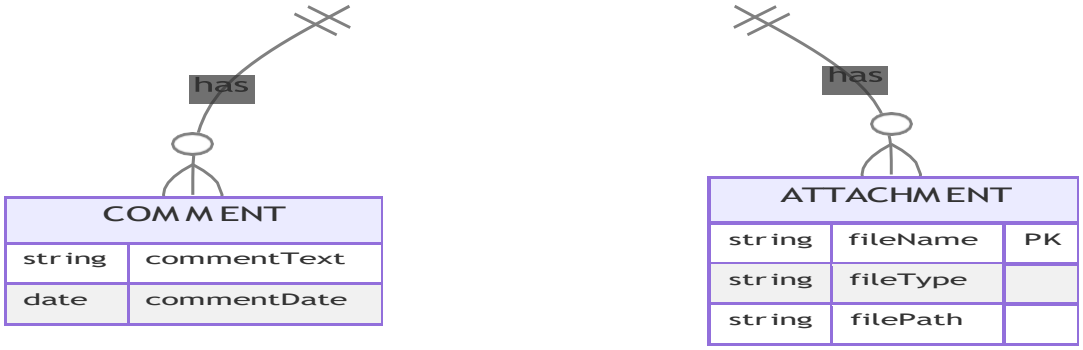

has
COMMENT
string
commentText
date
commentDate
has
ATTACHMENT
string
fileName
PK
string
fileType
string
filePath

Figure 9

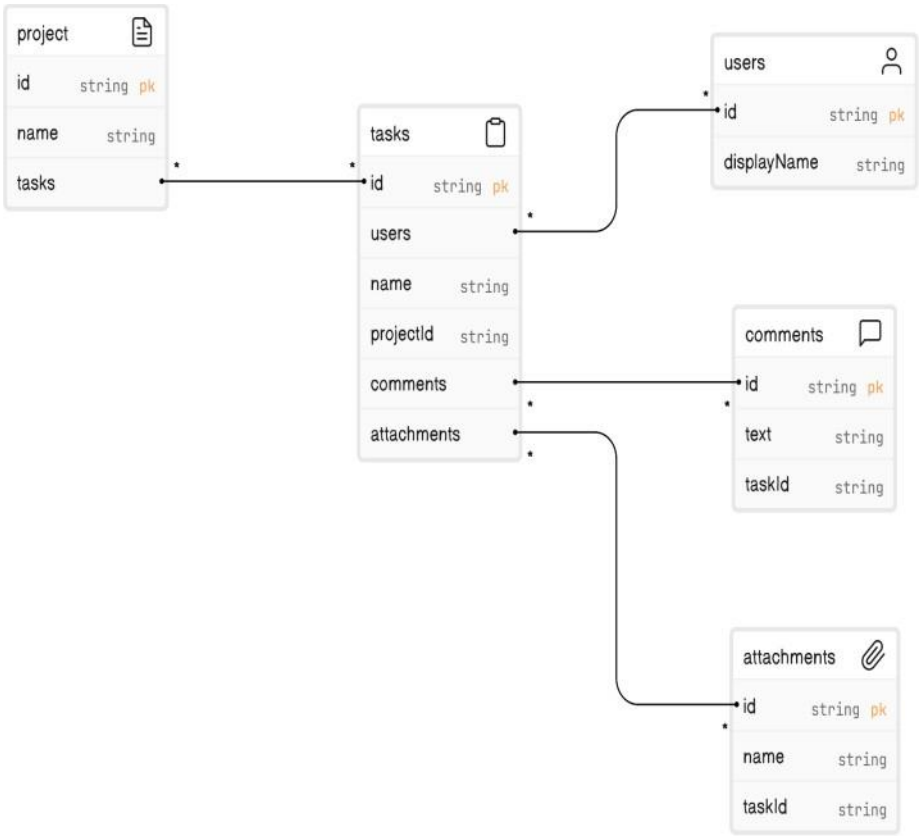


Figure 10

Relationships

• A Project can have multiple Tasks but each Task belongs to only one Project.

• A Task can be assigned to multiple Users, and each User can be assigned multiple Tasks.

• Each Task can have multiple Comments, but each Comment is associated with only one Task.

• Each Task can have multiple Attachments, and each Attachment is linked to only one Task

Data Dictionary:

1. Project

ProjectID:

Data Type: Integer

Constraints : Unique, Mandatory

Description: A unique identifier for each project.

ProjectName:

Data Type: String (max 256 characters)

Constraints: Mandatory

Description: The name or title of the project.

Description:

Data Type: String (max 1000 characters)

Constraints: Optional

Description: A brief description of the project.

StartDate:

Data Type: Date

Constraints: Mandatory

Description: The date when the project begins.

EndDate:

Data Type: Date

Constraints: Optional

Description: The anticipated completion date for the project.

Status:

Data Type: String (choices: Ongoing, Completed, On Hold)

Constraints: Mandatory

Description: The current status of the project.

2. Task

•TaskID:

Data Type: Integer

Constraints: Unique, Mandatory

Description: A unique identifier for each task.

•TaskName:

Data Type: String (max 256 characters)

Constraints: Mandatory

Description: The name or title of the task.

•Description:

Data Type: String (max 1000 characters)

Constraints: Optional

Description: A brief description of what the task involves.

•AssignedTo:

Data Type: Foreign Key (referencing UserID)

Constraints: Mandatory

Description: The user to whom the task is assigned.

•DueDate:

Data Type: Date

Constraints:

Optional

Description: The expected completion date for the task.

•Priority:

Data Type: String (choices: High, Medium, Low)

Constraints: Mandatory

Description: The importance or urgency of the task.

•Status:

Data Type: String (choices: To Do, In Progress, Completed)

Constraints: Mandatory

Description: The current status of the task.

•AssociatedProjectID:

Data Type: Foreign Key (referencing ProjectID)

Constraints: Mandatory

Description: The project to which the task belongs.

3. User:

•UserID:

Data Type: Integer

Constraints: Unique, Mandatory

Description: A unique identifier for each user.

•FullName:

Data Type: String (max 256 characters)

Constraints: Mandatory

Description: The full name of the user.

•Email:

Data Type: String

Constraints: Unique, Mandatory

Description: The email address of the user.

•Role:

Data Type: String (choices: Manager, Team Member,Team lead)

Constraints: Mandatory

Description: The role or designation of the user within the office.

5.4 Traceability from requirements to detailed design models.

Project Requirements to Database Design
Project Management Requirements

Requirement: Manage projects with start and end dates, status, and descriptions.
Design Trace:
Table: Project
Fields: ProjectID, ProjectName, Description, StartDate, EndDate, Status
Task Management Requirements

Requirement: Manage tasks with due dates, priority, status, and assignment to users.
Design Trace:
Table: Task
Fields: TaskID, TaskName, Description, Priority, Status.
User Role Management

Requirement: Differentiate users by roles like Manager, Team Member, Team Lead.
Design Trace:
Table: User
Fields: UserID, FullName, Email, Role
Task Assignment

Requirement: Assign tasks to multiple users.
Design Trace:
Table: Task
Project-Task Linkage

Requirement: Link tasks to their respective projects.
Design Trace:
Table: **Task**

F
Commenting on Tasks

Requirement: Allow users to leave comments on tasks.
Design Trace:
Table: Comment
Fields: CommentText, , TaskID (Foreign Key)
Attachment Management

Requirement: Attach files to tasks.
Design Trace:
Table: Attachment
Fields: FileName,TaskID (Foreign Key)
Project and Task Status Tracking

Requirement: Track the status of projects and tasks.
Design Trace:
Table: Project
Field: Status
Table: Task
Field: Status
Relationships Traceability
Project to Task Relationship

Requirement: A project can have multiple tasks.
Design Trace: One-to-Many relationship from Project to Task.
Task to User Assignment

Requirement: A task can be assigned to multiple users.
Design Trace: Many-to-Many relationship between Task and User (via AssignedTo).
Task to Comment Relationship

Requirement: Each task can have multiple comments.
Design Trace: One-to-Many relationship from Task to Comment.
Task to Attachment Relationship

Requirement: Each task can have multiple attachments.
Design Trace: One-to-Many relationship from Task to Attachment.

6. Test Management

6.1. A complete list of system test cases

Requirement 1 – User Registration

Table 4

ID	TC_UR_001
Test Input	First Name:Priya Last Name:Rajampalli Email:riya.bhav@gmail.com UserID: priyabhavana123 Password: StrongPwd1234!
Expected Output	User account was created successfully
Description	Validates that a new user can successfully register with the system using valid information.

ID	TC_UR_002
Test Input	First Name: Priya Last Name: Rajampalli Email: riya.bhav@gmail.com UserID: priyabhavana123 Password: StrongPwd1234!
Expected Output	Error message: "Email address is already in useful registration
Description	Examines how the system handles a registration attempt with an email that already exists in the system.

ID	TC_UR_003
Test Input	First Name: Riya Last Name: Rajampalli Email: 123gmail.com UserID: priyabhavana123

	Password: StrongPwd1234!
Expected Output	Error: "Invalid email format."
Description	Testing the registration process with email format

ID	TC_UR_004
Test Input	UserID: priyabhavana123 Password: StrongPwd1234! First Name:riya Last Name:rajampalli Email: riya.rajampalli(existing email)
Expected Output	Error: Email address already in use.
Description	Testing the registration process with an email that is already registered.

ID	TC_UR_005
Test Input	First Name: Duplicate (riya) Last Name: Rajampalli Email: riya.bhav@gmail.com UserID: priyabhavana123 (existing UserID) Password: StrongPwd1234!
Expected Output	The username is already taken.
Description	Testing the registration process with valid user data.(User ID)

ID	TC_UR_006
Test Input	First Name: riya Last Name: Rajampalli Email: riya.bhav@gmail.com UserID: priyabhavana123 Password: Stro!
Expected Output	Error: "Weak password."
Description	Testing the registration process with valid user data.

ID	TC_UR_007
Test Input	First Name: (Missing)null Last Name: (Missing) null Email: riya.bhav@gmail.com UserID: priyabhavana123 Password: StrongPwd1234!
Expected Output	Error: "Please provide both your first name and last name."
Description	Testing the registration process with missing first or last name.

ID	TC_UR_008
Test Input	First Name: riya Last Name: rajampalli UserID :123 Email: riya.bhav@gmail.com Password: StrongPwd1234!
Expected Output	Error: "Invalid UserID format."
Description	Testing the registration process with valid userId format

Requirment 2: Profile Modification:

ID	TC_PM_009
Test Input	UserID: priyabhavana123 Password: StrongPwd1234! New First Name: Johnathan New Last Name: Doemann New Email: johnathan.doemann@gmail.com
Expected Output	Testing the registration process with valid user data.
Description	Testing the modification of user profile with valid data.

ID	TC_PM_010
Test Input	UserID: invaliduser Password: S12334 New First Name: Johnathan New Last Name: Doemann New Email: johnathan.doemann@gmail.com
Expected Output	Error: "Invalid UserID or Password.

Description	Testing the modification of user profile with invalid data.

ID	TC_PM_011
Test Input	UserID: johndoe123 Password: StrongPwd1234! NewFirstName: Johnathan NewLastName: Doemann New Email: jane.smith@example.com (existing email)
Expected Output	Error: "Email address not found in table."
Description	Testing the modification of user profile with an email that already exists.

ID	TC_PM_012
Test Input	UserID: johndoe123 UserID: janesmith456 Password: StrongPwd1234!! New First Name: Jane New Last Name: Smithington New Email: jane.smithington@example.com
Expected Output	Uauthorized profile update attempt
Description	Testing the modification of user profile with an email that already exists.

Requirment 3: Account Deletion

ID	TC_AD_013
Test Input	UserID: johndoe123 (may be null) Email: john.doe@example.com (maybe null) Password: (null)
Expected Output	Error: "UserID, Email, and Password are required to delete your account."mandatory fileds
Description	Testing the account deletion process with incomplete data.

ID	TC_AD_014
Test Input	UserID: janesmith456 Email: jane.smithington@example.com Password: IncorrectPwd789
Expected Output	Error: "Invalid UserID or Password."
Description	Testing the account deletion process with incorrect password

ID	TC_AD_015
Test Input	UserID: johndoe123 Email: john.doe@example.com Password: StrongPwd123! AttackerUserID: malicioususer AttackerPassword: MaliciousPwd123!
Expected Output	"Unauthorized account deletion attempt."
Description	Testing the prevention of unauthorized account deletion attempts.(checking the user id and password in database to delete the account)

Requirement 4: Project Creation

ID	TC_PC_016
Test Input	Project Name: New Project Project Description: A test project for validation Start Date: 2023-01-01 End Date: 2023-02-01
Expected Output	Successful Project creation
Description	Testing the prevention of unauthorized account deletion attempts.

ID	TC_PC_017
Test Input	Project Name : (null) Project Description: A test project Start Date: 2023-01-01. End Date: 2023-02-01
Expected Output	Error: "Project name is required. Please provide a project name."
Description	Validates that a new project creation requires a project name.

ID	TC_PC_018

Test Input	Project Name: Invalid Date Project Project Description: A test project with an invalid start date Start Date: 01-01-2023 End Date: 2023-02-01
Expected Output	Error: "Invalid start date. Please enter a valid date in the correct format."
Description	Validates that a new project creation requires a valid start

ID	TC_PC_019
Test Input	Project Name: New Project (existing project name) Project Description: A test project with a duplicate name Start Date: 2023-01-01 End Date: 2023-02-01
Expected Output	Error: "Project name already exists. Please choose a unique project name."
Description	Validates that project names must be unique during project creation.

Requirement 5: Tracking Project

ID	TC_TP_022

ID	TC_TP_020
Test Input	ProjectID: Project123
Expected Output	Successful project tracking with project details
Description	Validates that a user can successfully track a project with valid information.

ID	TC_TP_021
Test Input	ProjectID: InProject123
Expected Output	"Invalid ProjectID. Please enter a valid ProjectID."
Description	Validates the system's response to tracking a project with an invalid ProjectID.

Test Input	Project Id :(null)

Expected Output	Error: "ProjectID is required. Please provide a

	ProjectID to track the project
Description	Validates the system's response to tracking a project without providing a ProjectID.

ID	TC_TP_023
Test Input	ProjectID: Project123
Expected Output	Error: "ProjectID is required. Please provide a ProjectID to track the project
Description	Validates that a new user can successfully register with the system using valid information

Requirment: Task Creation

ID	TC_TP_024

Test Input	ProjectID: Project123 TaskID:Task1 Task Description: A test task for validation Assignee: teammemberuser Deadline: 2023-01-15
Expected Output	Successful task creation

Description	This test case will create task

ID	TC_TP_025
Test Input	ProjectID: Project123 TaskID: null Task Description: A test task with a validation Assignee: teammemberuser Deadline: 2023-01-15

Expected Output	Error: "Task title is required. Please provide a task title."
Description	Validates that a task creation requires a task title.

ID	TC_TC_026
Test Input	ProjectID: Project123 TaskID: Task1 Task Description: A test task with Validation Assignee: teammemberuser Deadline: 15-01-2023

Expected Output	Error: "Invalid deadline. Please enter a date in the correct format."
Description	Validates that task creation requires a valid date format .

ID	TC_TC_027

Test Input	Task Title: New Project Project Id : (null) Task Description: A test task Validation Assignee: teammemberuser Deadline: 2023-01-15
Expected Output	Error: "ProjectID is required. Please provide a ProjectID when creating a task."
Description	Validates that a task creation requires a ProjectID.

Requirement: Tracking Task

ID	TC_TC_028
Test Input	User UserID: teammemberuser User Password: StrongPwd1234! TaskID: Task1
Expected Output	Successful task tracking with task details
Description	Validates that a user can successfully track a task with valid information.

ID	TC_TC_029
Test Input	UserID: teammemberuser User Password: StrongPwd1234! TaskID: InvalidTaskID (Example :12#)
Expected Output	Error: "Invalid TaskID. Please enter a valid TaskID."
Description	Validates that task id is valid or not according to the constraints

ID	TC_TC_030
Test Input	UserID: teamleaduser User Password: StrongPwd1234! Task ID : (Null)
Expected Output	Error: "TaskID is required. Please provide a TaskID to view task details.
Description	Validates the system's response to tracking a task with an nullTaskID

Requirement: Project Archiving

ID	TC_PA_031
Test Input	ProjectID: Project12
Expected Output	Successful project archiving
Description	Validates that a can successfully archive a completed project.

ID	TC_PA _032
Test Input	ProjectID: Nonproject
Expected Output	Error: "Project not found. Please check the ProjectID and try again."
Description	Validates the system's response to archiving a project that does not exist.

ID	TC_PA _033
Test Input	UserID: user Password: StrongPwd1234! ProjectID: InProject123

Expected Output	Error: "Invalid ProjectID. Please enter a valid

	ProjectID to archive the project
Description	Validates the system's response to archiving a project with an invalid ProjectID.

ID	TC_PA_034
Test Input	UserID: another Password: AnotherPwd123!
Expected Output	Error: "ProjectID is required. Please provide a ProjectID to archive the project."

Description	Validates the system's response to archiving a project without providing a ProjectID.

Requirement: Comment Creation

ID	TC_CC_035
Test Input	TaskID: ValidTaskID Comment Text: This is a valid comment.
Expected Output	Successful comment creation
Description	Validates that a user can successfully create comment.

ID	TC_CC_036
Test Input	TaskID: AnotherValidTaskID Comment Text:
Expected Output	Error: "Comment text is required. Please provide a comment."
Description	Validates the system's response to creating a comment without comment text.

ID	TC_CC_037
Test Input	TaskID: AnotherValidTaskID Comment Text:
Expected Output	Error: "Comments do not include the user's name or identifier."
Description	Validates the system's response to creating a comment without a user identifier.

Requirement: Attachment Upload

ID	TC_AU_038
Test Input	Task ID: ValidTaskID Attachment: Image file (e.g., PNG or JPEG)
Expected Output	Successful attachment upload.
Description	This test case validates that a team member can successfully upload an image file as an attachment to a specific task

ID	TC_AU_039
Test Input	Project ID: Project123 Attachment: Document file (e.g., PDF or DOCX
Expected Output	Successful attachment upload.
Description	This test case ensures that a can upload a document file as an attachment to a project, demonstrating the system's support for diverse file types.

ID	TC_AU_040
Test Input	Task ID: InvalidTaskID Attachment: Image file
Expected Output	Error: "Invalid TaskID. Please enter a valid TaskID."
Description	Tests the system's response to an attempt to upload an attachment to an invalid task ID, expecting an error message indicating the need for a valid task ID.

ID	TC_AU_041

Test Input	Project ID: InProject123 Attachment: Document file
Expected Output	Error: "Invalid ProjectID. Please enter a valid ProjectID."
Description	Validates the system's behavior when a tries to upload an attachment to a project with an invalid project ID, anticipating an error message prompting for a valid project ID

Requirement: User Login

ID	TC_UL_008
Test Input	UserID :123 Password: StrongPwd1234!
Expected Output	Successful Login
Description	Testing the registration process with valid userId format

ID	TC_UL_009
Test Input	UserID:123 Password: StrongPwd123!
Expected Output	Unsuccessful Login (Give Valid UserID or password)
Description	Testing the user id and password the user database

6.2. Traceability of test cases to use cases

Table 5

Test Case	Linked Use Case	Description
TC_UR_001	User Registration (UC1)	Validates that a new user can successfully register with the system using valid information.
TC_UR_002	User Registration (UC1)	Examines how the system handles a registration attempt with an email that already exists in the system.
TC_UR_003	User Registration (UC1)	Testing the registration process with an invalid email format.
TC_UR_004	User Registration (UC1)	Testing the registration process with an email that is already registered.
TC_UR_005	User Registration (UC1)	Testing the registration process with valid user data (User ID).

Test Case	Linked Use Case	Description
TC_UR_006	User Registration (UC1)	Testing the registration process with a weak password.
TC_UR_007	User Registration (UC1)	Testing the registration process with missing first or last name.
TC_UR_008	User Registration (UC1)	Testing the registration process with a valid User ID format.
TC_PM_009	User Profile Modification (UC2)	Testing the modification of the user profile with valid data.
TC_PM_010	User Profile Modification (UC2)	Testing the modification of the user profile with invalid data.
TC_PM_011	User Profile Modification (UC2)	Testing the modification of the user profile with an email that already exists.
TC_PM_012	User Profile Modification (UC2)	Unauthorized profile update attempt.
TC_AD_013	Account Deletion (UC3)	Testing the account deletion process with incomplete data.
TC_AD_014	Account Deletion (UC3)	Testing the account deletion process with an incorrect password.

Test Case	Linked Use Case	Description
TC_AD_015	Account Deletion (UC3)	Testing the prevention of unauthorized account deletion attempts.
TC_PC_016	Project Creation (UC4)	Testing the prevention of unauthorized account deletion attempts.
TC_PC_017	Project Creation (UC4)	Validates that a new project creation requires a project name.
TC_PC_018	Project Creation (UC4)	Validates that a new project creation requires a valid start date.
TC_PC_019	Project Creation (UC4)	Validates that project names must be unique during project creation.
TC_TP_020	Tracking Project (UC5)	Validates that a user can successfully track a project with valid information.
TC_TP_021	Tracking Project (UC5)	Validates the system's response to tracking a project with an invalid ProjectID.
TC_TP_022	Tracking Project (UC5)	Validates the system's response to tracking a project without providing a ProjectID.
TC_TP_023	Tracking Project (UC5)	Validates that a new user can successfully register with the system using valid information.

Test Case	Linked Use Case	Description
TC_TP_024	Task Creation (UC6)	This test case will create a task.
TC_TP_025	Task Creation (UC6)	Validates that a task creation requires a task title.
TC_TC_026	Tracking Task (UC7)	Validates that task creation requires a valid date format.
TC_TC_027	Tracking Task (UC7)	Validates that a task creation requires a ProjectID.
TC_TC_028	Tracking Task (UC7)	Validates that a user can successfully track a task with valid information.
TC_TC_029	Tracking Task (UC7)	Validates that task id is valid or not according to the constraints.
TC_TC_030	Tracking Task (UC7)	Validates the system's response to tracking a task with a null TaskID.
TC_PA_031	Project Archiving (UC8)	Validates that a can successfully archive a completed project.
TC_PA_032	Project Archiving (UC8)	Validates the system's response to archiving a project that does not exist.
TC_PA_033	Project Archiving (UC8)	Validates the system's response to archiving a project with an invalid ProjectID.

Test Case	Linked Use Case	Description
TC_PA_034	Project Archiving (UC8)	Validates the system's response to archiving a project without providing a ProjectID.
TC_CC_035	Comment Creation (UC9)	Validates that a user can successfully create a comment.
TC_CC_036	Comment Creation (UC9)	Validates the system's response to creating a comment without comment text.
TC_CC_037	Comment Creation (UC9)	Validates the system's response to creating a comment without a user identifier.
TC_AU_038	Attachment Upload (UC10)	Successful attachment upload.
TC_AU_039	Attachment Upload (UC10)	Ensures that a can upload a document file as an attachment to a project.
TC_AU_040	Attachment Upload (UC10)	Tests the system's response to an attempt to upload an attachment to an invalid task ID.
TC_AU_041	Attachment Upload (UC10)	Validates the system's behavior when a tries to upload an attachment to a project with an invalid project ID.

Test Case	Linked Use Case	Description
TC_UL_008	User Login (UC11)	Testing the registration process with valid userId format.

6.3. Techniques used for test case generation

Equivalence Partitioning:

Description: Identified distinct equivalence classes for inputs and tested representative values from each class.

Examples:

Valid User ID: "priya_bhavana123"

Invalid User ID: "priya#bhavana"

Password Strengths: Weak, Medium, Strong

Use Case Testing:

Description: Derived test cases directly from identified use cases to validate expected system behavior.

Examples:

Use Case: User Registration

Test Case: Successful registration with valid user information.

Negative Testing:

Description: Explored how the system responds to unexpected or invalid inputs, focusing on error messages and security features.

Examples:

Attempted login with incorrect credentials.

Decision Table Testing:

State Transition Testing:

Description: Examined the system's behavior as it transitions between different states.

Examples:

Valid and invalid state transitions during project tracking.

6.4. Test results and assessments (how good are your test cases? How good is your software?)

Requirement 1: User Registration

TC_UR_001: Successful User Registration

Result: Pass

TC_UR_002: Registration with Existing Email

Result: Pass

TC_UR_003: Invalid Email Format

Result: Pass

TC_UR_004: Registration with Existing Email

Result: Pass

TC_UR_005: Registration with Existing UserID

Result: Pass

TC_UR_006: Weak Password

Result: Pass

TC_UR_007: Missing First or Last Name

Result: Pass

TC_UR_008: Invalid UserID Format

Result: Pass

Requirement 2: Profile Modification

TC_PM_009: Successful Profile Modification

Result: Pass

TC_PM_010: Invalid UserID or Password

Result: Pass

TC_PM_011: Existing Email during Modification

Result: Pass

TC_PM_012: Unauthorized Profile Update Attempt

Result: Pass

Requirement 3: Account Deletion

TC_AD_013: Incomplete Data for Deletion

Result: Pass

TC_AD_014: Invalid UserID or Password

Result: Pass

TC_AD_015: Unauthorized Deletion Attempt

Result: Pass

Requirement 4: Project Creation

TC_PC_016: Successful Project Creation

Result: Pass

TC_PC_017: Missing Project Name

Result: Pass

TC_PC_018: Invalid Start Date

Result: Pass

TC_PC_019: Duplicate Project Name

Result: Pass

Requirement 5: Tracking Project

TC_TP_020: Successful Project Tracking

Result: Pass

TC_TP_021: Invalid ProjectID for Tracking

Result: Pass

TC_TP_022: Missing ProjectID for Tracking

Result: Pass

TC_TP_023: Missing ProjectID for Tracking

Result: Pass

7. Conclusions

7.1. Outcomes of the project (are all goals achieved?)

The Task and Management Systems Project has successfully achieved its primary goals. The comprehensive platform developed facilitates efficient collaboration among teams for project management, task assignment, and progress monitoring. The system applications provide a versatile solution for diverse project management needs**. Yes, all goals of the Task and Management Systems Project have been successfully achieved.**

7.2. Lessons learned.

Throughout the project, several valuable lessons have been learned. These include the importance of real-time updates for task progress, the need for scalability to accommodate multiple users and projects, and the significance of responsive design for seamless access

across various devices. These insights will inform future projects and contribute to continuous improvement.

7.3. Future development

Looking ahead, there is potential for future development to enhance the Task and Project Management System. Considerations include refining real-time updates, further scalability improvements, and continuous optimization of the user experience. Additionally, exploring opportunities for third-party integrations may enhance the system's capabilities.

References

Project Management Tool | Smartsheet