Table of Contents

Chapter 1: Introduction

1.1. Project Overview

The System is a comprehensive software solution designed to facilitate the management of software project development. It is specifically tailored to implement the principles of agile methodology and provide additional functionality to streamline the workflow of software development projects. The primary goal of the system is to establish a structured framework for effectively allocating and controlling resources, ensuring that all necessary tasks are completed within the defined scope, timeline, and constraints.

1.2. Project Purpose

The main perspective of the project is to develop a simple workflow management system where I will use simple functionality for managing workflow towards multiple team or individual team member. There will have several types of user role in the system for differentiating available feature in the system.

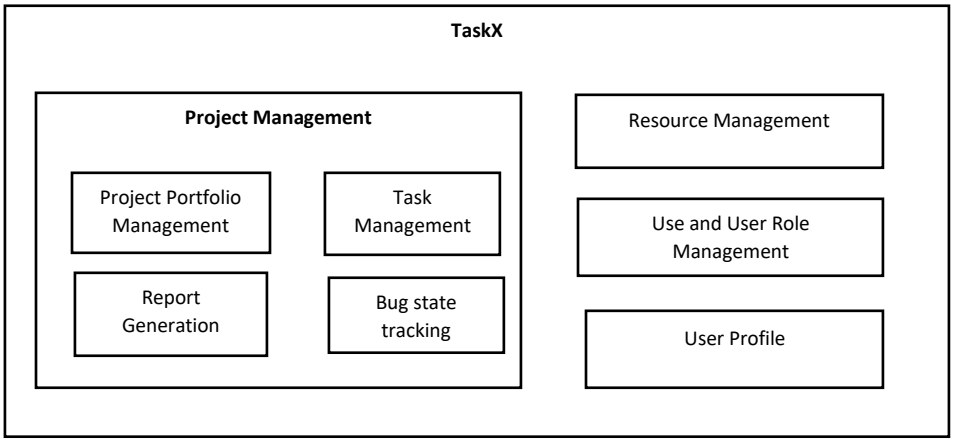


Fig 1: Project Management System Purpose

1.2.1. Background

In today's rapidly evolving software development landscape, effective project management is crucial to ensure the success of software projects. Traditional project management approaches often struggle to keep up with the dynamic nature of software development, leading to inefficiencies, delays, and unsatisfactory outcomes. Agile methodologies have emerged as a popular solution, emphasizing adaptability, collaboration, and iterative development. However, managing agile projects can still pose challenges without the right tools and systems in place.

To address these challenges, the Project Management System (TaskX) is being developed. By incorporating the principles of agile methodology and providing additional functionality tailored to software development, the TaskX aims to streamline project management processes, increase efficiency, and enhance overall project outcomes.

1.2.2. Benefits & Beneficiaries

The main benefits of the project is provide a convenient way to manage agile project between a team where each user or multiple user will be assigned to tasks to complete the project. List of benefits are listed below:

1. Improved Project Planning: The System provides a comprehensive platform for project planning, enabling project managers to define project scope, create schedules, allocate resources, and establish milestones. This structured approach leads to better planning accuracy and enables effective project execution.

2. Enhanced Collaboration and Communication: The project promotes collaboration among project team members by providing a centralized platform for communication, document sharing, and task assignment. It facilitates real-time collaboration, reduces miscommunication, and fosters a cohesive team environment.

3.Increased Efficiency and Productivity: With its agile methodologies used Scrum framework and workflow management features, the System enhances efficiency and productivity. It helps teams prioritize tasks, and track progress, resulting in optimized resource utilization and timely completion of project deliverables.

4.Project Visibility: A project can be divided multiple epic with related functionality and for completing each epic one or multiple sprint will be created where each sprint will contain one or more issues. Here clearly sprint completion will be able to determine project progress.

5.Resource Allocation: The system assists in allocating resources based on availability, skills, and workload. By optimizing resource allocation, it prevents overloading or underutilization of team members, leading to improved productivity and project efficiency.

1.2.3. Goals

The system main objective is to offer a reliable and effective software solution that satisfies the unique requirements of project management in software development. The following objectives are the focus of the project:

1.Streamline Software Project Management: The system implements a framework based on agile approaches in an effort to streamline the management of software projects. It attempts to make project planning, task distribution, progress monitoring, and reporting simpler so that project managers may more easily monitor and manage project operations.

2. Improve Project Productivity and Efficiency: The project's objective is to increase project productivity and efficiency by offering features that allow collaboration, optimize resource allocation, and enable efficient workflow management. The system seeks to boost the general efficiency of software development teams by reducing administrative duties and encouraging seamless communication.

3. Enhance Collaboration and Communication: By creating team multiple user can collaborate together to progress projects and commenting on issues and replying on issues makes individual tasks simple and clear to project stakeholders.

4. Improved Stakeholder Satisfaction: The project aims to improve stakeholders' satisfaction by offering open project visibility, frequent updates, and efficient communication methods. The system seeks to

manage stakeholder expectations, respond quickly to issues, and make sure that project deliverables meet stakeholder needs.

5. Enable Continuous Improvement: The project's goal is to encourage ongoing development of project management techniques for software. The objective is to make it possible for project teams to grow in capability over time and learn from their mistakes.

By accomplishing these goals, the system aims to establish itself as a valuable tool that contributes to the success of software projects, improves project outcomes, and enhances the overall efficiency of project management in software development.

1.3. Stakeholders

Numerous stakeholders who have an impact on the project either directly or indirectly are included in the Project Management System (PMS). The principal parties involved are:

1. **Project Owner:** Project Owner are administrative person of any project who is involved to create project and assigning project lead and configuring other project configurations.
2. **Project Lead:** Project lead are the stakeholders who will involved assigning developer, tester to particular tasks/issue/tasks.
3. **Development Teams:** The system immediately benefits the development teams by giving them a platform for work delegation, collaboration, and progress monitoring. It enables them to maintain organization, control their workload, and have good team communications.
4. **Quality Assurance (QA) Team:** The QA team plays a critical role in ensuring the quality and compliance of software projects. The PMS helps them track testing cycles, manage defects, and monitor the resolution of issues, enhancing their ability to conduct thorough quality assurance and deliver high-quality software.

1.4. Scope and Limitations

The scope of the Project Management System (PMS) is focused on the management of software project development. It encompasses the entire lifecycle of software projects, including planning, execution, monitoring, and closure. The PMS is designed to support the implementation of agile methodologies, providing a framework for efficient workflow management and resource allocation. It enables project managers to effectively oversee and control project activities, ensuring that all necessary work is accomplished within the specified scope, timeline, and defined restrictions.

The PMS encompasses various features and functionalities that aid in project management, such as task allocation, progress tracking, collaboration within team and reporting. It serves as a centralized platform for project stakeholders to access project information, communicate, and make informed decisions regarding project direction and resource utilization. Though **TaskX** provides a convenient way to manage software project , it has some limitation as well. Some limitation listed below:

Applicability to Software Projects Only: The Project Management System is specifically designed for software project management and may not be suitable for managing projects in other domains or

industries. Its functionalities and features are tailored to address the unique requirements and challenges of software development projects.

External System Integration: The PMS focuses solely on the required functionality for software project management and does not encompass the functionality of external systems such as data storage, change management, or version control systems. Integration with these external systems may be necessary to ensure a seamless and comprehensive project management experience.

Decision-Making Support: While the PMS provides some predefined reports pattern for decision-making, it is important to note that it serves as a support system rather than making decisions autonomously. The ultimate responsibility for decision-making lies with the project managers and stakeholders who utilize the information and insights provided by the PMS.

Project-Specific Constraints: The PMS operates within the constraints defined by the project, including scope, timeline, and resource availability. It does not guarantee the elimination of all project-related challenges or constraints but aims to provide a structured framework and tools to manage them effectively.

User Familiarity and Training: The successful implementation and utilization of the PMS may require user familiarity and training. Project team members and stakeholders may need to acquire a certain level of proficiency in using the system and its features to maximize its benefits and overcome potential learning curves.

1.5. Significance of the project

The significance of TaskX system is immense. It offers a streamlined workflow that helps teams efficiently organize tasks, allocate resources, and track project progress. With its visual interface and customizable boards, TaskX enables transparent collaboration and communication among team members, fostering teamwork and enhancing project coordination. By embodying agile principles, TaskX supports iterative development, adaptability to changing requirements, and prioritization of work. It provides a clear visual representation of project status, facilitating better decision-making and identifying potential bottlenecks. Tasks scalability and flexibility make it suitable for projects of various sizes and complexities. Its accessibility through web and mobile platforms makes it ideal for remote teams, ensuring seamless collaboration regardless of physical location. Integration with other tools extends TaskX functionality and enhances project management workflows. In summary, TaskX significantly contributes to project success, team productivity, and stakeholder satisfaction by providing a versatile and user-friendly platform for effective agile project management.

1.6. User Profiles

The system is intended to be used by various users. We can divide all users into three profiles, each with own responsibility and role in the PMS:

User	Functions and Responsibility
Project Owner	Responsible for the batch of the projects and controls overall development flow. Assigns projects to the project team leader and controls fulfilment of the project team leader's tasks.

Project Team Leader	Responsible for a particular project. Leads a project team of developers. Assigns tasks to project team members and controls their fulfilment. Reports to the project Owner.
Project Team Member/ Developer	Responsible for a particular task or part of a task. Reports to the Project Team Leader.

1.7. Acronyms & Glossary

Acronyms:

Term	Explanations
TaskX	The proposed system is entitled by TaskX.
GUI	Graphical User Interface
DBMS	Database Management System
CMS	Change Management System
PMS	Project Management System.

Glossary:

Word	Explanation
Project Management	The main aspect of this document. Represent the entire solution.
Host System	The main part of the system that resides on the server and where the business logic runs. Maintains physical connections to all external systems (data storage system, version control and change management systems)
Client System	The part of the system that runs on the user PC. Provide GUI and required system functionality. Maintains physical connection to the host system
Project Team Leader	The person who has the overall responsibility for the successful planning and execution of any project. Project Team Leader leads the team of developers.
Manager/ Project Owner	The person who has the overall responsibility for the project portfolio
Project Team Member	One of the developers who does not have responsibility for the project. The project team member has responsibility for carrying out the task assigned to him or her.
User	Any person who uses the system and is registered within the system. It means that he or she has the user login.
User Profile	Preferences of the registered user of the system that are saved within the system
Report	A defined view on the project that contains the specified project attributes tasks and resources and provides information about project status.

Authorized user	The user who has logged into the system and has a right to perform some operation. The system “knows” the identity of the user and permission that are granted to this user.
Authenticated user	The user who has logged into the system. The system “knows” the identity of the user.

Chapter 2: Project Planning and Management

2.1. Project Charter and Goals

Project Name: TaskX

Project Description: The System is a robust software solution that simplifies the management of software project development. Built on the principles of agile methodology, it offers enhanced functionality to optimize the workflow of software development projects. With a focus on resource allocation and control, the system establishes a structured framework to ensure timely completion of tasks within defined scope and constraints. It empowers teams to collaborate efficiently, track progress, and achieve project goals effectively.

Project Objectives:

- Provide a framework for project management
- Supports multiple project
- Support distributed development
- Support workflow management
- Support Scrum framework of agile methodology in project
- Support sprint wise task management in scrum frameworks project
- Allows to create dependencies between tasks
- Provide user role management
- Provide report generation
- Support messaging feature between users
- Provide user and user role management

Stakeholders:

- Developer
- Tester
- Project Lead
- Technical writer
- Project Owner

Deliverables:

- Project Plan

- Software Specification Requirement(SRS)
- Source Code
- User Interface Design
- Database Design and Implementation
- Project Documentation

2.2. Project Timeline and Milestones

Project Timeline and Milestones for the project, organized by weeks and phases:

Week 1-2: Project Initiation

- Identifying the project objectives
- Identify project stakeholders and their roles.
- Develop the project charter and establish the workflow.

Week 3-6: Requirements Gathering and Analysis

- Collecting requirement
- Analyze and prioritize requirements based on user needs and project goals.
- Create user stories and define acceptance criteria.
- Collaborate with stakeholders to ensure shared understanding of requirements.

Week 6-7: System Design and Architecture

- Design the system architecture and define the overall structure of the agile project management system on Scrum.
- Identify key modules, components, and integration points.
- Develop wireframes or prototypes to visualize the user interface and workflow.

Week 7-13: Development and Testing

- Implement the system functionalities according to the defined user stories.

Week 13-15: Deployment and Release

- Prepare the system for deployment to the production environment.
- Coordinate with IT operations to set up the necessary infrastructure.
- Conduct user training sessions to ensure users are familiar with the system.
- Define a release plan and schedule for deploying new features or updates.

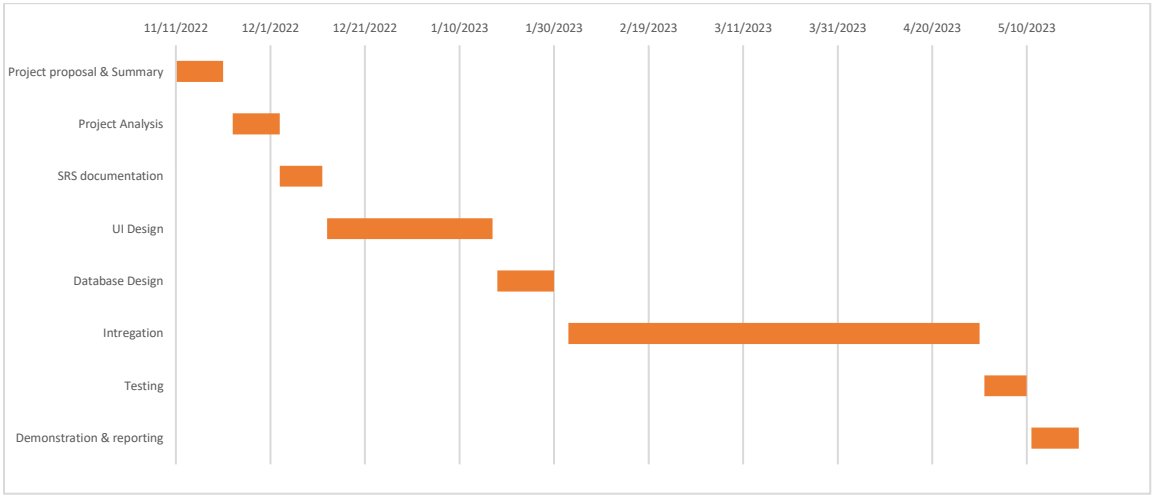
Week 16: Project Closure

- Reporting on final project

These milestones and timeline provide a structured plan for the project, allowing the team to effectively manage and track progress throughout the development and implementation of the agile project management system.

2.2.1. Gantt Chart

A Gantt chart is a type of bar chart that illustrates a project schedule. It shows the start and end dates of tasks, as well as the dependencies between tasks. Gantt charts are a valuable tool for software project management.



Gant Chart of Project

2.3. Resource Allocation and Budgeting

As the project was BSSE final year project, so it was developed by only me and supervised by one of my teacher so resource allocation and budgeting is not a complex or requirable arena to discussed.

2.4. Risk Assessment and Mitigation Strategies

Risk Assessment for the project:

1. Technical Risks:

- Integration Challenges: There may be difficulties integrating the project management system with existing tools or platforms used by the organization. This could impact data transfer, synchronization, or functionality.
- Scalability Issues: The system may face scalability challenges if the user base grows rapidly or if there is a significant increase in data volume. This could affect system performance and responsiveness.

2. Security Risks:

- Data Breach: There is a risk of unauthorized access or data breaches, which could compromise sensitive information, user data, or project details. Robust security measures, such as encryption and authentication, must be implemented to mitigate this risk.
- User Authentication: Inadequate user authentication mechanisms may lead to unauthorized access or account misuse. Strong password policies and multi-factor authentication can help minimize this risk.

3. User Adoption Risks:

- User Resistance: Users may resist transitioning from existing project management tools to the new system. Adequate user training, change management strategies, and clear communication can address this risk.
- Lack of User Engagement: If the system does not provide a user-friendly interface, intuitive workflows, or adequate features, users may not fully engage with it. Regular user feedback, usability testing, and continuous improvement efforts are essential to mitigate this risk.

4. Project Management Risks:

- At the time of sharing different issue/Tasks/Bugs to the different boards developers can leaks this information for bad purpose which could affect the whole project under a security risks.

To effectively manage these risks, it is important to conduct a comprehensive risk assessment at the beginning of the project and regularly review and update it throughout the project lifecycle. Risk mitigation strategies, contingency plans, and monitoring mechanisms should be implemented to minimize the impact of identified risks and ensure the successful delivery of the project management system.

2.5. Proposed System Model

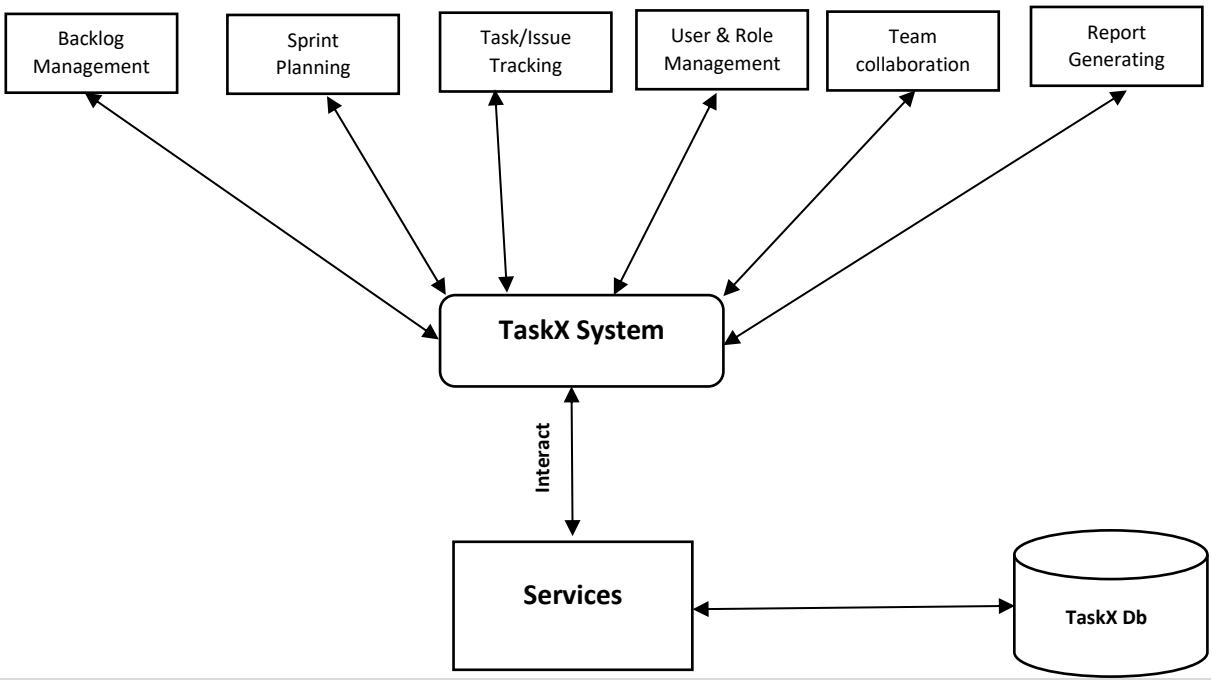


Fig: System Model Of TaskX

Chapter 3: Software Requirements Specification

3.1. Functional Requirements

A Functional Requirement (FR) is a description of the service that the software must offer. It describes a software system or its component. A function is nothing but inputs to the software system, its behavior, and outputs. It can be a calculation, data manipulation, business process, user interaction, or any other specific functionality which defines what function a system is likely to perform. Functional Requirements in Software Engineering are also called Functional Specification**.**

In TaskX functional requirements are the requirements which are mandatory to implement. Functional Requirements of TaskX:

3.1.1Users

Requirement Id	**R1.01.1**
Title	Users
Group	Main Functionality\Users
Description	The system shall support the concept of user. Every user of the system has a username and a password. The username must be unique. In addition, every user has a set of properties: Full Name, Full Business Title (Company Name, Position), E-Mail Address, Phone, Working Address. Each user is uniquely identified by its username within the system
Priority	High
Source	
Risk	
Reference	

3.1.2.User role

Requirement Id	**R1.01.2**
Title	User Role
Group	Main Functionality\Users Roles
Description	The system shall support the concept of **user roles**. The role will vary project to project. So an user will be able to play multiple role by multiple project but one role will have specific feature than

	other. The system feature will be enable and disable based on user role. One project will have one owner , one or more assignee/project manager and one or more assigned user.
Priority	High
Source	
Risk	
Reference	

3.1.3.User Role wise Permission level

User Role	**Is Allowed To**
Project Owner	Browse project list, Create/Delete/View/Update project, Assign/Re-assign a resource to the project . Create/Delete/View/Update Board.
Team Leader	Create/Delete/View/Update task and sprint, Assign/Re-assign a resource to the task, associated to him.
Team Member	View Task and can change the state of their assigned Tasks.

Fig: User Roles

3.1.3. Storing user and User role

Requirement Id	**R1.01.3**
Title	Storing Users and Users Roles
Group	Main Functionality\User Roles\Storage
Description	The system must permanently maintain the list of all users (together with all of their permitted properties), the list of all user roles, and any relationships between users and user roles. The system must have storage capacity. at least five user roles and 200 users, respectively.
Priority	High
Source	
Risk	
Reference	Security Requirements, Performance Requirement, Number of user

3.1.3 User role management

Requirement Id	R1.01.3
Title	User Role Management
Group	Main Functionality\User Roles\Manage
Description	The system shall provide the feature of managing user role. Project Owner will be able to manage user role to the project.
Priority	High
Source	
Risk	
Reference	

3.1.4. User profile

Requirement Id	**R1.01.04**
Title	User Profile
Group	Main Functionality\User Profile
Description	The system shall provide the concept of User Profile. The user profile contains the user-specific configurable parameters of the system. The user profile is associated with one and only one user that is registered in the system (has a user name and a password)
Priority	High
Source	
Risk	
Reference	

3.1.5. User Profile Edit

Requirement Id	**R1.01.05**
Title	User Profile Edite
Group	Main Functionality\User Profile\Edit
Description	The system shall provide the concept of User Profile. The user profile contains the user-specific configurable parameters of the system. The user profile is associated with one and only one user that is registered in the system (has a user name and a password)

Priority	High
Source	
Risk	
Reference	

3.1.6. System Login and Logout

Requirement Id	**R1.01.06**
Title	System Login
Group	Main Functionality\System Login
Description	Any user who wants to use the system his/her have to login the system first. The user need to log in the system by specifying exact username and password. There is no limit of login tries. After a successful login, the system will link the user to their roles and set the GUI's appearance to match their profiles. After logging in, the user becomes an authorized and authenticated user.
Priority	High
Source	
Risk	
Reference	

3.1.7. Show list of project

Requirement Id	**R1.01.07**
Title	Show List of Projects
Group	Main Functionality\Show project List
Description	The system will organize have functionality to show list of assigned and created project by them. List of project should be able to shown by their minimal properties Name, Description, Owner, and Creation Date. Every project is associated through the property Owner with one and only one user. The projects where the user is not performing any role, these project will not be shown to them.
Priority	High
Source	
Risk	
Reference	

3.1.8. Store Project Information

Requirement Id	R1.01.08
Title	Store Project information
Group	Main Functionality\Project\Storage
Description	Every project will have access to permanent storage to store its all necessary information in server permanent file storage and database
Priority	High
Source	
Risk	
Reference	

3.1.9. Filtering Project

Requirement Id	R1.01.09
Title	Filtering Project
Group	Main Functionality\Project\Filter
Description	List of project can be filter by its owner, created date, created by user himself/ herself. The projects where the user is not performing any role, these project will not be shown to them.
Priority	Medium
Source	
Risk	
Reference	

3.1.10. Searching Project

Requirement Id	R1.01.10
Title	Searching Project
Group	Main Functionality\Project\Search
Description	List of project can be search its name
Priority	Medium
Source	
Risk	
Reference	

3.1.11. Creating Project

Requirement Id	R1.01.11
Title	Creating Project
Group	Main Functionality\Project\Create
Description	Any user can create project who are registered to the system. A user who create a project by default he is the project owner. At the time of creating a project users have to give unique project name compare to his created project. A unique project key will be automatically generated by the system, however user can modify the project key but the condition is project key has to be unique within the whole system. List of other project properties like project type, Project Lead, Default Assignee fields has to fill for creating a project
Priority	High
Source	
Risk	
Reference	

3.1.12. Selecting Project

Requirement Id	R1.01.12
Title	Select Project
Group	Main Functionality\Project\Select
Description	From list of project a user can be able to select a project. After selecting a project the system GUI will detect the project as selected project and based the selected project information GUI will be render to the user.
Priority	High
Source	
Risk	
Reference	

3.1.13. Edit Project Configuration

Requirement Id	R1.01.13
Title	Edite Project configuration
Group	Main Functionality\Project\Edit
Description	The projects which the user is in owner role these project configuration can be editable by the user.

	Other role users are not permitted to edit project configuration. Project configuration editable properties will be project name, project key, Project Lead , default assignee.
Priority	High
Source	
Risk	
Reference	

3.1.14. Delete Project

Requirement Id	**R1.01.14**
Title	Delete Project
Group	Main Functionality\Project\Delete
Description	Only project owner will be able to delete the project. Deleting a project will also delete the project from other users who are not owner of the project but had dependency to the project.
Priority	High
Source	
Risk	
Reference	

3.1.15. Manage Project Leader

Requirement Id	**R1.01.15**
Title	Manage Project Leader
Group	Main Functionality\Mange Project Leader
Description	Under the condition that the user has permission “edit project”, the user must be able to assign or re-assign any of available users to the Project Leader property of the project. The user can be associated with any number of projects, but project can be associated only with one user.
Priority	High
Source	
Risk	
Reference	

3.1.16. Make active Project

Requirement Id	R1.01.16
Title	Make active Project
Group	Main Functionality\Project\Current Project
Description	The system will be able to detect current project after selecting a project from project list
Priority	High
Source	
Risk	
Reference	

3.1.17. Project as an root entity

Requirement Id	R1.01.17
Title	Project as an root entity
Group	Main Functionality\Project\Project
Description	The System shall provide the concept of projects. The project will have one or more board. Each board will have one or more sprints. Each sprint will have start and end date. And every sprint will have one or more issues. An issue will have one or more properties associated with them.
Priority	High
Source	
Risk	
Reference	

3.1.18. Create Board

Requirement Id	R1.01.18
Title	Create Board
Group	Main Functionality\Project\Create Board
Description	The project Leader of the project will be able to create one or more board.
Priority	High
Source	
Risk	
Reference	

3.1.19. Project view as a member of the projects

Requirement Id	R1.05.03

Title	Project view as a member of the projects
Group	Main Functionality\Project\Only view
Description	As a assigned member of the project a user can view assigned project but the user won't have edit permission of the project configuration
Priority	High
Source	
Risk	
Reference	

3.1.20. Reporting

Requirement Id	**R1.01.20**
Title	Reporting
Group	Main Functionality\Project\Reporting
Description	The system shall provide the authorized user with permission "create report" the ability to create a various reports on the project. Available report type will vary from user to user.
Priority	High
Source	
Risk	
Reference	

3.1.21. Changing State of issue

Requirement Id	**R1.01.21**
Title	Changing State of issue
Group	Main Functionality\Manage Issue\issue state
Description	Project lead and assigned member will be able to change the state of issue. The state of issue are To do, In progress, To review, Done
Priority	High
Source	
Risk	
Reference	

3.1.22. Comment on issue

Requirement Id	**R1.01.22**
Title	Comment on Issue

Group	Main Functionality\Manage Issue\issue
Description	User will be able to add comment to issues. The user who are assigned to the issue, project lead and owner of the project will be able to comment on the issue
Priority	Medium
Source	
Risk	
Reference	

3.1.23. Add helper link to a issue

Requirement Id	**R1.01.23**
Title	Add helper link to the issue
Group	Main Functionality\Manage Issue\Add helper link
Description	Any authorized user associated with the project will be able to add link on the issue.
Priority	Low
Source	
Risk	
Reference	

3.1.24. Update issue

Requirement Id	**R1.01.24**
Title	Update Issue
Group	Main Functionality\Manage Issue\Update
Description	Project Lead Will be able to update properties of issue Like issue title, issue description, Add issue Link, Update File Link.
Priority	Low
Source	
Risk	
Reference	

3.1.25. Assign developer to an issue

Requirement Id	**R1.01.25**
Title	Assign developer to an issue
Group	Main Functionality\Manage Issue\Update

Description	Project Lead Will be able to update properties of issue Like issue title, issue description, Add issue Link, Update File Link.
Priority	Low
Source	
Risk	
Reference	

3.2. Data Requirements

TaskX data requirements refer to the specific data that is necessary for the project to function properly and meet its objectives. These requirements define the types of data that the software system needs to handle, process, store, and present in order to fulfill its intended functionality.

Data requirements for a software project typically include:

1.Input Data: The data that is required as input to the software system for it to perform its functions. This can include user inputs, external data sources, or data generated by other systems.

2.Output Data: The data that the software system generates as output, which can include reports, calculations, notifications, or data to be shared with external systems or users.

3.Data Structures: The structure and format of the data required by the software system. This includes defining the data fields, data types, and relationships between different data elements.

4.Data Sources: Identifying the specific sources from which the software system will retrieve data. This can include databases, external APIs, files, or data entered manually by users.

5.Data Validation Rules: Defining the rules and constraints that need to be applied to the data to ensure its accuracy, integrity, and consistency. This can include format validation, range checks, and business rules validation.

6.Data Storage: Determining how the data will be stored and organized within the software system. This includes database design, data tables, and storage mechanisms.

7.Data Security: Identifying the security measures and access controls that need to be implemented to protect the data from unauthorized access, modification, or disclosure.

8.Data Integration: Specifying how the software system will integrate and exchange data with other systems or databases, ensuring interoperability and data consistency.

3.3. Performance Requirements

To maintain optimal system performance we have to consider certain requirements which include:

3.3.1. Speed and Latency Requirements

Speed and latency requirements for TaskX project will depend on the specific needs and priorities of the project. However, here are some example speed and latency requirements that could be considered:

1. User Interface Responsiveness:

System should respond to user interactions (e.g., button clicks, form submissions) within 1 second to provide a smooth and responsive user experience.

2. Real-Time Updates:

System should update project information and notifications in real-time or near real-time to ensure that team members have the most up-to-date information.

1. Any delays in updating project data should be kept to a minimum, ideally within a few seconds.

3. Communication and Collaboration:

1. Chat or messaging features should have low latency, allowing team members to exchange messages and collaborate seamlessly.
2. Notifications for new messages or updates should be delivered promptly, preferably within a few seconds.

4. Data Retrieval and Search:

1. Project data should be retrieved quickly to enable efficient browsing, searching, and filtering of project information.
2. Search queries should return results within a few seconds, even for large data sets.

5. Performance under Concurrent Usage:

1. The system should be able to handle multiple simultaneous users without significant performance degradation.
2. Response times for various operations should remain acceptable even when multiple team members are actively using the system concurrently.

6. File Upload and Download:

1. Uploading and downloading project-related files should be efficient and completed within reasonable time frames.
2. Large file transfers should be supported, with upload and download speeds optimized for the size of files commonly used in the project.

7. Integration with External Systems:

1. Integrations with other tools or systems, such as version control systems or bug tracking systems, should operate with minimal latency to ensure smooth data synchronization and workflow.

It's important to note that specific speed and latency requirements may vary depending on the project's size, complexity, and the number of users. These requirements should be defined in collaboration with the project stakeholders and adjusted based on the project's specific needs and priorities.

3.3.2. Precision or Accuracy Requirements

Precision or accuracy requirements for TaskX can be defined to ensure the reliability and effectiveness of the system. Here are some examples of precision or accuracy requirements for the project:

1. **Task Assignment:**
    a. The system should accurately assign tasks to specific team members, ensuring that responsibilities are allocated correctly.
    b. Task assignments should be precise, avoiding any confusion or ambiguity about who is responsible for each task.

2. **Task Status Tracking:**
    a. The system should accurately track the status of tasks, reflecting their current progress and completion.
    b. Task statuses should be precise, providing real-time updates on whether tasks are in progress, completed, or overdue.

3. **Deadlines and Due Dates:**
    a. The system should accurately calculate and display deadlines and due dates for tasks, considering dependencies and resource availability.
    b. Deadlines and due dates should be precise, allowing team members to effectively plan and manage their workloads.

4. **Progress Tracking:**
    a. The system should accurately track the progress of projects, providing insights into the completion status of different project phases or milestones.
    b. Progress tracking should be precise, allowing project managers and team members to monitor project health and make informed decisions.
5. **Reporting and Metrics:**
    a. The system should generate accurate reports and metrics, presenting project data in a precise and reliable manner.
    b. Reports and metrics should be precise, providing meaningful insights into project performance, team productivity, and progress.
6. **Collaboration and Communication:**
    a. The system should ensure accurate and reliable collaboration features, allowing team members to share information, exchange updates, and provide feedback in a precise and timely manner.
    b. Collaboration and communication should be precise, avoiding any misinterpretation or miscommunication among team members.
7. **Data Integrity and Security:**
    a. The system should maintain the integrity and accuracy of project data, ensuring that information is not lost, corrupted, or compromised.
    b. Data storage and security measures should be precise, guaranteeing the accuracy and confidentiality of project-related information.

These precision or accuracy requirements help ensure that TaskX delivers accurate and reliable information to support effective project planning, execution, and tracking. They contribute to maintaining project transparency, facilitating collaboration, and enabling efficient decision-making.

3.3.3. Capacity Requirements

Capacity requirements for TaskX refer to the system's ability to handle the expected workload, user traffic, and data volume effectively. Here are some examples of capacity requirements for the project:

1. **User Capacity:**
    a. The system should support a specific number of concurrent users accessing and interacting with the tool simultaneously.
    b. For example, it should be able to handle 1,000 concurrent users during peak usage periods without significant performance degradation.

2. **Project and Task Volume:**
    a. The system should accommodate a certain number of projects and tasks within its database or storage infrastructure.
    b. It should be able to handle thousands of projects and tens of thousands of tasks without significant impact on system performance.
3. **File Storage and Attachment Capacity:**
    a. The system should provide sufficient storage capacity to handle the expected number and size of file attachments associated with projects and tasks.
    b. For example, it should support file attachments up to 100 MB in size per project or task.

4. **Data Storage and Database Capacity:**
    a. The system's database or data storage infrastructure should have adequate capacity to store and manage the expected volume of project-related data.
    b. It should be able to handle large datasets, such as tens of gigabytes or terabytes of project data, without performance degradation.

5. **Performance and Response Time:**
    a. The system should maintain acceptable performance and response times, even under peak usage or high data loads.
    b. For example, the system should respond to user interactions within 2 seconds and load project information within 3 seconds.
6. **Scalability:**
    a. The system should be designed to scale and accommodate future growth in terms of user base, project volume, and data storage requirements.

b. It should support scalability by utilizing appropriate hardware resources, database optimization, or cloud infrastructure.

7. **System Availability:**
    a. The system should have high availability to ensure that it remains accessible to users and minimizes downtime or service disruptions.
    b. It should target a specified uptime percentage, such as 99.9% availability throughout the year.

These capacity requirements ensure that the project management tool can handle the anticipated workload, data volume, and user traffic without performance issues or system failures. They are important for providing a reliable and efficient platform that can support the needs of project teams, even as the project and user base grow over time.

3.4. Dependability Requirements

The dependability requirements of **TaskX** are of utmost importance to ensure the system's reliability, availability, and resilience. The system needs to be dependable, meaning it consistently performs its intended functions without failures or errors, delivering accurate and trustworthy results. It should be highly available, minimizing downtime and ensuring users can access the system whenever they need it. Fault tolerance mechanisms should be in place to withstand and recover from failures, maintaining critical functionality. Data integrity measures must protect against corruption or unauthorized modification of data. The system should be scalable to accommodate increased usage and user growth without compromising performance or reliability. Robust security measures are essential to safeguard sensitive information. Disaster recovery strategies should be defined to handle catastrophic events. Monitoring and maintenance practices should be implemented to proactively detect issues and ensure ongoing dependability. By meeting these dependability requirements, the system can provide a dependable platform for effective project collaboration, ensuring user confidence and satisfaction.

3.4.1. Reliability Requirements

Reliability requirement for of TaskX aim to ensure that the system functions consistently, reliably, and securely, providing a dependable platform for users to manage their projects effectively.

RIR-1	**The System must keep user data secured**
Description	The system should maintain the integrity of user data by ensuring accurate storage, retrieval, and modification of information. Data should be protected against loss, corruption, or unauthorized access.
Stakeholders	Project Owner, Project Lead, Developers/Testers

RIR-2	**The system should have disaster recovery mechanism**

Description	The system should have backup and disaster recovery mechanisms in place to mitigate the impact of system failures, natural disasters, or other unforeseen events. Regular backups, off-site storage, and recovery procedures should be established to restore the system and data in case of any disruptions
Stakeholders	N/A

3.4.2. Availability Requirements

Availability requirements of TaskX

RAR-1	**The system must be available on 24 X 7**
Description	Our system must be available all day long, every day in a week • The system must be updated regularly • System must be available allowed regions • System must be malware free
Stakeholders	Project Owner, Project Lead, Developers/Testers

3.4.3. Robustness or Fault-Tolerance Requirements

Ensuring the robustness and fault-tolerance of the system is crucial to provide a seamless experience to end users. It is imperative to eliminate any instances of system crashes or failures, aiming for a 0% occurrence rate. By doing so, the system can operate reliably and consistently, minimizing disruptions and maximizing user satisfaction. Additionally, the system should consistently produce accurate results, ensuring that the information or outputs presented to users are precise and reliable. This reliability in both system stability and result accuracy enhances user trust and confidence in the system's capabilities.

RFT-1	**The system handles all user access without system errors**
Description	At any given time, thousands of users could access our application system. Each and every one of their requests must be fulfilled flawlessly.
Stakeholders	N/A

3.4.4. Safety-Critical Requirements

Currently our system doesn’t have any safety critical requirements.

3.5. Maintainability and Supportability Requirements

Maintainability and supportability requirement include:

3.5.1. Maintenance Requirements

TaskX has maintenance requirements, which are the continuing tasks and factors to be taken into account to keep the tool functional, safe, and current. Some common requirements for TaskX:

MR-1	**Bug Fixes and Issue Resolution**
Description	The system should be regularly monitor for bug presence and will need to take necessary steps to solve the problem.
Stakeholders	N/A

MR-2	Security Maintenance
Description	The should be measured security measures regularly and should be followed best practices to protect the system and its data.
Stakeholders	N/A

3.5.2. Supportability Requirements

For the system there are no supportability requirements.

3.5.3. Adaptability Requirements

Adaptability requirements for the project refer to the system's ability to adjust and accommodate changes in user needs, project requirements, and evolving industry trends. Here a example of adaptability requirement:

AR-1	**The system should be Cross-Platform Compatible**
Description	The system should be compatible with various devices and platforms, including desktop computers, laptops, tablets, and mobile devices
Stakeholders	N/A

3.5.4. Scalability or Extensibility Requirements

The ability of the TaskX system to manage progressively heavier workloads, support expanding user bases, and enable the addition of additional features and functionalities is referred to as scalability or extensibility requirements. Below some this type of requirements are given:

SER-1	**The system should support load balancing**
Description	It should support load balancing techniques to evenly distribute user requests and workloads across multiple servers or resources.
Stakeholders	N/A

SER-2	**The system should be modular architecture for easy integration**
Description	The system module will be design separately so that it will be able to integrate with new requirements easily without affecting other modules.
Stakeholders	N/A

3.6. Security Requirements

The standards and controls required to safeguard a software system against unauthorized access, data breaches, and other security risks are referred to as system security requirements. Some security requirements for our TaskX project are discussed below.

3.6.1. Access Requirements

For accessing the application system, there remains some authentication and authorization techniques, every module of the system will provide it. An Example of access requirements given below:

AR-1	**Application will provide role wise feature access mechanism**
Description	Every module of the system will be designed such that for accessing it needs firstly to be a logged in and then user should be into the module as required role.
Stakeholders	Project Owner, project lead, developers/testers

3.6.2. Integrity Requirements

Integrity requirements encompass a security system that guarantees the desired level of data quality. These requirements aim to prevent unauthorized alteration or accidental loss of any data within the system. To fulfill this, we will employ a secure approach to store user passwords by encrypting them. The encryption process renders the passwords in an irreversible format, commonly referred to as hashed passwords. This ensures that even if unauthorized access occurs, the original passwords cannot be deciphered or retrieved. By implementing these integrity requirements, we strengthen the protection of user data, maintaining its integrity and mitigating the risk of malicious tampering or accidental destruction.

3.6.3. Privacy Requirements

Ensuring the privacy of system users is of utmost importance. Privacy requirements play a crucial role in safeguarding the confidentiality of stakeholders' personal information. These requirements dictate that data, whether in its entirety or in part, should only be disclosed in adherence to the system's privacy policy. To uphold privacy standards, it is imperative to implement robust measures to protect the central database. Anonymity should be maintained to prevent unauthorized access and ensure that users' sensitive information remains confidential. To further enhance privacy, a user login system can be implemented, granting users access only to the data associated with their specific accounts. This ensures that individuals can securely access and interact with the relevant data while maintaining the necessary privacy safeguards.

3.7. Usability and Human-Interaction Requirements

Usability and human-interaction requirements for a project refer to the specifications and considerations related to the user experience, ease of use, and effective interaction with the system. Some of these requirements discussed below:

3.7.1. Ease of Use Requirements

Our application should be easy to use and also easily understandable.

EUR-1	Application must be usable for the end user
Description	The system should be easy to use to its user.
Stakeholders	Project lead, Project Owner, Developers/Testers

3.7.2. Personalization and Internationalization Requirements

For my TaskX project personalization and internationalization requirement is not applicable.

3.7.3. Understandability and Politeness Requirements

For my TaskX project understandability and politeness requirements is not requirable.

3.7.4. Accessibility Requirements

AR-1	Application will provide role wise feature access mechanism
Description	Every module of the system will be designed such that for accessing it needs firstly to be a logged in and then user should be into the module as required role.
Stakeholders	Project Owner, project lead, developers/testers

3.7.5. User Documentation Requirements

User documentation requirements in a software project refer to the specific documentation needs of the end users or customers who will be using the software. It aims to provide clear and comprehensive instructions, information, and support materials to assist users in effectively using and understanding the software. Here a user documentation requirements in TaskX project is given:

UDR-1	Getting Started Guide
Description	A comprehensive guide that helps users understand the basic concepts and functionalities of the agile project management system. It should provide step-by-step instructions on creating an account, setting up a project board, inviting team members, and navigating the user interface.
Stakeholders	Project Owner, Project lead, Developers/Testers

3.7.6. Training Requirements

Training requirements for a project are essential to ensure that individuals involved in the project, such as team members, end users, and stakeholders, have the necessary knowledge and skills to effectively utilize and interact with the project deliverables. These training needs may vary depending on the nature of the project and the target audience.

For the TaskX project users need to understand it features, functionalities, and how to use them effectively. So user training is necessary for the project. For different role in a project user will be provided different features and functionality based on user needs. So role wise training may necessary for the user of TaskX. The project is a solution for working with agile scrum framework, so the users who are not familiar with scrum may need to give training on Scrum.

3.8. Look and Feel Requirements

Look and feel requirements mainly refers how the system will look like and how the user interface or graphical user interface of our system will display to the user

3.8.1. Appearance Requirements

Appearance requirements of a project refer to the desired visual and aesthetic aspects that contribute to the overall look and feel of the project deliverables. These requirements focus on the design, layout, and graphical elements that create a visually appealing and cohesive user experience. Here are some common aspects of appearance requirements:

AR-1	**State of the issue can be change by dragging**
Description	For ongoing sprint issues , issues will be draggable for changing their states to do, doing, in review, done by project lead and developers according to their role.
Stakeholders	Project lead, developers/testers

AR-2	**Error message will be in red colored**
Description	Validation message for any action to any form field will be shown in red colored
Stakeholders	Project Owner, Project Lead, Developers/Testers

AR-3	**Label for mandatory field must be in UI form**
Description	The fields which mandatory by business or database logic to a particular activity which is mandatory to place in the UI
Stakeholders	Project Owner, Project Lead, Developers/Testers

3.8.2. Style Requirements

Style requirements for TaskX encompass the specific guidelines and considerations related to the visual style and presentation of the user interface. These requirements focus on creating a consistent and visually appealing design that aligns with the system's branding and enhances the user experience. Here are some style requirements commonly associated with TaskX:

1. Minimalistic Design: The system should adopt a clean and uncluttered design, minimizing unnecessary visual elements and distractions. It should prioritize simplicity and clarity to enhance usability and ease of navigation.

2. Color Scheme: The chosen color scheme should be visually pleasing, easily distinguishable, and harmonize with the branding guidelines. Colors can be used to convey different states, categories, or priorities, providing a visual hierarchy that aids in quickly understanding information.

3. Typography: The system should utilize appropriate and legible fonts for both headings and body text. Consistency in font sizes and styles across different sections and components helps maintain a cohesive visual experience.

4. Iconography: Consistent and intuitive iconography should be employed to represent various actions, features, or objects within the system. Icons should be clear, recognizable, and meaningful to aid in navigation and understanding.

5. Layout and Spacing: The layout should be well-organized and utilize appropriate spacing to provide a balanced and visually appealing interface. Sufficient spacing between elements enhances readability and reduces clutter.

6. Visual Hierarchy: The system should employ visual cues, such as varying font sizes, color contrasts, and element placement, to establish a clear visual hierarchy. Important elements and information should stand out, guiding users' attention and aiding in quick comprehension.

7. Responsive Design: The system should be designed to be responsive, adapting seamlessly to different screen sizes and devices. The layout and visual elements should scale and rearrange appropriately, ensuring a consistent and optimal experience across devices.

8. Consistency: A consistent visual style and design should be maintained across different sections, screens, and components of the system. This includes consistent use of colors, typography, spacing, and visual elements to create a coherent and familiar user experience.

9. User-Friendly Feedback: The system should provide clear and meaningful feedback to user interactions. This can be achieved through appropriate animations, transitions, and visual cues to inform users about system responses, progress, or errors.

For creating TaskX visually appealing and user-friendly interface these requirements is needed to implement properly.

3.9. Operational and Environmental Requirements

Operational and environmental requirements refer to the specifications and considerations related to the operation, performance, and environmental conditions in which the project's software or solution will be used. These requirements focus on ensuring that the system operates effectively, efficiently, and reliably in its intended operational environment. Here are some common operational and environmental requirements:

3.9.1. Expected Physical Environment

There are no expected physical requirements for the TaskX project.

3.9.2. Requirements for Interfacing with Adjacent Systems

Requirements for interfacing with adjacent systems in the system refer to the specifications and capabilities necessary to integrate and exchange data with external systems or tools. Below a requirement for requirement for interfacing with adjacent system:

RIAR-1	**Data Import and Export feature to excel-the system and the system-excel**
Description	The system will be able to import/exports issues to excel/the system
Stakeholders	Project Lead

3.9.3. Projectization Requirements

The specifications and factors needed to make sure the project is properly planned, structured, and carried out as a distinct and manageable activity are referred to as the projectization requirements for the aforementioned project. The structure, responsibilities, procedures, and methodologies needed to manage the project successfully from start to finish are defined by these requirements. As the project is a part of BSSE final year Project it was planned, structured and carried out formal procedures which is discussed in chapter 1.

3.9.4. Release Requirements

Right Now the project doesn't have any release requirements.

3.10. Legal Requirements

Legal requirements for a project refer to the laws, regulations, and legal obligations that must be adhered to throughout the project's lifecycle. These requirements ensure that the project is conducted in compliance with applicable laws and regulations to mitigate legal risks and maintain ethical practices. Legal requirements can vary depending on the nature of the project and the jurisdiction in which it operates. For our project we don't have any requirable legal requirements which is must under consideration and has great importance directly to our system.

3.10.1. Compliance Requirements

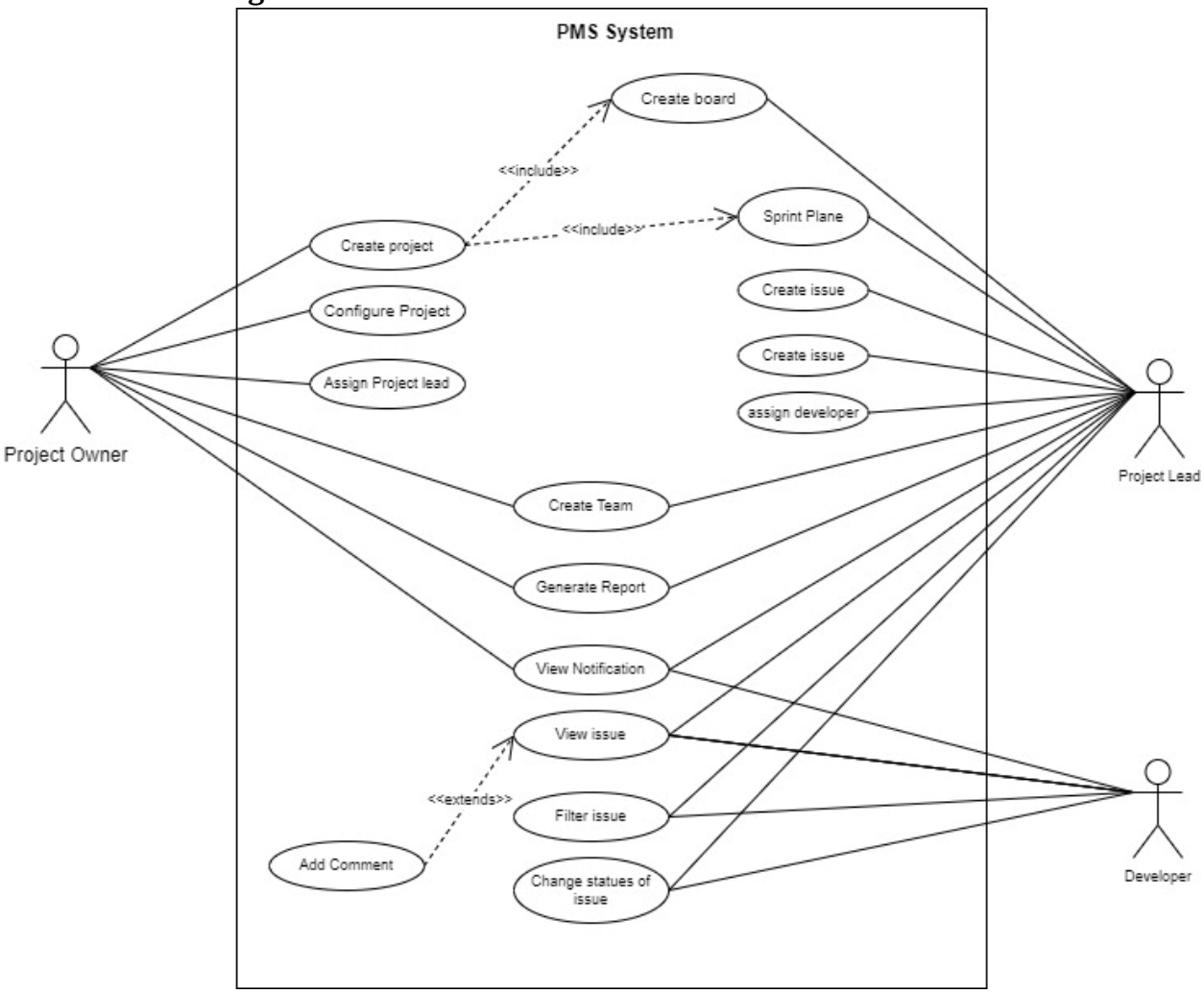
Currently there are no compliance requirements to our system.

3.10.2. Standards Requirements

Currently there are no standard requirements for our project.

Chapter 4: System Analysis

4.1. Use Case Diagram



Use Case Diagram

4.2. Use Case Description

In software and systems engineering, a use case is a list of action or event steps, typically defining the interactions between a role and a system, to achieve a goal while software requirements specification (SRS) is a document that captures complete description about how the system is expected to perform. PMS for agile team use cases description are given below.

4.2.1. Add Project

Use Case 1	Add Project
Goal	Any registered user create project
Preconditions	User have to sign in their account
Success End Condition	User successfully create project
Fail End Condition	User can't successfully create the project

Primary Actor: **Secondary Actor:**	Any registered user of the system	
Trigger	Click on create project button	
Main Success Flow	**Steps**	**Action**
	1.	User will provide input for following field • Project Name • Project Key • Project Category • Project default board Name • Project Assignee/ Lead • Default Assignee
	2	User will click on create button
	3	User go to created project dashboard
Alternative Success Flows	**Step**	**Action**
	1	
Quality Requirements	**Step**	**Action**
	1	
	2	

4.2.2. Create Issue

Use Case 1	Create Issue	
Goal	User create issue to a project sprint	
Preconditions	User have a selected project.	
Success End Condition	User will be able to create board	
Fail End Condition	User can't successfully create issue	
Primary Actor: **Secondary Actor:**	Project Lead	
Trigger	Click on Create issue button	
Main Success Flow	**Steps**	**Action**
	1.	User will provide input for following field • Issue title • Issue sprint • Issue type • Labels • priority
	2	Click on create button
	3	Created issue will be added to chose sprint

Alternative Success Flows	**Step**	**Action**
	1	
Quality Requirements	**Step**	**Action**
	1	
	2	

4.2.3. Create Board

Use Case 1	Create Board	
Goal	User create board for container of multiple sprint	
Preconditions	User have a selected project.	
Success End Condition	User will be able to create board	
Fail End Condition	User can't successfully create the board	
Primary Actor: **Secondary Actor:**	Project Lead, Project owner	
Trigger	Click on Create issue button	
Main Success Flow	**Steps**	**Action**
	1.	User will provide input for following field • Board Name • Board Type
	2	Click on create button
	3	Created board added to under the selected project
Alternative Success Flows	**Step**	**Action**
	1	
Quality Requirements	**Step**	**Action**
	1	
	2	

4.2.4. Configure Project

Use Case 4	Configure Project	
Goal	User edit/update project configuration according to user need	
Preconditions	User select target project to configure	
Success End Condition	User able to change project configuration	
Fail End Condition	User can't successfully change project configuration	
Primary Actor: **Secondary Actor:**	Project owner	
Trigger	Click on target configure project button	
Main Success Flow	**Steps**	**Action**

	1.	User will provide change configuration which his/her need from the following field • Project Name • Project Key • Project Lead • Default Assignee
	2	Click on save button
	3	Configuration will be saved and user will be redirected to list of projects page
Alternative Success Flows	**Step**	**Action**
	1	
Quality Requirements	**Step**	**Action**
	1	
	2	

4.2.5. Assign Project Lead

Use Case	Assign Project Lead	
Goal	User will be able to assign project lead under a project	
Preconditions	User select target project to configure	
Success End Condition	User able to change selected project 'Project Lead'.	
Fail End Condition	User can't successfully change project lead	
Primary Actor: **Secondary Actor:**	Project owner	
Trigger	Click on target configure project button	
Main Success Flow	**Steps**	**Action**
	1.	From list of configure input User will select project lead input
	2	User type user name of project lead which he/she want to assign.
	3	Click on 'save' button
Alternative Success Flows	**Step**	**Action**
	1	
Quality Requirements	**Step**	**Action**
	1	
	2	

4.2.6. Create Team

Use Case	Create Team	
Goal	User will be able to Create team	
Preconditions	1. User open his/her profile 2. User select a project	
Success End Condition	User send create team request to peoples	
Fail End Condition	User can't successfully send create team request to peoples	
Primary Actor: **Secondary Actor:**	Project Lead, Project owner	
Trigger	Click on create Team button	
Main Success Flow	**Steps**	**Action**
	1.	Enter team Name
	2	Add peoples in Invite people to your team field
	3	Click on 'Create team' button
Alternative Success Flows	**Step**	**Action**
	1	
Quality Requirements	**Step**	**Action**
	1	
	2	

4.2.7. Generate Report

Use Case	Generate Report	
Goal	User will be able to generate report	
Preconditions	1. User select project from which he/she want to generate report	
Success End Condition	User download required project report	
Fail End Condition	User can't successfully send create team request to peoples	
Primary Actor: **Secondary Actor:**	Project Lead, Project owner	
Trigger	Click on generate Report button	
Main Success Flow	**Steps**	**Action**
	1.	User Chose required report
	2	Click on 'generate' button
	3	Required report download will be started
Alternative Success Flows	**Step**	**Action**
	1	
Quality Requirements	**Step**	**Action**
	1	

	2	

4.2.8. Assign Developers to An Issue

Use Case	Assign Developer to an issue	
Goal	Project lead will be able to assign developer to an issue	
Preconditions	1. User select a project 2. User go to backlog page	
Success End Condition	Project lead successfully assign developer to an issue	
Fail End Condition	Project lead will not be able to assign to an issue	
Primary Actor: **Secondary Actor:**	Project Lead	
Trigger	Click on a issue	
Main Success Flow	**Steps**	**Action**
	1.	User will click on issue
	2	From list of issue configuration click on assign member
	3	Input one or more developer to an issue
Alternative Success Flows	**Step**	**Action**
	1	
Quality Requirements	**Step**	**Action**
	1	
	2	

4.2.9. View Notification

Use Case	View Notification	
Goal	Use will show their notification	
Preconditions	User has logged in the system	
Success End Condition	User successfully show their notification.	
Fail End Condition	Use can't successfully show their notification	
Primary Actor: **Secondary Actor:**	Project Lead, Project owner, Developers	
Trigger	Click on notification icon	
Main Success Flow	**Steps**	**Action**

	1.	User will show a spinner
	2	User will show at most 8 recent notification in a modal
Alternative Success Flows	**Step**	**Action**
	1	User show a spinner
	2	User will show at most 8 recent notification in a modal
	3	User click on show all notification
	4	User will redirected to all notification page
Quality Requirements	**Step**	**Action**
	1	
	2	

4.3.10. View Issue

Use Case	View issue	
Goal	User will view issue	
Preconditions	1. User is logged in the system 2. User is connected to a project 3. User click on My Work	
Success End Condition	User will successfully view issue.	
Fail End Condition	User will show message 'You have no issue to show'	
Primary Actor: **Secondary Actor:**	Developer, Project Lead	
Trigger	User click on assigned to me	
Main Success Flow	**Steps**	**Action**
	1.	User will see spinner
	2	User will see list of recent assigned issue which is assigned to him/her
Alternative Success Flows	**Step**	**Action**
	1	
Quality Requirements	**Step**	**Action**
	1	
	2	

4.2.11. Filter issues

Use Case 11	Filter issue
Goal	User will be able to filer issue

Preconditions	1. User is in all issues page	
Success End Condition	User will be able to filter issue	
Fail End Condition	User will be shown 'no issues to show'.	
Primary Actor: **Secondary Actor:**	Project Owner, Project lead, developer	
Trigger	Only My issue, recently updated	
Main Success Flow	**Steps**	**Action**
	1.	User will click on any one filter option button
	2	If user click on only my issue he/she will be shown only issues which they are related with.
	3	User will successfully shown filtered issues.
Alternative Success Flows	**Step**	**Action**
	1	
Quality Requirements	**Step**	**Action**
	1	
	2	

4.2.12. Change state of Issue

Use Case	Change state of issue	
Goal	The state of an issue will be change	
Preconditions	1. User is assigned to the issue 2. User is in sprint dashboard	
Success End Condition	User change the state of an	
Fail End Condition	User can't successfully send create team request to peoples	
Primary Actor: **Secondary Actor:**	Developers	
Trigger	Dragging the issue card one state to another state region	
Main Success Flow	**Steps**	**Action**
	1.	User dragged the issue where he/she want to changed the state of an issue
	2	Dragged over the issue on a issue state container like To do, In progress , Done.
	3	Dragged issue card html element will be replaced to the replace container.
	4	Issue state will be change in the database as well.
Alternative Success Flows	**Step**	**Action**
	1	
Quality Requirements	**Step**	**Action**

	1	
	2	

4.3. Activity Diagram

An activity diagram is a visual representation that illustrates the flow of activities, actions, and decision points within a system or process. It depicts the sequential and parallel activities, along with their dependencies, to provide a clear understanding of the system's behavior. Some activity diagram of the project include:

4.3.1. Create project

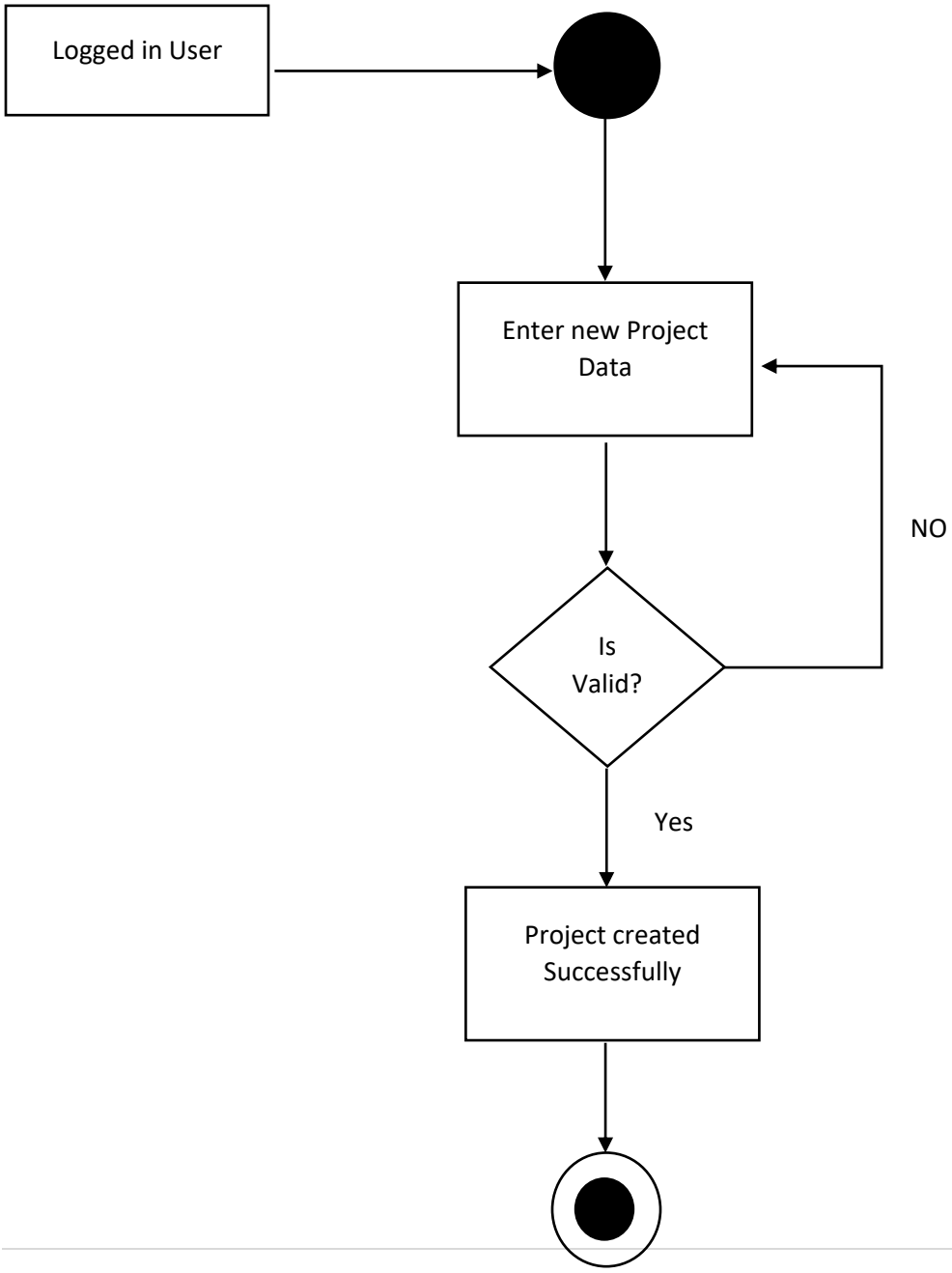


Fig: Create Project

4.3.2. Configure Project

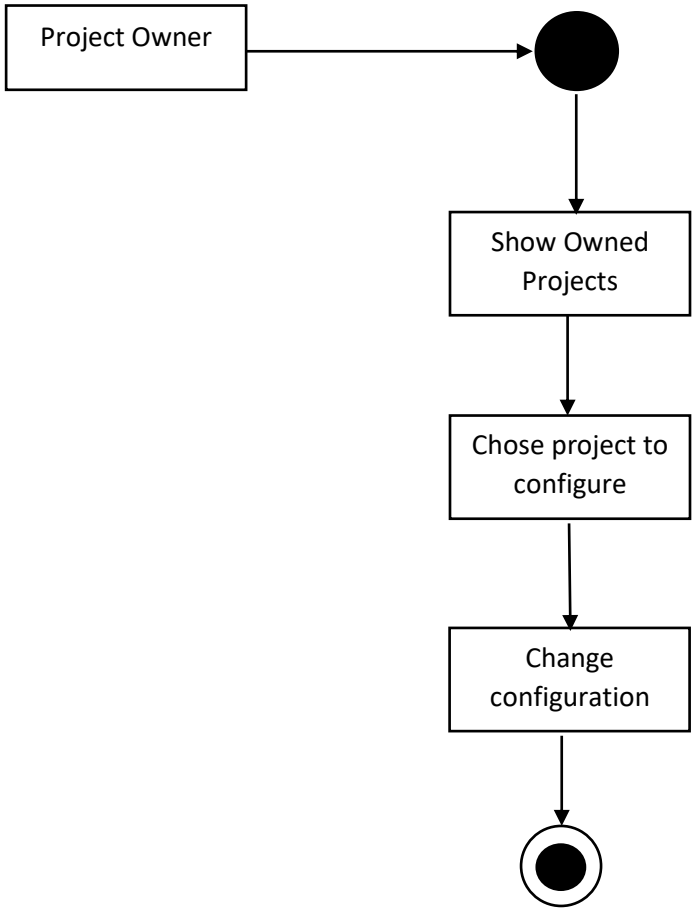


Fig: Configure Project

4.3.3. Assign Project Lead

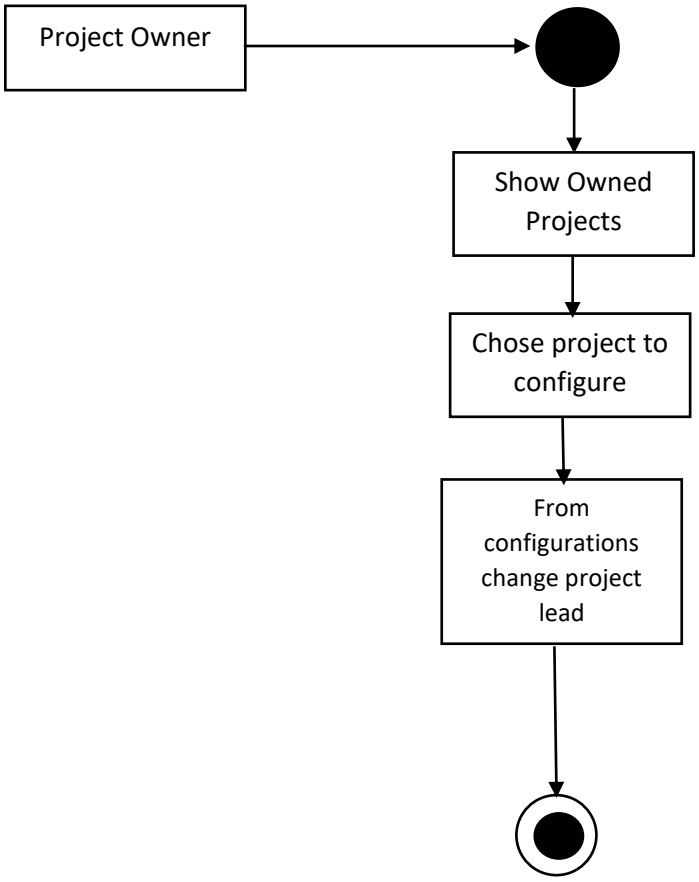


Fig: Assign Project Lead

4.3.4. Create Board

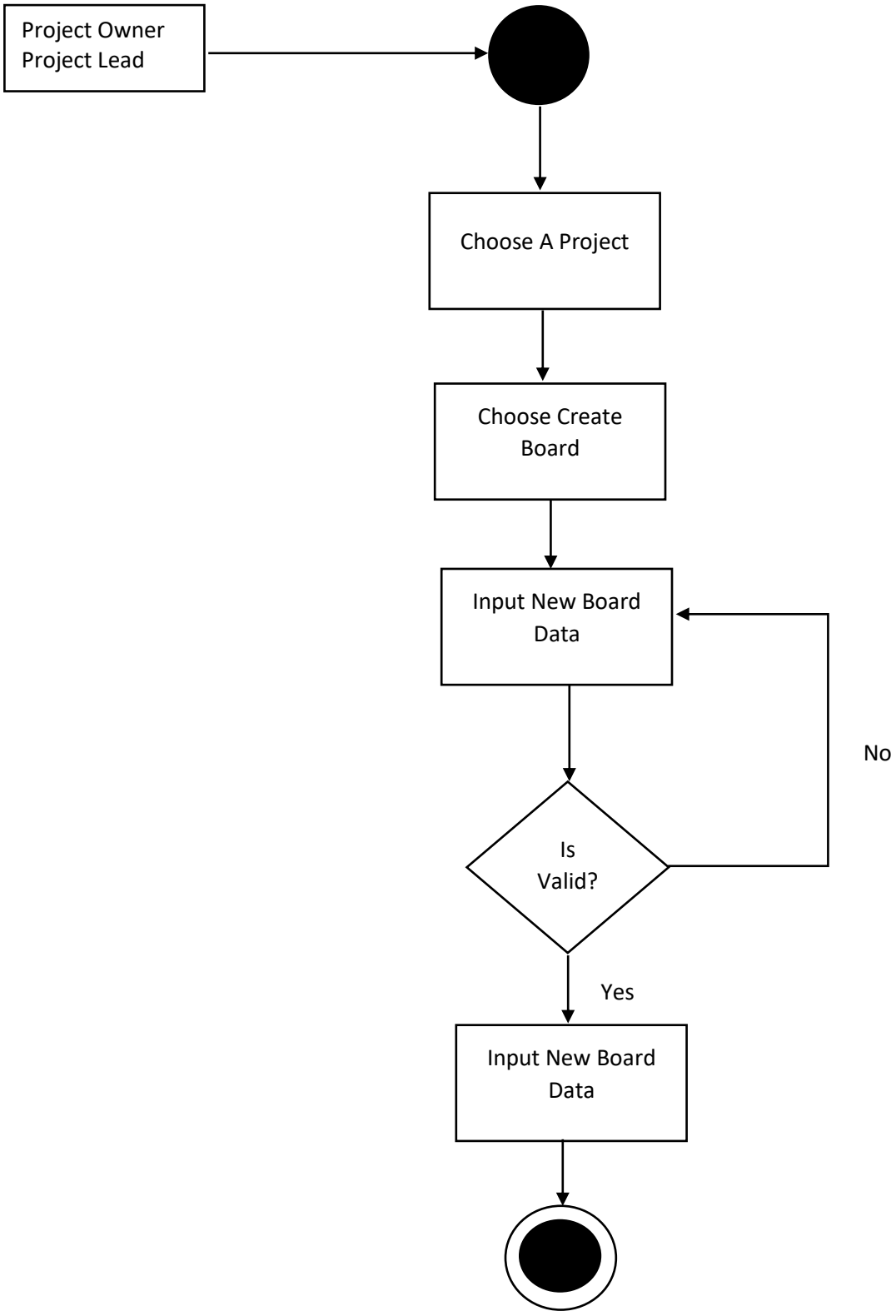


Fig: Create Board

Chapter 5: System Design and Architecture

4.1. System Architecture Overview

The TaskX project, an agile project management system based on agile scrum framework, follows the Model-View-Controller (MVC) architectural design pattern. MVC provides a robust and organized framework for developing the application, separating different aspects of the system's functionality into distinct components.

At the core of the MVC architecture lies the Model, which represents the data and business logic of the TaskX project. It encapsulates the application's data structures, database interactions, and rules for manipulating and processing the data. The Model ensures the integrity and consistency of the project's information, enabling efficient data management and retrieval.

The View component focuses on the user interface (UI) of TaskX. It is responsible for presenting the data from the Model to the users in a visually appealing and intuitive manner. Through the View, users can interact with the system, visualize project boards, cards, and various project management features. The View component is designed to be flexible and adaptable, accommodating different UI elements and layouts to enhance the user experience.

The Controller acts as the intermediary between the Model and the View. It receives user input from the View, interprets and processes it, and triggers appropriate actions in the Model. The Controller handles user interactions, updates the Model accordingly, and notifies the View of any changes. It ensures the separation of concerns by decoupling the UI from the underlying data and logic, promoting modularity and maintainability of the TaskX system.

By adopting the MVC architecture, TaskX benefits from several advantages. Firstly, it enables a clear separation of concerns, facilitating modular development and code reusability. This separation allows me to work on different components concurrently, enhancing productivity and collaboration. Additionally, MVC promotes maintainability and extensibility, as changes made to one component are less likely to affect others.

The MVC architecture also facilitates the scalability of TaskX. As the system grows, new features and functionalities can be added or modified without affecting the existing components. This scalability ensures that TaskX can adapt to the evolving needs of agile project management and accommodate future enhancements seamlessly.

5.2. Class Responsibilities Collaboration (CRC) Cards

Class Responsibilities Collaboration (CRC) cards, a collaborative design method, are used to examine and specify the class collaborations and responsibilities of an object-oriented system. CRC cards, which might be real or virtual index cards, are used to represent the system's classes. Every card explains in depth the responsibilities and cooperation required of the class.

Follow these procedures to use CRC cards in my project:

Identify Classes: Determine the key classes in the project that are essential to the system's functionality. Each class's CRC cards will be different.

Class Name: Write the name of the associated class at the top of each CRC card.

Responsibilities: Establish the responsibilities or tasks that each class must fulfill. List the responsibilities for each act or conduct on the CRC card to indicate what the class is in charge of.

Collaborations: List any additional classes or objects that each class depends on to perform its functions. On the CRC card, make a note of the associations and interactions the class has with other classes or things.

Connections: To demonstrate the connections and dependencies that exist between the classes, draw arrows or links between the CRC cards. Think about how the classes interact and work together to make the system function.

Refinement: Adjust and update the CRC cards as appropriate as you go with the design. Discuss and enhance responsibilities and partnerships in light of team observations and discussions.

The usage of CRC cards ensures that everyone is aware of the structure and behavior of the system and clarifies the roles and connections across classes. The development team may communicate and cooperate more successfully by employing CRC cards in order to produce a system that is well-designed and integrated.

5.3. Detailed Design and Component Selection

Detailed Design and Component Selection is a phase in software project development where the high-level design is translated into detailed technical specifications. It involves breaking down the system into smaller components and selecting the appropriate technologies, frameworks, and libraries to implement each component. This phase focuses on designing the internal structure, interfaces, and interactions of the system components, ensuring they meet the functional and non-functional requirements of the project. In the TaskX project I tried to divide all the functional requirement into multiple component and each component has several related functionality. Here components in low level is a service classes and they implement service contracts (interfaces).

5.4. Database Design Diagram

For Showing the database design clearly, zoom it.

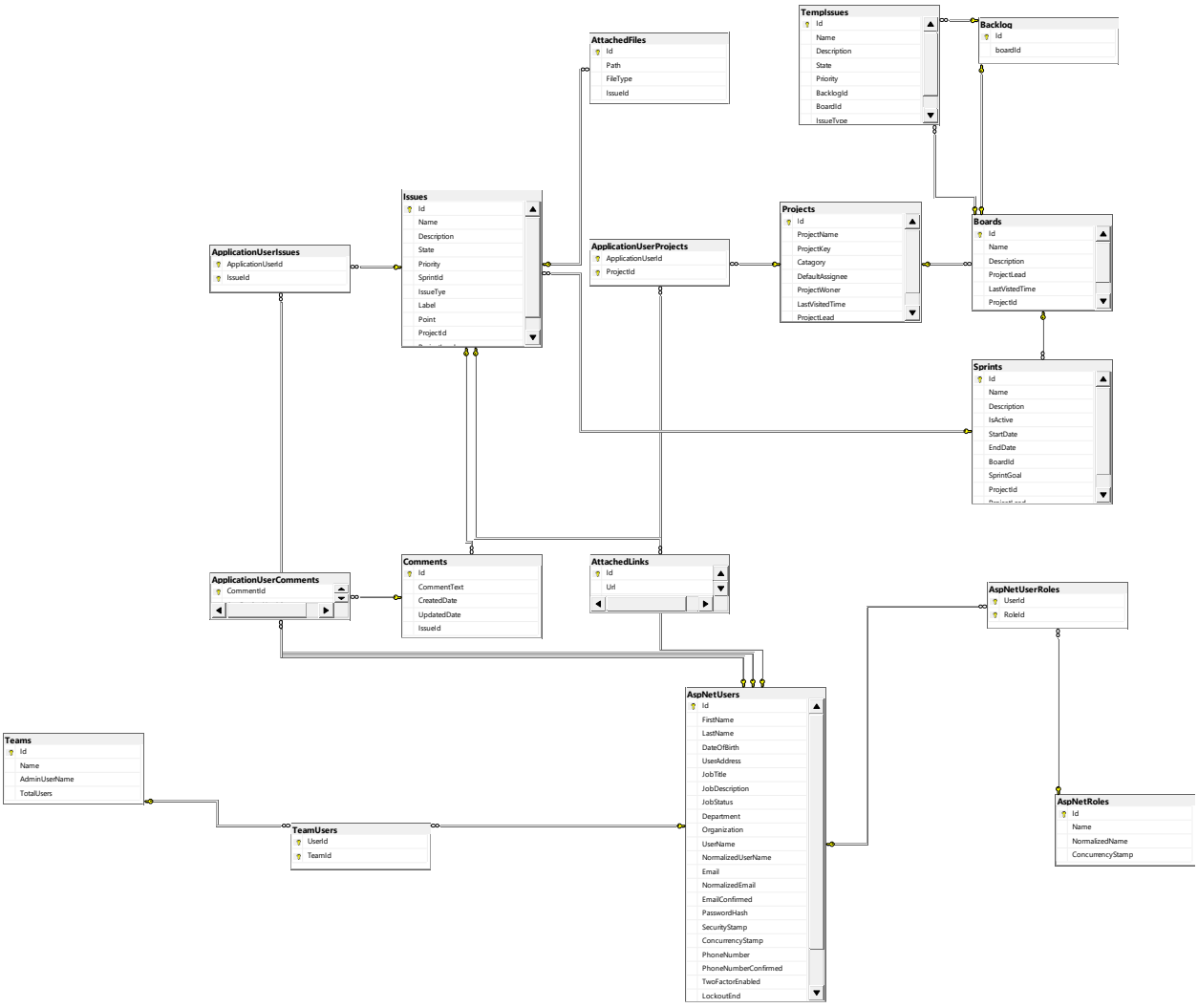


Fig: Database Design Diagram

5.5. User Interface Design

For designing user interface I have used a theme which I collected from online. But designing the whole interface I have to use immense amount of vanilla CSS and JavaScript. For meaningful design I have used good amount of icons which I collected from the downloaded theme and Font Awesome. I have also employed responsive design principles to ensure optimal display across various devices and screen sizes. The User Interface (UI) design of the system incorporates various approaches and tactics to ensure an intuitive and user-friendly interface.

User-Centered Design: Prioritize users' needs, preferences, and behaviors.

Consistency: Maintain a consistent design across the system with uniform colors, fonts, iconography, and layout patterns to create a seamless and recognizable user experience.

Simplicity: Emphasize simplicity and avoid unnecessary complexity by providing clear and concise directions, labeling, and organizing information using grouping and visual hierarchy strategies.

Responsive Design: Ensure the UI adapts to different screen sizes and devices, delivering an optimal viewing experience on computers, tablets, and mobile devices.

Visual Feedback: Enhance user engagement by incorporating visual signals and feedback through appropriate animations, transitions, and hover effects to indicate interactions with the interface.

Accessibility: Prioritize inclusivity by adhering to accessibility rules and standards, such as using appropriate color contrast, providing text alternatives for images, and ensuring keyboard accessibility.

Chapter 6: Implementation and Development

6.1. Development Tools & Technology

Throughout this project I have used different tools and technology.

6.1.1. User Interface Technology

6.1.1.1. ASP.NET Core MVC Framework

TaskX is a comprehensive project management system built primarily using the ASP.NET Core MVC framework. ASP.NET Core MVC serves as the core foundation for TaskX, providing a robust and flexible framework for developing web applications.

6.1.1.2. jQuery

jQuery has been extensively utilized in the TaskX project for client-side DOM manipulation. With its concise syntax and powerful features, jQuery enables efficient handling of HTML elements, event management, and AJAX requests, enhancing the interactivity and responsiveness of the TaskX user interface.

6.1.1.3. Twitter Bootstrap

Twitter Bootstrap has been used as the CSS framework for the TaskX project. With its extensive library of pre-designed components and responsive grid system, Bootstrap facilitates the creation of a visually appealing and mobile-friendly user interface. Its utility classes and JavaScript plugins enhance the styling and functionality, allowing for rapid development and consistent design across different devices for the project.

6.1.1.4. Font Awesome

Font Awesome icons have been utilized in the UI design of the TaskX project. With its extensive collection of scalable vector icons, Font Awesome provides a wide range of visually appealing and customizable icons that enhance the aesthetic appeal and user experience of the application. These icons offer flexibility, easy integration, and cross-browser compatibility, allowing for seamless incorporation into the TaskX interface.

6.1.2. Implementation Tools & Platforms

6.1.2.1. Microsoft Visual Studio 2022

Visual Studio 2022, a powerful integrated development environment (IDE), was utilized during the development of the project. With its advanced features and intuitive interface, Visual Studio 2022 offers a seamless coding experience, facilitating efficient project management, debugging, and deployment. Its robust set of tools and extensions enhances productivity, making it a preferred choice for software development tasks. Specially for .NET developer this IDE most popular because it has many built in features which is only found in Visual Studio.

6.1.2.2. MSSQL Server 2019

MSSQL Server 2019, a widely used relational database management system (RDBMS), was employed during the development of the project. With its robust features, scalability, and reliability, MSSQL Server 2019 provides efficient data storage and management capabilities. It offers advanced security measures, transactional support, and powerful querying capabilities, making it an ideal choice for handling the project's data requirements.

6.1.2.3. .NET Runtime

.NET Core 7 runtime was utilized during the development of the project. As a cross-platform and open-source framework, .NET Core 7 offers a highly performant and scalable environment for developing web applications. It provides a rich set of libraries, language features, and tools for efficient development, deployment, and management. With its ability to run on various platforms, including Windows, Linux, and macOS, .NET Core 7 ensures flexibility and compatibility for the project.

6.2. Coding Practices and Standards

Coding Standards and Practices encompass a set of guidelines and procedures that developers adhere to when writing code. The primary objectives of these standards are to ensure consistency, readability, maintainability, and efficiency within the codebase. In my project, I maintain several guidelines.

Firstly, I follow Naming Conventions by using descriptive and meaningful names for variables, functions, classes, and other code components. Consistency is crucial, so I adopt a specific naming convention such as camel case or snake case.

Secondly, Indentation and Formatting play a significant role in improving code readability and structure. I ensure proper indentation and consistent formatting, paying attention to alignment, line breaks, and spacing.

Thirdly, I emphasize the importance of Comments and Documentation. I add comments throughout the codebase to describe the function, logic, and purpose of the code. I utilize appropriate tools or frameworks to document critical functions, classes, and modules.

Furthermore, I focus on Modularity and Reusability. By breaking down complex code into smaller, modular components, I enhance maintainability and reusability. I apply the principles of abstraction and separation of concerns, encapsulating related functionality within classes or functions.

Error Handling and Exception Handling are also key considerations. I implement robust error management and exception handling procedures to detect and gracefully handle errors. I employ try-catch blocks to manage exceptions and provide detailed error messages.

To enhance Readability of Code, I ensure that the code is easy to understand. I follow a logical flow of execution, avoid lengthy and convoluted code blocks, and use clear and meaningful variable and function names.

Lastly, I prioritize Code Documentation. I document the purpose, inputs, outputs, and usage of functions and methods in the code. I use inline comments, function or method headers, or specialized documentation tools to provide comprehensive documentation.

By adhering to these coding standards and practices, I maintain a consistent and high-quality codebase that is readable, maintainable, and efficient.

6.3. Version Controlling

Using version control, even for a solo project, offers several advantages. For my project, I find version control beneficial for the following reasons:

1. Backups and History: Version control systems maintain a comprehensive record of all modifications made to project files. This allows for easy retrieval of earlier versions, functioning as a backup system and safeguarding against accidental data loss or file damage.

2. Branching and Experimentation: Version control enables the creation of branches to test new features, code improvements, or bug fixes. It allows independent work on different concepts or strategies without impacting the main codebase. If an experiment doesn't work out, branches can be easily discarded and the main code can be reverted to.

3. Collaboration Tools: Even when working alone, version control systems provide collaboration capabilities that can be beneficial. Technologies like pull requests can be used for self-code reviews and to establish a systematic workflow. This improves code quality and helps identify any issues early on.

4. Refactoring and Code Organization: Version control simplifies codebase maintenance and facilitates efficient code restructuring. Small changes can be made, committed, and evaluated for their impact on the overall project. If restructuring doesn't produce the desired outcomes, reverting to a previous version is straightforward.

5. Skill Enhancement and Future Collaboration: Version control systems are widely used in the industry. Mastering them enhances developer skills and prepares for larger projects involving teams or potential future collaborations. Familiarity with popular version control software like Git is valuable for professional growth.

By leveraging version control in my project, I gain benefits such as data backups, branching for experimentation, collaboration capabilities, streamlined code organization, and skill development.

Chapter 7: Testing and Quality Assurance

7.1. Testing Features

The system has certain amount of features and it is also using external system infrastructure to run. So for the system we have to take a feasible decision to perform testing and quality assurance.

7.1.1. Features to be tested

Features to be tested:

1. User Registration and Authentication: Ensure that users can successfully register and log in to the system, and their credentials are properly validated and secured.

2. Task Creation and Management: Verify that users can create, assign, update, and track tasks effectively, including setting due dates, adding descriptions, and assigning team members.

3. Board and List Management: Test the functionality to create and manage boards and lists, including adding, renaming, and deleting boards and lists, and organizing tasks within them.

4. Collaboration and Communication: Validate features such as commenting, tagging team members, and notifications to ensure smooth collaboration and effective communication among users.

5. Search and Filtering: Test the search functionality to ensure users can easily find tasks, boards, or specific information within the system. Validate filtering options to refine and customize views based on user preferences.

6. Integration with Third-Party Tools: If the system supports integration with other tools or platforms, ensure the seamless transfer of data and functionality between them.

7. Reporting and Analytics: Verify that the system generates accurate reports and provides analytics on project progress, task completion, and team performance.

7.1.2. Features not to be tested

Features not to be tested:

1. Third-Party Services: External services or APIs used within the system should not be tested, as their functionality and reliability are beyond the scope of the project itself.

2. Server Infrastructure: The underlying server infrastructure, network configurations, or hosting environment should not be tested, as it is typically managed separately by the hosting provider.

3. Operating System or Browser Compatibility: Compatibility with specific operating systems or browsers should not be tested, as the system should ideally be compatible with a wide range of platforms.

7.2. Testing Strategies

Testing strategies refer to the approach and plan adopted for testing a software system to ensure its quality and effectiveness. It involves defining the scope of testing, selecting appropriate test techniques, determining the test environment, and establishing criteria for test completion. Testing strategies may include various types of testing such as unit testing, integration testing, system testing, performance testing, and usability testing. The strategies aim to uncover defects, validate the system against requirements, and verify its functionality, reliability, and performance. The selection of testing strategies depends on project requirements, complexity, and available resources. Testing strategies of the project elaborately discussed below.

7.2.1. Test Approach

Unit Testing: Test each individual component of the TaskX project, such as task creation, assignment, and status updates, to ensure their proper functionality.

Integration Testing: Test the integration and interactions between different modules of the TaskX project, including task management, user roles, and collaboration features, to ensure seamless data flow and effective communication.

Functional Testing: Evaluate the features and capabilities of the TaskX project, including task assignment, deadline tracking, and progress monitoring, to ensure they meet the desired requirements.

Performance Testing: Assess the performance of the TaskX project under different usage scenarios, ensuring it can handle a large number of tasks, users, and concurrent actions without significant performance degradation.

Usability Testing: Gather user feedback to evaluate the user interface, navigation, and overall user experience of the TaskX project, aiming to provide an intuitive and user-friendly interface.

Security Testing: Verify the security measures implemented in the TaskX project, including secure user authentication, data encryption, and protection against unauthorized access or data breaches.

Thorough test strategies, test cases, and test scripts will be employed to support the testing approach. Identified issues or flaws will be promptly addressed to ensure that TaskX delivers a reliable, user-friendly, and high-performing project management experience.

7.2.2. Pass/Fail Criteria

The success of the TaskX project will be determined based on the following pass/fail criteria. In unit testing, the project will pass if all individual system components undergo their respective tests without critical failures or errors; otherwise, it will fail. For integration testing, a pass indicates seamless data flow and effective communication between modules, while a fail indicates issues hindering functionality. The project will pass functional testing if it meets all defined requirements and performs essential tasks, and it will fail if it falls short. In performance testing, a pass means it performs well under various load levels, while a fail indicates performance issues under load. Usability testing pass/fail is based on positive/negative user feedback, and security testing pass/fail depends on the presence/absence of vulnerabilities. The project will be considered successful if it passes all criteria; otherwise, identified issues must be addressed for success.

7.2.3. Suspension and Resumption

The suspension criteria specify the conditions under which testing operations can be temporarily halted, while the resumption criteria outline when testing can be resumed after being paused.

Suspension: Testing operations will be suspended if there is no internet access during the execution.

Resumption: The execution process will commence once the internet connection is restored.

7.3. Testing Environment (hardware/software requirements)

To ensure accurate and reliable testing of the Student Feedback Review and Automated Question Generate System, specific hardware and software requirements must be met in the testing environment. The system requires a computer with sufficient processing power, memory, and storage capacity to ensure proper functionality. Depending on the intended platform, the computer should have a compatible operating system, such as Windows, macOS, or Linux. A stable internet connection is also necessary for accessing the system and performing online testing tasks.

Since the project is web-based, a web browser is essential for accessing and testing the system. Popular web browsers like Microsoft Edge, Mozilla Firefox, and Google Chrome are compatible with the program. Depending on the technology stack and chosen database system, a suitable database management system (DBMS) may be required to store and retrieve data. The specific DBMS requirements will depend on the chosen technology and database system.

Furthermore, the testing process can be automated using testing frameworks and tools. These may include unit testing frameworks, performance testing tools, and test management systems. The specific testing tool requirements will vary based on the selected technology and testing approaches.

7.4. Test Cases

A test case is a specific set of conditions or actions performed to verify the functionality or behavior of a software system or component. It outlines the steps, inputs, and expected outcomes of a particular test scenario. Test cases are designed to assess whether the system meets the specified requirements and to identify any defects or errors. They serve as a documented guide for testers to execute tests consistently and systematically. Test cases are essential in ensuring the quality and reliability of a software system by providing a structured approach to validate its functionality and performance. Different test cases of the project include:

7.4.1. User Authentication

Test Case	**User Authentication**
Test Scenario	Verify user by user credential
Steps	• Go to login page • Enter valid login credentials (username and password). • Click on the login button.
Result	If user name and password is correct then system should user to redirect to the index page.

7.4.2. Enroll Project by Project Key

Test Case	**Enroll Project by Project Key**
Test Scenario	Developers/Testers can enroll project by project Key
Steps	• From project dropdown menus click ‘enroll project by project key’ button • Enter project key • Click ‘Enroll’ button
Results	If the project key is correct and then project current sprints page will be displayed.

7.4.3. Comment on Issue

Test Case	Comment on Issue
Test Scenario	Any one associated with the issue comment on the issue
Steps	• Select target issue • Go to comment section of the issue • Add comment • Click ‘comment’ button
Result	A new comment should be added at comment section by its commenting user and date time

7.4.4. Team Creating by list of User usernames

Test Case	Team Creating by list of User usernames
Test Scenario	Any user to create team by multiples user
Steps	• From team menus drop down click ‘create team’ button • Then search user field type username after typing system will auto suggest user by username. From list of user select target user • Perform previous steps for adding multiple user • After adding all the user click ‘create team’
Result	A new team should be created and it will redirect to this team index page

7.4.5. Security and Access Control

Test Case	Security and Access Control
Test Scenario	Ensure the system implements proper security measures and access control mechanisms by project associate user role
Steps	• Test user roles and permissions by attempting to access restricted functionalities. • Verify that unauthorized users are denied access to sensitive information or operations
Result	The system should enforce proper security measures and restrict access based on user roles and permissions

7.4.6 User Interface Testing

Test Case	User Interface Testing
Test Scenario	Check the user interface for responsiveness, consistency, and usability.
Steps	• Test the user interface with various screen resolutions and sizes. • Engage many features and functionalities to ensure proper operation. • Verify that the user interface follows established design principles and guidelines.
Result	The user interface should be adaptable to many devices and screen sizes, visually beautiful, and user-friendly.

7.4.7 Import issues from Excel

Test Case	Import Issues from Excel
Test Scenario	User try to import issues from excel
Steps	• From backlog page choose a excel file which has ‘Description’, ‘State’, ’Priority’, ’Issue type’ column with multiple records • Select a sprint to upload issues • Click ‘import’ button
Result	Selected sprint will be updated by adding selected excel file issues

Chapter 8: User Manual

8.1. User Manual

User Manual: TaskX

Welcome to TaskX, an agile project management system designed to streamline your project development process. This user manual will guide developers, project leads, and project owners through the key features and functionalities of TaskX.

For Developers:

1. Account Creation:

- Visit the TaskX website
- Sign up for a new account using your email address or Google account.

2.View Issue:

- Once logged in, Select target project
- Chose board.
- From List of sprints which holding again list of issues chose your target issue.

4. Task Management:

- View and manage your assigned tasks on the Scrum board.
- Update the task status by dragging and dropping cards across the board columns.
- Add comments, attachments, and relevant information to each task card.

5. Collaboration:

- Communicate with the project lead and team members through comments and mentions.
- Attend daily stand-up meetings to provide updates on your progress.
- Use the integrated chat feature for real-time discussions and problem-solving.

For Project Leads:

1. Account Creation:

- Visit the TaskX website
- Sign up for a new account using your email address or Google account.

2. Sprint Planning:

- Collaborate with developers to prioritize backlog items for each sprint.
- Define the sprint goals and communicate them to the team.

- Conduct sprint planning meetings to finalize the sprint backlog.

3. Task Assignment and Monitoring:

- Assign tasks to developers based on their skills and availability.
- Monitor the progress of each task and provide guidance when needed.
- Use the Scrum board to track the overall status of the project.

For Project Owners:

1. Account Creation:

- Visit the TaskX website
- Sign up for a new account using your email address or Google account.

2. Project Creation:

- Once a user logged in click create project button from projects dropdown navigation menu
- Give necessary data
- Click ‘Create project’

3. Assign Project lead :

- Go to projects Page
- From list of projects chose a project to configure
- From configuration Option change project lead
- Click ‘Update’ Button

4. Reporting and Metrics:

- Generate reports on project progress, sprint velocity, and team performance.
- Analyze metrics to identify bottlenecks, improve productivity, and make data-driven decisions.

Remember to explore TaskX's features and functionalities specific to your role. Customize the system to suit your project management needs and enjoy the benefits of streamlined collaboration, enhanced productivity, and efficient project delivery.

Chapter 9: Project Summary

9.1. Github Link

GitHub is a widely used platform for version control and collaboration in software development projects. For controlling version and changes record of TaskX project I used github. For the source code of the project below links is available publicly

Link: https://github.com/suvoislam123/BSSE-Final-Project

9.2. Critical Evolution

The Project Management System (PMS) described in the project overview appears to address a significant need in the software development industry by offering a comprehensive solution for managing software projects. By specifically tailoring the system to implement agile methodology, it acknowledges the growing preference for agile practices in project management and aims to cater to the needs of software development teams.

The primary goal of establishing a structured framework for resource allocation and control aligns well with the fundamental principles of project management. By providing functionality to streamline workflow, the system aims to enhance productivity and efficiency in software development projects.

However, a critical evaluation of the PMS requires a closer examination of its specific features, functionality, and user experience. It is essential to assess whether the system effectively supports agile practices and if it provides adequate flexibility and customization options to accommodate diverse project requirements.

Usability and ease of adoption are crucial factors to consider when evaluating any project management system. The PMS should offer an intuitive user interface, clear navigation, and robust documentation or training resources to support users in understanding and effectively utilizing the system. It should also facilitate collaboration and communication among team members, allowing for seamless sharing of information and updates.

Furthermore, the system's scalability and performance are vital aspects to evaluate. As software development projects vary in size and complexity, the PMS should be capable of handling projects of different scales without compromising its responsiveness or functionality. Additionally, it is crucial to assess the system's ability to integrate with existing software tools and platforms commonly used in the industry, ensuring seamless data transfer and synchronization.

Another critical evaluation aspect is the system's security and data protection measures. Since software development projects often involve sensitive and confidential information, the PMS should implement robust security protocols, such as data encryption, access control, and regular security audits, to safeguard against unauthorized access or data breaches.

Lastly, it is important to consider the system's support and maintenance. Regular updates, bug fixes, and responsive customer support are essential for the ongoing success and usability of the PMS. The development team should demonstrate a commitment to addressing user feedback, continuously improving the system, and providing timely assistance when needed.

9.3. Limitations

The project is mainly focused on agile project management system specially focuses on scrum framework.

The project management system:

- does not provide code management or code storage,
- does not provide version control,
- does not provide Employee time scheduling
- does not provide employee management,
- does not provide work time accounting and payroll.

9.4. Obstacles & Achievements

Obstacles:

- Managing custom dependencies within different teams
- Handling too many inputs fields
- Fitting every project requirement only on Scrum template
- The project was a solo project for this I felt lack of collaboration and knowledge sharing

Throughout the project I have achieved a good amount of skills and knowledge on agile scrum framework. Most importantly I have implement the system by different tools and technologies which helped me to learning new skills.

9.5. Future Scope

The project is mainly on agile scrum framework. In future I have plan to integrate the system to support Kanban board. The System includes the integration of a Kanban board, which will further enhance its capabilities and flexibility. By incorporating the Kanban methodology, the system will enable users to visualize and manage their workflow in a more efficient and streamlined manner. This integration will provide additional options for task management, allowing teams to leverage the benefits of both Scrum and Kanban frameworks within a single system. The introduction of Kanban boards will expand the system's applicability to a broader range of project management approaches, accommodating different project requirements and team preferences. This future enhancement will ensure that the Agile Project Management System remains versatile and adaptable to evolving project management needs.