Software Requirements Specification

For IIT Kanban Board

Prepared By

Md. Raju Mia-MUH2025016M

Israt Jahan Jhumu-BKH2025020F

Md Zahid Hasan-ASH2025012M

Md. Saiful Islam-MUH2025033M

Institute of Information Technology

Noakhali Science & Technology University

<11-06-2023>

Table of Contents

List of Figure

1. Introduction

The introduction of software requirement specification provides an in-depth document that describes what the software does, how will it be performed and includes information about all the functional and non-functional requirements.

Welcome to the groundbreaking IIT Kanban Board Software Project, where innovation meets efficiency in project management and collaboration. In this digital era, we recognize the need to optimize the way teachers and students handle their project work within the IIT community. Our purpose is clear: to revolutionize the existing manual processes and elevate productivity to new heights.

1.1 Purpose

The primary objective of the IIT Kanban Board Software Project is to enhance the efficiency and effectiveness of project work for both teachers and students. By transitioning from manual project handling to a digital platform, we seek to streamline the planning, tracking, and management of tasks. The purpose is to improve collaboration, communication, and productivity among team members.

1.2 Project Scope

The main scope of this project named IIT Kanban Board is to develop a web-based application. As more than 90% people use smartphone or pc and use internet, they can easily use this application by browsing in any web-browser. For this reason, we are targeting to implement our system for web-application.

The scope of this project is vast, encompassing an array of dynamic features that cater specifically to the needs of the IIT community. From creating project folders to assigning members, setting priorities and due dates, sharing and downloading documents, engaging in seamless messaging, and receiving real-time notifications, our software offers an all-encompassing ecosystem for project success.

1.3 References:

1.2 Software Requirements, Third Edition(Karl Wiegers and Joy Beatty),
Click up, Jira, Slack web platform.

1.4 Overview

In overview, the IIT Kanban Board Software Project aims to revolutionize the way teachers and students manage projects within the IIT community. By implementing a digital solution, we aim to enhance collaboration, communication, and productivity. The software will provide a centralized platform for planning, tracking, and managing project work, offering features such as task organization, assignment delegation, document sharing, messaging, and progress tracking. Through this project, we strive to create a more efficient and effective project management environment for the IIT community.

2. User Classes and Characteristics

In the context of the IIT Kanban Board Software Project, let's discuss the user classes and their characteristics. These user classes represent different roles or groups of individuals who will interact with the software. Understanding the characteristics of each user class is crucial for designing an effective and user-friendly system. Here are some possible user classes and their characteristics:

1. **Supervisor:**
    - Characteristics: Experienced professionals in their respective fields, responsible for supervising and guiding students' project work.
    - Needs and Expectations: Teachers require a comprehensive overview of the projects they are overseeing. They need the ability to assign tasks, set priorities, monitor progress, and provide feedback. They may also need to collaborate with other teachers and communicate with students easily.
2. **Project Member:**
    - Characteristics: Enthusiastic learners, actively engaged in project work assigned by their teachers.
    - Needs and Expectations: Students need a clear view of their assigned tasks, including priorities and deadlines. They require the ability to update task statuses, collaborate with team members, access project-related documents, and seek guidance from their teachers.

3. **Collaborators/Team Members:**
    - Characteristics: Individuals working together on a project, collaborating and contributing to its successful completion.
    - Needs and Expectations: Collaborators need a platform to view and update project tasks, communicate with team members, and share project-related documents. They may require task assignment notifications, reminders, and a way to track progress collectively.

4. **Administrators:**
    - Characteristics: Individuals responsible for system administration, user management, and ensuring the overall smooth functioning of the software.
    - Needs and Expectations: Administrators need access to system configuration settings, user management capabilities, and the ability to troubleshoot issues. They also require control over user permissions and security settings.

For this project, the top management team could include the following roles:

2.1 Top Management: The project manager would be responsible for overseeing the development and implementation of the app. They would work with the development team to ensure that the app is being built according to the project plan and that it meets the needs of the users. They would also be responsible for communicating with stakeholders and making sure that the project stays on track and within budget.

2.2 Development Team Lead: The development team lead would be responsible for managing the team of developers working on the app. They would assign tasks, ensure that the team is working efficiently, and help to resolve any technical issues that arise.

2.3 User Experience (UX) Designer: The UX designer would be responsible for designing the user interface and user experience of the app. They would work with the development team to ensure that the app is easy to use and navigate for the users.

2.4 Quality Assurance (QA) Tester: The QA tester would be responsible for testing the app to ensure that it is functioning properly and meeting the desired quality standards. They would work with the development team to identify and fix any bugs or issues with the app.

3. Design and Implementation Constraints

Design and implementation constraints are those that we have used to implement this project make successful. It also describes tool that enables developers and testers to view and interact with the user interface (UI) elements of this application.

3.1 User Interface Technology

User interface (UI) is everything designed into a system view that which person's associates with this system may like the interface of this system.

3.1.1 Programming Language:

JavaScript: JavaScript is a popular language for building web applications, and it is often used in conjunction with HTML and CSS to create the front-end of an app. JavaScript is known for its flexibility and versatility, and it is a good choice for building interactive and dynamic apps.

Java: Java is a popular language for building Android mobile apps, as it is the primary language supported by the Android operating system. Java is a powerful and versatile language, and it is widely used in many different types of projects.

3.1.2 JavaScript and jQuery Library

The most common use of JavaScript is to add client-side behavior to HTML pages, also known as Dynamic HTML (DHTML). Scripts are embedded in or included from HTML pages and interact with the Document Object Model (DOM) of the page. jQuery is a JavaScript library. jQuery greatly simplifies JavaScript programming. jQuery UI is a curated set of user interface interactions, effects, widgets, and themes built on top of the jQuery JavaScript Library. Whether you're building highly interactive web applications or you just need to add a date picker to a form control, jQuery UI is the perfect choice. jQuery UI is built for designers and developers alike. We've designed all of our plug-ins to get you up and running quickly while being flexible enough to evolve with your needs.

3.1.3 CSS Framework

CSS is a language that describes the style of an HTML document. CSS describes how HTML elements should be displayed. Build responsive, mobile-first projects on the web with the world's most popular front-end component library. Bootstrap is an open source toolkit for developing with HTML, CSS, and JS. Quickly prototype your ideas or build your entire app with our Sass variables and mix INS, responsive grid system, extensive pre built components, and powerful plug-ins built on jQuery. The bootstrap code is included minified, which means that white spaces are removed to make the file size smaller and therefore make the load time faster for the file which improves the load time for the whole page. The main design that bootstraps ads without specifically adding design to elements is that when hovering over a link. This is fixed with some simple CSS code added to the CSS-file, unless the bootstrap CSS-file is included after the original, then bootstrap will override the custom ones and the changes will not be seen. Having some basic knowledge about how Bootstrap works before starting to use it would increase the efficiency and speed one might achieve the goal one has in mind for including bootstrap into the project.

3.2 Implemented Tools and Platform

When implementing the IIT Kanban Board Software Project, several tools and platforms can be utilized to facilitate the development and deployment process. Here are some commonly implemented tools and platforms:

1. Programming Languages and Frameworks: Depending on the requirements and expertise of the development team, programming languages such as Python, JavaScript, or Java can

be used to build the software. Frameworks like Django, React, or Angular can provide a structured development environment and help expedite the development process.

2. Version Control System: A version control system, such as Git, enables collaborative development and ensures efficient management of source code. It allows multiple developers to work on the project simultaneously, tracks changes, and facilitates easy collaboration and code integration.
3. Integrated Development Environment (IDE): IDEs such as Visual Studio Code, PyCharm, or IntelliJ IDEA provide a comprehensive development environment with features like code auto-completion, debugging tools, and version control integration. These tools enhance productivity and streamline the development process.
4. Project Management and Collaboration Tools: Project management and collaboration tools like Jira, Trello, or Asana can aid in organizing tasks, tracking progress, assigning responsibilities, and facilitating effective communication among team members. These tools enable seamless collaboration and help manage project timelines and milestones.
5. Database Management System: A database management system, such as MySQL, PostgreSQL, or MongoDB, is essential for storing and retrieving project data. The choice of the database depends on factors like data structure, scalability, and performance requirements.
6. Cloud Platforms: Cloud platforms like Amazon Web Services (AWS), Microsoft Azure, or Google Cloud Platform (GCP) offer scalable infrastructure, storage, and hosting solutions. These platforms provide flexibility, scalability, and reliability for hosting the Kanban board software, ensuring its availability and performance.
7. Testing and Quality Assurance Tools: Testing and quality assurance tools, such as Selenium, JUnit, or Jest, can be used to automate testing processes and ensure software quality. These tools assist in unit testing, integration testing, and regression testing to identify and resolve software defects.
8. User Interface (UI) and User Experience (UX) Design Tools: UI/UX design tools like Adobe XD, Sketch, or Figma help create visually appealing and user-friendly interfaces. These tools assist in prototyping, wireframing, and designing intuitive user interfaces that enhance the user experience.

It is important to select tools and platforms based on the project's specific requirements, the development team's expertise, and compatibility with the desired features and functionalities of the Kanban board software. Effective utilization of these tools and platforms can significantly contribute to the successful implementation and deployment of the software project.

3.3 Web Server

When implementing the IIT Kanban Board Software Project, a web server plays a crucial role in hosting and delivering the software application over the internet. A web server handles client requests, processes them, and serves the requested web pages or resources. Here are some commonly used web servers:

1. Apache HTTP Server: Apache HTTP Server, often referred to as Apache, is one of the most widely used open-source web servers. It is known for its stability, performance, and extensive module support. Apache is compatible with various operating systems and can handle a large number of concurrent connections.
2. Nginx: Nginx (pronounced "engine-x") is a high-performance, lightweight, and scalable web server. It is renowned for its efficiency in handling concurrent connections and serving static content. Nginx is often used as a reverse proxy or load balancer in combination with other web servers or application servers.
3. Microsoft Internet Information Services (IIS): Microsoft IIS is a web server specifically designed for Windows environments. It provides robust performance, security features, and seamless integration with other Microsoft technologies. IIS supports various protocols and can host ASP.NET applications.
4. Node.js: Node.js is not a traditional web server but a JavaScript runtime built on the V8 engine. It allows developers to build scalable and high-performance server-side applications using JavaScript. With frameworks like Express, developers can create web servers and APIs using Node.js.
5. Lighttpd: Lighttpd (pronounced "lighty") is a lightweight and efficient web server that focuses on speed and low memory footprint. It is suitable for serving static content and handling a large number of concurrent connections. Lighttpd is often used in resource-constrained environments.

The choice of a web server depends on various factors such as performance requirements, scalability needs, compatibility with programming languages and frameworks, and the development team's familiarity with the server. It is crucial to select a web server that aligns with the project's specific requirements and provides a reliable and efficient platform for hosting the IIT Kanban Board Software.

3.4 Database Server

We will use MySQL database server to store all of the information of this system. The reason behind to choose the database server are given below:

- Security
- Reporting and Data Mining
- Fault tolerance
- Performance diagnostics

4. Requirement Specification

Requirement specification is the process of identifying and documenting the specific requirements that the Easy Food Tracker app needs to meet. It is an important step in the software development process as it helps to ensure that the final product meets the needs of the users and stakeholders.

Here are some examples of requirements that might be included in the requirement specification for this project:

4.1 Functional requirements

These requirements specify the specific functionality that the app needs to provide, such as the ability to user registration and Authentication, Project creation and management, Task Management, Document Sharing and Collaboration.

Functional requirements are the specific functionality that the IIT Kanban Board web-application needs to provide to meet the needs of the users and stakeholders. Here are some examples of functional requirements that might be included in this project:

4.1.1 Enter the System

FR-1	User registration/signup.		
Description	Users should be able to sign-up.		
Stakeholders	Project Member, Supervisor	**Priority**	High

4.1.2 Invite Members

FR-2	**Project invitation**		
Description	Users should be able to invite others to projects.		
Stakeholders	Project Member, Supervisor	**Priority**	High

4.1.3 Create a Project Folder

FR-3	**Create a Project Folder**		
Description	Users should be able to create new project folders		
Stakeholders	Project Member, Supervisor	**Priority**	High

4.1.4 Delete a Project Folder

FR-4	**Delete a Project Folder**		
Description	Users should be able to delete new project folders		
Stakeholders	Project Member, Supervisor	**Priority**	High

4.1.5 Create List

FR-5	**Create List**		
Description	Users should be able to create List.		
Stakeholders	Project Member, Supervisor	**Priority**	High

4.1.6 To Do

FR-6	**To Do**		
Description	A feature for Adding Task or Create Task.		
Stakeholders	Project Member, Supervisor	**Priority**	High

4.1.7 Add Task

FR-7	**Add Task**		
Description	Users should be able to add Task		
Stakeholders	Project Member, Supervisor	**Priority**	High

4.1.8 Delete Task

FR-8	**Delete Task**		
Description	Users should be able to delete Task.		
Stakeholders	Project Member, Supervisor	**Priority**	High

4.1.9 Assign Member

FR-9	**Assign Member**		
Description	Users should be able to assign member for each.		
Stakeholders	Project Member, Supervisor	**Priority**	High

4.1.10 Share Docs

FR-10	**Share Docs**		
Description	Users should be able to upload project documents and share it.		
Stakeholders	Project Member, Supervisor	**Priority**	Medium

4.1.11 Commenting

FR-11	Commenting on tasks/documents.		
Description	Users should be able to comment on tasks/documents		
Stakeholders	Project Member, Supervisor	**Priority**	Medium

4.1.12 Get Notifications

FR-12	**Get Notifications**		
Description	Users should receive notifications for updates		
Stakeholders	Project Member, Supervisor	**Priority**	High

4.1.13 Reporting and analytics

FR-13	Reporting and analytics		
Description	Managers/Admins should generate project reports		
Stakeholders	Project Member, Supervisor	**Priority**	High

4.2 Non-Functional Requirement

These requirements specify the quality characteristics that the app needs to possess such as performance, security, and usability.

4.2.1 Performance

NFR-1	Performance		
Description	The application should load quickly and be responsive to its user interactions.		
Stakeholders	Project Member, Supervisor	**Priority**	High

4.2.2 Scalability

NFR-2	Scalability		
Description	The application should be able to handle a high number of concurrent users.		
Stakeholders	Project Member, Supervisor /Administrative	**Priority**	High

4.2.3 Security

NFR-3	Security		
Description	The application should protect user data and prevent unauthorized access.		
Stakeholders	Project Member, Supervisor /Administrative	**Priority**	High

4.2.4 Usability

NFR-4	Usability		
Description	The application should be easy to navigate and use for all users, including those with disabilities.		
Stakeholders	Project Member, Supervisor	**Priority**	High

4.2.5 Accessibility

NFR-5	Accessibility		
Description	The application should meet accessibility standards and guidelines, to make it available to as many people as possible.		
Stakeholders	Project Member, Supervisor	**Priority**	Medium

4.2.6 Compatibility

NFR-6	Compatibility		
Description	The application should work on different devices and browsers		
Stakeholders	Project Member, Supervisor	**Priority**	Medium

4.2.7 Maintainability

NFR-7	Maintainability		
Description	The code should be well-organized and easy to maintain.		
Stakeholders	Development Team/Administrator	**Priority**	Medium

4.2.8 Backup and Recovery

NFR-8	Backup and Recovery		
Description	Regular backups should be performed, and data recovery ensured		
Stakeholders	Project Member, Supervisor, Administrator	**Priority**	Low

4.3 Usability requirements

These requirements specify how easy the app should be to use and navigate, and how accessible it should be to users with different abilities.

4.3.1 Simple and Intuitive navigation

UR-1	Simple and Intuitive navigation		
Description	The application should have a simple, consistent and easy-to-use navigation structure.		
Stakeholders	Project Member, Supervisor	**Priority**	High

4.3.2 Clear and consistent visual design

UR-2	Clear and consistent visual design		
Description	The application should have a clear and consistent visual design, which helps users understand the app's structure and organization.		
Stakeholders	Project Member, Supervisor	**Priority**	High

4.3.3 Accessible for all

UR-3	Accessible for all		
Description	The application should be designed to meet accessibility guidelines and standards, to make it available to as many people as possible.		
Stakeholders	Project Member, Supervisor	**Priority**	High

4.3.5 Error Prevention and Recovery

UR-5	Error Prevention and Recovery		
Description	The application should be designed to prevent errors, and provide clear and concise feedback when errors occur, helping users recover from those errors.		
Stakeholders	Administrator	**Priority**	High

4.4 Security requirement

These requirements specify the measures that need to be taken to protect the app and its users' data, such as data encryption, user authentication, and access controls.

4.4.1 User Authentication

SR-1	User Authentication		
Description	Users should be required to provide a valid username and password in order to access their account.		
Stakeholders	System	**Priority**	High

4.4.2 Access control

SR-3	Access control		
Description	The application should restrict access to sensitive data and functionality based on the user's role and authorization level.		
Stakeholders	Administrator	**Priority**	High

4.4.3 Secure Transmission

SR-4	Secure Transmission		
Description	Data transmission between the user's device and the server should be secured using HTTPS.		
Stakeholders	Administrator	**Priority**	High

4.4.4 Regular Software updates

SR-6	Regular Software updates		
Description	The application should be regularly updated with the latest security patches and fixes.		
Stakeholders	Administrator	**Priority**	High

4.4.5 Vulnerability scans

SR-7	Vulnerability scans		
Description	The application should be regularly scanned for vulnerabilities and any issues found should be addressed promptly.		
Stakeholders	Administrator	**Priority**	High

4.5 Performance requirements

These requirements specify the desired level of responsiveness and speed of the app, as well as the number of concurrent users it should be able to handle.

4.5.1 Load time

PR-1	Load time		
Description	The application should have a fast load time, ideally under a few seconds.		
Stakeholders	Project Member, Supervisor	**Priority**	High

4.5.2 Responsiveness

PR-2	Responsiveness		
Description	The application should be responsive to user interactions and provide instant feedback for actions taken.		
Stakeholders	Project Member, Supervisor	**Priority**	High

4.5.3 Server Response Time

PR-3	Server response time		
Description	The server should provide a response time under a few seconds when handling requests from the application.		
Stakeholders	Project Member, Supervisor	**Priority**	High

4.5.4 Concurrent users

PR-4	Concurrent time		
Description	The application should be able to handle a high number of concurrent users without any significant performance degradation.		
Stakeholders	Project Member, Supervisor	**Priority**	High

4.5.5 Scalability

PR-5	Scalability		
Description	The application should be able to scale to accommodate increased traffic or user numbers.		
Stakeholders	Administrator	**Priority**	High

4.5.6 Optimization

PR-6	Optimization		
Description	The application should be optimized for performance, including techniques like caching, magnification and code splitting.		
Stakeholders	Administrator	**Priority**	High

Performance requirements are important because they determine how well the app will perform under different conditions and how well it can handle high traffic. It's essential that the app is responsive, fast and can handle a high number of concurrent users, as slow loading times and unresponsive interactions can discourage users from using the app. It's also important that the app is designed with scalability in mind, so it can be easily adapted to accommodate increased traffic or user numbers in the future.

4.6 Technical requirements

These requirements specify the technical constraints and limitations that the app needs to meet, such as the need to work offline, or to work on different browsers and devices.

4.6.1 Web Platform

TR-1	Web Platform		
Description	The application will be a web-based platform that can be accessed through a browser on desktop and mobile devices.		
Stakeholders	Project Member, Supervisor	**Priority**	High

4.6.2 Database

TR-2	Database		
Description	A database will be used to store information about restaurants, menu items, prices, and user ratings.		
Stakeholders	Project Member, Supervisor	**Priority**	High

4.6.3 User Authentication

TR-3	User Authentication		
Description	The application will have user authentication features to allow users to log in and provide ratings		
Stakeholders	Project Member, Supervisor, System	**Priority**	High

4.6.4 Responsive Design

TR-4	Responsive Design		
Description	The application will have a responsive design to ensure a good user experience on different devices.		
Stakeholders	Project Member, Supervisor, Administrator	**Priority**	High

4.7 Speed and Latency Requirements

The speed and latency requirements of the IIT Kanban Board Software Project are crucial to ensure a responsive and efficient user experience. Here are some considerations for speed and latency:

1. Response Time: The system should provide quick response times when users interact with various features, such as creating tasks, updating task status, and generating reports. The response time should be minimal to avoid any noticeable delays and keep users engaged.
2. Task Loading: The time taken to load tasks, project boards, and associated data should be minimized to provide a seamless and smooth user experience. Users should be able to access and view their project information without significant delays.
3. Real-Time Updates: If the software offers real-time updates, such as task assignments or progress updates, the latency should be kept low. Users should receive prompt notifications and updates to maintain effective collaboration and project tracking.
4. File Uploads and Downloads: When users upload or download documents and files within the software, the transfer speeds should be optimized to ensure efficient file handling. Large files should be processed quickly to avoid unnecessary wait times.
5. Synchronization: If the software is accessible across multiple devices or platforms, any synchronization processes should occur promptly and seamlessly. Changes made on one device should be reflected in real-time on other devices to maintain data consistency.
6. Search Functionality: The search feature should provide quick results as users search for projects, tasks, or specific information within the software. The search algorithm should be optimized to deliver relevant results promptly.
7. Performance under Load: The software should be capable of handling a significant number of users and data without a noticeable decrease in performance. It should be scalable to ensure that the system can handle increased user activity and data volume without compromising speed and latency.

4.8 Precision and Accuracy Requirements

The precision and accuracy requirements of the IIT Kanban Board Software Project are essential to ensure that the information and calculations within the system are reliable and error-free. Here are some considerations for precision and accuracy:

1. Task and Project Data: The software should accurately capture and store task and project data, including task names, descriptions, due dates, assigned members, and status updates. Any modifications or updates made to the data should be reflected accurately in the system.
2. Calculations and Metrics: If the software includes calculations or metrics, such as project progress, task completion percentages, or performance indicators, they should be

calculated accurately based on the relevant data. The calculations should follow appropriate mathematical rules and formulas to ensure precision.

3. Notifications and Reminders: Any notifications or reminders sent by the software, such as task deadlines or upcoming events, should be accurate and timely. Users should receive notifications at the correct time and for the appropriate tasks or events.
4. Data Integrity: The software should maintain data integrity by preventing data corruption, loss, or unauthorized modifications. It should have appropriate measures in place to ensure that data remains accurate and consistent throughout its lifecycle.
5. User Input Validation: When users provide input, such as task descriptions or due dates, the software should validate the input to ensure its accuracy and prevent any potential errors. This includes checking for proper formatting, valid dates, and appropriate data types.
6. Reporting and Analytics: If the software includes reporting and analytics features, the generated reports and analytics should be accurate and reflect the underlying data accurately. Users should be able to rely on the information provided by the system for decision-making purposes.
7. Data Import and Export: When importing or exporting data from external sources, the software should accurately interpret and process the data. This ensures that data transferred into or out of the system remains accurate and maintains its integrity.

4.9 Capacity Requirements

The capacity requirements of the IIT Kanban Board Software Project refer to the system's ability to handle and accommodate a certain volume of users, data, and concurrent operations. Here are some considerations for capacity requirements:

1. User Capacity: The software should be able to handle the expected number of users who will be accessing and using the system simultaneously. This includes both teachers and students who will be collaborating on projects. The system should be designed to support the anticipated user load without significant performance degradation.
2. Data Volume: The software should be capable of managing a substantial amount of data related to projects, tasks, user profiles, and any associated documents or files. The database and storage infrastructure should be able to handle the expected data volume efficiently and provide quick access to information.
3. Concurrent Operations: The system should be able to handle multiple concurrent operations performed by different users without experiencing performance issues or delays. This includes tasks such as creating projects, adding tasks, updating statuses, assigning members, and generating reports.

4. Scalability: The software should be designed with scalability in mind to accommodate future growth and increasing demands. It should be able to scale up or scale out as needed, allowing for additional users, data, and concurrent operations without compromising performance.
5. Performance under Load: The system should be tested under simulated load conditions to ensure that it can handle peak usage periods without experiencing significant performance degradation. Load testing helps identify any bottlenecks or limitations in terms of processing power, memory, or network capacity.
6. Bandwidth Considerations: If the software involves file uploads, downloads, or real-time collaboration features, the available network bandwidth should be considered. The system should be optimized to efficiently utilize the available bandwidth without causing delays or disruptions.
7. Storage Capacity: The storage infrastructure should have sufficient capacity to accommodate the anticipated growth of data over time. This includes considering both database storage and any associated document or file storage requirements.

4.10 Dependability Requirements

- High reliability to ensure consistent and accurate performance of the IIT Kanban Board Software Project
- Availability measures to minimize system downtime and ensure continuous accessibility for users
- Fault tolerance mechanisms to handle errors, failures, and unexpected events without compromising system functionality
- Data integrity and security measures to protect user data and ensure its confidentiality, integrity, and availability
- Robust error handling and logging capabilities to detect, report, and handle any errors or exceptions that occur
- Disaster recovery plan to ensure prompt system recovery in the event of a major failure or disaster
- Performance monitoring to continuously monitor system performance, resource utilization, and identify potential performance issues
- Compliance with relevant industry standards, regulations, and legal requirements, such as data privacy and security regulations
- User trust and confidence in the dependability of the software, ensuring reliable project management and collaboration experiences.

4.11 Reliability and Availability Requirements

Reliability and availability are crucial aspects of the IIT Kanban Board Software Project to ensure consistent performance and uninterrupted access for users. Here are the specific requirements for reliability and availability:

Reliability Requirements:

- The software should have a high level of reliability, meaning that it consistently performs its intended functions accurately and without failures.
- System failures, such as crashes or errors, should be minimized to avoid disruptions in project management and collaboration.
- The software should accurately capture and store project data, ensuring the reliability of information and its availability when needed.
- Any calculations, metrics, or automated processes within the software should produce accurate and reliable results.

Availability Requirements:

- The software should be highly available to users, ensuring that it is accessible whenever needed.
- Downtime for maintenance or upgrades should be scheduled during non-critical periods to minimize disruptions.
- The system should have mechanisms in place to quickly restore service in the event of any failures or disruptions.
- Redundancy and backup systems should be implemented to ensure continuous availability and minimize the impact of hardware or software failures.
- Adequate server capacity and load balancing should be considered to handle the expected user load and prevent performance degradation.
- Any scheduled maintenance or downtime should be communicated to users in advance to manage expectations and minimize any inconvenience.

4.12 Robustness and Fault Tolerance Requirements

- Robustness and fault tolerance are important aspects of the IIT Kanban Board Software Project to ensure its resilience in the face of errors, failures, or unexpected events. Here are the specific requirements for robustness and fault tolerance:
- The software should be robust and able to handle unexpected inputs, errors, or invalid user actions without crashing or producing unpredictable results.

- It should include robust error handling mechanisms to gracefully handle exceptions and provide meaningful error messages to users.
- User input should be validated to ensure data integrity and prevent unexpected behaviors or security vulnerabilities.
- The system should recover gracefully from errors or exceptions and continue functioning without significant disruptions or data loss.
- In case of errors, the software should provide appropriate feedback to users, guiding them on how to correct the issue or seek assistance.
- Fault Tolerance Requirements:
- The software should be designed with fault tolerance in mind to minimize the impact of failures or system errors.
- It should include mechanisms such as redundancy, backup systems, or failover mechanisms to ensure continuous operation even in the event of hardware or software failures.
- Critical data should be regularly backed up to prevent data loss in case of system failures or disasters.
- The system should have proper error detection and recovery mechanisms to identify and resolve faults automatically whenever possible.
- Monitoring tools should be in place to detect and alert administrators about any potential issues or performance degradation.
- The software should be able to isolate faults to prevent them from affecting the entire system and ensure the uninterrupted functioning of other components.

4.13 Safety Critical Requirements

If the IIT Kanban Board Software Project has safety-critical requirements, they would typically involve ensuring the safety and well-being of the users and the integrity of the data. However, since the project primarily focuses on project management and collaboration, it may not have explicit safety-critical requirements. Nonetheless, if safety considerations are relevant, they could include:

1. User Data Protection: The software should have appropriate measures in place to safeguard user data, such as personal information or sensitive project details. This may involve encryption, secure data transmission, and access control mechanisms to prevent unauthorized access or data breaches.
2. System Security: The software should be designed with security in mind to protect against potential vulnerabilities or attacks. This may include regular security audits, vulnerability

assessments, and adherence to secure coding practices to mitigate the risk of unauthorized access or data manipulation.

3. Data Backup and Recovery: The software should have a robust backup and recovery mechanism to ensure the availability and integrity of project data. Regular backups should be performed to protect against data loss or corruption, and recovery procedures should be in place to quickly restore the system in case of failures or disasters.
4. User Awareness: The software should provide clear instructions, warnings, and prompts to ensure user awareness and prevent unintended actions that may lead to data loss, system errors, or other safety concerns. This may involve providing context-sensitive help, user guides, or tooltips to guide users in performing actions correctly.

4.14 Maintainability and Supportability

Requirement	Description	Stakeholders
Maintainability	The software should be easily maintainable, allowing for efficient troubleshooting, bug fixing, and updates.	Development team, IT support team
Supportability	The software should have proper documentation and support channels in place to assist users in case of issues or inquiries.	Supervisor, ProjectMember, IT support team
Version Control	The software should have a version control system in place to manage code changes, track revisions, and facilitate collaboration among the development team.	Development team, IT support team
Modularity and Code Organization	The software code should be modular and well-organized, allowing for easier maintenance, code reuse, and enhancement.	Development team, IT support team
Logging and Error Handling	The software should have comprehensive logging capabilities to capture system errors and exceptions, facilitating debugging and troubleshooting.	Development team, IT support team
Documentation and Knowledge Transfer	The software should have clear and comprehensive documentation to facilitate system understanding and knowledge transfer among the development and support teams.	Development team, Supervisor, ProjectMember
Scalability	The software should be designed to accommodate future growth and increasing demands, allowing for	Development team, IT support team

	scalability and the addition of new features without significant disruptions.	

4.15 Maintenance Requirements

- The software should be easily maintainable, allowing for efficient troubleshooting, bug fixing, and updates.
- It should have proper documentation and annotations to aid in understanding the codebase and facilitate future maintenance.
- The software should follow coding standards and best practices to ensure readability and ease of maintenance.
- Version control should be implemented to track code changes, facilitate collaboration, and provide a history of modifications.
- The system should have a modular and well-organized code structure, enabling developers to easily locate and modify specific components.
- Comprehensive logging capabilities should be implemented to capture system errors and exceptions, aiding in debugging and maintenance.
- The software should include clear and up-to-date documentation, including user guides and technical manuals, to assist developers and users in understanding its functionality.
- Knowledge transfer processes should be in place to share system knowledge among team members and ensure continuity of maintenance efforts.
- The software should be designed with scalability in mind, allowing for future enhancements and the addition of new features without significant disruptions.
- Regular maintenance tasks, such as database backups, system updates, and security patches, should be scheduled and performed to ensure the system's stability and security.

4.16 Supportability Requirements

- The software should have proper documentation, including user guides, FAQs, and knowledge base, to assist users in understanding and using the system effectively.
- A dedicated support team or help desk should be available to address user inquiries, provide technical assistance, and resolve issues in a timely manner.

- Support channels, such as email, phone, or a ticketing system, should be established for users to reach out to the support team.
- The software should have an intuitive user interface and provide contextual help or tooltips to guide users in performing tasks.
- Regular software updates and bug fixes should be released to address known issues and improve system performance.
- Compatibility with various platforms and devices should be ensured to provide support for a wide range of users.
- The support team should maintain a knowledge base or FAQ section to address common user questions and provide self-help resources.
- System administrators should have access to monitoring tools and logs to proactively identify and resolve potential issues.
- Training materials, workshops, or webinars should be provided to users to enhance their understanding of the software's capabilities and features.
- The software should have a feedback mechanism in place to gather user suggestions and incorporate improvements based on user needs and preferences.
- Ongoing user engagement and communication should be maintained to address user concerns, gather feedback, and provide updates on system improvements.
- The support team should have a defined service level agreement (SLA) specifying response and resolution times for different types of support requests.
- Collaboration with third-party vendors or consultants may be necessary to ensure specialized support for specific components or integrations within the software.
- The software should have mechanisms in place to handle user data privacy and security, complying with relevant regulations and standards.
- The support team should actively monitor and respond to user feedback on social media, forums, or community platforms to address concerns and maintain a positive user experience.

4.17 Adaptability Requirements

Adaptability requirements for the IIT Kanban Board Software Project involve ensuring that the software can easily adapt to changing needs and environments. Here are some adaptability requirements for the project:

- Configurability: The software should allow users to customize and configure various settings to adapt to their specific project management needs. This may include adjusting task priorities, notification preferences, or board layouts.
- Flexibility: The software should be flexible enough to accommodate different project management methodologies, such as Agile or Waterfall, allowing users to choose the approach that best suits their projects.
- Scalability: The software should be designed to scale and handle an increasing number of projects, tasks, and users as the organization grows. It should be able to handle large amounts of data without sacrificing performance.
- Integration capabilities: The software should support integrations with other tools and platforms commonly used in project management, such as communication tools (Slack, Microsoft Teams) or document sharing platforms (Google Drive, Dropbox). This allows users to leverage existing tools and workflows seamlessly.
- Localization and Internationalization: The software should be adaptable to different languages, cultures, and regional requirements. It should support localization efforts, such as providing translations and accommodating date and time formats specific to different regions.
- Upgrade and Migration: The software should have provisions for easy upgrades to new versions, ensuring compatibility with newer technologies and features. It should also support smooth data migration processes when transitioning to a new environment or hardware infrastructure.
- Platform Independence: The software should be designed to run on multiple platforms, such as desktop computers, mobile devices, and web browsers. This enables users to access and use the software regardless of their preferred device or operating system.
- Future-proofing: The software should be designed with a forward-looking approach, considering potential future advancements and requirements. This involves following best practices, using modern technologies, and adopting modular architecture to facilitate future enhancements and integrations.

4.18 Security Requirements

Security requirements are crucial for the IIT Kanban Board Software Project to protect sensitive data, maintain user privacy, and mitigate the risk of unauthorized access or malicious activities. Here are some security requirements for the project:

- User Authentication: The software should require strong user authentication mechanisms, such as usernames and passwords, to verify the identity of users accessing the system.
- Role-Based Access Control: The software should implement role-based access control to ensure that users have appropriate permissions and access levels based on their roles and responsibilities within the organization.
- Data Encryption: The software should employ encryption techniques to protect sensitive data, both during transmission and storage. This includes encrypting user credentials, project details, and any other confidential information.
- Secure Communication: The software should utilize secure communication protocols, such as HTTPS, to ensure that data transmitted between users and the system is encrypted and protected against eavesdropping or tampering.
- Security Auditing and Monitoring: The software should include logging and monitoring capabilities to track user activities, detect suspicious behavior, and provide an audit trail for investigation and forensic analysis if needed.
- Secure Password Storage: User passwords should be securely hashed and stored to prevent unauthorized access in case of a data breach.
- Protection Against Cross-Site Scripting (XSS) and SQL Injection Attacks: The software should implement measures to sanitize user inputs and prevent common web application vulnerabilities like XSS and SQL injection attacks.
- Secure Session Management: The software should handle user sessions securely, utilizing mechanisms such as session timeouts and secure session cookies to prevent unauthorized access to user accounts.
- Regular Security Updates: The software should be kept up to date with security patches and updates to address any vulnerabilities discovered in third-party libraries or underlying frameworks.
- Data Backup and Recovery: The software should have regular and secure backups of data to prevent data loss in the event of system failures, disasters, or security incidents.
- Security Awareness and Training: Users should receive security awareness training to understand best practices for maintaining the security of their accounts and data, such as using strong passwords and recognizing phishing attempts.

4.19 Access requirements

Access requirements for the IIT Kanban Board Software Project involve ensuring that authorized users can access the system efficiently and securely. Here are some access requirements for the project:

- User Registration: The software should provide a user registration process where users can create individual accounts with unique credentials to access the system.
- User Authentication: The software should require users to authenticate themselves using their registered credentials (such as username and password) to gain access to their accounts and the system's features.
- Access Control: The software should implement access control mechanisms to enforce user permissions and privileges based on roles and responsibilities. This ensures that users can only access the features and data relevant to their assigned roles.
- User Profiles: The software should allow users to manage their profiles, including updating personal information, contact details, and notification preferences.
- Forgot Password Functionality: The software should provide a password recovery mechanism that allows users to reset their passwords securely in case they forget or need to change them.
- Multi-Factor Authentication (MFA): The software should support optional MFA methods, such as SMS verification codes or authenticator apps, to enhance the security of user accounts and prevent unauthorized access.
- Session Management: The software should handle user sessions securely, including session timeouts and the ability to terminate active sessions remotely.
- Single Sign-On (SSO): The software may support SSO integration with other authentication systems or identity providers used within the organization, allowing users to access the system using their existing credentials.
- Guest Access: The software may provide limited guest access or public-facing features that allow non-registered users to view certain project details or collaborate with registered users.
- Secure Remote Access: The software should support secure remote access, such as through Virtual Private Networks (VPNs) or secure remote desktop protocols, to enable users to access the system from remote locations securely.
- Compliance with Privacy Regulations: The software should adhere to relevant privacy regulations and standards to protect user data and ensure compliance with applicable laws.

4.20 Privacy Requirements

Privacy requirements are essential for the IIT Kanban Board Software Project to protect user information and ensure compliance with privacy regulations. Here are some privacy requirements for the project:

- Data Protection: The software should implement measures to protect user data from unauthorized access, use, disclosure, or alteration. This includes encryption of sensitive data, secure storage practices, and access control mechanisms.
- Consent Management: The software should provide mechanisms to obtain and manage user consent for data collection, processing, and sharing. Users should have control over their data and the ability to revoke consent if desired.
- User Anonymity: The software should allow users to use the system without revealing their personal identities, where appropriate and in accordance with user preferences.
- Data Minimization: The software should collect and retain only the necessary user data required for the system's functionality and specified purposes. Unnecessary or excessive data collection should be avoided.
- Transparent Privacy Policy: The software should have a clear and easily accessible privacy policy that informs users about the types of data collected, how it is used, and with whom it is shared. Users should have the ability to review and understand the privacy practices of the system.
- User Data Rights: The software should support user data rights, such as the right to access, correct, delete, or restrict the processing of their personal data. Users should have mechanisms to exercise these rights within the system.
- Data Sharing and Disclosure: The software should clearly define the circumstances under which user data may be shared or disclosed to third parties, and obtain user consent when necessary. Data sharing should adhere to relevant privacy regulations.
- Data Retention and Deletion: The software should establish clear policies for data retention and deletion, ensuring that user data is not retained longer than necessary and is securely deleted when no longer needed.
- User Notification: The software should notify users of any changes to the privacy policy or data processing practices that may affect their privacy rights. Users should be informed of any data breaches or security incidents that may compromise their personal data.
- Compliance with Privacy Regulations: The software should comply with applicable privacy regulations, such as the General Data Protection Regulation (GDPR) or local data protection laws, ensuring that user privacy rights are respected and protected.

4.21 Usability and Human Integrity Requirements

Usability and human integrity requirements are crucial for the IIT Kanban Board Software Project to ensure a user-friendly and intuitive system that promotes efficiency and supports the integrity of users' actions. Here are some usability and human integrity requirements for the project:

- Intuitive User Interface: The software should have a user-friendly and intuitive interface that allows users to navigate and interact with the system easily. This includes clear and organized menus, buttons, and visual elements.
- Responsive Design: The software should be responsive and adapt to different screen sizes and devices, ensuring a consistent user experience across desktop computers, laptops, tablets, and mobile devices.
- Accessibility: The software should be accessible to users with disabilities, complying with accessibility standards and guidelines such as WCAG (Web Content Accessibility Guidelines). This includes providing alternative text for images, keyboard navigation support, and compatibility with assistive technologies.
- Efficient Task Management: The software should provide efficient and effective task management capabilities, allowing users to create, assign, prioritize, and track tasks easily. This includes features like drag-and-drop functionality, bulk editing, and quick task creation options.
- Collaboration and Communication: The software should facilitate collaboration and communication among team members, allowing them to share project details, assign tasks, and exchange messages or comments within the system. Real-time updates and notifications should be provided to keep users informed.
- Error Handling and Feedback: The software should have effective error handling mechanisms to provide clear and informative error messages when users encounter issues or make mistakes. Feedback should be provided promptly to confirm successful actions and guide users through the system.
- Customizability and Personalization: The software should allow users to customize their experience to suit their preferences. This may include the ability to personalize the interface, choose preferred display options, and set notification preferences.
- Training and Documentation: The software should provide comprehensive user documentation, tutorials, and help resources to assist users in understanding the system's functionality and using it effectively. Training materials, such as video tutorials or user guides, can be provided to onboard new users.

- Data Integrity: The software should ensure the integrity of user data, including accurate and reliable storage, retrieval, and updates. User actions, such as creating, editing, or deleting tasks, should be reflected accurately in the system.
- User Support: The software should provide user support channels, such as a help desk or customer support system, to address user inquiries, provide technical assistance, and resolve issues in a timely manner.

4.22 Ease of Use Requirements

Requirement	Stakeholder	Description
Intuitive User Interface	Supervisor, Project Member	The software should have a user-friendly interface with clear and intuitive navigation, making it easy for users to understand and interact with the system.
Simple Task Management	Supervisor, Project Member	Users should be able to create, update, and prioritize tasks with minimal effort. The task management features should be straightforward and user-friendly.
Quick Learning Curve	New Users	The software should be easy to learn, allowing new users to quickly understand its functionalities and start using it effectively without extensive training.
Clear Visualization	Supervisor, Project Member	The Kanban board should provide a visually appealing and clear representation of project progress, task statuses, and deadlines for easy monitoring and tracking.
Responsive Design	Supervisor, Project Member	The software should be accessible and usable across different devices (desktop, mobile, tablets) with responsive design, ensuring a seamless user experience.
Contextual Help and Support	Supervisor, Project Member	The system should provide contextual help and support features, such as tooltips, user guides, and tutorials, to assist users in understanding and using the software.
Efficient Search Functionality	Supervisor, Project Member	Users should be able to quickly search for projects, tasks, or specific information within the software, facilitating easy navigation and retrieval of relevant data.
Customizable User Preferences	Supervisor, Project Member	The software should allow users to personalize their settings, such as language preferences, notification preferences, and display options, to suit their individual needs.

4.23 Accessibility Requirements

Accessibility requirements are critical for ensuring that the Easy Food Tracker web-based application is usable by everyone, including individuals with disabilities. Here are a few examples of accessibility requirements that can be considered for this project:

- **Compliance with accessibility standards**: The platform should comply with accessibility standards such as Web Content Accessibility Guidelines (WCAG) 2.1, to ensure that it is usable by people with disabilities.
- **Keyboard-only navigation:** The platform should be usable with only a keyboard, with no mouse or other pointing device required.
- **Screen reader compatibility:** The platform should be compatible with screen readers, allowing people with visual impairments to access its content and functionality.
- **High contrast mode:** The platform should provide a high contrast mode option to make it easier for people with low vision to use the application.
- **Alternate text descriptions:** All images, videos, and graphics should have alternate text descriptions to make it easier for people with visual impairments to understand the content.
- **Simple and clear language:** The language used in the platform should be simple and clear, with short paragraphs and headings, to make it easier for people with cognitive disabilities to understand the content.

4.24 User Documentation

Section	Description	Priority
Introduction	Provides an overview of the software and its purpose, and explains how to access it.	High
Getting Started	Guides users through the registration and login process, and explains user roles.	High
Dashboard Overview	Provides an overview of the main dashboard, navigation options, and project organization.	High

Project Management	Explains how to create new project folders, invite and manage project members, and adjust project settings.	High
Task Management	Guides users through creating tasks, assigning them to team members, and tracking task progress.	High
Document Management	Explains how to upload and share project documents, manage versions, and control document access.	Medium
Notifications and Messaging	Guides users on managing notifications and utilizing messaging features within the software.	Medium
Reporting and Analytics	Explains how to generate project reports and utilize analytics for tracking and decision-making.	Medium
Customization Options	Explains how users can personalize their profile settings and customize project views and preferences.	Low
Troubleshooting and FAQs	Provides solutions to common issues and answers frequently asked questions.	Low
Glossary of Terms	Defines key terms and concepts used in the software.	Low
Support and Contact Information	Provides information on how to seek assistance or report issues, and contact details for support.	Low

4.25 Look and Feel Requirements

Requirement	Description	Stakeholders
Consistent Visual Design	The software should have a consistent visual design throughout, with a cohesive color scheme, typography, and graphical elements.	Supervisor, Project Member, Designers
Responsive and Adaptive Design	The interface should be responsive and adapt seamlessly to different screen sizes and resolutions, providing a consistent user experience across devices.	Supervisor, Project Member, Designers
Intuitive Navigation	The navigation should be intuitive and easily accessible, allowing users to move between different sections and features without confusion.	Supervisor, Project Member, Designers
Clear and Readable Typography	The typography used in the software should be clear and readable, with appropriate font sizes, line spacing, and contrast for optimal legibility.	Supervisor, Project Member, Designers
Appropriate Use of Visual Cues	Visual cues, such as icons, tooltips, and hover effects, should be used appropriately to provide	Supervisor, Project Member, Designers

	additional information and enhance user interactions.	
Visual Hierarchy and Organization	The interface should effectively communicate the hierarchy of information, with important elements emphasized and a logical arrangement of content.	Supervisor, Project Member, Designers
Pleasant and Engaging Interface	The software should provide a visually appealing and engaging interface, incorporating attractive visuals, animations, and transitions where appropriate.	Supervisor, Project Member, Designers
Consistent Branding	The software should reflect the branding guidelines and visual identity of the institution or organization associated with the IIT Kanban Board Software Project.	Institution Designers

4.26 Appearance Requirements

Appearance Requirements refer to the visual and design aspects of the Easy Food Tracker web application. This includes the color scheme, typography, and overall aesthetic of the application. These requirements are important to ensure that the application is visually appealing and consistent with the branding of the product. Some key areas to consider in terms of Appearance Requirements include:

- Consistent use of brand colors and typography
- User-friendly layout and design
- High-quality graphics and images
- Appropriate use of whitespace to create a clean and uncluttered look
- Responsiveness of the design for different screen sizes

It is important to involve stakeholders such as end-users, marketing, and restaurant owners in determining the specific Appearance Requirements for the Easy Food Tracker web application, to ensure that the final product is visually appealing and meets the needs of all stakeholders.

4.27 Legal Requirements

- Data Privacy and Protection: The application must comply with laws such as the European Union's General Data Protection Regulation (GDPR) and California Consumer Privacy Act (CCPA) regarding the collection, storage, and use of personal data.

- Intellectual Property: All content, including images and text, must be either original or properly licensed to avoid copyright infringement.
- Terms of Service: The application must have a clear and concise Terms of Service agreement that outlines the rights and responsibilities of users and the company.
- User Content: The application must have a policy in place to moderate user-generated content, including reviews and ratings, to ensure that it complies with laws regarding hate speech, defamatory statements, and other illegal or harmful material.
- Advertising: The application must comply with laws regarding advertising and marketing, including truth in advertising and disclosure of sponsored content.
- Accessibility: The application must comply with accessibility standards, such as the Web Content Accessibility Guidelines (WCAG), to ensure that it is usable by people with disabilities.

4.28 Standard Requirements

Standard requirements for the IIT Kanban Board Software Project typically include:

- User authentication and secure access control
- User registration and account management
- Project creation and management
- Task creation, assignment, and tracking
- Document management and sharing
- Collaboration features such as messaging and notifications
- Search functionality for projects, tasks, and documents
- Reporting and analytics capabilities
- Integration with other tools or platforms if required
- Responsive and user-friendly interface
- Cross-browser compatibility
- Scalability to handle a growing number of users and projects
- Robust data backup and recovery mechanisms
- Regular software updates and maintenance
- Performance optimization for efficient usage
- Compatibility with different operating systems
- Compatibility with mobile devices and responsive design
- Clear and concise error handling and feedback messages
- Localization support for multiple languages if applicable
- Compliance with coding and software development best practices
- Thorough testing and quality assurance processes
- Documentation of the software functionalities and user guide

5. Requirement Engineering Process

Requirements Engineering (RE) determines software requirements according to customer requirements or needs. Requirements engineering process includes –

- Requirements elicitation
- Needs modeling
- Requirements analysis
- Requirements assurance & validation
- Requirements management.

- **Requirement Elicitation:** Gathering information from stakeholders (such as project managers, users, administrators) to understand their needs, expectations, and desired functionalities of the software. This can be done through interviews, surveys, brainstorming sessions, and studying existing processes.
- **Requirement Analysis:** Analyzing and organizing the gathered requirements to identify their significance, prioritize them, and resolve any conflicts or inconsistencies. This involves categorizing requirements as functional or non-functional and identifying any dependencies or constraints.
- **Requirement Specification:** Documenting the requirements in a clear and unambiguous manner using techniques like use case diagrams, user stories, and requirement specifications. This step ensures that the requirements are well-defined and understandable to the development team.
- **Requirement Validation:** Reviewing the documented requirements with stakeholders to verify their accuracy, completeness, and consistency. This can involve feedback sessions, prototyping, or simulation exercises to ensure that the requirements align with stakeholders' expectations.
- **Requirement Management:** Managing changes to requirements throughout the project lifecycle. This involves tracking and documenting any changes, assessing their impact, and ensuring that all stakeholders are informed about the modifications.
- **Requirement Traceability**: Establishing traceability between requirements and other project artifacts, such as design documents, test cases, and implementation code. This allows for easier impact analysis, verification, and validation of requirements at different stages of the project.
- **Requirement Communication:** Effectively communicate with the requirements to the development team, designers, testers, and other project stakeholders. This ensures a shared understanding of the project objectives and facilitates collaboration among team members.
- **Requirement Prioritization:** Prioritizing requirements based on their importance, urgency, and feasibility. This helps allocate development resources efficiently and ensures that the most critical requirements are addressed first.
- **Requirement Documentation:** Maintaining a well-organized and up-to-date repository of requirement documents throughout the project lifecycle. This serves as a reference for all stakeholders and provides a basis for future enhancements or maintenance.

- **Requirement Validation:** Conducting periodic reviews and validation of requirements against the implemented software to ensure that they have been successfully met and that the software meets the desired objectives.

5.1 Requirements elicitation Technique

The IIT Kanban Board Software Project can employ various requirements elicitation techniques to gather information from stakeholders. Here are some commonly used techniques:

1. **Interviews:** Conduct one-on-one or group interviews with stakeholders, including project managers, teachers, students, and potential users. This allows for in-depth discussions to understand their needs, expectations, and challenges related to project management and collaboration.

2. **Questionnaires:** Distributing questionnaires to a larger group of stakeholders can help gather standardized responses and quantitative data. Questionnaires can be designed to collect information about specific requirements, preferences, and priorities. They are an efficient way to reach a wide range of stakeholders and obtain a large volume of data.

3. **Document Analysis:** Reviewing existing project management documents, such as project plans, task lists, and communication records, provide insights into current practices, challenges, and requirements. Analyzing these documents help us to identify gaps, inconsistencies, and areas for improvement. We collect some requirement like messaging (based on Gmail messaging), report generating (based on Supervisor Requirements), Gantt Chart (based on Click Up).

4. **Brainstorming:** Conducting brainstorming sessions with stakeholders fosters creativity and encourages stakeholders to generate ideas, suggestions, and requirements. This technique can uncover novel requirements and innovative solutions to enhance project management and collaboration.

5.1.1 Hold Elicitation Interviews

Stakeholder Information

1. Name of the stakeholder:
2. Role or position in the organization:
3. Contact information (email/phone):

Project Understanding

The groundbreaking IIT Kanban Board Software Project, where innovation meets efficiency in project management and collaboration. In this digital era, we recognize the need to optimize the way teachers and students handle their project work within the IIT community. Our purpose is clear: to revolutionize the existing manual processes and elevate productivity to new heights.

The primary objective of the IIT Kanban Board Software Project is to enhance the efficiency and effectiveness of project work for both teachers and students. By transitioning from manual project handling to a digital platform, we seek to streamline the planning, tracking, and management of tasks. The purpose is to improve collaboration, communication, and productivity among team members.

4. On a scale of 1-5, how would you rate the importance of the project in addressing the organization's needs? (1 - Not important at all, 5 - Extremely important)
    1. 1
    2. 2
    3. 3
    4. 4
    5. 5
5. How do you envision the project improving collaboration and project management within the organization?

    a. Streamlining communication channels

    b. Enhancing task assignment and tracking

    c. Providing centralized document management

    d. Improving visibility into project timelines and milestones

Stakeholder Requirements

6. Which of the following functionalities or features do you consider essential for the project? (Select all that apply)

    a. Task management and assignment

    b. Document sharing and version control

    c. Project timeline and milestone tracking

    d. Notifications and reminders

    e. Reporting and analytics

    f. Other (Please specify): _____

7. On a scale of 1-5, how important is the user interface and user experience of the project? (1 - Not important at all, 5 - Extremely important)
    1. 1
    2. 2
    3. 3
    4. 4
    5. 5
8. What level of access control or permissions do you think should be provided to different stakeholders?
    a. Read-only access
    b. Edit access
    c. Administrator access
    d. Customizable access levels Impact and Concerns
9. How do you anticipate the project will impact your daily work or responsibilities?
    a. Positive impact
    b. No significant impact
    c. Negative impact
    d. If negative, please specify concerns: _____
10. What are the key success factors for the project, in your opinion? (Select all that apply)
    a. User adoption and engagement
    b. Seamless integration with existing workflows
    c. Robust training and support resources
    d. Effective change management strategies
    e. Other (Please specify): _____

Additional Feedback

11. Is there any additional information or feedback that you would like to provide regarding the project?

12. Are there any other stakeholders or individuals who should be involved in the requirements gathering process?

5.1.2 Distribute Questionnaires

The questionnaire is an effective tool for gathering information on styles, changes in attitudes and preferences, and user satisfaction. To minimize fatigue or frustration for the respondent, our questions were kept concise and grouped together based on topics. This allowed the respondent to focus on specific areas and provided a clear rationale for each question. The main advantage of using this survey approach was the ability to collect responses in a standard manner, allowing for the consolidation of information from a large number of people.

We use two separate set of questionnaires for this process-

- For Supervisor
- For project members

5.1.3 Competitor Analysis

Analyzing competitor applications to understand what features and functionality are popular and in demand in the market.

5.2 Requirements analysis

Requirement' analysis is a critical phase in the software development lifecycle, where the project team identifies, understands, and documents the requirements of the IIT Kanban Board Software Project. This process involves analyzing user needs, goals, and constraints to define the scope of the project and determine the functionalities and features that the software should deliver. Here are some key steps involved in requirements analysis:

1. Requirement Identification: The project team identifies and documents the stakeholders' requirements by gathering information through interviews, surveys, and discussions. The focus is on understanding the users' needs, project objectives, and desired outcomes.
2. Requirement Prioritization: The identified requirements are prioritized based on their importance and impact on the project's success. Stakeholders and the project team collaborate to determine the critical requirements that must be addressed in the initial phases of development.

3. Requirement Documentation: The requirements are documented in a clear and concise manner. This includes capturing functional requirements, such as specific features and functionalities the software should provide, as well as non-functional requirements, such as performance, security, and usability criteria.
4. Requirement Validation: The documented requirements are reviewed and validated to ensure accuracy, completeness, and feasibility. This may involve conducting reviews, walkthroughs, or prototype demonstrations with stakeholders and subject matter experts.
5. Requirement Traceability: The project team establishes traceability between the requirements and other project artifacts, such as design documents, test cases, and user documentation. This allows for tracking and ensuring that all requirements are adequately addressed throughout the project lifecycle.
6. Requirement Change Management: As the project progresses, there may be changes or additions to the requirements. The project team should have a process in place to manage requirement changes effectively, including impact analysis, change approval, and proper documentation of revised requirements.
7. Requirement Validation with Users: The project team may conduct user acceptance testing or usability testing to validate the implemented features against the user's expectations. This feedback can help identify any gaps or areas for improvement in the requirements.

5.3 Requirements specification

Requirements specification in the IIT Kanban Board project involves documenting the requirements in a clear and concise manner, so that they can be used as a basis for the design and development of the application. The following are some of the steps involved in the requirements specification phase:

- **Requirements Documentation:** Writing detailed and unambiguous documentation of the requirements, including functional and non-functional requirements.
- **Requirements Organization:** Organizing the requirements in a clear and logical manner, with a focus on usability and maintainability.
- **Requirements Traceability:** Ensuring that each requirement is traced back to its source, so that changes can be tracked and evaluated.
- **Requirements Communication:** Communicating the requirements to stakeholders and other relevant parties, to ensure that everyone is on the same page and that the requirements are clearly understood.

5.4 Requirements validation

Requirements validation in the IIT Kanban Board project involves checking that the requirements have been correctly implemented in the application and that they meet the needs and goals of the stakeholders. The following are some of the methods that can be used for requirements validation in this project:

- **Prototype Testing:** Creating a working prototype of the application and testing it with users to validate the requirements and identify any issues or areas for improvement.

System Interface Analysis

The initial step in System Interface Analysis is to determine the systems with which the new system will need to communicate. This can involve various types of systems, such as servers on the internet, software on the same host, hardware devices, or other systems with different functions. To ensure effective communication between the new system and other systems, it is crucial to accurately identify the systems that will be involved and to understand their functions and requirements.

- **Requirements Traceability:** Using the traceability information to verify that all requirements have been implemented in the application.

6. Use Case Diagram

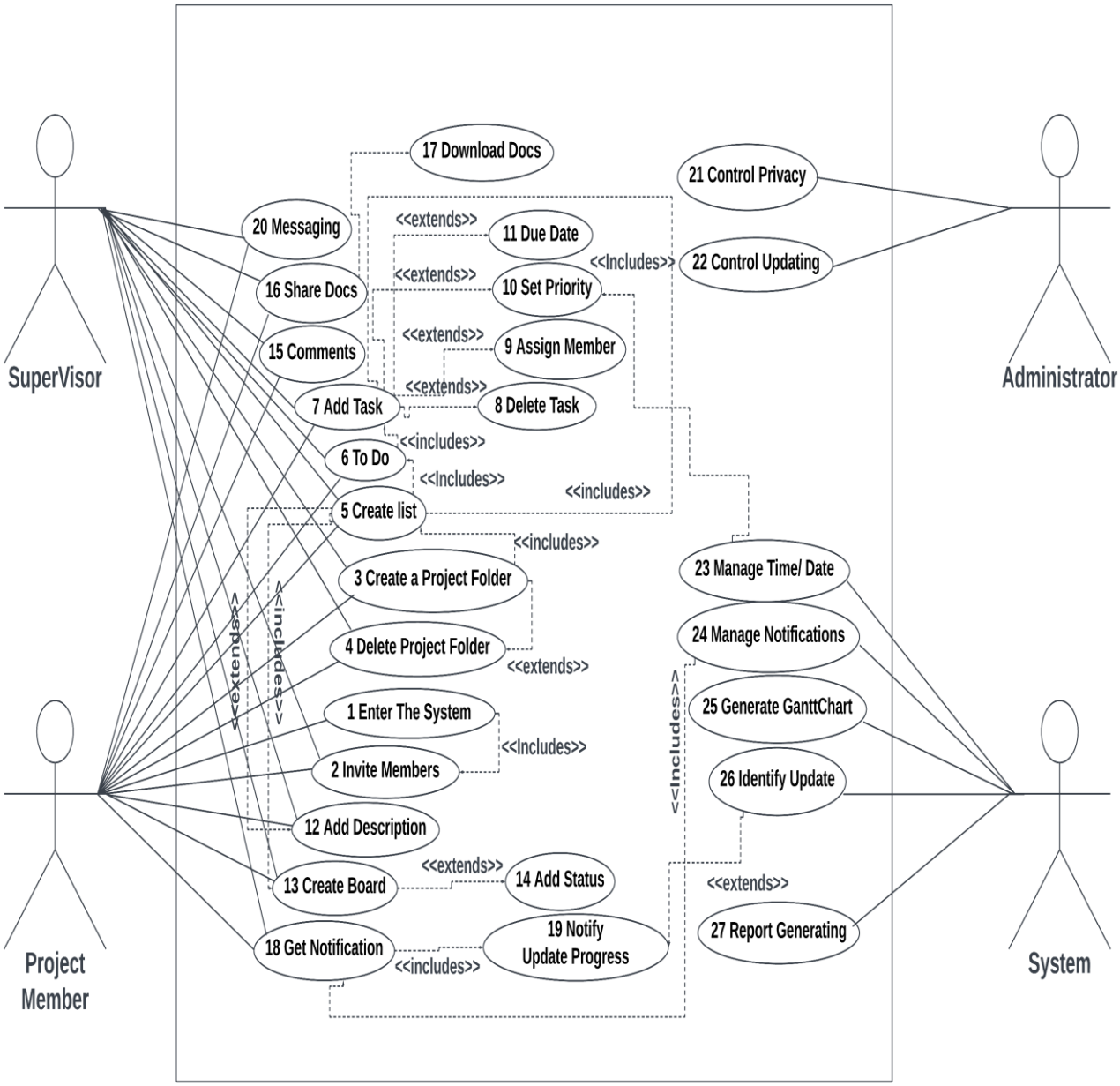


Figure 1: Use Case Diagram

7.1 Use Case Description

7.1.1 Enter the System

Enter the System

Use Case	Enter the System	
Goal	Access the IIT Kanban Board Software Project.	
Pre-conditions	The project member has valid login credentials.	
Post conditions	Project member successfully enters the system.	
Success End Condition	The project member successfully enters the system.	
Failed End Condition	Project member is unable to log in or access the system.	
Primary Actors: Secondary Actors:	Supervisor and Project Member N/A	
Trigger	The project member wants to use the system.	
Main Success Flows	Step	Action
	1	The project member opens the IIT Kanban Board Software Project application or website.
	2	The system presents the login screen.
	3	The project member enters their login credentials (username and password).
	4	The project member submits the login or sign-up information.
	5	If the login information valid, then enter into the system.
	6	The system displays the main dashboard of the IIT Kanban Board Software Project, showing relevant project information.

Alternative Flows	Step	Action
	4a	If the login credentials are invalid.
		4a1. The system displays an error message indicating the incorrect credentials.
		4a2. The project member can re-enter the login
		4a3. Information or request password reset if needed.
		4a4. If the project member re-enters the login
		4a5. Information, the system re-verifies the credentials.
		4a6. If the project member requests a password reset, the system follows the password reset process.
Quality Requirements	1. Performance: The system should provide quick and responsive login verification. 2. Usability: The login screen should be user-friendly, allowing project members to easily enter their credentials. 3. Security: The system should securely handle and validate login credentials to prevent unauthorized access.	

7.1.2 Invite Members

Use Case	Invite Members
Goal	Access the IIT Kanban Board Software Project.
Pre-conditions	The project member is logged into the system.
Post conditions	Project member receives an invitation to join the project
Success End Condition	The invitation is successfully sent to the new member(s).
Failed End Condition	Project member is not successfully invited.
Primary Actors:	Supervisor and Project Member

Secondary Actors:	N/A	
Trigger	The project member wants to invite new members.	
Main Success Flows	Step	Action
	1	The project member navigates to the "Invite Members" section of the IIT Kanban Board Software Project.
	2	The system displays the invite members interface, providing options to enter email addresses or select from a list of existing contacts.
	3	The project member enters the email address(es) of the new member(s) to be invited.
	4	The project member verifies the entered email address(es) and proceeds.
	5	The project member clicks the "Send Invitation" button.
	6	The system displays a confirmation message indicating that the invitations have been successfully sent.
Alternative Flows	Step	Action
	3a	If the project member selects the option to select from a list of existing contacts instead of entering email addresses:
		3a1. The system displays a list of contacts associated with the project member's account.
	6a	6a1. The system displays an error message indicating the failure to send the invitations
		6a2. The project member can choose to retry sending the invitations or cancel the operation.
Quality Requirements	1. Sent invite notification in 5sec. 2. Maintain well secure invitation.	

7.1.3 Create a Project Folder

Use Case	Enter the System	
Goal	Create a new project folder and associated task list.	
Pre-conditions	The supervisor is logged into the IIT Kanban Board Software Project.	
Post conditions	Project folder is successfully created	
Success End Condition	The project member successfully enters the system.	
Failed End Condition	Project folder is not created or an error occurs.	
Primary Actors: Secondary Actors:	Supervisor Project Member	
Trigger	The supervisor wants to organize a new project.	
Main Success Flows	Step	Action
	1.	The supervisor selects the "Create Project Folder" option from the main menu of the IIT Kanban Board Software Project.
	2.	The supervisor enters the project folder name, description, and any other required information.
	3.	The supervisor enters the project folder name, description, and any other required information.
	4.	The supervisor clicks the "Create" button to create the project folder.
	5.	The system validates the entered information and creates the project folder.
	6.	The system displays the created project folder on the main dashboard.
	7.	The supervisor selects the created project folder.

	8.	The system displays the project folder interface and provides an option to create a new task list within the folder.
	9.	The supervisor selects the "Create List" option.
	10.	The system prompts the supervisor to enter the name and other details of the new task list.
	11.	The supervisor enters the task list details and clicks the "Create" button.
	12.	The system validates the entered information and creates the task list within the project folder.
	13.	The system displays the created task list within the project folder.
Alternative Flows	Step	Action
	9a	If the supervisor chooses to create multiple task lists within the project folder:
		9a1. The supervisor repeats steps 9-13 for each additional task list.
Quality Requirements	1. Performance: The system should create project folders and associated task lists quickly and responsively. 2. Reliability: The creation process should be reliable, ensuring that project folders and task lists are created without errors. 3. Usability: The interface for creating project folders and task lists should be user-friendly and intuitive for the supervisor. 4. Security: The system should ensure the confidentiality and access control of project folders and task lists. 5. Availability: The creation feature should be available and accessible for the supervisor to create project folders and task lists at any time.	

7.1.4 Delete Project Folder

Use Case	Delete the Project Folder	
Goal	Delete a project folder and its associated contents.	
Pre-conditions	The project member has valid login credentials.	
Post conditions	The project folder is successfully deleted.	
Success End Condition	The project member successfully enters the system.	
Failed End Condition	Project folder is not deleted or an error occurs.	
Primary Actors: Secondary Actors:	Supervisor Project Member	
Trigger	The primary actor (Supervisor or Project Member) wants to delete a project folder.	
Main Success Flows	Step	Action
	1.	The primary actor (Supervisor or Project Member) selects the project folder to be deleted from the IIT Kanban Board Software Project interface.
	2.	The system displays the project folder interface, showing the folder's contents and options.
	3.	The primary actor (Supervisor or Project Member) locates the "Delete Folder" option within the interface.
	4.	The primary actor (Supervisor or Project Member) clicks on the "Delete Folder" option.
	5.	The system prompts the primary actor (Supervisor or Project Member) to confirm the deletion action.
	6.	The primary actor (Supervisor or Project Member) confirms the deletion by clicking the "Confirm" button.
	7.	The system verifies the primary actor's permissions and proceeds with the deletion process.

	8.	The system deletes the project folder and its associated contents, including tasks, lists, and files. The system displays a confirmation message indicating that the project folder has been successfully deleted.
Alternative Flows	Step	Action
	7a	If the primary actor (Supervisor or Project Member) does not have sufficient permissions to delete the folder:
		4a1. The system displays an error message indicating the lack of permissions.
Quality Requirements	1. Performance: The system should handle project folder deletions quickly, even with large amounts of associated data. 2. Reliability: The deletion process should be reliable, ensuring that the project folder and its contents are permanently removed. 3. Usability: The delete folder option should be easily accessible and understandable for both supervisors and project members. 4. Security: The system should verify the permissions of the primary actor (Supervisor or Project Member) before allowing the deletion to prevent unauthorized removal of important project data. 5. Availability: The delete folder feature should be available and functional at all times for authorized primary actors (Supervisor or Project Member).	

7.1.5 Create List

Use Case	Create List
Goal	Create a new list within a project folder.
Pre-conditions	The primary actor has appropriate permissions.
Post conditions	New list is successfully created and added to the project.

Success End Condition	The project member successfully enters the system.	
Failed End Condition	New list is not created or an error occurs.	
Primary Actors: Secondary Actors:	Supervisor N/A	
Trigger	The primary actor (Supervisor or Project Member) wants to create a new list.	
Main Success Flows	Step	Action
	1.	The primary actor (Supervisor or Project Member) navigates to the project folder where the list will be created.
	2.	The system displays the project folder interface, showing the existing lists and options.
	3.	The primary actor (Supervisor or Project Member) locates the "Create List" or "Add List" option within the interface.
	4.	The primary actor (Supervisor or Project Member) clicks on the "Create List" or "Add List" option.
	5.	The system presents a form or input field for the primary actor (Supervisor or Project Member) to enter the name of the new list.
	6.	The primary actor (Supervisor or Project Member) enters the desired name for the new list.
	7.	The primary actor (Supervisor or Project Member) submits the form or confirms the list creation.
	8.	The system validates the entered information and creates the new list within the project folder.
	9.	The system displays a confirmation message indicating that the new list has been successfully created.
Alternative Flows	Step	Action

	8a	If the primary actor (Supervisor or Project Member) enters a duplicate or invalid name for the new list:
		8a1. The system displays an error message indicating the issue with the entered name.
Quality Requirements	1. Performance: The system should create lists quickly, even within project folders with a large number of existing lists. 2. Reliability: The list creation process should be reliable, ensuring that the new list is accurately created and visible to authorized users. 3. Usability: The create list option should be easily accessible and intuitive for both supervisors and project members. 4. Security: The system should verify the permissions of the primary actor (Supervisor or Project Member) before allowing the list creation to prevent unauthorized additions. 5. Security: The system should verify the permissions of the primary actor (Supervisor or Project Member) before allowing the list creation to prevent unauthorized additions.	

7.1.6 To Do

Use Case	To Do
Goal	Create a new "To Do" task within a project.
Pre-conditions	The primary actor has appropriate permissions.
Post conditions	To-do task is successfully added or updated
Success End Condition	A new "To Do" task is successfully created within the project.
Failed End Condition	To-do task is not added or updated.
Primary Actors: Secondary Actors:	Supervisor, Project Member N/A

Trigger	The primary actor (Supervisor or Project Member) wants to create a new "To Do" task.	
Main Success Flows	Step	Action
	1	The primary actor (Supervisor or Project Member) navigates to the project or list where the task will be created.
	2	The system displays the project or list interface, showing the existing tasks and options.
	3	The primary actor (Supervisor or Project Member) locates the "Create Task" or "Add Task" option within the interface.
	4	The primary actor (Supervisor or Project Member) clicks on the "Create Task" or "Add Task" option.
	5	The system presents a form or input fields for the primary actor (Supervisor or Project Member) to enter task details such as title, description, due date, assignee, and priority.
	6	The primary actor (Supervisor or Project Member) enters the required information for the new "To Do" task.
	7	The primary actor (Supervisor or Project Member) submits the form or confirms the list creation.
	8	The system validates the entered information and creates the new "To Do" task within the project or list.
	9	The system displays a confirmation message indicating that the new "To Do" task has been successfully created.
Alternative Flows	Step	Action
	8a	If the primary actor (Supervisor or Project Member) enters incomplete or invalid information for the task:
		8a1. The system displays an error message indicating the missing or incorrect information.
		8a2.If the primary actor (Supervisor or Project Member) cancels the task creation, the system returns to the project or list interface without creating the task.

Quality Requirements	1. The system should validate and enforce any required fields for the task. 2. The system should provide real-time updates for task changes. 3. The task status should be synchronized across all project members.

7.1.7 Add Task

Use Case	Add task
Goal	Add a new task to a project
Pre-conditions	Project member is logged in and has access to the project
Post conditions	The new task is successfully added to the project
Success End Condition	The new task is added to the project
Failed End Condition	N/A
Primary Actors: Secondary Actors:	Supervisor, Project Member N/A

Trigger	Project member wants to add a task	
Main Success Flows	step	Action
	1.	Project member selects the project where they want to add the task.
	2.	Project member navigates to the "Tasks" section of the project
	3.	Project member clicks on the "Add Task" button.
	4.	Project member fills in the necessary details for the task (e.g., title, description, due date)
	5.	Project member clicks on the "Save" button to submit the task.
	6.	The system validates the input and creates a new task associated with the project.
Alternative Flow	N/A	
Quality Requirements	1. Add task as the project member requirement. 2. Set date and time, system will generate a notification for the users for alarming.	

7.1.8 Delete Task

Use Case	Delete Task
Goal	Delete an existing task within a project or list.
Pre-conditions	The primary actor has appropriate permissions.

Post conditions	Task is successfully deleted.	
Success End Condition	The task is successfully deleted from the project or list.	
Failed End Condition	Task is not deleted.	
Primary Actors: Secondary Actors:	Supervisor, Project Member N/A	
Trigger	The primary actor (Supervisor or Project Member) wants to delete a task.	
Main Success Flows	Step	Action
	1	The primary actor (Supervisor or Project Member) navigates to the project or list containing the task to be deleted.
	2	The system displays the project or list interface, showing the existing tasks and options.
	3	The primary actor (Supervisor or Project Member) locates the task to be deleted within the interface.
	4	The primary actor (Supervisor or Project Member) selects the task to be deleted.
	5	The system presents a confirmation dialog or prompts for confirmation before deleting the task.
	6	The primary actor (Supervisor or Project Member) confirms the deletion by clicking on the "Delete" or "Confirm" button.
	7	The system displays a confirmation message indicating that the task has been successfully deleted.
Alternative Flows	Step	Action
	6a	If the primary actor (Supervisor or Project Member) cancels the deletion:

		6a1. The system closes the confirmation dialog or returns to the project or list interface without deleting the task.
		6a2. The task remains in the project or list without any changes.
Quality Requirements	1. Performance: The system should delete tasks quickly, even within projects or lists with a large number of existing tasks. 2. Reliability: The task deletion process should be reliable, ensuring that the task is accurately removed from the project or list. 3. Usability: The task deletion option should be easily accessible and intuitive for both supervisors and project members. 4. Security: The system should verify the permissions of the primary actor (Supervisor or Project Member) before allowing the task deletion to prevent unauthorized removals. 5. Availability: The task deletion feature should be available and functional at all times for authorized primary actors (Supervisor or Project Member).	

7.1.9 Assign Member

Use Case	Assign Member
Goal	Assign a project member to a task within a project.
Pre-conditions	The primary actor has appropriate permissions.
Post conditions	Project member is successfully assigned to the task.
Success End Condition	A project member is successfully assigned to the task.
Failed End Condition	Project member is not assigned to the task.

Primary Actors: Secondary Actors:	Supervisor, Project Member N/A	
Trigger	The primary actor (Supervisor or Project Member) wants to assign a member to a task.	
Main Success Flows	Step	Action
	1.	The primary actor (Supervisor or Project Member) navigates to the project or list containing the task.
	2.	The system displays the project or list interface, showing the existing tasks and options.
	3.	The primary actor (Supervisor or Project Member) locates the task to which they want to assign a member within the interface.
	4.	The primary actor (Supervisor or Project Member) selects the task to be assigned.
	5.	The system displays the task details, including the assigned member (if any) or an option to assign a member.
	6.	The primary actor (Supervisor or Project Member) clicks on the "Assign Member" option.
	7.	The system presents a list of project members or allows the primary actor to search and select a member to be assigned.
	8.	The primary actor (Supervisor or Project Member) selects the project member to be assigned to the task.
	9.	The primary actor (Supervisor or Project Member) confirms the assignment by clicking on the or "Confirm" button.
	10.	The system updates the task details to reflect the assigned member and notifies the relevant parties.
	11.	The system displays a confirmation message indicating that the member has been successfully assigned to the task.

Alternative Flows	Step	Action
	6a	If the primary actor (Supervisor or Project Member) decides not to assign a member:
		6a1 The primary actor (Supervisor or Project Member) cancels the assignment process.
		6a2. The system returns to the task details without making any changes to the assignment.
Quality Requirements	1. Performance: The system should assign members to tasks quickly, even within projects or lists with a large number of existing tasks and members. 2. Reliability: The member assignment process should be reliable, ensuring that the assigned member is accurately recorded and visible to authorized users. 3. Usability: The member assignment option should be user-friendly, allowing supervisors and project members to easily assign and view assigned members. 4. Security: The system should verify the permissions of the primary actor (Supervisor or Project Member) before allowing the member assignment to prevent unauthorized assignments. 5. Availability: The member assignment feature should be available and functional at all times for authorized primary actors (Supervisor or Project Member).	

7.1.10 Set priority

Use Case	Set priority
Goal	Set the priority level of a task within a project.
Pre-conditions	The primary actor has appropriate permissions.
Post conditions	The priority level of the task is successfully updated.

Success End Condition	The priority of the task is successfully set.	
Failed End Condition	The priority level of the task is not updated.	
Primary Actors: Secondary Actors:	Supervisor, Project Member N/A	
Trigger	The primary actor (Supervisor or Project Member) wants to set the priority of a task.	
Main Success Flows	Step	Action
	1.	The primary actor (Supervisor or Project Member) navigates to the project or list containing the task.
	2.	The system displays the project or list interface, showing the existing tasks and options.
	3.	The primary actor (Supervisor or Project Member) locates the task for which they want to set the priority within the interface.
	4.	The primary actor (Supervisor or Project Member) selects the task to be prioritized.
	5.	The system displays the task details, including the current priority or an option to set the priority.
	6.	The primary actor (Supervisor or Project Member) clicks on the "Set Priority" or "Change Priority" option.
	7.	The system presents a list or menu of priority levels (e.g., high, medium, low) or allows the primary actor to enter a numeric value.
	8.	The primary actor (Supervisor or Project Member) selects the desired priority level or enters the numeric value for the task.
	9.	The primary actor (Supervisor or Project Member) confirms the priority setting by clicking on the or "Confirm" button.

	10.	The system updates the task details to reflect the new priority level and notifies the relevant parties.
	11.	The system displays a confirmation message indicating that the priority of the task has been successfully set.
Alternative Flow	N/A	
Quality Requirements	1. Performance: The system should assign members to tasks quickly, even within projects or lists with a large number of existing tasks and members. 2. Reliability: The member assignment process should be reliable, ensuring that the assigned member is accurately recorded and visible to authorized users. 3. Usability: The member assignment option should be user-friendly, allowing supervisors and project members to easily assign and view assigned members. 4. Security: The system should verify the permissions of the primary actor (Supervisor or Project Member) before allowing the member assignment to prevent unauthorized assignments. 5. Availability: The member assignment feature should be available and functional at all times for authorized primary actors (Supervisor or Project Member).	

7.1.11 Set Due Date

Use Case	Set Due Date
Goal	Set the due date for a task within a project
Pre-conditions	The primary actor has appropriate permissions.
Post conditions	Project member is successfully assigned to the task.

<table>
<tr><td>Success End Condition</td><td colspan="2">The due date of the task is successfully set.</td></tr>
<tr><td>Failed End Condition</td><td colspan="2">Project member is not assigned to the task.</td></tr>
<tr><td>Primary Actors:

Secondary Actors:</td><td colspan="2">Supervisor, Project Member

N/A</td></tr>
<tr><td>Trigger</td><td colspan="2">The primary actor (Supervisor or Project Member) wants to set the due date of a task.</td></tr>
<tr><td rowspan="8">Main Success Flows</td><td>Step</td><td>Action</td></tr>
<tr><td>1. 1</td><td>Supervisor navigates to the project dashboard</td></tr>
<tr><td>2.</td><td>Supervisor selects the project</td></tr>
<tr><td>3.</td><td>Supervisor selects the task to update the due date</td></tr>
<tr><td>4.</td><td>Supervisor updates the due date or change the due date of the task</td></tr>
<tr><td>5.</td><td>Supervisor selects ‘confirm’ to save the changes</td></tr>
<tr><td>6.</td><td>The due date of the task is updated</td></tr>
<tr><td>Alternative Flow</td><td colspan="2">N/A</td></tr>
<tr><td>Quality Requirements</td><td colspan="2">1. The system should display the current due date of the task.
2. The system should provide a date picker for selecting the due date.
3. The system should validate the due date selection and update the task accordingly.</td></tr>
</table>

7.1.12 Add Description

Use Case	Add Description	
Goal	Add a description to provide additional details or context to a task.	
Pre-conditions	The primary actor has appropriate permissions.	
Post conditions	The description is successfully added to the task.	
Success End Condition	The primary actor has appropriate permissions.	
Failed End Condition	The description is not added to the task.	
Primary Actors: Secondary Actors:	Supervisor, Project Member N/A	
Trigger	The primary actor (Supervisor or Project Member) wants to add a description to a task.	
Main Success Flows	Step	Action
	1.	The primary actor (Supervisor or Project Member) navigates to the project or list containing the task.
	2.	The system displays the project or list interface, showing the existing tasks and options.
	3.	The primary actor (Supervisor or Project Member) locates the task to which they want to add a description within the interface.
	4.	The primary actor (Supervisor or Project Member) selects the task to access the task details.
	5.	The system displays the task details, including any existing description or an option to add a description.
	6.	The primary actor (Supervisor or Project Member) clicks on the "Add Description" or "Edit Description" option.
	7.	The system presents a text field or editor where the primary actor can enter the description.

	8.	The primary actor (Supervisor or Project Member) enters the desired description for the task.
	9.	The primary actor (Supervisor or Project Member) confirms the addition of the description by clicking on the "Confirm" button.
	10.	The system updates the task details to include the new description and notifies the relevant parties.
	11.	The system displays a confirmation message indicating that the description has been successfully added to the task.
Alternative Flows	Step	Action
	6a	If the primary actor (Supervisor or Project Member) decides not to add a description:
		6a1 The primary actor (Supervisor or Project Member) cancels the description addition process.
		6a2. The system returns to the task details without making any changes to the description.
Quality Requirements	1. The system should provide a text editor for entering the description. 2. The system should allow formatting options for the description. 3. The system should display the current description of the task.	

7.1.13 Create Board

Use Case	Create Board
Goal	Create a new board within the project.
Pre-conditions	The primary actor has appropriate permissions.
Post conditions	The new board is successfully created in the project.

Success End Condition	A new board is successfully created within the project.	
Failed End Condition	The new board is not created in the project.	
Primary Actors: Secondary Actors:	Supervisor, Project Member N/A	
Trigger	The primary actor (Supervisor or Project Member) wants to create a new board.	
Main Success Flows	Step	Action
	1.	The primary actor (Supervisor or Project Member) navigates to the project where they want to create a new board.
	2.	The primary actor (Supervisor or Project Member) navigates to the project where they want to create a new board.
	3.	The primary actor (Supervisor or Project Member) locates the option to create a new board within the interface.
	4.	The primary actor (Supervisor or Project Member) selects the "Create New Board" option.
	5.	The system presents a form or dialog where the primary actor can enter the details of the new board.
	6.	The primary actor (Supervisor or Project Member) enters the required information, such as the board name, description, or any other relevant details.
	7.	The primary actor (Supervisor or Project Member) confirms the creation of the new board by clicking on the "Confirm" button.
	8.	The system creates the new board within the project, assigns appropriate permissions, and updates the project interface.

	9.	The system displays a confirmation message indicating that the new board has been successfully created.
Alternative Flows	Step	Action
	6a	If the primary actor (Supervisor or Project Member) decides not to create a new board:
		6a1. The primary actor (Supervisor or Project Member) cancels the board creation process.
		6a2. The primary actor (Supervisor or Project Member) cancels the board creation process.
Quality Requirements	1. The system should validate that a unique name is provided for the new board. 2. The system should provide options for categorizing the new board. 3. The system should allow customization of the board's layout and design.	

7.1.14 Add Status

Use Case	Add Status
Goal	Add a status to indicate the progress of a task.
Pre-conditions	The primary actor has appropriate permissions.
Post conditions	The status is successfully added to the task.
Success End Condition	The status is successfully added to the task.
Failed End Condition	N/A
Primary Actors:	Supervisor, Project Member

Secondary Actors:	N/A	
Trigger	The primary actor (Supervisor or Project Member) wants to add a status to a task.	
Main Success Flows	Step	Action
	1.	The primary actor (Supervisor or Project Member) navigates to the project or list containing the task.
	2.	The system displays the project or list interface, showing the existing tasks and options.
	3.	The primary actor (Supervisor or Project Member) locates the task to which they want to add a status within the interface.
	4.	The primary actor (Supervisor or Project Member) selects the task to access the task details.
	5.	The system displays the task details, including any existing status or an option to add a status.
	6.	The primary actor (Supervisor or Project Member) clicks on the "Add Status" or "Edit Status" option.
	7.	The system presents a list of predefined statuses or allows the primary actor to enter a custom status.
	8.	The primary actor (Supervisor or Project Member) selects the desired status from the list or enters a custom status.
	9.	The primary actor (Supervisor or Project Member) confirms the addition of the status by clicking on the "Confirm" button.
	10.	The system updates the task details to include the new status and notifies the relevant parties.
	11.	The system displays a confirmation message indicating that the status has been successfully added to the task.
Alternative Flows	Step	Action
	6a	If the primary actor (Supervisor or Project Member) decides not to add a status:

		6a1.The primary actor (Supervisor or Project Member) cancels the status addition process.
		6a2.The system returns to the task details without making any changes to the status.
Quality Requirements	1. The system should provide a clear and intuitive interface for selecting the status. 2. The system should allow customization of status options based on project requirements. 3. The system should validate that a status is selected before saving.	

7.1.15 Add Comments

Use Case	Add Comments	
Goal	Add comments to provide additional information or updates to a task.	
Pre-conditions	The primary actor has appropriate permissions.	
Post conditions	The comments are successfully added to the task.	
Success End Condition	The status is successfully added to the task.	
Failed End Condition	N/A	
Primary Actors: Secondary Actors:	Supervisor, Project Member N/A	
Trigger	The primary actor (Supervisor or Project Member) wants to add a comment to a task.	
Main Success Flows	Step	Action

	1.	The primary actor (Supervisor or Project Member) navigates to the project or list containing the task.
	2.	The system displays the project or list interface, showing the existing tasks and options.
	3.	The primary actor (Supervisor or Project Member) locates the task to which they want to add a comment within the interface.
	4.	The primary actor (Supervisor or Project Member) selects the task to access the task details.
	5.	The system displays the task details, including any existing comments or an option to add a comment.
	6.	The primary actor (Supervisor or Project Member) clicks on the "Add Comment" option.
	7.	The system presents a text field or dialog where the primary actor can enter the comment text.
	8.	The primary actor (Supervisor or Project Member) enters the comment text providing relevant information or updates.
	9.	The primary actor (Supervisor or Project Member) confirms the addition of the comment by clicking on the "Confirm" button.
	10.	The system updates the task details to include the new comment and notifies the relevant parties.
	11.	The system displays a confirmation message indicating that the comment has been successfully added to the task.
Alternative Flows	Step	Action
	6a	If the primary actor (Supervisor or Project Member) decides not to add a comment.
		6a1. The primary actor (Supervisor or Project Member) cancels the comment addition process.
Quality Requirements	1.	The system should provide a user-friendly interface for entering comments.

	2. The system should support formatting options for comments (e.g., text styling, links, etc.). 3. The system should ensure the comments are associated with the correct task.

7.1.16 Share Documents

Use Case	Share Documents	
Goal	Share documents with other project members.	
Pre-conditions	The primary actor has appropriate permissions.	
Post conditions	Documents are successfully shared with the project members.	
Success End Condition	The document is successfully shared with the intended recipients.	
Failed End Condition	N/A	
Primary Actors: Secondary Actors:	Supervisor, Project Member N/A	
Trigger	The primary actor (Supervisor or Project Member) wants to share a document.	
Main Success Flows	Step	Action
	10.	The primary actor (Supervisor or Project Member) navigates to the project or list containing the document.
	11.	The system displays the project or list interface, showing the existing documents and options.
	12.	The primary actor (Supervisor or Project Member) locates the document they want to share within the interface.
	13.	The system presents a "Share" option for the selected document.

	14.	The primary actor (Supervisor or Project Member) clicks on the "Share" option.
	15.	The system displays a dialog or form to enter the email addresses or usernames of the intended recipients.
	16.	The primary actor (Supervisor or Project Member) enters the email addresses or usernames of the recipients.
	17.	The primary actor (Supervisor or Project Member) confirms the sharing action by clicking on the "Confirm" button.
	18.	The system sends email notifications or generates access links to the shared document for the intended recipients.
	19.	The system updates the document sharing status, indicating that it has been shared with the specified recipients.
	20.	The system displays a confirmation message indicating that the document has been successfully shared.
Alternative Flows	Step	Action
	6a	If the primary actor (Supervisor or Project Member) wants to share the document with all project members:
		6a1. The primary actor selects an option to share the document with all project members.
		6a2. The system automatically includes all project members as recipients.
Quality Requirements	1. The system should provide a secure and reliable document sharing mechanism. 2. The system should allow the Supervisor to select multiple documents to share. 3. The system should provide an efficient way to select project members for document sharing. 4. The system should ensure that only authorized project members have access to the shared documents.	

7.1.17 Download Documents

Use Case	Download Documents	
Goal	Download a document for offline access or storage.	
Pre-conditions	The primary actor has appropriate permissions.	
Post conditions	Supervisor or Project Member receives the notifications.	
Success End Condition	The document is successfully downloaded to the primary actor's device or storage.	
Failed End Condition	N/A	
Primary Actors: Secondary Actors:	Supervisor, Project Member N/A	
Trigger	The primary actor (Supervisor or Project Member) wants to download a document.	
Main Success Flows	Step	Action
	1	The primary actor (Supervisor or Project Member) navigates to the project or list containing the document.
	2	The system displays the project or list interface, showing the existing documents and options.
	3	The primary actor (Supervisor or Project Member) locates the document they want to download within the interface.
	4	The primary actor (Supervisor or Project Member) selects the document to access the document details or options.
	5	The system presents a "Download" option for the selected document.
	6	The primary actor (Supervisor or Project Member) clicks on the "Download" option.

	7	The system initiates the download process and transfers the document to the primary actor's device or storage.
	8	The primary actor (Supervisor or Project Member) receives the downloaded document and can access it offline or store it as needed.
Alternative Flow	N/A	
Quality Requirements	1. The system should provide a secure and reliable document download mechanism. 2. The system should allow the Supervisor or Project Member to select multiple documents for download. 3. The system should ensure that only authorized users have access to download documents. 4. The system should provide an efficient and fast download process.	

7.1.18 Messaging

Use Case	Messaging
Goal	Easily collaboration with the project member, upload file and download file.
Pre-conditions	Must have to be included as a project member, or supervisor.
Post conditions	Identification update is generated and available for viewing.
Success End Condition	Identification update is successfully generated and available.
Failed End Condition	N/A
Primary Actors: Secondary Actors:	Supervisor, Project Members Supervisor, Project Members

Trigger	Press the Message Icon.	
Main Success Flows	Step	Action
	1	Select List.
	2	Press the Message Icon.
	3	Write text and share docs.
	4	For send click on send icon.
Alternative Flow	N/A	
Quality Requirements	1. Delivery within 5 second. 2. Message seen Understanding with good interface in blue color.	

7.1.19 Get Notifications

Use Case	Get Notifications
Goal	Receive timely notifications about project updates or relevant activities.
Pre-conditions	The primary actor has subscribed to receive notifications and has appropriate permissions.
Post conditions	The new board is successfully created in the project.
Success End Condition	The primary actor receives the notification with the relevant information.
Failed End Condition	N/A
Primary Actors: Secondary Actors:	Supervisor, Project Member N/A

Trigger	The system detects a project update or activity that requires notification.	
Main Success Flows	Step	Action
	1	The primary actor receives the notification with the relevant information.
	2	The system generates a notification containing the relevant information, such as project name, task details, user name, or comment text.
	3	The system checks the notification preferences and permissions of the primary actor (Supervisor or Project Member).
	4	The system sends the notification to the primary actor via their preferred communication channel, such as email, in-app notification, or mobile push notification.
	5	The primary actor (Supervisor or Project Member) receives the notification.
	6	The primary actor reviews the notification and takes appropriate action based on the information provided
Alternative Flows	Step	Action
	3a	If the primary actor (Supervisor or Project Member) has specified custom notification settings:
		3a1. The system checks the primary actor's customized notification preferences and channels.
		3a2. The system sends the notification through the specified channels according to the primary actor's preferences.
Quality Requirements	1. The system should provide customizable notification settings to allow users to choose their preferred notification types.	

	2. The system should ensure timely delivery of notifications to the Supervisor or Project Member. 3. The system should provide clear and concise notifications with relevant information. 4. The system should support various notification channels (e.g., email, in-app notifications).

7.1.20 Notify Update Progress

Use Case	Notify Update Progress	
Goal	Notify progress updates to relevant project members.	
Pre-conditions	The primary actor has appropriate permissions to update progress and has subscribed to receive notifications.	
Post conditions	Supervisor and Project Member receive the progress update.	
Success End Condition	A new board is successfully created within the project.	
Failed End Condition	The new board is not created in the project.	
Primary Actors: Secondary Actors:	Supervisor, Project Member N/A	
Trigger	The primary actor (Supervisor or Project Member) updates the progress of a task or project.	
Main Success Flows	Step	Action
	1	The primary actor (Supervisor or Project Member) updates the progress of a task or project.
	2	The system detects the progress update and generates a notification.

	3	The system checks the notification preferences and permissions of relevant project members.
	4	The system sends notifications to the relevant project members regarding the progress update.
	5	The relevant project members receive the notifications containing the progress update information.
Alternative Flows	Step	Action
	6a	If the primary actor (Supervisor or Project Member) wants to notify specific project members:
		6a1. The primary actor selects the specific project members to notify about the progress update.
		6a2. The system checks the notification preferences and permissions of the selected project members.
Quality Requirements	1. The system should promptly notify the Supervisor and Project Member of the progress update. 2. The progress update notification should include relevant information such as task/project name, completed percentage, and any additional notes. 3. The system should handle concurrent progress updates accurately to avoid conflicts or data loss. 4. The notification should be delivered to the Supervisor and Project Member through their preferred communication channels (e.g., email, in-app notification).	

7.2 Description for Administrator

7.2.1 Control Privacy

Use Case	Control Privacy
Goal	Manage privacy settings and access permissions for users or project members.
Pre-conditions	The Administrator has appropriate privileges and access to the system.

<table>
<tr><td>Post conditions</td><td colspan="2">Privacy settings are updated and enforced in the system.</td></tr>
<tr><td>Success End Condition</td><td colspan="2">Privacy settings and access permissions are successfully modified.</td></tr>
<tr><td>Failed End Condition</td><td colspan="2">Privacy settings are not updated or enforced as intended.</td></tr>
<tr><td>Primary Actors:

Secondary Actors:</td><td colspan="2">Administrator

N/A</td></tr>
<tr><td>Trigger</td><td colspan="2">The Administrator needs to modify privacy settings or access permissions.</td></tr>
<tr><td rowspan="4">Main Success Flows</td><td>Step</td><td>Action</td></tr>
<tr><td>1.</td><td>Administrator accesses the privacy control interface</td></tr>
<tr><td>2.</td><td>Administrator selects the desired privacy settings or options</td></tr>
<tr><td>3.</td><td>The system updates the privacy settings and enforces them</td></tr>
<tr><td>Alternative Flow</td><td colspan="2">N/A</td></tr>
<tr><td>Quality Requirements</td><td colspan="2">1. The system should provide a user-friendly interface for privacy control.
2. Privacy settings should be granular and customizable.
3. The system should enforce the privacy settings consistently.
4. Privacy settings should be properly documented and communicated
5. The system should log any privacy control actions for audit purposes
6. The privacy control interface should be secure and protected</td></tr>
</table>

7.2.2 Control Updating

Use Case	Control Privacy	
Goal	Control and manage software updates in the system.	
Pre-conditions	Administrator has appropriate access and permissions.	
Post conditions	Software updates are controlled and managed in the system.	
Success End Condition	Software updates are successfully controlled and managed.	
Failed End Condition	Software updates are not controlled or managed as intended.	
Primary Actors: Secondary Actors:	Administrator N/A	
Trigger	Administrator initiates update control actions.	
Main Success Flows	Step	Action
	1.	Administrator accesses the update control interface.
	2.	Administrator selects the desired update control options.
	3.	The system applies the update control actions.
Alternative Flow	N/A	

Quality Requirements	1. The system should provide a user-friendly interface for update control. 2. Update control options should be comprehensive and flexible. 3. The system should enforce update control consistently 4. Update control actions should be properly documented and auditable. 5. The update control interface should be secure and protected.

7.3 Description For System

7.3. 1 Manage Date/Time

Use Case	Control Privacy
Goal	Manage the date and time settings of the system.
Pre-conditions	System is operational and capable of managing date/time.
Post conditions	Date and time settings are successfully managed by the system.
Success End Condition	Date and time settings are successfully managed by the system.
Failed End Condition	N/A
Primary Actors: Secondary Actors:	System N/A
Trigger	System initialization or user request to change date/time settings.

Main Success Flows	Step	Action
	1.	System initializes or user requests to change date/time settings.
	2.	System displays the current date and time settings.
	3.	User provides input to change the date and time settings.
	4.	System verifies the validity of the new settings
	5.	System updates and applies the new date and time settings
Alternative Flow	N/A	
Quality Requirements	1. The system should provide an intuitive and user-friendly interface for managing date/time settings. 2. The system should validate the input for date and time settings to ensure they are within acceptable ranges. 3. The system should handle time zone conversions accurately and seamlessly. 4. Date and time changes should be reflected across the system in a synchronized manner. 5. The system should have the capability to automatically synchronize with external time sources.	

7.3.2 Manage Notification

Use Case	Control Privacy
Goal	Manage the notification settings of the system
Pre-conditions	System is operational and capable of managing notifications.
Post conditions	Notification settings are successfully managed by the system.
Success End Condition	Notification settings are successfully managed by the system.

<table>
<tr><td>Failed End Condition</td><td colspan="2">N/A</td></tr>
<tr><td>Primary Actors:

Secondary Actors:</td><td colspan="2">System

N/A</td></tr>
<tr><td>Trigger</td><td colspan="2">System initialization or user request to manage notifications</td></tr>
<tr><td rowspan="5">Main Success Flows</td><td>Step</td><td>Action</td></tr>
<tr><td>1</td><td>System initializes or user requests to manage notifications.</td></tr>
<tr><td>2</td><td>System displays the current notification settings.</td></tr>
<tr><td>3</td><td>User provides input to enable/disable or customize settings.</td></tr>
<tr><td>4</td><td>System updates and applies the new notification settings.</td></tr>
<tr><td>Alternative Flow</td><td colspan="2">N/A</td></tr>
<tr><td>Quality Requirements</td><td colspan="2">1. The system should provide an intuitive interface for managing notification settings.
2. The system should allow users to enable/disable notifications for specific events or modules.
3. The system should provide options for customizing notification preferences.
4. The system should support multiple notification channels, such as email, push notifications, or SMS.
5. The system should deliver notifications in a timely and reliable manner</td></tr>
</table>

	6. The system should provide the ability to schedule notifications for specific times or recurring intervals.

7.3.3 Manage Notification

Use Case	Control Privacy	
Goal	Manage the notification settings of the system	
Pre-conditions	System is operational and capable of managing notifications.	
Post conditions	Notification settings are successfully managed by the system.	
Success End Condition	Notification settings are successfully managed by the system.	
Failed End Condition	N/A	
Primary Actors: Secondary Actors:	System N/A	
Trigger	System initialization or user request to manage notifications	
Main Success Flows	Step	Action
	1	System initializes or user requests to manage notifications.
	2	System displays the current notification settings.

	3	User provides input to enable/disable or customize settings.
	4	System updates and applies the new notification settings.
Alternative Flow	N/A	
Quality Requirements	1. The system should provide an intuitive interface for managing notification settings. 2. The system should allow users to enable/disable notifications for specific events or modules. 3. The system should provide options for customizing notification preferences. 4. The system should support multiple notification channels, such as email, push notifications, or SMS. 5. The system should deliver notifications in a timely and reliable manner 6. The system should provide the ability to schedule notifications for specific times or recurring intervals.	

7.3.4 Generate Gantt Chart

Use Case	Generate Gantt Chart
Goal	Generate a Gantt chart for project scheduling.
Pre-conditions	Project data and scheduling information are available.
Post conditions	Gantt chart is generated and available for viewing.
Success End Condition	Gantt chart is successfully generated and available for viewing.
Failed End Condition	N/A

<table>
<tr><td>Primary Actors:

Secondary Actors:</td><td colspan="2">System

N/A</td></tr>
<tr><td>Trigger</td><td colspan="2">User request to generate a Gantt chart.</td></tr>
<tr><td rowspan="5">Main Success Flows</td><td>Step</td><td>Action</td></tr>
<tr><td>1</td><td>User requests to generate a Gantt chart.</td></tr>
<tr><td>2</td><td>System retrieves project data and scheduling information.</td></tr>
<tr><td>3</td><td>System generates the Gantt chart based on the project data.</td></tr>
<tr><td>4</td><td>System presents the generated Gantt chart to the user.</td></tr>
<tr><td>Alternative Flow</td><td colspan="2">N/A</td></tr>
<tr><td>Quality Requirements</td><td colspan="2">1. The system should accurately represent the project schedule in the generated Gantt chart.
2. The Gantt chart should provide a clear visualization of project tasks, durations, and dependencies.
3. The system should handle complex project schedules and dependencies.
4. The Gantt chart should be generated in a timely manner.
5. The system should allow users to customize the appearance and format of the Gantt chart.
6. The system should provide options for exporting or sharing the generated Gantt chart.</td></tr>
</table>

	7. The Gantt chart should be easily readable and understandable by project stakeholders. 8. The system should handle updates to project data and reflect the changes in the generated Gantt chart.

7.3.5 Identify Update

Use Case	Identify Update	
Goal	Generate an identification update for a project.	
Pre-conditions	Project data and status information are available.	
Post conditions	Identification update is generated and available for viewing.	
Success End Condition	Identification update is successfully generated and available.	
Failed End Condition	N/A	
Primary Actors: Secondary Actors:	System N/A	
Trigger	User request to generate an identification update.	
Main Success Flows	Step	Action
	1	User requests to generate an identification update.
	2	System retrieves project data and status information.
	3	System generates the identification update.

	4	System presents the generated identification update to the user.
Alternative Flow	N/A	
Quality Requirements	3. The system should accurately reflect the project status in the identification update. 4. The identification update should provide a concise summary of the project status 5. The system should handle updates to project data and reflect the changes in the identification update 6. The identification update should be generated in a The system should allow users to customize the content and format of the identification update 7. The identification update should be easily readable and understandable by project stakeholders 8. The system should provide options for exporting or sharing the generated identification update	

7.3.6 Generate Report

Use Case	Generate Report
Goal	Generate a comprehensive evaluation report for project members.
Pre-conditions	Project data and evaluation metrics are available.
Post conditions	Report is generated and made available.
Success End Condition	Report is successfully generated and accessible
Failed End Condition	N/A
Primary Actors: Secondary Actors:	System N/A

Trigger	System detects the need to generate a report.	
Main Success Flows	Step	Action
	1	Enter into IIT Kanban Board Software.
	2	Select the report.
	3	Click on Generate report.
	4	System will show the report containing necessary information like project name, Supervisor and teammate identity, Project Description, Assign task, date and time, Gantt Chart.
	5	System will provide final report
Alternative Flow	N/A	
Quality Requirements	1. Generate the report within 5 sec 2. Generate report in pdf format	

8. Activity diagram

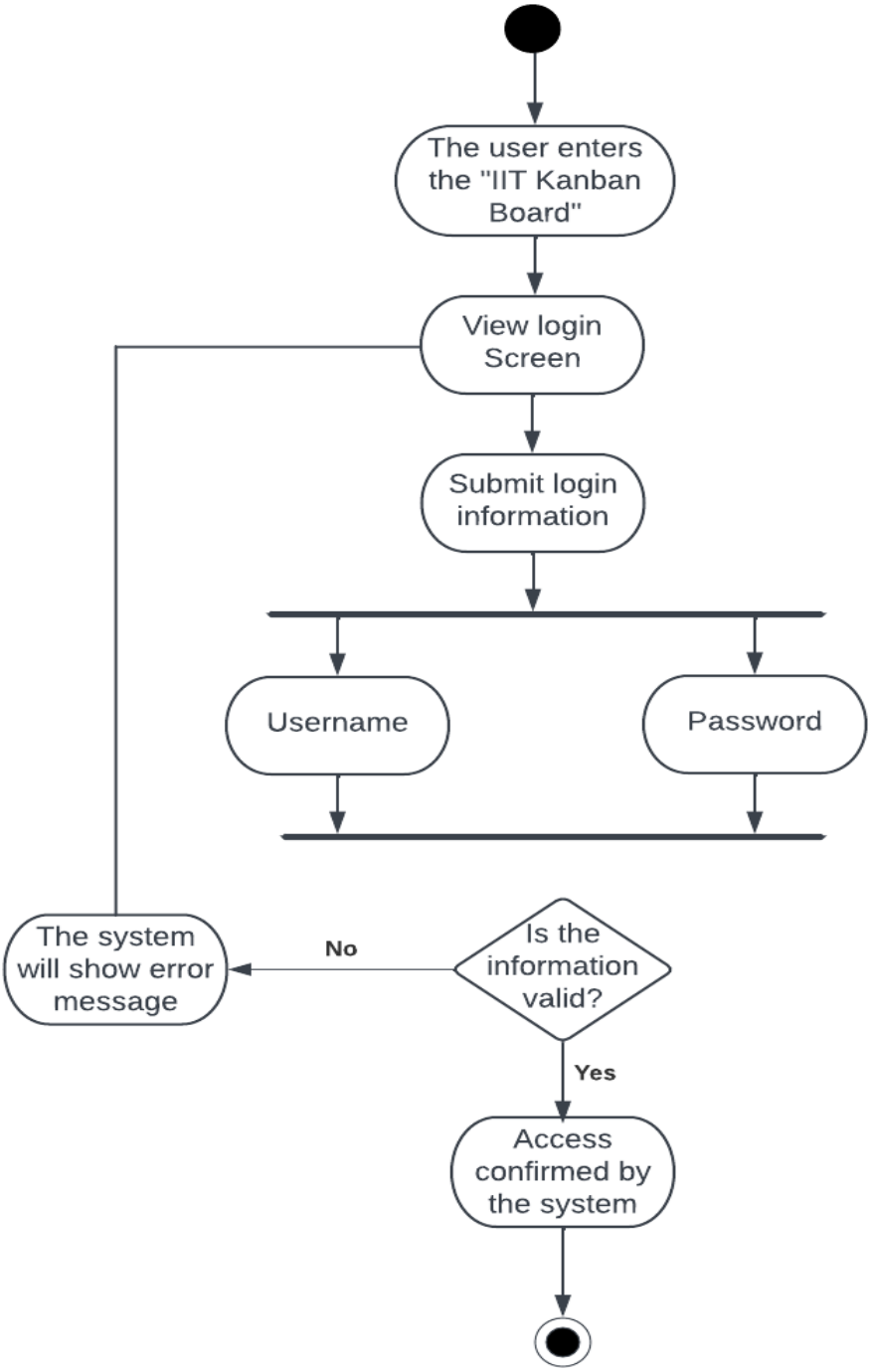


Fig 02-Enter the System

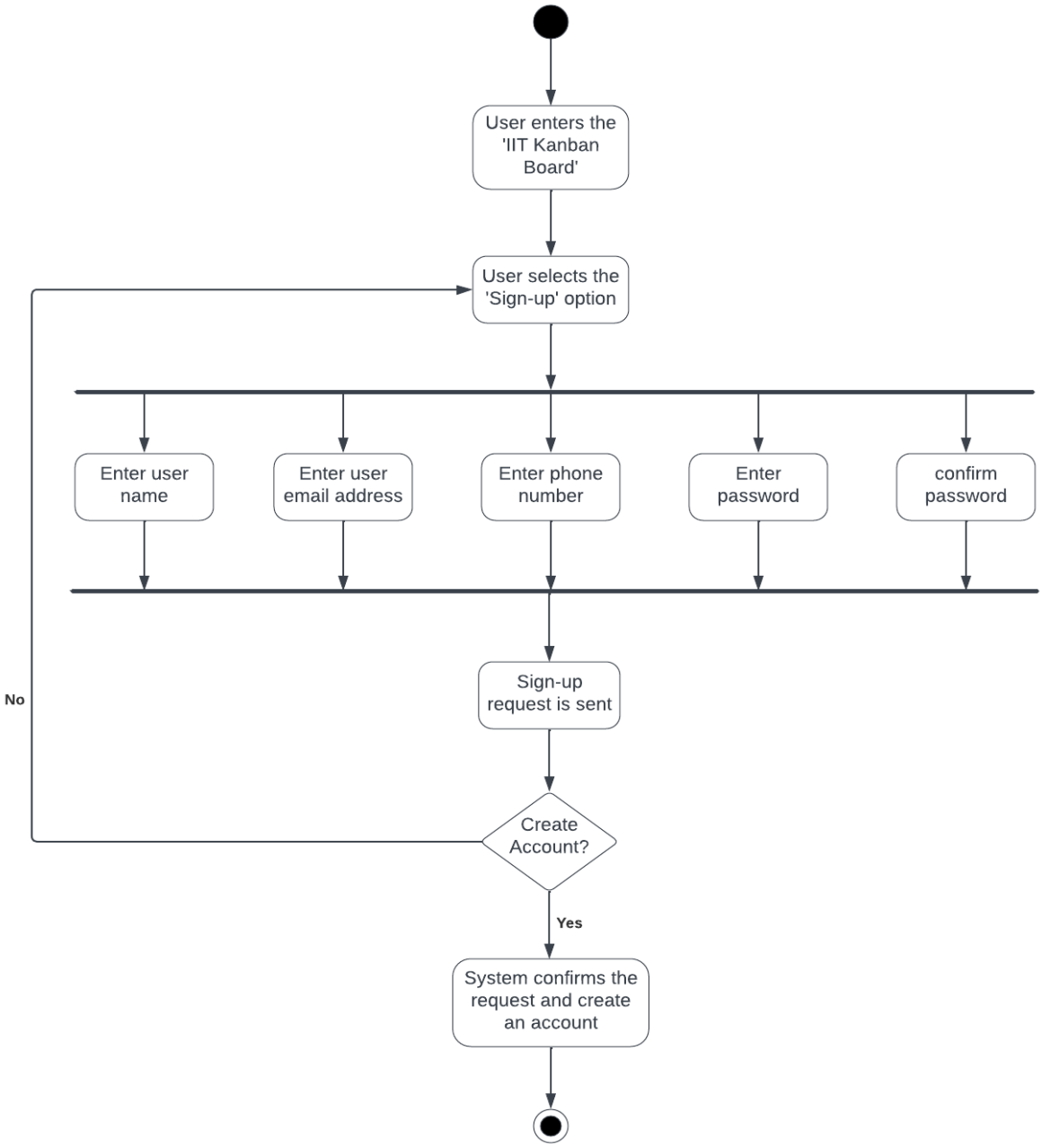


Fig 03- Sign Up

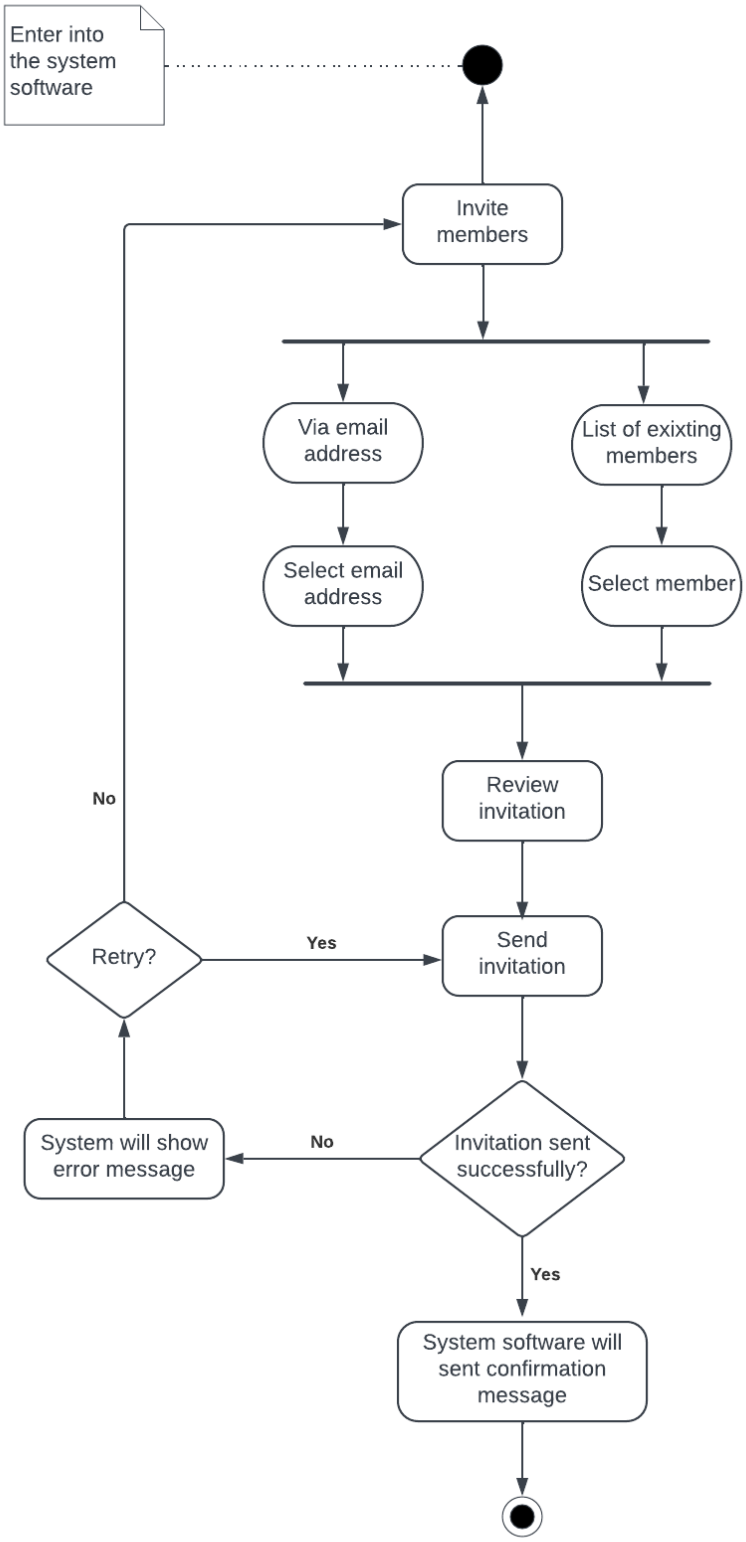


Fig 04– Invite member

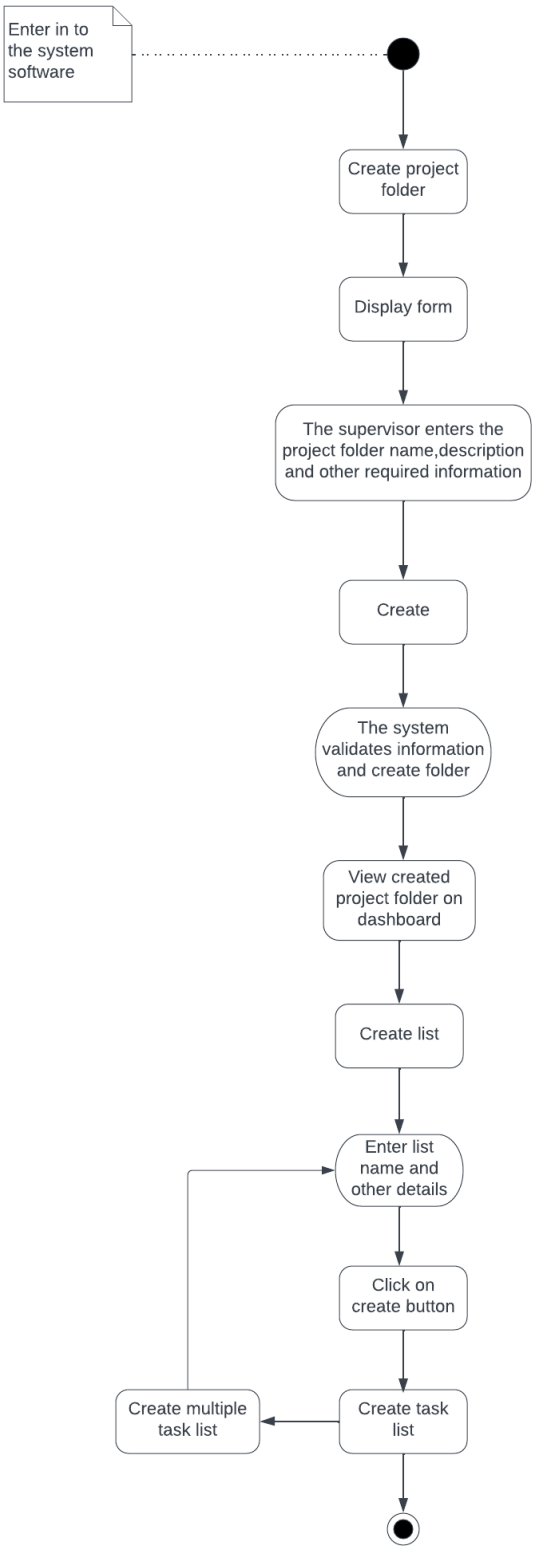


Fig 05- Create Project Folder

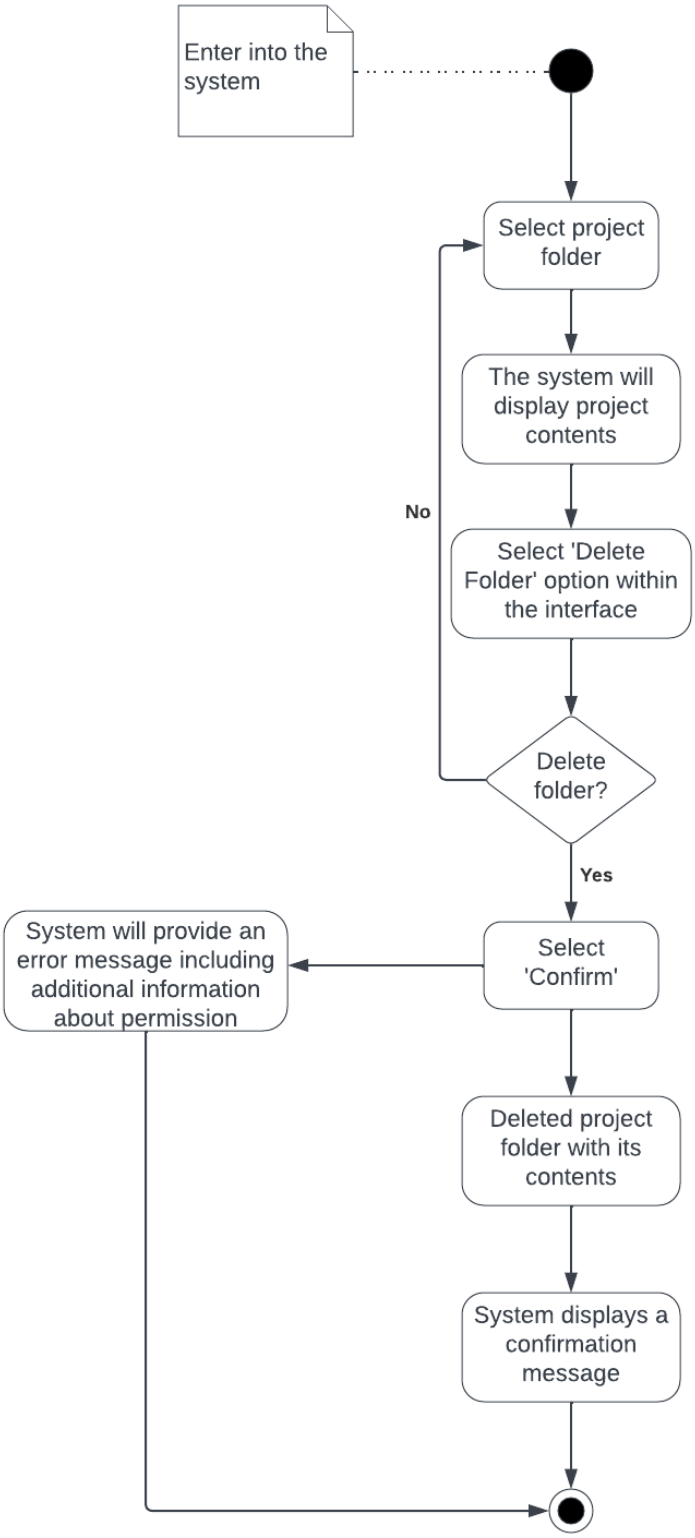


Fig 06- Delete project Folder

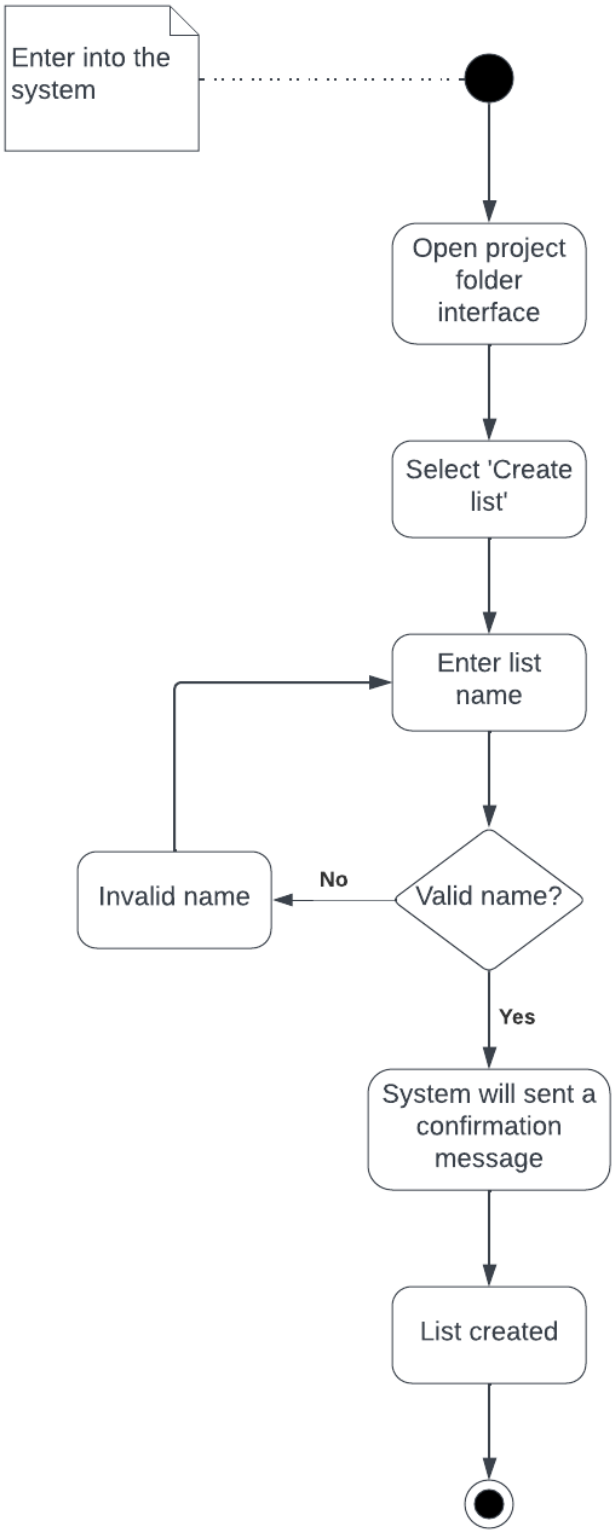


Fig 07- Create List

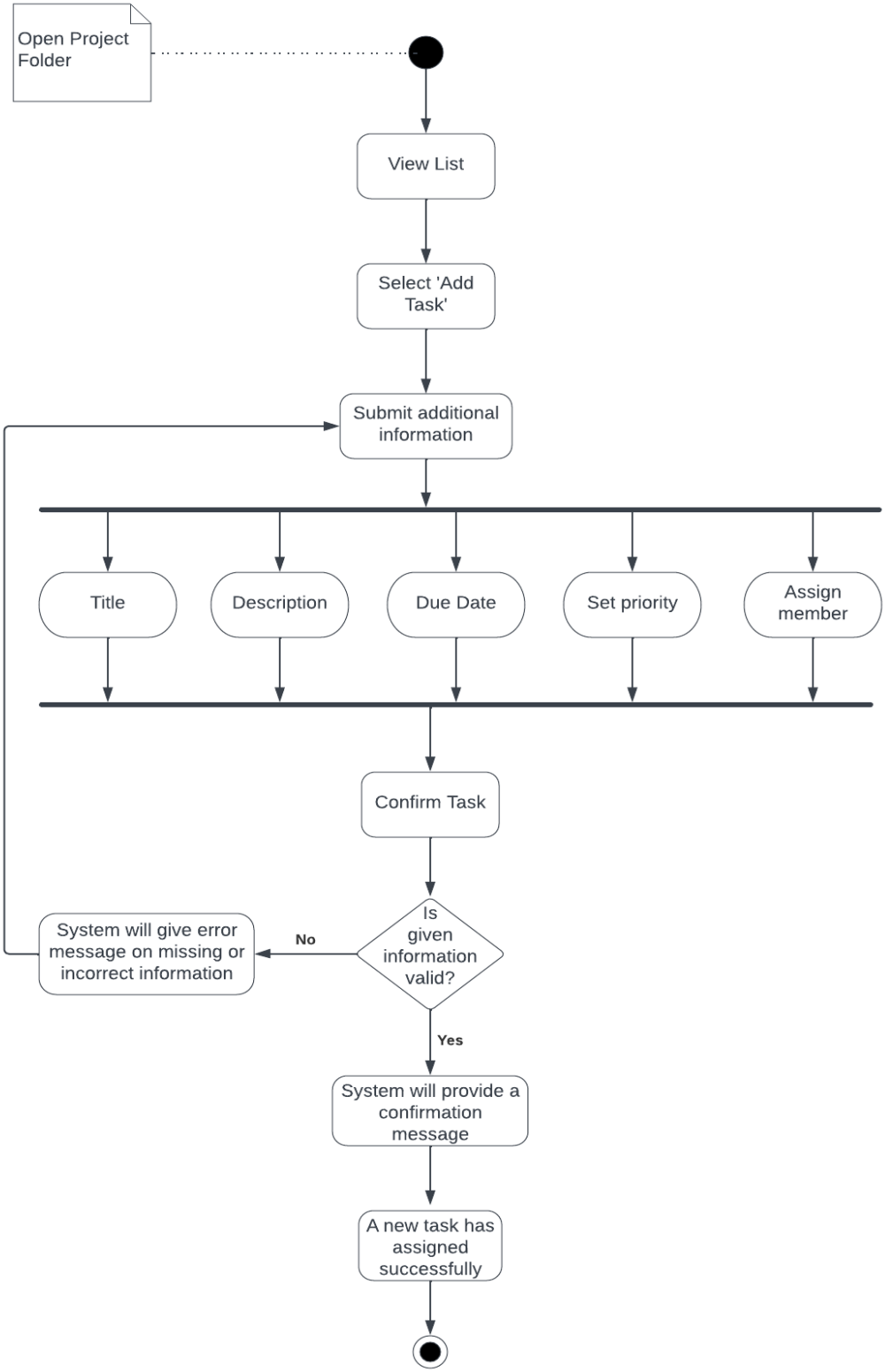


Fig 08- Add Task

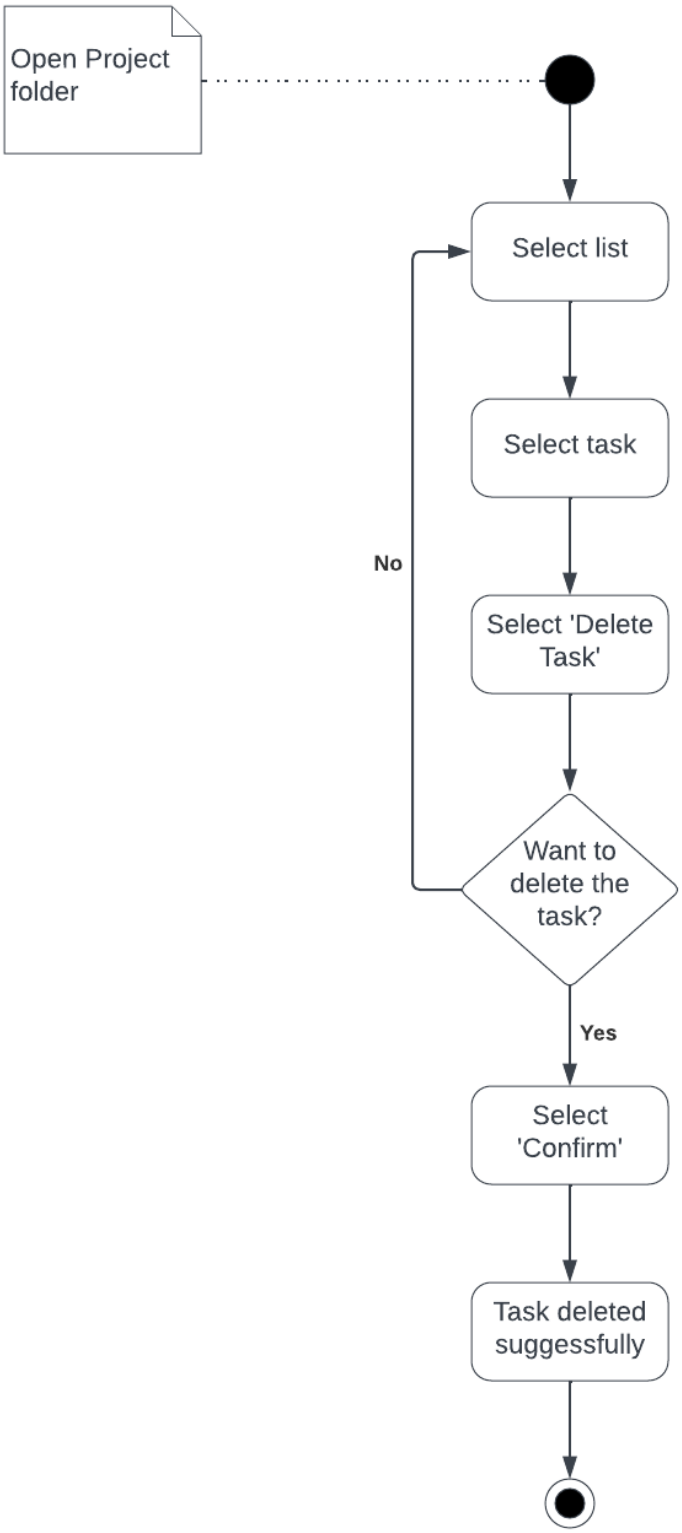


Fig 09- Delete Task

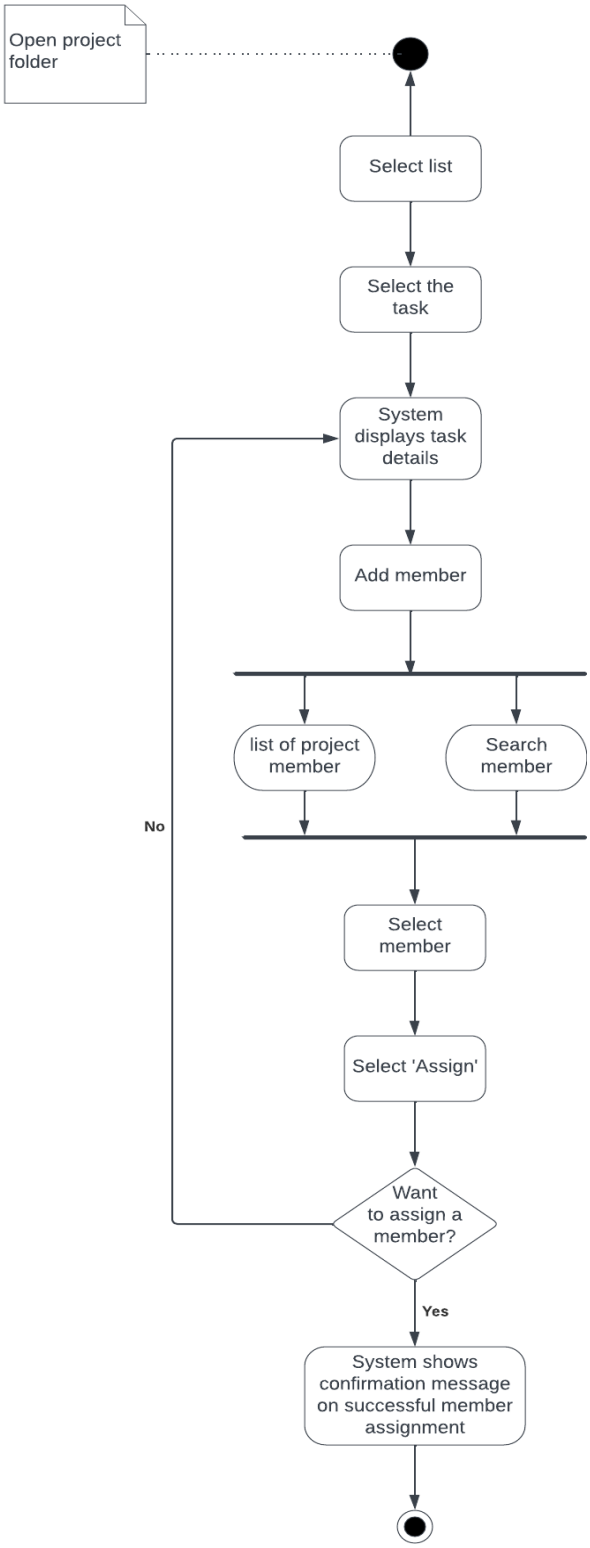


Fig 10 – Assign Member

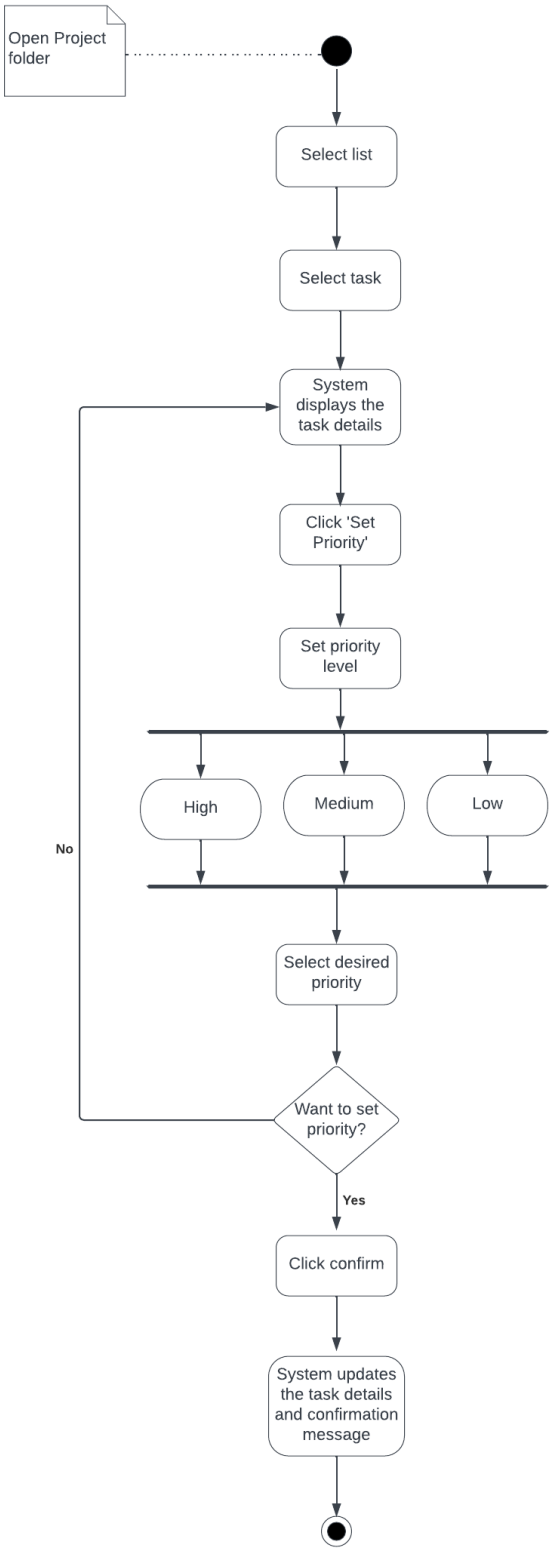


Fig 11 -Set Priority

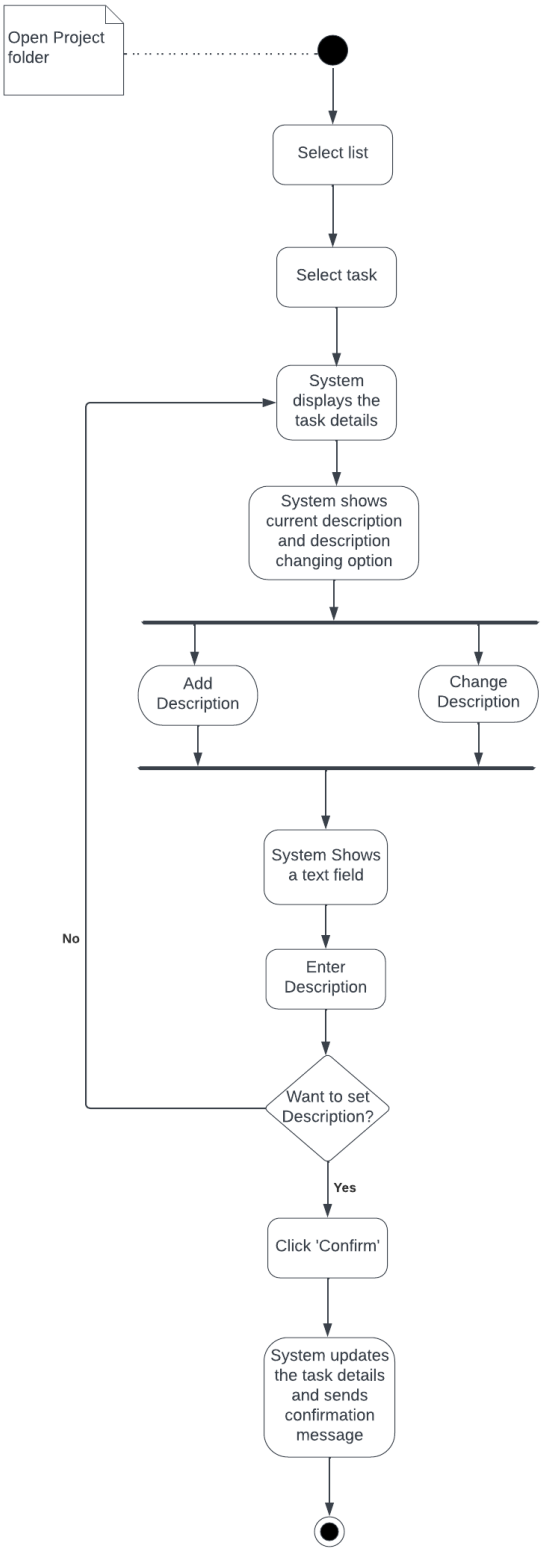


Fig 12- Add Description

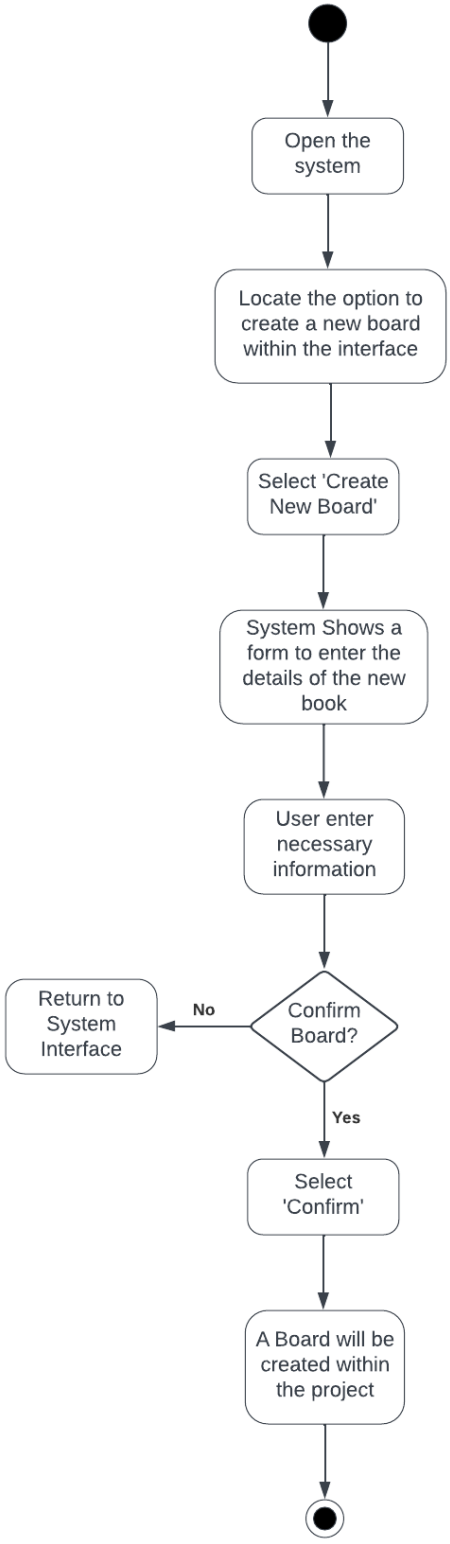


Fig 13 – Create Board

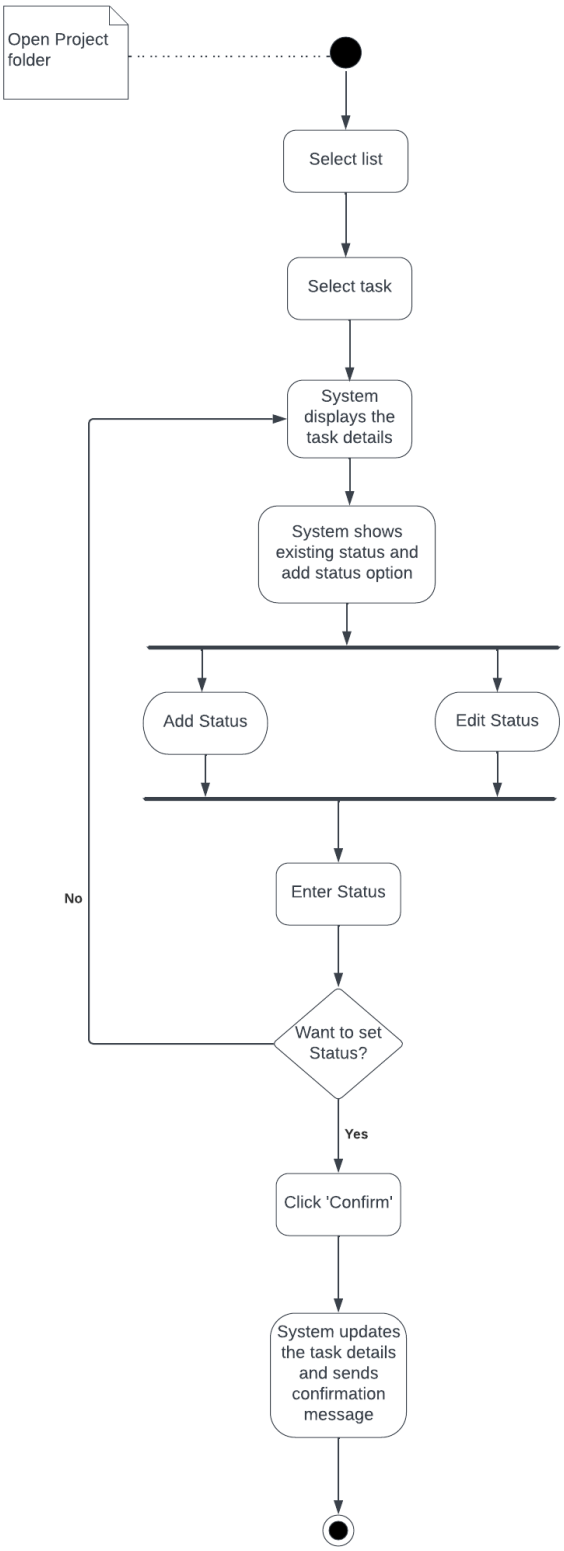


Fig 14-Add Status

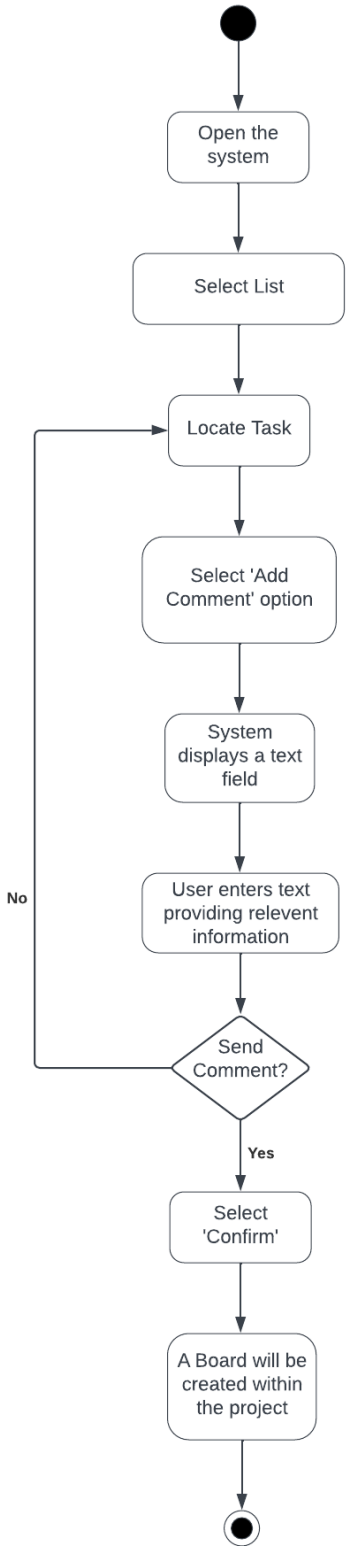


Fig 15-Add Comments

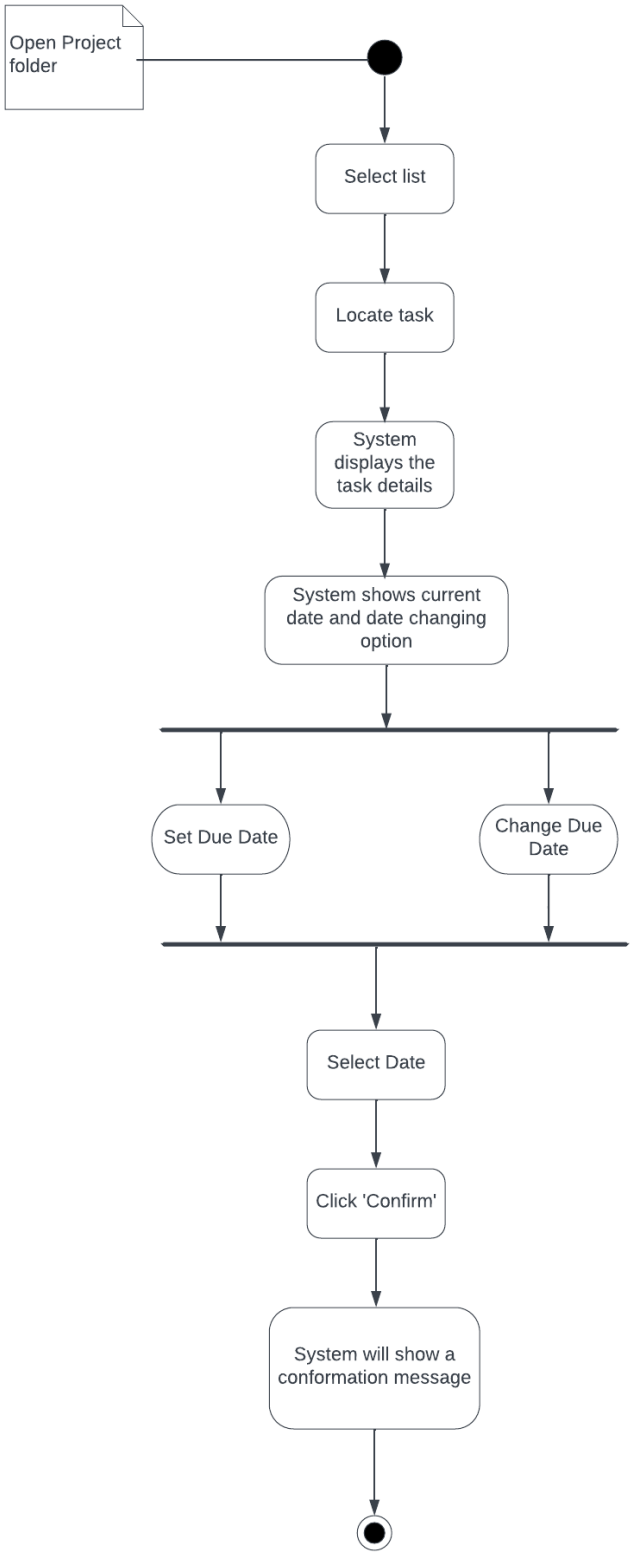


Fig 16- Set Due Date

Fig 17- Generate Report

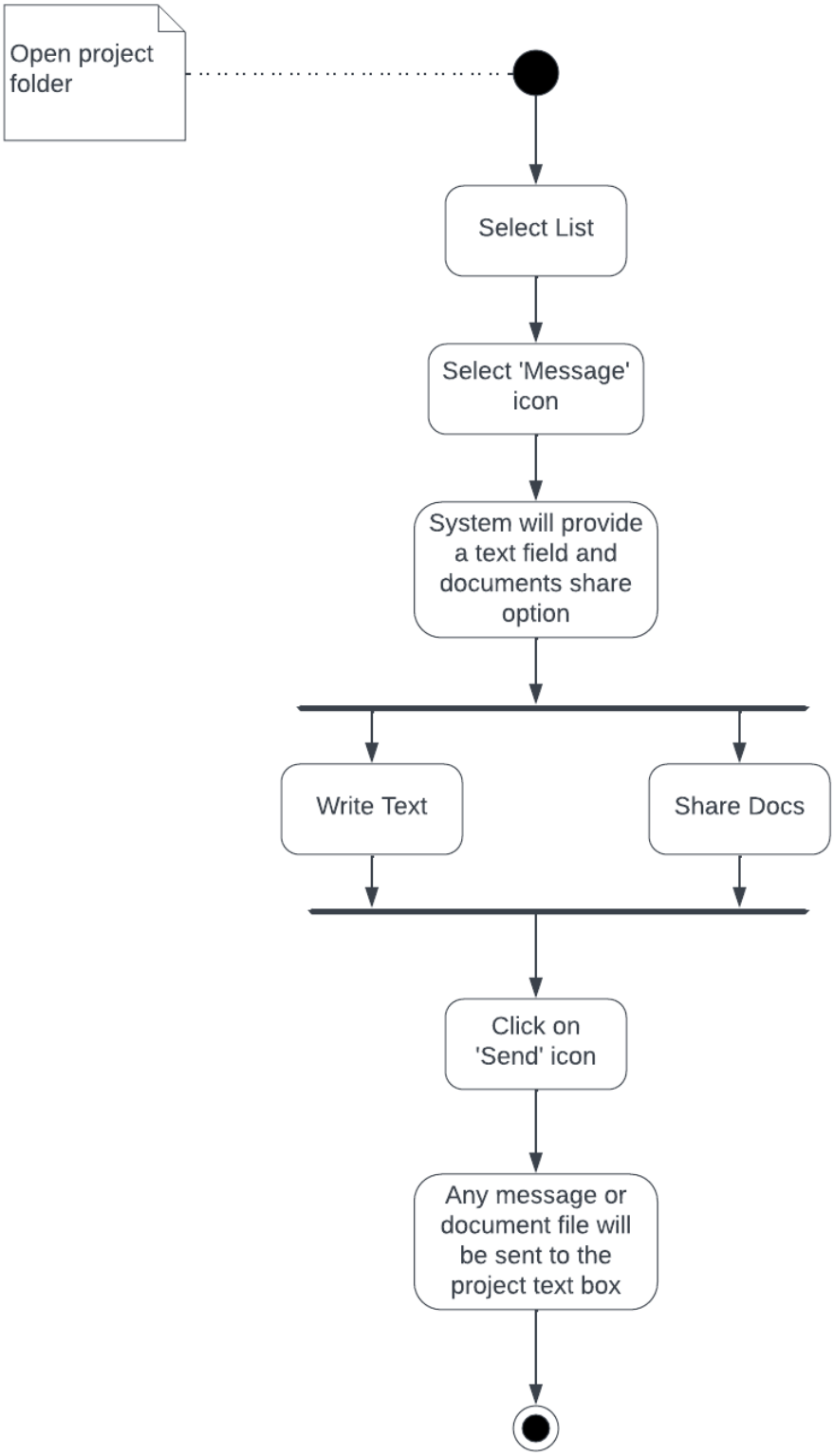


Fig 18- Messaging

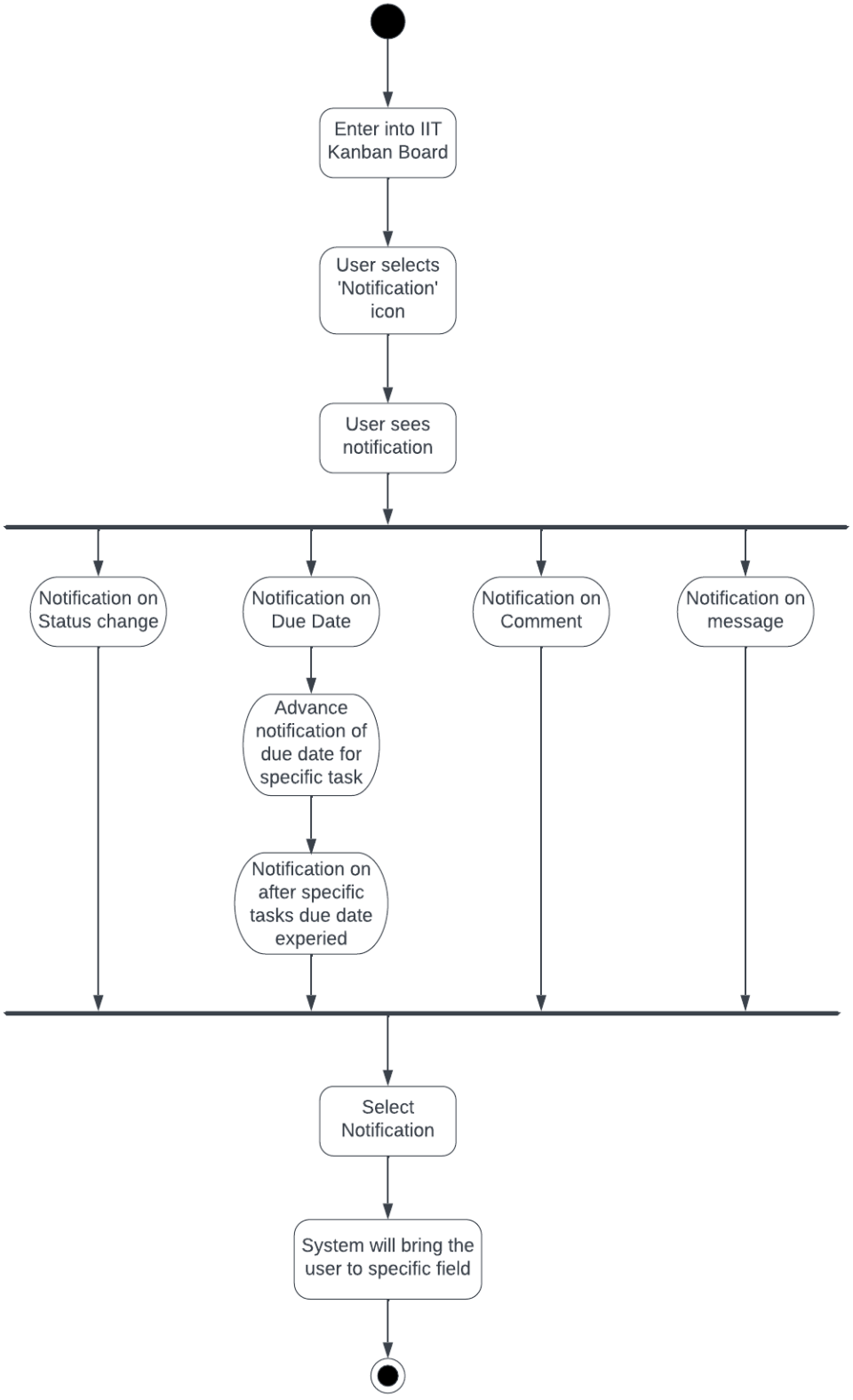


Fig 19- Notification

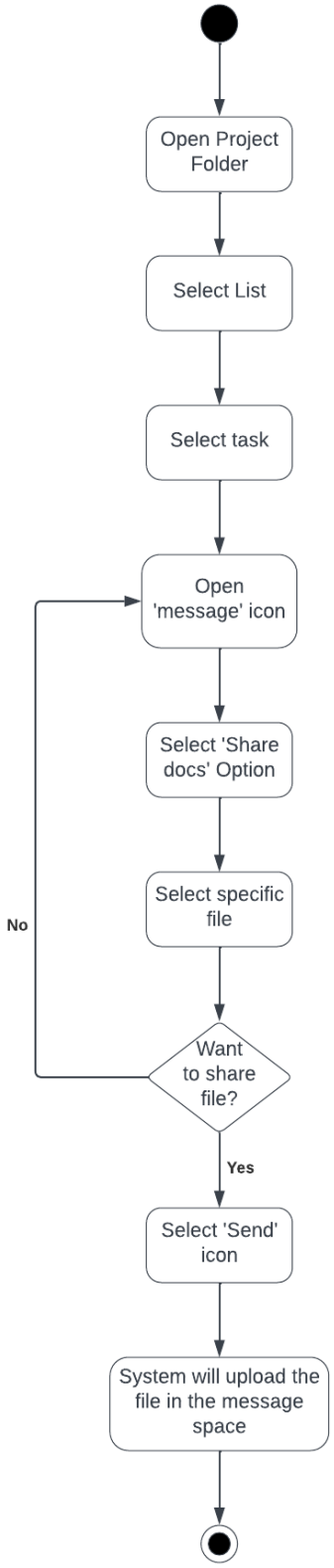
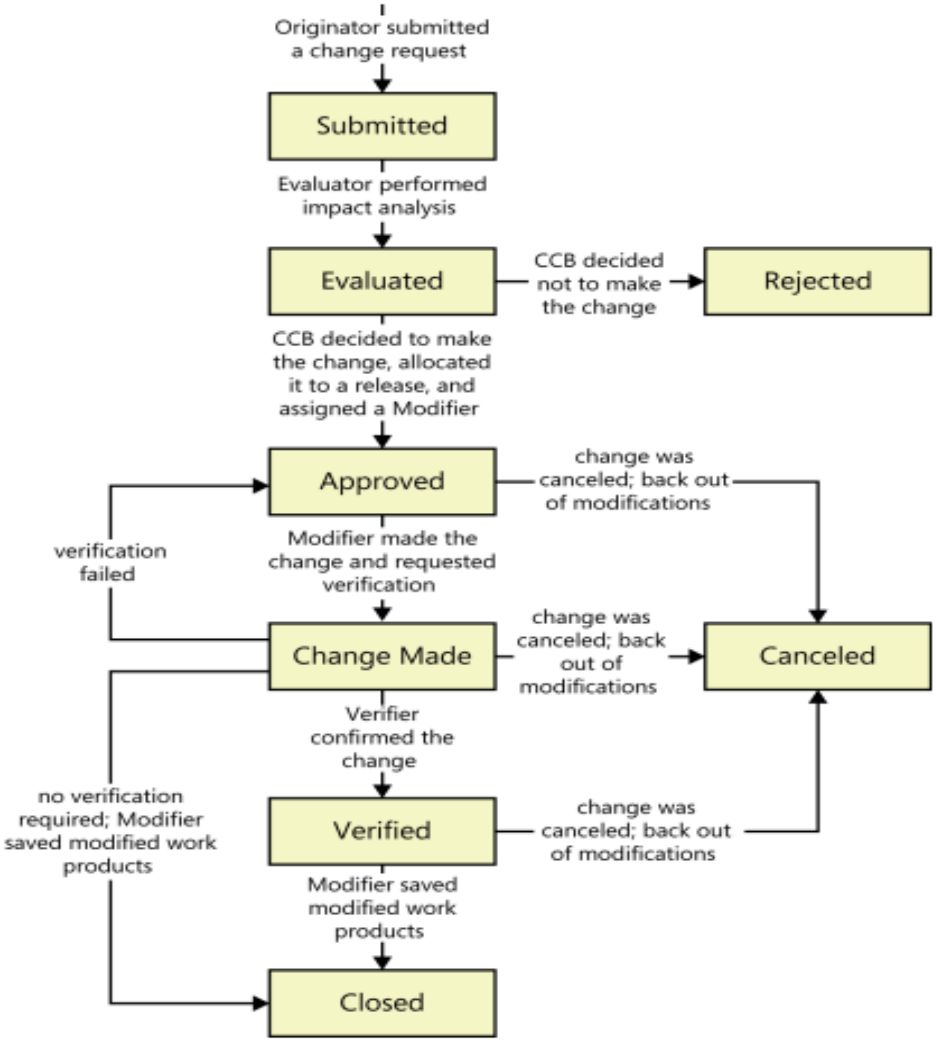


Fig 20- Share Docs

9. Requirements management

Requirements management in the Easy Food Tracker project involves the processes of planning, tracking, and controlling changes to the requirements throughout the project lifecycle. The following are some of the key aspects of requirements management in this project:

- **Requirements Planning:** Defining the approach and resources needed for requirements management, and establishing the processes and tools to be used.
- **Requirements Tracking:** Monitoring and controlling changes to the requirements throughout the project, to ensure that they are correctly implemented and that the application meets the needs of the stakeholders.
- **Requirements Change Control:** Managing the process of making changes to the requirements, including evaluating the impact of changes, communicating changes to stakeholders, and updating the requirements documentation accordingly.



- **Requirement Traceability Matrix**

A requirement traceability matrix (RTM) is a document or tool that is used to track and manage the relationship between project requirements and other project artifacts. The RTM provides a way to trace requirements throughout the project development lifecycle, from initial conception through to final delivery, ensuring that all requirements have been addressed and tested.

The purpose of an RTM is to establish a clear link between each requirement and the various stages of the project, such as design, development, testing, and delivery. The matrix typically consists of a table that maps each requirement to a corresponding design specification, test case, or other project deliverable. The matrix also provides a way to track the status of each requirement, such as whether it has been implemented, tested, or verified.

Some of the key benefits of an RTM include:

7 Improved visibility: By providing a clear and structured view of the requirements and how they relate to other project artifacts, an RTM can help improve visibility and reduce ambiguity around the project requirements.
8 Better requirement management: An RTM can help ensure that all requirements are properly documented, tracked, and tested throughout the project lifecycle, reducing the risk of missed requirements or incomplete testing.
9 Improved quality assurance: By providing a systematic approach to requirement testing and verification, an RTM can help improve the overall quality of the project deliverables.

Overall, an RTM is an important tool for ensuring that project requirements are properly documented, tracked, and tested, and can help improve the overall success of the project.

✔ **Description:**

FR----Functional Requirement
UC---Use Case
Functional Requirement:
FR1--- Enter the system
FR2---Invite Members
FR3---Create a Project folder
FR4---Delete a Project folder
FR5---Create List
FR6---To Do
FR7--- Add Task
FR8---Delete Task
FR9---Assign Member
FR10—Share Docs
FR11--- Commenting
FR12--- Get Notification
FR13—Reporting and Analytics

Use Case:

UC1: Enter the System
UC2: Invite Members
UC3: Create a Project folder
UC4: Delete the project Folder
UC5: Create List
UC6: To Do
UC7: Add Task
UC8: Delete Task
UC9: Assign Member
UC10: Set Priority
UC11: Set Due Date
UC12: Add Description
UC13: Create Board
UC14: Add Status
UC15: Add Comments
UC16: Share Documents
UC17: Download Documents
UC18: Get Notifications
UC19: Notify Update Progress
UC20: Messaging
UC21: Control Privacy
UC22: Control Updating
UC23: Manage Date/Time
UC24: Manage Notification
UC25: Generate Gantt Chart
UC26: Identify Update
UC27: Report Generating

UC/FR	FR1	FR2	FR3	FR4	FR5	FR6	FR7	FR8	FR9	FR10	FR11	FR12	FR13
UC1	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
UC2	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	
UC3		✓			✓	✓	✓	✓	✓	✓	✓	✓	
UC4			✓					✓				✓	
UC5					✓	✓	✓	✓	✓	✓	✓	✓	
UC6		✓				✓	✓	✓	✓	✓	✓	✓	
UC7		✓					✓	✓	✓	✓	✓	✓	
UC8				✓				✓			✓	✓	
UC9		✓					✓	✓		✓	✓	✓	
UC10		✓				✓					✓	✓	
UC11		✓				✓	✓	✓	✓	✓			
UC12			✓		✓	✓		✓	✓	✓		✓	
UC13						✓	✓				✓	✓	
UC14		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		
UC15		✓	✓		✓	✓	✓		✓	✓	✓	✓	
UC16		✓							✓	✓	✓	✓	
UC17		✓						✓	✓	✓		✓	
UC18		✓		✓	✓	✓	✓	✓	✓	✓		✓	
UC19											✓	✓	✓
UC20		✓					✓		✓	✓	✓	✓	
UC21		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
UC22											✓	✓	✓
UC23						✓	✓		✓	✓		✓	
UC24						✓	✓		✓	✓	✓	✓	
UC25												✓	✓
UC26						✓						✓	
UC27	✓					✓	✓	✓	✓	✓		✓	✓

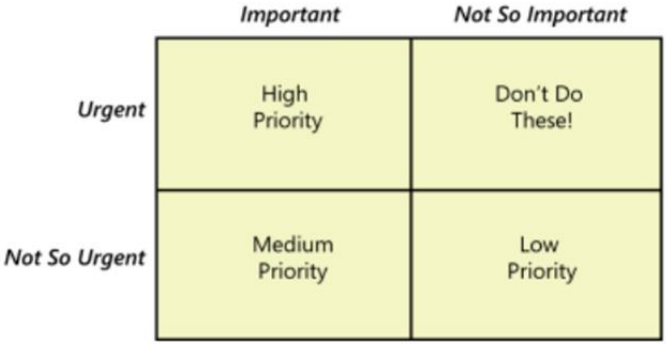
10 Appendix

10.1 Prioritization of Requirements

We've prioritized the functional requirements by following Three-level Scale technique.

10.1.1 Three-level Scale

A three-level scale of requirement prioritization could be:



1.High Priority: Requirements that are critical to the success of the project and must be implemented for the project to be considered successful. These requirements are typically related to the core functionality of the product or service being developed, and failure to implement them could result in significant negative consequences.

2.Medium Priority: Requirements that are important to the success of the project but are not critical. These requirements may have an impact on the user experience or provide additional functionality, but their absence would not necessarily result in project failure.

3.Low Priority: Requirements that are nice-to-have but not essential. These requirements may provide additional features or enhance the user experience, but their absence would not have a significant impact on the success of the project.

4.Don't Do This: These items are less important but still urgent. They should be addressed after completing the higher priority medium priority items. The items on the right side of the dividing line within this category have a higher priority.

This prioritization scale can help teams make informed decisions about where to focus their efforts and resources, and can also be used to guide trade-offs when there are competing demands on the project.

10.1.2 Prioritization of the requirements of IIT Kanban Board

According to the requirements of the different stakeholder we use the three-level scale and prioritized

our requirements as follow:

FR1--- Enter the system (Priority=HIGH)

FR2---Invite Members (Priority=MEDIUM)

FR3---Create a Project folder (Priority=HIGH)

FR4---Delete a Project folder (Priority=HIGH)

FR5---Create List (Priority=HIGH)

FR6---To Do (Priority=HIGH)
FR7--- Add Task (Priority=HIGH)

FR8---Delete Task (Priority=HIGH)
FR9---Assign Member (Priority=HIGH)
FR10—Share Docs (Priority=MEDIUM)

FR11--- Commenting (Priority=MEDIUM)
FR12--- Get Notification (Priority=HIGH)

FR13—Reporting and Analytics (Priority=HIGH)

NFR1—Performance (High)
NFR2— Scalability (High)
NFR3— Security (High)
NFR4— Usability (High)
NFR5— Accessibility (Medium)
NFR6— Compatibility (Medium)
NFR7— Maintainability (Medium)
NFR8— Testability (High)
NFR9---Backup and Recovery (Low)

UR1-- Simple and Intuitive navigation (High)

UR2-- Clear and consistent visual design (High)

UR3-- Accessible for all (High)

UR4--Search and filter options (High)

UR5-- Error Prevention and Recovery (High)

UR6-- Feedback and confirmation (High)

UR7-- Help and documentation (High)