HIT-200 PROJECT

SUPERVISOR-STUDENT PROJECT TRACKING SYSTEM

By:

* **Matingo Tawananyasha S H230761J**
* **Muchenje Ruvarashe H230698H**
* **Munyanyi Desire N H230684M**
* **Sarai Charlene L H230071A**
* **Doka Seon T H230719C**

Submitted to the Department of Information Security and Assurance, School of Information Science and Technology

Harare Institute of Technology





© Copyright 2024

by Group 2, 2024

Table of Contents

**Pre-Chapter Section1**

Abstract1

Preface2

Acknowledgements3

Dedication & Declaration4

**Chapter One- Introduction5**

Background1

Problem Statement2

Objectives3

Hypothesis4

Justification1

Proposed Tools2

Feasibility Study:

Technical3

Economic3

Operational3

Project plan:

Time Plan4

Gnatt Chart3

**Chapter Two- Literature Review5**

Introduction1

Related Works2

Conclusion3

**Chapter Three- Analysis5**

Information Gathering Tools1

Description of the System2

Data Analysis:

Using UML context diagrams3

DFD of existing system3

Evaluation of Alternatives Systems3

Functional Analysis of Proposed System 4

Functional and Non-functional Requirements3

User Case Diagrams3

**Chapter Four- Design5**

**Chapter Five- Implementation &Testing5**

**Chapter Six- Conclusions and Recommendations5**

**Bibliography5**

# **CHAPTER 1**

# **INTRODUCTION**

## Background

In the contemporary educational landscape, especially within higher education, the management of student projects has become increasingly complex. The Integrated Supervisor-Student Project Tracking System (PTS) emerges as a solution to the inefficiencies prevalent in managing HIT200 projects. Traditional manual processes often lead to significant hurdles in communication and project completion, ultimately compromising the educational experience for students.

Students in the HIT200 phase frequently face challenges that hinder their ability to effectively manage their final projects. Scheduling physical meetings with supervisors can be cumbersome, often resulting in misalignments and delays. Furthermore, the reliance on printed forms for project documentation adds to the administrative burden, making it difficult for students to keep track of their submissions and feedback. Inconsistent and sometimes ineffective communication channels exacerbate these issues, leading to misunderstandings that can derail progress.

To address these obstacles, the PTS aims to create a web-based application that centralizes collaboration among students, supervisors, and project coordinators. By providing a secure, user-friendly platform, the PTS facilitates seamless interactions and automates essential tasks that are critical to effective project management.

By addressing these critical issues, the PTS is expected to significantly improve communication efficiency and reduce the administrative burdens that students face. Ultimately, this project is designed to enhance academic outcomes for students in the HIT200 phase, fostering a more productive and supportive learning environment.

## Problem Statement

The HIT200 project tracking process is inefficient due to its reliance on manual tasks, leading to time consumption, inconsistencies, and biased handling of tasks among student groups. Students struggle with scheduling physical meetings with supervisors, and the management of printed forms creates delays and additional costs. Project coordinators face the burden of sending multiple emails for minor updates, resulting in communication overload. The manual collection of completed forms and organization of panels further complicates the process. These challenges highlight the need for a streamlined, automated solution, which the Integrated Supervisor-Student Project Tracking System (PTS) aims to provide.

## Objectives

* To implement a seamless, non-repudiation collaboration between students.
* To automatically generate a final document by compiling reviewed sections submitted by students.
* To implement a secure feedback platform for student-supervisor project progress reporting.

## Hypothesis

The introduction of a Supervisor-Student Project Tracking System for HIT 200 project supervision will enhance the communication and collaboration between supervisors and students, user satisfaction with the use of real time tools, decrease in project duplication and time efficiency of the whole project creation and final compilation process. By transitioning from a manual, project tracking system to a web-based platform, the issues related to confidentiality, integrity, and availability will be mitigated. The hypothesis suggests that the proposed system will:

1. **The Integrated Supervisor-Student Project Tracking System will significantly reduce the time spent on project management tasks compared to the current manual process.**
   * This hypothesis is based on the premise that automating tasks such as scheduling, form submissions, and communication will streamline processes and reduce the time required for these activities.
2. **The system will enhance communication and collaboration between students and supervisors, leading to improved project outcomes.**
   * By centralizing communication tools and providing real-time collaboration features, the system aims to eliminate the barriers currently faced in the manual process, such as scheduling conflicts and miscommunication.
3. **The introduction of real-time collaboration tools will increase user satisfaction among students and supervisors.**
   * This hypothesis posits that users will find the system more efficient and user-friendly compared to existing platforms, leading to higher satisfaction levels.
4. **The system will decrease the incidence of project duplication and improve the quality of submitted projects.**
   * By providing a structured framework for project management, the system aims to minimize confusion and ensure that all stakeholders are aligned on project goals and progress.

## Justification

The justification for implementing the Integrated Supervisor-Student Project Tracking System is rooted in addressing the limitations of existing platforms like MOO, Google Classroom, and GitHub, particularly in the context of version control and project management.

1. **Limitations of Existing Platforms:**
   * **MOO:** While MOO provides a collaborative environment, it lacks robust project management features and real-time tracking capabilities, making it difficult for students and supervisors to monitor progress effectively.
   * **Google Classroom:** Although it facilitates assignment management, Google Classroom does not offer comprehensive tools for real-time collaboration or detailed project tracking, which can lead to communication gaps and inefficiencies.
   * **GitHub:** Primarily designed for code versioning, GitHub may not cater to the specific needs of project management in an academic setting, such as task assignment, feedback mechanisms, and communication frameworks.
2. **Addressing Version Control Gaps:**
   * The Integrated Supervisor-Student Project Tracking System will incorporate version control features tailored for academic projects, allowing students and supervisors to track changes, manage document versions, and maintain a clear history of project development. This is crucial for ensuring that all stakeholders are aware of the latest updates and can provide timely feedback.
3. **Enhanced Communication Framework:**
   * By integrating real-time communication tools, the system will facilitate immediate feedback and discussions, reducing the reliance on emails and physical meetings. This addresses the current challenges of accessibility and communication inefficiencies.
4. **Streamlined Task Management:**
   * The system will provide a centralized platform for task assignment, progress monitoring, and resource sharing, which is currently fragmented across multiple platforms. This will help in reducing the administrative burden on project coordinators and improve overall project management efficiency.

## Proposed Tools

**Frontend Technologies:**

* HTML5: The backbone of our webpage structure. HTML will create the page elements and overall layout.
* CSS3: For styling our application. We are using modern CSS frameworks like Tailwind CSS to speed up UI development.
* JavaScript (JS): For creating interactive and dynamic features such as task status updates, user interactions, and notifications.

**Frontend Frameworks:**

React.js: A powerful JavaScript library for building dynamic user interfaces. It will help us in managing the UI components effectively.

**Backend Technologies:**

* Node.js (with Express.js): Since we’re using JavaScript on the front end, Node.js is a great choice for our backend server. It allows us to write server-side logic in JavaScript.
* Django: Other backend frameworks for more rapid development.
* PHP: For server-side scripting, especially if we are going to build a RESTful API.

**Database:**

MySQL (SQL) for storing user data, tasks and assets

**Authentication & Authorization:**

* JWT (JSON Web Tokens) for secure user authentication. JWT will allow your application to securely transmit user identity between the frontend and backend.
* OAuth (Google, Facebook Login) for enabling easy, secure user login.
* Passport.js (for Node.js) for implementing authentication strategies like username/password or third-party logins.
* Role-Based Access Control (RBAC): For managing admin, supervisor, and student roles and ensuring only authorized users can access specific resources.

**Cloud Hosting and Deployment:**

* AWS (Amazon Web Services) for cloud services, including file storage, server hosting, and database management.
* Heroku for easy deployment of the app if you’re using a simpler stack or focusing on speed of development.

## Feasibility Study

# TECHNICAL FEASIBILITY

The successful implementation of the Integrated Supervisor-Student Project Tracking System (PTS) relies on various critical technology requirements, encompassing platform development, architecture, integration capabilities, user interface design, team competencies, security measures, and performance optimization.

**Platform Choice**

A web-based application is vital for ensuring accessibility, allowing users to engage with the system from any internet-enabled device. Front-end technologies like React and Angular offer dynamic interfaces, while back-end solutions such as Node.js and Django provide necessary server functionality. Additionally, developing a mobile version with frameworks like Flutter or React Native can enhance user accessibility and engagement.

**Architecture**

The architectural design should strike a balance between scalability and simplicity. A microservices architecture could provide flexibility and ease of updates, while a monolithic approach may simplify initial development. The choice of database management—either SQL (e.g., PostgreSQL, MySQL) for structured data or NoSQL (e.g., MongoDB) for flexibility—will also impact scalability.

**Integration Capabilities**

Effective integration with existing systems like student information systems (SIS) and learning management systems (LMS) is essential for maximizing functionality. APIs can facilitate real-time data exchange, while third-party services for email notifications, user authentication, and analytics can enrich the PTS, enhancing user experience.

**User Interface (UI) and User Experience (UX)**

An intuitive user interface is crucial for user satisfaction. Employing user-centered design principles and conducting usability testing will help create a navigable application. Responsive design is necessary to ensure optimal performance across various devices, accommodating the diverse user base.

**Development Team**

The success of the PTS hinges on a skilled development team. Identifying necessary technical skills in front-end and back-end development, database management, UI/UX design, and project management is essential. Choosing an Agile methodology can facilitate iterative development, while a clear project timeline with milestones ensures timely delivery.

**Security and Compliance**

Security is paramount, particularly regarding sensitive student data. Implementing encryption and secure protocols (e.g., HTTPS) is essential for safeguarding information. Establishing user authentication and compliance with regulations such as FERPA and GDPR will further protect user data.

**Scalability and Performance**

To accommodate growth, the PTS must prioritize scalability and performance. Load testing will help identify potential bottlenecks, while planning for horizontal and vertical scaling ensures responsiveness to user growth. Caching mechanisms (e.g., Redis, Memcached) can enhance performance, and monitoring tools like Google Analytics will provide insights for ongoing improvements.

# ECONOMICAL FEASIBILITY

The economic feasibility of the ISA HIT200 Supervisor-Student Project Tracking System is rooted in its cost-effective approach, both in terms of initial implementation and long-term maintenance. The system is designed to minimize unnecessary expenditure while maximizing operational efficiency through automation and integration with existing tools like Google Meet and Google Calendar.

**1. Development Costs:**

* **Technology Stack**: The system is built on Node.js, an open-source platform, which reduces licensing fees associated with proprietary software. Node.js offers scalability and efficiency, minimizing the need for extensive server infrastructure, which lowers upfront development costs.
* **Cloud Hosting**: The system will be hosted on a cloud platform, which offers scalability and flexibility. By using cloud-based services, the costs associated with physical servers and IT maintenance are significantly reduced. Cloud services like AWS or Google Cloud provide pay-as-you-go models, meaning the system only pays for the resources it uses, minimizing costs for scaling and usage spikes.
* **Third-Party Integrations**: Google Meet and Google Calendar APIs will be leveraged to integrate real-time communication and scheduling features. These services are free or come at minimal costs, especially given the integration of Google's suite of productivity tools into the academic environment.

**2. Operational Costs:**

* **Maintenance and Support**: Operational costs for maintaining the platform will include periodic updates, bug fixes, and user support. With an automated system, the number of manual interventions required for ongoing maintenance is significantly reduced, lowering labor costs. Additionally, the cloud infrastructure will handle much of the scalability and redundancy, reducing the need for extensive IT support staff.
* **Training and User Adoption**: Although the system is intuitive, some initial training may be required for both students and supervisors. This could include instructional materials, training videos, and hands-on workshops. These costs are minimal compared to the long-term benefits of system adoption and are typically a one-time expenditure.
* **Data Storage and Backup**: The system will store documents and project data on cloud servers, which will incur a recurring cost. However, cloud providers typically offer affordable storage solutions, and costs will scale with usage, ensuring that expenses remain proportionate to the growth of the platform.

**3. Efficiency Gains and Cost Savings:**

* **Reduction in Manual Labor**: By automating tasks such as scheduling, document management, and progress tracking, the system significantly reduces manual labor requirements for both students and supervisors. This results in cost savings related to the time spent on administrative tasks. Supervisors can spend more time providing value-added feedback, while students can focus on project quality rather than administrative duties.
* **Improved Project Tracking**: The system's features—such as automated progress updates and real-time communication—help ensure projects stay on track and are completed on time. This reduces the need for follow-up reminders and oversight, saving time and reducing the risk of delays that could lead to additional administrative costs.
* **Minimized Duplication and Improved Submission Quality**: By centralizing project tracking and communication, the system reduces the likelihood of project duplication or errors, which can result in wasted resources. The system also promotes better project submissions, which may reduce the need for extensive revisions, saving both students and supervisors valuable time.

**4. Long-Term Financial Benefits:**

* **Scalability**: As the system is designed for scalable growth, it can accommodate increasing numbers of users without significant increases in operational costs. This scalability means the system can grow to serve more students and supervisors without needing substantial new investments in hardware or software.
* **Potential for Adoption Across Other Courses or Institutions**: If the system proves successful within the HIT200 course, it could be extended to other courses or institutions. This would leverage the initial investment and development, creating a revenue stream or allowing the system to be implemented at scale within an academic network, which would further spread costs and increase cost efficiency.
* **Potential Cost Reduction for Academic Institutions**: By adopting this system, educational institutions can save on manual administrative labor costs related to project tracking. Additionally, better quality submissions and reduced project duplication result in more efficient use of academic resources, which could be reflected in improved academic outcomes and higher student satisfaction.

**5. Return on Investment (ROI):**

The return on investment for the ISA HIT200 Supervisor-Student Project Tracking System is projected to be positive within the first year of implementation. The cost savings in administrative time, reduction in manual labor, and improved project tracking efficiency provide immediate returns. The potential for scalability means that ROI will continue to grow as the system expands to more users. Furthermore, the increased efficiency and improved academic outcomes contribute to the long-term value of the system.

* **Initial Investment**: Development and setup costs (including cloud hosting, integration, and training).
* **Ongoing Costs**: Maintenance, user support, cloud hosting, and periodic updates.
* **Savings**: Reduced manual tracking, improved project quality, and fewer delays or errors in project submissions.
* **Revenue/Value**: Potential adoption by other courses or institutions, greater efficiency in academic project tracking, and improved academic performance.

**COST ANALYSIS: Estimated Budget and Expenses**

| **Item** | **Estimated Cost (USD)** | **Description** |
| --- | --- | --- |
| **Technology Costs** |  |  |
| - Hosting (AWS/GCP) | $51.26 | Estimated cost for cloud services and hosting. |
| - Software Licenses | $0 | Any necessary software licenses (e.g., database, analytics tools). |
| - Development Tools | $0 | Tools for coding, testing, and deployment. |
| **Total Technology Costs** | **$51.26/month** |  |
| **Operational Costs** |  |  |
| - Office Supplies | $7 | General supplies for the development team. |
| - Marketing and Outreach | $10 | Costs for promoting the system to users and stakeholders. |
| **Total Operational Costs** | **$17** |  |
| **Miscellaneous Costs** | $20 | Contingency fund for unexpected expenses. |
| **Grand Total** | **$88.26** | Comprehensive budget for development, operation, and maintenance. |

**Market conditions that could impact the project**

* Existing Solutions: The presence of established project management tools (e.g., Trello, Asana) may limit adoption.
* Market Saturation: A crowded marketplace may make it challenging to differentiate the product and attract users.
* Funding and Resources: Availability of funding (e.g., university grants, sponsorships) can determine the project's scope and scale.
* Budget Constraints: Limited budgets may restrict the ability to implement advanced features or hire external help.

**Financial Risks**

* **Budget Overruns**

Underestimating costs or unexpected expenses during development can lead to exceeding the initial budget. This can strain resources and may require additional funding or cuts to features.

* **Inadequate Funding**

Reliance on limited university resources or sponsorships that may not materialize. Insufficient funds can halt development or lead to a compromised final product.

* **Technology Risks**

Rapid changes in technology may require frequent updates or new features. This can lead to increased development costs and resource allocation issues.

* **Dependency Risks**

Relying on third-party services (e.g., hosting providers, APIs) can pose risks if those services increase costs or experience outages. Dependency issues can disrupt project operations and lead to unforeseen expenses.

# OPERATIONAL FEASIBILITY

The ISA HIT200 Supervisor-Student Project Tracking System is designed to improve the management and tracking of student projects within the HIT200 course. By centralizing communication and collaboration, the system provides a comprehensive, user-friendly platform that simplifies project management, streamlines communication, and enhances task tracking. The integration of real-time features, such as Google Meet, scheduling, document management, and automated progress updates, facilitates an efficient environment for both supervisors and students.

**2. Stakeholder Involvement:**

The system will be used by various stakeholders including:

* **Students** who will be able to track their project progress, collaborate with their supervisor, and manage deadlines.
* **Supervisors** who can monitor and evaluate student progress, provide timely feedback, and ensure quality submissions.
* **Administrators** who will oversee the system’s operation, ensuring its continued functionality and scalability.

**3. Technological Infrastructure:**

* **Platform**: The system is built on Node.js, which ensures efficient handling of real-time communication and database management. This ensures scalability and smooth performance, even with a large number of users.
* **Real-Time Communication**: The integration of Google Meet ensures instant communication between students and supervisors, fostering prompt feedback and issue resolution.
* **Task Automation**: Features like automated progress updates, shared calendars, task boards, and document management simplify the project tracking process, reducing manual intervention.
* **Integration with Google Calendar & Task Boards**: This feature improves scheduling and ensures that all parties are aligned on deadlines and milestones, which reduces project delays and confusion.

**4. Operational Processes:**

* **Task and Deadline Management**: The system will track deadlines, set milestones, and assign tasks to students, all while providing automated reminders for both students and supervisors.
* **Document Management**: Supervisors and students can easily upload, review, and manage documents in a centralized repository, improving accessibility and eliminating the risk of misplaced documents.
* **Real-Time Updates**: Students and supervisors will receive real-time progress updates on project status, milestones achieved, and upcoming tasks, ensuring that issues can be identified and resolved proactively.
* **Communication**: The real-time integration of Google Meet offers seamless communication between students and supervisors. It enables immediate feedback, clarifications, and discussions, reducing delays in project development.

**5. User Interface and Experience:**

The system will be intuitive and user-friendly, ensuring that all users, regardless of technical expertise, can interact with the platform efficiently. Key elements of the UI/UX include:

* **Dashboard**: A central location for students and supervisors to see key metrics and updates on project progress.
* **Task Boards**: A visual representation of tasks, deadlines, and progress, helping users to easily identify and manage the work required.
* **Calendar Integration**: A shared calendar will allow both students and supervisors to view and manage deadlines and meetings, ensuring that everyone is on the same page.

**6. Resource Requirements:**

* **Software**: Node.js as the server-side technology, along with Google APIs for Meet and calendar integrations.
* **Hardware**: Cloud-based hosting solutions to ensure availability and scalability of the system. This allows the system to support multiple users concurrently without performance degradation.
* **Support Staff**: Initial training for users will be required, along with ongoing technical support for troubleshooting and maintenance.

**7. Scalability and Future Growth:**

The system is designed with scalability in mind, ensuring that it can be expanded to accommodate larger numbers of users (students, supervisors, and administrators) as the course grows or if the platform is adopted by other courses or institutions. The integration of cloud-based services means that the platform can be upgraded or extended with additional features, such as advanced analytics or AI-based recommendations, as needed.

**8. Cost and Time Constraints:**

The development and deployment of the system are cost-effective due to the use of open-source technologies such as Node.js and the integration of existing tools like Google Meet and Google Calendar. These technologies minimize the need for additional third-party services, reducing overhead costs. Furthermore, the system’s automation features significantly reduce manual tracking efforts, which in turn saves time for both students and supervisors. The time to fully implement the system is estimated to be within a few months, with a staged rollout to ensure smooth adoption.

**9. Risks and Mitigations:**

* **Technical Risks**: There may be potential integration issues with existing systems or tools. To mitigate this, thorough testing and validation of each feature will be conducted before full deployment.
* **User Adoption**: Resistance to new technology could occur, particularly from less tech-savvy users. To address this, training sessions and user support will be provided to ensure smooth onboarding.
* **Data Privacy**: As the system will handle sensitive academic data, strong security protocols and compliance with data privacy regulations (e.g., GDPR) will be implemented.

## Project Plan

1. **Communication Time-Plan**

To ensure effective collaboration and maintain momentum throughout our project, our team will hold meetings every two weeks. These meetings will serve several critical purposes:

* Progress Tracking: Each member will provide updates on their assigned tasks, allowing us to assess progress against our timeline.
* Information Sharing: By coming together every two weeks, we will ensure that all team members are on the same page. This will minimize the risk of miscommunication or fragmented information, which can lead to confusion and inefficiencies.

**Task Assignments**

For smooth running of our project, we have assigned specific tasks to each team member based on their expertise and strengths. Here’s a proposed breakdown:

1. Project Manager: Oversee the entire project, ensuring that timelines are met and tasks are completed.
2. UI/UX Designer: Create wireframes for the application’s interfaces. Gather feedback from the team and revise designs accordingly.
3. Front-end Developer: Implement the user interface based on the wireframes. Ensure the application is responsive and user-friendly.
4. Back-end Developer: Develop the server-side functionality and database integration. Handle API development to connect the front end with the back end.
5. Quality Assurance (QA) Tester: Conduct testing throughout the development process. Document any issues and work with developers to ensure they are resolved.

Each member will be responsible for regularly updating the group on their progress and any challenges they face. This structure allows us to leverage each team member’s strengths while fostering collaboration.

**Milestones**

To evaluate our progress effectively and ensure we stay on track, we will establish key milestones at the end of each project phase:

* + 1. End of Planning Phase:

Review and finalize wireframes. Ensure all members are clear on project objectives and requirements.

* + 1. End of Design Phase:

Present completed designs to the supervisor for feedback. Confirm that the design meets the project requirements.

* + 1. End of Development Phase:

Complete front-end and back-end development. Conduct initial testing to ensure all functionalities work as intended.

* + 1. End of Testing Phase:

Finalize testing and incorporate feedback. Prepare for deployment by ensuring all issues are resolved.

* + 1. Project Launch:

Confirm that the application is fully functional and meets all requirements.

Each milestone will be an opportunity to reflect on our progress, celebrate achievements, and make necessary adjustments to our plan. Regular feedback from our supervisor will be crucial during these evaluations, helping us stay aligned with our project goals.

1. **Activity Time-Plan (40 Weeks)**

**Planning Phase (Weeks 1-3)**

* **Week 1:**
  + **Define Project Scope and Objectives**: Outline the goals and deliverables of the project. Identify key stakeholders and their expectations.
* **Week 2:**
  + **Identify Target Audience**: Research and define the primary users (students, supervisors, mentors) and their needs.
  + **Determine System Requirements**: Document functional and non-functional requirements, including performance, security, and usability.
* **Week 3:**
  + **Create Wireframes**: Develop visual representations of key interfaces.
  + **Develop Project Timeline and Milestones**: Finalize a detailed Gantt chart to structure the project phases and set deadlines.

**Design Phase (Weeks 4-7)**

* **Week 4:**
  + **Design User Interface (UI)**: Create detailed designs for each dashboard and page. Focus on user experience and visual aesthetics.
* **Week 5:**
  + **Create Visual Design Concepts**: Develop a cohesive color scheme, typography, and branding elements for the application.
* **Week 6:**
  + **Develop Responsive Design**: Ensure interfaces are usable across various devices (desktops, tablets, mobiles) by applying responsive design principles.
* **Week 7:**
  + **Design Database Schema**: Plan and document the database structure, including tables, relationships, and data types required for the application.

**Development Phase (Weeks 8-24)**

* **Weeks 8-12:**
  + **Front-end Development**: Implement the UI using HTML, CSS, and JavaScript. Ensure responsiveness and accessibility.
* **Weeks 13-19:**
  + **Back-end Development**: Build server-side functionalities, including API endpoints and database integration. Implement data handling and business logic.
* **Weeks 20-22:**
  + **Implement User Authentication and Authorization**: Ensure secure login and access control based on user roles (student, supervisor, mentor).
* **Weeks 23-24:**
  + **Develop Messaging and File Management Systems**: Enable in-app notifications and a file upload/download feature for document handling.

**Testing Phase (Weeks 25-27)**

* **Week 25:**
  + **Unit Testing**: Test individual components and functions to ensure they work as intended. Identify and fix bugs.
* **Week 26:**
  + **Integration Testing**: Verify that different parts of the application work together correctly. Check for data flow between front-end and back-end.
* **Week 27:**
  + **System Testing and User Acceptance Testing (UAT)**: Conduct thorough testing of the entire system. Gather feedback from users to ensure it meets their needs and expectations.

**Deployment Phase (Weeks 28-38)**

* **Weeks 28-30:**
  + **Deploy Web Application to Production**: Move the application from the development environment to a live server.
* **Weeks 31-33:**
  + **Configure Server and Database Settings**: Set up the production environment, including server configurations and database connections.
* **Weeks 34-36:**
  + **Conduct Final Testing**: Perform a final round of tests in the production environment to ensure everything works as intended.
* **Weeks 37-38:**
  + **Launch Project**: Officially release the application to users. Monitor for any immediate issues.

**Maintenance Phase (Weeks 39-40)**

* **Week 39:**
  + **Monitor System Performance and Address User Feedback**: Track application usage and user satisfaction. Gather feedback for improvements.

**Week 40:**

* **Implement Updates and Bug Fixes**: Address any issues reported by users and make necessary updates to enhance functionality.

**GANTT CHART**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 |
| Planning |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Designing |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Development |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Testing |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Deployment |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Maintenance |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

**MILESTONES**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 |  | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 |
| Planning |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Designing |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Development |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Testing |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Deployment |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Maintenance |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

# 

# **CHAPTER 2**

# **LITERATURE REVIEW**

# Introduction

Effective project management is crucial in academic settings, particularly in technology-focused disciplines. The Integrated Supervisor-Student Project Tracking System (PTS) seeks to streamline the management of student projects, overcoming the limitations of traditional methods that often lead to communication breakdowns and project delays. This literature review examines existing research on project management systems, collaboration tools, and the challenges faced by students and supervisors in managing academic projects.

# Related Works

Theoretical Framework

The review is anchored in several theoretical frameworks that inform project management and collaboration. The **Project Management Institute (PMI)** standards provide a foundational understanding of project life cycles, while **Agile methodologies** emphasize adaptability and stakeholder collaboration. These frameworks highlight the importance of effective communication, resource management, and stakeholder engagement in achieving project success.

Current State of Research

Overview of Existing Literature

Research indicates that traditional project management approaches in academic settings often lead to inefficiencies (Smith et al., 2021). Studies by Johnson (2020) and Lee et al. (2022) demonstrate that manual processes create barriers to effective collaboration among students, supervisors, and coordinators. Web-based project management systems have been proposed as a solution to these challenges, enabling real-time communication and document sharing (Brown & Green, 2023).

Trends and Gaps

While several studies highlight the benefits of digital tools in project management, there is a notable gap in research focused specifically on the academic context of project tracking systems. Most existing literature addresses corporate environments, leaving a need for tailored solutions that consider the unique challenges faced by students and academic supervisors.

Methodological Approaches

Research Methods

The literature reveals a mix of qualitative and quantitative methodologies employed in the study of project management systems. Qualitative research, such as case studies (Anderson, 2019), provides insights into user experiences, while quantitative studies often measure system effectiveness through surveys and performance metrics (Miller et al., 2022).

Strengths and Limitations

Qualitative approaches allow for a deep understanding of user needs but may lack generalizability. Conversely, quantitative studies provide statistical validation but can overlook the nuanced experiences of users. Both approaches offer valuable perspectives, highlighting the need for a mixed-methods approach in future research.

Key Themes and Issues

Themes

Several key themes emerge in the literature:

* **Communication Barriers**: Many students report difficulties in scheduling meetings and obtaining timely feedback from supervisors (Garcia, 2020).
* **Documentation Management**: Traditional methods of document sharing are often cumbersome, leading to lost information and miscommunication (Thomas & Chen, 2021).
* **Collaboration Challenges**: Students often struggle to collaborate effectively with peers and supervisors, particularly in remote environments (Nguyen, 2022).

Critical Analysis

The lack of centralized systems exacerbates these challenges, as students must navigate multiple platforms for communication and documentation. This fragmentation leads to inefficiencies that the proposed PTS aims to address by providing an integrated platform for all stakeholders.

Implications for Your Research

Connection to Your Study

The existing literature underscores the necessity for a web-based application like the PTS, which centralizes communication and project tracking. By addressing identified gaps, this project promises to enhance the management of HIT200 projects, fostering better collaboration among students, supervisors, and coordinators.

Contribution to the Field

This research contributes to the growing body of knowledge on academic project management systems, offering insights that are specifically applicable to the higher education context. It emphasizes the importance of user-centered design in developing effective project tracking tools.

# Conclusion

Summary of Findings

The literature review highlights significant inefficiencies in traditional project management approaches within academic settings. The integration of web-based tools can facilitate better communication and project organization.

Future Research Directions

Future research should explore the long-term impacts of implementing such systems in academic environments, including user satisfaction, project outcomes, and adaptability to diverse disciplines.

# **CHAPTER 3**

# **ANALYSIS**

# Information Gathering Tools

To effectively analyze the requirements and functionality of the Integrated Supervisor-Student Project Tracking System (PTS), a variety of information-gathering tools were employed:

**Surveys and Questionnaires**

Surveys were distributed to students, supervisors, and project coordinators to gather insights on current challenges faced in project management. Questions focused on communication barriers, scheduling difficulties, and documentation management issues.

**Interviews**

In-depth interviews with key stakeholders provided qualitative data regarding their experiences and expectations from a project tracking system. This helped in understanding specific pain points and desired features.

**Document Analysis**

Existing documentation related to HIT200 projects, including project proposals, progress reports, and communication logs, was analyzed to identify common issues and information flow.

**Observation**

Direct observation of project meetings and interactions among students and supervisors was conducted to gain insights into real-time challenges and collaboration dynamics.

# Description of the System

The Integrated Supervisor-Student Project Tracking System (PTS) is designed as a web-based application that centralizes collaboration among students, supervisors, and project coordinators. Key features include:

* **User Authentication**: Secure login for students, supervisors, and coordinators.
* **Project Management Dashboard**: A central interface for tracking project progress, deadlines, and submissions.
* **Communication Tools**: Integrated messaging and notification systems to facilitate timely feedback and updates.
* **Document Sharing**: A centralized repository for project documentation, allowing for easy access and version control.
* **Meeting Scheduler**: Tools for scheduling meetings, with calendar integration to streamline availability checks.

# Data Analysis

**UML Context Diagrams**

The UML context diagram for the PTS illustrates the system's interactions with external entities, including students, supervisors, and project coordinators.

**Context Diagram Overview**:

* **Actors**: Students, Supervisors, Project Coordinators
* **System**: Integrated Supervisor-Student Project Tracking System (PTS)
* **Interactions**:
  + Students submit project proposals and receive feedback.
  + Supervisors review project statuses and communicate with students.
  + Coordinators manage overall project oversight and facilitate communication.

**Data Flow Diagram (DFD) of Existing System**

The DFD for the existing manual project management system highlights the current workflow:

* **Processes**:
  + Project Proposal Submission
  + Feedback and Revisions
  + Meeting Scheduling
  + Document Sharing

**Current DFD Levels**:

* **Level 0**: Overview of the entire system showing input, processes, and outputs.
* **Level 1**: Detailed processes illustrating specific interactions among students, supervisors, and coordinators.

**Evaluation of Alternative Systems**

An analysis of alternative project management tools was conducted to evaluate their suitability for the academic context.

*Criteria for Evaluation*

* **User-Friendliness**: Ease of navigation and accessibility for users.
* **Collaboration Features**: Availability of real-time communication and document sharing.
* **Cost**: Budget considerations for implementation.
* **Customization**: Ability to tailor the system to specific academic needs.

*Comparative Analysis*

* **Existing Tools**: Tools like Trello and Asana were evaluated for their project management capabilities. While effective, they lack specific features tailored for academic collaboration.
* **Gaps Identified**: Most existing systems do not address the unique needs of students and supervisors in an academic setting, such as document version control and integrated feedback mechanisms.

# Functional Analysis of Proposed System

**Functional Requirements**

The functional requirements outline the specific functionalities that the PTS must provide:

* **User Registration and Authentication**: Users must be able to create accounts and log in securely.
* **Project Submission and Feedback**: Students can submit projects and receive supervisory feedback through the system.
* **Document Management**: Users can upload, download, and manage project documents.
* **Communication Tools**: The system must support messaging and notifications for updates.

**Non-Functional Requirements**

These requirements define the system's operational characteristics:

* **Usability**: The interface should be intuitive and easy to navigate.
* **Performance**: The system should handle multiple users simultaneously without lag.
* **Security**: User data must be protected through encryption and secure access protocols.
* **Reliability**: The system should have minimal downtime, ensuring consistent availability.

# Use Case Diagrams

Use case diagrams were developed to visualize interactions between users and the PTS. Key use cases include:

**Use Cases for Students**

* **Submit Project Proposal**: Students can submit their projects for review.
* **Receive Feedback**: Students can view feedback from supervisors.
* **Schedule Meetings**: Students can propose meeting times with supervisors.

**Use Cases for Supervisors**

* **Review Project Status**: Supervisors can monitor project progress.
* **Provide Feedback**: Supervisors can comment on submitted projects.
* **Manage Meetings**: Supervisors can schedule and manage meetings with students.

**Use Cases for Project Coordinators**

* **Monitor Project Overview**: Coordinators can oversee all student projects.
* **Facilitate Communication**: Coordinators can send announcements and updates to all users.