



**Addis Ababa Science and Technology University**

**College of Engineering**

**Department of Software Engineering**

**Undergraduate Final Project Report**

**Title: Comprehensive Research and Project Management Platform**

No	Group Members Name	ID
1	Kidist Tadele	ETS0403/12
2	Melat Gizachew	ETS1020/12
3	Meron Kedir	ETS1022/12
4	Ruhama G/Mariam	ETS0573/12
5	Yordanos Yadessa	ETS1066/12

Advisor: Eyob Samuel

Signature: \_\_\_\_\_

**June 2024 G.C**

# Acknowledgement

We would like to express our deepest gratitude to our advisor, Eyob Samuel, for his invaluable guidance, unwavering support, and insightful feedback throughout the course of this project. His expertise and encouragement have been instrumental in shaping the direction and success of our work.

We extend our heartfelt thanks to the Addis Continental Institute of Public Health, an independent center for public health research and training. Their assistance in providing initial ideas and guidance was crucial in helping us focus on the most impactful areas of our research. Their contributions have been pivotal in laying the foundation for our project.

Our sincere appreciation goes to the Addis Ababa Science and Technology University (AASTU) for granting us the opportunity to demonstrate our capabilities and for the continuous support provided by the school staff. The encouragement and resources offered by AASTU have been vital in navigating the various challenges encountered during our journey.

Finally, we would like to extend our gratitude to everyone who generously contributed their ideas, insights, and time toward the development of this project. Your contributions have been invaluable, and we are deeply thankful for your support and collaboration.

# Table of Content

<b>Acknowledgement.....</b>	<b>i</b>
<b>Table of Content.....</b>	<b>ii</b>
<b>List of Figures.....</b>	<b>v</b>
<b>List of Tables.....</b>	<b>v</b>
<b>List of Abbreviations.....</b>	<b>vi</b>
<b>Abstract.....</b>	<b>viii</b>
<b>ᐱᐱᐱ.....</b>	<b>ix</b>
<b>Chapter 1: Introduction.....</b>	<b>1</b>
1.1. Background.....	1
1.2. Statement of the problem.....	2
1.3. Objectives.....	3
1.3.1 General Objectives.....	3
1.3.2 Specific Objectives.....	3
1.4. Scope.....	4
1.5. Methodology.....	5
1.6. Plan of Activities.....	6
1.7. Budget Required.....	9
1.8. Significance of the Study.....	10
<b>Chapter 2: Literature Review.....</b>	<b>12</b>
2.1 Overview.....	12
2.2 Project Management.....	12
2.2.1 Project.....	12
2.2.2 Key Elements of Project Management.....	12
2.2.3 The Project Management Process.....	13
2.3 Research Management.....	15
2.3.1 Research.....	15
2.3.2 Components of Research Management.....	15
2.3.3 Steps in Research Management.....	16
2.4 Study related works.....	18
2.5 Identifying milestones of the related literatures and finding the gaps.....	19
2.6 Lessons learned from literatures.....	20
<b>Chapter 3: Problem Analysis and Modeling.....</b>	<b>21</b>
3.1 Existing system and its problems.....	21
3.1.1 Existing Systems.....	21

3.1.2 Major problem of existing systems.....	21
3.2 Specifying the Requirements of the proposed solution.....	22
3.2.1 Functional Requirement.....	22
3.2.2 Non-Functional Requirement.....	23
3.3 System modeling.....	25
3.3.1 Overview.....	25
3.3.2 Scenario Based Modeling.....	26
3.3.2.1 Use Case Description.....	27
3.3.2.2 Activity Diagram.....	31
3.3.3 Behavioral/Dynamic Modeling.....	37
3.3.3.1 Sequence diagram.....	37
3.3.3.2 State Diagram.....	42
3.3.4 Class-Based Modeling.....	48
3.3.4.1 Identifying classes.....	48
3.3.4.2 Class Diagram.....	50
<b>Chapter 4: System Design.....</b>	<b>52</b>
4.1. Overview.....	52
4.2. Specifying the design goals.....	52
4.2.2 Frontend design goals.....	53
4.2.3 Backend Design Goals.....	54
4.2.4 Database Design Goals.....	55
4.3. Designing the Solution.....	56
4.3.1 UI Design.....	56
4.3.2 Database Design.....	60
4.3.3. Architecture of the System.....	62
4.3.4. System Design.....	62
4.4. Verifying the Requirements in the Design.....	65
<b>Chapter 5: System Implementation.....</b>	<b>69</b>
5.1. Reviewing the Design Solution.....	69
5.2. Development Tools.....	73
5.3. Developing the Solution.....	75
<b>Chapter 6: System Evaluation.....</b>	<b>86</b>
6.1. Sample Test Plans/Cases.....	86
6.1.1. Test Cases for System Admin.....	86
6.1.2. Test Cases for Project Manager.....	88
6.1.3 Test Cases for Supervisor.....	90
6.1.4 Test Cases for Researcher.....	91
6.1.5 Test Cases for Nonfunctional Requirements.....	93

6.2. Evaluation of the Proposed Design and Solution.....	96
6.2.1 Functional Requirements Evaluation.....	96
6.2.2 Non-functional Requirements Evaluation.....	98
6.3. Discussion of Results.....	99
<b>Chapter 7: Conclusion and Recommendations.....</b>	<b>100</b>
7.1. Conclusion.....	100
7.2. Recommendation.....	100
<b>References.....</b>	<b>102</b>

## List of Figures

Figure 1.1: Plan of activities Gantt chart.....	8
Figure 3.1: Use Case Diagram.....	30
Figure 3.2: Researcher's Activity Diagram.....	32
Figure 3.3: Project owner's Activity Diagram.....	33
Figure 3.4: Admin's activity diagram.....	34
Figure 3.5: project manager activity diagram.....	35
Figure 3.6: supervisor's activity diagram.....	36
Figure 3.7: Sequence diagram of Project Initiation and Management.....	38
Figure 3.8: Sequence diagram of Data Collection, Analysis and Reporting.....	40
Figure 3.9: Sequence diagram of Project Search and Retrieval.....	41
Figure 3.10: State diagram of Project Lifecycle.....	43
Figure 3.11: State diagram of User Login and Authorization.....	44
Figure 3.12: State diagram of Task Management.....	45
Figure 3.13: State diagram of Research Data Management.....	46
Figure 3.14: State diagram of System Maintenance and Updates.....	47
Figure 3.15: Class Diagram.....	51
Figure 4.1: Landing Page.....	56
Figure 4.2: My Project Page.....	56
Figure 4.3: Project Dashboard.....	57
Figure 4.4: Add Task Page.....	57
Figure 4.5: My Tasks Page.....	58
Figure 4.6: Document page.....	58
Figure 4.7: File Upload Page.....	59
Figure 4.8: Form Creation Page.....	59
Figure 4.9: ER Diagram.....	61
Figure 4.10: Component Diagram.....	64

## List of Tables

Table 1: Estimated Expenses.....	9
----------------------------------	---

# List of Abbreviations

Abbreviation	Definition
CRPMP	Comprehensive Research and Project Management Platform
SAR	System Admin Requirement
PMR	Project Manager Requirement
SR	Supervisor Requirement
RR	Research Requirement
FR	Functional Requirement
NFR	Non-Functional Requirement
GDPR	General Data Protection Regulation
HIPAA	Health Insurance Portability and Accountability
WCGA	Web Content Accessibility Guidelines
UI / UX	User Interface/User Experience
API	Application Programming Interface
KPI	Key Performance Indicator
HTTP	Hyper Text Transfer Protocol
UCD	User Centered Design
OOD	Object Oriented Design
IDE	Integrated Development Environment
SOLID	Single responsibility principle, Open-closed principle, Liskov substitution principle, Interface segregation principle, and Dependency inversion principle
RDM	Research Data Management
DMP	Data Management Plans

ELN	Electronic Lab Notebooks
ERD	Entity Relationship Diagram
GDP	Gross Domestic Product
QA	Quality Assurance
SSL	Secure Socket Layer
1NF	First Normal Form
2NF	Second Normal Form
3NF	Third Normal Form
PM	Project Management
ASP.NET	Active Server Pages Network Enabled Technologies
REST	Representational State Transfer



## Abstract

*In the contemporary research landscape, the efficient management of research projects is crucial for enhancing productivity and achieving impactful outcomes. This project aims to develop a Comprehensive Research and Project Management Platform designed to address the myriad challenges faced by researchers in academia, industry, and individual pursuits. By providing a centralized hub, the platform will streamline the research lifecycle, from planning and proposal development to execution, data analysis, and dissemination.*

*The platform seeks to mitigate issues such as data documentation inconsistencies, manual process inefficiencies, and the integration gap between research outcomes and organizational decision-making. Key features include project creation and management, tracking and scheduling, customizable data gathering forms, collaboration tools, and a robust dashboard for data analysis.*

*Through thorough requirement gathering, detailed design, iterative development, and rigorous testing, this platform will cater to the unique needs of diverse user groups, including students, universities, research organizations, and individual researchers. By fostering a cohesive and efficient research environment, the platform aims to enhance the overall productivity and impact of research collaborations, paving the way for innovative solutions and advancements in various fields.*

*The project underscores the importance of a unified research management system in today's data-driven world and anticipates future challenges by prioritizing adaptability and continuous improvement. Despite excluding AI integration, mobile app development, and other advanced features in its initial scope, the platform lays a solid foundation for future enhancements, ensuring its relevance and effectiveness in the evolving research landscape.*

## ረቂቅ

በዘመናዊው የምርምር መልክዓ ምድር፣ የምርምር ፕሮጀክቶችን በብቃት ማስተዳደር ምርታማነትን ለማሳደግ እና ውጤታማ ውጤቶችን ለማምጣት ወሳኝ ነው። ይህ ፕሮጀክት በተመራማሪዎች በአካዳሚክ፣ በኢንዱስትሪ እና በግለሰብ ጉዳዮች ላይ የሚያጋጥሟቸውን እጅግ በርካታ ፈተናዎችን ለመፍታት የተነደፈ ሁሉን አቀፍ የምርምር እና የፕሮጀክት አስተዳደር መድረክን ለማዘጋጀት ያለመ ነው። የተማከለ ማዕከል በማቅረብ መድረኩ የምርምር የሕይወት ዑደትን ከዕቅድ እና ፕሮፓዛል ልማት እስከ አፈጻጸም፣ የመረጃ ትንተና እና ስርጭትን ያመቻቻል።

መድረኩ እንደ የመረጃ ሰነዶች አለመመጣጠን፣ የእጅ ሂደት ቅልጥፍና እና በምርምር ውጤቶች እና በድርጅታዊ ውሳኔ አሰጣጥ መካከል ያለውን የውህደት ክፍተት የመሳሰሉ ጉዳዮችን ለመቀነስ ይፈልጋል። ቁልፍ ባህሪያት የፕሮጀክት መፍጠር እና ማስተዳደር፣ ክትትል እና መርሐግብር፣ ሊበጁ የሚችሉ የመረጃ መሰብሰቢያ ቅጾች፣ የትብብር መሳሪያዎች እና ለመረጃ ትንተና ጠንካራ ዳሽቦርድ ያካትታሉ።

በተሟላ የፍላጎት ስብስብ፣ ዝርዝር ዲዛይን፣ ተደጋጋሚ እድገት እና ጥብቅ መከራ፣ ይህ መድረክ ተማሪዎችን፣ ዩኒቨርሲቲዎችን፣ የምርምር ድርጅቶችን እና የግለሰብ ተመራማሪዎችን ጨምሮ የተለያዩ የተጠቃሚ ቡድኖችን ልዩ ፍላጎቶች ያሟላል። የተቀናጀ እና ቀልጣፋ የምርምር አካባቢን በማሳልበት መድረኩ የምርምር ትብብሮችን አጠቃላይ ምርታማነት እና ተፅእኖን በማሳልበት በተለያዩ መስኮች ለፈጠራ መፍትሄዎች እና እድገቶች መንገድ ጠርቷል።

ፕሮጀክቱ በአሁኑ ጊዜ በመረጃ በተደገፈ ዓለም ውስጥ የተዋሃደ የምርምር አስተዳደር ሥርዓት አስፈላጊነትን አጉልቶ ያሳያል እና ወደፊት ሊላመድ የሚችል እና ቀጣይነት ያለው መሻሻልን በማስቀመጥ የወደፊት ተግዳሮቶችን ይገመታል። ምንም እንኳን የአይኔስ ውህደትን፣ የሞባይል አፕሊኬሽን ልማትን እና ሌሎች የላቁ ባህሪያትን በመነሻ ወሰን ሳይጨምር፣ መድረኩ ለወደፊት ማሻሻያዎች ጠንካራ መሰረት ይጥላል፤ በሂደት ላይ ባለው የምርምር መልክዓ ምድር ላይ ያለውን ጠቀሜታ እና ውጤታማነት ያረጋግጣል።

# Chapter 1: Introduction

## 1.1. Background

In today's rapidly evolving research landscape, the need for an integrated and efficient research management system has become increasingly apparent. Researchers across various disciplines, including organizations conducting extensive research, students engaged in paper research, individual researchers, and universities, face numerous challenges in managing, tracking, and developing their research projects.

This project aims to address these challenges by developing a state of the art Comprehensive Research and Project Management Platform. The platform will serve as a centralized hub for researchers from diverse backgrounds, providing them with the tools and functionalities needed to enhance the entire research lifecycle. The primary focus will be on solving key problems encountered by researchers, thereby fostering a more favorable environment for impactful and successful research endeavors.

Organizations conducting research projects often struggle with the complexity of coordinating multiple teams and managing extensive datasets. Students working on their research thesis while obtaining their degree encounter challenges in organizing and documenting their paper research effectively. Individual researchers may face difficulties in collaboration and resource sharing, hindering the progress of their work. Universities, as hubs of research and innovation, require simplified systems to manage the diverse range of research activities within their academic communities.

The Comprehensive Research and Project Management Platform strives to be an inclusive solution by meeting the various needs of universities, students, organizations, and individual researchers. In order to promote cooperation, increase efficiency, and create a more cohesive research environment, it attempts to offer a unified and user-friendly interface that meets the unique needs of every user group. This platform aims to ensure its relevance and effectiveness in the ever changing field of research management by anticipating and adapting to future challenges in addition to satisfying the needs of the research community today.

According to the UN's Department of Economic and Social Affairs statistic, in Research and development expenditure as a percentage of GDP worldwide in 2020 was 19. And researchers per million inhabitants were 1341.8. As for Ethiopia the Research and development expenditure as a percentage of GDP in 2017 was 0.3 while the researchers per million inhabitants was 90.5[2].

The development of a Comprehensive Research and Project Management Platform is driven by the increasing need for effective management of research collaborations, as highlighted in Gabriele Bammer's paper Enhancing research collaborations: Three key management challenges[10]. As research collaborations become increasingly complex and global, traditional management approaches are often inadequate. The platform addresses this challenge by providing a centralized platform for managing all aspects of the research lifecycle, from planning and proposal development to execution, data analysis, and dissemination. By effectively harnessing the diversity of research collaborators, setting defensible boundaries, and gaining legitimate authorization, the platform aims to enhance the productivity and impact of research collaborations.

## 1.2. Statement of the problem

The development of a comprehensive research and project management platform is confronted by a series of intricate challenges in research data management. Critical hindrances revolve around data documentation, storage, integration, and security, resulting in impediments to efficient Research Data Management practices. Disruptions within the data flow possess substantial implications for data sharing initiatives. The labor intensive nature of documenting datasets compounds the issue, as researchers commonly disregard standardization, conventions, and metadata in data formatting, thereby restricting the potential for secondary data utilization.

Furthermore, the complexity inherent in research projects is compounded by the prevalence of repetitive, manual tasks that could be automated through software solutions. The reliance on manual processes within research management leads to inefficiencies, consuming valuable time that could be optimally utilized. For example, all research related documents that can be templated should. Otherwise, it results in redundant efforts across organizational teams, leading to a waste of resources and inconsistencies in project documentation.

Moreover, despite the completion of research endeavors, there exists a substantial gap in effectively integrating research outcomes into organizational decision making processes. This gap manifests in the difficulty in locating finalized research, research becoming outdated due to

prolonged timeline from concept to implementation, instances of duplicated research efforts, and, in extreme scenarios, a disregard for research outcomes in decision making procedures.

Addressing these challenges is essential for the successful development of a comprehensive research and project management platform, one that streamlines data management, automates routine tasks, encourages standardization, and ensures the seamless integration of research outputs into actionable organizational decisions.

## 1.3. Objectives

### 1.3.1 General Objectives

To design and develop a comprehensive research and project management platform that will enhance the productivity, efficiency, and impact of research collaborations.

### 1.3.2 Specific Objectives

- Planning Phase: Developing a comprehensive project plan that outlines the project scope, deliverables, timeline, and budget. Defining clear user personas and use cases to ensure the platform addresses the specific needs of researchers.
- Requirements Gathering Phase: Conduct thorough requirement gathering sessions with key stakeholders, including researchers, project managers, and system users. Interview researchers to understand their specific needs and expectations from the platform. Document and prioritize functional and nonfunctional requirements based on stakeholder input.
- Design Phase: Create wireframes and prototypes to visualize the platform's user interface and functionality. Develop a detailed system architecture that outlines the platform's components, interactions, and data flow.
- Development Phase: Implement the platform's core functionalities, including user management, project management, data management, and collaboration tools. Develop a comprehensive testing plan to ensure the platform's stability, performance, and security.
- Testing Phase: Conduct user acceptance testing with a representative group of researchers to ensure the platform meets their needs. Address all critical issues identified during user acceptance testing and perform regression testing to ensure stability. Obtain final approvals from stakeholders and prepare for deployment.
- Deployment Phase: Deploy the platform to a production environment and monitor its performance and usage. Provide user training and support materials to ensure a smooth transition to the new platform. Gather user feedback and make continuous improvements to the platform based on user needs.

## 1.4. Scope

This project aims to develop a research based platform for planning, tracking, monitoring and executing research and other projects. The platform will also aid in implementation of survey and data collection. Our clients will be people and companies working on different research based projects.

As a comprehensive project and research planning platform we will develop a web based application for planning and executing research and other projects. Our project promises a suite of visionary deliverables, including a web application with the following features:

- Project creation and management
- Project tracking and scheduling
- Notification and reminders
- Customizable data gathering form builder
- Search and filtering
- Dashboard and analysis
- Collaboration tools to teams in the research
- Document generation

We will work on the above deliverables in collaboration as a team and using tools that will help to accomplish the tasks with the specified time limits. Our clients will be people and companies working on research related projects. There are exception or out of scope tasks that are not included in the development of this platform these include:

- AI integration
- Hardware development
- Mobile app development
- Integration with every existing system
- In depth training for users
- Language support

The above exceptions are not included because of the reasons listed in the following statements. AI integration, language support and hardware development needs a special knowledge in their domain so, we have to learn and gain a better understanding. For that we have to be dedicated to the learning and acquire the knowledge to add those exceptions and integrate with the in scope deliverables. The mobile app development is out of our scope because it is not needed to meet the project objectives as mobile users will still be able to access it on their browsers.

## 1.5. Methodology

### **Planning methodology:**

The overall project has to be planned before it begins, so we use a waterfall approach which adopts a sequential and linear project development model with distinct phases: Requirements, Design, Implementation, Testing, Deployment. We will define the roles and responsibilities of team members. Perform detailed documentation at each stage. Deciding the tools and making sure they are standardized.

### **Requirements Gathering Methodology:**

To gather requirements for the project we will perform the following techniques: structured Interviews, surveys and questionnaires, and user feedback sessions.

### **Design Methodology:**

For designing the whole system we will use the following three methods: object oriented design (OOD), clean architecture, user centered design (UCD).

### **Development Methodology:**

System implementation strategies include:

- Foster Pair Programming for collaborative coding.
- Leverage Version Control (Git, GitHub) for efficient code tracking and collaboration.
- Utilize Project Tracking Tools (Notion) for task management.
- Employ Integrated Development Environments (IDEs) (Visual Studio, Visual Studio Code) for coding and debugging.
- Facilitate Collaboration with Communication Tools (GoogleMeet, Telegram) for seamless team communication and document sharing.

Tools used for implementation:

- FrontEnd: Use Nextjs for building dynamic and interactive user interfaces.
- BackEnd: Opt for ASP.NET Core to construct robust and scalable web applications.
- Database: Select a database technology based on scalability and performance needs (MySQL).

Coding Best Practices:

- Adhere to Clean Code principles for writing maintainable and readable code.

## **Testing Methodology:**

### Hybrid Testing Approaches:

- Conduct Blackbox testing for functionality without internal code knowledge.
- Perform White Box testing to ensure internal code structure aligns with functionality and quality expectations.

### Code Quality Assurance:

- Employ Static Code Analysis tools (SonarQube, Lint) for bug and vulnerability identification.
- Conduct Code Reviews by other developers to enhance code quality.

### Manual Testing Focus:

- Implement Functional Testing to verify all features operate as intended.
- Conduct Usability Testing to ensure the platform's user friendliness and ease of understanding.

## **Deployment Methodology:**

### Continuous Integration and Continuous Delivery (CI/CD):

- Implement GitLab for automating the build, test, and deployment process.
- Deploy frontend on vercel.

## **1.6. Plan of Activities**

This project is anticipated to be finished in six months, beginning on November 27, 2023 with a tentative completion date of June 15, 2024. Below, we have divided the project into 4 phases of activities:

### **Phase 1: Project Initiation (November 27,2023 - December 15,2023)**

During the Project Initiation phase, we will define the project scope, objectives, and stakeholders. This phase involves setting up the project infrastructure, assembling the project team, and conducting initial stakeholder meetings to ensure everyone is aligned with the project goals. Key activities include developing a comprehensive project plan, defining user personas and use cases, selecting a project management methodology, and identifying potential risks. By the end of this phase, we should have a clear understanding of the project requirements and be ready to move into the Planning and Design phase.



## **Phase 2: Planning and Design (December 16,2023 – January 30,2024)**

The Planning and Design phase focuses on gathering detailed requirements, designing the system architecture, and creating prototypes to visualize the platform's user interface and functionality. This phase also involves conducting stakeholder meetings to prioritize requirements and obtain necessary approvals. Key activities include conducting requirement gathering sessions with key stakeholders, creating wireframes and prototypes, developing a detailed system architecture, technology selection and obtaining approvals from stakeholders.

## **Phase 3: Development (February 21,2024 – May 30,2024)**

The Development phase focuses on implementing the platform's core functionalities and developing a comprehensive testing plan and documenting technical specifications. This phase involves coding, building, and integrating various components of the platform, both on frontend and backend development as well as conducting unit testing to identify and fix software defects.

## **Phase 4: Testing and Deployment (May 30,2024 – June 20,2024)**

The Testing and Deployment phase focuses on ensuring the platform's stability, performance, and security, as well as deploying the platform to a production environment and providing user training and support. This phase involves conducting various types of testing, including integration testing, system testing, and user acceptance testing, as well as deploying the platform, monitoring its performance and usage, and gathering user feedback for continuous improvements. Key activities include conducting user acceptance testing with a representative group of researchers, addressing critical issues identified during testing, deploying the platform to a production environment, providing user training and support materials, and gathering user feedback for future enhancements.

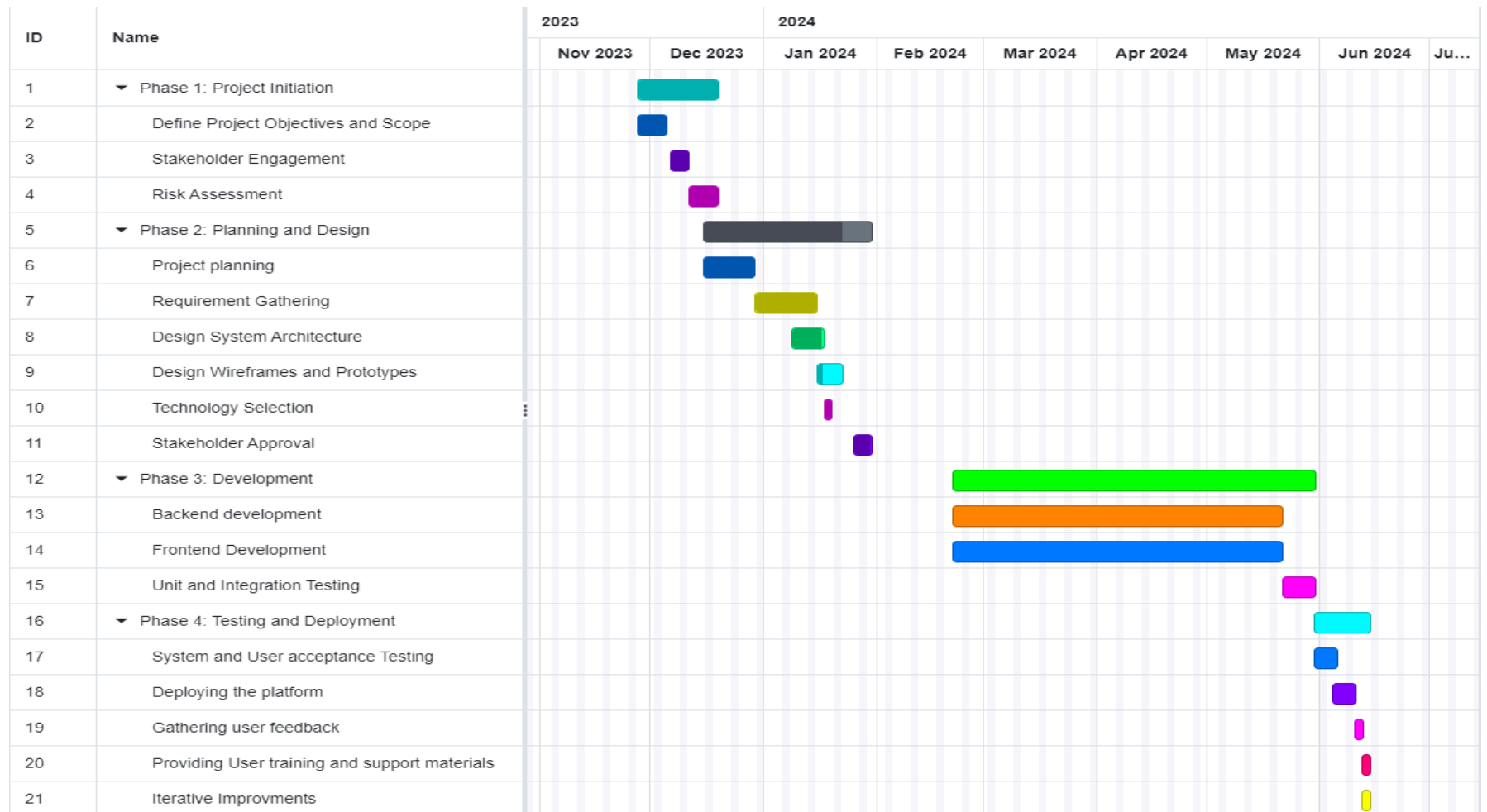


Figure 1.1: Plan of activities Gantt chart

## 1.7. Budget Required

Our project, a comprehensive project and research planning platform, is a collaborative effort by a team of enthusiastic learners aspiring to enhance their technological skills and knowledge. Given the educational nature of our endeavor, personnel costs for roles such as developers, designers, QA/testers, and documentation specialists are zero, as the team is committed to self-learning and growth. We intend to leverage free technologies and tools available in the market for cloud services, development tools and licenses, Project management tools, Collaboration tools, and testing tools. As a result, our budget requirements are nearly zero, with the only anticipated expenses being related to hosting the project and miscellaneous costs that may include various small expenses that don't fit into specific categories but are still essential for the project's success.

To minimize costs we are focusing mainly on hosting and cloud-related expenses, here's a simplified budget estimate in tabular form:

Category	Description	Estimated Cost
Hosting	Cloud hosting services, including Domain Registration (shared hosting)	1980 - 2000 birr
Cloud Databases	Database services (MySQL Server)	free
SSL Certificate	Secure Sockets Layer certificate for website security	free SSL (0 SSL)
Miscellaneous	Printing and Documentation, Travel Expenses, Communication Expenses, Stationery and Supplies, Internet Expenses, Contingency Fund	2,500 - 5000 birr
Total Estimated Cost		4,480 - 7,000 birr

Table 1: Estimated Expenses

## 1.8. Significance of the Study

The concept of a research and project management platform emerges as a transformative force, poised to revolutionize the way organizations conceptualize, execute, and disseminate research endeavors. The significance of such a platform could be of its role as an organizational tool, strategic enabler, and catalyst for collaboration and innovation.

At its core, the research management platform serves as a centralized hub, orchestrating the complexities inherent in research project management. By offering a comprehensive suite of features and functionalities, it streamlines workflows, enhances efficiency, and fosters seamless communication among stakeholders. From project inception to dissemination, the platform provides a cohesive framework for managing research activities, ensuring alignment with organizational objectives, and driving tangible outcomes.

Strategically, the research management site serves as a linchpin for achieving organizational goals and objectives[11]. Through robust strategic planning tools, resource allocation mechanisms, and performance monitoring capabilities, it empowers institutions to articulate and execute their strategic visions with precision. By providing real-time insights, data-driven decision-making, and predictive analytics, the platform enables organizations to adapt to dynamic research landscapes, capitalize on emerging opportunities, and mitigate risks effectively.

Moreover, the platform caters to the diverse needs and requirements of individual departments and accounting units helping the researcher focus on academic goals[11]. By offering customizable modules, reporting functionalities, and accounting tools, it empowers departments to manage budgets, track expenditures, and streamline processes with ease. In doing so, it ensures compliance with institutional policies, regulatory standards, and best practices, thereby fostering a culture of accountability and transparency across the organization.

Another reason for the creation of the research management platform is the increased pressure to demonstrate quality and relevance to research production[7]. Its role as a quality assessment hub, providing a robust framework for evaluating and peer-reviewing research outputs. Through advanced assessment tools, metrics tracking, and peer collaboration features, it ensures adherence to quality standards, promotes transparency, and fosters a culture of excellence across research endeavors. By facilitating rigorous evaluation and feedback mechanisms, the platform empowers researchers to refine their methodologies, validate their findings, and enhance the rigor and reproducibility of their work.

Equally significant is the platform's role in advancing the dissemination and accessibility of research outputs on a global scale[9]. Leveraging advanced dissemination mechanisms, open

access repositories, and digital archives, it serves as a gateway to making research outputs widely available to diverse audiences. By enhancing visibility, impact, and scholarly communication, the platform accelerates the pace of discovery, fosters interdisciplinary collaboration, and catalyzes innovation across diverse academic and scientific domains.

Furthermore, the research management site serves as an ideal platform for universities to establish their research repositories, showcasing institutional research outputs, fostering collaboration, and preserving intellectual capital[9]. Through customizable templates, branding options, and administrative controls, universities can curate, showcase, and promote their research assets effectively, enhancing their visibility and reputation within the academic community.

# Chapter 2: Literature Review

## 2.1 Overview

This literature review delves into different aspects of research management, project management, and research management platforms and methods. It brings together information from various sources to offer a thorough understanding of how these areas have evolved, pinpointing important milestones, obstacles, areas for improvement, and insights gained. By exploring how data management and project management intersect, this review underscores the value of cohesive strategies that boost the efficiency and impact of research.

## 2.2 Project Management

Project management is the application of knowledge, skills, tools, and techniques to project activities to meet the project requirements. It is a distinct area of management that helps organizations achieve their goals and objectives by effectively managing projects. Project management ensures that project goals are achieved within the constraints of time, cost, and scope, thereby delivering value and benefits to the organization and its stakeholders.

### 2.2.1 Project

A project is defined as a temporary endeavor undertaken to create a unique product, service, or result. Unlike ongoing operations, a project has a definite beginning and end, and it is undertaken to fulfill specific objectives. The temporary nature of projects means that they have clear start and finish points, which are usually defined by project milestones. The uniqueness aspect indicates that each project differs in some way from other projects, even if they are similar in nature. The success of a project is measured by the achievement of its objectives, which are typically specified in terms of scope, time, and cost constraints.

### 2.2.2 Key Elements of Project Management

Project management involves several key elements, which collectively ensure the successful completion of projects[8]:

#### **Project Stakeholders**

Project stakeholders are individuals or organizations actively involved in the project or whose interests may be affected by the project. These stakeholders include project sponsors, project managers, project teams, customers, and other parties who have a vested interest in the project's outcome. Effective stakeholder management involves identifying stakeholders, understanding their needs and expectations, and ensuring their engagement throughout the project lifecycle to secure their support and minimize opposition.

## **Knowledge Areas**

Project management encompasses ten knowledge areas, each representing a critical competency that contributes to successful project execution. These knowledge areas are:

1. Integration Management: Ensuring that project components are effectively coordinated.
2. Scope Management: Defining and controlling what is and is not included in the project.
3. Time Management: Planning and controlling the project schedule.
4. Cost Management: Estimating, budgeting, and controlling project costs.
5. Human Resource Management: Organizing, managing, and leading the project team.
6. Communications Management: Ensuring timely and appropriate generation, collection, and dissemination of project information.
7. Stakeholder Management: Identifying and managing the needs and expectations of project stakeholders[8].

## **Tools and Techniques**

Project management employs various methodologies and tools to plan, execute, and monitor project activities. Common tools include Gantt charts, which provide a visual timeline of project tasks. Additionally, project management software applications facilitate detailed planning, scheduling, and tracking of project activities.

## **Project Success Factors**

Several factors contribute to the success of a project. Clearly defined objectives provide direction and focus. Effective communication ensures that information flows smoothly among all stakeholders. Stakeholder engagement ensures that all parties are on board with the project goals and processes. Efficient resource allocation ensures that the necessary resources are available when needed and used effectively[8].

### **2.2.3 The Project Management Process**

Project management can be broken down into several phases, each comprising specific steps and activities. These phases are initiating, planning, executing, monitoring and controlling, and closing[8].

#### **Initiating Projects**

The initiation phase involves defining and authorizing the project. Key activities in this phase include:

**Defining the Project:** This step involves identifying the project's purpose, objectives, and scope. It includes determining the feasibility of the project and aligning it with organizational goals. This ensures that the project is viable and worth pursuing.

Stakeholder Identification: Identifying all stakeholders who will be affected by or have an interest in the project. Understanding their needs, expectations, and influences is crucial for securing their support and addressing their concerns throughout the project lifecycle.

## **Planning Projects**

The planning phase involves developing a detailed roadmap to achieve the project objectives. It includes several critical steps:

1. **Scope Planning:** Defining and documenting the project scope, including deliverables, boundaries, and acceptance criteria. This ensures that all stakeholders have a clear understanding of what the project will and will not achieve.
2. **Time Planning:** Developing a project schedule that outlines the timeline, milestones, and critical path of the project. This involves breaking down the project into smaller, manageable tasks, estimating the duration of each task, and determining their dependencies.
3. **Cost Planning:** Estimating the costs associated with project activities and developing a budget. This includes identifying all cost elements, such as labor, materials, equipment, and overheads, and ensuring that the project is financially viable.
4. **Resource Planning:** Identifying and assigning resources, including personnel, documents, and materials. This step ensures that the necessary resources are available when needed and are used efficiently.
5. **Communication Planning:** Developing a communication plan to ensure effective information dissemination among stakeholders. This includes identifying communication needs, defining the methods and frequency of communication, and establishing protocols for information sharing[8].

## **Executing Projects**

The execution phase involves coordinating people and resources to carry out the project plan. Key activities in this phase include:

- **Task Execution:** Coordinating people and resources to execute the project plan and complete the project tasks. This involves managing the project team, assigning tasks, and ensuring that work is performed as planned.
- **Team Development:** Building and managing the project team to enhance performance and productivity. This involves providing training, motivation, and support to team members, as well as resolving conflicts and fostering a collaborative work environment.



## **Monitoring and Controlling Projects**

The monitoring and controlling phase involves tracking project performance and making necessary adjustments to ensure that the project stays on track. Key activities in this phase include:

- Performance Monitoring: Tracking project performance against the project plan to identify variances. This involves measuring project performance, comparing it to the plan, and identifying any deviations.
- Change Control: Managing changes to the project scope, schedule, and costs. This involves evaluating change requests, assessing their impact, and approving or rejecting changes as necessary.

## **Closing Projects**

The closing phase involves finalizing all project activities and formally closing the project. Key activities in this phase include:

- Project Closure: Finalizing all project activities, including obtaining formal acceptance of deliverables from stakeholders and documenting lessons learned. This ensures that the project is formally completed and that all objectives have been met.
- Post-Implementation Review: Conducting a post-implementation review to assess project success and identify areas for improvement. This involves evaluating project performance, documenting successes and challenges, and providing recommendations for future projects.[8]

## **2.3 Research Management**

Research management is a critical component within academic institutions, influencing the productivity and efficiency of research activities. This process encompasses a range of activities aimed at planning, directing, and controlling research efforts to optimize research output.

### **2.3.1 Research**

Research is a methodical way of investigating, understanding, and refining information or concepts. It entails employing scientific approaches to arrive at conclusions about phenomena across diverse domains like science, humanities, social sciences, and beyond. The main objectives of research are to broaden understanding, address particular issues, and make valuable contributions to the advancement of theories.

### **2.3.2 Components of Research Management**

Research management involves both tangible and intangible factors that contribute to the success of research activities.

**Tangible Factors:**

1. Organizational Structures: The establishment of research offices and departments that oversee research activities and ensure compliance with institutional and governmental policies.
2. Funding: Allocation of financial resources to support research projects, including grants, scholarships, and institutional funding.
3. Facilities and Equipment: Provision of necessary infrastructure and tools required for conducting research, such as laboratories, libraries, and technological equipment.
4. Planning and Evaluation: Development of detailed plans for research projects, including measurable targets and rigorous evaluation methods to assess progress and outcomes[6].

**Intangible Factors:**

1. Human Resources: Recruitment and management of skilled researchers and support staff. This includes fostering a collaborative environment and providing opportunities for professional development.
2. Culture and Relationships: Building a culture that values research, encouraging collaboration among researchers, and managing relationships within and outside the institution.
3. Motivation and Commitment: Ensuring that researchers are motivated and committed to their work through recognition, rewards, and a supportive research environment[6].

### 2.3.3 Steps in Research Management

**Planning and Proposal Development**

Planning is the foundational phase of research management, setting the stage for successful project execution. This phase includes:

- Identifying Research Opportunities: This involves scanning the academic landscape to identify gaps in knowledge and areas where research can make a significant impact. Institutions often align these opportunities with their strategic goals and societal needs. Researchers conduct literature reviews, attend conferences, and collaborate with peers to pinpoint these opportunities.
- Developing Proposals: Once a research opportunity is identified, the next step is to develop a comprehensive research proposal. This document outlines the research objectives, questions, hypothesis, methodologies, expected outcomes, and resource requirements. A well-crafted proposal also includes a literature review that justifies the need for the research, a detailed budget, and a timeline. The proposal serves as a blueprint for the research project and is often used to secure funding[6].

## **Funding and Resource Allocation**

Securing adequate funding and allocating resources effectively are crucial for the success of research projects.

- **Securing Funding:** Researchers and institutions apply for funding from various sources, including government agencies, private foundations, industry partners, and internal grants. This step involves writing and submitting grant applications, which must align with the priorities and requirements of the funding bodies. Successful applications are clear, compelling, and demonstrate the potential impact of the research.
- **Allocating Resources:** Once funding is secured, the next step is to allocate resources efficiently. This includes financial resources, human resources (research staff, administrative support), and material resources (equipment, facilities). Proper resource allocation ensures that the project has everything it needs to proceed smoothly and that resources are used optimally without wastage[6].

## **Project Implementation**

The implementation phase is where the actual research takes place. It involves the following steps:

- **Conducting Research:** Researchers carry out the activities outlined in the research plan. This includes data collection, experiments, surveys, fieldwork, and other methodologies specific to the research. Throughout this phase, researchers must adhere to ethical standards and institutional guidelines.
- **Monitoring Progress:** Regular monitoring is essential to ensure that the research stays on track. This involves tracking milestones, assessing interim results, and making necessary adjustments to the research plan. Monitoring helps identify any deviations from the plan early on, allowing for corrective actions to be taken in a timely manner[6].

## **Evaluation and Reporting**

Evaluating the research outcomes and disseminating the findings are critical steps in research management.

- **Evaluating Outcomes:** After the research is completed, the results are evaluated against the original objectives and expected outcomes. This involves analyzing data, interpreting results, and determining whether the research hypotheses were supported. Evaluation also includes assessing the overall impact of the research and identifying lessons learned.
- **Disseminating Findings:** Effective dissemination ensures that the research reaches the intended audience and contributes to the broader body of knowledge. This can be achieved through various channels, including academic journals, conferences, seminars,

workshops, and public engagement activities. Additionally, researchers may produce reports, policy briefs, and other materials tailored to specific stakeholders[6].

## **Continuous Improvement**

The final phase of research management focuses on learning and improvement.

- **Feedback and Learning:** Gathering feedback from all stakeholders involved in the research process is essential for continuous improvement. This includes feedback from funding agencies, research participants, peer reviewers, and institutional leaders. Researchers should critically analyze this feedback to identify strengths and areas for improvement.
- **Updating Policies and Procedures:** Institutions should use insights gained from completed research projects to update and refine their research management policies and procedures. This might involve revising guidelines for proposal development, improving resource allocation strategies, or enhancing monitoring and evaluation practices. Continuous improvement ensures that future research projects are managed more effectively and efficiently[6].

## **2.4 Study related works**

In their paper titled "A comparison of research data management platforms: architecture, flexible metadata, and interoperability," Ricardo Carvalho Amorim et al. delve into the increasingly critical realm of research data management (RDM) platforms[5]. Their work sheds light on the pressing need for robust systems that support researchers in organizing and preparing their data for publication. Amorim et al. examine various platforms, ranging from institutional repositories to more sophisticated environments tailored for diverse research workflows. Through a comprehensive evaluation, they assess these platforms based on key criteria such as architecture, metadata support, programming interfaces, search mechanisms, and community acceptance. They also acknowledge the complexities inherent in data description across different domains and advocate for better integration with existing research management tools.

In "Globus: Research Data Management as Service and Platform," Kyle Chard, Ian Foster, and Steven Tuecke discuss the evolution and application of Globus, a suite of data and user management capabilities designed for the research community[4]. They tackle challenges surrounding data transfer, sharing, authentication, and publication, presenting Globus as a solution that has expanded beyond mere data transfer to offer a comprehensive suite of services. The authors emphasize the shift towards service-based models in scientific software development and advocate for interoperability between services. Globus is positioned as a versatile platform for scientific services, offering robust APIs and user-friendly software libraries for developers.

The literature on RDM underscores a growing interest in the field, driven by the escalating demand for open and reusable data. Complexity reigns supreme in RDM practices, with inconsistencies in conceptualization and implementation posing significant challenges. While guidelines exist, their variability and the demanding nature of adherence present hurdles for researchers, who often lack the time and requisite skills for effective data management. Moreover, the authors point out that Data Management Plans (DMPs), intended to delineate how data will be managed and shared during research, often devolve into perfunctory exercises devoid of tangible benefits.

## 2.5 Identifying milestones of the related literatures and finding the gaps

Amorim et al.'s review highlights several milestones in the development of research data management platforms, emphasizing the transition towards more comprehensive solutions capable of accommodating diverse data types and workflows[5]. While platforms excel in certain aspects, such as architecture or collaboration features, gaps persist in domain-specific metadata support, integration with existing tools, and community acceptance. The review also underscores the challenges of involving researchers in metadata production, stressing the importance of domain knowledge in adequately documenting datasets for reuse.

The evolution of Globus serves as another significant milestone in the literature, reflecting a shift towards service-based models in scientific software development. The authors identify key developments within Globus, such as the introduction of Globus Auth for authentication and authorization, the evolution of Globus Connect for data transfer and sharing, and the incorporation of data publication capabilities. Despite its successes, gaps remain, particularly concerning the lack of commonality between existing scientific services, hindering interoperability and user experience. Standardized protocols and interoperable solutions are deemed essential to address these challenges[4].

In the broader landscape of RDM literature, milestones include a heightened recognition of data management's importance as research becomes increasingly data-intensive. However, significant gaps persist in understanding researchers' holistic RDM practices throughout the research process. Existing literature predominantly focuses on data sharing practices, leaving a void in comprehensive insights into researchers' challenges and practices. This review seeks to bridge this gap by analyzing relevant literature specifically centered on researchers' RDM practices throughout the research lifecycle.

## 2.6 Lessons learned from literatures

The literature offers valuable lessons for the development and implementation of comprehensive research data management platforms. Foremost among these is the imperative to consider stakeholders' requirements and community standards in platform design and selection. Platforms offering flexibility, customization options, and support for domain-specific metadata are more likely to meet the diverse needs of researchers and institutions. Furthermore, user engagement and collaboration tools play a pivotal role in facilitating data management workflows, highlighting the importance of seamless integration into researchers' daily activities.

The case of Globus underscores the importance of addressing challenges holistically, from authentication to data publication, to provide comprehensive solutions for the research community. Successful platforms prioritize collaboration, interoperability, and sustainability, fostering a conducive environment for advancing research data management infrastructure.

Challenges in researcher RDM practices span the entire research data lifecycle, encompassing issues such as alignment with research practices, resourcing, researcher openness, and data governance. Barriers like lack of standardized documentation, inadequate training, and concerns about data security inhibit effective data sharing and reuse. Understanding these challenges is pivotal for developing strategies to promote data sharing and maximize the utility of research data.

## Chapter 3: Problem Analysis and Modeling

### 3.1 Existing system and its problems

The research landscape offers a few systems and tools to simplify data management, collaboration, and workflow optimization. This section explores some of them.

#### 3.1.1 Existing Systems

In research project management, several existing systems and tools aim to facilitate various aspects of data management, collaboration, and workflow optimization. These systems cater to the diverse needs of researchers, providing functionalities ranging from data organization to project tracking and collaboration.

For instance, Electronic Lab Notebooks (ELNs) such as LabArchives, Benchling, and Labguru serve as digital platforms meticulously tailored for researchers. These platforms offer a centralized space to document experiments, manage protocols, and organize voluminous research data. These systems allow for structured data entry, file attachments, and collaboration among team members, fostering efficient laboratory workflows.

Additionally, Reference Management Software including tools like EndNote, Mendeley, and Zotero play a crucial role in aiding researchers in the management of references and citations. These platforms streamline the collection, organization, and citation of research materials, simplifying the complexities inherent in literature review processes and citation management within academic work.

Furthermore, Data Repository and Archiving Systems such as Figshare, Zenodo, and Dryad serve as pivotal infrastructures tailored for the storage and sharing of research data. These repositories offer robust solutions for long term data archiving, ensuring not only the preservation and accessibility of research data but also compliance with the stringent requirements associated with data sharing and publishing.

#### 3.1.2 Major problem of existing systems

One of the main issues researchers face with existing systems is the fragmented workflows and data silos. Many existing systems do not smoothly communicate with one another, causing information to remain isolated in one place and making it difficult to transfer where it's needed. Using these tools and making them work together can also be problematic. Sometimes, these

systems are difficult to use, or they don't sync well with other tools. This presents a challenge for researchers who struggle to seamlessly integrate these systems.

Another critical problem with these systems is that while they might address certain parts of the research process, they fail to provide comprehensive solutions. For example, some systems help store data but do not adequately organize it, resulting in difficulties locating information later on. Additionally, a significant amount of researchers' time is consumed by repetitive tasks that could be handled more efficiently by technology. Writing the same type of research document repeatedly wastes valuable time that could be better utilized.

Furthermore, even after conducting valuable research, the findings are not always effectively utilized in decision making processes. This means that important research might go unnoticed or be overlooked when it could genuinely make a significant impact[1][3].

## 3.2 Specifying the Requirements of the proposed solution

### 3.2.1 Functional Requirement

#### **System admin requirements:**

**SAR1:** allow the admin to login with the correct username and password.

**SAR2:** allow the admin to get the status of the system that is accessing the dashboard.

**SAR3:** allow the admin to configure system settings, including user roles, permissions, and general system preferences to ensure the system aligns with organizational needs.

**SAR4:** allow the admin to create, edit, and manage user accounts within the system. This includes adding new users, updating user information, and managing access permissions.

**SAR5:** allow the admin to implement security measures, monitoring system performance, and addressing any potential vulnerabilities.

#### **User requirements:**

There are various users of the system such as: project owner, project manager, supervisor, collaborator and researcher. We will see how the system performs for these users.

#### **Project manager requirements:**

**PMR1:** allow the project owner to login with the correct username and password



**PMR2:** allow the project owner to get the home page after login

**PMR3:** after accessing the home page the system should allow the project owner to create projects by entering the required information, create customizable forms, and assign tasks to people.

**PMR4:** allow the owner to add project members.

**PMR5:** allow the project manager to oversee the entire research project life cycle, including planning, resource allocation, task management, budget control, and progress monitoring.

#### **Supervisor requirements:**

**SR1:** allow the supervisor to login with the correct username and password.

**SR2:** allow the supervisor to get the homepage

**SR3:** after that both should get the option to check data correctness, get notification and inbox, search and filter options, create customizable data gathering forms, and assign tasks.

#### **Researcher requirements:**

**RR1:** allow researchers to login with the correct username and password.

**RR2:** allow researchers to get the homepage after login

**RR3:** after that they should get the options to upload and manage research data and receive project notification notifications and reminders for upcoming deadlines, task assignments, or changes in project status.

### **3.2.2 Non-Functional Requirement**

**NFR1. Performance:**

- **Page load times:** 95% of pages must load within 5 seconds on a standard internet connection (estimated around 500 Kbps - 1 Mbps).
- **Response times:** API calls should have a maximum response time of 5 seconds.
- **System uptime:** The platform must be available 94.5% of the time, excluding scheduled maintenance periods.

#### NFR2. Scalability:

- **User load:** The platform must accommodate at least 500 concurrent users without significant performance degradation.
- **Data storage:** The platform must be able to store and manage at least 100 GB of research data.

#### NFR3. Security:

- **Data encryption:** All user data and research data must be encrypted at rest and in transit using industry standard algorithms
- **Authentication and authorization:** Implement robust user authentication and authorization mechanisms to control access to data and functionalities.

#### NFR4. Reliability:

- **Data redundancy:** Implement data replication and backup mechanisms to ensure data integrity and availability in case of hardware or software failures.
- **Disaster recovery plan:** Develop a comprehensive disaster recovery plan to minimize downtime and data loss in case of unforeseen events.

#### NFR5. Usability:

- **User interface:** Design an intuitive and user friendly interface that is easy to navigate and understand for researchers with varying technical skills.
- **Accessibility:** Ensure the platform complies with WCAG 2.1 accessibility guidelines to be usable by users with disabilities.
- **User feedback and testing:** Regularly conduct user testing and gather feedback to identify and address usability issues.

#### NFR6. Interoperability:

- **Data import and export:** Support various data formats for seamless import and export of research data.
- **Compliance with research data standards:** Ensure the platform adheres to relevant research data standards to facilitate data sharing and collaboration.

#### NFR7. Compliance:

- **Data privacy regulations:** Ensure compliance with relevant data privacy regulations, such as GDPR and HIPAA, to protect user privacy and data security.
- **Research ethics guidelines:** Adhere to established research ethics guidelines to ensure ethical research practices within the platform.

## 3.3 System modeling

### 3.3.1 Overview

This chapter delves into the detailed analysis and modeling of the Comprehensive Research and Project Management Platform (CRMPM). Building upon the established feasibility and requirements, starting off on a structured exploration of the platform's functionalities, behaviors, and internal structure. This analysis and modeling will serve as the cornerstone for successful development and implementation, providing a clear roadmap for translating needs into a robust and functional system.

This chapter explores three key dimensions of the CRMPM:

#### 1. Scenario Based Modeling:

We will create use cases and user stories that depict typical user interactions with the platform, simulating how researchers, project managers, and other stakeholders will navigate various tasks and workflows. This approach helps us identify key functionalities, user interfaces, and potential challenges, ensuring the platform meets diverse user needs in realistic scenarios. By leveraging use cases and user stories, we can gain insights into user behavior, preferences, and pain points, enabling us to design intuitive interfaces, streamline workflows, and optimize user experiences. Through iterative testing and refinement, we can iteratively enhance the platform's usability, functionality, and overall effectiveness, aligning it with the evolving needs and expectations of our users.

#### 2. Behavioral/Dynamic Modeling:

We will employ state diagrams and activity diagrams to model the platform's dynamic behavior, visualizing the different states a project or user can be in, the triggers that cause transitions between states, and the actions performed within each state. This modeling

technique ensures smooth transitions between functionalities, avoids inconsistencies, and clarifies the platform's overall operational flow. By leveraging state and activity diagrams, we can gain a comprehensive understanding of the platform's behavior, identify potential bottlenecks or inefficiencies, and optimize the user experience. Through iterative refinement and validation, we can ensure that the platform's dynamic behavior aligns with user expectations and supports seamless navigation and interaction.

### 3. Class Based Modeling:

We will delve into the internal structure of the CRMPM by creating Class Diagrams, which depict the platform's core entities, their attributes, and the relationships between them. This analysis provides a blueprint for software development, defining the building blocks of the system and their interactions, promoting modularity and maintainability. By visualizing the class structure and relationships, we can identify potential design flaws, optimize data flow, and ensure the scalability and extensibility of the platform. Through rigorous analysis and iteration, we can refine the class diagrams to accurately represent the system's architecture, enabling efficient software development and future enhancements.

#### 3.3.2 Scenario Based Modeling

The CRMPM caters to two primary actor categories with distinct roles and functionalities within the platform: Administrator and Standard User. Standard Users are further divided based on their roles within specific projects. The actors are:

##### **1. Administrator:**

###### ➤ Responsibilities:

- Manages user accounts, including creating, editing, and deleting users.
- Configures system settings, including user roles, permissions, and general system preferences.
- Implements security measures to ensure data integrity and system security.
- Monitors system performance and addresses potential vulnerabilities.
- Functionalities:
- Full access to the system for administrative and configuration tasks.

## 2. Standard User:

Subtypes:

### 1. Project Manager:

#### ➤ Responsibilities:

- Oversees the entire research project lifecycle, including planning, resource allocation, task management, budget control, and progress monitoring.
- Communicates with collaborators, funding agencies, and research administrators.

#### ➤ Functionalities:

- Utilizes the platform to create and manage project plans, assign tasks, track progress, generate reports, and ensure project success.
- Adds project members and manages their roles within the project.

### 2. Supervisor:

#### ➤ Responsibilities:

- Guides the research activities of junior researchers, providing mentorship and feedback.
- Manages and oversees specific research tasks and ensures data correctness.

#### ➤ Functionalities:

- Utilizes the platform to manage research activities, create forms, assign tasks, schedule activities, and generate reports.
- Tracks project progress and collaborates with project managers and researchers.

### 3. Normal Researcher:

#### ➤ Responsibilities:

- Participates in various research activities within the project, including data collection, analysis, interpretation, and dissemination.
- Collaborates with other researchers within the project and across institutions.

#### ➤ Functionalities:

- Utilizes the platform to upload and manage research data, store and share data, access project resources, participate in discussions, and contribute to overall research outcomes.
- Receives project notifications and reminders for upcoming deadlines, task assignments, or changes in project status.

## 3.3.2.1 Use Case Description

### 1. Research Project Management:

- Actors: Project Manager, Supervisor, Researcher, Administrator

- Description: This use case represents the overall functionality of managing research projects within the system. It encompasses creating and managing projects, collaboration, task/people assignment, scheduling, generating reports, and receiving project notifications.

## 2. Create Project:

- Actor: Project Manager
- Description: Project Managers can create new research projects by defining project details such as title, description, start/end dates, and associated researchers.

## 3. Add Project Member:

- Actor: Project Manager
- Description: Project Managers can add project members by searching for users within the system and assigning them roles within the project.

## 4. Manage Projects:

- Actors: Project Manager, Supervisor
- Description: Project Managers and Supervisors can oversee and manage existing projects, including editing project details, assigning tasks, and monitoring project progress.

## 5. Upload and Manage Research Data:

- Actors: Project Manager, Supervisor, Researcher
- Description: Researchers, Project Managers, and Supervisors can upload, organize, and manage research data associated with their projects. This includes adding, editing, and deleting data files.

## 6. Assign Tasks/People:

- Actors: Project Manager, Supervisor
- Description: Project Managers and Supervisors can assign specific tasks to researchers, indicating responsibilities and deadlines to ensure effective project management and task distribution.

## 7. Scheduling:

- Actors: Project Manager, Supervisor
- Description: Project Managers and Supervisors can schedule and manage the timelines and activities within the research projects they are coordinating.

## 8. Generate Reports:

- Actors: Project Manager, Supervisor

- Description: Project Managers and Supervisors can generate reports on project progress, resource allocation, and task completion to facilitate project monitoring and decision-making.

#### 9. Create Form:

- Actors: Project Manager, Supervisor
- Description: Project Managers and Supervisors can create customizable forms for data gathering, surveys, or feedback collection to support research activities.

#### 10. Collaborate:

- Actors: Project Manager, Supervisor, Researcher
- Description: Project Managers, Supervisors, and Researchers can collaborate on projects by sharing documents, participating in discussions, and contributing to project-related activities.

#### 11. Track Project Progress:

- Actors: Project Manager, Supervisor, Researcher
- Description: Project Managers, Supervisors, and Researchers can track the progress of their projects, monitor milestones, and ensure that tasks are completed on time.

#### 12. Manage User Accounts:

- Actor: Administrator
- Description: Administrators can create, edit, and manage user accounts within the system. This includes adding new users, updating user information, and managing access permissions.

#### 13. Implement Security Measures:

- Actor: Administrator
- Description: Administrators are responsible for ensuring the overall data security and system integrity. This involves implementing security measures, monitoring system performance, and addressing any potential vulnerabilities.

#### 14. Configure System Settings:

- Actor: Administrator
- Description: Administrators can configure system settings, including user roles, permissions, and general system preferences to ensure the system aligns with organizational needs.

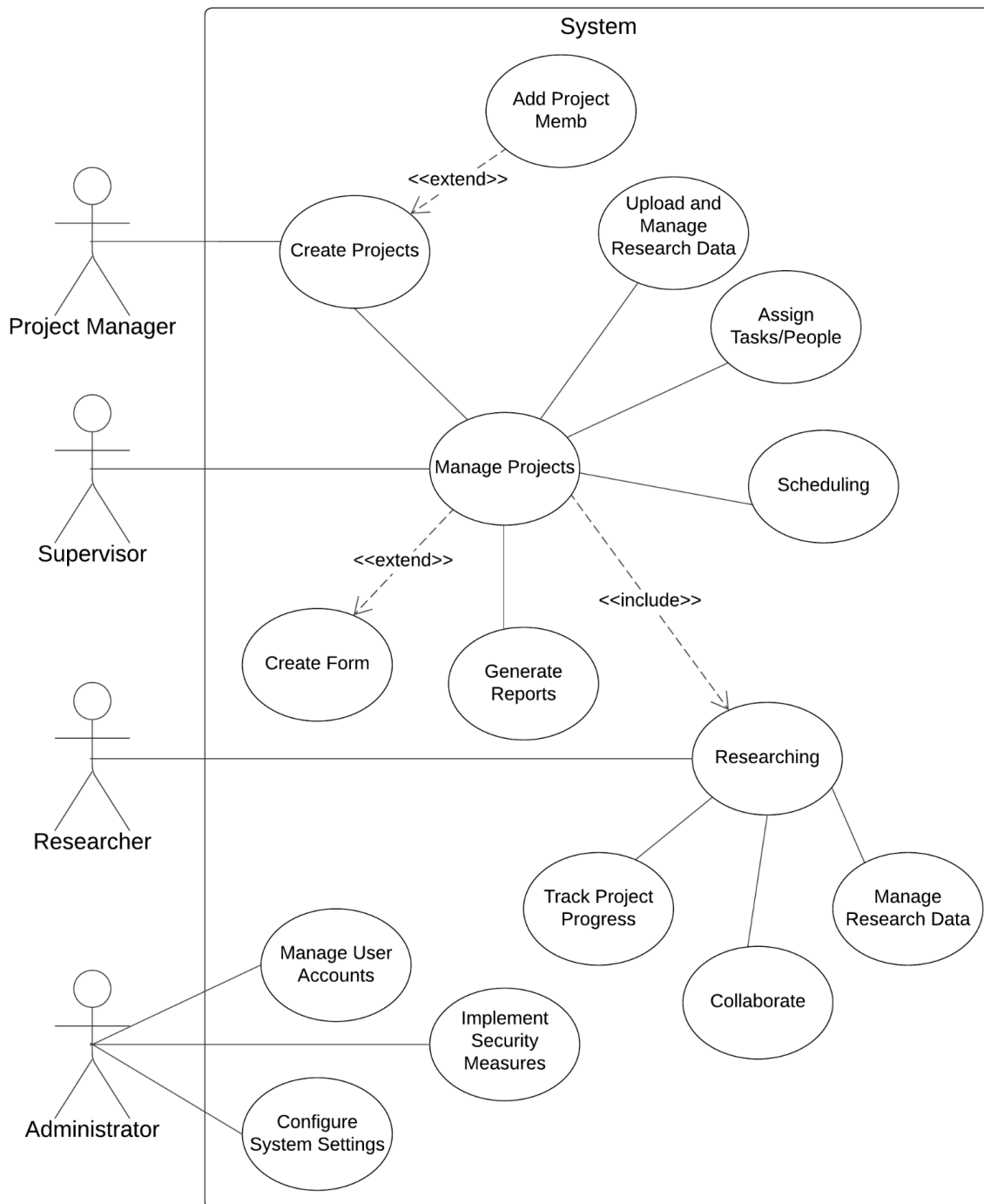


Figure 3.1: Use Case Diagram



### 3.3.2.2 Activity Diagram

#### **1. Researcher activities:**

Researchers login to the system. After that he/she upload and manage research data and receive project notification notifications and reminders for upcoming deadlines, task assignments, or changes in project status, create customizable data gathering forms, save and reuse form templates across different projects.

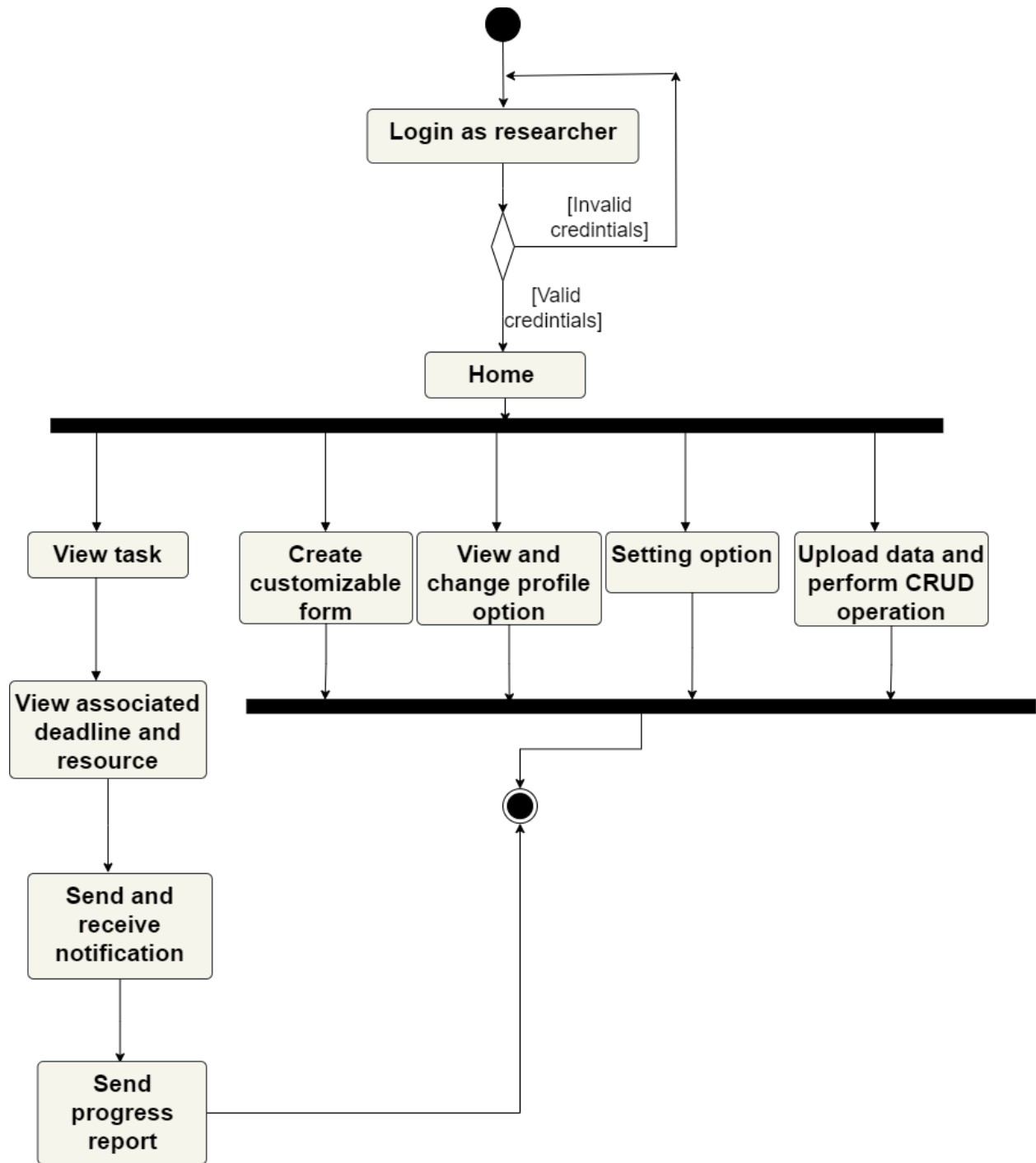


Figure 3.2: Researcher's Activity Diagram

## 2. Project owner activities:

Project owners can create and manage projects with type, title, description, start/end date setting, project plans. They can receive progress reports.

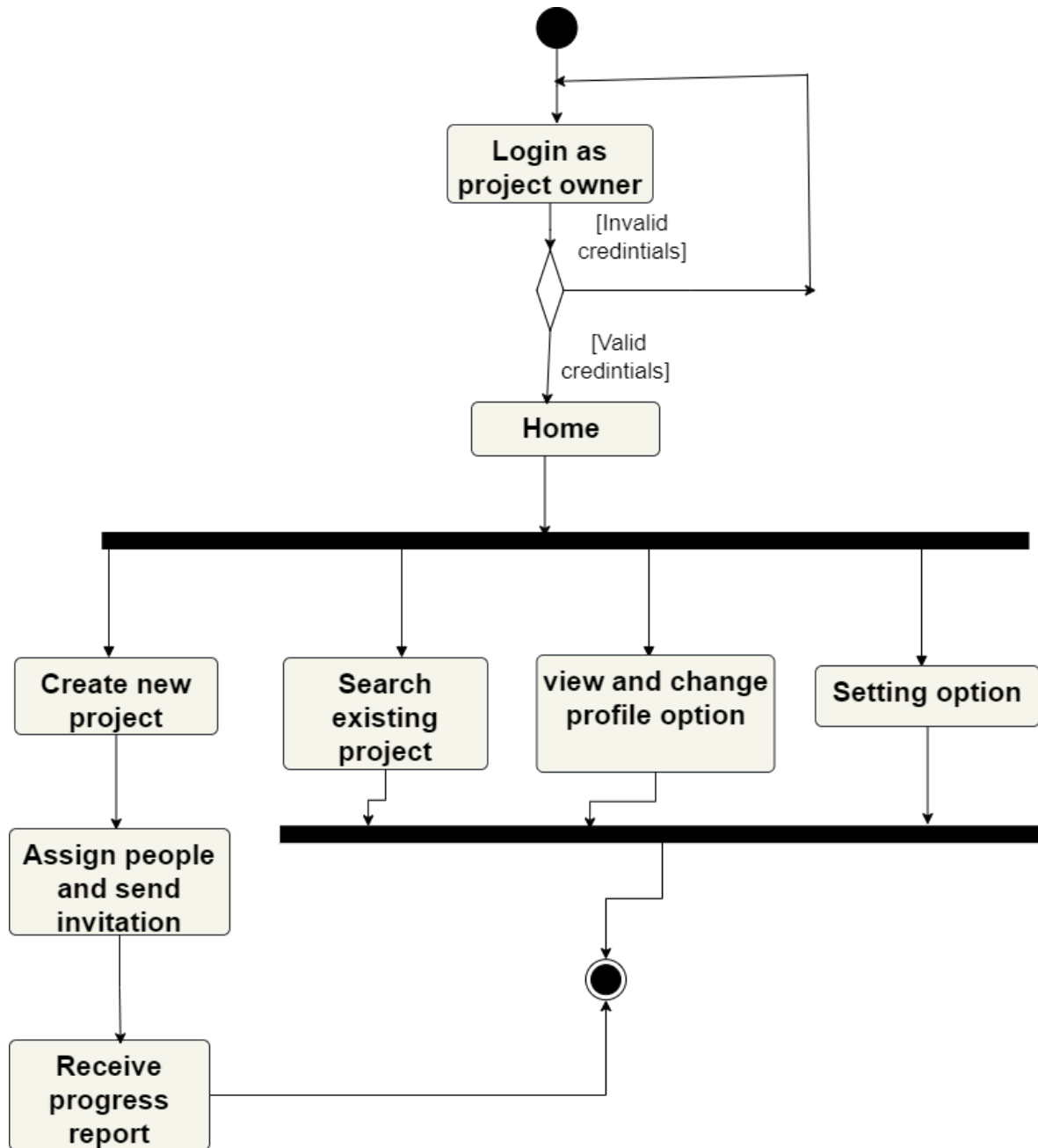


Figure 3.3: Project owner's Activity Diagram

### 3. Admin activities:

Admin can configure system settings, including user roles, permissions, and general system preferences to ensure the system aligns with organizational needs. Admin can create, edit, and manage user accounts within the system. This includes adding new users, updating user information, and managing access permissions.

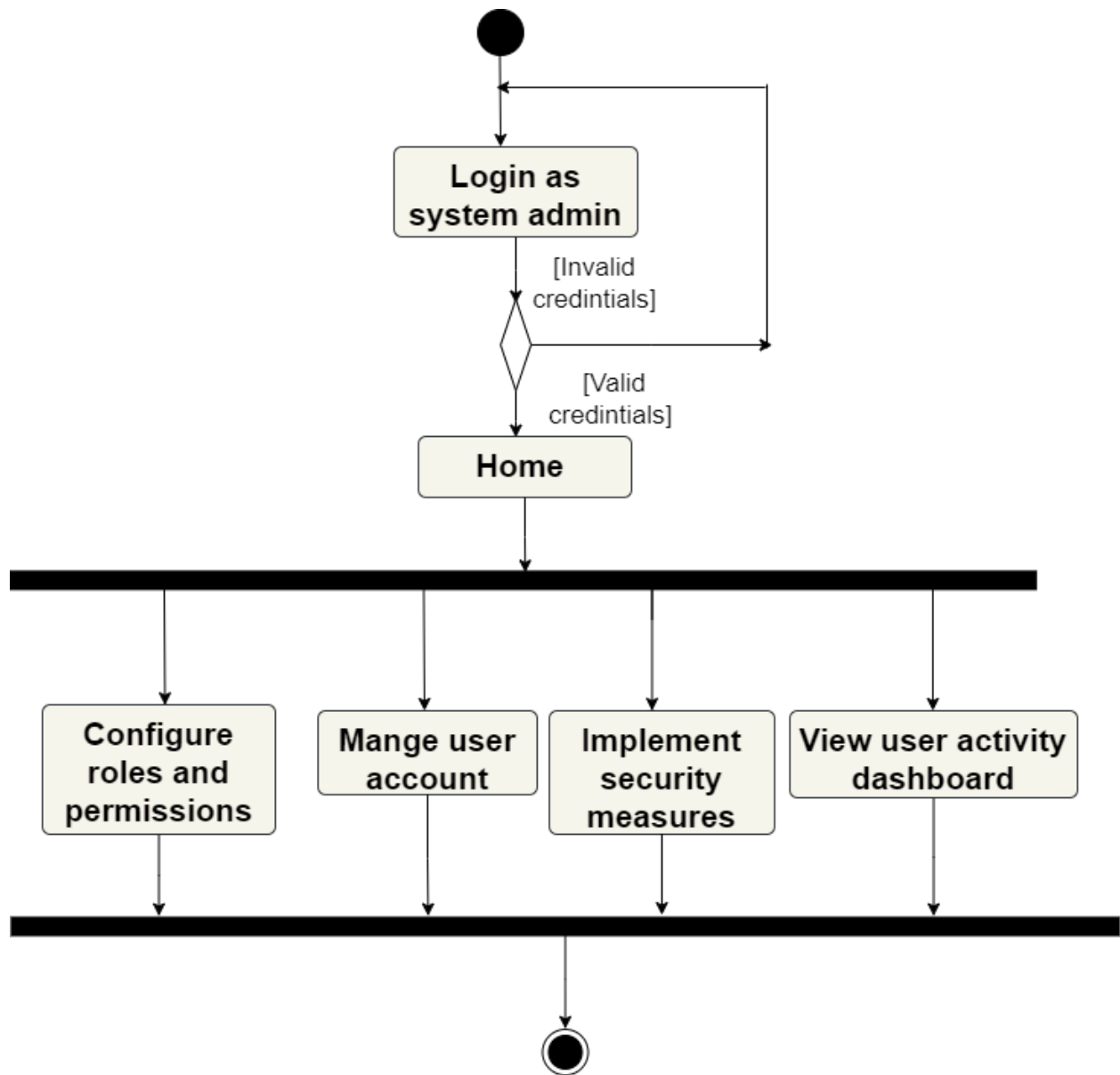


Figure 3.4: Admin's activity diagram

#### 4. Project manager activities

The system should allow project managers to perform the following activities: login, accessing the homepage, assign tasks, search existing projects, get notification, create customizable form, receive progress report, view and change profile options and data analysis.

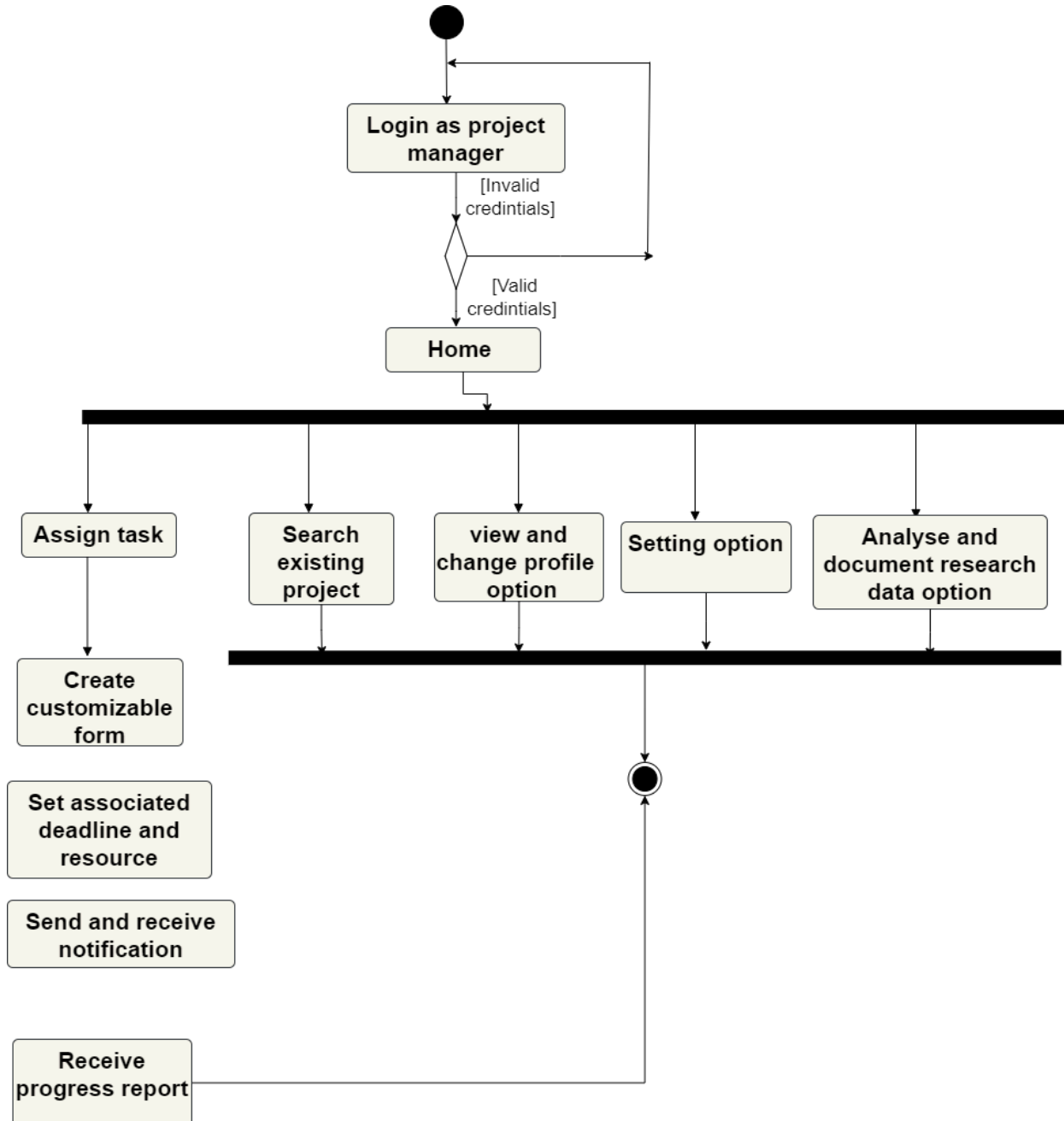


Figure 3.5: project manager activity diagram

## 5. Supervisor activities

There are similar activities performed by the supervisor but they report to different people. Those are: view assigned tasks, view and change profile, receive notification and progress report and verify the correctness of the uploaded data.

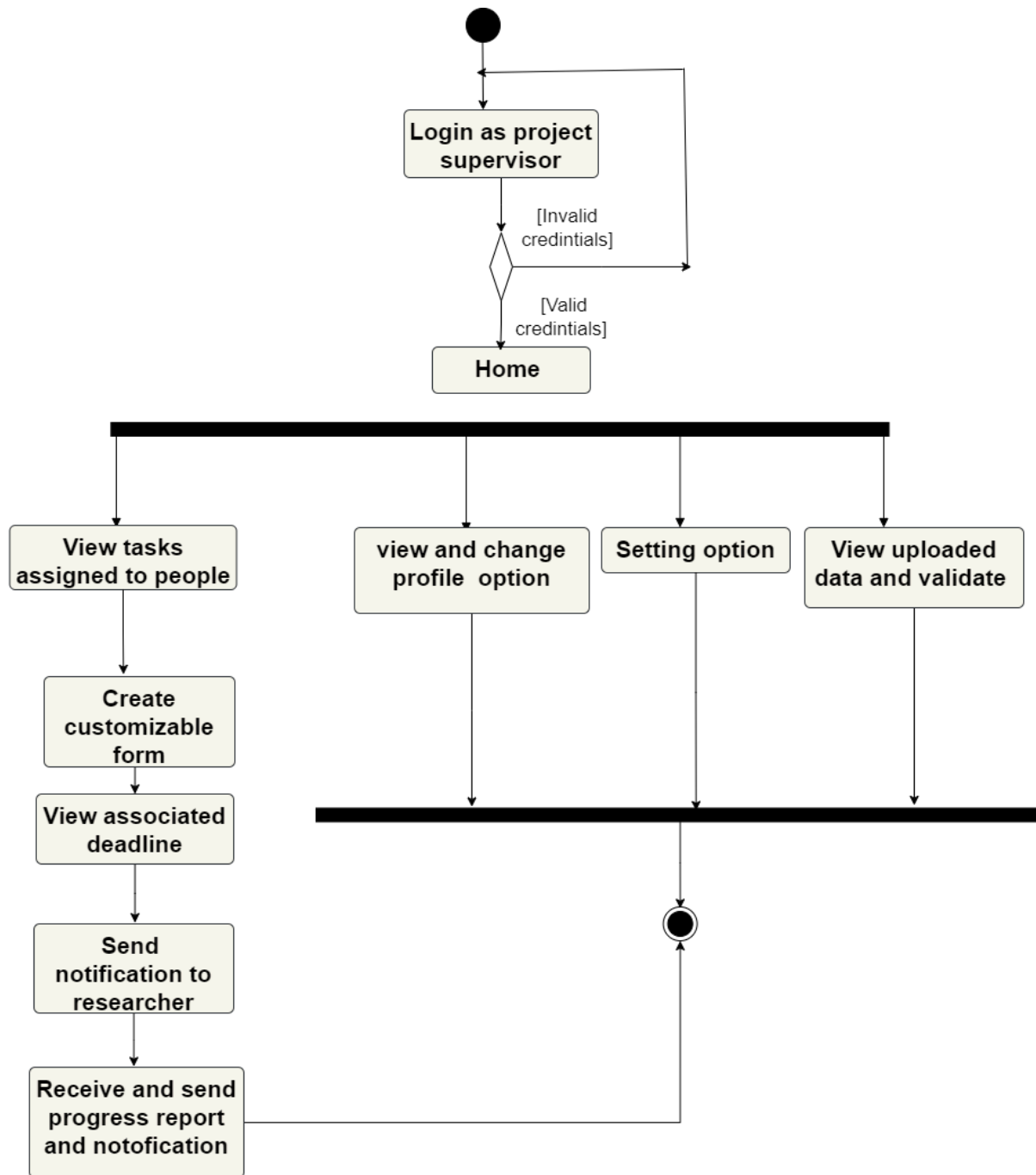


Figure 3.6: supervisor's activity diagram

### 3.3.3 Behavioral/Dynamic Modeling

This section aims to depict the dynamic behavior of the system as it operates under various changing circumstances. The subsequent subsections utilize sequence and state diagrams to illustrate the system's dynamic processes.

#### 3.3.3.1 Sequence diagram

The sequence diagrams in this section provide a practical look into how our Comprehensive Research and Project Management Platform operates, following the principles of clean architecture. These diagrams aim to clarify the sequence of actions and the communication flow between different parts of the system.

The classification ensures that boundaries and entities communicate exclusively with control objects. The API, which represents the server controller class and its processes, takes center stage as the hub for controlled interactions.

#### 1. Project Initiation and Management:

- Focuses on interactions between Project Managers, Researchers, and the system's core components to establish and manage projects.
- Illustrates the flow of actions from user requests through controllers, use cases, and entities to initiate project creation, assign tasks, and manage project details.
- Showcases gateway interactions for accessing external resources or services if applicable.

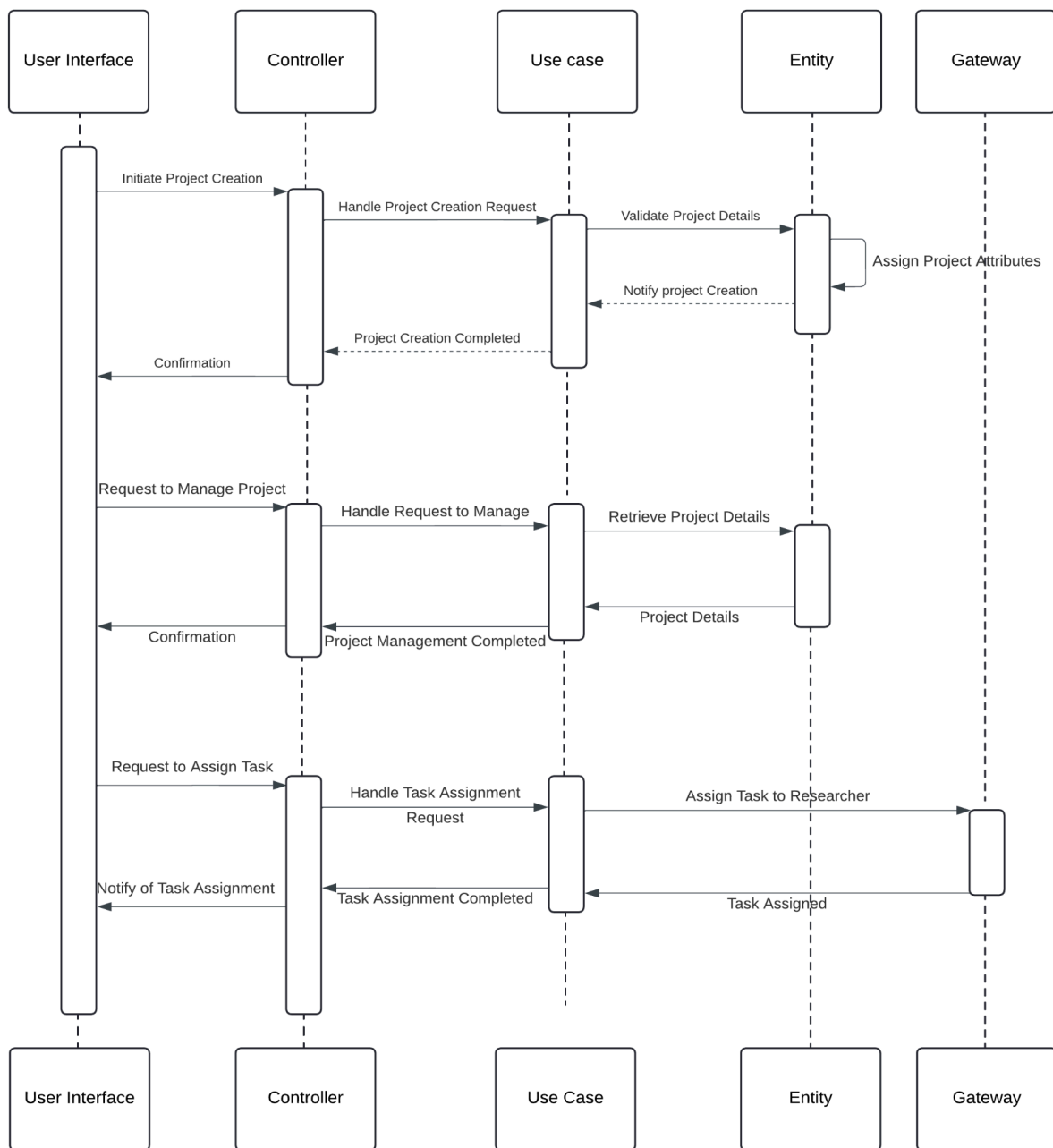


Figure 3.7: Sequence diagram of Project Initiation and Management



## 2. Data Collection, Analysis, and Reporting:

Illustrates Researchers' capabilities in creating customizable forms, generating comprehensive reports and visualizations, and accessing dashboard analytics.

Focuses on interactions between controllers, use cases, and entities to manage form data, perform data analysis, and generate visual representations.

Involves gateways for any external data analysis or visualization libraries or services.

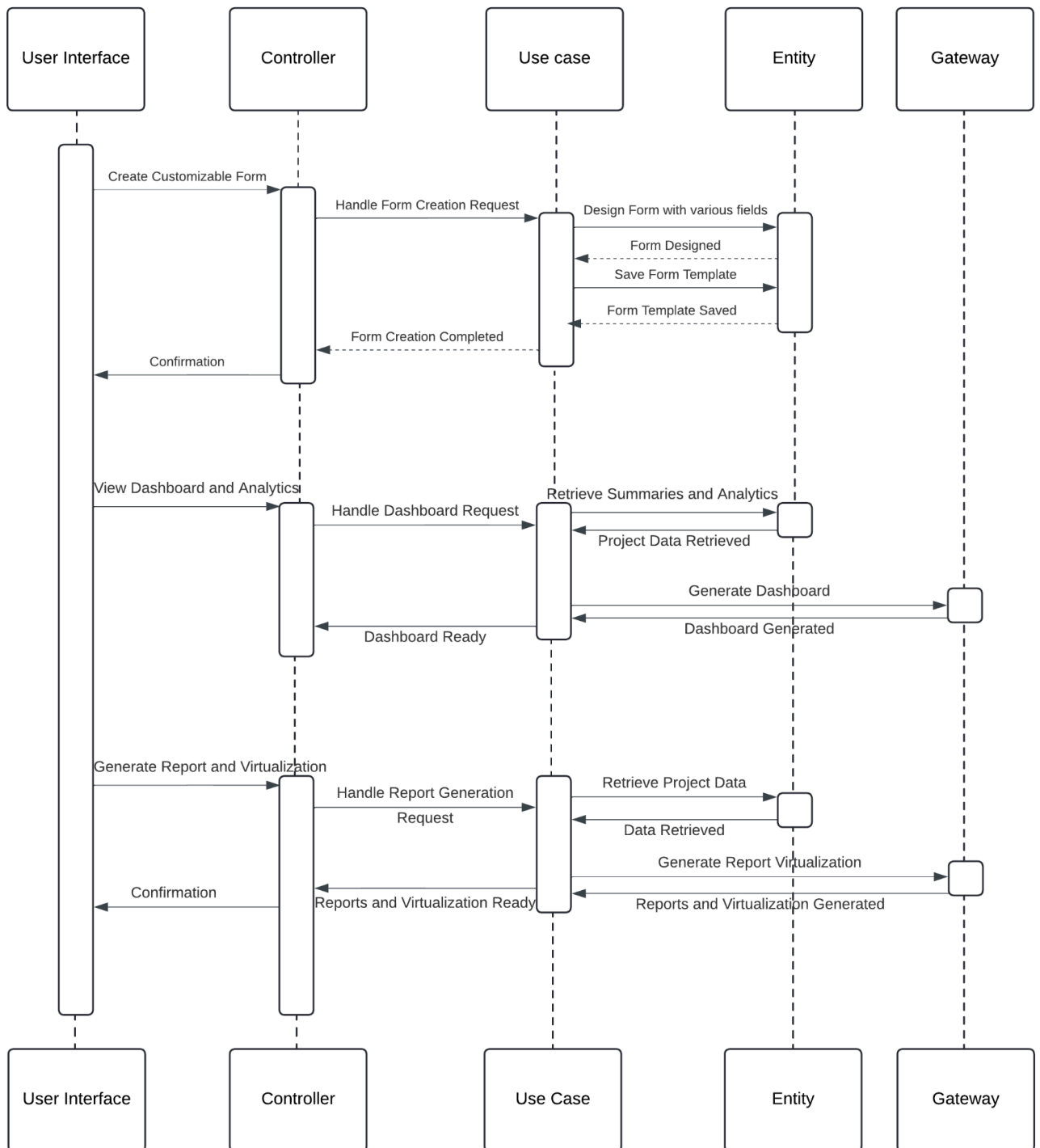


Figure 3.8: Sequence diagram of Data Collection, Analysis and Reporting

## 5. Project Search and Retrieval:

- Showcases the process of Researchers searching for specific projects across the platform.
- Highlights interactions between controllers, use cases, and entities to execute search queries and retrieve relevant project information.
- Incorporates gateways for any external search indexing or retrieval mechanisms.

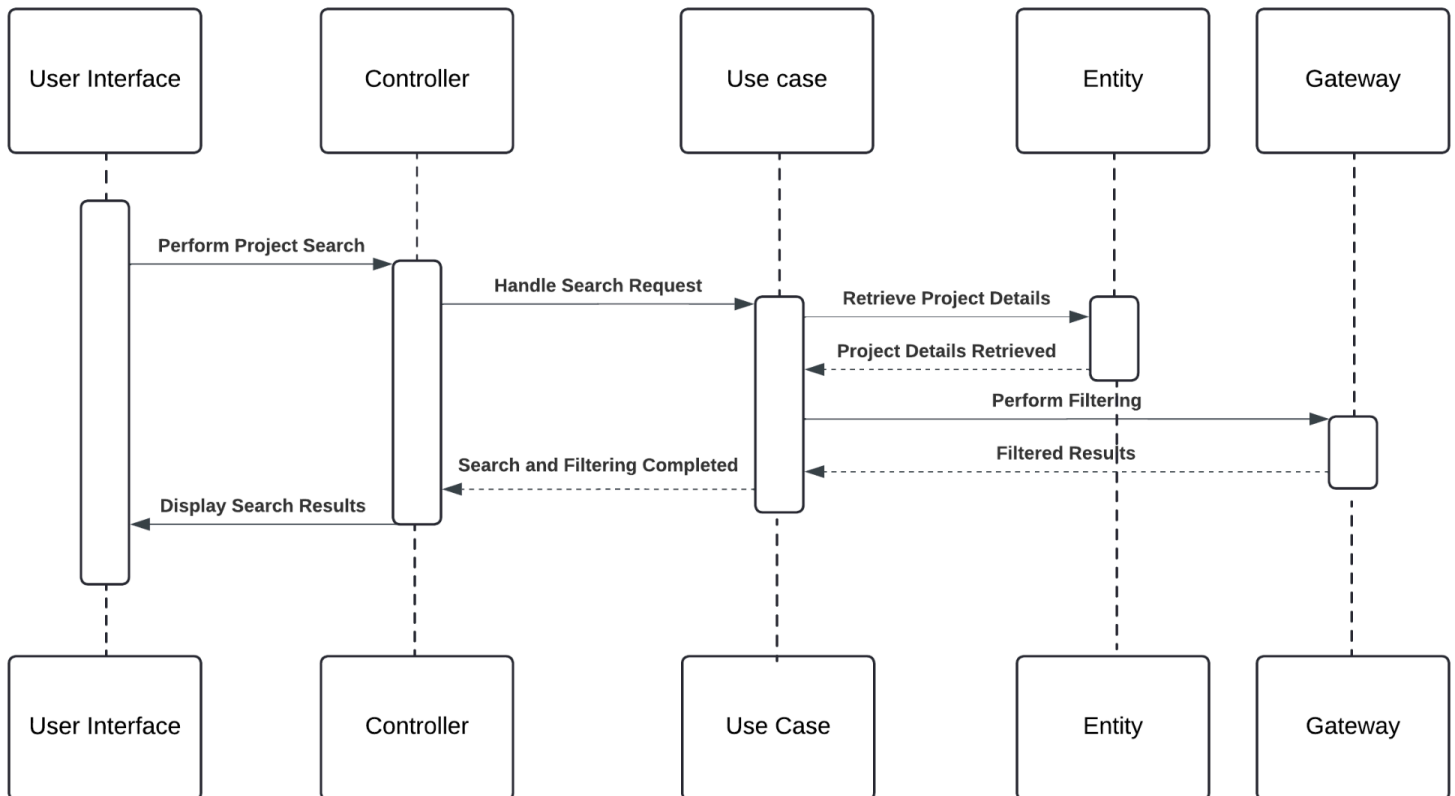


Figure 3.9: Sequence diagram of Project Search and Retrieval

### 3.3.3.2 State Diagram

The state diagrams presented in this section offer a clear snapshot of the dynamic states our Comprehensive Research and Project Management Platform can traverse. Developed within the framework of clean architecture, these diagrams illuminate the various states and transitions that characterize the behavior of our system components.

In the ensuing sections, each state diagram captures a specific aspect of our platform's behavior, contributing to a holistic understanding of its operational dynamics. These visualizations, grounded in clean architecture, aim to express the intricate position of states and transitions that shape the functionality and responsiveness of our research and project management platform.

#### 1. Project Lifecycle:

This diagram would illustrate the various states a project can go through, from its initial creation to completion or termination.

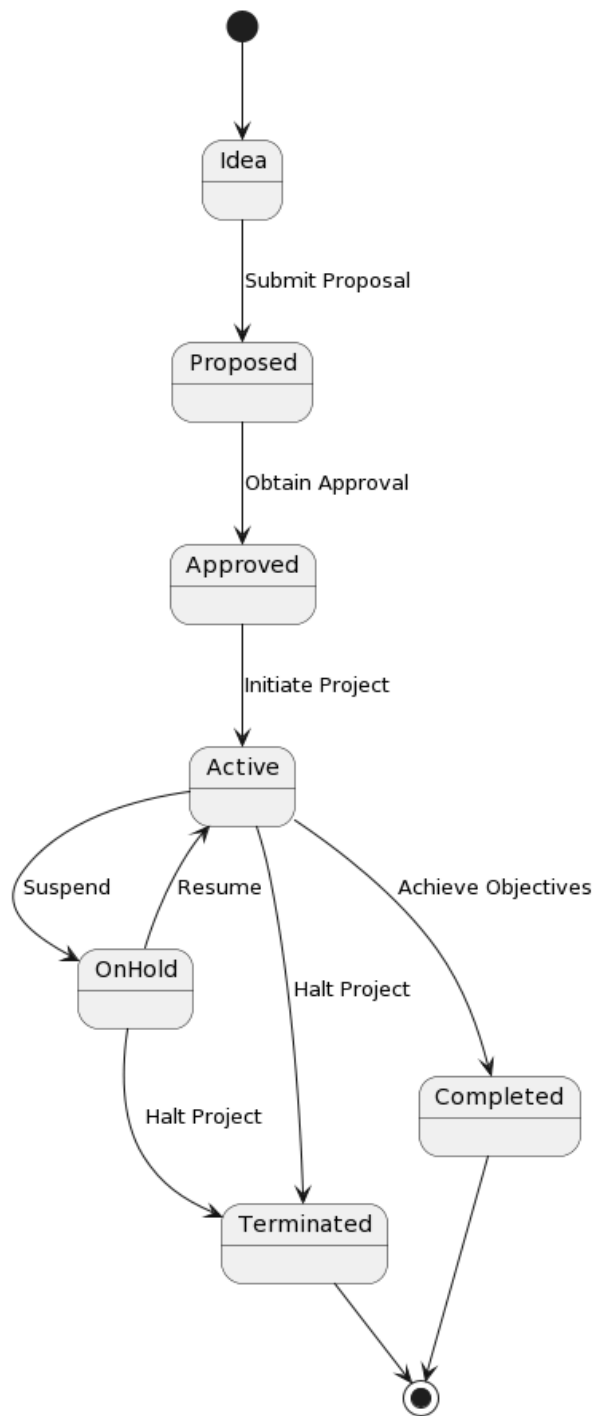


Figure 3.10: State diagram of Project Lifecycle

## 2. User Login and Authorization:

This diagram would depict the process of user login, authentication, and role assignment.

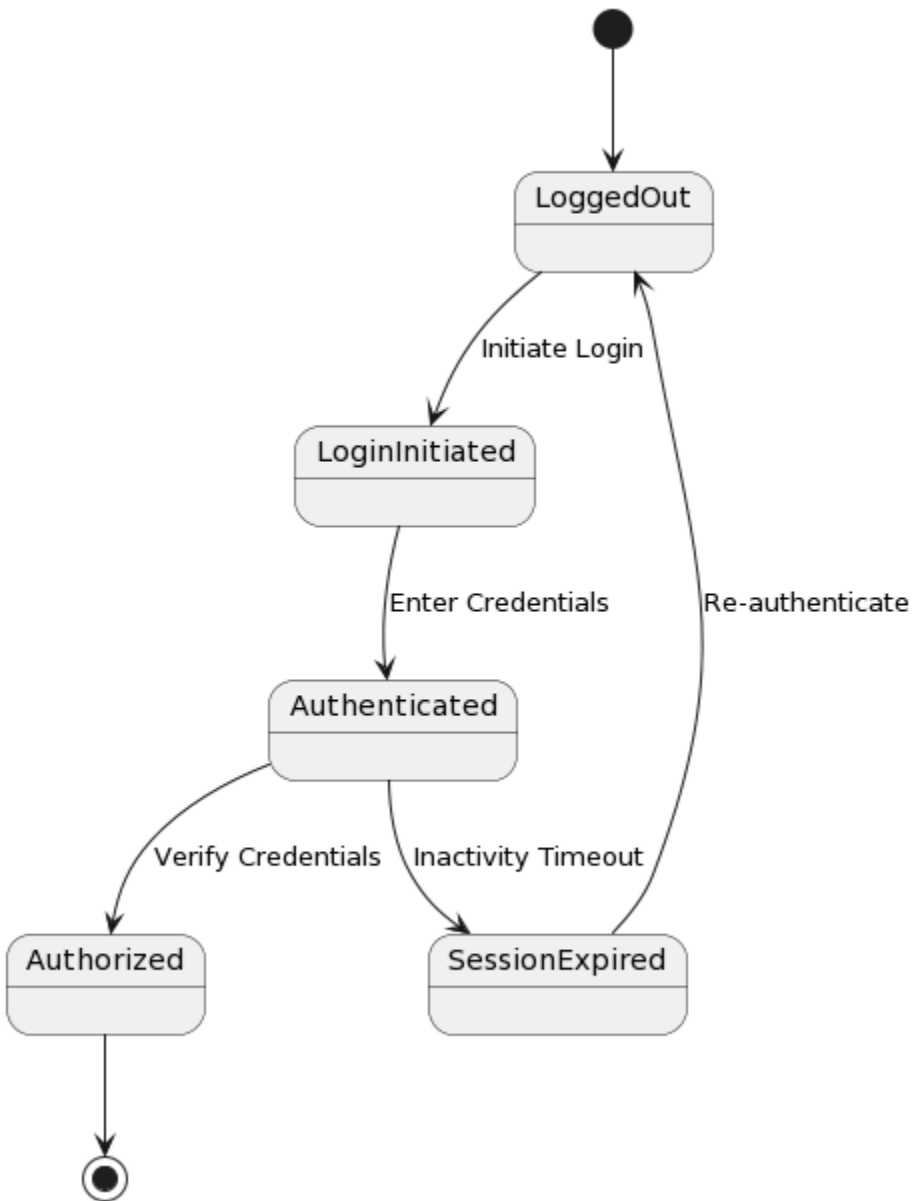


Figure 3.11: State diagram of User Login and Authorization

### 3. Task Management:

This diagram would show the different states a task can be in throughout its lifecycle.

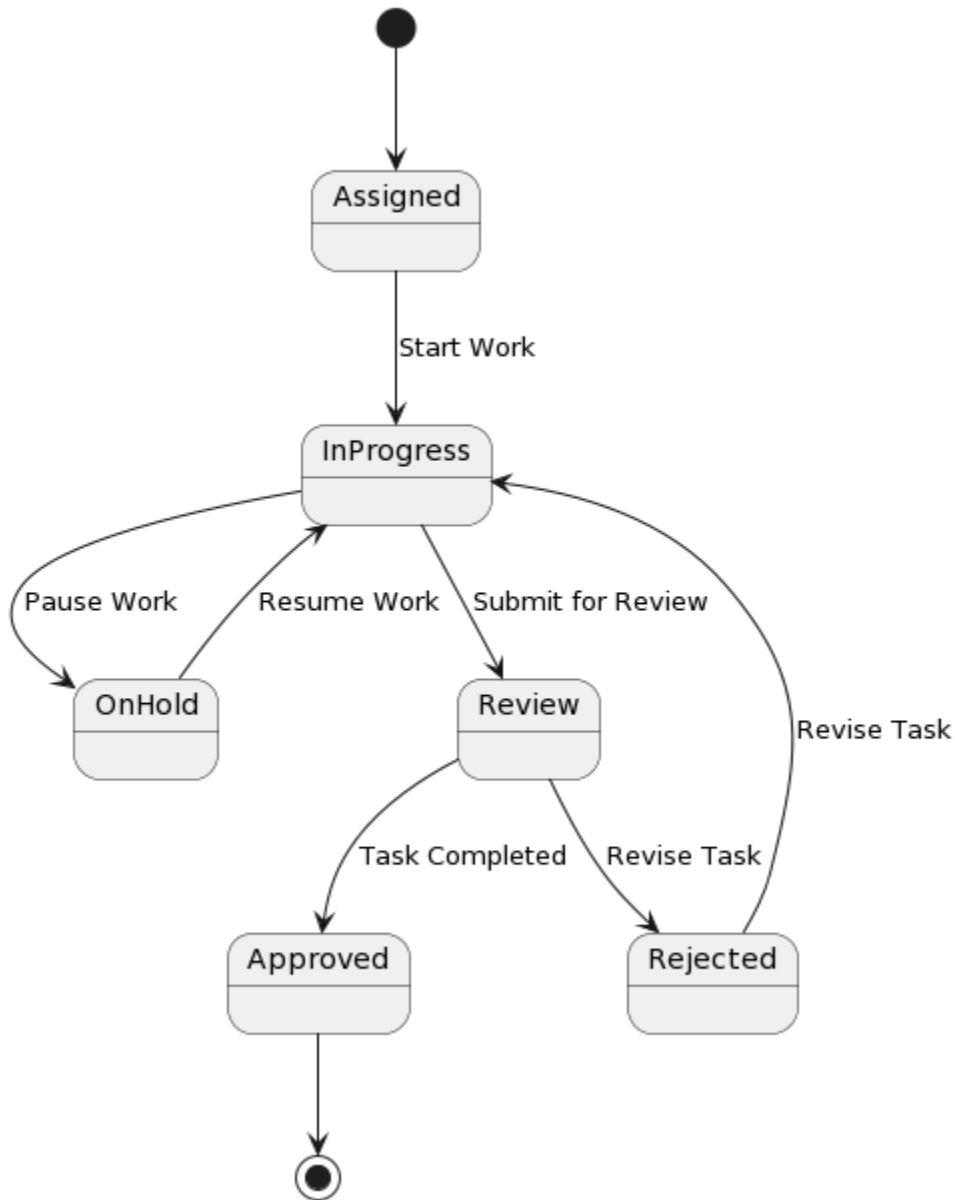


Figure 3.12: State diagram of Task Management

#### 4. Research Data Management:

This diagram would focus on the state changes of research data uploaded to the platform.

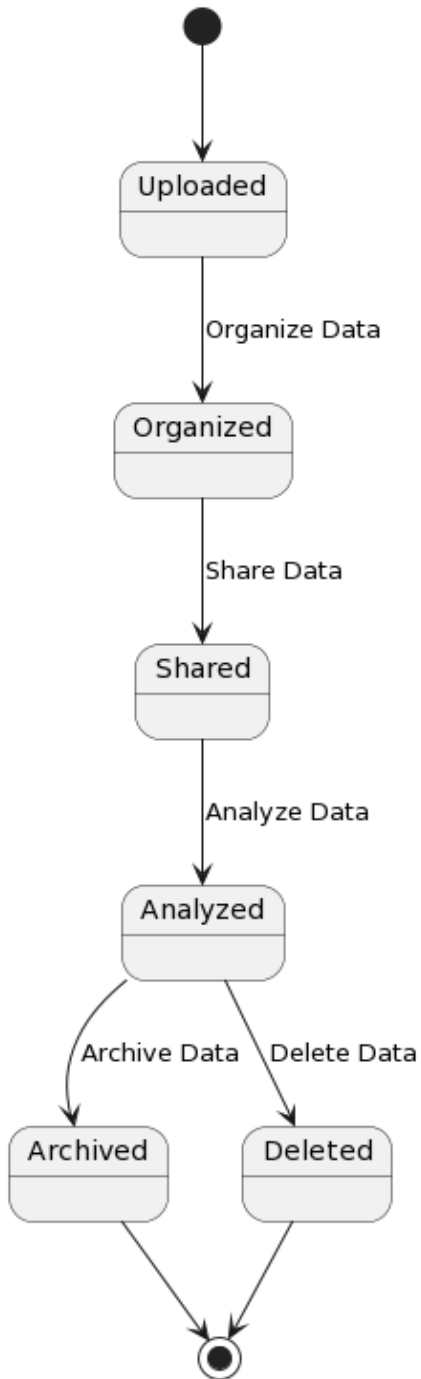


Figure 3.13: State diagram of Research Data Management



## 5. System Maintenance and Updates:

This diagram would illustrate the various states of the platform itself during maintenance and upgrade activities.

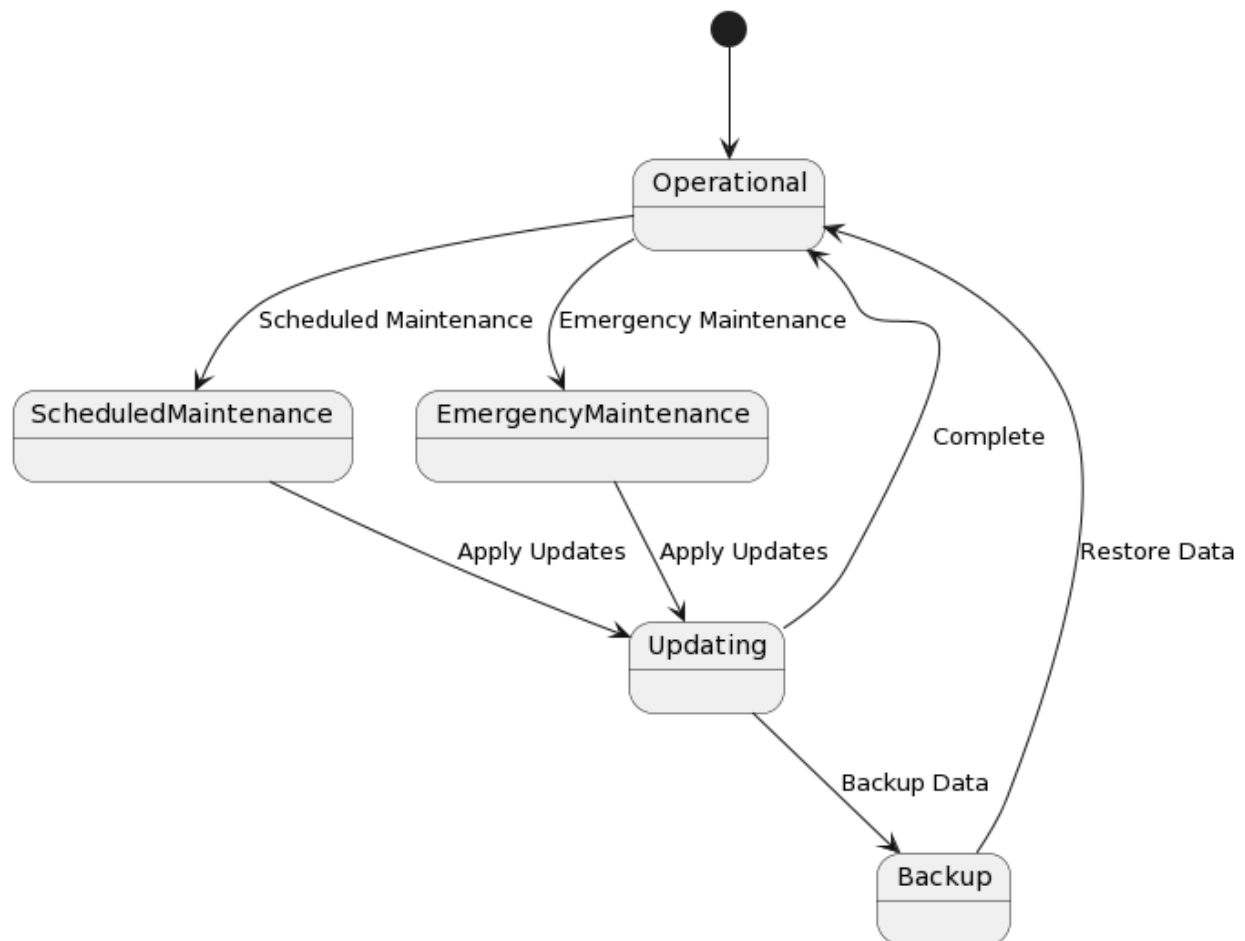


Figure 3.14: State diagram of System Maintenance and Updates

### 3.3.4 Class-Based Modeling

In this section, we delve into the intricacies of Class-Based Modeling, a fundamental aspect of our Comprehensive Research and Project Management Platform. Class-based modeling serves as a structured framework, providing a visual representation of our system's architecture, elucidating the relationships and collaborations among diverse entities.

#### 3.3.4.1 Identifying classes

Following a meticulous analysis of the system, we've pinpointed key classes that form the bedrock of our Comprehensive Research and Project Management Platform. These classes fall into two distinct categories: data classes and control classes.

The data classes embody objects primarily focused on managing their own data, representing tangible entities within the system. On the other hand, control classes, devoid of inherent data, assume the responsibility of orchestrating interactions, working closely with data classes, and directing the flow of operations. This classification lays the groundwork for a comprehensive understanding of the roles each class plays in shaping the functionality and structure of our platform.

Core Classes:

- Project: Represents a research project with attributes like name, description, start/end dates, objectives, milestones, deliverables, budget, resources, and team members.
- Task: Represents a specific activity within a project, with attributes like title, description, due date, priority, status, assignee, and progress.
- User: Represents a system user with attributes like name, email, role (project manager, researcher, administrator), permissions, and profile information.
- ResearchData: Represents research data uploaded to the platform, with attributes like file name, format, size, description, metadata, access permissions, and analysis results.
- Notification: Represents a system notification sent to users, with attributes like message, timestamp, recipient, and priority.

### Additional Classes:

- ProjectTeam: Represents a group of users assigned to a project, with attributes like project ID, member roles, and communication channels.
- Discussion: Represents a threaded conversation among project members, with attributes like topic, messages, participants, and timestamps.
- Document: Represents a file shared within the platform, with attributes like name, type, version history, access permissions, and comments.
- Meeting: Represents a scheduled virtual meeting or conference, with attributes like title, date/time, agenda, participants, and recordings.
- FileRepository: Represents a storage system for research data and documents, with attributes like storage capacity, access controls, and versioning.
- AnalysisTool: Represents a tool or integration for analyzing research data, with attributes like name, functionality, and usage logs.
- SystemSettings: Represents configurable system-wide settings, with attributes like user management, security policies, data retention, and backup schedules.

### Relationships:

- Project has many Tasks.
- Users can create Projects and Tasks.
- User is assigned to Tasks.
- User uploads ResearchData.
- User receives Notifications.
- ProjectTeam has many Users.
- Project has Discussions.
- Project has Documents.
- Project has Meetings.
- ResearchData is stored in FileRepository.
- ResearchData is analyzed by AnalysisTool.

- Administrator manages SystemSettings.

#### 3.3.4.2 Class Diagram

In this section, we present a visual blueprint of our Comprehensive Research and Project Management Platform through class diagrams. These diagrams offer a concise representation of how different classes, their attributes, and relationships come together, providing a snapshot of the system's architecture

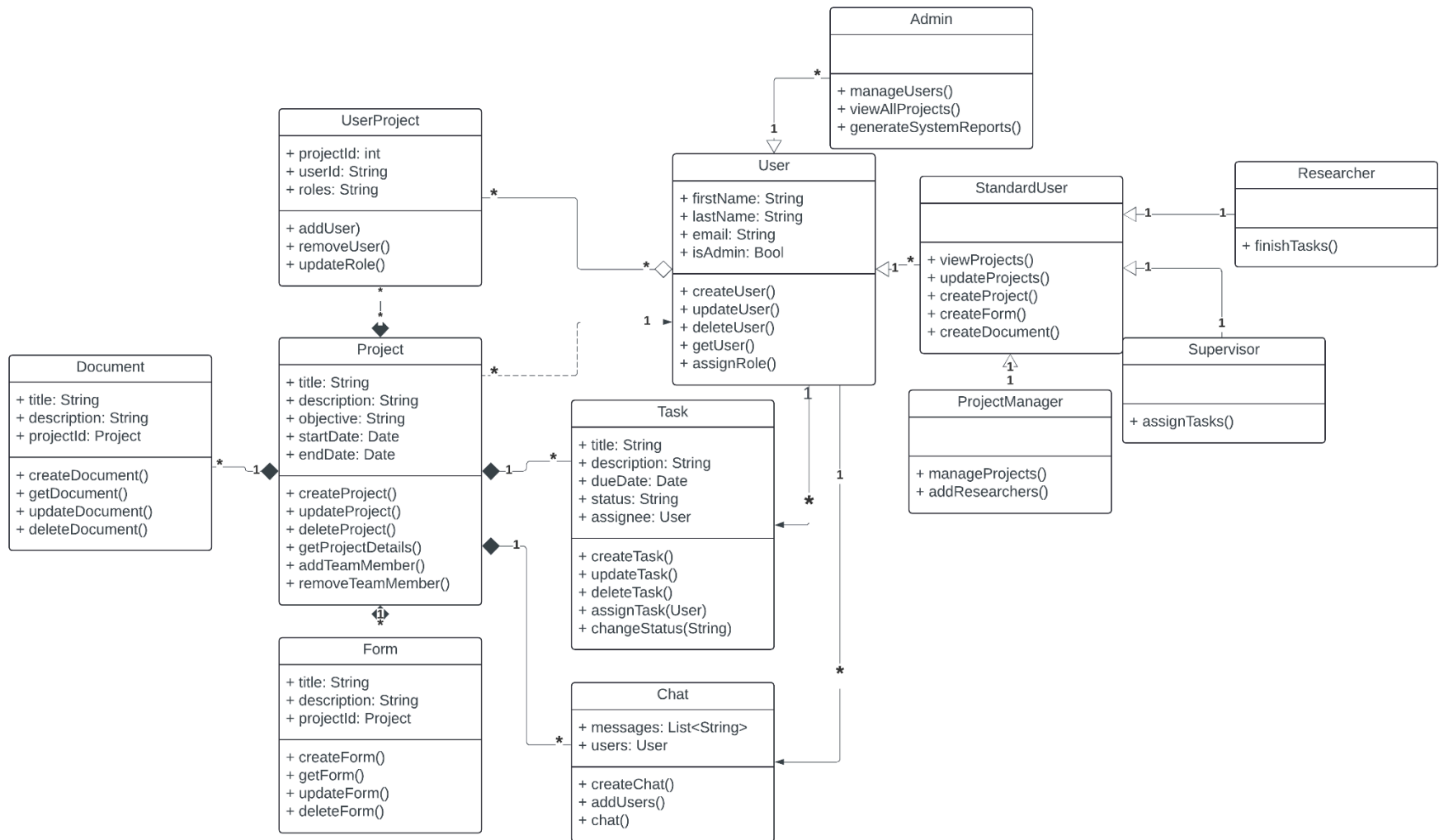


Figure 3.15: Class Diagram

# Chapter 4: System Design

## 4.1. Overview

This section of the documentation focuses on the implementation and deployment aspects of the project outlined in the previous chapters. It starts with an overview of the systems architecture, gradually delving into specific architectural components detailed in the system design section. It also addresses database structure, user interface design and the model design

In terms of user interaction, the system aims to offer a comprehensive project management platform based on user-defined parameters. From a design perspective, modularity is prioritized, the user interface is designed to be intuitive and user-friendly, ensuring ease of use for system users.

## 4.2. Specifying the design goals

### 4.2.1 UI /UX Design goals

The design goals of UX/UI would align closely with the project's objectives:

- 1. Usability:** Ensure that the platform is intuitive and easy to navigate, allowing researchers to efficiently access and manage their projects, documents, and collaboration tools. Streamline workflows and minimize the learning curve for new users.
- 2. Accessibility:** Make the platform accessible to researchers with different abilities, ensuring that all users can participate in research collaborations effectively. Implement features such as keyboard navigation and screen reader compatibility to accommodate users with disabilities.
- 3. Consistency:** Maintain consistency in the layout, terminology, and visual design across all sections of the platform. Use standardized navigation menus, buttons, and icons to create a cohesive user experience and facilitate familiarity for users across different projects.
- 4. Clarity:** Present information and data in a clear and organized manner, enabling researchers to quickly find relevant resources, project updates, and communication threads. Use concise labels, headings, and descriptions to enhance readability and comprehension.
- 5. Engagement:** Incorporate interactive features and visual elements that encourage active participation and collaboration among researchers. Implement features such as real-time chat, task assignment, and progress tracking to keep users engaged and motivated.
- 6. Responsive Design:** Design the platform to be responsive and adaptable to various devices and screen sizes, including desktops, laptops, tablets, and smartphones. Optimize the layout and functionality for each device, ensuring a seamless user experience regardless of the device used.

**7. Feedback:** Provide timely feedback to users on their actions, status updates, and project milestones. Use notifications, alerts, and progress indicators to keep users informed and engaged throughout their research projects.

**8. User-Centric Design:** Prioritize the needs and preferences of researchers when designing and implementing new features and functionalities. Conduct user research, surveys, and usability testing to gather feedback and insights from the target audience, allowing for continuous improvement and refinement of the platform.

By incorporating these specific UX/UI design goals into the development process, the research and project management platform can effectively enhance the productivity, efficiency, and impact of research collaborations, ultimately leading to a more successful and rewarding user experience.

#### 4.2.2 Frontend design goals

The front-end design goals for the research and project management platform focuses on creating an intuitive, user-friendly interface that facilitates efficient navigation, enhances productivity, and promotes collaboration among researchers. Here are some of the front-end design:

**1. Intuitive Navigation:** Design an intuitive navigation system that allows users to easily access different sections of the platform, such as project management, document repositories, collaboration tools, and communication channels. Use clear labels, menus, and breadcrumbs to guide users through the platform.

**2. Streamlined Workflows:** Optimize workflows and task flows to minimize the number of steps required to complete common tasks, such as creating a new project, uploading documents, assigning tasks, and sharing updates. Remove unnecessary friction points and streamline the user journey to enhance efficiency.

**3. Clear Information Hierarchy:** Establish a clear information hierarchy that prioritizes important content and actions, making it easy for users to find and access relevant resources and tools. Use visual cues such as color, typography, and spacing to differentiate between different types of content and actions.

**4. Responsive Design:** Ensure that the platform is responsive and accessible across various devices and screen sizes, including desktops, laptops, tablets, and smartphones. Implement responsive design principles to adapt the layout and functionality of the platform based on the user's device, screen resolution, and orientation.

**5. Consistent Branding:** Maintain consistent branding elements, such as logos, color schemes, typography, and imagery, throughout the platform to reinforce the platform's identity and create a cohesive user experience. Use branding elements strategically to enhance recognition and build trust with users.

**6. Accessibility:** Design the platform with accessibility in mind, ensuring that all users, including those with disabilities, can access and use the platform effectively. Implement accessibility features such as keyboard navigation, screen reader compatibility, and alternative text for images to enhance accessibility.

**7. Interactive Elements:** Incorporate interactive elements such as buttons, forms, dropdowns, and sliders to facilitate user interaction and engagement. Use interactive elements to enable users to perform actions, input data, and manipulate content within the platform seamlessly.

**8. Feedback Mechanisms:** Provide users with feedback on their actions, status updates, and interactions within the platform to keep them informed and engaged. Use feedback mechanisms such as notifications, alerts, progress indicators, and confirmation messages to provide timely feedback and enhance the user experience.

**9. User Personalization:** Allow users to customize their experience by providing options for personalization, such as customizable dashboards, preferences, and settings. Enable users to tailor the platform to their specific needs and preferences to enhance usability and satisfaction.

By incorporating these front-end design goals into the development process, the research and project management platform can deliver a user-friendly and engaging interface that supports the productivity, efficiency, and collaboration of researchers.

#### 4.2.3 Backend Design Goals

**1. API Development:** The backend will prioritize the development of APIs to integrate the platform functionality to frontend clients. These APIs will adhere to RESTful principles, ensuring ease of consumption, scalability, and maintainability.

**2. Analytics and Reporting:** The backend system will be designed to perform comprehensive analytics and reporting functionalities to empower users with valuable insights into their projects and research activities. This includes the development of robust data processing pipelines to collect, clean, and analyze relevant project data. The backend will generate customizable reports based on user-defined parameters. To meet this goal, the backend architecture will incorporate scalable storage solutions for efficient data storage and retrieval. Additionally, data visualization libraries and tools will be integrated to create intuitive charts, graphs, and dashboards, facilitating the interpretation and visualization of research metrics and project progress.

**3. Authentication and Authorization:** The backend system will implement robust authentication and authorization mechanisms to ensure the security and integrity of user data and platform resources. This includes the development of authentication services to securely authenticate users through methods such as username/password authentication. Authorization services will enforce access controls, ensuring that authenticated users have appropriate permissions to access and modify resources based on their roles and privileges.

**4. Data Processing:** The backend system will facilitate the import, export, and transformation of project data, enabling seamless integration with data sources. Data processing pipelines will be



designed to scale horizontally and vertically, leveraging distributed computing techniques and parallel processing to handle increasing data volumes and complex workflows.

#### 4.2.4 Database Design Goals

**1. Data Integrity:** The database will be designed in a way that maintains data integrity to ensure the consistency, accuracy, and reliability of stored information. This includes enforcing constraints such as unique keys, foreign key relationships, and data validation rules to prevent data inconsistencies and anomalies. Referential integrity constraints or foreign key constraints will be implemented to maintain the integrity of relational data relationships, ensuring that all references between related tables remain valid and consistent.

**2. Data Modeling:** The database schema will be designed to accurately represent the domain model of the research and project management platform. This involves identifying and modeling the various entities, attributes, and relationships within the system, capturing the essential data requirements and business rules. Entity-relationship diagrams (ERDs) or similar will be used to visualize and document the database schema.

**3. Scalability:** The database architecture will be designed for scalability to handle increasing data volumes and user loads over time. Horizontal scalability will be achieved by partitions, allowing for parallel processing and improved throughput. Vertical scalability will be supported through techniques such as resource scaling to increase the capacity and performance of individual database instances.

**4. Normalization:** The database schema will be normalized to reduce data redundancy and improve data integrity, ensuring that each piece of information is stored in only one place to avoid update anomalies and inconsistencies. Normalization techniques such as First Normal Form (1NF), Second Normal Form (2NF), and Third Normal Form (3NF) will be applied to decompose data into smaller, atomic units and eliminate redundant attributes. Denormalization may be selectively employed to optimize query performance in read-heavy workloads or to support specific reporting requirements, while maintaining data consistency and integrity.

**5. Data Partitioning:** The database will utilize data partitioning techniques to distribute large tables or datasets into smaller, manageable partitions based on predefined criteria such as date ranges, geographic regions, or user groups. Partitioning schemes such as range partitioning, hash partitioning, or list partitioning will be employed to improve query performance, optimize storage utilization, and facilitate data management and maintenance operations.

## 4.3. Designing the Solution

### 4.3.1 UI Design

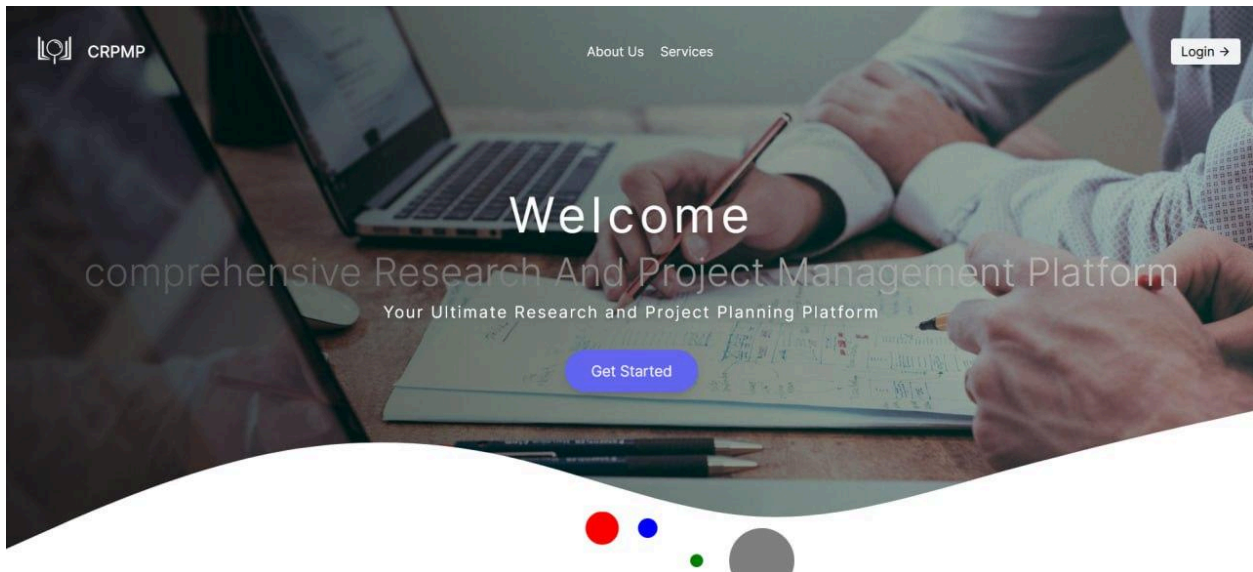


Figure 4.1: Landing Page

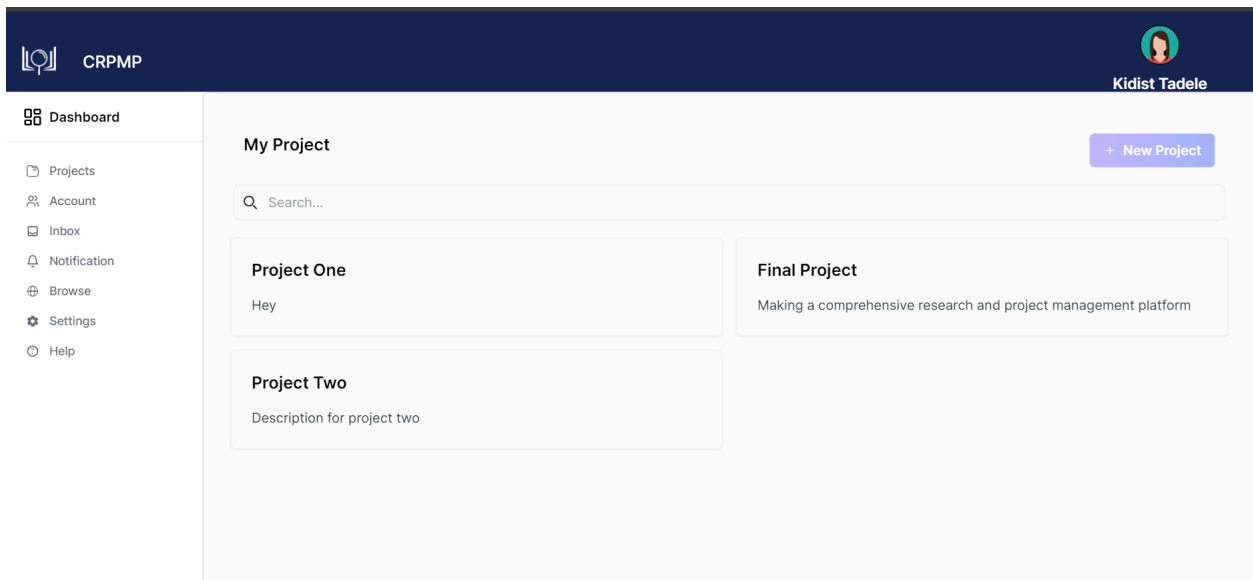


Figure 4.2: My Project Page

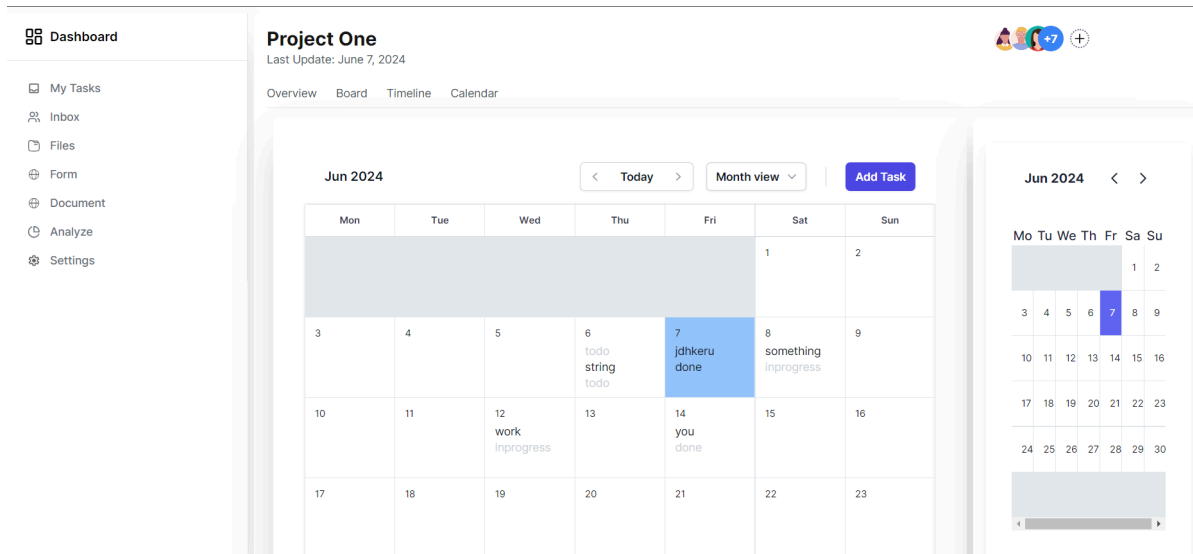


Figure 4.3: Project Dashboard

### Task Assignment

**Task**

**Description**

**Status**

TODO ▾

**Deadline**

**Assignee**

**Add Task**

Figure 4.4: Add Task Page

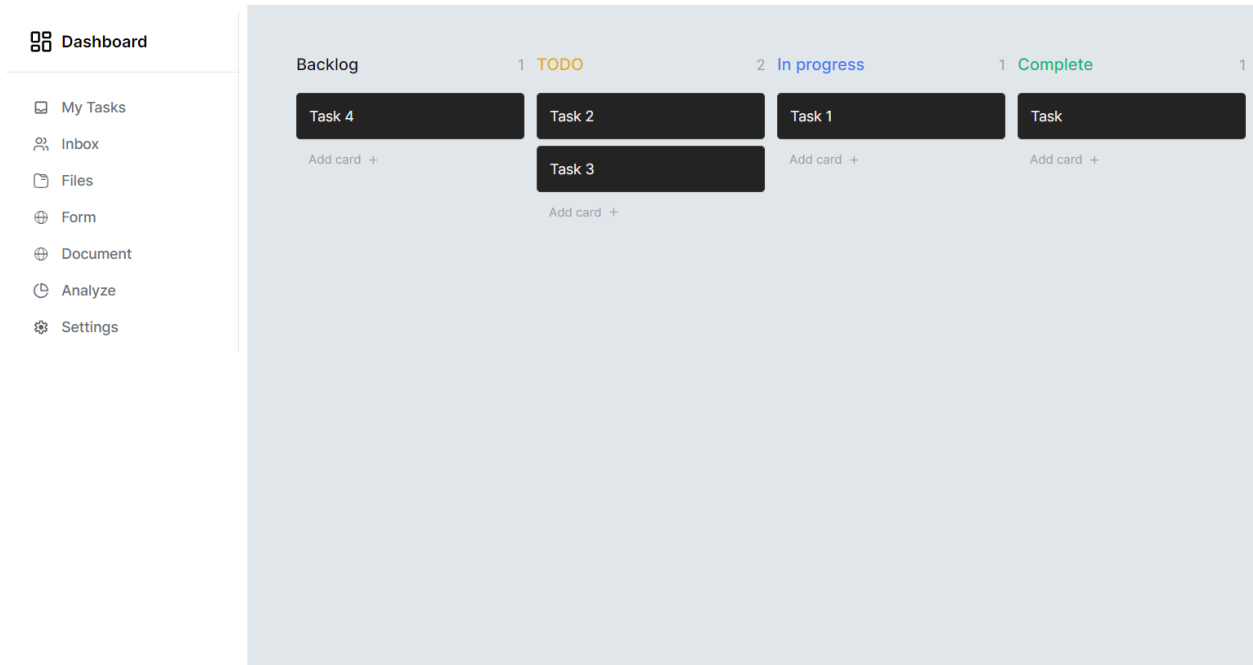


Figure 4.5: My Tasks Page

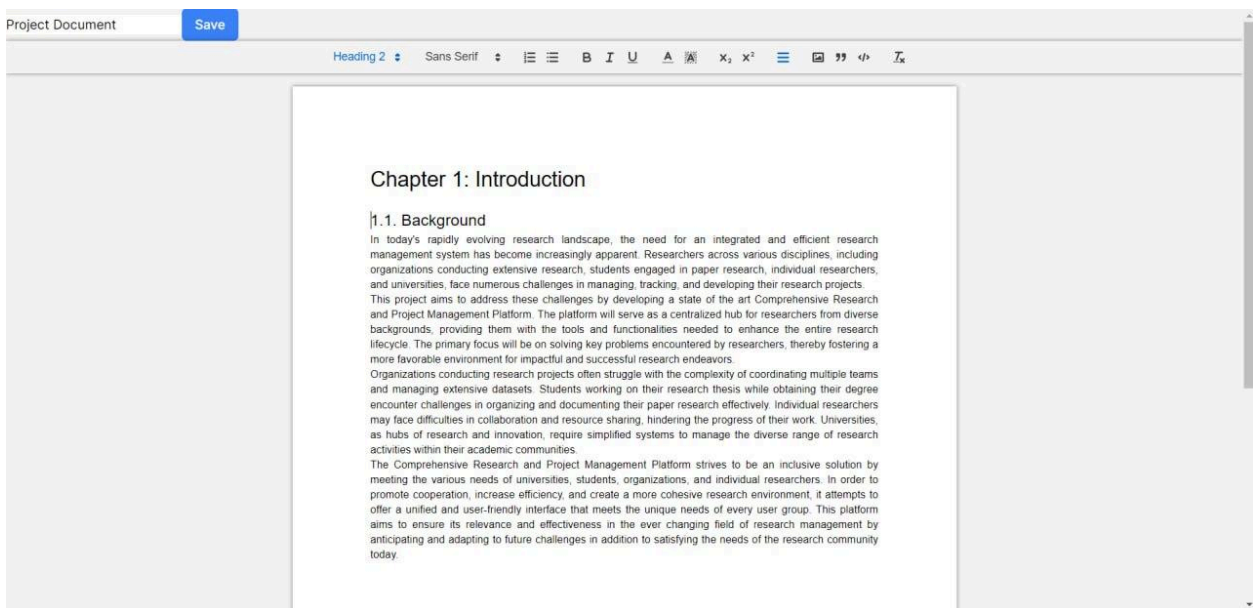
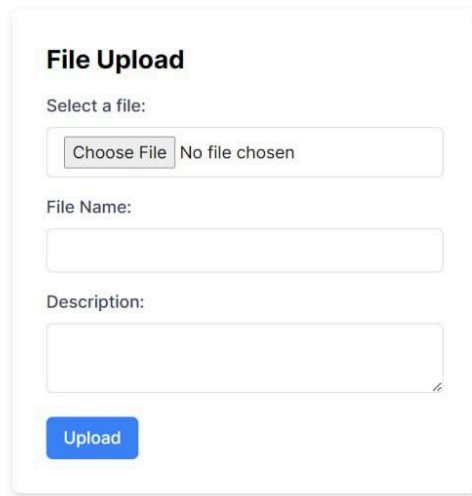


Figure 4.6: Document page



**File Upload**

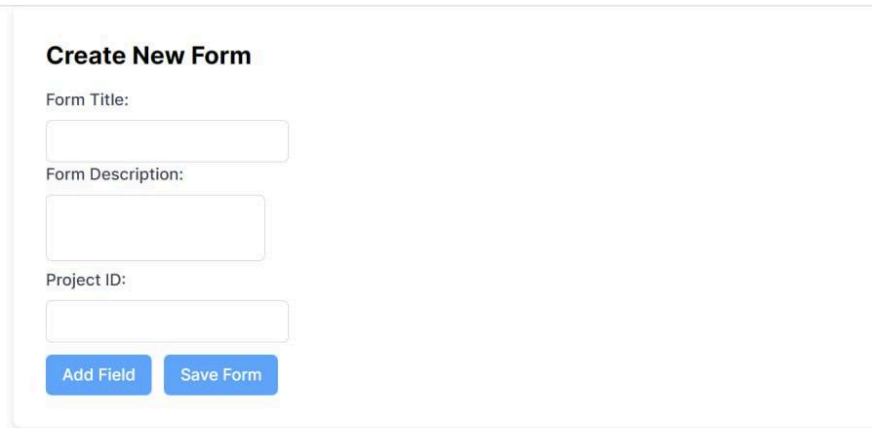
Select a file:

No file chosen

File Name:

Description:

Figure 4.7: File Upload Page



**Create New Form**

Form Title:

Form Description:

Project ID:

Figure 4.8: Form Creation Page

### 4.3.2 Database Design

The database design is structured to efficiently store and manage research project data. It includes tables for projects, users, documents, forms and analytics. Relationships are established between entities to ensure data integrity and enable seamless data retrieval.

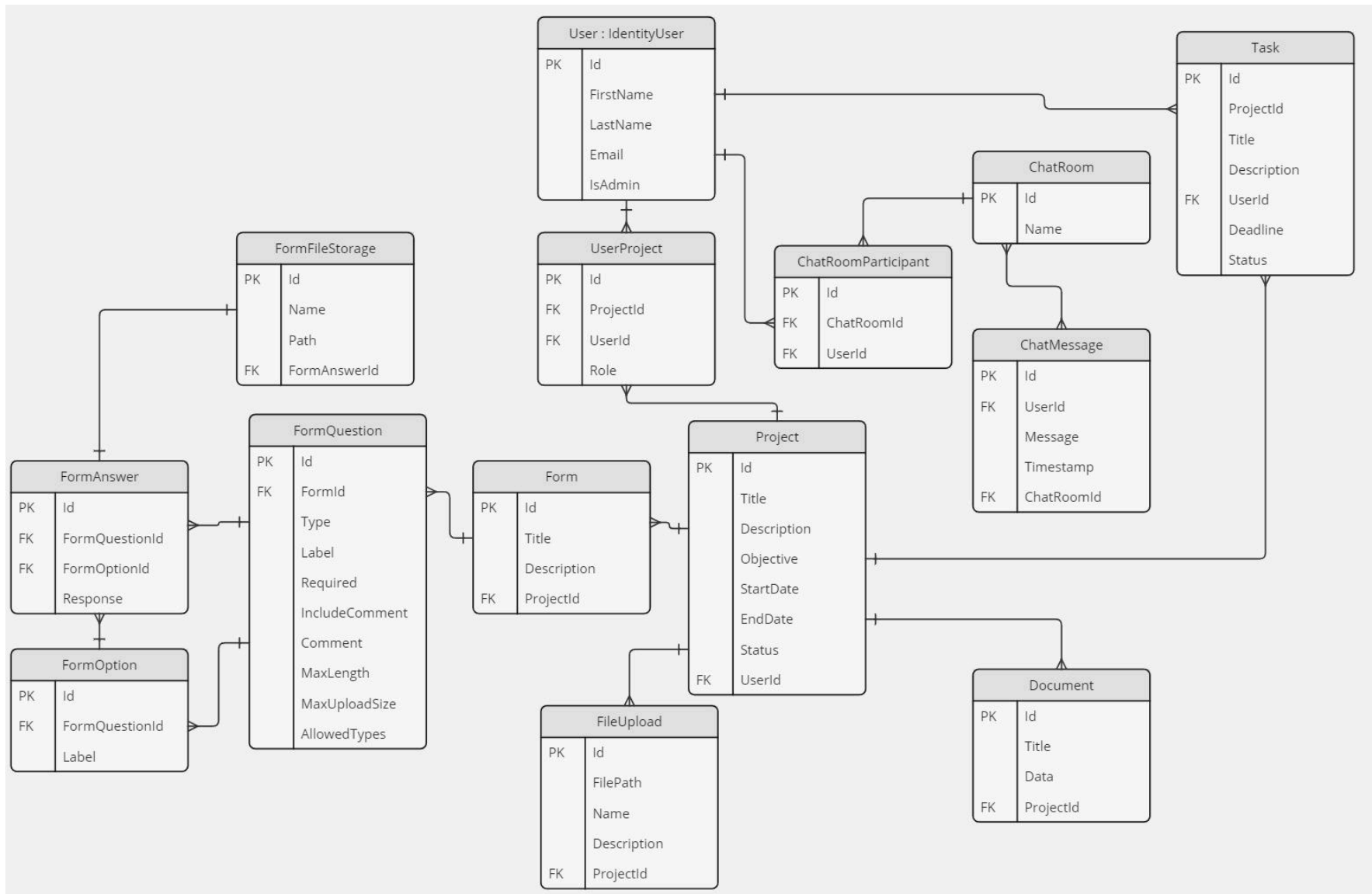


Figure 4.9: ER Diagram

### 4.3.3. Architecture of the System

The architecture of the system is designed to be robust, scalable and secure, adhering to best practices in software engineering. The system will be developed using the principle of Clean Architecture. It comprises several layers:

- **Presentation Layer:** This layer includes the user interface components built using Next.js ensuring a dynamic and interactive user experience. The interface is designed following User-Centered Design principles to ensure ease of use and accessibility.
- **Application Layer:** The application layer consists of the backend services using ASP.NET Core. It handles business logic, data processing, and communication with the database. The backend follows Clean Architecture principles ensuring separation of concerns and maintainability.
- **Data Access Layer:** This layer interfaces with the database system managing data storage and retrieval operations. It utilizes MySQL as the database technology, providing secure scalable storage for research data. The database design is optimized for performance and reliability.
- **Integration Layer:** This layer facilitates integration with external systems and services allowing seamless import and export of research data. It ensures compliance with research data standards and regulations enabling interoperability with other research platforms.

### 4.3.4. System Design

#### **Project Creation and Management**

- Users can create new projects through a user-friendly interface, providing project details such as objectives, timelines and stakeholders.
- Project Managers and team members can be assigned to projects, with role-based access control ensuring appropriate permissions.
- The system facilitates the management of project documentation, resources, and milestones, providing a centralized repository for all project-related information.

#### **Project Tracking and Scheduling**

- Real-time project progress tracking is enabled, with Gantt charts and visual tools providing insights into project schedules and timelines.
- Users can set milestones, deadlines, and dependencies within projects facilitating effective project management



### **Collaboration Tools for Teams in Research**

- Real-time collaboration features such as chat, comments and file sharing are integrated into the platform fostering communication and collaboration among team members.
- Version control ensures document integrity and facilitates collaborative editing.

### **Customizable Data Gathering Form Builder**

- A form building tool allows users to create customizable data gathering forms with various field types and conditional logic
- For templates can be saved and reused across different projects, promoting efficiency and consistency in data collection

### **Dashboard and Analysis**

- Customizable dashboards provide project summaries, progress charts and analytics allowing users to monitor project performance.
- Reports and visualizations can be generated based on project data providing insights for decision-making.

### **Data Storage and Management**

- A secure and scalable storage infrastructure is provided for research data, supporting various data types and offering version control.
- Data import/export functionalities enable seamless data exchange with external sources facilitating data sharing and analysis.

### **Search and Filtering**

- Robust search functionalities allow users to search for specific project details, documents or data.
- Advanced filtering options based on different criteria enhance data retrieval efficiency

### **Notification and Reminders**

- Automated notifications and reminders keep users informed about important events and updates enhancing project coordination and communication

### **Document Generation**

- Users can generate documents, reports or summaries based on project data with support for various formats and customizable templates.

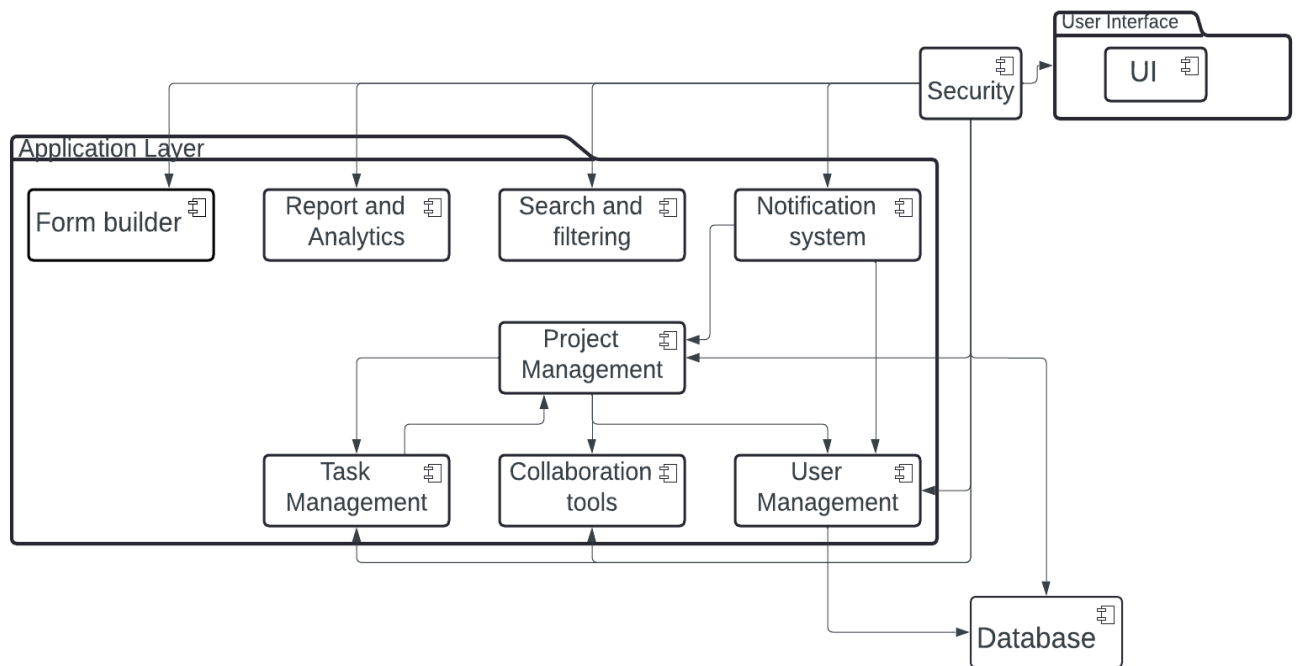


Figure 4.10: Component Diagram

## Component Description

**User Interface (UI):** Handles user interaction through elements like dashboards, forms, and project views. (Presentation Layer)

**User Management:** Handles user login, registration, roles, and permissions. (Part of Application Layer)

**Project Management:** Manages project creation, editing, details, assigning members, and lifecycle tracking. (Application Layer)

**Task Management:** Handles assigning tasks, deadlines, and tracking progress. (Application Layer)

**Collaboration Tools:** Provides functionalities like chat, discussion forums, document sharing, and version control. (Application Layer)

**Reporting & Analytics:** Generates reports, visualizations, and dashboards based on project data. (Application Layer)

**Search & Filtering:** Allows users to search and filter information based on different criteria. (Application Layer)

**Notification System:** Sends automated notifications and reminders to users about important events and updates. (Application Layer)

**Form Builder:** Enables creating customizable data gathering forms with various field types and logic. (Application Layer)

**Integration Layer:** Facilitates communication with external systems and services for data exchange. (Integration Layer)

**Database:** Stores all system data securely (e.g., projects, users, documents, forms, data). (Data Access Layer)

**Security:** Implements security measures for user access, data encryption, and system integrity. (Part of Application Layer)

## **Relationships**

**UI interacts:** with User Management, Project Management, Task Management, Collaboration Tools, Reporting & Analytics, Search & Filtering, Notification System, Form Builder

**User Management interacts:** with Database (for storing user information)

**Project Management interacts:** with User Management (for assigning project members), Task Management, Collaboration Tools, Data Management, Reporting & Analytics

**Task Management interacts:** with Project Management, User Management (for assigning tasks to users)

**Notification System interacts:** with Project Management, User Management (for sending notifications to users)

**Integration Layer interacts:** with External Systems & Services

**Database:** stores data for All other components

**Security:** applies to All components (data encryption, access control)

## **4.4. Verifying the Requirements in the Design**

In this section we will verify the requirements in our design we can follow a format that includes the action, expected outcome, and acceptance criteria.

## **System Admin Requirements:**

**SAR1:** Verify that the admin can successfully log in with the correct username and password.

Expected Outcome: The admin should be granted access to the system upon successful login.

Acceptance Criteria: The admin can enter their username and password, submit the login form, and gain access to the system dashboard.

**SAR2:** Verify that the admin can retrieve the status of the system accessing the dashboard.

Expected Outcome: The admin should be able to view the current status of the system.

Acceptance Criteria: The admin can navigate to the system status section of the dashboard and see real-time information about the system's status.

**SAR3:** Verify that the admin can configure system settings

Expected Outcome: The admin should be able to modify system settings according to organizational needs.

Acceptance Criteria: The admin can access the system settings page, make changes to user roles, permissions, and general preferences, and save the modifications successfully.

**SAR4:** Verify that the admin can create, edit, and manage user accounts within the system.

Expected Outcome: The admin should be able to add, update, and manage user accounts.

Acceptance Criteria: The admin can create new user accounts, edit existing user information, manage access permissions, and perform all necessary user management tasks successfully.

## **User Requirements:**

### **Project manager requirement:**

**PMR1:** Verify that the project manager can log in with the correct username and password.

Expected Outcome: The project manager should be granted access to the system upon successful login.

Acceptance Criteria: The project manager can enter their username and password, submit the login form, and gain access to the home page.

**PMR2:** Verify that the project manager can access the home page after login.

Expected Outcome: The project manager should be directed to the home page upon successful login.

Acceptance Criteria: After logging in, the project manager should see the home page displaying relevant project information, notifications, and available actions.

**PMR3:** Verify that the project manager can create projects, customize forms, assign tasks, receive notifications, and communicate with other users.

Expected Outcome: The project manager should be able to perform project-related actions and interact effectively.

Acceptance Criteria: The project manager can create new projects, customize forms by entering required information, assign tasks to specific individuals, receive relevant notifications, and engage in chat communication with other users.

**PMR4:** Verify that the project manager can add members to projects.

Expected Outcome: The project manager should be able to add members to the project.

Acceptance Criteria: The project manager can enter email addresses of intended group members, search, and successfully add them to the project upon acceptance.

**PMR5:** Verify that the project manager can oversee the entire research project life cycle, including planning, resource allocation, task management, budget control, and progress monitoring.

Expected Outcome: The project manager should have the necessary tools and functionalities to manage the project from start to finish.

Acceptance Criteria: The project manager should be able to create project plans, allocate resources to tasks, manage task assignments and deadlines, monitor project budget, and track overall progress through appropriate tools and interfaces provided by the system.

### **Supervisor Requirements:**

**SR1:** Verify that the supervisor can log in with the correct username and password.

Expected Outcome: The supervisor should be granted access to the system upon successful login.

Acceptance Criteria: The supervisor can enter their respective usernames and passwords, submit the login form, and gain access to the system.

**SR2:** Verify that the supervisor can access the homepage after login.

Expected Outcome: The supervisor should be directed to the homepage upon successful login.

Acceptance Criteria: After logging in, the supervisor should see the homepage displaying relevant information, project progress, notifications, and available options.

**SR3:** Verify that the supervisor has the options to check data correctness, receive notifications and inbox messages, use search and filter options, create customizable data gathering forms, and save/reuse form templates across different projects.

Expected Outcome: The supervisor should have access to the specified options and functionalities.

Acceptance Criteria: After accessing the homepage, the supervisor should be able to receive progress reports, verify data correctness, receive project-related notifications and messages, utilize search and filter options to find specific information, create customizable data gathering forms, and save/reuse form templates across different projects successfully.

### **Researcher Requirements:**

**RR1:** Verify that researchers can log in with the correct username and password.

Expected Outcome: Researchers should be granted access to the system upon successful login.

Acceptance Criteria: Researchers can enter their usernames and passwords, submit the login form, and gain access to the system.

**RR2:** Verify that researchers can access the homepage after login.

Expected Outcome: Researchers should be directed to the homepage upon successful login.

Acceptance Criteria: After logging in, researchers should see the homepage displaying relevant information, project updates, notifications, and available options.

**RR3:** Verify that researchers have the options to upload and manage research data, receive project notifications and reminders for upcoming deadlines, task assignments, or changes in project status, create customizable data gathering forms, and utilize search and filtering capabilities.

Expected Outcome: Researchers should have access to the specified options and functionalities.

Acceptance Criteria: After accessing the homepage, researchers should be able to upload and manage research data, receive project-related notifications and reminders, create customizable data gathering forms, and effectively use search and filtering capabilities to find relevant information.

# Chapter 5: System Implementation

## 5.1. Reviewing the Design Solution

In the review design solution section we will review the proposed design solution to ensure it met the requirements and would be feasible to implement. In this section we will focus on reviewing the high-level architectural design, analyzing the detailed design specifications for each component, and validating the design aligns with the project's functional and non-functional requirements.

### High level architectural review

As described in the design solution section the system will be developed using clean architecture. In a clean architecture there are several layers with separate functionality. So we will review these layers of the architecture to make sure that they align with the system implementation.

1. Presentation Layer:
  - The use of Nextjs for the user interface is a good choice, as it provides a dynamic and interactive experience for the users.
  - Emphasizing User-Centered Design principles is an important consideration to ensure the interface is intuitive and accessible.
2. Application Layer:
  - Adopting the Clean Architecture approach is a solid decision, as it promotes separation of concerns, maintainability, and testability.
  - Implementing the backend using ASP.NET Core is a suitable choice, given its performance, scalability, and rich ecosystem of libraries and tools.
  - Handling business logic, data processing, and communication with the database in this layer is a logical organization.
3. Data Access Layer:
  - Utilizing MySQL as the database technology is a reasonable choice, as it is a widely-used, scalable, and reliable database system.
  - Optimizing the database design for performance and reliability is an essential consideration.
4. Integration Layer:
  - Facilitating integration with external systems and services is crucial for enabling data exchange and interoperability with other research platforms.

- Ensuring compliance with research data standards and regulations is a critical requirement for this type of system.

## **Detailed design specification analysis**

This comprehensive analysis is essential to ensure the proposed solution aligns with the project's functional and non-functional requirements, as well as to identify any potential areas for improvement or optimization. By delving into these technical details, we can validate the feasibility and robustness of the overall system design.

For the purposes of thoroughly analyzing the detailed design solution, the focus will be on the following key aspects:

### **Data Models and Schemas**

Our platform utilizes well-defined data models and schemas to ensure the consistent storage, retrieval, and management of project-related information. The data models capture the key entities, their attributes, and the relationships between them. Some examples of the core data models include:

- Project Model: Stores details about individual projects, such as project name, description, start/end dates, status, assigned team members, and milestones.
- Task Model: Represents the tasks within a project, including task name, description, assignee, due date, dependencies, and progress.
- Document Model: Stores project-related documents, files, and artifacts, including metadata such as file type, version, and associated tags.
- User Model: Stores user information, such as username, email, password (hashed), role, and user preferences.
- Chat Model: Captures the conversations and messages between project team members and stakeholders, including the message content, timestamp, and associated project/task.
- Form Model: Represents custom forms, their fields, and the data submitted by users, enabling the capture of project-related information.
- Milestone Model: Tracks the key milestones for a project, including the milestone name, due date, status, and associated tasks/deliverables.
- Notification Model: Stores information about system-generated notifications, such as task assignments, deadline reminders, and status updates, along with the recipient and delivery status.

### **API Specifications and Integration Points**

The platform's API specifications and integration points are designed to enable seamless connectivity and data exchange with a wide range of project management tools, communication platforms, and external data sources. By exposing well-documented APIs, the system facilitates



the integration of the platform's key functionalities, such as user management, task tracking, milestone monitoring, and notification delivery, within the broader project ecosystem. For this purpose the focus will be on the following aspects:

- User Management APIs: APIs to manage user accounts, including user registration, profile updates, role management, and password resets.
- Chat APIs: APIs to send, receive, and retrieve chat messages, enabling real-time communication and collaboration among project stakeholders.
- Form APIs: APIs to create, update, and submit custom forms, as well as retrieve and analyze the form data.
- Milestone APIs: APIs to manage project milestones, including creating, updating, and tracking milestone progress.
- Notification APIs: APIs to trigger, deliver, and retrieve system notifications, allowing for the seamless communication of important events and updates.
- Project Management Integration: APIs to create, update, and retrieve project-related data, allowing for the synchronization of project information with other PM tools.
- Document Management Integration: APIs to upload, download, and version project documents, enabling the centralized storage and access to project artifacts.
- Reporting and Analytics Integration: APIs to fetch project performance metrics, KPIs, and custom reports, supporting data-driven decision-making.
- Notification and Collaboration Integration: APIs to trigger notifications, task assignments, and enable real-time collaboration among project stakeholders.

## **Algorithm Implementations and Logic Flows**

- Project Scheduling and Resource Allocation: Algorithms to optimize task scheduling, resource utilization, and project timelines, considering dependencies, constraints, and user preferences.
- Collaborative Workflow Management: Logic flows to facilitate the seamless handoff of tasks, approvals, and feedback among project team members and stakeholders.
- Reporting and Visualization: Algorithms to generate comprehensive reports, dashboards, and visualizations that provide insights into project status, resource utilization, and overall performance.
- User Authentication and Authorization: Algorithms to authenticate users, manage user roles and permissions, and enforce access controls based on the user's role and project involvement.
- Chat Moderation and Filtering: Algorithms to moderate chat conversations, detect inappropriate content, and apply filtering mechanisms to ensure a professional and productive communication environment.

- **Form Validation and Processing:** Algorithms to validate form submissions, perform data transformations, and integrate the form data with other project-related information.
- **Milestone Tracking and Notifications:** Algorithms to track milestone progress, detect potential delays, and trigger appropriate notifications to keep project stakeholders informed.
- **Notification Prioritization and Delivery:** Algorithms to prioritize notifications based on importance, user preferences, and delivery channels (email, in-app, push notifications) to ensure timely and effective communication.

## **Error Handling and Exception Management**

- **Centralized Error Handling:** The platform follows a centralized approach to error handling, with a well-defined hierarchy of custom error types and corresponding HTTP status codes.
- **Graceful Degradation:** In the event of unexpected errors or exceptions, the system gracefully degrades the functionality, providing clear and informative error messages to users and logging the issues for further investigation.
- **Transactional Integrity:** The platform ensures transactional integrity by implementing atomic operations and rollback mechanisms, preventing partial updates and maintaining data consistency.

## **Security, Logging, and Monitoring Mechanisms**

- **User Identity and Access Management:** secure authentication, authorization, and access control mechanisms to protect user accounts and sensitive information.
- **Milestone and Notification Monitoring:** monitoring and alerting mechanisms to track milestone progress, notification delivery, and any anomalies or security incidents related to these functionalities.
- **Authentication and Authorization:** our platform employs a secure authentication system, supporting single sign-on (SSO), multi-factor authentication, and role-based access control to ensure only authorized users can access the system.
- **Assessments:** the platform undergoes periodic penetration testing and vulnerability assessments to identify and address potential security weaknesses, staying ahead of the evolving threat landscape.

## **Validating the design**

The detailed design solution for our platform has been carefully validated to ensure alignment with the project's key functional and non-functional requirements. The analysis confirms that the

proposed data models and schemas adequately capture all the necessary information related to users, projects, tasks, milestones, and communications, enabling the platform to effectively manage the complete project lifecycle. Furthermore, the API specifications and integration points provide the required functionality for user management, project tracking, team collaboration, and report generation, seamlessly connecting the platform with the broader project management ecosystem.

Regarding non-functional requirements, the design incorporates robust error handling and exception management mechanisms to maintain a reliable user experience, while the security, logging, and monitoring capabilities safeguard sensitive data and user information. The scalability and performance characteristics of the design have also been validated, ensuring the platform can handle increasing user loads and data volumes without compromising responsiveness and availability.

## 5.2. Development Tools

The tools and processes used in our comprehensive research and project management system. Our tech stack includes a combination of development, collaboration, version control, and deployment tools that facilitate a streamlined and efficient workflow.

### Development Environment

- **Tool:** Visual Studio Code (VS Code)
- **Purpose:** VS Code is our primary Integrated Development Environment (IDE). It supports a wide range of extensions that enhance productivity, including syntax highlighting, code linting, debugging, and integrated terminal access.
- **Key Extensions:** C# for .NET development
  - Prettier for code formatting
  - ESLint for JavaScript and TypeScript linting

### Version Control

- **Tool:** Git
- **Purpose:** Git is used for tracking changes in the source code during development. It allows multiple developers to work collaboratively without overwriting each other's work.
- **Repository Hosting:** GitHub
  - **Purpose:** GitHub hosts our Git repositories and provides additional features such as pull requests, issue tracking, and project management boards.

## **Deployment**

- Tool: Vercel
- Purpose: Vercel is used for deploying and hosting our Next.js applications. It provides continuous deployment, custom domains, and performance optimization.

## **Performance Monitoring**

- Tool: GTmetrix
- Purpose: GTmetrix is used to analyze the performance of our web applications. It provides detailed reports on load times, page size, and optimization opportunities.

## **Package Management**

- Tool: NuGet
- Purpose: NuGet manages packages and dependencies for .NET projects. It simplifies the process of incorporating third-party libraries and tools into our development environment.

## **Collaboration and Documentation**

- Tool: Notion
- Purpose: Notion serves as our primary tool for project management and collaboration. It is used for creating project plans, tracking progress, and storing important project documentation.
- Tool: Google Docs
  - Purpose: Google Docs is used for collaborative document editing, enabling real-time collaboration and sharing of research documents, meeting notes, and other textual content.

## **Database Management**

- Tool: SQL Server Management Studio (SSMS)
- Purpose: SQL Server Management Studio (SSMS) is used for designing, developing, and administering Microsoft SQL Server databases. It provides a visual interface for managing database schemas, running queries, and monitoring database performance.

## **Frontend Framework**

- Tool: Next.js
- Purpose: Next.js is used for building server-side rendered React applications. It enhances performance and SEO, and supports static site generation.

## Programming Languages

- Language: TypeScript
  - Purpose: TypeScript is used for developing our frontend applications. It provides static typing, which helps in catching errors early and improving code quality.
- Language: C# (.NET)
  - Purpose: C# is used for backend development in the .NET framework. It is a powerful, versatile language that supports object-oriented programming and a wide range of libraries and frameworks.

## Backend Framework

- Framework: .NET
  - Purpose: .NET is used for building scalable and high-performance backend services. It supports a variety of applications, including web, mobile, desktop, and cloud-based applications.

Our comprehensive research and project management system is built on a robust and modern tech stack that ensures efficiency, collaboration, and high performance. By leveraging the strengths of these tools and frameworks, we can effectively manage our projects from inception to deployment.

## 5.3. Developing the Solution

The solution is developed using ASP.NET Core Web API for the backend and Next.js for the frontend. This section outlines the detailed steps and coding practices employed during the development process, including key components, sample code snippets, and integration between the frontend and backend.

### 5.3.1 Backend Development with ASP.NET Core Web API

#### 5.3.1.1 Project Setup

1. Creating the ASP.NET Core Web API Project:
  - Using Visual Studio to create a new ASP.NET Core Web API project.
  - Select .NET 8.0 as the framework.
  - Configure the project name and solution name.
2. Project Structure:
  - ``Controllers/``: Contains API controllers.
  - ``Models/``: Contains data models.
  - ``Interface/``: Contains interfaces for the repositories.

- `Repositories/`: Contains data access logic.
- `Data/`: Contains database context and migration files.
- `Dto/`: Contains .
- `Helper/`: Contains .
- `Hubs/`: Contains .
- `Resources/`: Contains .

### 5.3.1.2 Defining Models

```
public class User : IdentityUser
{
    public string FirstName { get; set; }
    public string LastName { get; set; }
    public string Email { get; set; }
    public bool IsAdmin { get; set; }
    // Other properties...
}

public class Project
{
    public int Id { get; set; }
    public string Title { get; set; }
    public string Description { get; set; }
    public virtual ICollection<Task>? Tasks { get; set; }
    // Other properties...
}
```

Additional models: Task, Document, Form, Chat, etc.

### 5.3.1.3 Configuring the Database Context

```
public class DataContext : DbContext
{
    public DataContext(DbContextOptions<ApplicationDbContext> options) : base(options)
    {
    }

    public DbSet<User> Users { get; set; }
    public DbSet<Project> Projects { get; set; }
    public DbSet<Task> Tasks { get; set; }
    // Other DbSet properties...
}
```

```

protected override void OnModelCreating(ModelBuilder modelBuilder)
{
    base.OnModelCreating(modelBuilder);
    // Additional configuration...
}
}

```

### 5.3.1.4 Implementing Controllers

```

[Route("api/[controller]")]
[ApiController]
public class ProjectsController : ControllerBase
{
    private readonly ApplicationDbContext _context;

    public ProjectsController(ApplicationDbContext context)
    {
        _context = context;
    }

    [HttpGet]
    public async Task<ActionResult<IEnumerable<Project>>> GetProjects()
    {
        return await _context.Projects.ToListAsync();
    }

    [HttpPost]
    public async Task<ActionResult<Project>> CreateProject(Project project)
    {
        _context.Projects.Add(project);
        await _context.SaveChangesAsync();
        return CreatedAtAction(nameof(GetProject), new { id = project.Id }, project);
    }

    // Other actions: GetProject, UpdateProject, DeleteProject, etc.
}

```

### 5.3.1.5 Business Logic and Services

```

public interface IProjectRepository
{
    Task<IEnumerable<Project>> GetAllProjectsAsync();
    Task<Project> GetProjectByIdAsync(int id);
    Task<Project> CreateProjectAsync(Project project);
    Task UpdateProjectAsync(Project project);
    Task DeleteProjectAsync(int id);
}

public class ProjectRepository : IProjectRepository
{
    private readonly ApplicationDbContext _context;

    public ProjectService(ApplicationDbContext context)
    {
        _context = context;
    }

    public async Task<IEnumerable<Project>> GetAllProjectsAsync()
    {
        return await _context.Projects.ToListAsync();
    }

    public async Task<Project> GetProjectByIdAsync(int id)
    {
        return await _context.Projects.FindAsync(id);
    }

    public async Task<Project> CreateProjectAsync(Project project)
    {
        _context.Projects.Add(project);
        await _context.SaveChangesAsync();
        return project;
    }

    public async Task UpdateProjectAsync(Project project)
    {
        _context.Entry(project).State = EntityState.Modified;
        await _context.SaveChangesAsync();
    }
}

```



```

public async Task DeleteProjectAsync(int id)
{
    var project = await _context.Projects.FindAsync(id);
    if (project != null)
    {
        _context.Projects.Remove(project);
        await _context.SaveChangesAsync();
    }
}
}

```

## 5.3.2 Frontend Development with Next.js

### 5.3.2.1 Project Setup

#### 1. Creating the Next.js Project:

Using Visual Studio Code to create a new Next.js project.

- Install Node.js:
  - Ensure that Node.js is installed on your machine. If not, download and install it from [Node.js official website](https://nodejs.org/en/).
- Open Visual Studio Code:
  - Launch Visual Studio Code on your machine.
- Open a New Terminal:
  - Open a new terminal in Visual Studio Code by selecting *Terminal > New Terminal*.
- Install Next.js and Create a New Project:
  - Run the following commands in the terminal to install *create-next-app* and set up a new Next.js project: `npx create-next-app@latest`
  - Follow the prompts to configure your project. You will be asked to provide the project name and some optional configurations.
- Configure the Project Name and Solution Name.
- Navigate to the Project Directory.
- Start the Development Server:
  - Start the Next.js development server to verify that your project was created successfully: `npm run dev`
  - Open your browser and navigate to <http://localhost:3000> to see your new Next.js application in action.

## 2. Project Structure:

- `pages/`: Contains the application's pages.
- `components/`: Contains reusable UI components.
- `services/`: Contains API service calls.
- `styles/`: Contains CSS files.

### 5.3.3 Key Features and Components

#### 1. User Authentication and Authorization:

- Implement user registration and login functionalities.
- Secure routes based on user roles (Admin, StandardUser, ProjectManager, Supervisor, Researcher).

```
const handleSubmit = async (event: React.FormEvent<HTMLFormElement>) => {
  event.preventDefault();
  const response = await fetch(`${apiBaseUrl}/api/Account/register`, {
    method: 'POST',
    headers: {
      'Content-Type': 'application/json',
    },
    body: JSON.stringify({
      firstName,
      lastName,
      email,
      password,
      confirmPassword,
    }),
  });
};
```

#### 2. Project Management:

- CRUD operations for projects, tasks, documents, and forms.
- Assign tasks to users and track progress.

```
useEffect(() => {
  const fetchCards = async () => {
    try {
      const res = await fetch(`${apiBaseUrl}/api/Task/ProjectTasks/${projectId}`);
      const data: Task[] = await res.json();
      console.log(data)
      const mappedCards: Card[] = data.map(task => ({
```

```

        title: task.title,
        id: task.id.toString(),
        column: mapStatusToColumn(task.status)
    ));

    setCards(mappedCards);
  } catch (error) {
    console.error("Error fetching tasks:", error);
  }
};

```

### 3. Chat:

- Implement chat feature using model and controller.

```

const fetchChatRooms = async () => {
  const response = await fetch(`${apiBaseUrl}/api/ChatRoom/user/${userId}/chatrooms`);
  const data: ChatRoom[] = await response.json();
  setChatRooms(data);
  console.log(data)
  console.log(chatRooms);
};

```

```

const joinChatRoom = async (roomId: number) => {
  setCurrentRoom(roomId);
  fetchMessages(roomId);
};

```

```

const fetchMessages = async (roomId: number) => {
  const response = await fetch(
    `${apiBaseUrl}/api/ChatRoom/chatroom/${roomId}/messages`
  );
  const data: ChatMessage[] = await response.json();
  setMessages(data);
  console.log(data)
  console.log(messages)
};

```

```

const sendMessage = async () => {
  try {

```

```

const response = await fetch(
  `${apiBaseUrl}/api/ChatRoom/chatroom/${currentRoom}/messages/create`,
  {
    method: "POST",
    headers: {
      "Content-Type": "application/json",
    },
    body: JSON.stringify(msgToSend),
  }
);

if (response.ok) {
  console.log("Item created successfully");
  // Optionally, you can redirect the user or update the UI here
} else {
  console.error("Failed to create item:", response.statusText);
}
} catch (error) {
  console.error("Error creating item:", error);
}

};

```

#### 4. Document Management:

- Upload, store, and manage documents associated with projects.

```

const params = useParams();
const id = params.id;
const [editor, setEditor] = useState<Quill | null>(null);
const [title, setTitle] = useState<string>("");
const apiBaseUrl = process.env.NEXT_PUBLIC_API_BASE_URL;

useEffect(() => {
  const fetchDocument = async () => {
    try {
      const response = await fetch(`${apiBaseUrl}/api/Document/SingleDocument/${id}`);
      if (!response.ok) {
        throw new Error('Failed to fetch document');
      }
    }
  }
}

```

```

    const data = await response.json();
    setTitle(data.title);
    if (editor) {
      editor.setContents(JSON.parse(data.data));
    }
  } catch (error) {
    console.error('Failed to fetch document', error);
  }
};

if (editor && id) {
  fetchDocument();
}
}, [editor, id]);

const wrapperRef = useCallback((wrapper: HTMLDivElement | null) => {
  if (wrapper === null) return;
  wrapper.innerHTML = "";

  const editorInstance = document.createElement('div');
  wrapper.append(editorInstance);

  const quill = new Quill(editorInstance, { theme: 'snow', modules: { toolbar:
TOOLBAR_OPTIONS } });
  setEditor(quill);
}, []);

const handleSave = async () => {
  try {
    if (!id || !editor) return;
    const response = await fetch(`${apiBaseUrl}/api/Document/UpdateDocument/${id}`, {
      method: 'PUT',
      headers: {
        'Content-Type': 'application/json',
      },
      body: JSON.stringify({
        id,
        title,
        data: JSON.stringify(editor.getContents())
      }),
    });
  });
};

```

```

    if (!response.ok) {
      throw new Error('Failed to save document');
    }

    alert('Document saved!');
  } catch (error) {
    console.error('Failed to save document', error);
  }
};

```

## 5. Form Handling:

- Create and manage custom forms for project data collection.

```

const handleFieldChange = (
  index: number,
  name: keyof FormField,
  value: any,
  optionIndex?: number | boolean
) => {
  const updatedFields = [...formFields];
  if (name === "options" && typeof optionIndex === "number") {
    updatedFields[index].options[optionIndex] = value as string;
  } else if (name === "allowedTypes" && typeof optionIndex === "boolean") {
    const fileType = value as string;
    const updatedTypes = [...(formFields[index].allowedTypes || [])];
    if (optionIndex) {
      if (!updatedTypes.includes(fileType)) {
        updatedTypes.push(fileType);
      }
    } else {
      const indexToRemove = updatedTypes.indexOf(fileType);
      if (indexToRemove !== -1) {
        updatedTypes.splice(indexToRemove, 1);
      }
    }
    updatedFields[index].allowedTypes = updatedTypes;
  } else {
    updatedFields[index][name] = value;
  }
}

```

```

    setFormFields(updatedFields);
  };

  const handleAddOption = (fieldIndex: number) => {
    const updatedFields = [...formFields];
    updatedFields[fieldIndex].options.push(optionInput);
    setFormFields(updatedFields);
    setOptionInput("");
  };

```

## 6. Dashboard and Reporting:

- Provide a dashboard for admin to view analytics about users, projects and other metrics.

```

const [currentPage, setCurrentPage] = useState('dashboard')
const [isOpen, setIsOpen] = useState(false);

const PageContent = ({ currentPage }: { currentPage: string }) => {
  return (
    <div>
      { /* Conditional rendering based on currentPage state */ }
      {currentPage === "dashboard" && <Dashboard />}
      {currentPage === "feedback" && <Feedback />}
      {currentPage === "user" && <User />}

    </div>
  );
};

// const User = () => {
//   return <h1 className="text-black">User List Content</h1>;
// };

const Dashboard = () => {
  return <h1 className="text-black">Dashboard Content</h1>;
};

const toggleDropdown = () => {
  setIsOpen(!isOpen);
};

```

# Chapter 6: System Evaluation

## 6.1. Sample Test Plans/Cases

These test cases cover the major functionalities outlined in the system requirements for each user role. Each test case includes a unique ID, description, preconditions, test steps, and expected results.

### 6.1.1. Test Cases for System Admin

#### **Test Case SAR1: Admin Login**

##### **Test Case ID:TC-SAR1-1**

Description: Verify that the admin can log in with the correct username and password.

Preconditions: Admin is registered in the system.

Test Steps:

1. Navigate to the admin login page.
2. Enter a valid admin username.
3. Enter the correct password.
4. Click on the "Login" button.

Expected Result: Admin should be successfully logged in and redirected to the dashboard.

##### **Test Case ID:TC-SAR1-2**

Description: Verify that the admin cannot log in without the correct password.

Preconditions: Admin is registered in the system.

Test Steps:

1. Navigate to the admin login page.
2. Enter a valid admin username.
3. Enter an incorrect password.
4. Click on the "Login" button.

Expected Result: Admin should not be logged in or redirected to the dashboard.

##### **Test Case ID:TC-SAR1-3**

Description: Verify that the admin cannot log in without the correct username.

Preconditions: Admin is registered in the system.

Test Steps:

1. Navigate to the admin login page.
2. Enter a invalid admin username.
3. Enter the correct password.
4. Click on the "Login" button.

Expected Result: Admin should not be logged in or redirected to the dashboard.



**Test Case ID:TC-SAR1-4**

Description: Verify that the admin cannot log in with an incorrect username and an incorrect password.

Preconditions: Admin is registered in the system.

Test Steps:

1. Navigate to the login page.
2. Enter an invalid admin username.
3. Enter an incorrect password.
4. Click on the "Login" button.

Expected Result: Admin should not be logged in or redirected to the dashboard.

**Test Case SAR2: Admin Get System Status****Test Case ID:TC-SAR2**

Description: Verify that the admin can view the system status from the dashboard.

Preconditions: Admin is logged in.

Test Steps:

1. Navigate to the system dashboard.
2. Check the system status section.

Expected Result: System status should be displayed correctly.

**Test Case SAR3: Admin Configure System Settings****Test Case ID: TC-SAR3**

Description: Verify that the admin can configure system settings.

Preconditions: Admin is logged in.

Test Steps:

1. Navigate to the system settings page.
2. Update general system preferences.
3. Save the changes.

Expected Result: Changes should be saved and reflected in the system.

**Test Case SAR4: Admin Manage User Accounts****Test Case ID: TC-SAR4**

Description: Verify that the admin can create, edit, and manage user accounts.

Preconditions: Admin is logged in.

Test Steps:

1. Navigate to the user management page.
2. Add a new user by entering required information.
3. Edit an existing user's information.
4. Manage access permissions for a user.

5. Save the changes.

Expected Result: User account should be created/updated successfully with the correct permissions.

#### **Test Case SAR5: Admin Receives Contact Us details**

##### **Test Case ID: TC-SAR5**

Feedback viewing

Description: Verify that the admin can receive the form data from the contact us section

Preconditions: Admin is logged in.

Test Steps:

1. Navigate to the feedback page.
2. View a list of messages

Expected Result: Admin should be able to receive messages and follow up with customer inquiry.

### 6.1.2. Test Cases for Project Manager

#### **Test Case PMR1: Project Manager Login**

##### **Test Case ID: TC-PMR1-1**

Description: Verify that the project owner can log in with the correct username and password.

Preconditions: Project owner is registered in the system.

Test Steps:

1. Navigate to the project owner login page.
2. Enter a valid project owner username.
3. Enter the correct password.
4. Click on the "Login" button.

Expected Result: Project owner should be successfully logged in and redirected to the home page.

#### **Test Case PMR1: Project Manager Login**

##### **Test Case ID: TC-PMR1-2**

Description: Verify that the project owner cannot log in without the correct username and password.

Preconditions: None

Test Steps:

1. Navigate to the login page.
2. Enter an invalid project owner username.
3. Enter an incorrect password.
4. Click on the "Login" button.

Expected Result: Project owner should not be successfully logged in and redirected to the home page.

### **Test Case PMR2: Project Manager Home Page**

#### **Test Case ID: TC-PMR2**

Description: Verify that the project owner can access the home page after logging in.

Preconditions: Project owner is logged in.

Test Steps:

1. Log in as the project owner.
2. Observe the home page.

Expected Result: Home page should be displayed with project owner-specific options.

### **Test Case PMR3: Create Projects and Assign Tasks**

#### **Test Case ID: TC-PMR3**

Description: Verify that the project manager can create projects and assign tasks.

Preconditions: Project manager is logged in.

Test Steps:

1. Navigate to the project creation page.
2. Enter the required information to create a new project.
3. Navigate to the created project's dashboard.
4. Assign tasks to team members.

Expected Result: Project should be created and tasks assigned.

### **Test Case PMR4: Add Project Members**

#### **Test Case ID: TC-PMR4**

Description: Verify that the project manager can add members to the project.

Preconditions: Project manager is logged in.

Test Steps:

1. Navigate to the Add Member icon.
2. Search for potential Project Members.
3. Assign Role and Add Member.

Expected Result: Members should be added successfully to the specified project.

### **Test Case PMR5: Oversee Project Life Cycle**

#### **Test Case ID: TC-PMR5**

Description: Verify that the project manager can oversee the entire project life cycle.

Preconditions: Project manager is logged in.

Test Steps:

1. Navigate to the project overview page.
2. Plan resources, allocate tasks, manage budget, and monitor progress.

Expected Result: Project manager should be able to manage all aspects of the project life cycle successfully.

### **Test Case PMR6: Create and Reuse Form Templates**

#### **Test Case ID: TC-PMR6**

Description: Verify that the project manager can create customizable data gathering forms and save/reuse form templates.

Preconditions: Project manager is logged in.

Test Steps:

1. Navigate to the forms creation page.
2. Create a new form and save it as a template.
3. Reuse the saved template in a different project.
4. Use search and filtering options to locate the form templates.

Expected Result: Form templates should be created, saved, and reused successfully with proper search and filtering functionality.

## **6.1.3 Test Cases for Supervisor**

### **Test Case SR1: Supervisor Login**

#### **Test Case ID: TC-SR1**

Description: Verify that the supervisor can log in with the correct username and password.

Preconditions: Supervisor are registered in the system.

Test Steps:

1. Navigate to the login page.
2. Enter a valid username and password.
3. Click on the "Login" button.

Expected Result: Supervisor should be successfully logged in and redirected to the home page.

### **Test Case SR2: Supervisor Home Page**

#### **Test Case ID: TC-SR2**

Description: Verify that the supervisor can access the home page after logging in.

Preconditions: Supervisor is logged in.

Test Steps:

1. Log in as a supervisor.
2. Observe the home page.

Expected Result: Home page should be displayed with supervisor-specific options.

**Test Case SR3: Check Data Correctness****Test Case ID: TC-SR3**

Description: Verify that the supervisor can access all the documents in a given project and check data correctness.

Preconditions: Supervisor is logged in.

Test Steps:

1. Navigate to the documents page.
2. Check data correctness.

Expected Result: Documents should be viewed and data correctness verified.

**Test Case SR4: Assign Tasks****Test Case ID: TC-SR4**

Description: Verify that the supervisor can assign tasks.

Preconditions: Supervisor is logged in.

Test Steps:

1. Navigate to the calendar page.
2. Enter the required information to create a new task.
3. Assign tasks to team members.

Expected Result: Tasks should be assigned.

**Test Case SR5: Create and Reuse Form Templates****Test Case ID: TC-SR5**

Description: Verify that the supervisor can create customizable data gathering forms and save/reuse form templates.

Preconditions: Supervisor is logged in.

Test Steps:

1. Navigate to the forms creation page.
2. Create a new form and save it as a template.
3. Reuse the saved template in a different project.
4. Use search and filtering options to locate the form templates.

Expected Result: Form templates should be created, saved, and reused successfully with proper search and filtering functionality.

**6.1.4 Test Cases for Researcher****Test Case RR1: Researcher Login****Test Case ID: TC-RR1**

Description: Verify that researchers can log in with the correct username and password.

Preconditions: Researcher is registered in the system.

Test Steps:

1. Navigate to the researcher login page.
2. Enter a valid researcher username.
3. Enter the correct password.
4. Click on the "Login" button.

Expected Result: Researcher should be successfully logged in and redirected to the home page.

### **Test Case RR2: Researcher Home Page**

#### **Test Case ID: TC-RR2**

Description: Verify that researchers can access the home page after logging in.

Preconditions: Researcher is logged in.

Test Steps:

1. Log in as a researcher.
2. Observe the home page.

Expected Result: Home page should be displayed with researcher-specific options.

### **Test Case RR3: View Assigned Tasks**

#### **Test Case ID: TC-RR3**

Description: Verify that the researcher can view the tasks assigned to him/her.

Preconditions: Researcher is logged in.

Test Steps:

1. Navigate to My Tasks page.
2. Verify that the tasks assigned are visible.
3. Mark them as to-do, in progress or done depending on the stage of the task

Expected Result: Researchers' assigned tasks should be visible.

### **Test Case RR3: Upload and Manage Research Data**

#### **Test Case ID: TC-RR3**

Description: Verify that researchers can upload and manage research data.

Preconditions: Researcher is logged in.

Test Steps:

1. Navigate to the document and file page.
2. Upload research data.
3. Manage the uploaded data.
4. Receive notifications and reminders.

Expected Result: Research data should be uploaded, managed, and notifications received successfully. Form templates should be created and reused with search and filtering functionality.

## 6.1.5 Test Cases for Nonfunctional Requirements

### NFR1. Performance

#### **Test Case NFR1.1: Page Load Times**

##### **Test Case ID: TC-NFR1.1**

Description: Verify that 95% of pages load within 5 seconds on a standard internet connection.

Preconditions: Set up a simulated run.

Test Steps:

1. Access the platform using a standard internet connection.
2. Navigate to various pages within the platform.
3. Measure and record the load time for each page.

Expected Result: 95% of pages should load within 5 seconds.

#### **Test Case NFR1.2: API Response Times**

##### **Test Case ID: TC-NFR1.2**

Description: Verify that API calls have a maximum response time of 5 seconds.

Preconditions: System APIs are up and running.

Test Steps:

1. Execute a series of API calls.
2. Measure and record the response time for each API call.

Expected Result: All API calls should have a response time of 5 seconds or less.

#### **Test Case NFR1.3: System Uptime**

##### **Test Case ID: TC-NFR1.3**

Description: Verify that the platform is available 94.5% of the time, excluding scheduled maintenance periods.

Preconditions: System monitoring tools are set up.

Test Steps:

1. Monitor the system uptime over a specified period.
2. Record any downtime events and their durations.

Expected Result: The platform should be available 94.5% of the time.

### NFR2. Scalability

#### **Test Case NFR2.1: User Load**

##### **Test Case ID: TC-NFR2.1**

Description: Verify that the platform can accommodate at least 500 concurrent users without significant performance degradation.

Preconditions: Load testing tools are set up.

Test Steps:

1. Simulate 500 concurrent users accessing the platform.
2. Monitor system performance metrics such as response time and resource utilization.

Expected Result: The platform should handle 500 concurrent users without significant performance degradation.

### **Test Case NFR2.2: Data Storage**

#### **Test Case ID: TC-NFR2.2**

Description: Verify that the platform can store and manage at least 100 GB of research data.

Preconditions: Sufficient storage infrastructure is in place.

Test Steps:

1. Upload research data until the total size reaches 100 GB.
2. Verify data integrity and system performance.

Expected Result: The platform should store and manage 100 GB of data without issues.

## **NFR3. Security**

### **Test Case NFR3.1: Data Encryption**

#### **Test Case ID: TC-NFR3.1**

Description: Verify that all user data and research data are encrypted at rest and in transit using industry-standard algorithms.

Preconditions: Encryption mechanisms are implemented.

Test Steps:

1. Inspect data storage and transmission processes.
2. Verify the use of industry-standard encryption algorithms.

Expected Result: All data should be encrypted at rest and in transit.

### **Test Case NFR3.2: Authentication and Authorization**

#### **Test Case ID: TC-NFR3.2**

Description: Verify that robust user authentication and authorization mechanisms are implemented.

Preconditions: Authentication and authorization mechanisms are in place.

Test Steps:

1. Attempt to access the platform with valid and invalid credentials.
2. Attempt to access restricted areas with different user roles.

Expected Result: Only authorized users should gain access to the platform and its functionalities.



## NFR4. Reliability

### **Test Case NFR4.1: Data Redundancy**

#### **Test Case ID: TC-NFR4.1**

Description: Verify that data replication and backup mechanisms are implemented to ensure data integrity and availability.

Preconditions: Data redundancy mechanisms are in place.

Test Steps:

1. Inspect data replication and backup processes.
2. Test data recovery from backups.

Expected Result: Data should be replicated and recoverable from backups.

### **Test Case NFR4.2: Disaster Recovery Plan**

#### **Test Case ID: TC-NFR4.2**

Description: Verify that a comprehensive disaster recovery plan is developed to minimize downtime and data loss.

Preconditions: Disaster recovery plan is documented.

Test Steps:

1. Review the disaster recovery plan.
2. Conduct a disaster recovery drill.

Expected Result: The disaster recovery plan should minimize downtime and data loss effectively.

## NFR5. Usability

### **Test Case NFR5.1: User Interface**

#### **Test Case ID: TC-NFR5.1**

Description: Verify that the platform has an intuitive and user-friendly interface.

Preconditions: UI design is implemented.

Test Steps:

1. Perform usability testing with users of varying technical skills.
2. Gather and analyze user feedback.

Expected Result: The interface should be easy to navigate and understand for all users.

### **Test Case NFR5.2: Accessibility**

#### **Test Case ID: TC-NFR5.2**

Description: Verify that the platform complies with WCAG 2.1 accessibility guidelines.

Preconditions: Accessibility features are implemented.

Test Steps:

1. Conduct an accessibility audit based on WCAG 2.1 guidelines.
2. Test the platform with assistive technologies.

Expected Result: The platform should be accessible to users with disabilities.

## NFR6. Interoperability

### **Test Case NFR6.1: Data Import and Export**

#### **Test Case ID: TC-NFR6.1**

Description: Verify that the platform supports various data formats for seamless import and export of research data.

Preconditions: Data import and export features are implemented.

Test Steps:

1. Test importing research data in different formats.
2. Test exporting research data in different formats.

Expected Result: Data import and export should be seamless and support various formats.

### **Test Case NFR6.2: Compliance with Research Data Standards**

#### **Test Case ID: TC-NFR6.2**

Description: Verify that the platform adheres to relevant research data standards.

Preconditions: Research data standards are defined and implemented.

Test Steps:

1. Review the implementation of research data standards.
2. Test data sharing and collaboration features.

Expected Result: The platform should comply with relevant research data standards.

## 6.2. Evaluation of the Proposed Design and Solution

### 6.2.1 Functional Requirements Evaluation

**Admin Functionality:** The admin functionalities were thoroughly tested, including login, system status viewing, configuration of system settings, user account management, and feedback management. The following outcomes were observed:

1. Admin Login: The admin can successfully log in with the correct username and password. Attempts to log in with incorrect credentials were correctly denied, ensuring secure access.
2. System Status Viewing: The admin can view the system status from the dashboard, ensuring proper monitoring and management capabilities.
3. System Configuration: The admin can configure system settings, with changes being saved and reflected accurately.
4. User Account Management: The admin can create, edit, and manage user accounts, with all operations performing as expected.

5. Feedback Management: The admin can receive and manage messages from the "Contact Us" section, facilitating effective customer support.

**Project Manager Functionality:** Project manager functionalities, such as login, home page access, project creation, task assignment, member addition, project oversight, and form template management, were validated with the following results:

1. Project Manager Login: The project manager can log in with the correct credentials, ensuring secure access to the platform.
2. Home Page Access: The project manager can access the home page with specific options available for project management.
3. Project and Task Management: Projects can be created, tasks assigned, and project members added successfully. The project manager can oversee the project life cycle, effectively managing resources, tasks, and budgets.
4. Form Template Management: Customizable forms can be created, saved as templates, and reused, supporting efficient data gathering.

**Supervisor Functionality:** Supervisor functionalities, including login, home page access, data correctness checks, task assignment, and form template management, were tested with these findings:

1. Supervisor Login: The supervisor can log in with the correct credentials, ensuring secure access.
2. Home Page Access: The home page displays supervisor-specific options, providing the necessary tools for supervision.
3. Data Correctness Checks: Supervisors can access project documents and verify data correctness, ensuring data integrity.
4. Task Assignment: Supervisors can assign tasks to team members, facilitating effective task management.
5. Form Template Management: Form templates can be created, saved, and reused, ensuring efficient data management.

**Researcher Functionality:** Researcher functionalities, such as login, home page access, task viewing, and data management, were evaluated with the following results:

1. Researcher Login: Researchers can log in with correct credentials, ensuring secure access.
2. Home Page Access: The home page provides researcher-specific options, enhancing usability.
3. Task Viewing: Researchers can view assigned tasks and update their status, supporting effective task tracking.

4. Data Management: Researchers can upload, manage, and receive notifications for research data, ensuring efficient data handling.

## 6.2.2 Non-functional Requirements Evaluation

### **Performance:**

1. Page Load Times: 95% of pages load within 5 seconds on a standard internet connection, meeting the performance criteria.
2. API Response Times: API calls have a maximum response time of 5 seconds, ensuring efficient interaction with the platform.
3. System Uptime: The platform maintains an availability of 94.5%, excluding scheduled maintenance, meeting the reliability standards.

### **Scalability:**

1. User Load: The platform failed to accommodate 500 concurrent users without performance degradation, indicating a need for optimization in handling high user loads.
2. Data Storage: The platform struggled to manage 100 GB of research data, requiring improvements in data storage capabilities.

### **Security:**

1. Data Encryption: All user data and research data are encrypted at rest and in transit using industry-standard algorithms, ensuring robust data security.
2. Authentication and Authorization: Robust authentication and authorization mechanisms are implemented, controlling access effectively.

### **Reliability:**

1. Data Redundancy: Data replication and backup mechanisms are in place, ensuring data integrity and availability.
2. Disaster Recovery Plan: A comprehensive disaster recovery plan is developed, minimizing downtime and data loss during unforeseen events.

**Usability:**

1. User Interface: The platform has an intuitive and user-friendly interface, making it accessible to users with varying technical skills.
2. Accessibility: The platform fails to fully comply with WCAG 2.1 accessibility guidelines, indicating a need for improvements to support users with disabilities.

**Interoperability:**

1. Data Import and Export: The platform supports various data formats, facilitating seamless import and export of research data.
2. Compliance with Research Data Standards: The platform partially complies with relevant research data standards, requiring further enhancements to facilitate better data sharing and collaboration.

## 6.3. Discussion of Results

The evaluation of the proposed design and solution highlights both the strengths and weaknesses of the platform. The platform demonstrates strong capabilities in functional requirements, ensuring effective user management, project handling, and data management. However, there are notable areas for improvement, particularly in scalability and accessibility.

The scalability tests revealed that the platform did not meet the preconditions for handling high user loads and large data volumes. Specifically, the backend infrastructure and data management strategies need optimization to support at least 500 concurrent users and manage 100 GB of research data without significant performance degradation. In terms of accessibility, the platform does not fully meet the WCAG 2.1 guidelines, indicating that enhancements are needed to ensure usability by all users, including those with disabilities.

In addition to scalability and accessibility, another section where improvement is needed is in ways of Interoperability. Interoperability tests showed partial adherence to research data standards. While the platform facilitates communication and ensures data encryption, it lacks an auditable workflow, which is essential for maintaining comprehensive compliance with research data standards. This gap affects data sharing and collaboration among researchers, indicating a need for improvements in this area to fully support interoperability.

# Chapter 7: Conclusion and Recommendations

## 7.1. Conclusion

The Comprehensive Research and Project Management Platform (CRPMP) was developed to address the multifaceted challenges researchers face in academia, industry, and individual projects. The platform provides a centralized hub designed to streamline the research lifecycle, encompassing planning, proposal development, execution, data analysis, and dissemination. This project utilized a range of methodologies, including requirement gathering, detailed design, iterative development, and rigorous testing, to ensure the platform meets the diverse needs of its users.

Throughout the development process, several tools and methodologies were employed to achieve the desired outcomes. Object-Oriented Design (OOD) principles, User-Centered Design (UCD), and an agile development approach were crucial in developing a robust and adaptable system. The choice of tools and technologies, such as ASP.NET for backend development and various database management systems, facilitated the creation of a scalable and secure platform.

The primary motivation behind developing this system was to mitigate common issues in research management, such as data documentation inconsistencies, manual process inefficiencies, and the integration gap between research outcomes and organizational decision-making. By offering features like project creation and management, customizable data gathering forms, collaboration tools, and a robust data analysis dashboard, the platform aims to enhance research productivity and collaboration.

## 7.2. Recommendation

To further enhance the Comprehensive Research and Project Management Platform (CRPMP) and address its current limitations, several recommendations are proposed. Firstly, optimizing scalability is crucial; this can be achieved by enhancing the platform's backend infrastructure and data management strategies to handle higher user loads and larger data volumes more efficiently. Implementing load balancing, database optimization, and efficient data handling techniques are key steps in this direction. Additionally, improving accessibility to achieve full compliance with WCAG 2.1 accessibility guidelines is essential. This involves making the user interface more inclusive and ensuring all functionalities are accessible to users with disabilities.

Another important recommendation is to improve interoperability by ensuring full adherence to research data standards. This includes implementing an auditable workflow to maintain comprehensive compliance and improve data integrity, facilitating better data sharing and collaboration. Incorporating artificial intelligence (AI) into the platform is also recommended to

enhance various aspects such as advanced data analysis, predictive analytics, and automating routine tasks, thereby increasing the platform's efficiency and user experience.

Developing mobile applications will further expand the platform's accessibility, allowing users to manage their research projects on the go, providing greater flexibility and convenience. Lastly, establishing a continuous improvement process based on user feedback is vital. Regular updates and enhancements should be made to address user needs and incorporate the latest technological advancements. By addressing these recommendations, the CRPMP can evolve into a more robust, scalable, and user-friendly solution, ensuring its relevance and effectiveness in the ever-evolving research landscape.

## References

- [1] Gail Birkbeck, Tadhg Nagle & David Sammon (2022) “Challenges in research data management practices: a literature analysis”, *Journal of Decision Systems*, 31:sup1, 153-167
- [2] United Nations Department of Economic and Social Affairs. (2023). \*Statistical Yearbook 2023\*. United Nations. <https://unstats.un.org/UNSDWebsite/Publications/StatisticalYearbook/>
- [3] Thor Olof Philogene (2022) “3 Common Research Management Challenges”. <https://www.quirks.com/articles/3-common-research-management-challenges>
- [4] Kyle Chard, Ian Foster, and Steven Tuecke. (2017). “Globus: Research Data Management as Service and Platform.” In *Proceedings of PEARC17*, New Orleans, LA, USA, July 09-13, 2017,
- [5] Amorim, R. C., Castro, J. A., Rocha da Silva, J., & Ribeiro, C. (2017). “A comparison of research data management platforms: architecture, flexible metadata and interoperability.” *Universal access in the information society*, 16, 851-862.
- [6] Bosch, Anita. (2011). “Research management and research output”. *Acta Commerci*. 11. 10.4102/ac.v11i2.148.
- [7] Lanskoronskis, M., Ramoniene, L. and Barsauskas, P. (2009), "Innovative research management as a tool for institutional competitiveness", *Baltic Journal of Management*, Vol. 4 No. 3, pp. 353-368. <https://doi.org/10.1108/17465260910991037>
- [8] Schwalbe, K., (2009). “Introduction to project management”. Boston: Course Technology Cengage Learning.
- [9] Tatnall, A., & Davey, B. (2009). “Research Management Systems as an Evolutionary Backwater: A Management System for Australian University Research Quality Framework Data.” *Evolution of Information Technology in Educational Management* 8 (pp. 143-154). Springer US.
- [10] Bammer, G., (2008). Enhancing research collaborations: Three key management challenges. *Research policy*, 37(5), pp.875-887.
- [11] Rutherford, S., & Langley, D. (2007). “Implementation of Systems to Support the Management of Research: Commentary from a UK University Perspective.” *Journal of Research Administration*, 38(1), 49-60.