RoboNav

SE4450: Software Engineering Design
Project Estimation Report

Faculty Advisor: **Dr. Yili (Kelly) Tang**

Email: ytang564@uwo.ca

Team 7:

Bryson Crook
bcrook4@uwo.ca
251217987

Christopher Higgins
chiggi24@uwo.ca
251245390

Mohamed El Dogdog
meldogdo@uwo.ca
251239204

Seth Langendoen
slangend@uwo.ca
251226610

October 10, 2024

GitHub Repo: https://github.com/meldogdo/RoboNav

Revision History

T – Team | C – Chris | M – Mohamed | B – Bryson | S - Seth

Date	Version	Description	Authors
October 10, 2024	1.0	Initial draft of the estimation report.	B
October 10, 2024	1.1	Added project scope and deliverables.	T
October 11, 2024	1.2	Revised risk and mitigation strategies.	T
October 12, 2024	1.3	Updated timeline and key phases.	T
October 12, 2024	1.4	Final edits and formatting adjustments.	B

Table of Contents

1. Project Overview

1.1 Purpose

The purpose of the *RoboNav* project is to enhance the backend API and develop a user-friendly mobile application to improve the remote control and monitoring of autonomous robots. This project addresses the increasing demand for efficient, real-time robot management systems, particularly in structured indoor environments such as labs or atriums, where efficient navigation and oversight are crucial. By optimizing the backend API and creating a systematic mobile app, the project aims to simplify robot operations, enabling real-time control, live status updates, route assignment, and alerts. Ultimately, *RoboNav* aims to provide a practical and effective solution that enhances users' ability to control and monitor autonomous robots efficiently, leading to improved task performance and reduced operational interruptions.

1.2 Objectives

The primary objectives of the *RoboNav* project are to enhance the backend API, ensuring reliable and efficient communication between autonomous robots and the mobile application, while also creating a user-friendly mobile app that facilitates seamless control and monitoring of these robots. Specifically, the project aims to implement features such as real-time status updates, remote control capabilities, and route assignment functionalities, allowing users to interact intuitively with the robots. Additionally, there will be a strong emphasis on optimizing the backend for scalability and efficiency, ensuring it can accommodate multiple simultaneous users and devices without compromising performance. Through these objectives, *RoboNav* seeks to significantly improve the user experience and operational efficiency in managing autonomous robots across various indoor environments.

1.3 Scope

The scope of the *RoboNav* project focuses on enhancing the existing backend API to improve its performance, scalability, and functionality, rather than creating a new backend from scratch. Key activities include optimizing the API to support real-time data handling, ensuring seamless communication between the mobile application and autonomous robots, and establishing a cloud infrastructure for secure data storage and access. Additionally, the project involves developing a user-friendly mobile application that allows users to effectively control and monitor robots, including features for real-time status updates and route assignments. The project will primarily target structured indoor environments, ensuring that the improvements address the specific needs of users interacting with autonomous robots in these settings.

1.4 Deliverables

- Project Estimation Report
- Software Requirements Specification (SRS)
- Software Design Specification (SDS)
- API Documentation
- Mobile App Wireframes and Design Mock-ups
- Mobile App Source Code
- Backend Source Code
- Test Plan and Test Cases
- Cloud Infrastructure Setup
- Mobile-Backend Integration Plan
- Verification and Validation Document
- User Manual (for both API and mobile app)

2. Estimated Methodology

2.1 Approach

To estimate the time and resources needed for the *RoboNav* project, we will use a mix of expert opinions, past project data, and step-by-step planning. Team members with experience in improving backend APIs and developing mobile apps will share insights from their previous work, helping us identify challenges and necessary resources. We will also look at data from similar projects to set realistic timelines and resource needs. The project will be broken down into smaller tasks, each with its own estimated time to complete, making it easier to manage. By using an agile approach, we can adjust our estimates as the project moves forward, holding regular reviews to gather feedback and make changes as needed. We will also use tools like Gantt charts and project management software to visualize timelines and tasks, helping us see the overall plan and spot any potential issues early on.

2.2 Tools

For the *RoboNav* project, we will use several tools and software to ensure effective planning and management. Gantt charts will help visualize the project timeline, showing the sequence of tasks and key milestones. We will utilize project management software like Jira to track tasks, assign responsibilities, and monitor progress, keeping the team informed and organized. Additionally, we will use Discord for communication among team members, ensuring everyone stays connected and can easily share updates. By using these tools, we aim to streamline our workflow and improve collaboration throughout the project.

3. Key Phases and Tasks

The *RoboNav* project will consist of the following key phases:

- Project Proposal (October 9th)
- Walkthrough (November 6th)
- Sprint 1 and 2 Plan Report (November 6th)
- Demo of Release 1 (December 4th)
- Presentation and Final Demo (March 19th)
- Final Report Retrospective (March 26th)

The project will include 13 key tasks within these key phases:

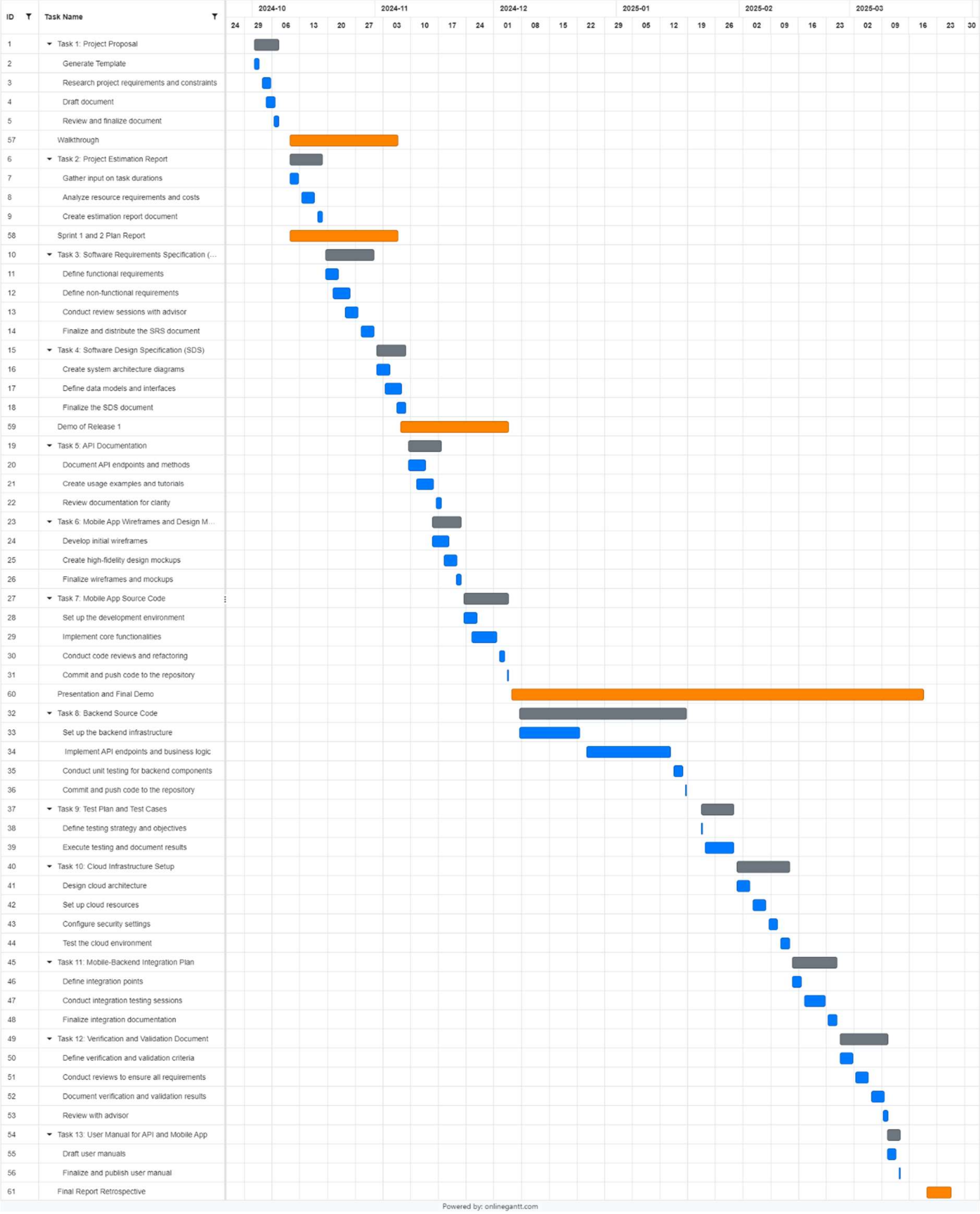
- Project Proposal
    - Generate template
    - Research project requirements and constraints
    - Draft document
    - Review and finalize document
- Project estimation report
    - Gather input on task durations
    - Analyze resource requirements and costs
    - Create estimation report document
- Software requirements specification (SRS)
    - Define functional requirements
    - Define non-functional requirements
    - Conduct review sessions with advisor
    - Finalize and describe the SRS document
- Software design specification (SDS)
    - Create system architecture diagrams
    - Define data models and interfaces
    - Finalize the SDS document

- API documentation
    - Document API endpoints and methods
    - Create usage examples and tutorials
    - Review documentation for clarity
- Mobile app wireframes and design mockups
    - Develop initial wireframes
    - Create high-fidelity design mockups
    - Finalize wireframes and mockups
- Mobile app source code
    - Set up the development environment
    - Implement core functionalities
    - Conduct code reviews and refactoring
    - Commit and push code to the repository
- Backend source code
    - Set up the backend infrastructure
    - Implement API endpoints and business logic
    - Conduct unit testing for backend components
    - Commit and push code to the repository
- Test plan and test cases
    - Define testing strategy and objectives
    - Execute testing and document results
- Cloud infrastructure set up
    - Design cloud architecture
    - Set up cloud resources
    - Configure security settings
    - Test the cloud environment

- Mobile-backend integration plan
    - Define integration points
    - Conduct integration testing sessions
    - Finalize integration documentation
- Verification and validation document
    - Define verification and validation criteria
    - Conduct reviews to ensure all requirements
    - Document verification and validation results
    - Review with advisor
- User manual for API and mobile app
    - Draft user manuals
    - Finalize and publish user manual

4. Timeline and Milestones

The following Gantt chart illustrates the project timeline and key milestones. This timeline was established by our chosen methodology and is designed to accommodate a 10-15% error rate, ensuring flexibility throughout the project.



ID	Task Name
1	Task 1: Project Proposal
2	Generate Template
3	Research project requirements and constraints
4	Draft document
5	Review and finalize document
57	Walkthrough
6	Task 2: Project Estimation Report
7	Gather input on task durations
8	Analyze resource requirements and costs
9	Create estimation report document
58	Sprint 1 and 2 Plan Report
10	Task 3: Software Requirements Specification (...
11	Define functional requirements
12	Define non-functional requirements
13	Conduct review sessions with advisor
14	Finalize and distribute the SRS document
15	Task 4: Software Design Specification (SDS)
16	Create system architecture diagrams
17	Define data models and interfaces
18	Finalize the SDS document
59	Demo of Release 1
19	Task 5: API Documentation
20	Document API endpoints and methods
21	Create usage examples and tutorials
22	Review documentation for clarity
23	Task 6: Mobile App Wireframes and Design M...
24	Develop initial wireframes
25	Create high-fidelity design mockups
26	Finalize wireframes and mockups
27	Task 7: Mobile App Source Code
28	Set up the development environment
29	Implement core functionalities
30	Conduct code reviews and refactoring
31	Commit and push code to the repository
60	Presentation and Final Demo
32	Task 8: Backend Source Code
33	Set up the backend infrastructure
34	Implement API endpoints and business logic
35	Conduct unit testing for backend components
36	Commit and push code to the repository
37	Task 9: Test Plan and Test Cases
38	Define testing strategy and objectives
39	Execute testing and document results
40	Task 10: Cloud Infrastructure Setup
41	Design cloud architecture
42	Set up cloud resources
43	Configure security settings
44	Test the cloud environment
45	Task 11: Mobile-Backend Integration Plan
46	Define integration points
47	Conduct integration testing sessions
48	Finalize integration documentation
49	Task 12: Verification and Validation Document
50	Define verification and validation criteria
51	Conduct reviews to ensure all requirements
52	Document verification and validation results
53	Review with advisor
54	Task 13: User Manual for API and Mobile App
55	Draft user manuals
56	Finalize and publish user manual
61	Final Report Retrospective

5. Resource Allocation

The following RACI matrix outlines the distribution of tasks for the *RoboNav* project, designed to leverage each group member's background and strengths in various areas.

R – Responsible | A – Accountable | C – Consulted | I – Informed

Tasks/Responsibilities	**Bryson Crook**	**Chris Higgins**	**Mohamed El Dogdog**	**Seth Langendoen**
Task 1: Project Proposal				
Generate Template	R/A	I	I	I
Research project requirements and constraints	C	C	R/A	C
Draft document	R/A	R	I	I
Review and finalize document	R/A	R/A	R/A	R/A
Task 2: Project Estimation Report				
Gather input on task durations	A	I	R	I
Analyze resource requirements and costs	R	C	A	C
Create estimation report document	R/A	C	I	I
Task 3: Software Requirements Specification (SRS)				
Define functional requirements	R	C	A	C
Define non-functional requirements	R	I	C	A
Conduct review sessions with advisor	I	C	R/A	I
Finalize and distribute the SRS document	R/A	R/A	R/A	R/A

Task 4: Software Design Specification (SDS)				
Create system architecture diagrams	R/A	C	R/A	I
Define data models and interfaces	C	R	I	A
Finalize the SDS document	R/A	R/A	R/A	R/A
Task 5: API Documentation				
Document API endpoints and methods	C	R/A	R/A	I
Create usage examples and tutorials	A	C	R	I
Review documentation for clarity	R/A	R/A	R/A	R/A
Task 6: Mobile App Wireframes and Design Mockups				
Develop initial wireframes	R/A	C	I	R/A
Create high-fidelity design mockups	R/A	I	R/A	C
Finalize wireframes and mockups	R/A	R/A	R/A	R/A
Task 7: Mobile App Source Code				
Set up the development environment	C	R/A	C	R/A
Implement core functionalities	R/A	I	R/A	C
Conduct code reviews and refactoring	I	R/A	C	R/A
Commit and push code to the repository	R/A	C	I	C
Task 8: Backend Source Code				

Set up the backend infrastructure	C	R	A	C
Implement API endpoints and business logic	I	A	I	R
Conduct unit testing for backend components	R/A	C	R/A	C
Commit and push code to the repository	R/A	I	R/A	C
Task 9: Test Plan and Test Cases				
Define testing strategy and objectives	R/A	C	R	C
Execute testing and document results	R/A	I	C	R
Task 10: Cloud Infrastructure Setup				
Design cloud architecture	C	R/A	I	C
Set up cloud resources	I	R/A	C	I
Configure security settings	A	C	R	I
Test the cloud environment	R/A	C	C	C
Task 11: Mobile-Backend Integration Plan				
Define integration points	R	A	I	C
Conduct integration testing sessions	R/A	C	R/A	C
Finalize integration documentation	R/A	R/A	R/A	R/A
Task 12: Verification and Validation Document				

Define verification and validation criteria	R/A	R/A	C	I
Conduct reviews to ensure all requirements	I	R/A	C	R/A
Document verification and validation results	C	R/A	C	R/A
Review with advisor	I	R	A	I
Task 13: User Manual for API and Mobile App				
Draft user manuals	R/A	C	R/A	I
Finalize and publish user manual	R/A	R/A	R/A	R/A

6. Risk and Mitigation Strategies

The *RoboNav* project encounters various risks that could impact its success. To navigate these challenges effectively, we have identified key risks and developed corresponding mitigation strategies:

1. **Integration Challenges:**
   - **Risk:** Integration issues between the backend API and mobile application may lead to communication failures.
   - **Mitigation:** Conduct regular integration testing throughout the development process and implement continuous integration practices to catch issues early.
2. **Timeline Pressure:**
   - **Risk:** The six-month project timeline may result in rushed development and insufficient testing.
   - **Mitigation:** Develop a detailed project schedule with defined milestones and regular progress reviews to ensure all tasks are completed on time.
3. **User Acceptance:**
   - **Risk:** The mobile application may not meet user expectations or usability standards.
   - **Mitigation:** Involve end-users in the design process and conduct usability testing to ensure the app is intuitive and meets their needs.
4. **Budget:**
   - **Risk:** Budget limitations may restrict access to essential tools and resources.
   - **Mitigation:** Perform a thorough budget analysis to identify cost-effective solutions and explore open-source alternatives.
5. **Data Security:**
   - **Risk:** Storing and transmitting sensitive data poses security vulnerabilities.
   - **Mitigation:** Implement robust security measures, including data encryption and secure authentication protocols, along with regular security audits.

6. **Skill Gaps:**
   - **Risk:** Team members may not possess all the necessary technical skills required for the project.
   - **Mitigation:** Offer training sessions and encourage knowledge sharing to enhance team members' skills.
7. **Performance Issues:**
   - **Risk:** Latency in data communication could impact system performance.
   - **Mitigation:** Optimize network configurations and utilize efficient data transmission protocols to minimize latency.
8. **Stakeholder Engagement:**
   - **Risk:** Insufficient stakeholder involvement may lead to misaligned project goals.
   - **Mitigation:** Schedule regular check-ins with stakeholders for updates and feedback, ensuring their needs are met throughout the project lifecycle.

By addressing these risks early on, the *RoboNav* project aims to enhance its likelihood of success and deliver a reliable solution for managing autonomous robots.

7. Assumptions and Constraints

7.1 Assumptions

1. **Team Availability**
   - It is assumed that all team members will be available and committed throughout the project duration, providing the necessary time and effort to complete their assigned tasks.
2. **Technology Familiarity**
   - The team assumes proficiency in the required programming languages and tools, specifically Python for backend development and chosen mobile frameworks, enabling effective project execution.
3. **Access to Resources**
   - It is assumed that the team will have access to necessary hardware components, cloud services, and other resources required for development and testing.
4. **Stable Requirements**
   - The project assumes that the initial requirements will remain relatively stable, minimizing the need for significant changes during the development phase.
5. **Support from Stakeholders**
   - It is assumed that stakeholders will provide timely feedback and support throughout the project, facilitating necessary adjustments and improvements.

7.2 Constraints

1. **Time Constraints**
   - The project must be completed within a 6-month timeframe, limiting the scope of features that can be realistically developed and tested.
2. **Resource Limitations**
   - Limited access to specific hardware and external datasets may restrict feature implementation, focusing primarily on software development and requiring cloud integration for heavy computations.
3. **Technical Skill Requirements**
   - Success relies on the team's proficiency in specific programming languages, which may limit technology choices.

4. **Budget Constraints**
   - Budget limitations may affect the choice of cloud services, requiring cost-effective solutions.
5. **Real-Time Data Handling**
   - Minimizing latency in communication between robots and the control system requires optimization of the backend API and cloud infrastructure.
6. **User-Friendly Interface**
   - The mobile app must be simple enough for non-technical users, focusing on intuitive design.
7. **Indoor Mapping Limitations**
   - The accuracy of indoor maps used for navigation must be ensured.
8. **Compatibility with Existing Systems**
   - The new backend API must integrate smoothly with existing systems.
9. **Security Concerns**
   - Strong authentication, data encryption, and secure communication protocols are necessary to mitigate security risks.
10. **Testing and Validation Time**
    - Limited testing time may affect the thoroughness of acceptance testing and the final product's quality.

8. Confidence Level

We feel confident in our ability to execute the *RoboNav* project within the six-month timeframe due to our well-structured approach and thorough planning. The project's clear objectives and defined milestones allow us to focus our efforts and track progress effectively. Our team's diverse skill set and experience in software development provide a solid foundation for overcoming potential challenges. Additionally, we have proactively identified risks and developed mitigation strategies to address them, ensuring that we can adapt to any unforeseen issues that may arise. By leveraging tools like Jira for task management and maintaining open communication among team members, we can stay organized and responsive throughout the project. Overall, we believe that our combination of planning, teamwork, and risk management will enable us to deliver the project successfully within the established timeframe.

9. Summary

The RoboNav project aims to enhance the existing backend API for autonomous robot control while developing a user-friendly mobile application for monitoring and controlling robots. The project will be executed over a six-month period, with key milestones including the project proposal, walkthrough, demo of Release 1, and final presentation. The deliverables encompass a comprehensive project estimation report, software requirements and design specifications, API documentation, mobile app wireframes, source code, and testing plans. The methodology will involve a structured approach to project planning using tools like Gantt charts, Jira for task management, and Discord for team communication. Risks such as integration issues, budget constraints, and user acceptance have been identified, along with mitigation strategies to address them effectively. Constraints include limited resources, technical skill requirements, security concerns, and the need for a user-friendly interface. With a clear plan and proactive risk management strategies in place, the team is confident in its ability to deliver a successful project within the stipulated timeframe, ensuring that the final product meets user needs and enhances robot management capabilities.