Building Scalable and Secure Software Solutions to Empower Modern Digital Transformation

by

Ranan Biswas
20341023

An internship submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science and Engineering

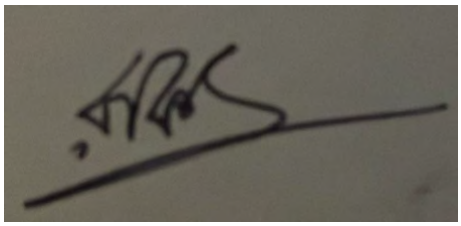
Department of Computer Science and Engineering
BRAC University
October 2025

Declaration

It is hereby declared that

1. The internship submitted is my/our own original work while completing degree at BRAC University.
2. The report does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The report does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. I/We have acknowledged all main sources of help.

Student's Full Name & Signature:



Ranan Biswas
20341023

Approval

The Internship titled "Building Scalable and Secure Software Solutions to Empower Modern Digital Transformation " submitted by

1. Ranan Biswas (20341023)

Of Summer, 2025 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on October 2025.

Examining Committee:

Supervisor:
(Member)

Mr. Tamkin Mahmud Tan
Lecturer
Department of Computer Science and Engineering
BRAC University

Thesis Coordinator:
(Chair)

Md. Golam Rabiul Alam, PhD
Professor
Department of Computer Science and Engineering
BRAC University

Head of Department:
(Chair)

Sadia Hamid Kazi, PhD
Chairperson and Associate Professor
Department of Computer Science and Engineering
BRAC University

Abstract

With the current advent of digital, practically all businesses quickly go through their digital transformations adopting software-driven business environments that require to be scalable and secure. The internship project at RubyTech was centered around the development of a scalable and highly efficient software system to be able to handle project management processes at enterprise level, that would include task management, authentication security and performance enhancements. The system was built with the ability to transmit all the data in real-time with an increased level of reliability using such tools as React, TypeScript, Node.js, PostgreSQL, MongoDB, and security protocol such as OAuth 2.0 and JWT. The methodology involved the use of waterfall in the project which entailed the design of use case, ER diagrams, Class diagrams and activity diagrams. Real examples of implementation were also shown, including code snippets and UI elements, and combine the theoretical concepts of the show into working solutions. This report also offers summaries the process of development, issues encountered and the experience gained as an insight into designing secure and scalable system architecture as it is developed by a developer.

Table of Contents

Chapter 1

Introduction

1.1 About the Internship

Regular internship programs let students learn practical professional experience through authentic job applications. Through hands-on experiences students use their acquired knowledge to learn in actual workplace situations. My internship at RubyTech involved developing scalable and secure software solutions in software development roles. My internship experience provided me integral access to multiple areas within software engineering such as system architecture as well as security protocols alongside performance optimization. Through this experience I collaborated with professional experts who taught me about corporate challenges while working on projects which supported RubyTech's goal to deliver advanced technology solutions.

I executed different assignments throughout this internship that extended my knowledge in software development approaches. My learning covered the essential aspects of secure coding as well as API development and effective methods for system design to manage heavy traffic demands. The professional environment let me learn important soft skills pertaining to teamwork alongside effective communication and time management abilities. My professional placement allowed me to bring academic knowledge into practice by closing the gap between academic education and industrial standards which built my ability to handle real-life situations consecutively.

1.2 About this Report

This report summarizes my internship work at RubyTech. I recapitulate the organizational culture of the company and my work projects and responsibilities from when I served as an intern at RubyTech. This document explains the software development approaches that RubyTech uses while detailing my work in various stages of development. I took part in various tasks that involved improving system scalability and implementing API security measures and enhancing database performance at RubyTech.

The report examines the obstacles I faced as well as the solution approaches and acquired competencies while completing the internship. This document documents my developmental process to create a self-reflection on career progression as a professional. The written record of experiences will serve me well in my future career path.

The technical and practical aspects of software industry employment are explained by this document since it fulfills both my academic needs and provides practical reference information.

1.3 Problem Statement

With the increasing digital transformation rate, most companies experience a continuous problem of combining complex software project management without system scalability or end-to-end security. Traditional project management systems are not very flexible and resilient to facilitate the work of large-scale teams, monitor activities efficiently, and protect sensitive operations against the latest cyber-attacks.
Further, legacy systems normally experience performance bottlenecks, low visibility on project status, real-time analytics and vulnerability left to patching like SQL injection, unauthorized access, and data breach. All these restriction affects the productivity, the delivery timeline, and jeopardizes the business continuity.
The major issue identified as part of this internship is the lack of a single, secure and scalable software platform, which enables carrying out work in real time collaboration, planning the projects, monitoring tasks and tracking the performance, and moreover, within a secure security model.
The rationale for taking up this project was to create and design an industry-grade solution that answers these industry pain points. It was supposed to equip enterprise level demands like role based access control, audit logging, encrypted transmission of data as well as streamlining of the backend processing besides live-time analytics within an architecture that was scalable to reflect the increases in organizational demands.

1.4 Project Objectives

The central aim of this internship project was to design and create a web-based platform capable of modern project management and collaboration work that could serve as the basis of a software engineering solution of industry level. The system was developed to enable organizations that have digitalized to remain highly efficient as it offers them a strong system that can handle fluctuating workloads, different users, and real-time information.
The specific objectives included:

- Developing a secure authentication system with role-based access control using OAuth 2.0 and JWT standards.
- Designing scalable system architecture capable of supporting large volumes of tasks, users, and concurrent operations.
- Creating a real-time project and task tracking environment, including Kanban boards, reporting features, and dashboard analytics.
- Implementing API-level security, including rate limiting, encrypted communication, and endpoint protection.

- Ensuring database optimization through indexing, query refinement, and caching with Redis for faster response times.
- Establishing a complete audit log system to track and record user actions for compliance and monitoring.
- Following the Waterfall Model to execute all development phases systematically from requirement analysis to deployment and maintenance.

This functional requirement driven methodology made sure that the system will not only need to fit the functional demands of a growing business, but also would have high standards regarding reliability, maintainability and security.

1.5 Literature Review

Scalable and secure software solutions have made a tremendous progress in recent years due to the rising need in enterprise-level systems capable of supporting heavy workloads and preserving their integrity in face of stress. Study of the related literature provides the in-depth list of important theoretical frameworks and technologies, which are the basis of modern software design.
The first reason is that scalable systems need to be prepared using the modularity and extensibility concepts to accommodate the high user numbers and large amounts of data. According to Bass, Clements, and Kazman (2012), scalability could be obtained, even with rigorous system design and microservices architecture has become one of the dominant models in this area together with containerization. Microservices make it possible to scale resources dynamically, as well as to manage services independently of one another [3]. Although microservices are beneficial in terms of elasticity, monolithic architectures maintain its significance in a controlled environment where close coupling facilitates deployments. Secondly, software security is not an option anymore it is an essential in-built requirement. The key aspects that must be undertaken to reduce risks SQL injection, XSS, and authentication bypasses, according to OWASP (2023), include secure codes, threat modeling, and continuous testing. Data-at-rest encryption (e.g., AES) and data-in-transit encryption (e.g., TLS) became an industry standard in assuring confidence to a given data and data integrity [4]. Thirdly, real-time monitoring and analytics and is an important part of the maintenance of system performance and health. According to Dean and Ghemawat (2008), real-time analytics is required whenever anomalies are to be detected, when load balancing is to be conducted, and when resources are to be optimized. Improved methods of metrics visualization with a means of alerting against module failures, available in modern tools such as Prometheus and Grafana, enable developers to actively monitor the state of the system to keep it functioning healthily [2]. Moreover, Boehm (1988) has described Waterfall Model as fully structured and linear approach to project development, especially that with clearly-defined requirements. In this internship, the Waterfall was used and each of the four phases (planning, design, implementation, and testing) was completed before the next one started; this results in stronger control and documentation when compared to the iterative models [1]. Finally, a study conducted by Atlassian (2022) demonstrates that visual project management tools, including Kanban boards, increase collaboration and accountability among the team members. These when combined with

role-based dashboards and status updates enhance the efficiency of delegating tasks and make software teams more transparent [5].
RubyTech is evaluated in this report through an organized format which demonstrates both the technological expertise and professional understanding I developed during my work period. The document presents vital information about my education progression because it records my work importance to company projects.

Chapter 2

Company Profile

2.1 Overview of RubyTech

The company RubyTech builds bespoke software solutions for business operations. The company develops applications which provide digital transformation support through secure scalable features at high performance levels. RubyTech supports clients from multiple sectors to develop sturdy software platforms which process extensive data volumes and large user numbers efficiently. Through its use of advanced technologies and industry best practices RubyTech helps businesses achieve operation improvements as well as better system security and performance.

2.2 Vision and Mission

The company aims to rank as a top software firm which boosts business expansion through secure and scalable digital solutions. The company strives to develop performance-ready software capable of processing enormous user traffic with solid security measures across every solution for data protection. As its core purpose RubyTech uses modern development methods together with current technologies for creating innovative solutions enabling digital transformation of businesses. As an organization that prioritizes excellence RubyTech seeks to better its services and develop advanced software solutions which adapt to emerging business requirements across all industries.

2.3 Core Services and Solutions

The technology services at RubyTech span multiple business demands. The main service RubyTech delivers to clients includes developing customized software for specific business needs. The company has expertise in cloud-based solutions so it helps businesses move to cloud platforms to gain enhanced scalability and better reliability while reducing their costs. RubyTech delivers its cyber security services through a vital role which enables full compliance with present-day security benchmarks to secure crucial information against security threats. RubyTech stands out in its expertise of building secure API connections to create smooth inter-system communication through its effective APIs. Performance optimization serves as a

primary objective at the company because they enhance system efficiency to deliver users seamless experiences.

2.4 Agile Software Development Approach (Scrum) for Node.js Projects

RubyTech moves away with the strict Waterfall model to the Agile Scrum model. It implies that we are more responsive to the changes of clients then adhering to the old plan. This provides essential work advantages. We are successful in delivering quickly via our Sprints where we produce pieces of developing software based on the MERN Stack every two to four weeks and provide the clients with immediate usable code. The client validation is a part of it: clients can instantly test functions, and their feedback makes our React and Node.js code the right one that solves the problem. This is a risk reduction strategy as well since in case of a change of requirement we can change the plan at the beginning of the next Sprint and the amount of time that goes to waste is also minimised.

2.5 Continuous Client Collaboration (Product Backlog)

We substitution one long requirements phase with constant communication, with the most valuable work which is the most important in our attention. This starts with project work management using User Stories, in which all features are specified using the need of the client (e.g., As an Admin, I want an API endpoint to update product details... using Node.js/Express.js). Product Backlog is the one and ranked list of all work. This list is managed by the Product Owner to give precedence to features. In Refinement (Estimating), the team refines the complex stories, estimates the effort in Story Points, and defines the technical requirements on the development of the next node.js and TypeScript.

2.6 The Iterative Sprint Cycle (How We Work)

The design, coding and quality assurance are all concurrent, and occur in a two-to-four week repeatable Sprint cycle. It starts with Sprint Planning, in which the team undertakes to work in the next two weeks and has a clear Sprint Goal (e.g., "Complete secure login and multi-role access control.). We keep the pace with the help of the Daily Scrum (Stand-up) that is a short fifteen-minute synchronization at which we recognize blockers (impediments) and continue the work. Coding and Testing This happens simultaneously with the team creating the design and writing JavaScript/TypeScript code and executing automated tests. Test-Driven Development (TDD) is used to achieve quality at the beginning. The Sprint Review (Demo) is the point where we demonstrate the client the new, valuable features (e.g., live Kanban tracking in React). They authenticate whether the software suits their requirements. Lastly, the Retrospective (Improvement) enables the team to talk about how they will work better, faster and smarter the next time they make a cycle part.

2.7 Development & Implementation

During the development phase the writing of clean secure efficient code transforms a software concept into its operational form. Security features get implemented through industry best practices by developers for encryption and authentication which ensure complete protection of user data during every stage. API development plays an essential role during the process because it allows different software components and external services to exchange information with complete efficiency. The project implementation requires developers and testers and project managers to collaborate closely as they verify that the implementation matches both project goals and requirements.

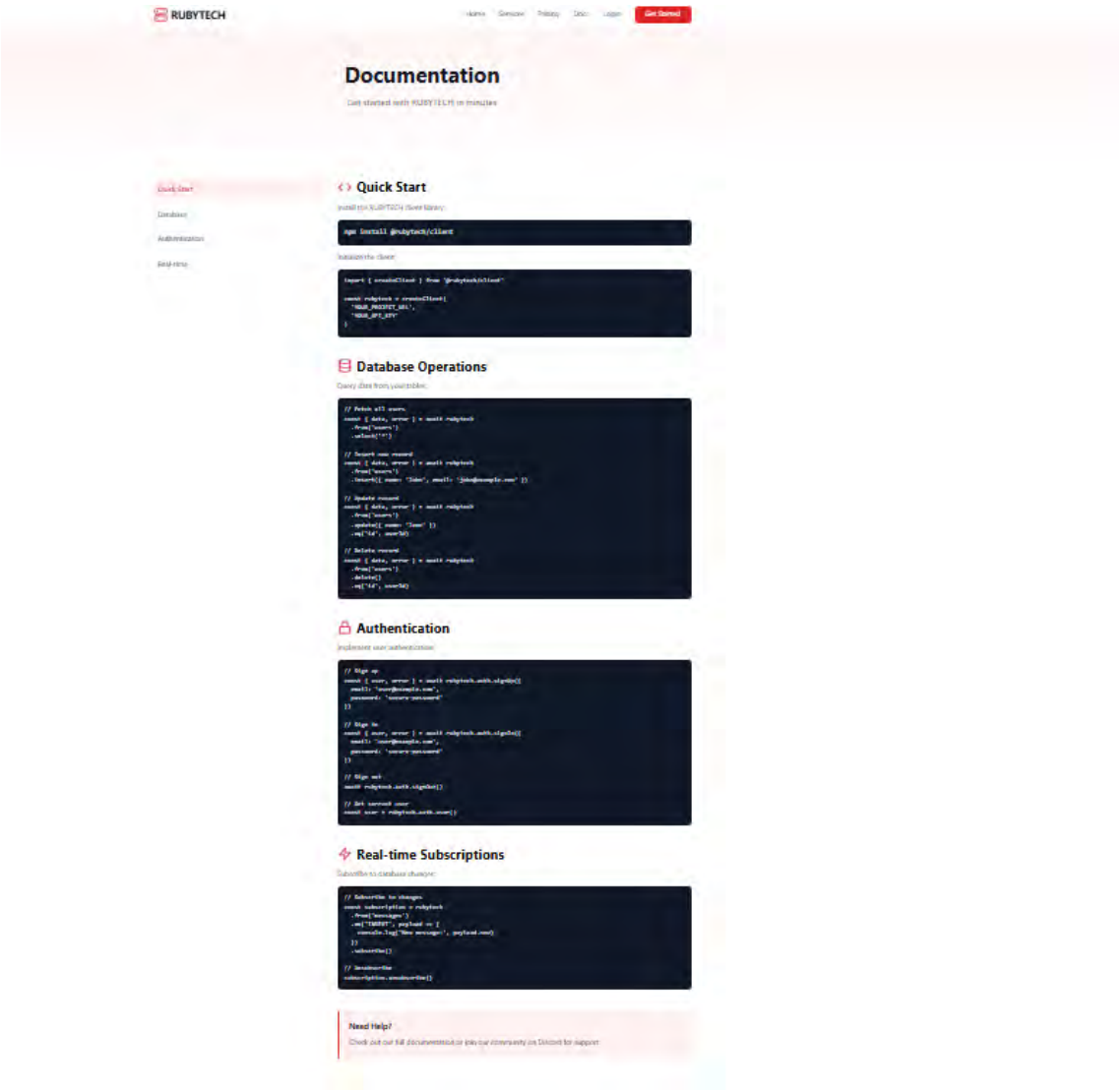


Figure 2.1: Implementation

2.8 Testing & Quality Assurance

Quality assurance tests and deployment validations must be performed thoroughly before the software launch to verify its expected functionality. Manual in addition to

automated testing approaches are utilized in order to detect and resolve any software bugs or system inconsistencies. Testing occurs under heavy traffic situations to verify that the system operates effectively at different scale levels. Security testing stands as an essential phase during which SQL injection vulnerabilities and unauthorized access attempts get detected for mitigation purposes. RubyTech conducts thorough testing procedures to guarantee that the final system achieves complete stability together with deployment security readiness.

2.9 Deployment & Maintenance

The software deployment occurs when testing and quality assurance are successful making the application ready for servers or cloud platforms. During deployment operations engineers configure software platforms to set up system security protocols while integrating the system with present infrastructure. The software run continues after deployment as expert teams monitor its behavior to detect new problems. RubyTech conducts periodic software maintenance procedures which combine security enhancements alongside performance boost operations while also delivering new functionalities as needed. RubyTech's proactive system of software maintenance enables them to maintain clients' solutions as state-of-the-art yet dependable options.

2.10 Technologies and Frameworks Used

RubyTech uses a collection of contemporary technologies as well as framework systems to create its software solutions. The company supports various programming languages like Python together with JavaScript and C for creating software applications that run on different technological platforms. The development of high-performance dynamic applications primarily relies on web frameworks that include Django along with React and ASP.NET. The company runs its data management through three main databases PostgreSQL, MongoDB and MySQL together with systems that support scalable architectures. The three cloud services AWS, Microsoft Azure together with Google Cloud support the deployment and management of applications that deliver high security protocols and consistent operation. Security tools featuring OAuth 2.0, JWT combined with SSL Encryption are built into applications to perform secure authentication protection of data. RubyTech relies on Selenium, JMeter together with Postman to perform automated testing and quality assurance of software functionality which enhances user experience smoothness through validation.

2.11 Workflow

The development is now in the Agile Scrum Framework, which is an iterative process of quick, flexible delivery of features, which RubyTech has shifted to. The system creation is organized into incremental Sprints that start with Product Backlog Creation (User Stories) that leads to System Design and MERN Stack implementation. This is then accompanied with persistent in-Sprint Quality Assurance and Testing. The process is based on a secure CI/CD pipeline to Deploy the application to the

AWS cloud environment followed by continued Maintenance and repeated support to meet strict quality, performance, and security requirements.

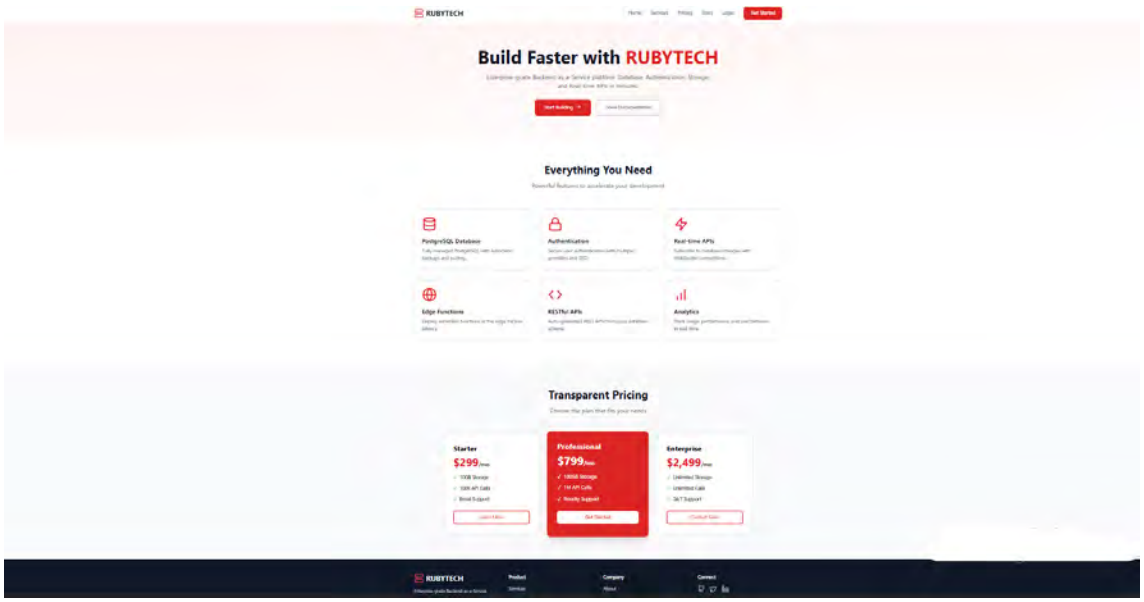


Figure 2.2: Homepage Development using the Agile Scrum Framework

2.12 Key Projects

Through various major projects RubyTech has proven its ability to build flexible and safe software solutions for diverse client needs. The company's main initiative focuses on Enterprise Resource Planning (ERP) system development which offers businesses unified control software to manage operational activities alongside financial matters and human resources and further essential operational divisions. The development of e-commerce platforms stands as a major project for RubyTech because they build secure platforms that provide scalable shopping experiences to both business operators and their customers. RubyTech constructed healthcare management platforms that guarantee safe record handling alongside booking management and remote healthcare solutions. RubyTech builds banking and fintech software and implements security features which provide users with safe and efficient transaction capabilities within the financial technology sector. RubyTech maintains its role as a dependable software partner by delivering top-notch reliable solutions to different business industries through their ongoing projects.

Chapter 3

Training Phase

3.1 Overview of Training Program

RubyTech taught me two main competencies regarding scalable software architecture while implementing best security practices. The traditional classroom training methods were replaced by accessing real-world projects where I learned directly through practical projects. Report generation tools and technological frameworks commonly used in contemporary software development became accessible to me since day one at RubyTech. My hands-on learning experience took place with developers since this method permitted actual project experience.

The educational program delivered comprehensive instructions on software scalability as well as security protocols and database enhancement techniques and performance evaluation procedures. Developers received starting assignments which let them grasp how separate software system elements function when combined. The later part of my development included tackling advanced challenges that involved system architecture design and security implementation work. The training aimed to provide me with both technical competencies and industrial expertise needed for making effective contributions during the software development cycle at the company.

3.2 Understanding Scalable System Architecture

Efficient design of software systems for scaling requirements was a fundamental training subject I learned during my education. I studied two different architectural frameworks which are monolithic and microservices-based architectures. The examination of benefits and limitations between these options allowed me to identify the reasons behind microservices adoption for scalability needs.

The training provided instruction on scaling approaches that include both horizontal and vertical methods. The method of using multiple machines across a system to spread workload represents horizontal scaling yet vertical scaling achieves power enhancement by strengthening existing machines. During the training I studied cloud-based deployment by examining features within AWS and Microsoft Azure platforms. The cloud deployment of applications creates high availability systems which react quickly to sudden user activity increases.

Scalability depends heavily on the rate at which systems acquire and return data

as a main element. The training taught me about Redis caching together with Content Delivery Networks (CDNs) which both minimize database requests and boost system responses. The topic of load balancing was essential because it taught how equal distribution of incoming requests between multiple servers protects a single machine from overloading. I achieved better knowledge of software scalability by implementing these fundamental principles.

3.3 Security-First Development Approach

The major security concern of contemporary software development motivates RubyTech to implement a security-first design across its project work. The training period exposed me to numerous security protocols which both defend user information and stop cyber threats. The core subject I investigated was data encryption because the technique safeguards important information from unauthorized parties who attempt to intercept it. The system's different sections were securely connected through my work with SSL/TLS protocols together with AES encryption techniques.

Applications need to prioritize authentication together with authorization to achieve security measures. The authentication systems based on OAuth 2.0 and JWT managed access controls for different features of the software platform. Application features are accessible to authorized users only through this implementation method. Security studies revealed that SQL injection and cross-site scripting (XSS) attacks offer attackers a way to breach authorized system access. The evaluation of these threats taught me to develop secure code which defends against potential attacks.

The training also provided me with experience in the secure development of APIs. The lesson taught me about endpoint defense systems involving authentication tokens together with rate limiting features and access control protocols. The implementation of security measures through the complete development timeline enables software developers to make applications which resist attacks while protecting stored user information. Through this training I obtained a robust cybersecurity knowledge base that all contemporary software developers need to learn.

3.4 Database and Backend Optimization for Performance

Efficient software operation depends heavily on databases so database queries need precise optimization to develop functional programs. Within my training I utilized PostgreSQL alongside MongoDB and PostgreSQL as SQL and NoSQL database systems. Comparison of these database types enabled me to select appropriate systems that match different application needs.

Learning to index data became one of my key optimization skills because it assists databases to search data faster during query execution. Database queries run inefficiently and slowly when databases lack proper indexing especially while working with datasets that keep growing. The training covered efficient SQL query writing techniques and methods to avoid retrieving excessive database data.

The implementation of Redis data caching serves to boost database performance levels. When data caching stores regularly used information in memory it short-

ens the number of essential visits to the database which results in faster responses. Database normalization along with denormalization represented two important concepts I worked with. The normalization technique helps minimize duplicate data and maintain data consistency yet practitioners need to employ denormalization to boost read operations performance. Such optimizations enable quick application reactions even when numerous users access the system.

3.5 Code Review and Bug Fixing Process

The company RubyTech focuses intensely on preserving code quality as its main organizational priority. During training I joined code reviews to study proper approaches for developing clean maintainable code. Participation in code reviews enabled me to get feedback from more skilled developers who taught me better coding practices and code conformance to industry requirements.

During my training period I gave special attention to debugging techniques alongside other essential knowledge. I mastered the techniques of using log files together with error tracking tools to discover system problems. Debugging necessitates patient analytical thinking to successfully track down the base origin of programming anomalies. I acquired the skill of creating unit tests that confirm every application segment operates properly before release. The creation of test cases remains mandatory because it detects system defects early to avert potential issues which may affect the finished product.

Version control stands as an essential component of present-day software development so I learnt how to use Git as a code change management system. I used GitHub along with GitLab to perform standard development workflows through branching and merging code and pull request execution. The established procedures allow different developers to work on the same project through shared systems.

3.6 Communication and Collaboration in Software Teams

The achievement of successful software development relies heavily on personal competencies regarding both technical abilities and effective communication together with teamwork capabilities. I developed cooperative abilities to work with developers testers and project managers during my training experience. Team members used daily standup meetings to share their work progress including discussion about difficulties they faced and the upcoming planned activities. These regular team meetings enabled every member to stay in harmony regarding project targets and schedule deadlines.

Team members had to effectively manage their assigned tasks as an important part of collaboration activities. Project progress and task management used the combination of Jira and Trello tools. Team members use these programs to build work schedules while giving assignments and tracking due dates. Documentation stood as an important aspect of my training since creating comprehensive detailed reports about my developed code provided other team members with the necessary skills to understand and support my work.

While in my training phase I gathered personal experience about how collaboration with clear communication creates successful software development outcomes. I learned critical work capabilities such as problem-solving skills together with deadline management and adaptability by working in my professional environment. My combined set of soft interpersonal abilities with programming knowledge enabled me to deliver meaningful contributions in actual software development projects.

Chapter 4

My Role and Contributions

4.1 Overview of Methodology

The working process of the internship project was organized in accordance with Waterfall Model. This model has been chosen because it is linear, sequential and since each step is performed, there is direct progress into the other step. Waterfall Model provides high levels of transparency, documentation and control which is highly demanded in a system where data integrity, user access levels and data integrity are vital elements.

- **Requirement Analysis:** The functional and non-functional requirements were obtained by consulting supervisors and client use cases. These needs were used in determining all phases of development.
- **System Design:** Various diagrams in UML were created involving Flowcharts, Use Case Diagrams, Activity Diagrams, ER Diagrams, and Class Diagrams. These acted as the checklist plans of everything in development.
- **Implementation:** Front end was programmed in React.js and the back end developed in Node.js utilizing TypeScript. Relational and non relational databases were achieved through PostgreSQL and MongoDB respectively.
- **Testing:** Tools like Postman, JMeter and automated scripts were set up to perform functional testing, performance testing and security testing. UI verification was also done through Manual testing.
- **Deployment:** The implementation was in a cloud-ready system having the choice of CI/CD integration. All the functions related to secure login, generation of reports, and real-time tracking of tasks were tested in production-like conditions.

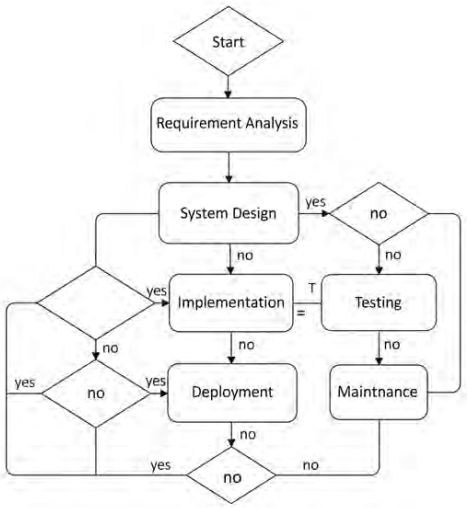


Figure 4.1: WorkFlow

This structured methodology ensured that all system components were built with scalability, security, and maintainability in mind.

4.2 Flowchart of Project Lifecycle

Flowchart Diagram was drawn in detail to create the lifecycle of the whole system along with entry of the user and completion of the task. This flow chart explains the sequence of activities that took place in a classical user journey- authentication through project establishment, assignment of jobs, monitoring the progress and lastly producing reports.

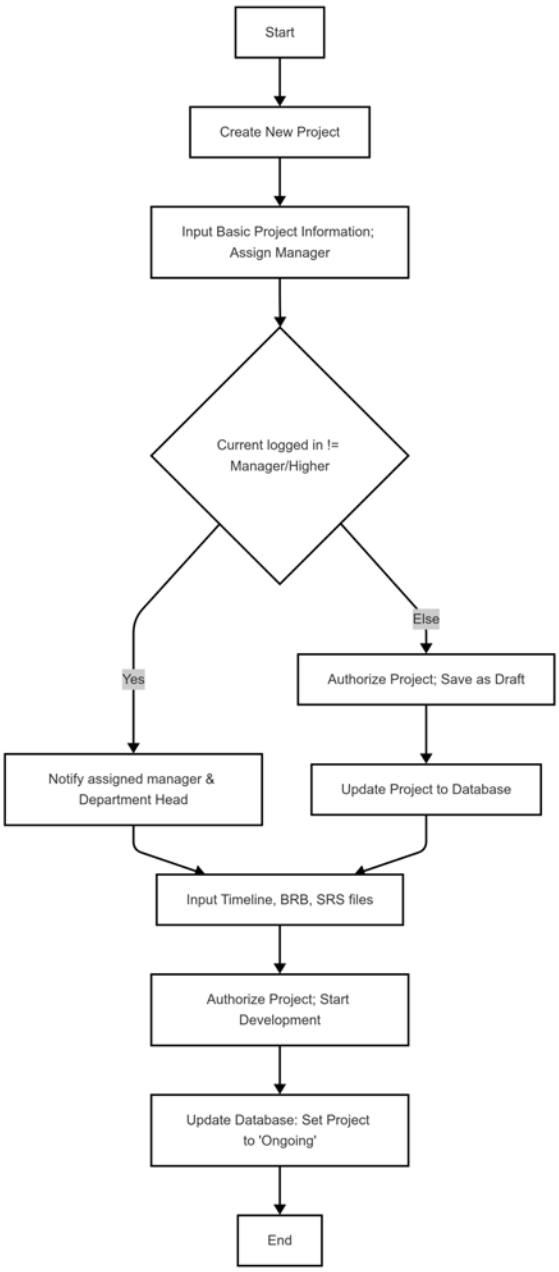


Figure 4.2: System Flowchart

This process is initiated by an authentication system of logging in with credentials processed via a token-based security (JWT/OAuth 2.0). The users are directed to their different dashboards depending on their role. Project Managers will be able to create and delegate work whereas the employees can check and update their own work. Admins, however, have read access and permissions over user management and analytics.
Upon successful login:

- A user with Admin rights can create new projects, add team members, or view system-wide reports.
- A Project Manager can input project details, define timelines, set budgets, and allocate tasks to team members.

- Employees receive assigned tasks and are allowed to update task status (e.g., pending, in progress, completed).

All actions are traced in audit log. As soon as the tasks are completed or adjusted, they are reported in several formats (PDF, Excel, JSON) and the project status is always updated in real-time. This visual flow makes backend controller logic clearer and adds to the user experience overall since interactions tend to be stable and secure.

4.3 Use Case Modeling

In an endeavor to outline the extent of involvement of users and based on their positions that include privileges, a Use Case Diagram was created. The use case diagram was useful in mapping the roles and functions of each of the systems actors, which included Administrator, Project Manager, and Employee and their respective use cases.

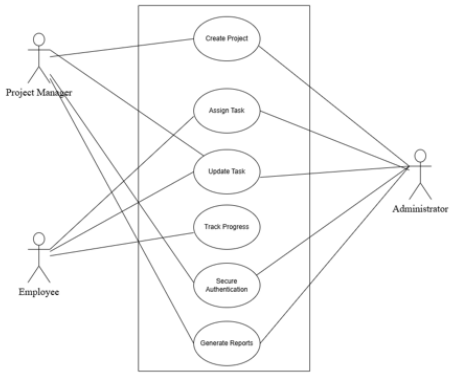


Figure 4.3: Use Case Diagram

Key use cases include:

- **Administrator**: Manage users, create projects, assign managers, generate reports, access audit logs.
- **Project Manager**: Create and manage projects, assign tasks to employees, view project timelines, update statuses.
- **Employee**: View assigned tasks, update task status, submit progress reports.

This diagram was used to create a definitive guide to the definition of feature access levels, authentication of user stories as well as enforcement of role-based access control within the application. It also assisted in backend authentication and authorization planning logic as well.

4.4 Activity Diagram

The Activity Diagram simulates the logic and conditional course of the action, especially the action that the user might perform when creating a project and transferring

tasks to it. This chart offered a design model that is sequence-based to programmers to use during writeback using scripts to the backend.

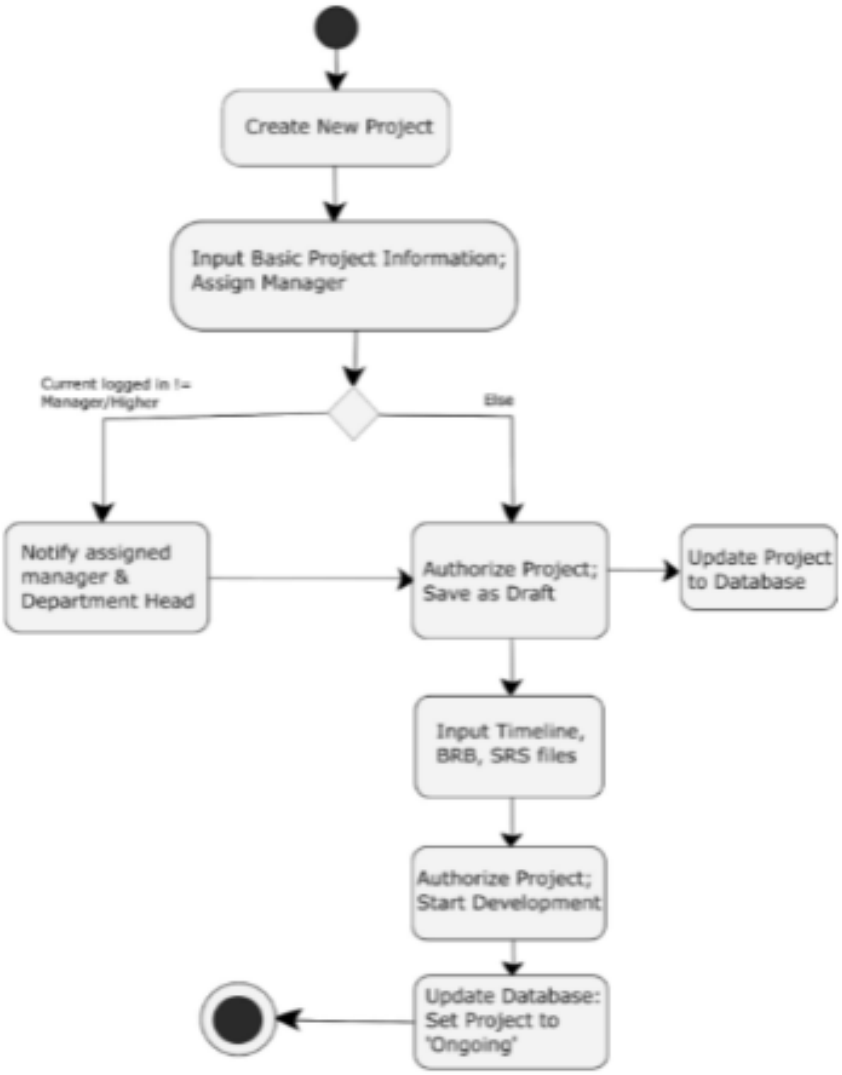


Figure 4.4: Activity Diagram

The flow begins with a user selecting the "Create Project" option. Upon entering details such as the title, description, start date, deadline, and team members, a series of validations are performed:

- Ensuring fields are not empty
- Checking deadline logic
- Validating assigned roles

After confirmation, the system generates the project and saves the information at the database. This will then activate the second activity path that is task assignment. Tasks are input with features such as level of priority, periods and dependencies. Tasks are stored and then linked with a corresponding user and project only after all fields are validated.

The diagram facilitated the avoidance of ambiguity in procedural use of controller functions as well as making sure tasks could not be generated without a valid parent project and/or team set up.

4.5 Entity Relationship (ER) Diagram

An entity relationship diagram (ERD) planning was well undertaken to provide structure to the database, consistency of data, as well as optimization in the storage of the data throughout the operations. It was designed to have a hybrid usage of PostgreSQL in structured information and MongoDB in unstructured logging and JSON-like, reports.

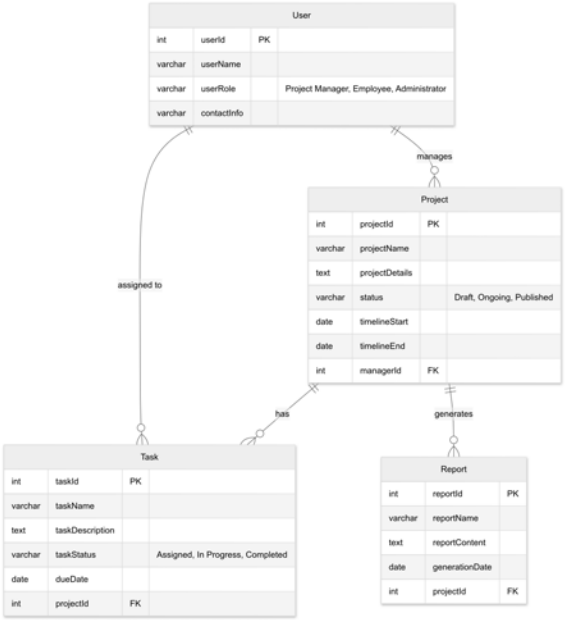


Figure 4.5: Entity Relationship Diagram

Key entities and their relationships include:

- **User**: Each user has a role (Admin, Manager, Employee) and a unique set of permissions.
- **Project**: Contains metadata, timeline, budget, and links to one or more tasks.
- **Task**: Each task is related to a project and assigned to a user, with real-time status tracking.
- **Report**: Generated from project and task completion records, tied to timestamps and formats.
- **ActivityLog**: Captures system-level logs such as login events, task updates, and changes in project status.

The ER model supports many-to-one and one-to-many relationships, ensuring efficient JOIN operations, referential integrity, and simplified queries. Foreign keys, primary keys, and indexing strategies were implemented during schema generation.

4.6 Class Diagram

A detailed Class Diagram has been developed to provide direction in the development of object oriented modules (classes, attributes and methods they interact with). This became the blueprint of backend development in TypeScript and thus, the given way the various entities would act within the application logic.

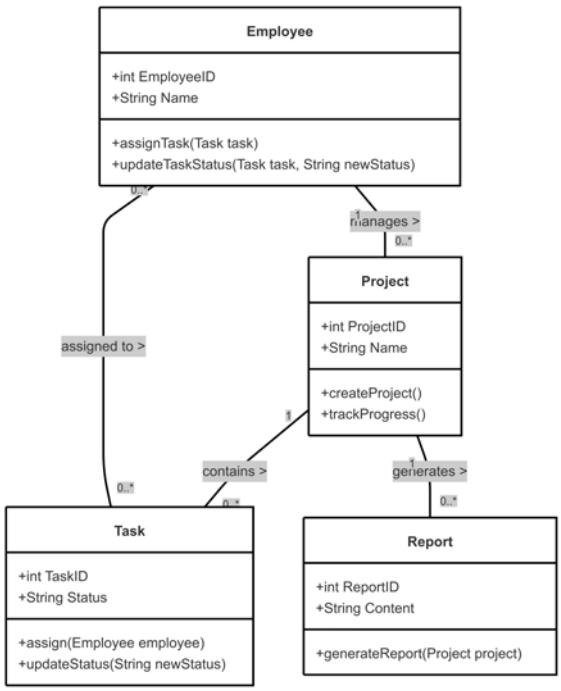


Figure 4.6: Class Diagram

Major classes include:

- **ProjectManager**: Contains methods such as `createProject()`, `assignTask()`, and `generateReport()`.
- **Employee**: Contains methods like `viewTask()`, `updateTaskStatus()`, and `submitReport()`.
- **Task**: Holds properties like title, deadline, status, priority, and methods for tracking progress or dependencies.
- **ReportService**: Provides methods to generate and format reports across multiple file types.

The behavior of each class is enclosed within the logic, the attributes, as well as the responsibilities of the classes thus promoting modularity, reusability, and the clear separation of the responsibility. To have consistency in various roles based operation, method overloading and interface inheritance were applied.

Chapter 5

Challenges and Learning Outcomes

5.1 Development Process (Agile Scrum Framework)

The Agile Scrum methodology was used to develop the system during a set of 2 week Sprints. This incremental process provided the delivery of value features early and constantly, and this facilitated the ability to accommodate changes at a much faster rate than in a conventional Waterfall model. The key functions that influenced this process were well outlined as the Product Owner (who assigns the priority of the features and maximizes the product value) and the Scrum Master (who facilitates the process and removes the impediments), and the Development Team (who builds the product).

5.1.1 Product Backlog Creation

In the first step, the project requirements were gathered in the form of User Stories and ranked into the main artifact called the Product Backlog. It was a crucial stage of work including a lot of co-operation with the project supervisor and internal developers in RubyTech. The key product requirements, defined as Epics, were the creation of a secure system of user authentication and authorization, the implementation of multi-role access control (Admin, Manager, Employee), the development of a full Project and Task Management dashboard with a Kanban-style view, the introduction of real-time Analytics and Reporting (exportable to PDF, JSON, and Excel) and providing a complete Audit Trail to ensure compliance with security requirements.

5.1.2 System Design and Technology Stack (MERN)

The design of the system was done in iterative Sprint Planning sessions which made sure that the architecture was flexible to changes in requirements. It was built on the strong full-stack JavaScript framework, which is commonly referred to as the MERN Stack (modified). A hybrid data storage was used in this stack, with PostgreSQL being used to store project and task data as relational, and MongoDB used to provide flexibility of logs and non-relational data. In addition, the Redis was integrated to

perform effective session management and cache frequently accessed information. Our backend was built on Node.js utilizing the Express.js framework which offered an event-driven, non-blocking architecture which is important in supporting real-time functionality and high concurrency. The frontend or user-facing layer was built with React.js to form a dynamic component-based interface that manages the state a centralized context of state management, in the form of the AuthContext. The Architecture diagrams, such as the Flowchart, Use Case Diagram, Activity Diagram, and ER Diagram, have been developed and modified when required to support the features that were planned to be implemented in the future Sprints to ensure a modular, scalable and maintainable design.

5.1.3 Development & Sprint Execution

The team conducted development in 2-week Sprints, which were observed every day during the Daily Scrums to maintain the constantly maintained alignment as well as to eliminate any immediate blockers. Pulling of work was done out of the Sprint Backlog on the basis of priority and the estimated effort that was set during the planning process. One of the most important products of these early Sprints was the developer documentation, which was constantly evolving, which can be considered an example of a core system artifact required in transferring knowledge and scalability in the future.

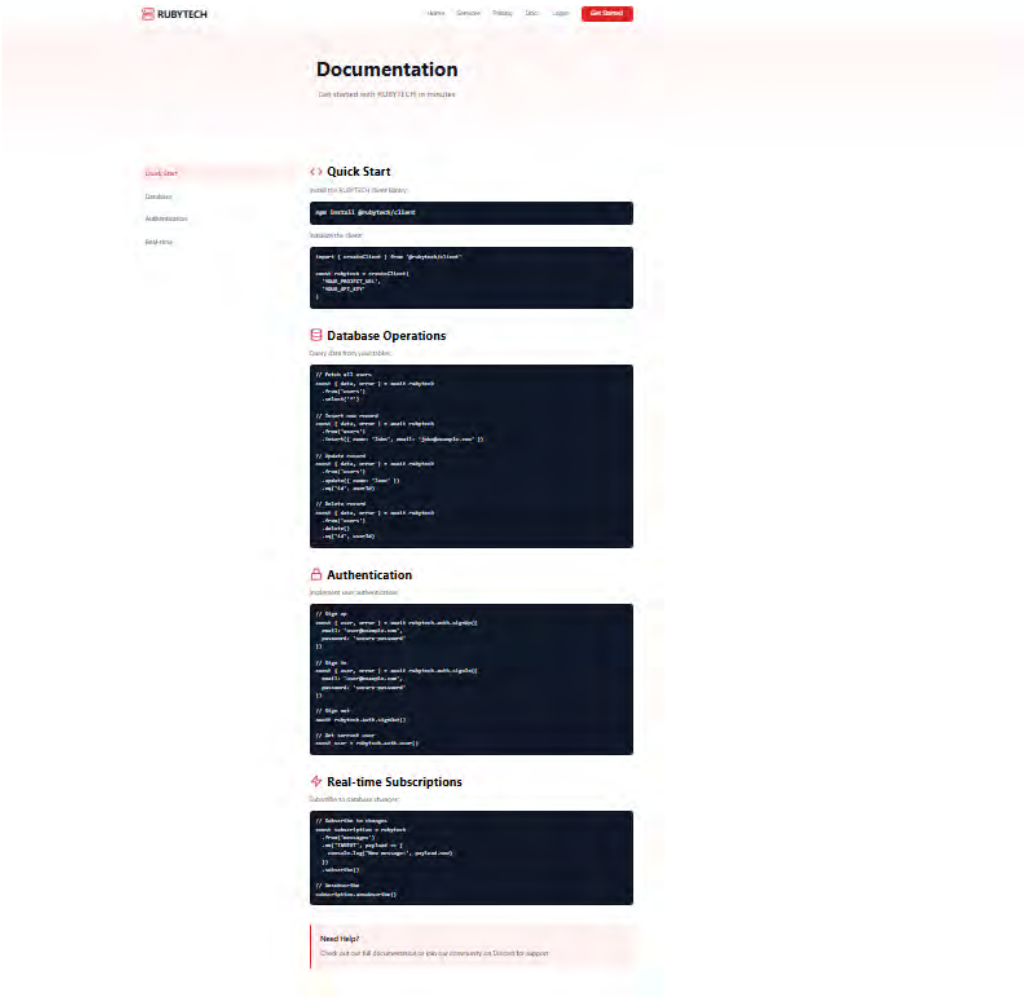


Figure 5.1: Preview of Developer Documentation

Code Snippet – Secure Route Authentication:
This module handles protected routes and user session control based on roles.

```tsx
import React from 'react';
import { BrowserRouter as Router } from 'react-router-dom';
import { AuthProvider, useAuth } from './context/AuthContext';
import Login from './components/Login';
import EnhancedDashboard from './components/enhanced/EnhancedDashboard';

const AppContent: React.FC = () => {
  const { isAuthenticated } = useAuth();

  return isAuthenticated ? <EnhancedDashboard /> : <Login />;
};

function App() {
  return (
    <Router>
      <AuthProvider>
        <AppContent />
      </AuthProvider>
    </Router>
  );
}

export default App;
```

Figure 5.2: Secure Route Authentication

Code Snippet – Project Controller:
The backend logic validates project inputs, handles data fetches, and returns secure responses.

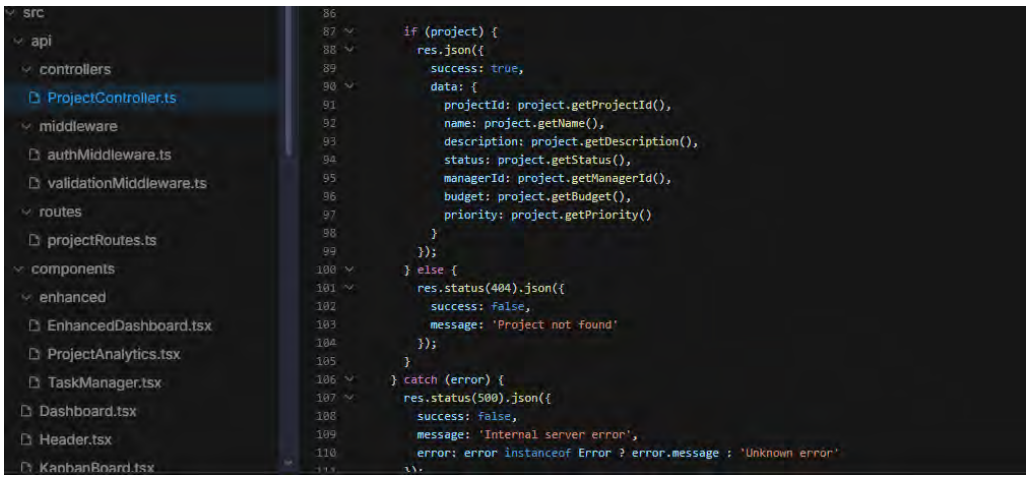


Figure 5.3: Project Controller Code

5.1.4 Testing, Debugging Quality Assurance

Testing was an ongoing, in-house effort of all Sprints, and was organically a part of the development process and not a stage. The systematic quality assurance plan encompassed several areas of system integrity. To be stable, Unit Testing of core modules and utility functions was done using Jest/Mocha to determine logic stability. Postman was used to test the individual API endpoints in a functional

validation, whereas Cypress to test the whole system (End-to-End) was used to precisely replicate the user flow, e.g. by login, creating a task, and viewing a report. Security was strictly handled by carrying out Security Testing which involved enhancing vulnerability scans continuously, fake SQL injection, Cross-site Scripting (XSS) attacks and unauthorized access. Lastly, Performance Testing was done to monitor the response time and server performance under stress by simulating desired load conditions through the use of JMeter. Every Sprint was followed by a Sprint Review that showed the features that were already done to the stakeholders and were validated and provided feedbacks that had direct influence on the priorities of the next Sprint.

5.1.5 Deployment CI/CD

The deployment was based on an effective Continuous Integration/Continuous Delivery (CI/CD) system, which was done with GitLab CI or GitHub Actions. This pipeline outlined a series of automated tasks: a code change (git push) was automatically followed by the running of Automated Tests (Unit/E2E), the building of a Docker Image Build, the pushing of the Image to a Registry and the last step of Automated Deployment. The application was based on Containerization with Docker to be capable of being consistent and portable in all of the settings development, staging and production. The end product was deployed to Cloud Infrastructure offered by AWS, which included the usage of such services as ECS (Elastic Container Service) to achieve a smooth container orchestration, and RDS (Relational Database Service) to have a stable PostgreSQL hosting.

Final Product Interface - Homepage:

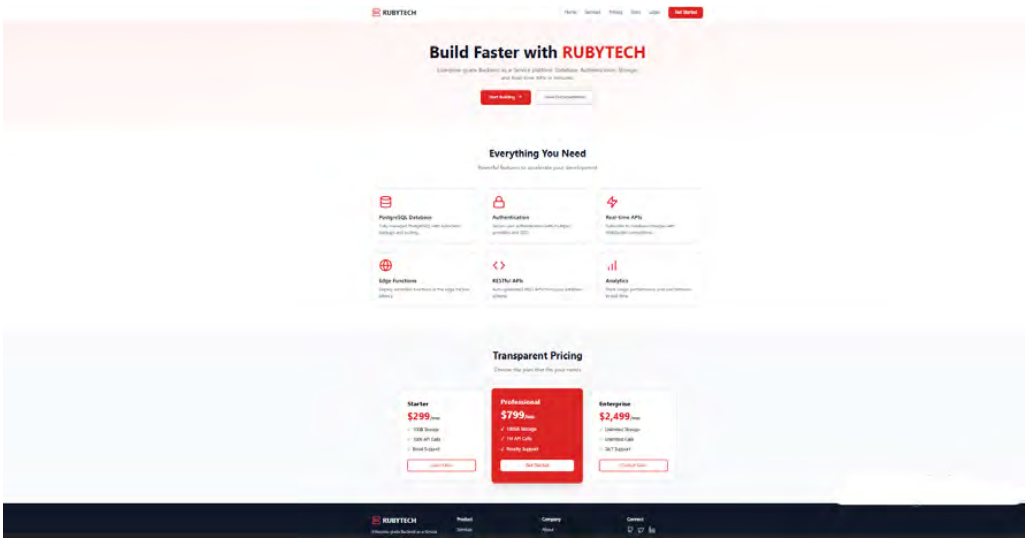


Figure 5.4: Homepage Interface of Final Product

Final Login Page UI:

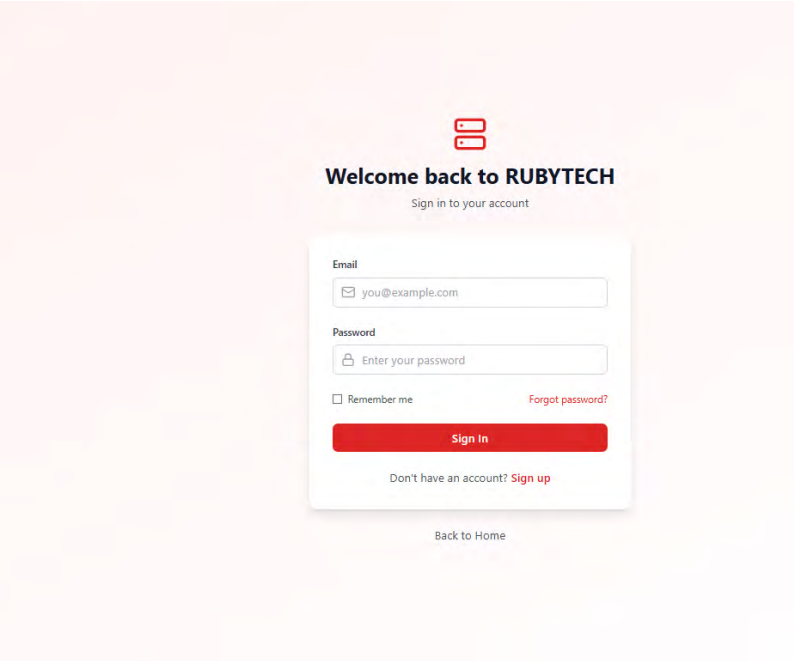


Figure 5.5: Login Interface of Final Product

The Agile Scrum process and MERN stack provided a scalable, secure, and completed project management platform ready for enterprise use.

Chapter 6

Future Plans

Although the internship project was able to provide a viable and secure system of a project management, some improvements and extensions of the features have been found to enhance the scalability of the system, its usability and the overall effectiveness that the system has within the enterprise settings.

6.0.1 Feature Expansion

To improve the system's flexibility and competitiveness, the following features are proposed for future releases:

- **Notification System:** Integration of real-time notifications via email, SMS, and in-app alerts to improve task tracking and collaboration.
- **Gantt Chart Module:** Adding Gantt-based project visualization for better timeline forecasting and task dependency tracking.
- **Role Customization Engine:** Allowing administrators to define and customize user roles and access permissions beyond preset options.
- **Third-Party Integration:** Expanding API support to integrate with tools like Slack, Trello, Google Calendar, and Microsoft Teams.
- **AI-Based Suggestions:** Leveraging machine learning models to predict task delays and suggest workload redistribution.

6.0.2 Technical Upgrades

To support wider adoption and maintain high performance under heavy load, the following technical improvements are under consideration:

- **Horizontal Scaling with Kubernetes:** Deploying containers via Kubernetes for better load balancing and failover management.
- **Advanced Monitoring with Prometheus & Grafana:** Introducing system health dashboards with alerting mechanisms to monitor uptime, latency, and server metrics.
- **Multi-Tenant Architecture:** Redesigning the backend structure to support multi-organization deployment from a single codebase.

- **ElasticSearch Integration:** Enhancing search performance across large datasets through full-text indexing and relevance-based queries.

6.0.3 Security Reinforcements

Cybersecurity will remain a top priority. Future releases will include:

- Two-Factor Authentication (2FA) for user logins
- Geo-fencing and IP whitelisting to restrict system access
- Encrypted audit logs and role-based log access filters
- End-to-end encryption of critical communications

6.0.4 Usability Enhancements

End-user experience will be improved with:

- Dark Mode and Accessibility Features
- Drag-and-drop task editing with real-time sync
- Inline comments and user tagging for better collaboration
- Mobile-responsive interface for cross-device usage

These features are part of the company long-term roadmap: to bring an all-inclusive, enterprise-ready, smart, and safe software platform that will bring digital transformation at a grand scale.

Chapter 7

Conclusion

The internship was a three-month program called Building Scalable and Secure Software Solutions to Empower Modern Digital Transformation, and it took place at RubyTech, October 1, 2024, through September 30, 2025, the activities in this report are up to January 27, 2025. The experience that I have acquired in these first months in the field of scaling and robust software system creation is invaluable. The primary one was the Agile Scrum development cycle that was the central part of my work as I applied theoretical knowledge of software engineering to develop a large-scale project management platform on the MERN stack (Node.js and React). The given iterative approach enabled me to continuously enhance the APIs, run hybrid databases (PostgreSQL / MongoDB), and transfer the application to AWS. The key milestones are the delivery of a production-level and fully functional platform which has a secure authentication, multi-role access control, Kanban task tracking, real-time reporting, and an in-depth audit trail. Also, the implementation of the new technologies, like Redis caching, indexing of databases, and the performance testing with JMeter helped to make the system serviceable, high-performing, and safe under the load. Besides the technical performance, the internship has provided me with the fundamental professional skills, including collaborative development, disciplined code reviews, agile communication, and skills to plan tasks with the assistance of such tools as Jira and Trello. Lastly, the experience has expressly trained me with the information I need to create secure, scalable, and maintainable systems, and effectively readies me to join full-time software engineering work.

Bibliography

[1] B. W. Boehm, "A spiral model of software development and enhancement," *ACM SIGSOFT Software Engineering Notes*, vol. 11, no. 4, pp. 14–24, 1988.

[2] J. Dean and S. Ghemawat, "Mapreduce: Simplified data processing on large clusters," *Proceedings of the 6th conference on Symposium on Operating Systems Design and Implementation*, pp. 137–150, 2008. [Online]. Available: https://dl.acm.org/doi/10.5555/1396636.1396642.

[3] S. Newman, "Building microservices: Designing fine-grained systems," *O'Reilly Media*, 2015.

[4] W. Stallings, "Cryptography and network security: Principles and practice," *Pearson*, 2017.

[5] Atlassian, "Agile project management with kanban," *Atlassian*, 2022, Available online: https://www.atlassian.com/agile/kanban.