TRIBHUVAN UNIVERSITY INSTITUTE

OF ENGINEERING PURWANCHAL

CAMPUS DHARAN

"Multiplayer Ping Pong game "

A COURSE PROJECT SUBMITTED TO THE DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE PRACTICAL COURSE ON
OBJECT ORIENTED PROGRAMMING [CT 451]

Submitted By:
Sujan Adhakari(PUR079BCT075)
Rabin Poudel(PUR079BCT056)
Sandip Rai(PUR079BCT086)
Samir Acharya(PUR079BCT070)

Submitted to:
Department of Electronics and Computer Engineering,
Purwanchal Campus Institute of Engineering,
Tribhuvan University
Dharan, Nepal

Ashad,2081

Abstract

The "PingPongGame" project seeks to create an engaging and dynamic two-player ping pong game using the C++ programming language and the Raylib library for game development. This project focuses on delivering a responsive and enjoyable gaming experience, with smooth animations and sound effects to enhance gameplay. The game features an AI opponent, player controls, and score tracking, ensuring an immersive and competitive environment. Hosted locally, the game will be developed over a two-week timeline, emphasizing design, coding, testing, and iteration. Success will be measured by the game's playability, stability, and user satisfaction.

Acknowledgment

We would like to express our sincere gratitude to our mentors and instructors for their unwavering support and guidance throughout the development of this PingPong Game project. Their expertise in C++ programming and game development has been pivotal in steering this project towards success.

We also extend our thanks to the Raylib community for providing the tools and resources that made the creation of this game possible. Their contributions have been invaluable.

Our heartfelt appreciation goes to our team members for their dedication and hard work in planning, developing, and testing the PingPong Game. Their collaboration and effort have been crucial to the project's progress.

Finally, we are grateful to our friends and family for their continuous encouragement and support. Their belief in us has been a constant source of motivation, driving our commitment to achieve our project goals.

Thank you to everyone who has contributed to the development of this project.

1 Introduction

The "LibBookAPI" project is an ambitious initiative aimed at developing a comprehensive and scalable API to provide access to a library of books stored in a database or file system. In an era where information accessibility and digital management are paramount, this project seeks to bridge the gap by offering a robust solution for managing and retrieving book information.

The primary objective of this project is to create an API that is accessible to a wide range of users, including developers, bookstores, online retailers, and individuals who require a reliable system for book management. By leveraging the powerful capabilities of C++ and the framework Drogon, the API will ensure high performance and flexibility. Additionally, hosting the API on Azure cloud services guarantees scalability, reliability, and broad accessibility.

The target audience for the LibBookAPI spans various sectors, from educational institutions and developers learning about API development to commercial entities like bookstores and online retailers looking to enhance their book management systems. By providing comprehensive access to book data and enabling efficient book management operations, LibBookAPI aims to become an indispensable tool in the digital ecosystem.

This proposal outlines the key aspects of the project, including its objective, description, target audience, technologies used, timeline, potential challenges, and success criteria. With a clear plan and structured approach, we aim to deliver a high-quality, functional, and reliable API that meets the needs of our users and stakeholders.

1.1 Objective

1. **Develop a Comprehensive API:**
   i.Create an API that provides detailed information about books stored in a database or file system.
   ii.Ensure the API supports key functionalities such as retrieving book details, searching for books by various criteria, and managing the book collection through CRUD operations.

2. **Ensure Usability and Accessibility**:
   i.Design the API to be user-friendly and accessible to a broad audience, including developers, bookstores, online retailers, and individuals.
   ii.Implement robust documentation to facilitate easy integration and usage by end-users.

3. **Leverage Modern Technologies:**
   i.Utilize the C++ programming language along with the Drogon framework to develop the API.
   ii.Host the API on Azure cloud services to take advantage of its scalability, reliability, and global accessibility.

4. Enable Efficient Testing and Development:
   i. Create a test environment to allow developers to test the API efficiently.
   ii.Ensure the API is designed with testing in mind, providing clear endpoints and predictable responses for automated and manual testing.

5. Achieve High Performance and Scalability:
   i.Optimize the API for high performance, ensuring quick response times and efficient handling of large volumes of data.
   ii.Design the system to scale seamlessly with increased load and usage, leveraging Azure's cloud infrastructure.

6. Meet Project Timeline:
   i.Complete the project within the proposed two-week timeline, ensuring all major features are developed, tested, and deployed.
   ii.Follow a structured development plan with clear milestones to track progress and ensure timely completion.

7. Address Potential Challenges:
   i.Identify and mitigate potential challenges related to using older framework like Drogon.
   ii.Ensure smooth deployment and operation on Azure cloud services, addressing any configuration and scaling challenges.

By achieving these objectives, the LibBookAPI project aims to deliver a robust, reliable, and user-friendly API that meets the diverse needs of its target audience.

2 Existing System

Current library and bookstore management systems are often outdated, relying on manual record-keeping or legacy software that lacks modern features. These systems suffer from limited accessibility, inefficient search capabilities, scalability issues, and poor integration with other technologies. Additionally, they are not optimized for remote access or cloud integration, resulting in performance bottlenecks and higher maintenance costs. These limitations underscore the need for a modern solution like LibBookAPI, which aims to provide a scalable, efficient, and accessible API for managing book collections using advanced C++ frameworks and azure cloud services.

3 Proposed System

1. **Modern API**: Utilizes C++ with Drogon framework.
2. **Cloud Hosting**: Hosted on Azure for scalability and reliability.
3. **Comprehensive Features**: Supports CRUD operations and advanced book search.
4. **User-Friendly**: Designed for ease of use and integration.
5. **High Performance**: Optimized for quick response times and efficient data handling.

This system aims to address existing limitations with enhanced accessibility, scalability, and robust functionality for managing book collections effectively.

4 Methodology

4.1 Development Tools

- **Programming Language**: C++
- **IDE**: VISUAL STUDIO CODE, CLION
- **Libraries/framework**: Drogon

4.2 Development Process

- **Requirement Analysis**: The objective here is to gather and document comprehensive requirements for LibBookAPI, encompassing both functional and non-functional aspects. This involves engaging stakeholders to understand their needs and expectations, prioritizing features based on importance and feasibility, and creating detailed documentation to guide development.
- **Design Phase:** This phase focuses on translating the gathered requirements into a well-defined API architecture and technical specifications. Activities include defining API endpoints, designing an efficient database schema for book storage, planning for scalability with Azure cloud services integration, and producing a detailed design document as a blueprint for implementation.
- **Implementation:** The goal is to transform the design specifications into functional components of LibBookAPI. This involves writing code using C++ with Drogon framework, implementing CRUD operations for book management, integrating search functionalities, and ensuring robust error handling. Adherence to coding standards, version control practices, and regular code reviews are critical during this phase to maintain code quality and consistency.
- **Testing:** This phase verifies the functionality, reliability, and performance of LibBookAPI. Activities include conducting unit tests to validate individual components, integration tests to ensure seamless interaction between modules, and system tests to evaluate overall API performance under various conditions. Performance testing is also conducted to ensure the API meets expected response times and handles concurrent users effectively.
- **Deployment:** The objective is to deploy LibBookAPI into a production environment on Azure cloud services. This involves setting up deployment environments, configuring Azure for optimal performance and scalability, deploying API components, and closely monitoring the deployment process to detect and address any issues promptly, ensuring a smooth transition to production.
- **Testing:** This phase aims to provide comprehensive resources for users and developers to understand and effectively use LibBookAPI. Activities include creating detailed API documentation with usage guidelines, endpoint descriptions, and examples, developing user guides and tutorials for integration and usage, and conducting training sessions or workshops to facilitate onboarding and maximize API utilization.
- **Maintenance and Support**: The goal is to ensure ongoing functionality, performance, and user satisfaction with LibBookAPI post-deployment. This involves monitoring API performance and availability, proactively addressing reported issues and bugs through support channels, planning and implementing updates and enhancements based on user feedback and technological advancements, and providing continuous support to users to resolve queries and maintain high satisfaction levels.

5 Project Scope

- **API Functionality:**

  1. Develop an API that provides comprehensive access to a library of books stored in a database or file system.

  2. Implement CRUD operations for managing book data, including creating, reading, updating, and deleting book records.

  3. Integrate advanced search functionalities to allow users to search for books based on various criteria such as title, author, genre, etc.

  4. Ensure the API supports efficient data retrieval and manipulation to enhance usability for developers and end-users.

- **Assumptions:**
  1. Assumptions will be detailed during the requirement analysis phase.

6 Project Schedule

1. Timeline

Phase	Start Date	End Date
Requirement Analysis	2081-04-01	2081-04-03
System Design	2081-04-04	2081-04-07
Implementation	2081-04-08	2081-04-09
Testing	2081-04-10	2081-04-11
Deployment	2081-04-12	2081-04-13
Maintenance	2081-04-13	---

Table 1: Project Schedule