Sprint 4 Technical Requirements Documentation
T03
MyGame
Sports League App

Harlan Ferguson 101133838
Owen Beattie 101379668
Richard Jordan Wilson 101370635
Raleigh Desmond 101374701

Technical Requirements Document

1. **Overview of Technologies**

Category	Technology	Description
Database	MongoDB	NoSQL database used for storing app data
Backend Programming	Node.js	JavaScript runtime for building server-side applications
Frontend Framework	React	JavaScript library for building user interfaces
Server	Express.js	Web application framework for Node.js
CSS Framework	Bootstrap/Material-UI	Styling framework for responsive design
Deployment Platform	Azure	Cloud platforms for hosting applications
Version Control	Git/GitHub	Tools for source code management

2. Pros and Cons Analysis

MongoDB

Pros:

- Schema-less: Provides flexibility in storing different types of data.
- Scalable: Good for applications expecting growth in user base and data.
- JSON Data Format: Easy integration with JavaScript-based stack.

Cons:

- Limited ACID Transactions: Not ideal for applications requiring complex transactions.
- Memory Intensive: Can require significant system resources.

Node.js

Pros:

- High Performance: Non-blocking I/O operations enhance performance.
- Large Ecosystem: Extensive npm packages available for various functionalities.
- Unified JavaScript Development: Simplifies development process (same language for front and back end).

Cons:

- Callback Hell: Managing asynchronous code can be challenging.
- Not Suitable for CPU-Intensive Tasks: Can be a bottleneck for heavy computations.

React

Pros:

- Component-Based: Facilitates reusable UI components.
- Strong Community: Well-supported by developers and companies.
- Virtual DOM: Efficient updates and rendering.

Cons:

- Learning Curve: JSX and component lifecycle can be complex for beginners.
- Only the UI Layer: Requires additional libraries for state management, routing, etc.

Express.js

Pros:

- Minimal and Flexible: Simple to use and easy to extend with middlewares.
- Fast Development: Streamlines backend development.
- Wide Range of HTTP Utilities: Facilitates robust API development.

Cons:

- Middleware Heavy: Reliance on multiple middlewares can become complex.
- No Conventions: Lack of strict conventions can lead to unstructured code.

CSS Framework (Bootstrap/Material-UI)

- Pros:
- Responsive Design: Easy to create mobile-friendly interfaces.
- Pre-designed Components: Accelerates UI development.
- Customizable: Themes can be tailored to fit brand identity.
- Cons:
- Size: Can be heavy, affecting load times.
- Overreliance: Might hinder CSS skill development.

Deployment Platform (Azure)

Pros:

- Scalability: Easy to scale resources as per demand.
- Managed Services: Reduces the hassle of server management.
- Integration Options: Supports a wide range of tools and services.

Cons:

- Cost: Can become expensive with increased usage.
- Learning Curve: Initial setup and management can be complex.

Git/GitHub

Pros:

- Version Control: Keeps track of changes and facilitates collaboration.
- Community and Support: Large community and extensive documentation.
- Integration: Easily integrates with various CI/CD tools.

Cons:

- Complexity: Can be overwhelming for beginners.
- Merge Conflicts: Requires careful management of branches and merges.

Specialized Learning Plan for MyGame

Richard - MongoDB (Database Management)

- **Focus Areas**: Database design, data modeling, MongoDB queries, indexing.
- **Learning Resources**: MongoDB University courses, official documentation, hands-on projects.
- **Goal**: Master database management, ensuring efficient data storage and retrieval.

Raleigh - Node.js and Express.js (Backend Development)

- **Focus Areas**: Server-side logic, API development, integration with MongoDB.
- **Learning Resources**: Node.js and Express.js documentation, online courses, building RESTful APIs.
- **Goal**: Develop robust server-side applications and APIs, ensuring smooth data integration.

Owen - React and Frontend Styling (Frontend Development)

- **Focus Areas**: React components, hooks, state management, integration with backend APIs, CSS frameworks (Bootstrap/Material-UI) for styling.
- **Learning Resources**: React documentation, interactive tutorials, CSS framework tutorials, building dynamic user interfaces.
- **Goal**: Master front-end development and UI styling, creating intuitive and responsive interfaces.

Harlan - Git/GitHub and Azure Deployment (Version Control and Deployment)

- **Focus Areas**: Version control best practices, branch management, conflict resolution, deployment strategies, Azure services.
- **Learning Resources**: Git tutorials, GitHub documentation, Microsoft Azure courses, deploying applications.
- **Goal**: Enhance skills in version control and cloud deployment, ensuring efficient code management and application availability.

Collaborative Learning and Integration

- **Cross-Training Sessions**: Regular sessions where each member teaches others about their specialty to foster a comprehensive understanding of the stack.
- **Joint Projects**: Collaborative work on smaller projects or components of the main application.
- **Code Reviews**: Regularly reviewing code to provide feedback, share best practices, and maintain quality.

Continuous Skill Enhancement

- **Keeping Up-to-Date**: Staying informed about the latest developments through blogs, online communities, and webinars.
- **Advanced Topics**: Exploring more complex aspects in their areas of expertise.
- **Mentorship**: Seeking guidance from industry professionals for advanced skill development.

Outcome and Evaluation

- **Project Contribution**: Each member applies their specialized knowledge to contribute to a robust MERN stack application.
- **Peer Feedback**: Regular feedback from team members to assess progress and real-world skill application.
- **Self-Evaluation**: Regular self-assessment to identify improvement areas and further learning needs.

Conclusion

This document outlines a detailed and strategic learning plan for each team member of the "MyGame" project. Through focused areas of study, collaborative efforts, and continuous skill enhancement, the team is equipped to address the technical requirements of building a comprehensive MERN stack application. The plan is dynamic and will be adjusted as the project progresses to accommodate new learnings and challenges encountered.