REQUIREMENTS

Udara Liyanage

Version 1.0

09 Dec 2024

No	Requirement	Status
1	Use a programming language and framework of your choice to create the project. a. Use of Java 17 and Spring Boot framework is an added bonus	Done
2	Configurable to run in multiple environments	Done
3	Use a package manager to manage project dependencies.	Done
4	Implement proper data validation and error handling.	Done
5	Use a database to store borrower and book data. a. Justify your choice of database **Answer** Using an H2 database for development has its advantages and limitations, and the choice of database for production depends on various factors. Here's a justification for using a different database for production 1. Scalability - While H2 is suitable for development due to its lightweight nature and ease of setup, it might not be the best choice for production environments requiring high scalability. Consider databases like PostgreSQL or MySQL, which are known for their ability to handle large-scale applications and heavy workloads effectively. 2. Performance - Production databases need to be optimized for performance, especially in scenarios with a large number of concurrent users or complex queries. While H2 performs well for small-scale applications, other databases like PostgreSQL, MySQL, or even commercial options like Oracle or SQL Server, offer better performance tuning options and optimizations for production use.	

	3. Durability and Reliability - H2 is an in-memory database, which means that data is lost when the application is shut down unless configured otherwise. In production environments, durability and data reliability are crucial. Databases like PostgreSQL and MySQL provide mechanisms for data durability through features like transaction logs, replication, and backups. 4. Feature Set - Depending on the requirements of your application, you might need features like full-text search, JSON support, spatial data support, etc. Consider databases that provide the required feature set while also aligning with your application's architecture and development stack. 5. Community and Support - Consider the availability of community support, documentation, and expertise for the chosen database. Databases with large communities, active development, and comprehensive documentation make it easier to troubleshoot issues, find solutions, and stay updated with best practices. In summary, while H2 is suitable for development and testing due to its lightweight nature and ease of use, for production environments, consider databases like PostgreSQL, MySQL, or others based on factors such as scalability, performance, durability, feature set, and community support.	
6	Implement REST API endpoints for each action mentioned above.	Done
7	Multiple books with the same ISBN number should be registered as books with different ids.	Done
8	Ensure that no more than one member is borrowing the same book (same book id) at a time.	Done
9	Provide clear documentation for how to use your API	Done

10

Provide documentation of all your assumptions for any requirements that are not explicitly stated in this task.

Answer

Documentation of assumptions is crucial for ensuring clarity and alignment between stakeholders, developers, and testers. Here are some assumptions that might have been made during the implementation of the task,

1. Database Choice - The assumption could be that the choice of database for production would differ from the one used in development. This assumption is based on best practices in software development, where lightweight databases like H2 are often used for development and testing, while more robust databases like PostgreSQL or MySQL are preferred for production.

2. CI/CD Pipeline Configuration - It might be assumed that a CI/CD pipeline needs to be set up for automated testing, building, and deployment of the application. This assumption aligns with modern software development practices, where continuous integration and continuous deployment pipelines streamline the development workflow.

3. Containerization with Docker - The assumption could be that the application would be containerized using Docker to ensure consistency across different environments and facilitate easier deployment. Docker containers provide isolation and portability, making them a common choice for packaging applications.

4. Kubernetes for Orchestration - Assuming that Kubernetes would be used for orchestrating and managing the Docker containers in a production environment. Kubernetes simplifies the deployment, scaling, and management of containerized applications, providing features like automatic scaling, service discovery, and rolling updates.

5. Testing Frameworks - It could be assumed that testing frameworks like JUnit and Mockito are used for writing unit tests, while tools like

MockMvc and TestRestTemplate are used for integration testing of REST APIs in Spring Boot applications.

6. Exception Handling - An assumption might be that the application includes robust exception handling mechanisms to gracefully handle errors and failures, ensuring a better user experience and system stability.

7. Security Considerations - Assumptions could be made regarding security measures such as input validation, authentication, authorization, and encryption to protect sensitive data and prevent security vulnerabilities.

8. Logging and Monitoring - It might be assumed that the application includes logging and monitoring functionalities to track application behavior, identify errors, and gather performance metrics for troubleshooting and optimization purposes.

9. Documentation Standards - Assuming adherence to documentation standards such as Javadoc for documenting code, README files for project setup and usage instructions, and architecture diagrams for depicting system design and component interactions.

10. Deployment Environment - Assuming deployment to a cloud provider like AWS, Azure, or Google Cloud Platform, and configuring necessary services like databases, containers, load balancers, and auto-scaling groups based on the application's requirements and scalability needs.

By documenting these assumptions, it helps ensure that all stakeholders have a clear understanding of the project scope, requirements, and implementation decisions, leading to a more cohesive and successful project outcome.