User-Management System

In implementing the CRUD operations for User Management, I followed a structured approach using Java with Spring Boot for the backend and Html, CSS, Bootstrap for the frontend. Here's a summary of my approach:

1.Backend Development:

- Created a User entity with appropriate fields.
- Implemented for CRUD operations with proper input validation.
- Utilized Spring Boot for easy configuration and dependency management.
- Implemented services to handle business logic and interact with the database.
- Ensured security practices, such as input validation and password hashing.
- Used HTTP methods and status codes correctly.

2.Frontend development(HTML, CSS, Bootstrap, thymeleaf):

- Build a registration form with validation.
- Submit data to the Spring Boot backend via Thymeleaf templates.
- Use Thymeleaf templates to display a user list from the backend.
- Apply client-side validation and use Thymeleaf for updates.
- Add delete buttons in Thymeleaf templates to remove users via Spring Boot.

3.Documentation:

- Provide clear usage instructions for your HTML, CSS, and Thymeleaf frontend with Spring Boot backend.
- Ensured that the codebase followed best practices and was well-structured.
- Set up a GitHub repository for version control and tracking progress.

Enhancements (if more time):

- Improve styling with advanced CSS.
- Implement Thymeleaf layouts for consistent design.
- Consider session management for authentication.

Challenges Faced:

- **Authentication and Authorization**: Implementing user authentication and authorization would have added an additional layer of complexity but is essential for real-world applications to secure user data.

- **Complex Validation Rules**: Depending on requirements, user input validation can become complex. Handling various validation rules for fields like email, password, and mobile numbers required careful consideration.

- **Display Error Message:** Providing clear and meaningful error messages that guide users on how to resolve the issue. Validating user inputs and displaying errors next to the corresponding fields in forms can be complex, especially when dealing with complex validation rules