# 

# 

# **Technical**

# **Specification**

# **Document**

## 

Prepared by : Ahmad Bahauddin

[**Technical 1**](#_mc5ho5c2n6nv)

[**Specification 1**](#_335gythl1psj)

[**Document 1**](#_bfc223mo73ce)

[1. Overview 3](#_9qyan0dfojwv)

[2. System Architecture 3](#_nvwsbq2rekv3)

[2.1 Architecture Overview 3](#_lty5j6p5aigi)

[2.2 Microservices 3](#_7gccy4s0dwsh)

[3. Key Functionalities 3](#_yvodme4z2fcn)

[3.1 Customer Management 3](#_ntq7bz2qbmmm)

[3.1.1 Customer Registration 3](#_bh25muqw42fp)

[3.1.2 Customer Details 3](#_cycws68872ie)

[3.2 Account Management 4](#_4rhatlyiqr9f)

[3.2.1 Account Creation 4](#_2dvvvrqj6gei)

[3.2.2 Account Transactions 4](#_5fkrn3l6srbp)

[3.2.3 Account Deletion 4](#_f7bsm2pl68bj)

[3.2.4 Account Details 4](#_rlstexwesdr)

[4.3 Reporting 4](#_sbilqaktm8t5)

[4.3.1 Transaction History 4](#_48eijbsxne2f)

[5. Technologies and Dependencies 4](#_nv3ji3lnf8hs)

[6. Project Structures 5](#_mtk2q8eq5dow)

[7. Development Environment 8](#_3tjp37jqmxv4)

[7.1 Visual Studio Code 8](#_vdk8jizaptgv)

[7.2 Spring Tool Suite 8](#_d47wooiu11sb)

[7.3 Mysql WorkBench 9](#_j2273dw174x8)

[7.4 Postman 9](#_8n3y0o70ll6c)

[8. Testing 9](#_v35aehnl7b32)

[8.1 Unit Testing 9](#_hl86vsux33m3)

[8.2 Integration Testing 9](#_purj7363aw8z)

[8.3 Security Testing 9](#_2pgmp8kv1qlf)

[9. Conclusion 9](#_vqa9qmug7ty2)

## 

## 

## 

## 

## **1. Overview**

The purpose of this technical specification document is to outline the key features and requirements for the development of a comprehensive banking application. This document serves as a guide for the development team to ensure the successful implementation of a secure, scalable, and user-friendly bank application.

## **2. System Architecture**

## **2.1 Architecture Overview**

The application will follow a multi-tier architecture, comprising a presentation layer, business logic layer, and data access layer. The chosen technology stack includes Spring Boot for the backend and React JS for the frontend.

### **2.2 Microservices**

Consideration should be given to implementing microservices architecture for better scalability, maintainability, and fault isolation.

## **3. Key Functionalities**

### **3.1 Customer Management**

#### 3.1.1 Customer Registration

* Implement a secure customer onboarding process, collecting essential details such as name and assign them with auto generated ID

#### 3.1.2 Customer Details

* Develop a comprehensive customer profile with the ability to display customer details

### **3.2 Account Management**

#### 3.2.1 Account Creation

* Allow customers to create various account types (savings, checking, etc.) with unique account numbers.

#### 3.2.2 Account Transactions

* Facilitate standard banking transactions, including deposits, withdrawals, and fund transfers.

#### 3.2.3 Account Deletion

* Implement account deletion function for the customer to deactivate the account

#### 3.2.4 Account Details

* Allow customer to retrieve all accounts that belongs to them

### **3.3 Reporting**

#### 4.3.1 Transaction History

* Provide a detailed transaction history for customers to track their financial activities.

## **4. Technologies and Dependencies**

1. React JS as the frontend web service with the usage of axios for api request
2. Spring Boot Framework for REST API application structure
3. MySQL as the database management
4. Spring Data JPA as data access
5. Postman as Unit Testing Integration

## **5. Project Structures**

**5.1 POM.xml Overviews**

| <!-- Dependencies -->  <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-starter-data-jpa</artifactId>  </dependency>  <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-starter-web</artifactId>  </dependency>  <dependency>  <groupId>com.mysql</groupId>  <artifactId>mysql-connector-j</artifactId>  <scope>runtime</scope>  </dependency>  <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-starter-test</artifactId>  <scope>test</scope>  </dependency>  <dependency>  <groupId>org.projectlombok</groupId>  <artifactId>lombok</artifactId>  <optional>true</optional>  </dependency>  <dependency>  <groupId>com.googlecode.json-simple</groupId>  <artifactId>json-simple</artifactId>  <version>1.1</version>  </dependency>  <dependency>  <groupId>commons-lang</groupId>  <artifactId>commons-lang</artifactId>  <version>2.6</version>  </dependency> |
| --- |

**5.2 Project Tree**

| Com  +- baha  | +- bankapp  | | +- common  | | | +- BaseDto.java  | | | +- BaseEntity.java  | | | +- BaseValidator.java  | | | +- CommonConstant.java  | | | +- CommonUtils.java  | | | +- StringUtil.java  | | +- controller  | | | +- AccountController.java  | | | +- CustomerController.java  | | +- dto  | | | +- AccountDto.java  | | | +- CustomerDto.java  | | +- exception  | | | +- AccountInsufficentBalanceAdvice.java  | | | +- AccountInsufficentBalanceException.java  | | | +- AccountNotActiveAdvice.java  | | | +- AccountNotActiveException.java  | | | +- AccountNotFoundAdvice.java  | | | +- AccountNotFoundException.java  | | | +- CustomerNotFoundAdvice.java  | | | +- CustomerNotFoundException.java  | | +- model  | | | +- Account.java  | | | +- Customer.java  | | +- repository  | | | +- AccountRepository.java  | | | +- CustomerRepository.java  | | +- services  | | | +- impl  | | | | +- AccountServiceManager.java  | | | | +- CustomerServiceManager.java  | | | +- AccountServiceManager.java  | | | +- BaseService.java  | | | +- CustomerServiceManager.java  | | +- validator  | | | +- CustomerValidator  | | +- BankApplication.java |
| --- |

**5.3 Application.properties**

| spring.jpa.hibernate.ddl-auto=update  spring.datasource.url=jdbc:mysql://localhost:3306/bankdb  spring.datasource.username=root  spring.datasource.password=password  spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver |
| --- |

**5.4 Database Architecture**

bankdb database

1. tcustomer Table
2. taccount Table

| tcustomer Table   | **Field** | **Type** | **Null** | **Key** | **Default** | | --- | --- | --- | --- | --- | | cust\_no | varchar(255) | NO | PRI | NULL | | created\_by | varchar(20) | NO |  | NULL | | created\_dt | datetime(6) | NO |  | NULL | | updated\_by | varchar(20) | NO |  | NULL | | updated\_dt | datetime(6) | NO |  | NULL | | cust\_name | varchar(255) | NO |  | NULL | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

| taccount Table   | **Field** | **Type** | **Null** | **Key** | **Default** | | --- | --- | --- | --- | --- | | acc\_no | varchar(255) | NO | PRI | NULL | | created\_by | varchar(20) | NO |  | NULL | | created\_dt | datetime(6) | NO |  | NULL | | updated\_by | varchar(20) | NO |  | NULL | | updated\_dt | datetime(6) | NO |  | NULL | | acc\_ball | decimal(38,2) | NO |  | NULL | | acc\_status | int | NO |  | NULL | | acc\_type | int | NO |  | NULL | | cust\_no | varchar(255) | NO |  | NULL | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

### **6. Development Environment**

### **6.1 Visual Studio Code**

* Use for designing the user interface using React JS
* Use axios for api request management

### **6.2 Spring Tool Suite**

* Use for designing the REST API Structure and its application layer
* Use Spring and Springboot Framework for designing the REST API and its management
* Use for database management and access

### **6.3 Mysql WorkBench**

* Use for database management and access

### **6.4 Postman**

* Use for REST API testing and error handling message

## **7. Testing**

### **7.1 Unit Testing**

* Develop and execute comprehensive unit tests for each component of the application.

### **7.2 Integration Testing**

* Conduct thorough integration testing to validate the seamless interaction between different modules.

### **7.3 Security Testing**

* Perform regular security assessments, including penetration testing, to identify and address vulnerabilities.

## **8. Conclusion**

This technical specification document serves as a roadmap for the development team, providing a clear outline of the features, security measures, and performance considerations for the banking application. Adherence to these specifications will result in a robust, secure, and user-friendly banking system.