Project Plan

1. Introduction

The Bank Management System is a web-based application that facilitates efficient management of customer accounts, transactions, and branches. It leverages a MySQL database for backend operations and React for the frontend. This document outlines the project's planning details, including deliverables, process models, and resource allocation strategies.

2. Deliverables of the Project

- **User Registration Module**: Secure customer registration with validation.
- **Account Management Module**: Features for opening, updating, and closing customer accounts.
- **Transaction Management Module**: Support for deposits, withdrawals, and fund transfers.
- **Report Generation Module**: Generate account statements and transaction histories.
- **Frontend User Interface**: React based UI for intuitive user interaction.
- **Backend Database Schema**: MySQL tables for customers, accounts, and transactions.

3. Process Model

The project follows the Waterfall Model, which is a linear and sequential approach suitable for projects with well-defined requirements and minimal expected changes. The system's design can be fully planned before development, making it ideal for this Bank Management System project. The stages include:

1. **Requirements Analysis**
   In this stage, the project requirements are gathered and documented. For the Bank Management System, the requirements focus on the functionalities such as account management, transactions, and secure access.

2. **Architecture**
   The high-level system architecture is defined, specifying the major components and their interactions. This includes selecting the technology stack (MySQL, React/Next.js), defining the API layers, and structuring the database tables and schema.

3. **System Design**
   Detailed designs are created for the database, UI wireframes, and API structure. This step outlines the logical flow for each feature, such as account opening, transaction management, and reporting.

4. **Implementation**
   The actual coding takes place in this phase. The frontend, backend, and database components are developed according to the specifications created during the design phase. Each module (e.g., user registration, fund transfer) is built and tested individually.

5. **Testing**
   Comprehensive testing is conducted to ensure that all functionalities work as expected. This includes unit testing for individual components, integration testing for combined modules, and system testing for the entire application.

6. **Deployment**
   The fully tested system is deployed to the production environment. Configuration and setup are handled to make the system accessible to end-users.

7. **Maintenance**
   After deployment, the system enters the maintenance phase, where any issues are addressed, and necessary updates are applied. This includes bug fixes, performance optimization, and the addition of new features as required.

4. Organization of Project

The project is structured into the following teams:

- **Frontend Development Team**: Responsible for creating user interfaces using React/Next.js.
- **Backend Development Team**: Handles database design and API development using Node.js.
- **Testing Team**: Ensures functionality, performance, and security.
- **Project Management**: Oversees timelines, resource allocation, and coordination.

5. Standards, Guidelines, Procedures

- **Coding Standards**: Follow JavaScript and SQL best practices.
- **Database Guidelines**: Use normalized tables to avoid redundancy.
- **Security Procedures**: Implement secure password storage (e.g., hashing), data encryption, and role-based access control.
- **Documentation Standards**: Use JSDoc for frontend and backend code documentation.

6. Management Activities

- Regular project meetings to review milestones and identify roadblocks.
- Documentation updates after each major development phase.
- Tracking and managing issues using a project management tool like Jira.

7. Risks

- **Data Breach Risk**: Potential security vulnerabilities in user data handling.
- **Performance Issues**: Possible delays with concurrent transactions.
- **Integration Challenges**: React/Next.js frontend may face issues interfacing with the backend.
- **Scope Creep**: Additional requirements might be requested during development.

8. Staffing

- **Frontend Developers**: 2 developers for building and maintaining UI components.
- **Backend Developers**: 2 developers for API and database management.
- **Database Administrator**: 1 for managing MySQL configurations and maintenance.
- **Quality Assurance (QA) Engineer**: 1 for testing and validation.
- All roles will be fulfilled by 2 members in the team.

9. Methods and Techniques

- **Agile Techniques**: Sprints for iterative development and feedback.
- **Testing Methods**: Unit testing for individual modules, integration testing, and user acceptance testing.
- **Version Control**: Git for source code management.

10. Quality Criteria/Assurance

- **Performance**: The system should support multiple users simultaneously with minimal lag.
- **Security**: All data should be encrypted, and access should be restricted based on roles.
- **Usability**: User interface should be intuitive, responsive, and accessible.

11. Work Packages

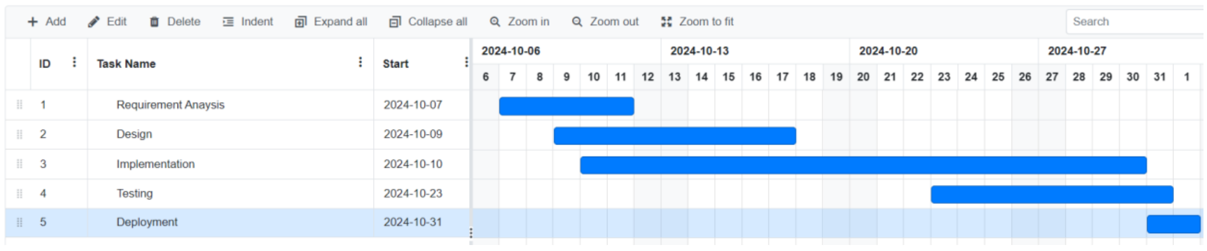
- **Frontend UI Development**: Design and implement customer and admin dashboards.
- **Backend API Development**: Create secure endpoints for account management and transaction handling.
- **Database Design and Integration**: Develop schema and manage data consistency.
- **Testing and Validation**: Comprehensive testing of functionalities and edge cases.

12. Resources

- **Development Tools**: Visual Studio Code, MySQL Workbench, Postman.
- **Project Management Tools**: Jira or Trello for tracking tasks.
- **Servers**: Deployment server (AWS or Azure) for hosting the web application.

13. Budget and Schedule

- **Budget**: Estimated budget for the project is $0, as this is an academic project and does not use cost-intensive frameworks or tools.
- **Schedule**:
    - Requirements Analysis: 1 week
    - Design: 1 week
    - Implementation: 4 weeks
    - Testing: 2 weeks
    - Deployment: 1 weeks



ID	Task Name	Start
1	Requirement Anaysis	2024-10-07
2	Design	2024-10-09
3	Implementation	2024-10-10
4	Testing	2024-10-23
5	Deployment	2024-10-31

14. Change Control Process

Changes to the project will be managed through a change request process:

1. **Submission of Change Request**: Any stakeholder can submit a change request.
2. **Impact Analysis**: Evaluate the impact on cost, schedule, and quality.
3. **Approval or Rejection**: Project manager reviews and approves/rejects the request.
4. **Implementation**: If approved, integrate the changes in a scheduled release.

15. Delivery Means

The final product will be delivered as a web-based application accessible through a browser. Documentation and user manuals will be included, and the system will be hosted on a cloud platform for accessibility.