Personal Finance Management System Project Report

1. Introduction

1.1. Project Overview

The Finance Management System is a comprehensive web application designed to provide essential banking services through a user-friendly and secure interface. The system allows users to manage their accounts, send money, view transaction histories, maintain wallet accounts, and securely manage user credentials. This system integrates modern web technologies, real-time data handling, and security protocols to ensure a robust banking experience.

2. System Features

2.1. Home

Functionality: The Home section serves as the dashboard, providing an overview of the user's financial activities, balance, recent transactions, and shortcuts to key actions (e.g., Send Money, View Wallet).
User Experience: Designed for quick access to critical information and easy navigation to other features.

2.2. Send Money

Functionality:
- Allows users to transfer funds to other accounts within the bank or to external accounts.
- Users can select recipients from saved contacts or manually enter new recipient details.
- The amount, currency, and method of transfer (such as instant transfer, scheduled payments) can be configured.
- Includes a review and confirmation step for transaction security.

Process Flow:
1. User selects recipient.
2. Enters amount and transfer details.
3. Confirms transaction.
4. Receives confirmation message and transaction ID.

2.3. Transaction History

Functionality:
- Displays a list of the user's past transactions, including details such as date, recipient, amount, and transaction status (completed, pending, failed).

- Users can filter transactions by date, type (debit/credit), or status.
- Export functionality (PDF, CSV) for record-keeping or tax purposes.

User Experience: Allows for seamless navigation and in-depth search of past financial transactions, with detailed information and downloadable records.

2.4. Wallet Account

Functionality:
- Provides users with a digital wallet where they can store funds.
- The wallet can be used for quick transactions without linking to the primary bank account for each payment.
- Users can add or withdraw money from the wallet, check wallet balance, and use the wallet for recurring payments or subscriptions.

Additional Capabilities:
- Integrates with payment systems for mobile top-ups, utility payments, or e-commerce.
- Tracks wallet usage and history.

2.5. User Account

Functionality:
- Manages user profile and settings.
- Allows users to update their personal information, such as email, phone, and address.
- Security options such as password change, two-factor authentication (2FA), and account recovery.

Security Considerations:
- Implements secure password hashing, email validation, and session management.
- 2FA and security questions for additional security.

2.6. Logout

Functionality:
- Logs the user out of the system and clears session data, ensuring no unauthorized access after the session ends.
- Provides a confirmation message upon successful logout.

3. System Architecture

3.1. MVC Architecture

The system follows the Model-View-Controller (MVC) design pattern, ensuring a clean separation between data (Model), UI (View), and business logic (Controller). This architecture allows for scalability and maintainability.

3.2. Technologies Used

Frontend:
- Angular: For dynamic, responsive UI components and seamless user interactions.
- HTML5/CSS3: For layout and styling.

- Bootstrap: For responsive and mobile-friendly design.

Backend:
- ASP.NET Core Web API: For RESTful services handling business logic and database operations.
- C#: Primary programming language for server-side logic.

Database:
- SQL Server: Stores all user data, transaction histories, and wallet information.
- Entity Framework Core: Manages database interactions using LINQ queries, providing a data access layer.

3.3. Database Structure

UserAccounts Table:
- Stores user information, including name, email, hashed password, phone number, and address.

Transactions Table:
- Records all financial transactions, including sender, recipient, amount, transaction type, and timestamp.

Wallets Table:
- Maintains wallet balances and transaction histories for each user.

4. System Flow

4.1. User Registration

1. User navigates to the registration page.
2. Inputs required information (name, email, phone, password).
3. Email validation and password strength verification are performed.
4. Upon successful registration, the user is redirected to the login page.

4.2. User Login

1. User inputs email and password.
2. Password is securely hashed and verified against the stored hash in the database.
3. On successful login, a session token is generated.
4. The user is redirected to the Home dashboard.

4.3. Sending Money

1. User selects recipient and inputs transfer details.
2. Transaction is validated (balance check, recipient verification).
3. Upon confirmation, the transaction is processed and recorded in the database.
4. The system updates the sender’s and recipient's account balances.

4.4. Viewing Transaction History

1. User navigates to the Transaction History section.
2. The system retrieves past transactions from the database.
3. Transactions are displayed in a table with filters for easy search.

5. Security and Data Integrity

5.1. Authentication and Authorization

Authentication: Handled through session tokens generated at login. Ensures that only authorized users can access the system.
Authorization: Role-based access control (RBAC) restricts certain functionalities to specific user roles (e.g., admin).

5.2. Data Encryption

All sensitive data (passwords, financial transactions) are encrypted in transit (SSL/TLS) and at rest.
Passwords are hashed using a secure hashing algorithm (e.g., bcrypt).

5.3. Session Management

Implements session timeout and automatic logout after a period of inactivity to enhance security.
Uses session tokens to maintain user login across multiple requests.

5.4. Input Validation

Server-side validation for all forms (e.g., registration, sending money) to prevent SQL injection, XSS attacks, and other vulnerabilities.
Email and password validation ensures proper formatting and strength before user account creation.

6. Testing and Quality Assurance

6.1. Unit Testing

Unit tests are implemented for critical business logic, such as transaction processing, wallet management, and user authentication. Tests ensure that the core functionalities work as expected under various conditions.

6.2. Integration Testing

Integration tests are conducted to verify that all system components (frontend, backend, database) work together seamlessly. Tests cover scenarios such as successful money transfers, registration/login flows, and transaction history retrieval.

6.3. User Acceptance Testing

User Acceptance Testing (UAT) is performed with real users to ensure that the system meets the requirements and provides a satisfactory user experience. Feedback from users is collected and used to improve the interface and functionality.

7. Conclusion

The Finance Management System provides a secure, scalable, and user-friendly platform for managing various banking operations. It ensures a seamless user experience while maintaining high standards of security and data integrity. The system is designed to handle real-time transactions, secure user authentication, and efficient wallet management, making it suitable for modern banking needs.