SMARTBANK : A UNIFIED BANKING SOLUTION FOR ORGANIZATIONS

Mrs.G.VENKATESHWARI,M.E,MBA
Assistant Professor: Department of Information Technology
Er. Perumal Manimekalai College of Engineering, Hosur, Tamil Nadu.

RAJ MOHAN.S
Department of Information Technology
Er. Perumal Manimekalai College of Engineering, Hosur, Tamil Nadu.

DINESH.M
Department of Information Technology
Er. Perumal Manimekalai College of Engineering, Hosur, Tamil Nadu.

HARISH.S
Department of Information Technology
Er. Perumal Manimekalai College of Engineering, Hosur, Tamil Nadu.

PANENDRA.V
Department of Information Technology
Er. Perumal Manimekalai College of Engineering, Hosur, Tamil Nadu.

ABSTRACT

Our banking application is a cross-platform solution designed to facilitate seamless corporate banking for businesses. It enables organizations to efficiently manage and perform transactions across multiple accounts within their network, streamlining financial operations and enhancing internal processes. This app ensures secure and convenient management of financial activities within a specific organization.

Keywords : Corporate banking, Cross-platform solution ,Multi-account management Transaction management, Financial management, Secure banking

1. Introduction

Modern corporate financial management requires secure, efficient, and compliant solutions for handling internal transactions. Traditional banking systems often lack the necessary flexibility and security features required for managing complex organizational structures and financial activities. To address these challenges, this paper presents a cross-platform banking prototype developed for internal transactions within a private-sector organization. The application is designed to facilitate secure and efficient financial operations, ensuring compliance with corporate banking standards and leveraging modern architectural patterns.

2. Core Features

The banking application offers a range of core features designed to streamline internal financial operations and enhance security. These features include user authentication, dashboard functionalities, fund transfers, account management, bill payments, and notifications.

2.1 User Authentication

User authentication is a critical aspect of the application, ensuring that only authorized personnel can access sensitive financial data and perform transactions. The application

implements multiple authentication mechanisms to provide robust security:

2.1.1 Login Credentials

Users can log in using their unique username and password. The application tracks and displays user login history to monitor account access and detect any unauthorized activity. Password security is enhanced through the use of the BCrypt encryption algorithm with asymmetric hashing. BCrypt is a password-hashing function based on the Blowfish cipher. It is designed to be computationally intensive, making it resistant to brute-force attacks. Asymmetric hashing refers to the use of a one-way hashing algorithm, where the hash function is easy to compute but difficult to reverse.

2.1.2 Azure AD Integration

To facilitate enterprise-level authentication, the application integrates with Azure Active Directory (AD). This integration provides several benefits:

- **Single Sign-On (SSO)**: Enables a seamless SSO experience for users using their Azure AD credentials, eliminating the need for separate login credentials.
- **SAML 2.0 Protocol**: Utilizes the SAML 2.0 protocol for secure authentication and authorization, ensuring secure communication between the application and Azure AD.
- **Auto-Login Mapping**: Automatically maps Azure AD users to the application, streamlining the user onboarding process and reducing administrative overhead.
- **Role-Based Access Control (RBAC)**: Integrates Azure AD roles with the application's RBAC, ensuring users have appropriate permissions based on their organizational roles.

2.2 Dashboard

The dashboard provides users with an overview of their account activities and quick access to common operations. Key features of the dashboard include:

- **Account Summary**: Displays an overview of account balances, providing users with a snapshot of their financial status.
- **Recent Transactions**: Shows a log of recent transactions and account activity, allowing users to monitor their financial activities and identify any discrepancies.
- **Quick Action Shortcuts**: Provides shortcuts for common operations, such as fund transfers and bill payments, streamlining user workflows and enhancing efficiency.

2.3 Fund Transfers

Fund transfers are a core function of the application, enabling users to perform internal transactions and manage beneficiaries. The application supports:

- **Internal Transactions**: Facilitates transfers between current, savings, and credit accounts within the organization.
- **Beneficiary Management**: Allows users to manage their list of beneficiaries, streamlining the process of sending funds to frequently used accounts.
- **Scheduled and Recurring Payments**: Enables users to schedule payments for future dates or set up recurring payments for regular expenses, automating routine financial tasks.

2.4 Account Management

Account management features provide administrators with the tools to manage user

accounts and access important account information. Key features include:

- **View and Manage Account Details**: Allows administrators to view and manage account details, such as user roles, permissions, and contact information.
- **Downloadable Transaction Statements**: Provides users with the ability to download transaction statements in various formats, such as PDF and CSV, for record-keeping and analysis.
- **Account Settings and Security Options**: Enables users to configure account settings and manage security options, such as password resets and two-factor authentication.

2.5 Bill Payments

The application supports the payment of various types of bills, streamlining the process of managing corporate expenses. Supported bill payment types include:

- **Utility Bill Payments**: Enables users to pay utility bills, such as electricity, water, and gas, either on a recurring or non-recurring basis, depending on the organization's provisioning.
- **Corporate Payments Processing**: Facilitates the processing of corporate payments, such as vendor invoices and supplier payments.

2.6 Notifications

Notifications keep users informed about important system updates and account activities. The application provides:

- **System Updates**: Notifies users about system updates, maintenance schedules, and new features, ensuring they are always aware of the latest changes.
- **Maintenance Notifications**: Informs users about scheduled maintenance periods, minimizing disruption and allowing them to plan accordingly.

3. Target Audience

The banking application is designed to serve several key user groups within the organization:

- **Corporate Users and Employees**: Provides employees with a secure and convenient platform for managing their internal accounts and performing financial transactions.
- **Internal Banking Operations Team**: Offers the banking operations team the tools they need to manage user accounts, process transactions, and monitor system activity.
- **Financial Administrators**: Provides financial administrators with comprehensive control over the application, allowing them to configure settings, manage security, and generate reports.

4. System Architecture

The banking prototype follows a microservices-based architecture, which offers several advantages in terms of scalability, maintainability, and fault tolerance.

4.1 High-Level Architecture

The high-level architecture of the application consists of several key layers:

- **Frontend Layer**: The user interface (UI) is built using OutSystems, a low-code development platform, and React Native, a cross-platform mobile framework. This combination allows for rapid development and deployment across multiple devices.

- **API Gateway**: A secure API gateway manages requests between the frontend and backend services, providing a single point of entry for all API calls and enforcing security policies.
- **Business Logic Layer**: The business logic layer consists of several microservices that handle banking operations, user management, and transactions. These microservices are designed to be independent and scalable, allowing the application to handle increasing workloads.
- **Database Layer**: The database layer uses SQL Server to manage all transaction and account data, providing a reliable and scalable storage solution.
- **Security Layer**: The security layer ensures data protection through encryption, secure authentication, and authorization mechanisms.

4.2 Supporting Systems

In addition to the core layers, the application includes several supporting systems to enhance reliability and security:

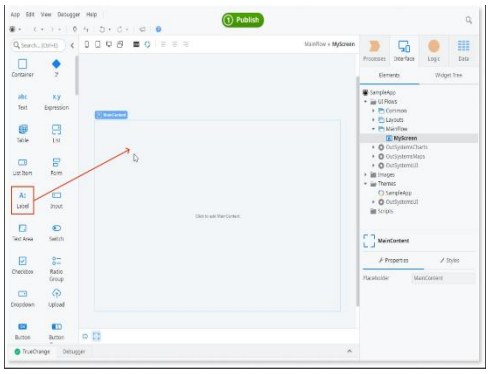
- **Backup System**: A backup system is implemented to back up transaction logs from the production database to a data warehouse, ensuring data durability and facilitating disaster recovery.
- **Jump Servers**: Jump servers are used to manage security threads, providing a secure entry point for accessing backend systems and minimizing the risk of unauthorized access.

5. Technology Stack

The banking application leverages a modern technology stack to ensure performance, scalability, and security.

5.1 Frontend

- **OutSystems**: A low-code UI development platform that enables rapid development and deployment of web and mobile applications.



- **React Native**: A cross-platform mobile framework that allows developers to build native mobile apps using JavaScript.

5.2 Backend

- **SQL Server**: A robust and scalable transactional database for storing and managing financial data.
- **Dockerized Microservices**: A modular service architecture that allows for independent deployment and scaling of individual services.
- **Other Backend Services**: Additional backend services handle

authentication, notifications, reports, and other ancillary functions.

5.3 Infrastructure

- **On-Premise Server**: The application is hosted on an on-premise server to ensure secure hosting within the corporate environment.
- **Kubernetes (K8s)**: A container orchestration platform that automates the deployment, scaling, and management of containerized applications.
- **Windows Server**: Used for hosting backend services and the backup system, with master-slave orchestration and RAID technology to ensure high availability and data redundancy.
- **Nginx Load Balancer**: A load balancer that distributes incoming network traffic across multiple servers, ensuring high system availability and performance.

6. Security & Compliance

Security and compliance are paramount concerns for any banking application. The banking prototype implements a comprehensive security framework to protect sensitive data and ensure compliance with relevant regulations.

6.1 Data Protection

- **AES-256 Encryption**: AES-256 encryption is used to protect sensitive data at rest and in transit, providing a high level of security.
- **SSL/TLS Communication**: SSL/TLS communication protocols are used to secure transactions and protect data in transit between the client and server.
- **Audit Logs**: Comprehensive audit logs are maintained to track user actions and system events, facilitating compliance tracking and security monitoring.

6.2 Authentication & Authorization

- **OAuth 2.0 / JWT-Based Authentication**: OAuth 2.0 and JWT-based authentication are used for secure user login sessions, providing a standardized and secure authentication mechanism.
- **LDAP Integration**: LDAP integration is used for enterprise login, allowing permanent employees to authenticate using their existing corporate credentials.
- **Session Management & Auto-Logout**: Session management and auto-logout features are implemented to protect user sessions and prevent unauthorized access.

7. API & Integrations

The banking application provides a set of APIs and integrations to facilitate communication with other systems and services.

- **User Services**: APIs for user registration, login, and profile management.
- **Transaction Services**: APIs for fund transfers, bill payments, and transaction history retrieval.
- **Third-Party Integrations**: Possible integrations with external banking systems, if required, to facilitate external transactions and data exchange.

8. Deployment & Infrastructure

The banking application is deployed using a containerized deployment approach,

leveraging Docker for portability and Kubernetes for orchestration.

- **Containerized Deployment**: Using Docker containers to package and deploy the application, ensuring consistency and portability across different environments.
- **Kubernetes (K8s) Cluster**: Managing scalability and resilience through a Kubernetes cluster, which automates the deployment, scaling, and management of containerized applications.
- **Windows Server**: Hosting microservices securely on the internal corporate network, providing a stable and reliable platform for running the application.
- **CI/CD Pipelines**: Automating deployment and updates using CI/CD pipelines, ensuring that new features and bug fixes are rapidly deployed to production.

9. User Interface & Accessibility

The user interface is designed to be intuitive and accessible, ensuring that all users can easily navigate and use the application.

- **Adaptive UI**: An adaptive UI that adjusts to different screen sizes and devices, providing a consistent user experience across desktop, tablet, and mobile devices.
- **WCAG Compliance**: Compliance with the Web Content Accessibility Guidelines (WCAG) to ensure that the application is accessible to users with disabilities.
- **Dark Mode Support**: Dark mode support is planned as an enhancement to reduce eye strain and improve usability in low-light environments.

10. Next Steps & Enhancements

The banking application is an ongoing project, and several next steps and enhancements are planned to further improve its functionality and security.

- **User Testing & Feedback Collection**: Conducting user testing and collecting feedback to identify areas for improvement and prioritize new features.
- **Security Audits & Compliance Validation**: Performing regular security audits and compliance validation to ensure that the application remains secure and compliant with relevant regulations.
- **Feature Expansion Based on User Needs**: Expanding the application's feature set based on user needs and feedback, ensuring that it continues to meet the evolving requirements of corporate users.

CONCLUSION

the cross-platform banking application successfully provides a secure and efficient solution for managing internal financial operations within organizations. By leveraging a microservices-based architecture, robust authentication mechanisms, and comprehensive data protection measures, the application streamlines transaction processing, enhances security, and improves overall financial management. Future enhancements will focus on user feedback, security audits, and expanding the feature set to meet evolving corporate needs.