iJRASET
International Journal For Research in
Applied Science and Engineering Technology

INTERNATIONAL JOURNAL
FOR RESEARCH
IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 12 Issue: XII Month of publication: December 2024

DOI: https://doi.org/10.22214/ijraset.2024.66162

Object-Oriented Analysis of a Modern Scalable and Secure Banking System Using UML

Retical Sravanthi[1], Matam Bhramari[2], Kontham Himabindhu[3], Natheer Abdulghani Saeed[4], Vangala Indumythri[5]
Master of Computer Applications, Chaitanya (Deemed to be) University, Hyderabad, India

Abstract: This study explores UML (Unified Modelling Language) and Object-Oriented Analysis and Design (OOAD) usage to build a banking system that is both secure and scalable. The paper suggests a modern approach for banking applications, focusing on important aspects like transaction security, the ability to handle system growth, and real-time monitoring. The paper also shows how object-oriented methods can handle the complexity of banking systems. It explores technologies like microservices and blockchain to make the system more reliable and efficient.

Lastly, it highlights the importance of security and fraud prevention, especially with real-time transaction tracking in today's fast-moving financial worlds.

Keywords: UML, Banking Systems, Object-Oriented Analysis, Scalability, OOAD, Mobile Banking Applications.

I. INTRODUCTION

In today's digital world banking systems need to be strong, scalable, secure and real time. Traditional banking applications face scalability, security and flexibility issues especially with online banking and digital finance. This paper looks into how Object-Oriented Analysis and Design (OOAD) principles can be used to create a modern banking system that addresses these issues using UML for system design. It also explores advanced architectural patterns like microservices their usage in emerging technologies like blockchain and real time transaction monitoring systems.

With the growth of technologies like real-time transaction processing, UPI, online banking, and mobile banking, banking systems have become much more advanced. Earlier, most banking applications were built in a way that made them difficult to scale or maintain, as they were monolithic systems, with the help of object-oriented principles by focusing more on modularity and reusability, many of these problems can be solved. The Unified Modelling Language (UML) has become a popular tool as it helps in clearly defining, visualizing, and documenting different parts of the system, mainly useful for complex systems like banking applications, as it allows us to design both broad overviews and detailed structures, ensuring the system is scalable, reliable, and easy to manage.

The objective of this paper is to show how a thorough object-oriented approach can be used to design a banking system that is both scalable and secure and meets the functional and non-functional requirements of today's digital banking.

This paper is organized into five chapters, Chapter I gives introduction, Chapter II gives the information about banking systems, Various UML diagrams required, designed UML diagrams for scalable, secure and reliable banking systems, and microservices, Chapter III gives the information about proposed modern architecture diagram using various microservices, Chapter IV provides the conclusion and future scope of the paper.

II. DESIGN AND IMPLEMENTATION

This chapter provides a detailed description of the architecture and system design using Object-Oriented Analysis and Design (OOAD) principles. The foundation of the banking system design is based on object-oriented principles, ensuring that the system is modular, flexible, and easy to maintain.

The main steps of OOAD include:

- Requirements Analysis: Identifying key banking functionalities such as account management, transaction processing, customer support, and security protocols.
- System Design: Using UML to model both structural and behavioural aspects of the system, such as class diagrams, use case diagrams, sequence diagrams, and activity diagrams.
- Implementation: design of UML system's architecture that ensures the scalability, security, and maintainability.

The design of a banking system requires a combination of scalability, security, reliability, and portability. The UML is used to visualize the banking system's design. The following types of UML diagrams are used for design and implementation.

- Use Case Diagrams: To capture the functional requirements and user interactions with the banking system.
- Class Diagrams: To define the system’s objects, attributes, and relationships between them.
- Sequence Diagrams: To explain interactions between objects over time, focusing on transaction flows and customer interactions.
- Component Diagrams: To represent the physical architecture and communication between different system components, including microservices.
- Deployment Diagrams: To map out how the banking system will be deployed on the physical infrastructure.

The Unified Modelling Language (UML) is utilized to model the system, focusing on core features such as user authentication, transaction handling, data storage, and monitoring.

The High-level architecture of the system key components are:

- Web Portal: The interface for user interaction.
- Authentication and Authorization Services: For secure login and role-based access.
- Microservices: Decomposes the banking system into smaller, independently deployable services, that can be scaled as per the demand.
- Database: For secure storage of customer and transactional data.
- Blockchain: To ensures secure, tamper-proof financial transactions, providing a decentralized ledger to prevent fraud.
- Monitoring Services: For real-time system health and security monitoring.
- Backup and Cloud Storage: To ensure reliability and disaster recovery.
- Audit and Compliance: To Track all transactions and ensures compliance with regulatory standards such as KYC.

A. UML Diagrams for secure, portable, and reliable banking System

1) *Use Case Diagram*: This use case diagram illustrates the use case of the banking system, that is user interactions like login, fund transfer, account management.

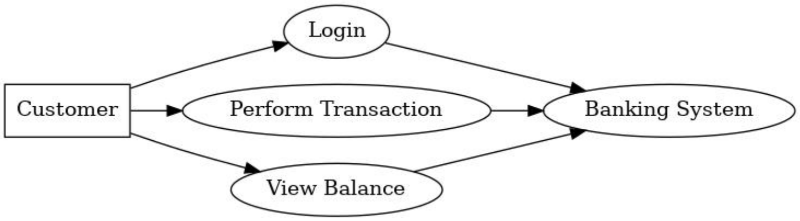

Fig. 1 Use case diagram

2) *Sequence Diagram*: This UML diagram illustrates the sequence diagram of the banking system, as:

- User initiates the login request
- Web portal: receives / forwards the requests
- Authentication services: Handles the credential validation
- Transaction services: handles services like Fund transfer, balance enquiry, mini statement, account management.
- Audit services: maintain the ledgers, track of transactions, and to maintain the standards.
- Encryption: to encrypt the sensitive data
- Database: to store the data

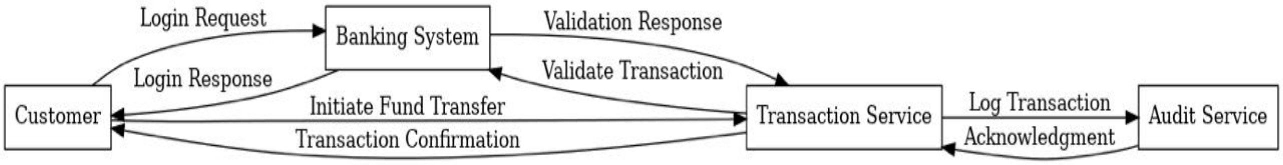

Fig. 2 Sequence diagram

3) *Class Diagram*: The class diagram depicts the relationship between four main entities in a banking system, They are: 1) Customer, 2) Bank Manager, 3) Account, and 4) Transactions.

- Customer: Represents the individual, with attributes like Customer ID, Name, Email, Phone number etc. (Owns the relationship)
- Bank Manager: Represents the bank staff, with attributes as: Manager ID, Name, Branch etc. (Manages the relationship)
- Account: Holds the attributes like Account Id, Balance, Type of account, etc. (Has relationships)
- Transactions: Represents the activities linked to accounts, with details as: Transaction Id, Amount, Type of transaction, Dat and time of transaction etc.

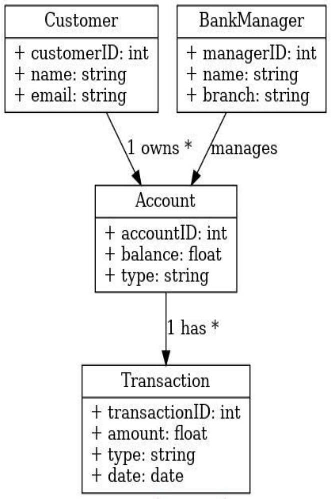

Fig. 3 Class diagram

4) *Activity Diagram*: This activity diagram shows how two-factor authentication (2FA) works for user authentication. The user inputs their credentials, which are then verified, to start the workflow. Access is refused if the credentials are invalid. A 2FA code is delivered to the user if the credentials are valid. After receiving the 2FA code, the user inputs it to be further verified. Access is allowed if the 2FA validation is successful and refused otherwise. The decision-making points and sequential actions necessary to provide safe system access are explained in this diagram.

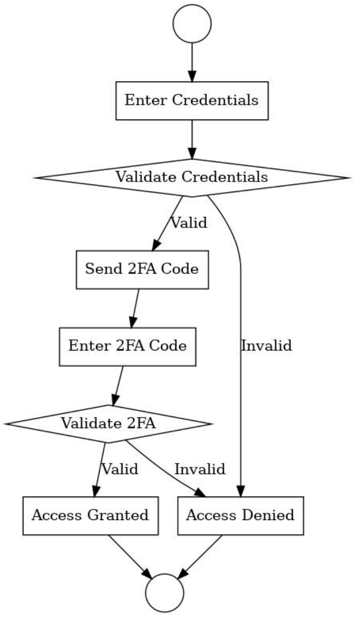

Fig. 4 Activity diagram

5) *Microservices Diagram*: The microservices architecture UML diagram represents the modern, scalable, and secure banking system.

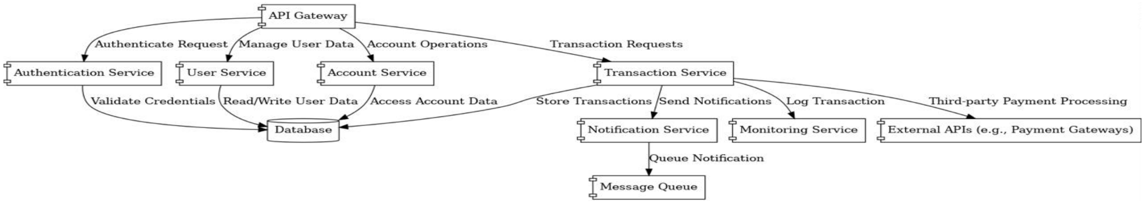

Fig. 5 Microservices diagram

Components of above microservices diagram are:

- API Services: All client queries are entered through API Gateway Services, which then forward them to the relevant microservices. It manages load balancing, rate-limiting, and authentication.
- Authentication Service: manages the user authentication, confirming credentials. This may also interact with database service for validation.
- User Service: It manages services like retrieving user data, updating personal information, and managing profiles etc.
- Account Service: This manages account-specific tasks, including creating, updating, and viewing balances.
- Transaction Service: This Manages financial transactions like processing payments and transferring funds, it can also interact with other microservice as:

➢ Notification Service: Sends transaction-related notifications to users.
➢ Monitoring Service: Logs transactions for auditing and monitoring purposes.
➢ External APIs: For third-party payment gateways or financial institutions.

- Notification Service: It manages the Queues and sends notifications (e.g., SMS, emails) using a Message Queue.
- Monitoring Service: Continuously logs and monitors system activities for auditing, analytics, and anomaly detection.
- Database: A shared repository that securely stores data for authentication, user accounts, transactions, and logs.
- Message Queue: Ensures reliable communication and processing of asynchronous tasks like sending notifications.
- External APIs: Represents external systems, such as payment gateways, integrated with the transaction service.

III.PROPOSED SECURE, PORTABLE, SCALABLE ARCHITECTURE WITH VARIOUS MICROSERVICES

The proposed diagram shows a modern, secure, and scalable design for a web-based banking system. It starts with user requests being filtered through a firewall and then distributed by a load balancer to ensure smooth traffic management. The web portal handles key functions like user authentication, logging, and business logic while delegating specific tasks to different services. Important components include the Authentication Service, which validates users and controls access, and the Encryption Service, which secures sensitive data. Real-time monitoring is handled by the Monitoring Service and Intrusion Detection System (IDS) to track activities and detect threats. Data is processed by the Data Processing Service and securely stored in a database, which is synced with cloud storage and a backup server to ensure reliability and portability.

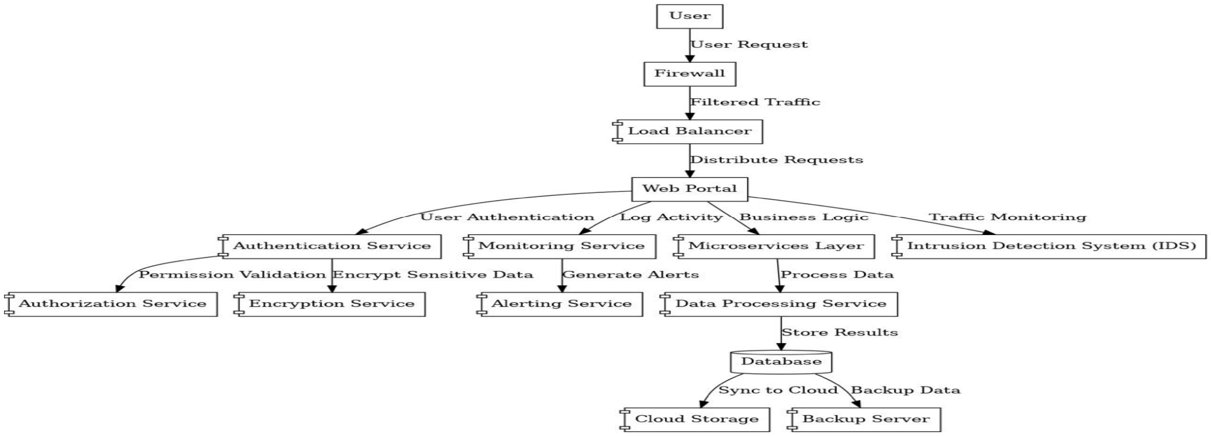

Fig. 6 Proposed Architecture diagram with various microservices

IV. CONCLUSIONS

This paper focuses on designing a secure and scalable banking system using Object-Oriented Analysis and Design (OOAD) principles and UML diagrams. The proposed system uses a modern microservices architecture, offering features like real-time monitoring, enhanced security, and better reliability. The UML diagrams simplify the understanding and implementation of the system, making it more practical to maintain as per the demand. In the future, adding AI-based features like predictive analytics and anomaly detection can further improve its efficiency and security.

REFERENCES

[1] Singh, A., & Kaur, R. "Object-Oriented Software Engineering and UML", 2021, Wiley.

[2] Habtamu Keno "Object Oriented System Analysis and Design (OOSAD)", WOLISO, Ethiopia.

[3] Pandey, Sunil & Singh, G.P. & Kansal, V.. (2011). Study of Object Oriented Analysis and Design Approach. Journal of Computer Science. 7. 143-147. 10.3844/jcssp.2011.143.147.

[4] Dr. Harsh Dev, Suman Kumar Mishra "UML Modelling for online banking system using object oriented databases", 2021, Wiley.