A SECURE BANKING SYSTEM

CSE 545: Software Security Course Project - Design Document

SUBMITTED TO

Professor Stephen.S.Yau

Ira A. Fulton Schools of Engineering

Arizona State University

SUBMITTED BY - GROUP 13:

Atin Singhal(Leader)

Shivank Tiwari(Deputy Leader)

Sriprashanth Ramamoorthy

Ayush Ray

Raj Buddhadev

Kangjian Ma

Uttam Bhat

Amitabh Das

Table of Contents

1. Introduction

1.1 Purpose of the project

This project is to design and implement a secure banking system, where fundamental functionalities are developed, basic performance is satisfied, and critical security requirements are met. The purpose of the project is to gain a deeper knowledge of the instructed material on the software security by a comprehensive hands-on project with intensive and extensive teamwork.

1.2 Scope

The developed secure baking system can accommodate multiple users to use the system under sufficient security at any time and from any place as long as the Internet and a web browser is accessible. Five user categories are implemented in the system including bank employees with three tiers and customers with two kinds, i.e. individual customers and merchants/organizations. Several critical security measurements are taken to prevent malicious attacks, including public key certificates, one time password, blockchain techniques. Log files are created to record the log-in history, transaction history of customers for the customers and employees with authority to review.

1.3 Definitions and Acronyms

A list of definitions are as follows.

(1) Secure Banking System: It is a software system developed primarily to facilitate secure banking transactions and user account management through the Internet.
(2) One Time Password: It is valid for only one login and for a specific amount of time stated at the time of generation. In contrast to static passwords, it is impervious to replay attacks.
(3) Personally Identifiable Information: In this system design, the personally identifiable information refers to account number, email, phone number.
(4) Internal Users: The employees of the bank system, including tier-1 employees, tier-2 employees and tier-3 employees.
(5) External users: two groups of banking system users, including individual users and merchants/organizations.

A list of acronyms are as follows.

(1) SBS: Secure Banking System
(2) UML: Unified Modeling Language

(3) PII: Personally Identifiable Information
(4) SSL: Secure Sockets Layer
(5) TLS: Transport Layer Security
(6) PKI: Public Key Infrastructure
(7) OTP: One Time Password
(8) ORM: Object-relational mapping

2. System Overview

The secure banking system is hosted on a virtual machine and be 24*7 available to customers and employees online through a web browser.

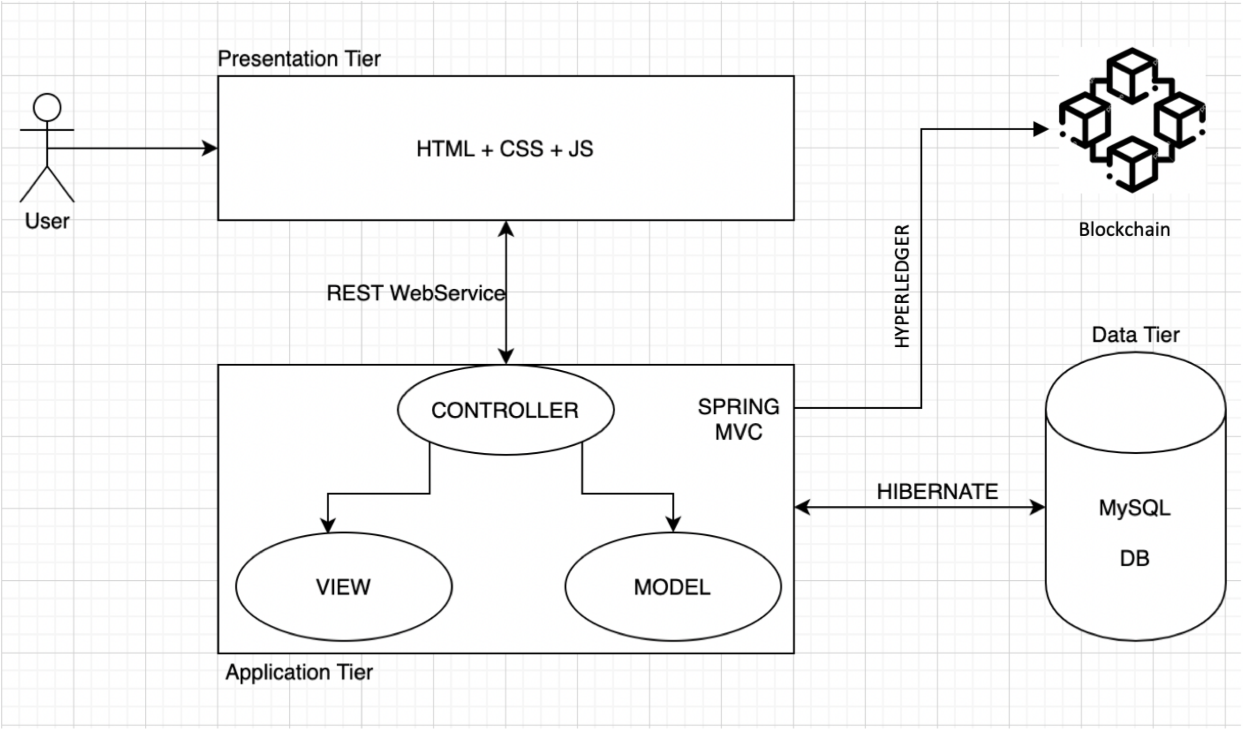
User-friendly interfaces are designed to the frontend. Logic and security realization are implemented in the backend. Database is initiated and maintained to support multi-functions of the system. Two kinds of users are hypothesized to use the system, categorized as internal users and external users. The internal users can be classified into 3 groups with gradually higher privileges, i.e. tier-1 employees, tier-2 employees, and tier-3 employees. The external users include individual customers and merchants/ organizations. Three kinds of accounts, i.e. checking account, saving account and credit accounts, are designed as in the real world. Basic functionalities are developed to allow users to create, view, modify, close accounts, debit, credit, and transfer money, request, grant and decline authorization, view log-in and transaction history or bank statement in a secure environment. All valid and approved transactions are captured in Hyperledger blockchain platform to guarantee safety along with the traditional security measurements taken including Secure Sockets Layer/ Transport Layer Security, Public Key Infrastructure, One Time Password, Data Masking and Hashing techniques. Besides the account number, associated emails and phone numbers are valid Personal ID as well for money sending. Technical account access is realized for employees with proper privilege to assist customers in troubleshooting and maintenance.

The banking system is developed by comprehensively considering real-world functionalities and potential malicious security attacks, which helps the teammates attain a deeper knowledge of software security in theory and in practice.

3. System Architecture

3.1 Architectural Design

The system will be using a three tier architecture: Presentation, application and data tier as shown in the figure below. There are many benefits to using a 3-layer architecture including speed of development, scalability, performance, and availability. As mentioned, modularizing different tiers of an application gives development teams the ability to develop and enhance a product with greater speed than developing a singular code base because a specific layer can be upgraded with minimal impact on the other layers. It can also help improve development efficiency by allowing teams to focus on their core competencies. Many development teams have separate developers who specialize in front- end, server back-end, and data back-end development, by modularizing these parts of an application you no longer have to rely on full stack developers and can better utilize the specialties of each team. Scalability is another great advantage of a 3-layer architecture. By separating out the different layers you can scale each independently depending on the need at any given time.

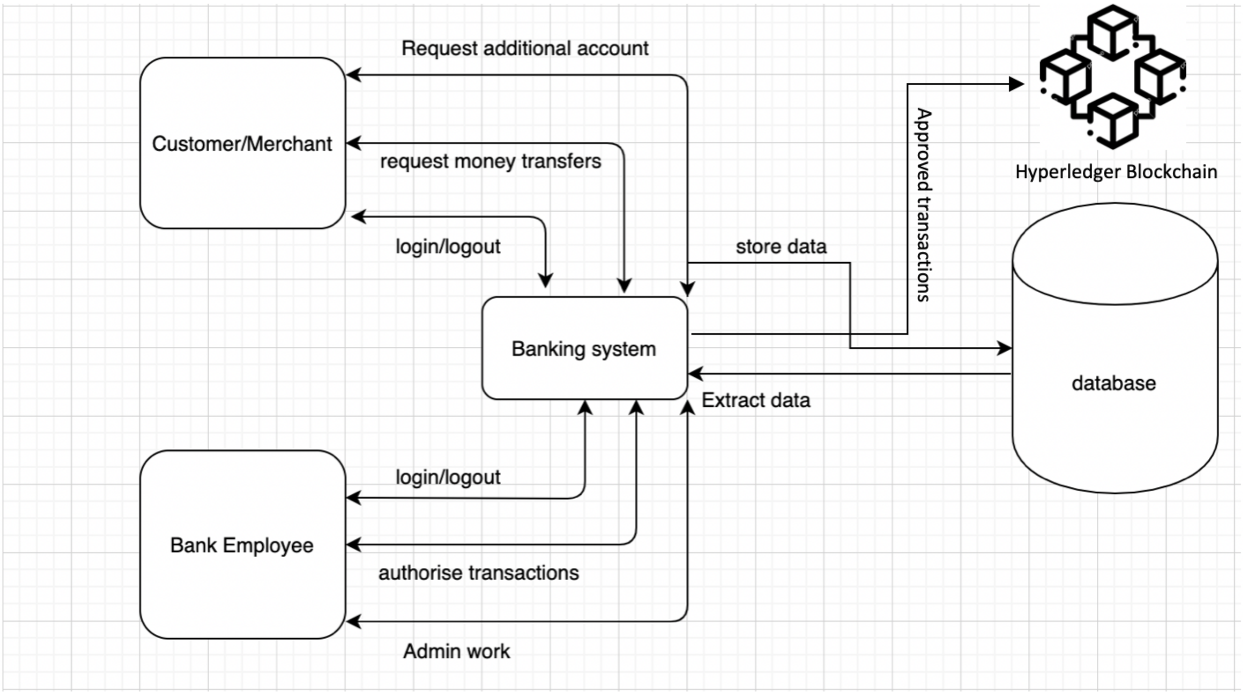


- **Presentation Tier-** The presentation tier is the front-end layer in the 3-tier system and consists of the user interface. This tier is built on web technologies such as HTML5, JavaScript, CSS and communicates with other layers through API calls.
- **Application Tier-** The application tier contains the functional business logic which drives an application's core capabilities. It will be built using Java and the Spring framework. The application layer would communicate to the presentation layer through web services.
- **Data Tier-** The data tier comprises of the database/data storage system and data access layer. Data would be stored in MySQL database for the banking system and would be accessed by Application Layer using JPA.

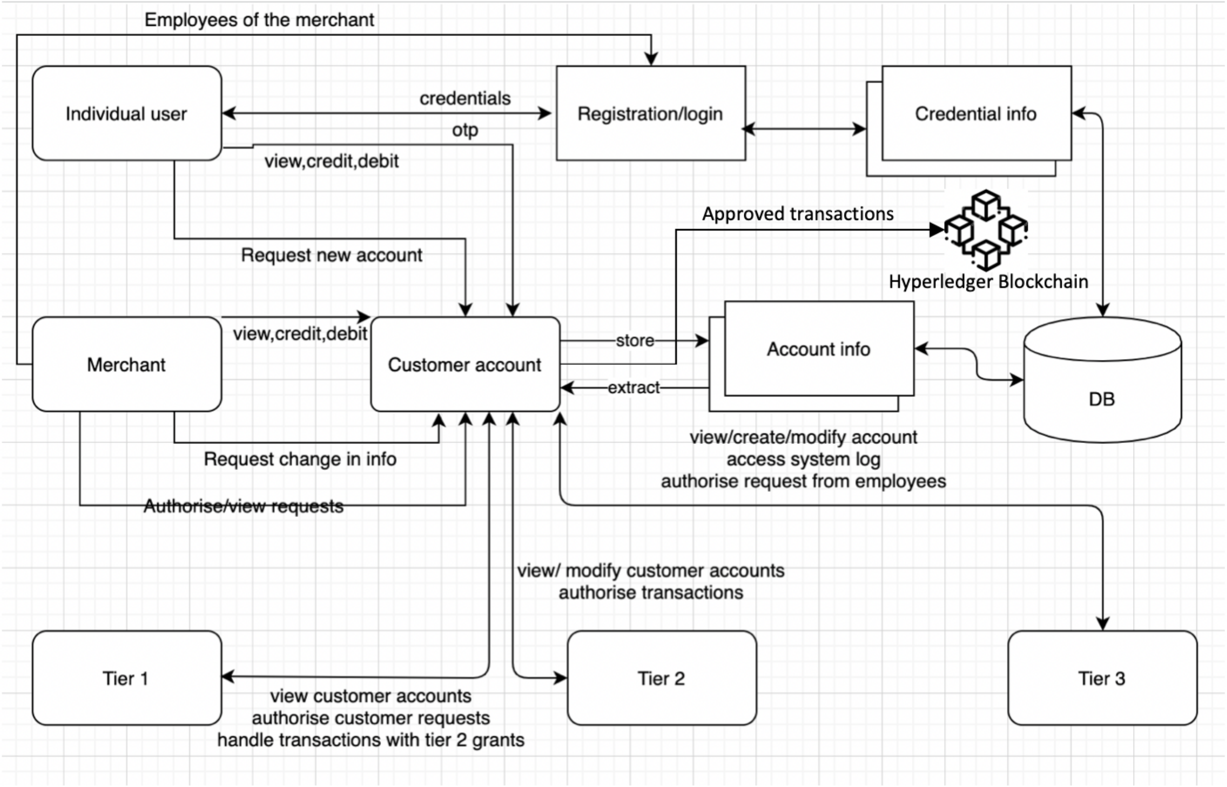
3.2 Decomposition Description

The following data flow diagram explains how information flows within the system. It indicates how data is utilized, processed and stored after each user action.

3.2.1 Overview



3.2.2 Detailed flow diagram



The figure above shows the modular level of decomposition of the entire system. It shows the relations between different users and data flow during a transaction. Security is the most important component in mind while deciding the architecture. We are using the spring framework and server-side backend programming. The reason being Spring is easy to use/build and comes with strong security features.

4. Data Design

4.1 Data description

Database is one of the important components of the secure banking system as there is a need of storing a lot of information pertaining to the banking system. Information about users, login details, account details, transactions, and appointments needs to be stored in the database. Hence, tables storing these different details would need to be created as early as possible to support the development of the banking system.

The important fields that would be required to be stored in the database are explained below.

1. User Login Details - The user login details such as username, password, user created date, etc. would be required to be stored in the database.
2. User Type - The type of user i.e. Tier 1, Tier 2, Tier 3, Individual or Merchant would need to be stored in the database. Different functionalities would be made available to the user based on their type.
3. User Profile - All the details of the user such as name, email, phone, etc. would be required to be stored in order to retrieve user data when necessary.
4. Account Details of all users - Account details such as current balance, account open date, user associated with an account, etc. would need to be stored in the database.
5. Transaction Details of every transaction initiated - Record of all the transactions would need to be maintained. Hence, all the transaction details would be required to be stored.
6. Appointment Details of all users - All the details regarding the appointment requested and the details of the appointment would be required to be stored in the database.
7. Login History of all users - Login history of all the users along with the details like last login time, ip address, etc. would need to be stored in order to keep track of user activity and avoid security issues from unknown sources.

Communication between database and backend will be done by using Hibernate framework supported by Java. By implementing hibernate, we will create a Java class for each table that will contain fields corresponding to columns in a table. Through this framework, we will be able to interact with the data in each of the tables, as queries to retrieve and insert data would be written in ORM files.

4.2 Data dictionary

The data of our system will be stored in different tables that are created by implementing database design principles. Below are the tables that will be created in MySQL database.

USER

Column Name	DataType
user_id	int
username	varchar(255)
password	varchar(60)
status	int
incorrect_attempts	int
created_date	datetime
modified_date	datetime
is_external_user	boolean

USER_DETAILS

Column Name	DataType
user_id	int
first_name	varchar(255)
middle_name	varchar(255)
last_name	varchar(255)
email	varchar(255)
phone	varchar(15)
tier	varchar(10)
address1	varchar(255)

address2	varchar(255)
city	varchar(255)
province	varchar(255)
zip	int(5)

TRANSACTION

Column Name	**DataType**
transaction_id	int
transaction_type	int
transaction_status	int
transaction_amount	int
is_critical_transaction	boolean
transaction_created_date	datetime
transaction_updated_date	datetime
from_account	int
to_account	int
transaction_approved_by	int

REQUESTS

Column Name	**DataType**
request_id	int
requested_by	int
type_of_request	int

request_assigned_to	int
type_of_account	varchar(25)

ACCOUNT

Column Name	DataType
account_id	int
user_id	int
account_type	varchar(25)
current_amount	int
created_date	datetime
status	int

APPOINTMENT

Column Name	DataType
appointment_id	int
appointment_user_id	int
assigned_to_user_id	int
created_date	datetime
appointment_status	varchar(25)

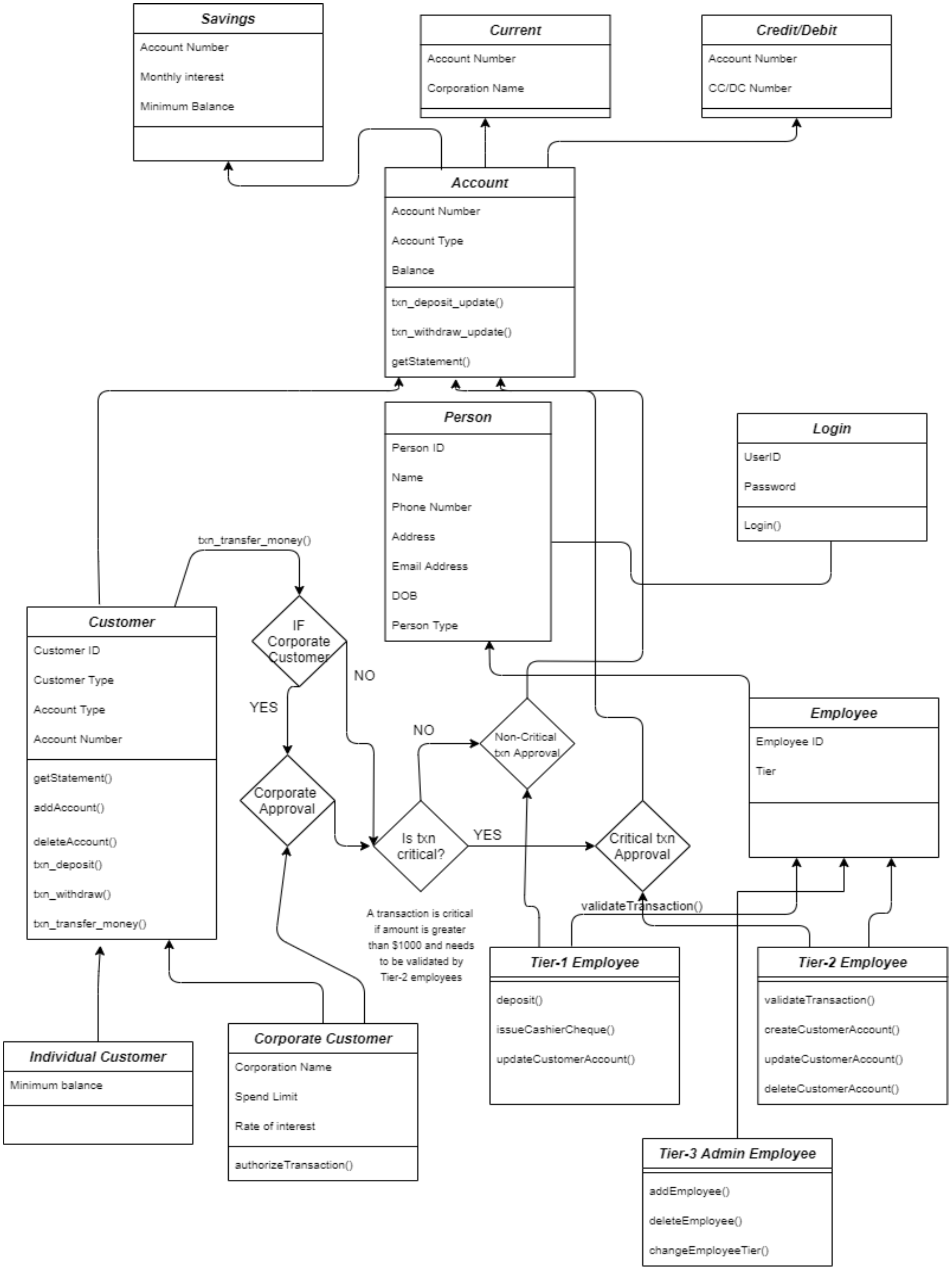
LOGIN_HISTORY

Column Name	DataType

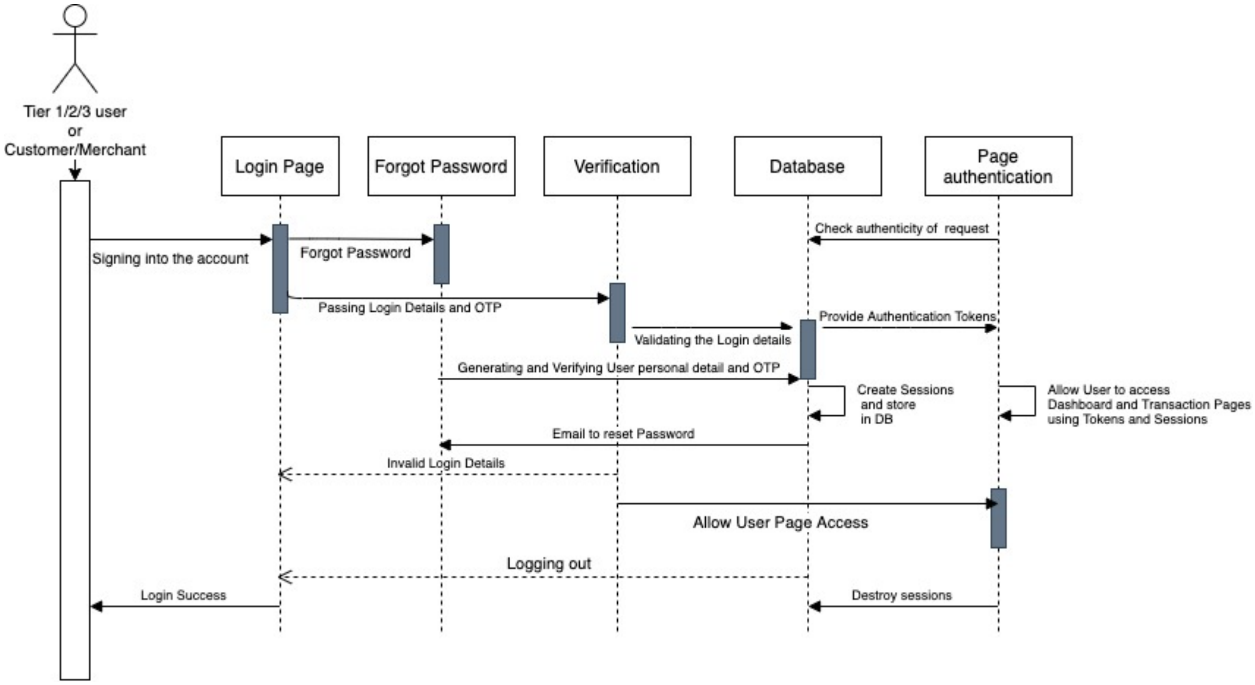
user_id	int
logged_in	datetime
logged_out	datetime
ip_address	varchar(25)
device_type	varchar(25)

5. Component Design

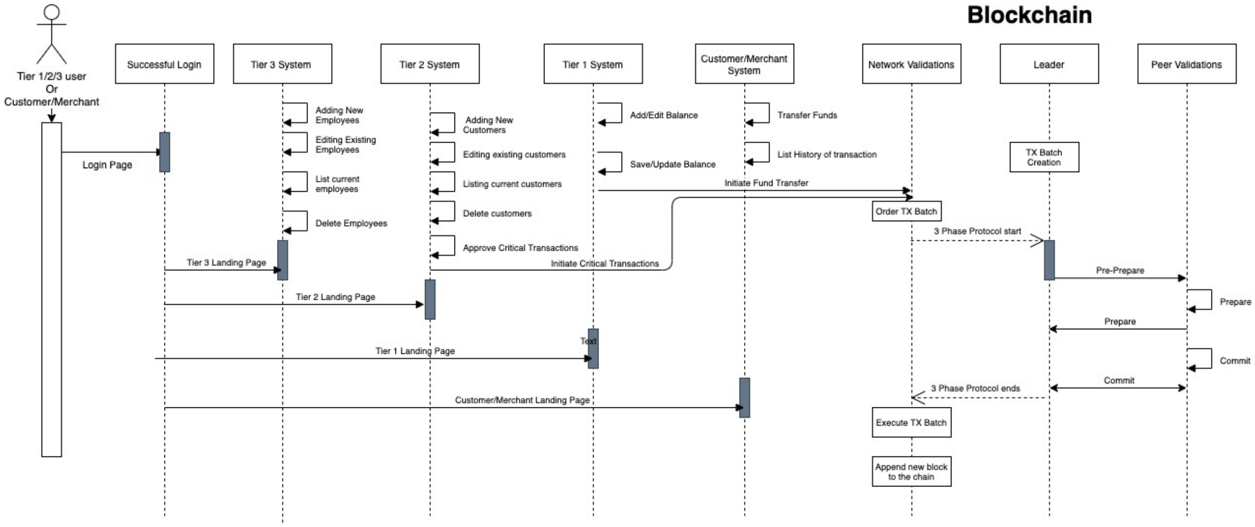
5.1 Class diagrams



5.2 Sequence diagrams

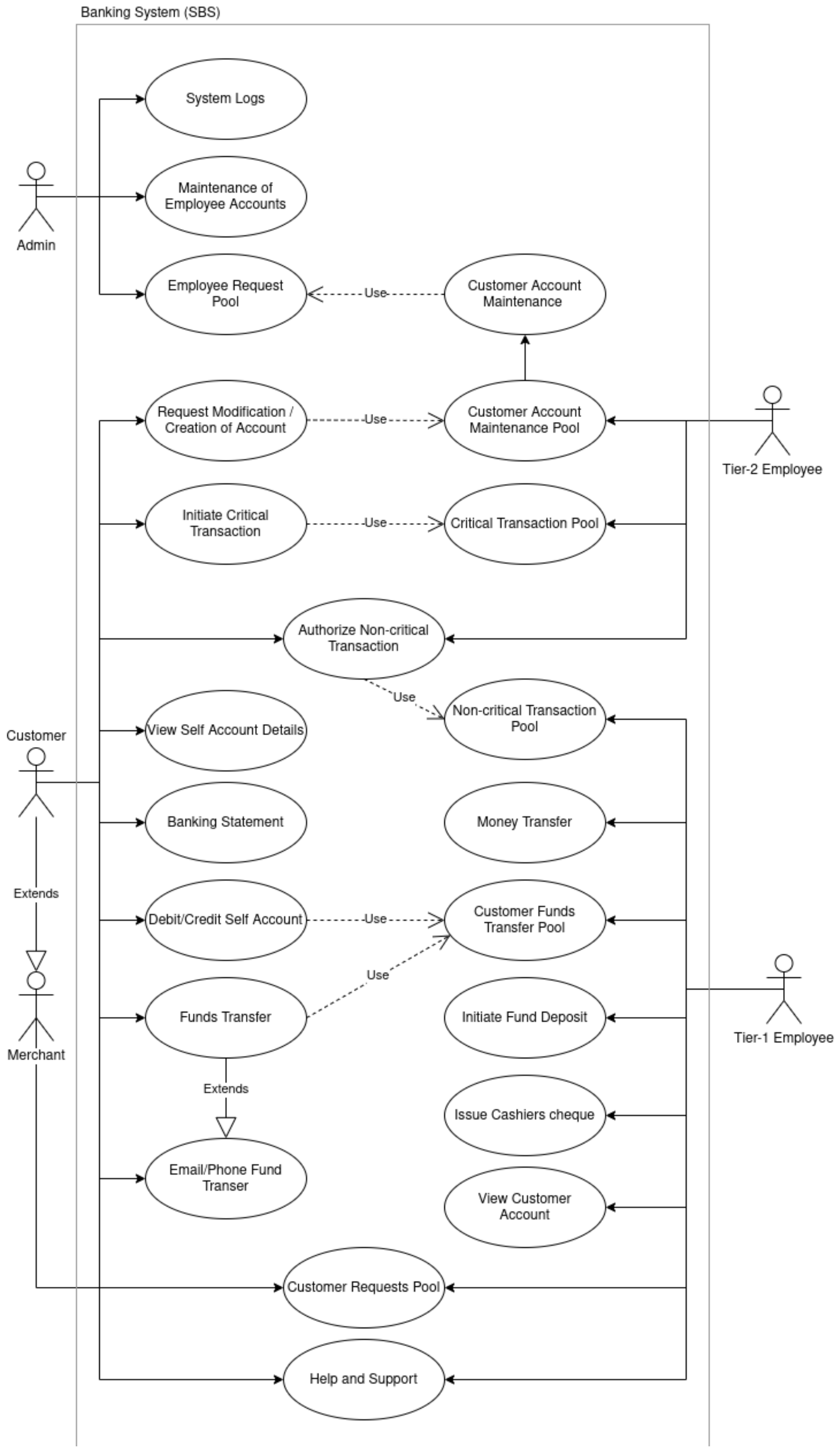


Login functionality for users

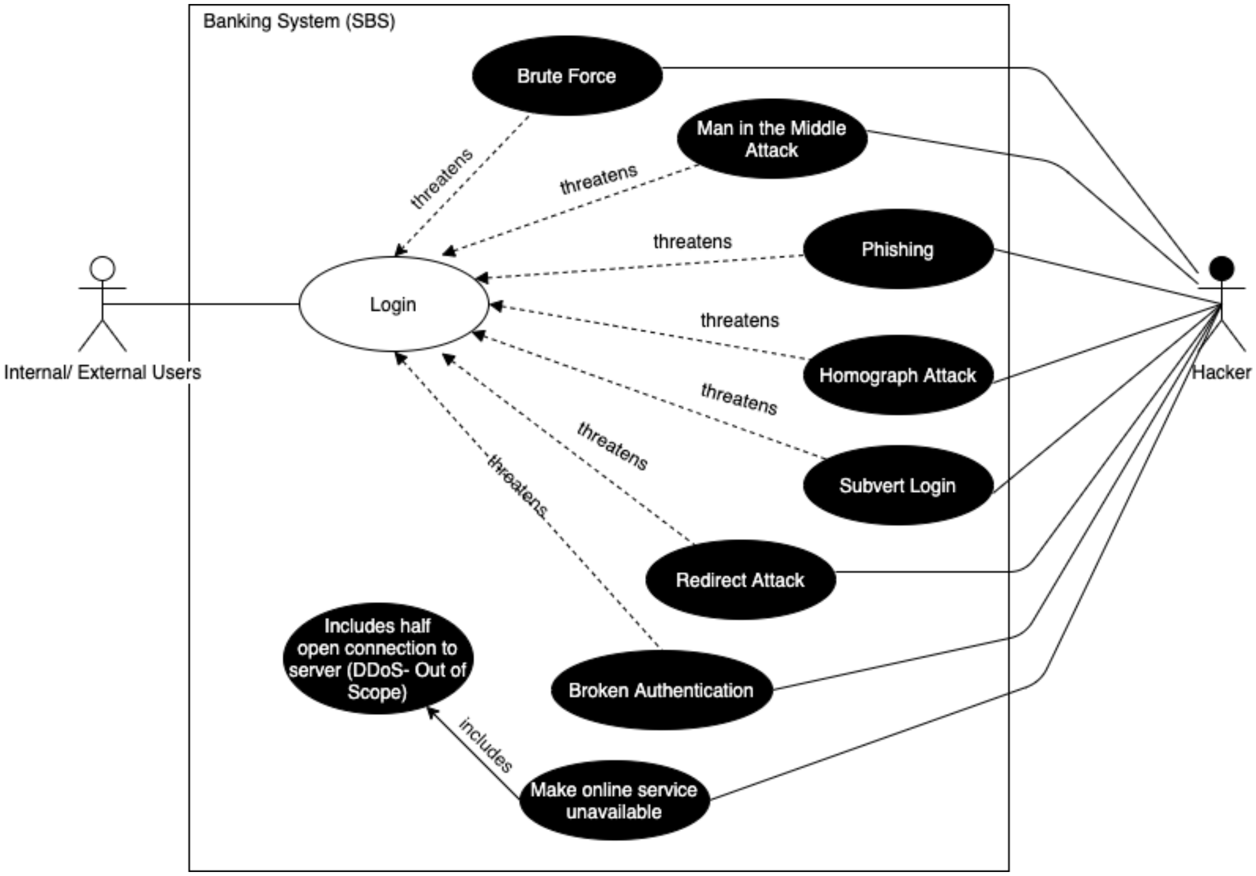


Banking functionality for users [2]

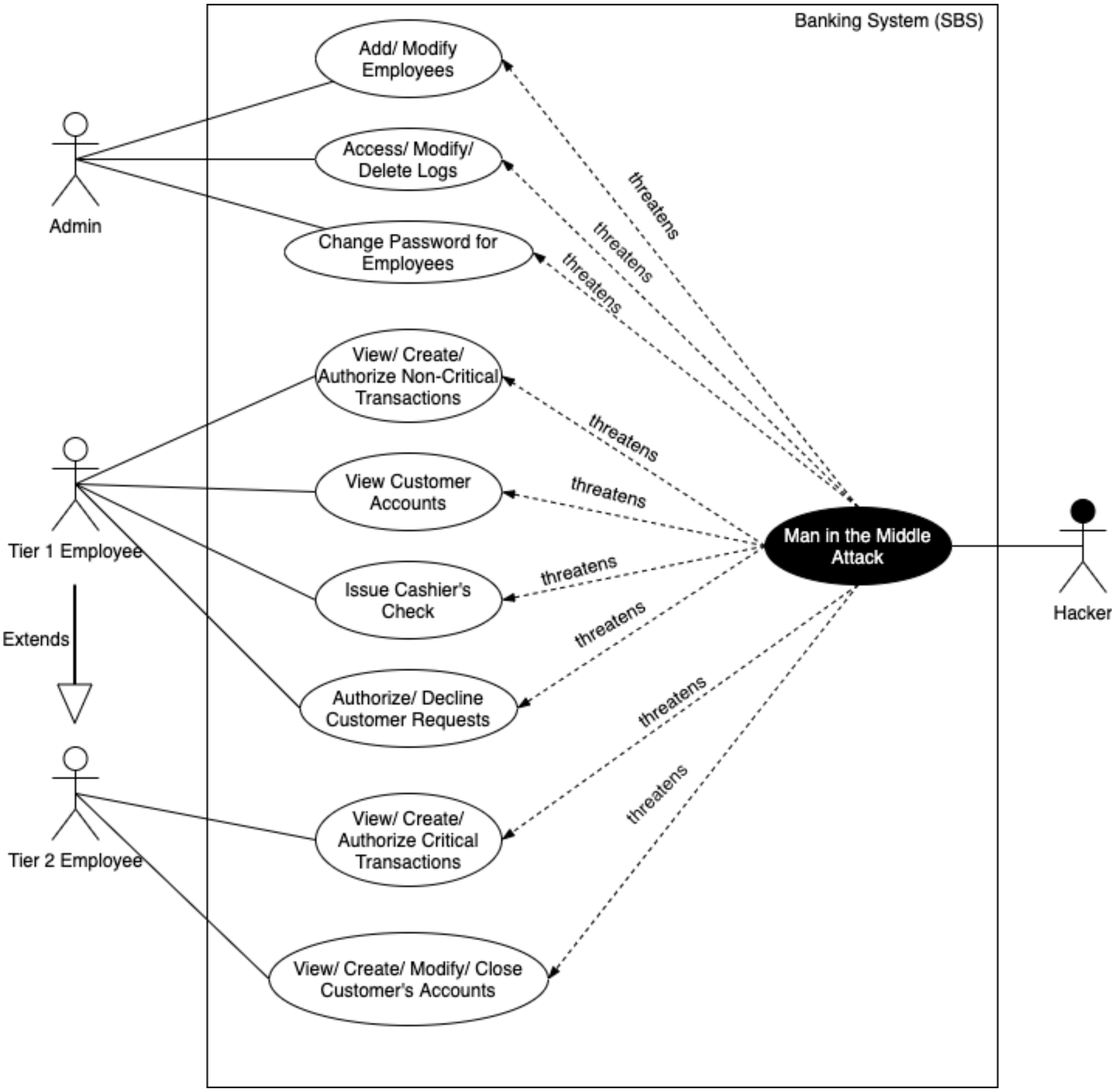
5.3 Use case diagram


Banking System (SBS)
System Logs
Admin
Maintenance of Employee Accounts
Employee Request Pool
Use
Customer Account Maintenance
Request Modification / Creation of Account
Use
Customer Account Maintenance Pool
Tier-2 Employee
Initiate Critical Transaction
Use
Critical Transaction Pool
Authorize Non-critical Transaction
Use
Non-critical Transaction Pool
Customer
View Self Account Details
Banking Statement
Money Transfer
Extends
Debit/Credit Self Account
Use
Customer Funds Transfer Pool
Use
Merchant
Funds Transfer
Initiate Fund Deposit
Tier-1 Employee
Extends
Issue Cashiers cheque
Email/Phone Fund Transer
View Customer Account
Customer Requests Pool
Help and Support

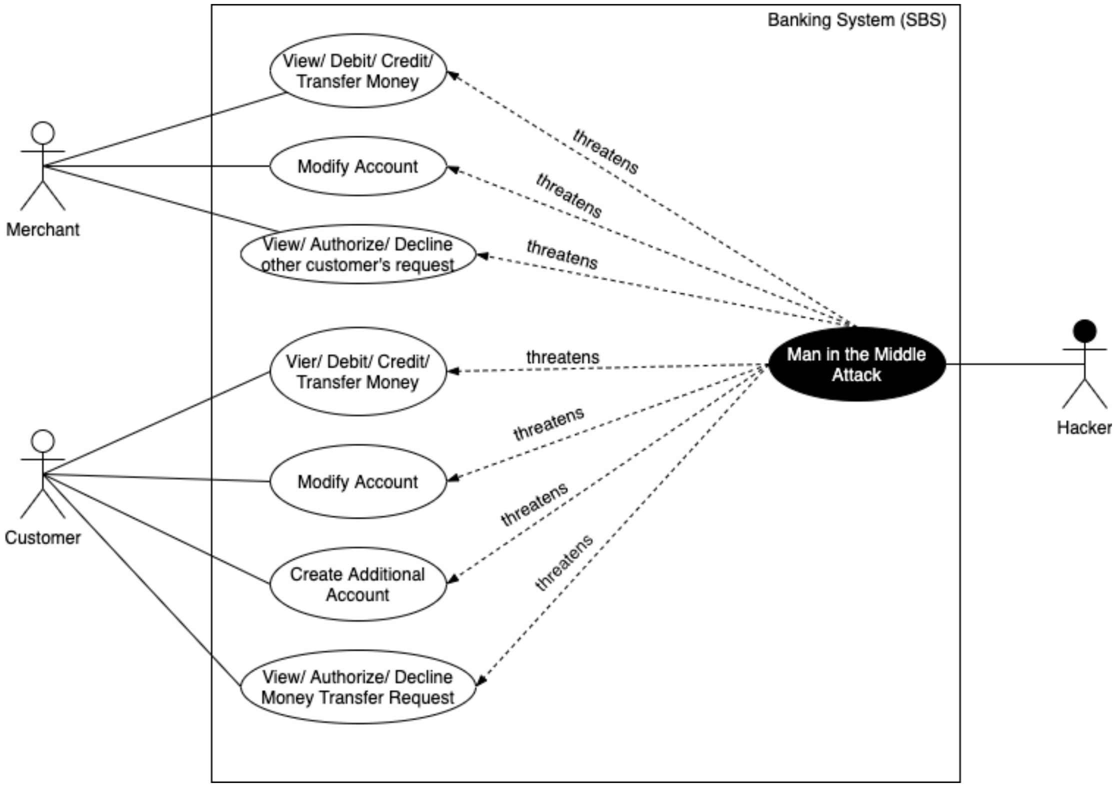
5.4 Misuse case diagrams



Misuse cases at Login



Misuse cases for Internal Users


Banking System (SBS)
View/ Debit/ Credit/ Transfer Money
Modify Account
View/ Authorize/ Decline other customer's request
Vier/ Debit/ Credit/ Transfer Money
Modify Account
Create Additional Account
View/ Authorize/ Decline Money Transfer Request
Merchant
Customer
threatens
threatens
threatens
threatens
threatens
threatens
threatens
Man in the Middle Attack
Hacker

Misuse cases for External Users

6. Requirement Traceability Matrix

S.No.	REQUIREMENT	LOGIN	REGISTRATION	MONEY TRANSACTIONS	TXN REQUESTS	USER MGMT.	USER REQUESTS.	EMPLOYEE MGMT	AUTHORIZATION REQUESTS	LOG
1	An individual user (individual customer) / Merchant	X	X	X	X		X			
2	View / Debit / Credit / Transfer Money			X	X					
3	Email / Phone Transfer			X						
4	Technical Account Access						X			
5	Banking statements					X	X			
6	Opening additional account						X			
7	Help & Support Center						X			
8	Sign-in history function					X	X			
9	View / Authorize / Decline Transfer requests				X					
10	Modification of personal account						X			
11	Tier 1 employees			X	X	X	X		X	
12	View / Create / Authorize Non-critical transactions			X	X					
13	View customers' accounts					X				
14	Manage Fund Transfer			X	X					
15	Issue cashier cheques			X						
16	Authorize / Decline customer's request						X		X	
17	Sign-in history function							X		X
18	Tier 2 employees		X		X	X	X		X	
19	View / Create / Modify / Close customer' accounts		X			X				
20	Authorize / Decline transactions				X				X	
21	Modification of accounts					X	X			
22	Sign-in history function							X		X
23	An administrator							X	X	X
24	View / Create / Modify / Delete Employee's accounts							X		
25	Authorize / Decline employees' request.								X	
26	Access the system log file									X
27	Sign-in history function							X		X

7. References

[1] https://www.jinfonet.com/resources/bi-defined/3-tier-architecture-complete-overview/

[2] Sukhwani, Harish, et al. "Performance modeling of pbft consensus process for permissioned blockchain network (hyperledger fabric)." *2017 IEEE 36th Symposium on Reliable Distributed Systems (SRDS)*. IEEE, 2017.