**Team\_101**

**Table of Contents**

[**1. Introduction 4**](#_1fob9te)

[1.1 Purpose 4](#_3znysh7)

[1.2 Document Conventions 4](#)

[1.3 Intended Audience and Reading Suggestions 4](#)

[1.4 Product Scope 4](#)

[1.5 References 5](#)

[2. Overall Description 5](#)

[2.1 Product Perspective 5](#_2s8eyo1)

[2.2 Product Functions. 5](#)

[2.3 User Classes and Characteristics 5](#)

[2.4 Operating Environment 6](#)

[2.5 Design and Implementation Constraints 6](#)

[2.6 User Documentation 6](#)

[2.7 Assumptions and Dependencies 6](#)

[3. External Interface Requirements 6](#)

[3.1 User Interfaces 6](#_2jxsxqh)

[3.2 Hardware Interfaces 6](#)

[3.3 Software Interfaces 7](#)

[3.4 Communications Interfaces 7](#)

[4. System Features 7](#)

[4.1 Login and Creating an Account 7](#)

[4.2 Get Account Details 7](#_k03ftjmn5oqu)

[4.3 Analyze Account 8](#_33i2w8kkay4r)

[4.4 Upload Statement for Analysis 8](#_p5bsgvluns6e)

[4.5 Send and Verify OTP 8](#_rzimq4m7vrex)

[5. Other Nonfunctional Requirements 9](#_3whwml4)

[5.1 Performance Requirements 9](#_2bn6wsx)

[5.2 Safety and Security Requirements 9](#)

[5.3 Software Quality Attributes 9](#)

[6. Design Diagrams 10](#_6o6prv6zcvxp)

[6.1 Control Flow Diagram 10](#_2ile29j3qekl)

[6.2 Data Flow Diagram 10](#_ha8ryeg24sl2)

[6.3 Class Diagram 12](#_q6a3di7qo1z5)

[6.4 Analysis Sequence Diagram 13](#_c47vx0le1rd8)

**Revision History**

| **Name** | **Date** | **Reason For Changes** | **Version** |
| --- | --- | --- | --- |
| Version 0.1 | 25-03-2023 | Initial SRS Document | 0.1 |
| Version 1.0 | 01-04-2023 | Updated the SRS Document with diagrams | 1.0 |

# Introduction

## Purpose

The purpose of this document is to present a detailed description of ShiftBank. It will explain the objective/purpose of this project and its scopes of application, features of the projects, the interface and the details about interactions through the interface with the database, data management strategies, constraints under which it will operate.

## Document Conventions

This document has been written using Times New Roman font with font size 12 and important points are written in bold. Additionally, the headings are written in bold, using ‘Times New Roman’ font with font size 18 and subheadings with font size 14.

## Intended Audience and Reading Suggestions

This document is intended for developers contributing towards the project, the client, users, and testers and anyone who is reviewing the project to easily view descriptions about functional and no-functional requirements and understand product scope.

This document contains six (6) sections:

a. An overall description, which includes a summary of the scope, assumptions and limitations of the project, types and characteristics of users, plan for implementation.

b. A brief description of external interfaces requirements.

c. A description of the system, which includes an explanation of the system’s purpose, features and benefits, characteristics, and technical and operational background; and,

d. A list of nonfunctional requirements, which contains information regarding performance,safety, security and business rules.

e. A description of other uncategorised requirements.

f. A list of the appendices.(definition of terms used in the document, analysis model UML and future extension possibilities)

This is a suggested sequence to go through the SRS:

* Product scope and perspective
* Product function
* User classes and characteristics
* Appendix B
* System features
* Other requirements

## Product Scope

It is proposed to design and build a modernized hybrid cloud architecture for ShiftBank using new container and cloud-native technologies to ensure that their critical operations such as analyzing bank statements can be done with a hybrid cloud pattern where the user and his banking data is present only in an on-premises private cloud, and the analyzer module is hosted publicly.

## References

* 830-1998 — IEEE Recommended Practice for Software Requirements Specifications: <https://ieeexplore.ieee.org/document/720574>

# Overall Description

## Product Perspective

In Shift Bank to avoid any total downtime for any major disruption in the central IT core in a monolithic architecture, it is being planned to design and build a hybrid cloud solution, using new container and cloud native technologies to ensure that their critical operations empower financial services. The hybrid cloud pattern also allows us to separately host sensitive user information on a private cloud, while the analyser module can be hosted publicly. The product developed would be powering the platform to accept payments , transactions and Data Services which would have privacy compliant integrations for customer data. The product offers actionable insights from a customer's bank statement about their income, expenditure and financial history and provides these as a feature and provides an interface for customer and staff.

## Product Functions.

The software is intended for the use of two types of users - customer and staff. The functions for each type of user provided by this software are listed as follows:

1. **User**
   1. Users can add his/her bank accounts and bank statements and track the transactions through the software's friendly interface.
   2. Ask for an analysis report of a particular period of months having useful analysis graphs and data fields.
2. **Partner Banks**
   1. Partner banks can provide insights on a client’s bank account with consent
3. **Admin**
   1. Admin (Staff) has access to the Django Admin Interface.

## User Classes and Characteristics

The following classes are used in the software:

1. User : Every user has their personal details like mobile number, password, Aadhaar number, etc.
2. Account: Account holds information about ifsc code,account number, account opening date, account type, bank name, branch name, branch address and phone number.
3. Transaction: It holds details about a particular transaction like date ,description, debit, credit, balance, category, count of reports.
4. OTPModel: It stores the details for OTP like phone number, OTP type, validity.

## Operating Environment

We are testing this to work on the following tech stack - Python 3.10+, PostgreSQL 14+. We have tested it to run on reasonably recent modern browsers like Google Chrome 88+, Firefox 86+ on amd64 architecture.

## Design and Implementation Constraints

* Login and password for user identification are used as means of security and authentication.
* Users need to upload the statements in pdf/csv format to get the report

## User Documentation

The front-end of the software will be kept sufficiently intuitive and straightforward for anyone who is familiar with browsing the internet through their computer or smartphones.

## Assumptions and Dependencies

It is assumed that the user has access to:

1. An email account
2. A working contact number (mobile)
3. A desktop or laptop or smartphone to access and run the software

Additionally the users need to have a browser through which they will access this software

# External Interface Requirements

## User Interfaces

The user-interface is a browser-based GUI. It begins with a *login* page, where the user logs in with his/her username and password. He is then taken to the correct dashboard, depending on the type of user he/she is. The customer can also sign-up from the *sign-up* page.

After a customer has logged in, they are taken to the dashboard where they can view bank details. A customer could then also select an account and analyze the statements associated with that account. There is also a feature to upload statement details for analysis and get a report. The software offers actionable insights from a customer's bank statement about their income, expenditure and financial history and provides these as a feature to the end users.

## Hardware Interfaces

The product is intended to have two components - a backend API server divided into multiple microservices and the frontend which will be accessed by users through their browsers. Multiple instances of the microservices will be hosted and orchestrated using docker and Openshift. For the backend, it will be handled by powerful computing resources having powerful processors and physical memory. However, the users should have a desktop/laptop(preferably with 4gb ram or more) or a smartphone(preferably with 2gb ram or more) for smoothly using this software through their browser.

## Software Interfaces

* The backend of the software will consist of the various databases based on the SQL framework, and deployed using the PostgreSQL database management.
* The frontend will consist of a graphical user interface developed on Javascript, HTML and CSS made using React Libraries.
* OperShift Routers routes the HTTPS request to the appropriate load balancer services for each microservice deployment.
* The product is containerized using Openshift.

## Communications Interfaces

There will be a direct communication between the backend and frontend of this software using standard HTTPS (HyperText Transfer Protocol Secure) using REST API.

# System Features

## Login and Creating an Account

**4.1.1** Description and Priority  
 This is the first task of the customer to create an account. Only after the customer has created an account can they access their Bank account(s) and their transaction details.

**4.1.2** Stimulus/Response Sequences

The customer has to enter their personal details like name, contact details, and address. The software will create an account for the customer and store the data.

**4.1.3** Functional Requirements

**Req 1:** Prompt the user to log in and give an option to create an account if they don’t have one.  
**Req 2:** Ask for the user details for necessary information.  
**Req 3:** Allow the user to modify the details if required.

**Req 4:** Allow the user to add multiple bank accounts

## Get Account Details

* + 1. Description and Priority  
        The customers can view their bank accounts on ShiftBank after logging in. It will also show various other features like the number of reports, banks analyzed, and average balance.
    2. Stimulus/Response Sequences  
        The customer has to log in to view their account details. The dashboard includes a list of bank accounts, some quick links, and the status of previous reports.
    3. Functional Requirements  
        **Req 1:** Ask the user to select a particular bank to analyze their transaction  
        **Req 2:** Prompt the user for selecting the time period in months in between which the user wants to analyze.

## Analyze Account

* + 1. Description and Priority  
        Analyzes the credit and debit transactions from a selected bank and generates reports corresponding to that.
    2. Stimulus/Response Sequences  
        The user selects the bank account, the time period between which they want to analyze the transaction. The analyzer then generates the list of credits/ debits in that time period.
    3. Functional Requirements  
        **Req 1:** User selects a bank account to analyze.  
        **Req 2:** Prompt the user for selecting the bank account they want to analyze.

**Req 3:** Various details like Overview, Recent Transactions, and Monthly Summary are displayed along with the time period. The user is also given the option to download this analysis in pdf format.

## Upload Statement for Analysis

* + 1. Description and Priority  
        Asking the user to upload a bank statement to analyze.

**4.4.2** Stimulus/Response Sequences  
 The user can upload a bank statement in .PDF or .CSV format to pass it through the analyzer. After the action, report gets generated specifying the following details–

* + Payment mode bar graph
  + Expenditure details pie chart
  + Loan details graph
    1. Functional Requirements  
        **Req 1:** Ask the user if they want to generate an analyzer report.  
        **Req 2:** Based on their choice, make an appropriate API call and reflect the changes in the graphs.

## Send and Verify OTP

* + 1. Description and Priority  
        Send OTP to mobile and verify from user. This is required while registering users and also delivery partners verify OTP when delivering orders to customers to make sure it’s delivered to the correct person.
    2. Stimulus/Response Sequences  
        The software prompts from an OTP that was sent to the user. The software then checks if the OTP entered is indeed the one that was sent by it.
    3. Functional Requirements  
        **Req 1:** The user enters his phone number to register

**Req 2:** The user is directed to the next page where an OTP is sent to the entered mobile number.

**Req 3:** The user is asked to verify the OTP sent

**Req 4:** Once the OTP is verified, the user is directed to the create account page where he is asked to enter other details to create their account.

# Other Nonfunctional Requirements

## Performance Requirements

* The system response should be fast. Time required for fetching the user details from the database and displaying them on the frontend should ideally be less than 1 second.
* Also the system should immediately relay the issues to the admin.
* When a customer adds a new bank account, it should be immediately available to analyze statements.
* The OTP sent for verification should reach the user in a few seconds.

## Safety and Security Requirements

The user will be able to report any bugs they encounter to the admin so that the developers can fix it in the next release. The user details will be kept completely private and will be accessible only to the user himself. Additionally every user will be allotted a unique user ID that will be known only to them and will act as the gateway to access his account. All the data will be protected in order to prevent any possible data theft.

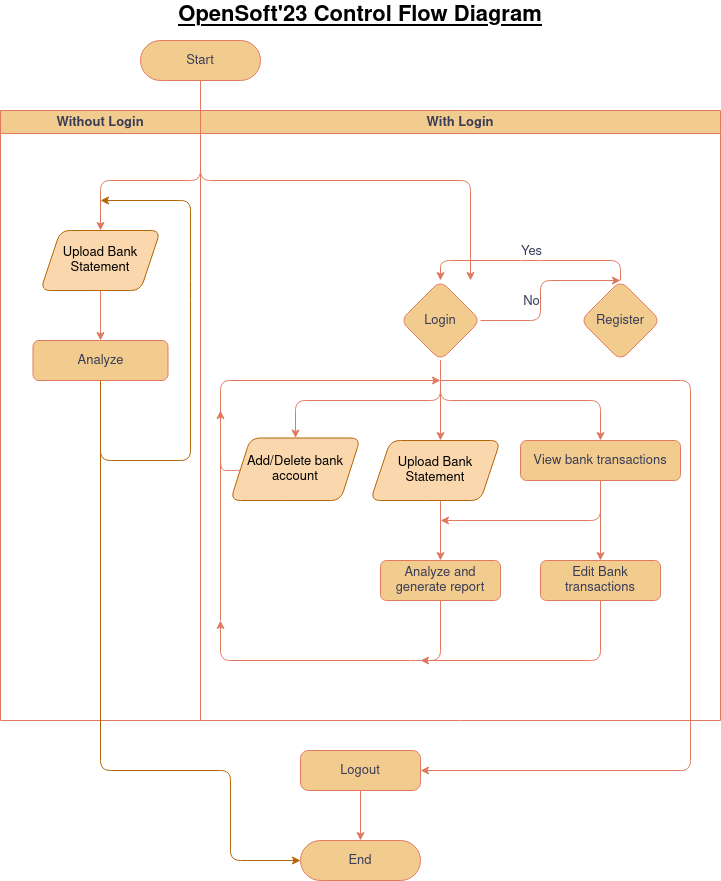
The transactional information of a user’s bank account is also stored on a private cloud, making it much more secure. It can be accessed by the user and the corresponding bank for the account.

## Software Quality Attributes

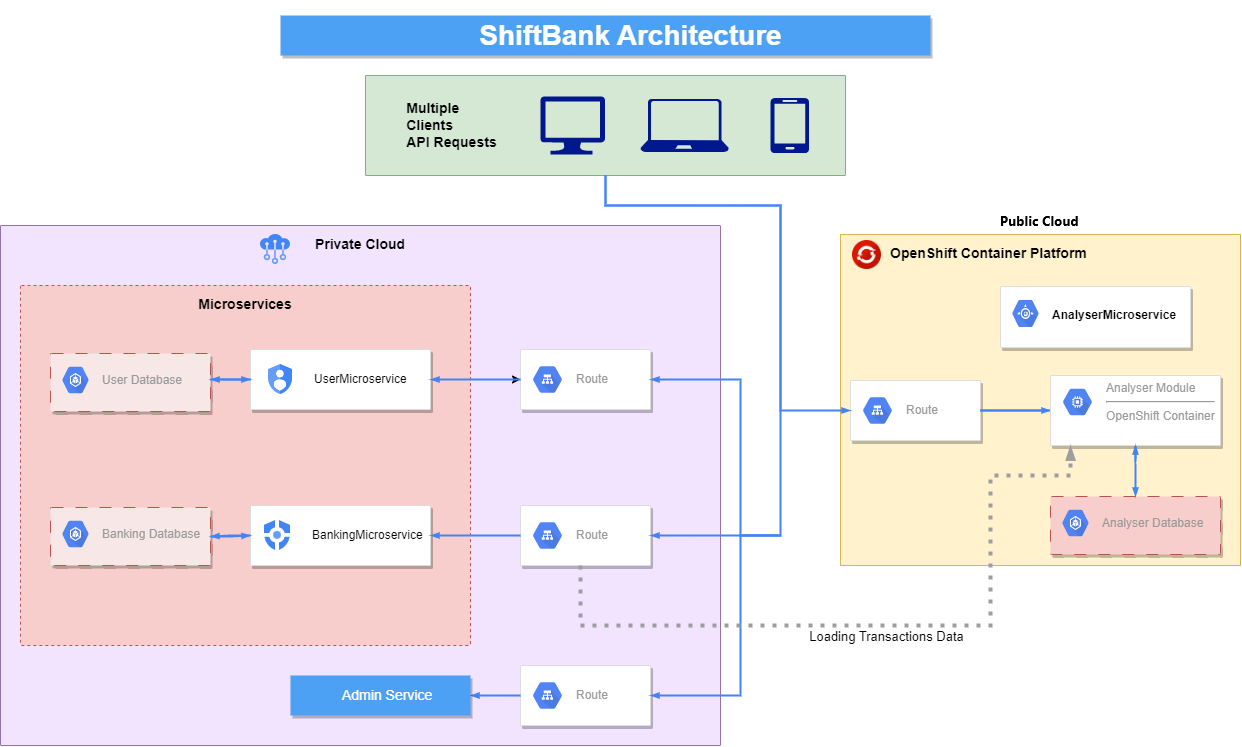
The software provides a well-designed, easy to use interface that allows anyone to use the software.

# Design Diagrams

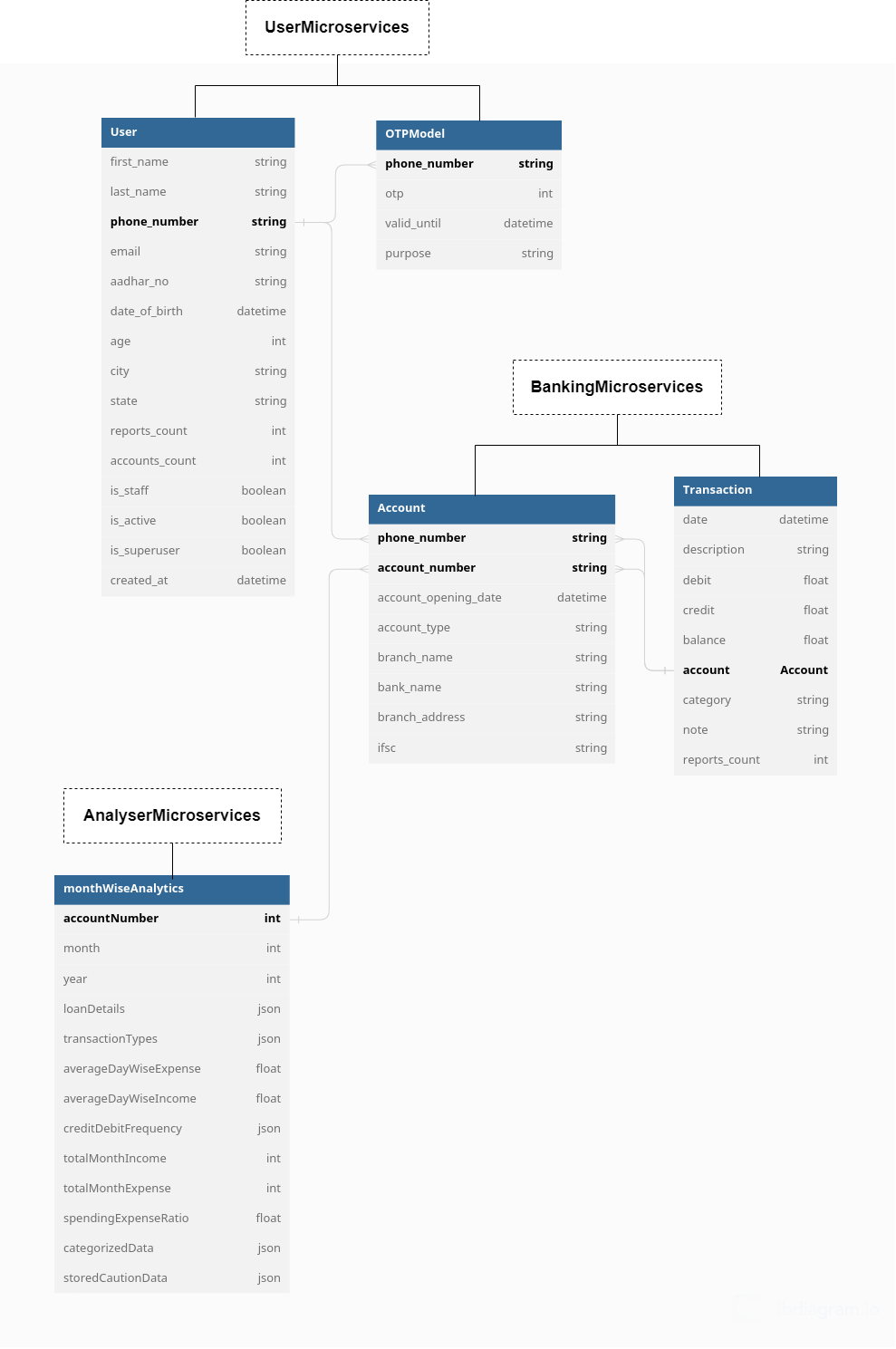
## Control Flow Diagram



## Data Flow Diagram



## Class Diagram

****

## Analysis Sequence Diagram