Forex Management System

Technical Design Document



# 

Table of Contents

[*Technical Overview:* 3](#_Toc138071313)

[*Activity Diagram of workflow:* 4](#_Toc138071314)

[*Architecute Document:* 5](#_Toc138071315)

[*Service Details:* 6](#_Toc138071316)

[*Database design:* 6](#_Toc138071317)

[*Component Diagram:* 7](#_Toc138071316)

[*Others* 8](#_Toc138071318)

[*Database connection:* 8](#_Toc138071319)

[*Maker & Checker flow:* 9](#_Toc138071322)

[*Fund Transfer flow:* 16](#_Toc138071322)

# Technical Overview:

Our client HDFC wanted us to create a web application for managing Forex related

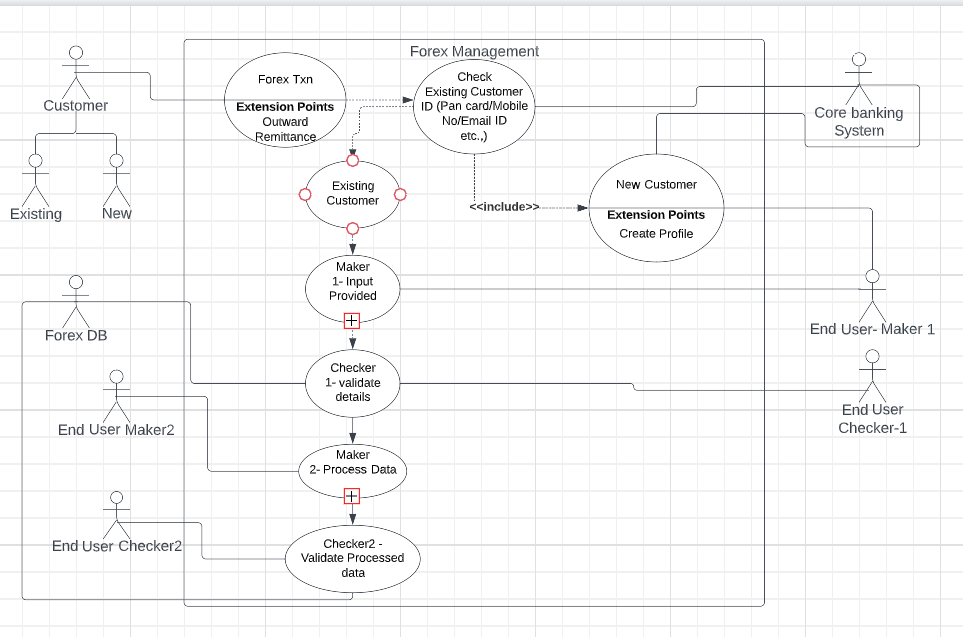
Transactions. The application should automate all the forex transactions for managing complex

Scenarios and implement following features:

* Allow acceptance of foreign currency
* FC Travelers Cheques management
* Handling Foreign Inward and Outward remittance
* upholding accounts in foreign currency
* Import Letter of Credit
* Handling Foreign Currency cash balances
* Foreign Currency Loans
* Fixed Deposit creation in Foreign Currency
* Investment in Foreign Currency – Forward Contract

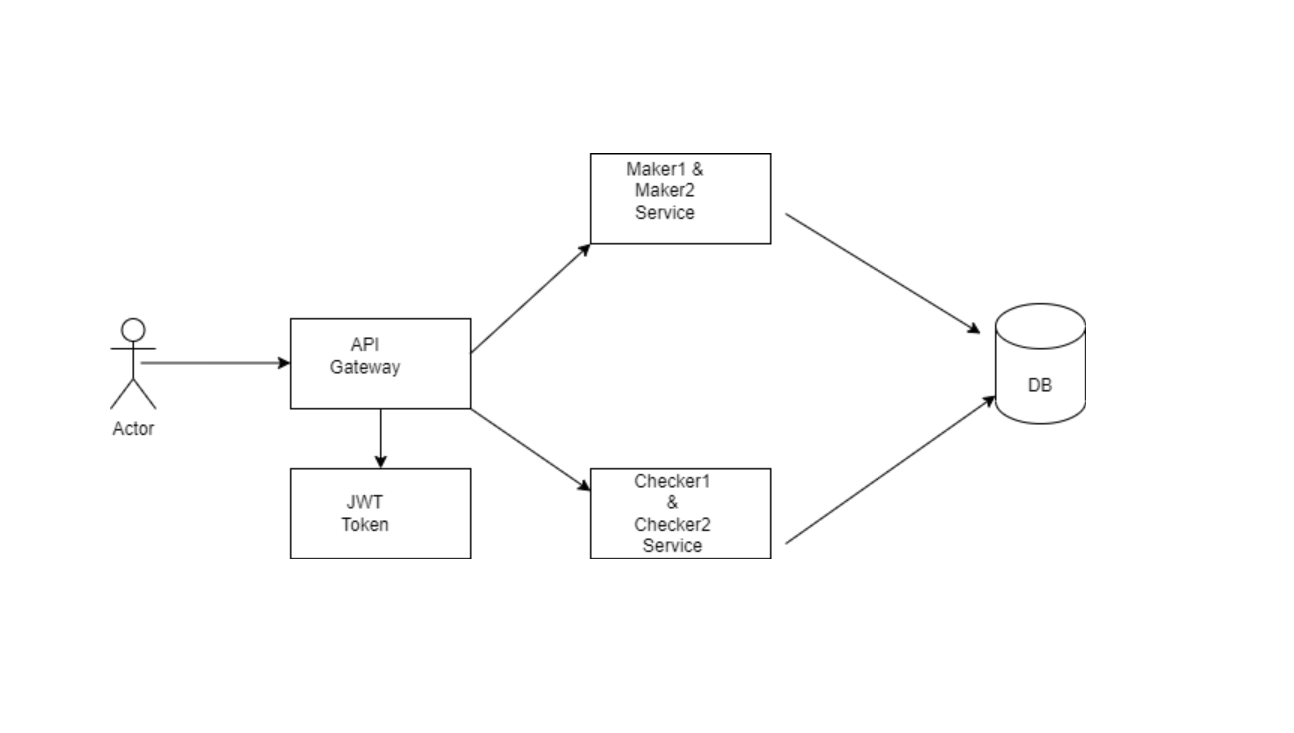
# 

# Maker and Checker workflow:

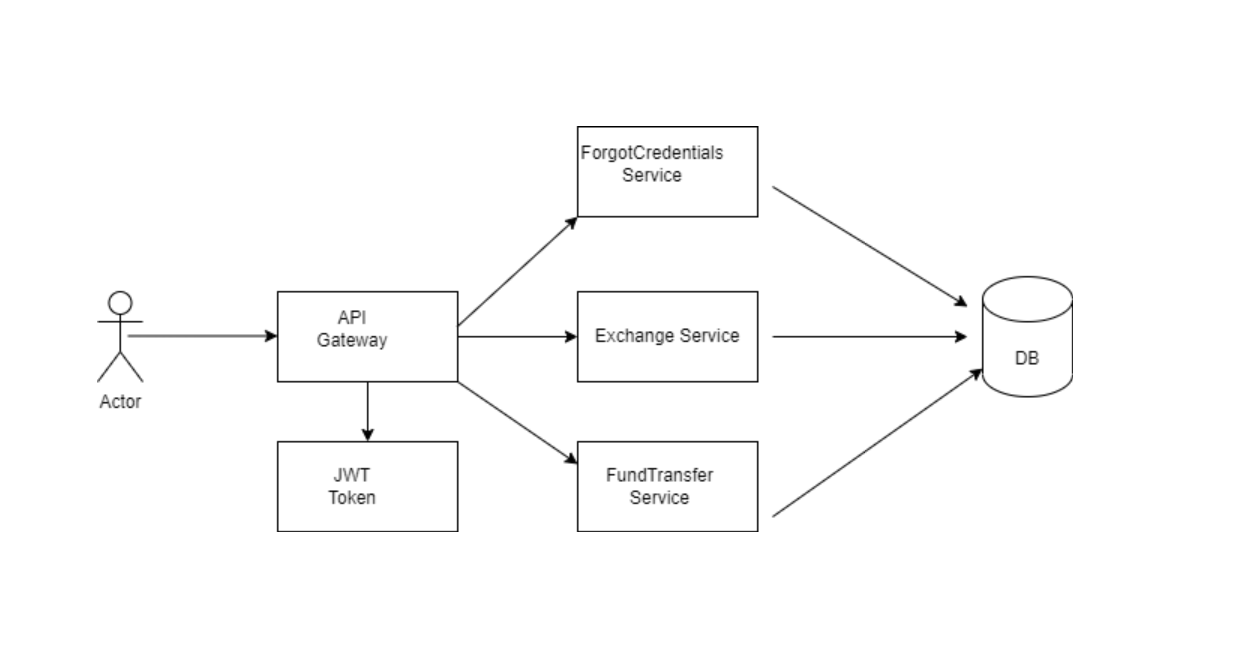


# Architecture Document:

1. **Maker and Checker design:**



1. **Fund Transfer Design:**



**Service Details.**

Based on the above requirment we have divided it into 2 microserives:

* **Maker1&Maker2:** login for making entries at the Frond end level based on customer documentation.
* **Checker1&Checker2:** login for validating and authorizing the Maker-1 inputted details

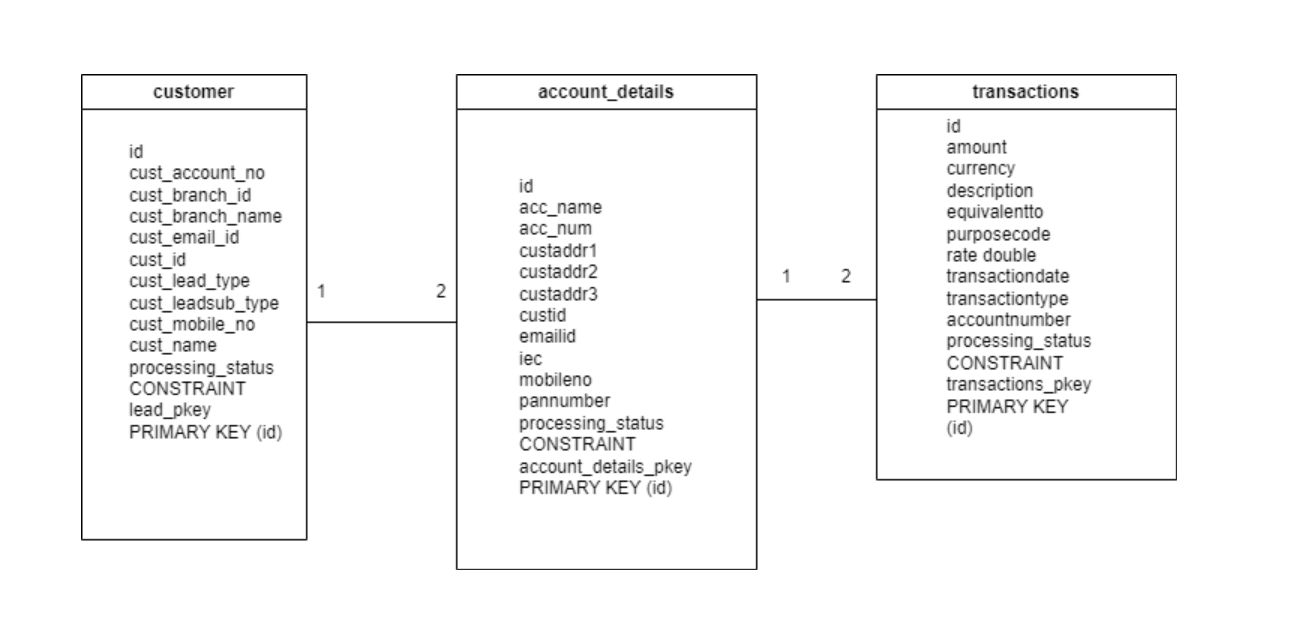
Based on the FundTransfer requirment we have divided it into 3 microserives:

* **Exchange Service :** This service used to get dynamic currency rate.
* **ForgotCredentials Service :** To reset password and login ID.
* **FundTransfer Service:** This Service used in order to transfer the fund from one account to another account.

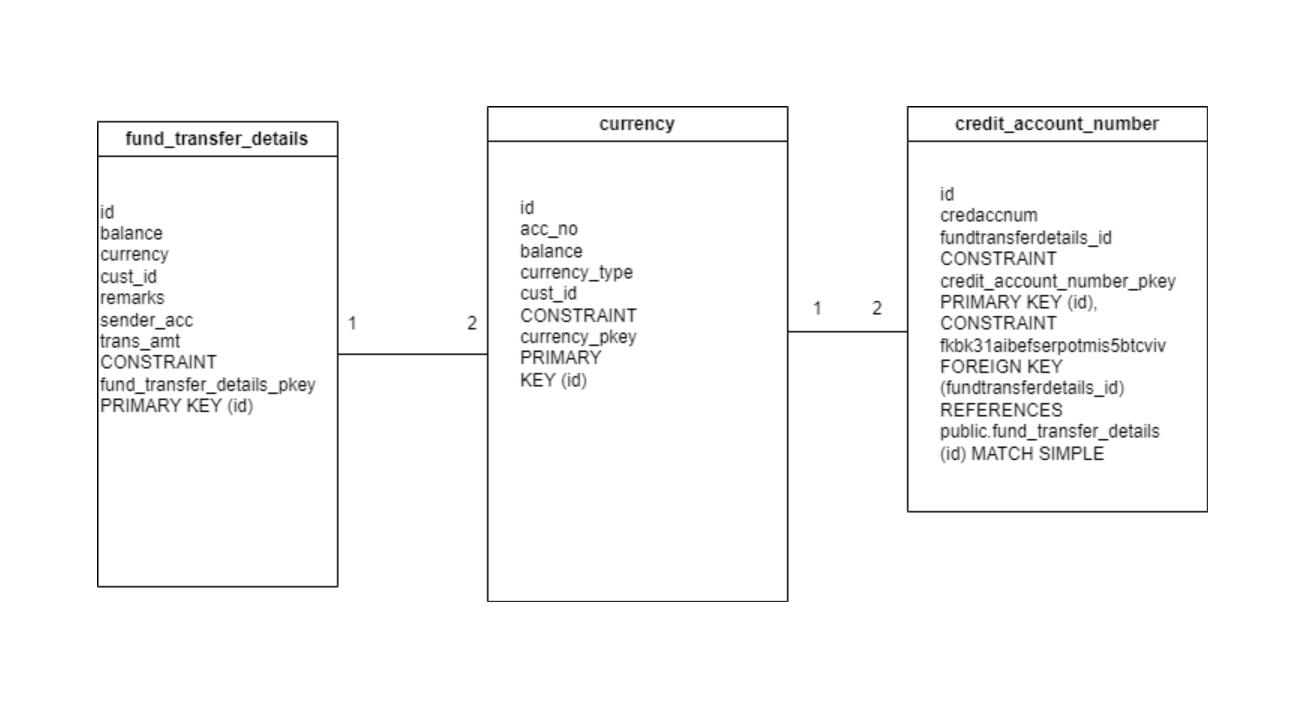
# 

# Database design:

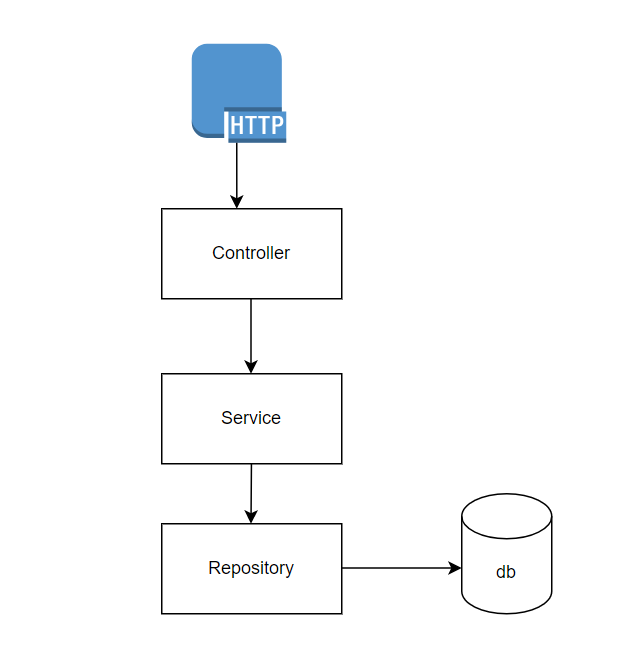
1. **Maker and Checker:**



1. **FundTransfer:**



# Component Diagram:



Normal flow works from http request to the Controller endpoint and Service layer and Repository and finally to the DB.

# Others

Exception handling:

To handle exceptions consistently across microservices, we can use a combination of @ExceptionHandler and @ControllerAdvice annotations. The @ExceptionHandler annotation is used to detect certain runtime exceptions and send responses according to the exception. In this article we’ll Spring MVC project to show how to intercept and define exceptions, we’ll look at how to define method level as well as the class level exception.

Security:

REST APIs must be secured by requiring an [Access Token](https://microservices.io/security/access-token.html)/JWT

JSON Web Token (JWT) is an open standard ([RFC 7519](https://tools.ietf.org/html/rfc7519)) that defines a compact and self-contained way for securely transmitting information between parties as a JSON object. This information can be verified and trusted because it is digitally signed. JWTs can be signed using a secret (with HMAC algorithm) or a public/private key pair using RSA.

# Database connection:

Configure the database connection and set up appropriate transaction management using Spring Data JPA. Will configure the database design once the business requirement document is finalized.

Database setup and transaction management:

Configure the database connection and set up appropriate transaction management using Spring Data JPA.

spring:

datasource:

url=jdbc:postgresql://localhost:5432/forexMgn

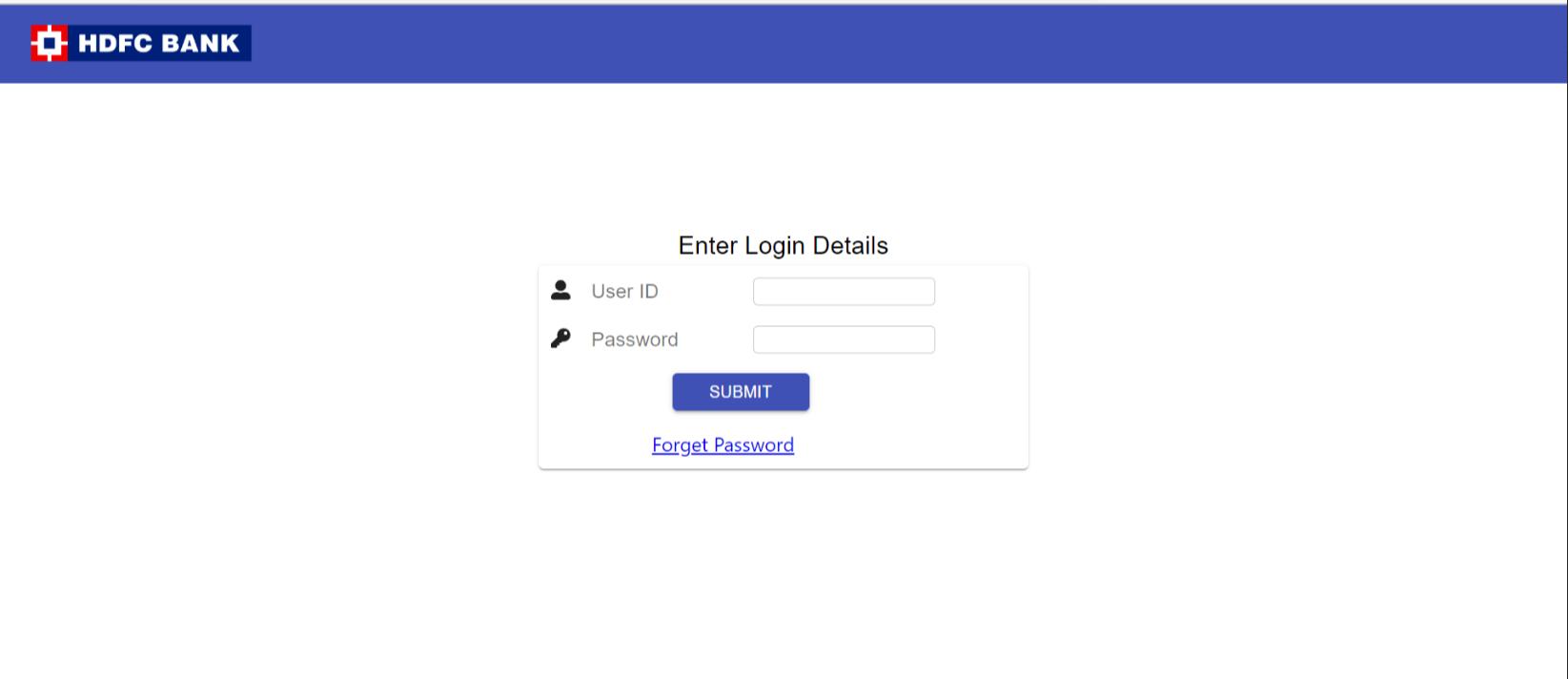
username=postgres

password=Postgre

dialect = org.hibernate.dialect.PostgreSQLDialect

# **Maker and checker flow**:

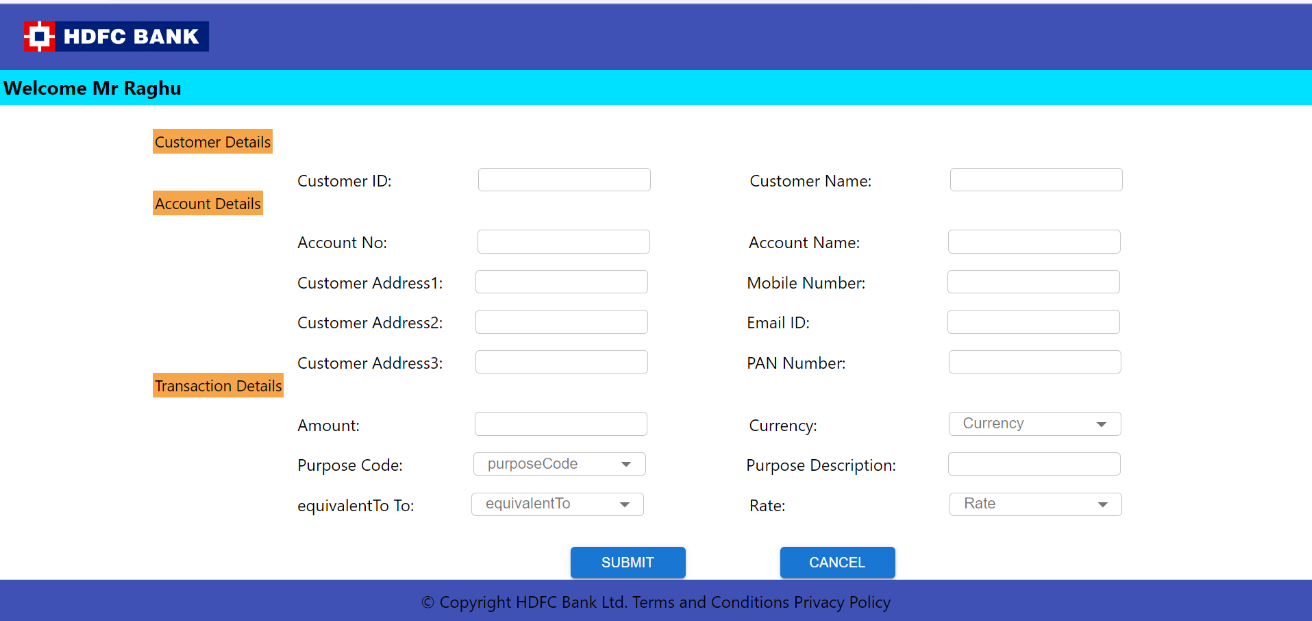
This is logging screen where Maker and checker can login into the system separately.



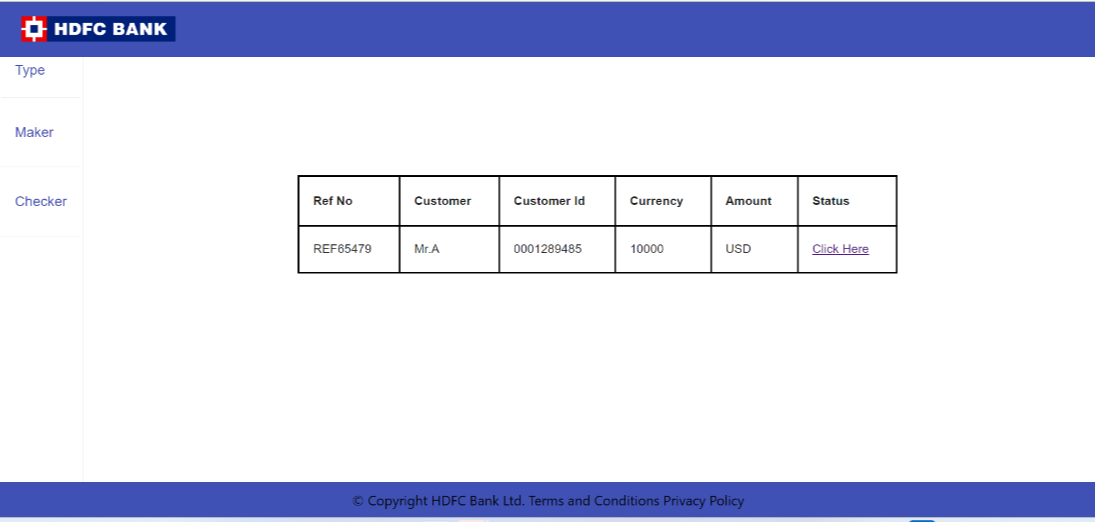
1. Maker can add new case by adding required data.
2. Also Maker can see existing case created already.



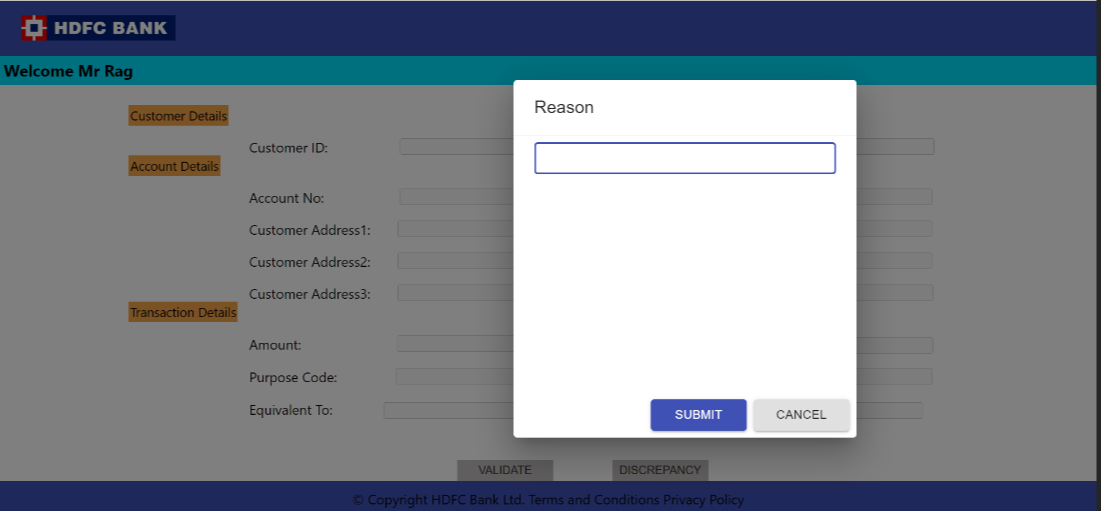
1. This is new screen where maker can create new case with required data.

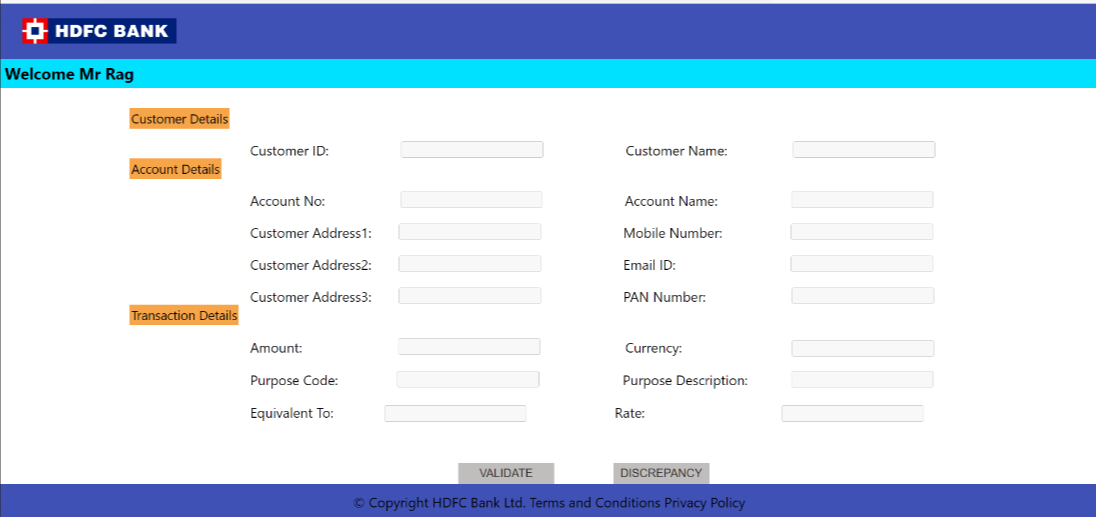


1. After adding the data, maker can see the added data and can update the fields.

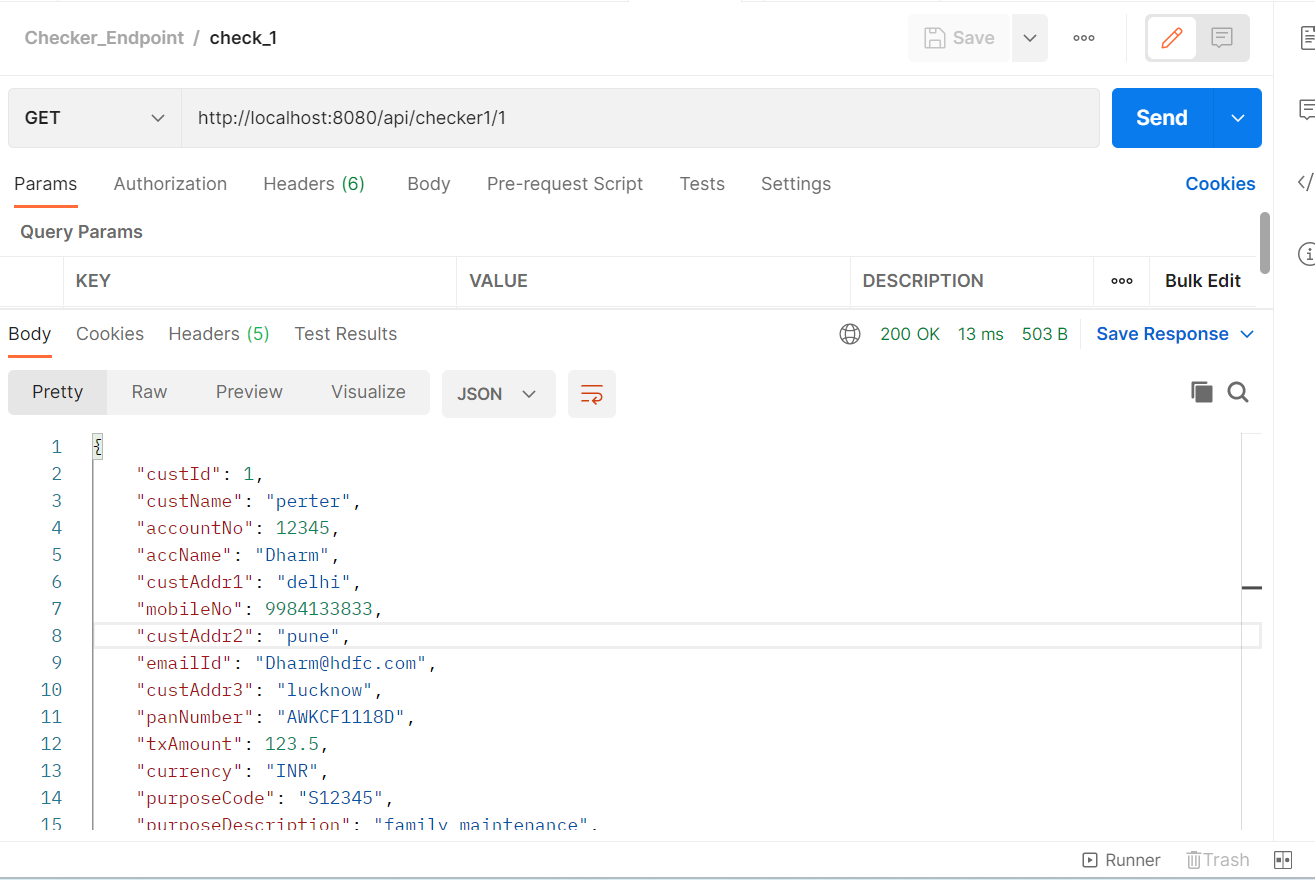


1. Once the checker logged in into the system, if there is any discrepancy, checker can raise the discrepancy or validate the form if data is proper and can submit the form.

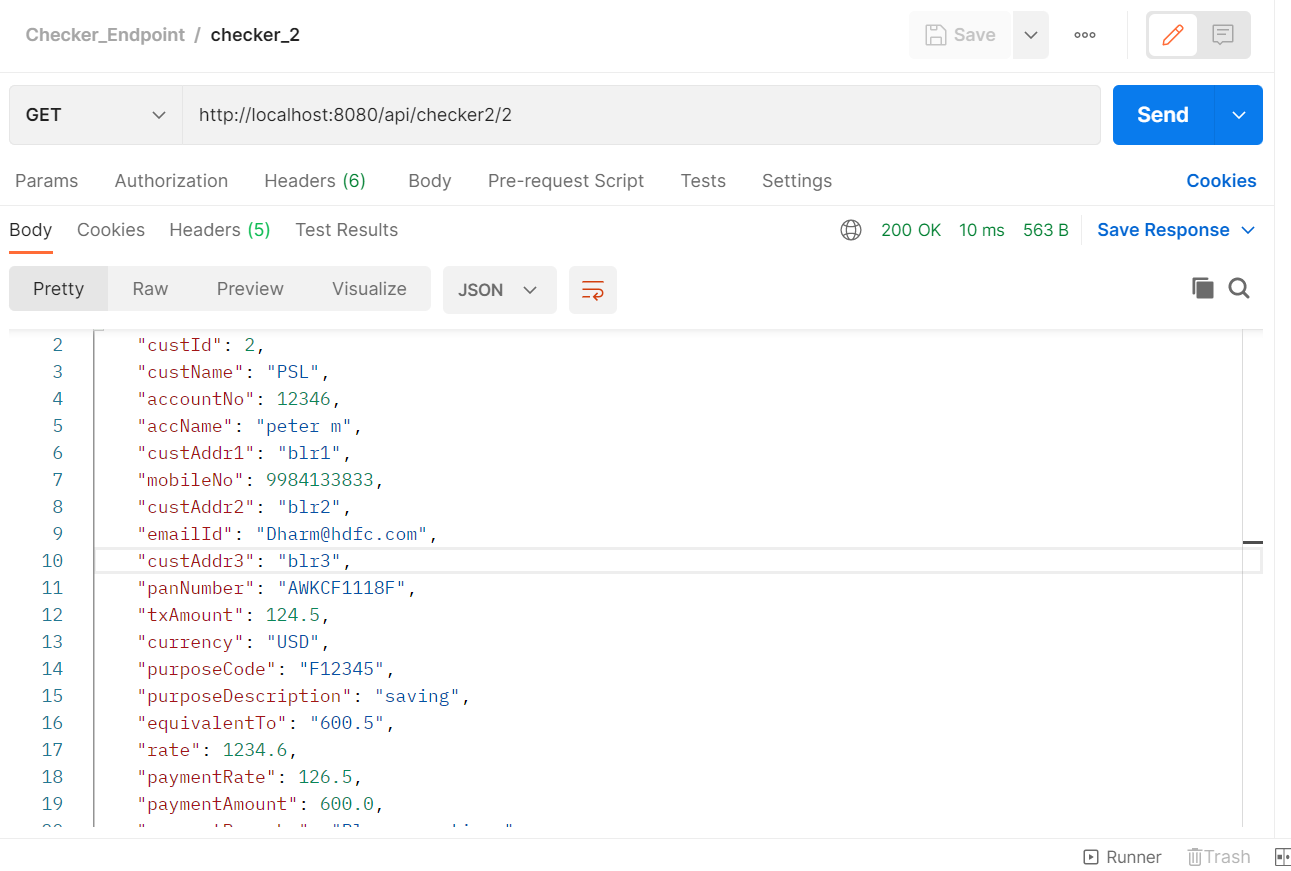




1. Checker 1 functionality for getting details about customer.



1. Checker 2 functionality for getting details about customer.



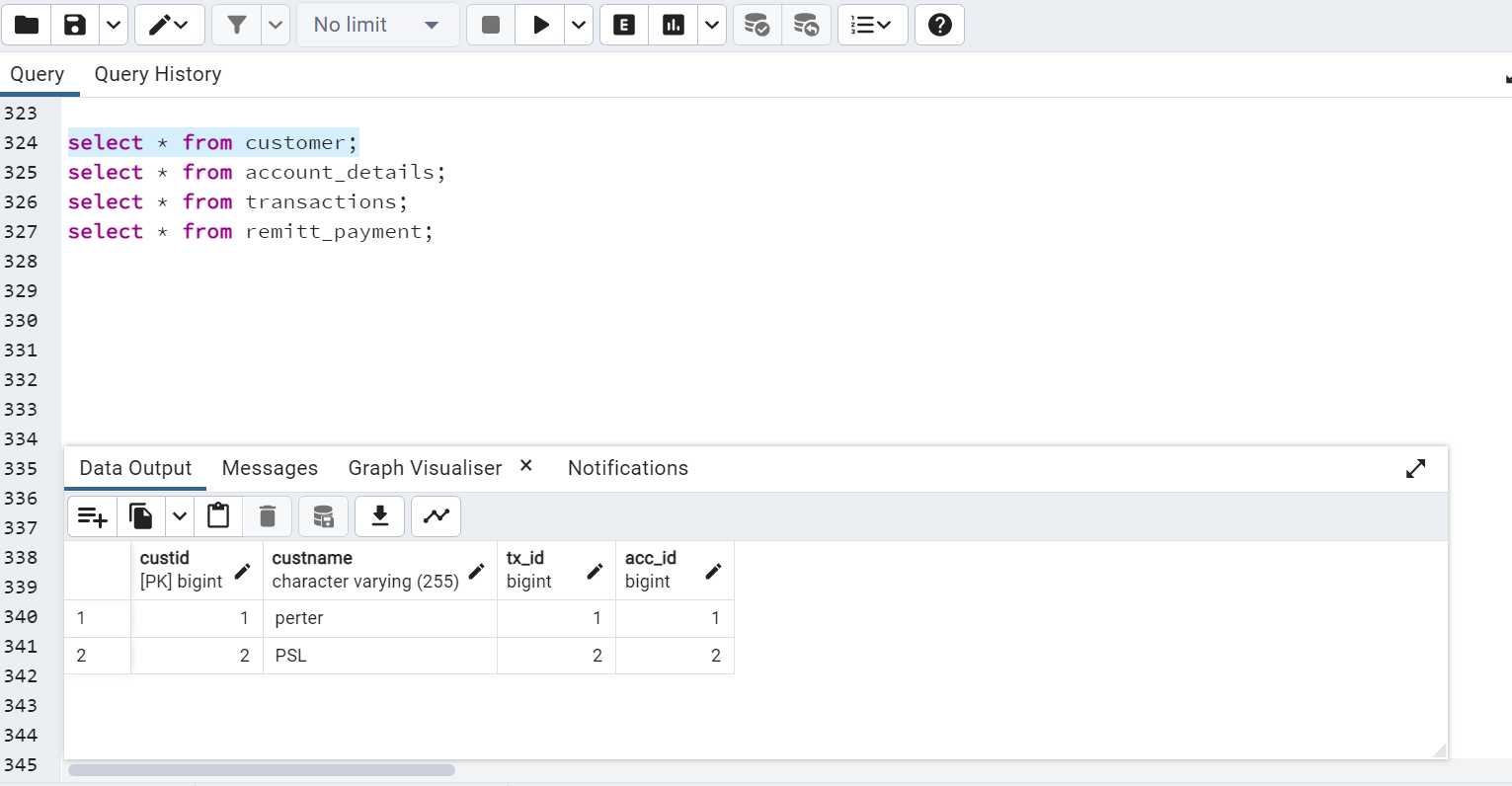
**Database details**

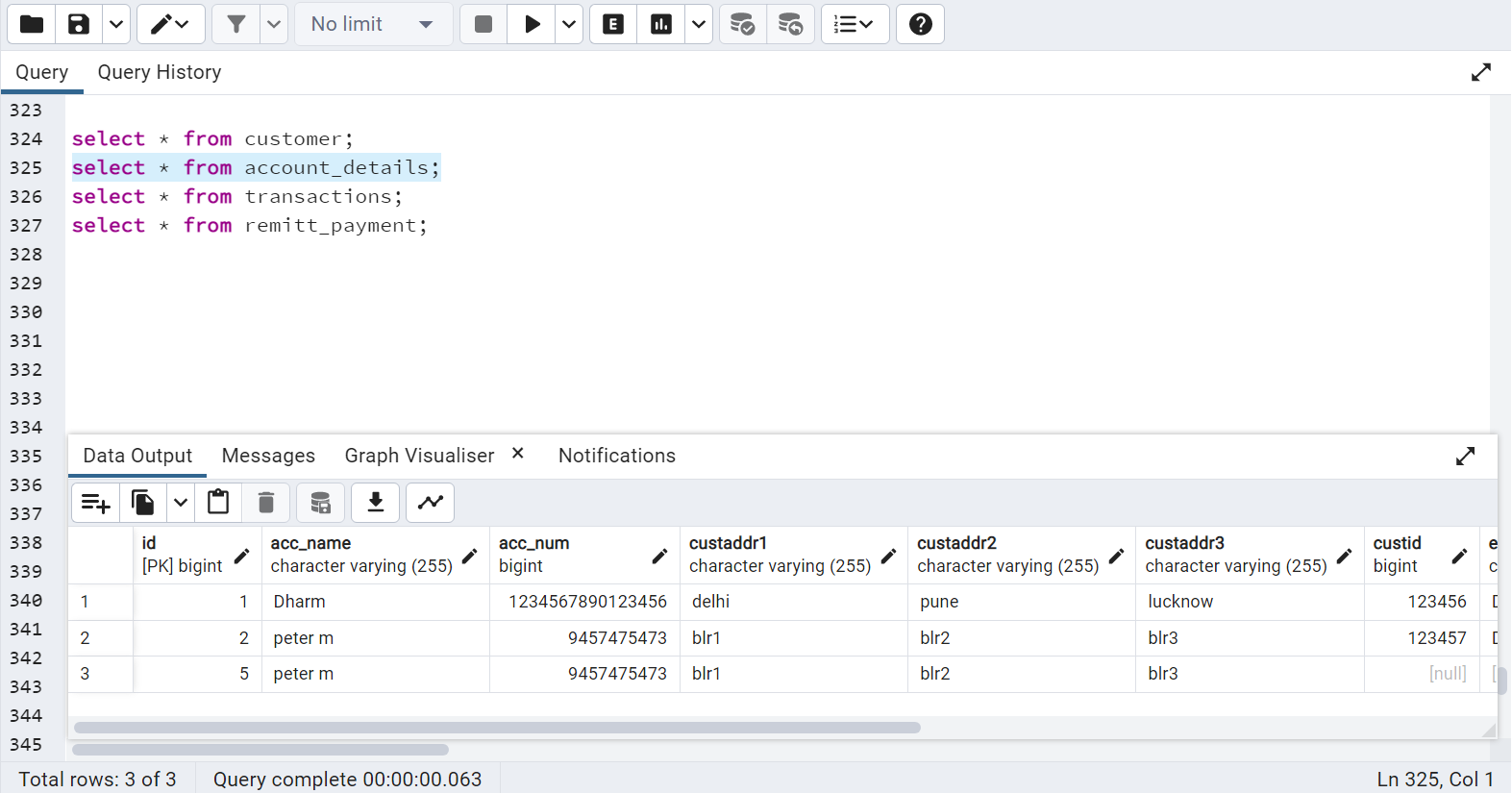
Table name:

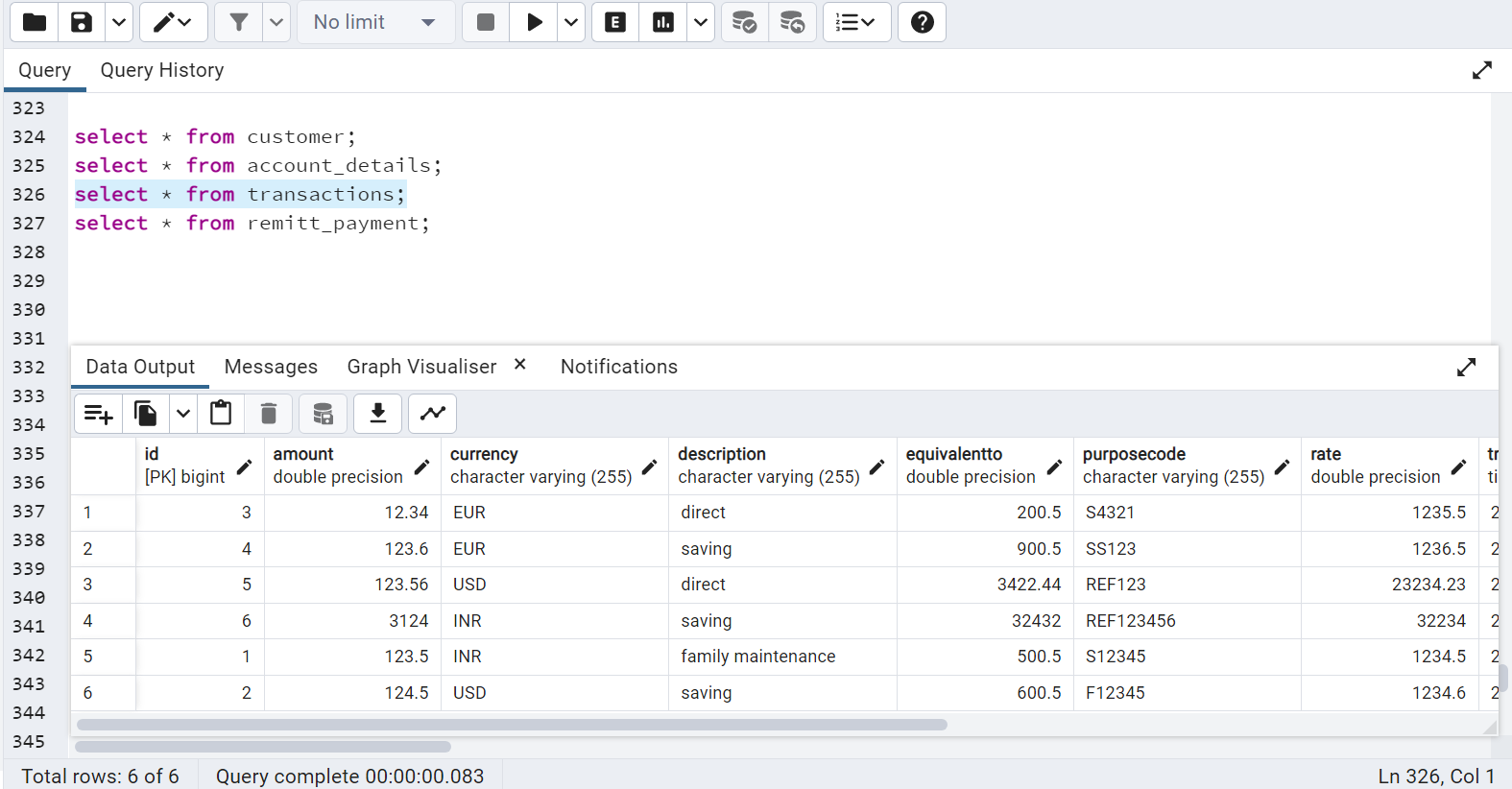
customer

account\_details

transactions







Maker1 and Maker2 service name: **ForexmanagementSyetem**.

Checker 1 and checker 2 service name: **checkerFunctionalities**

**Steps to run the application.**

1. Import project as a maven project in the IDE.

2. Create database in the PostgreSQL with the username and password.

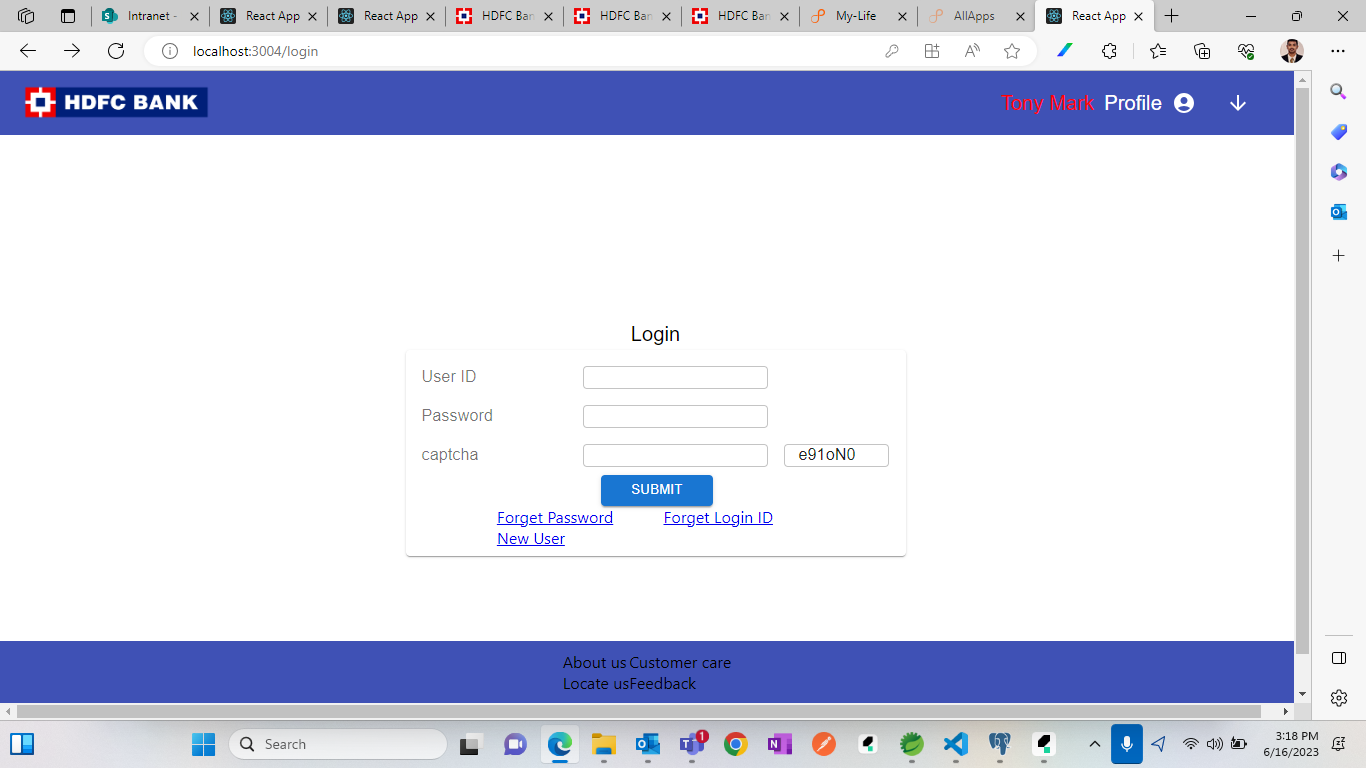
3. Add username and password and connection URL in the application.properties file

4. Start the service.

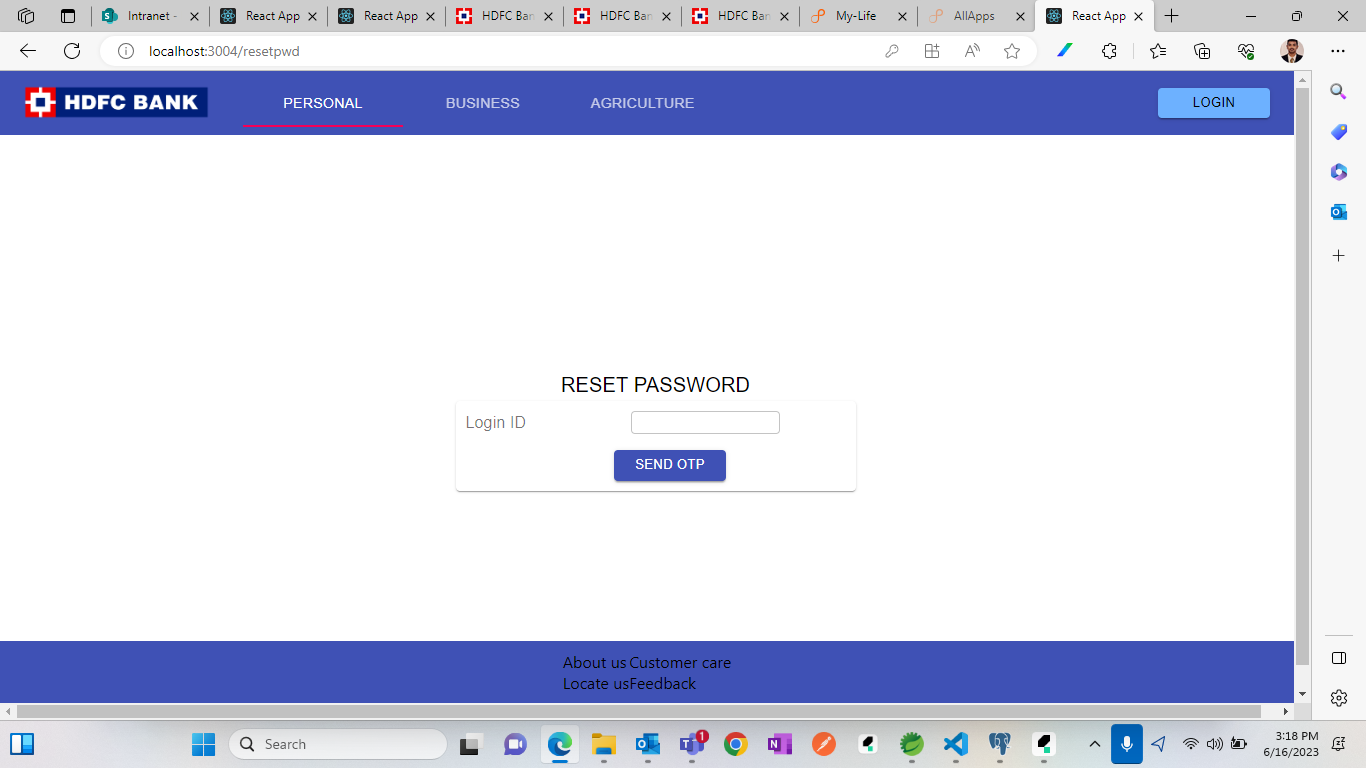
5. Import the postman collection in the postman and run the API.

# **Fund Transfer:**

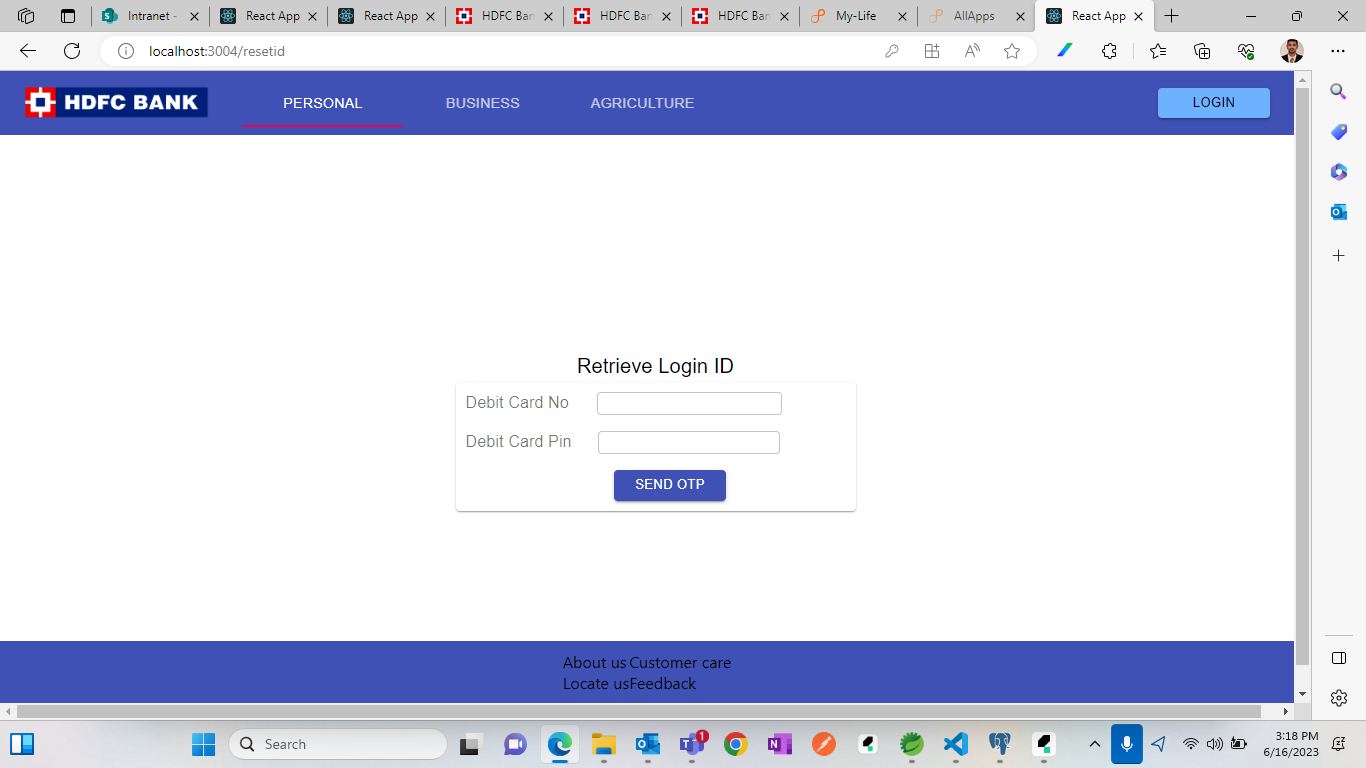
1. This is User login page where User can login into system.



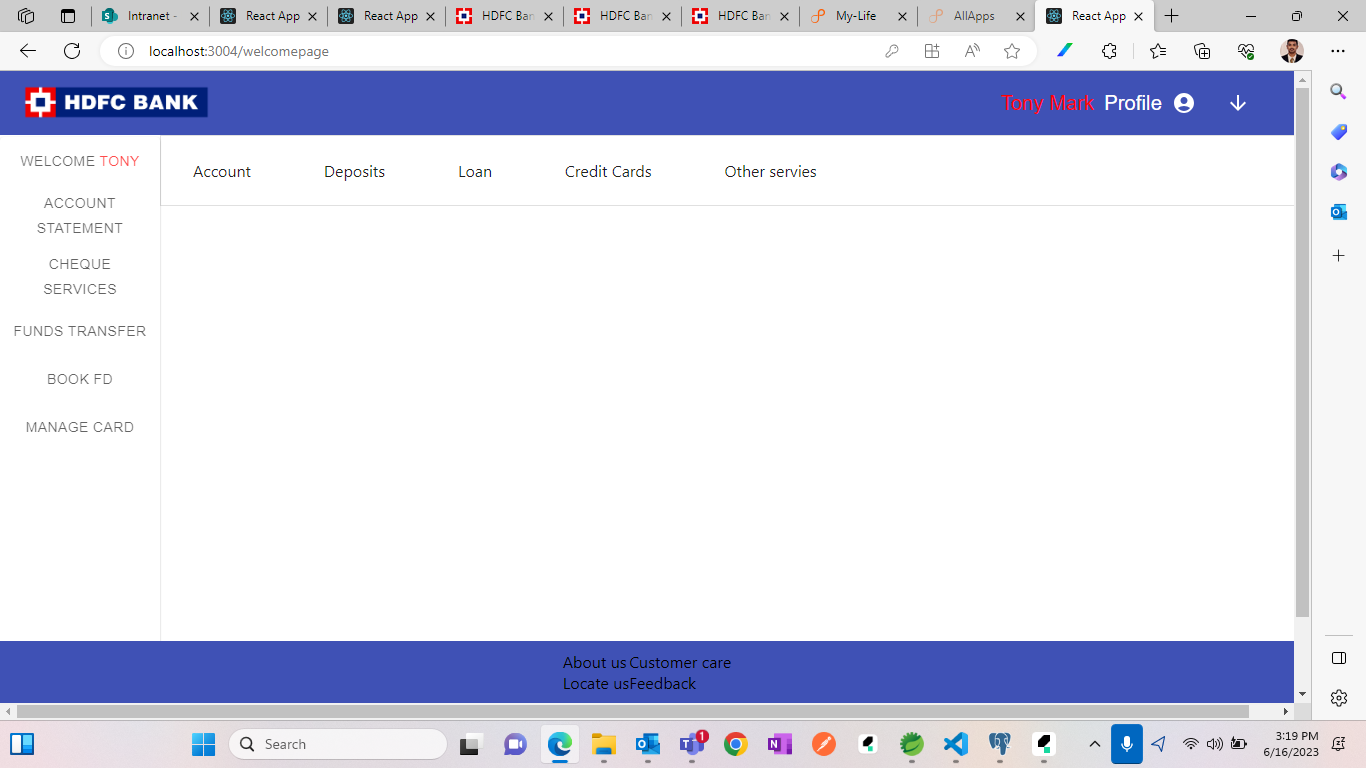
1. In case User forgot his password, User can reset the password.



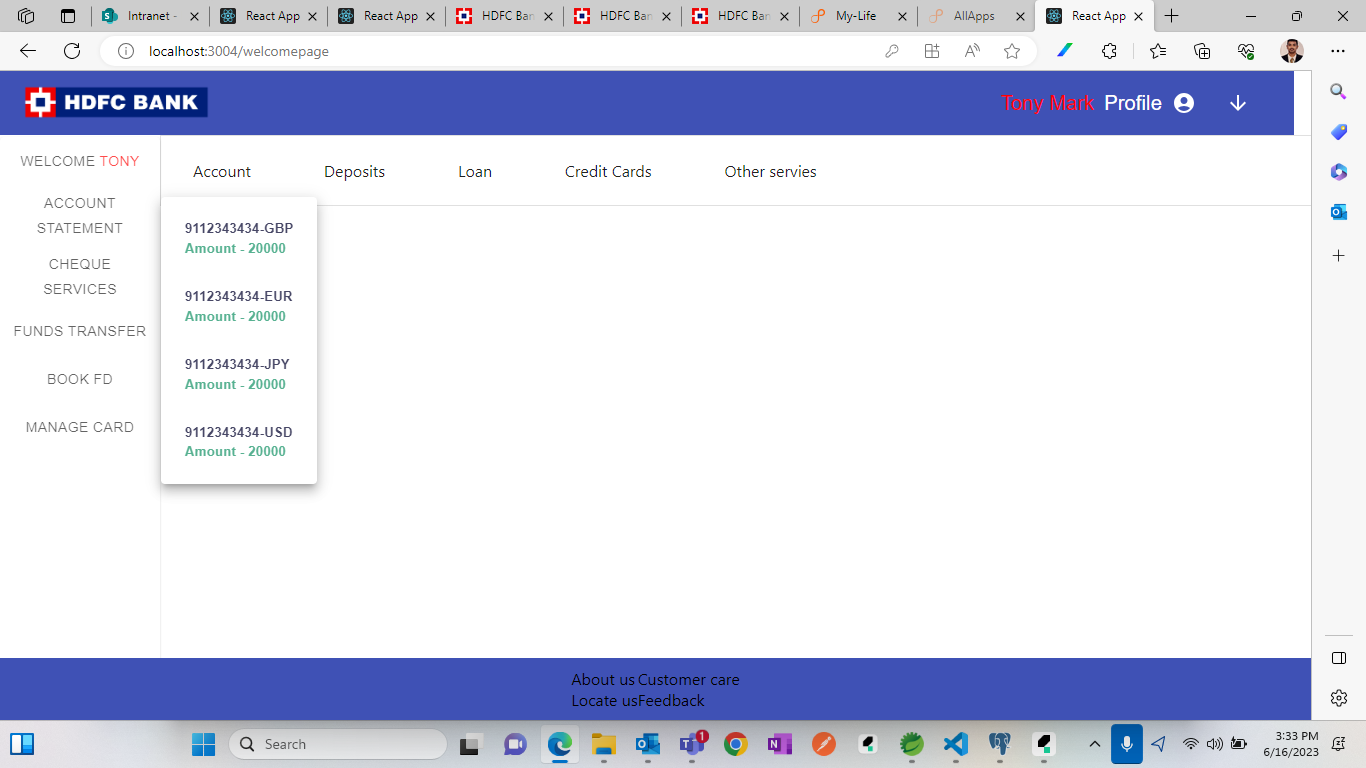
1. User will get an OTP on his mobile number after hitting the button.
2. User will get option to reset his password and can use latest password to login into the system.
3. User have an option too, to get his login Id with this system.
4. User must enter his Debit card no and pin to get his login Id via OTP on his registered mobile number.



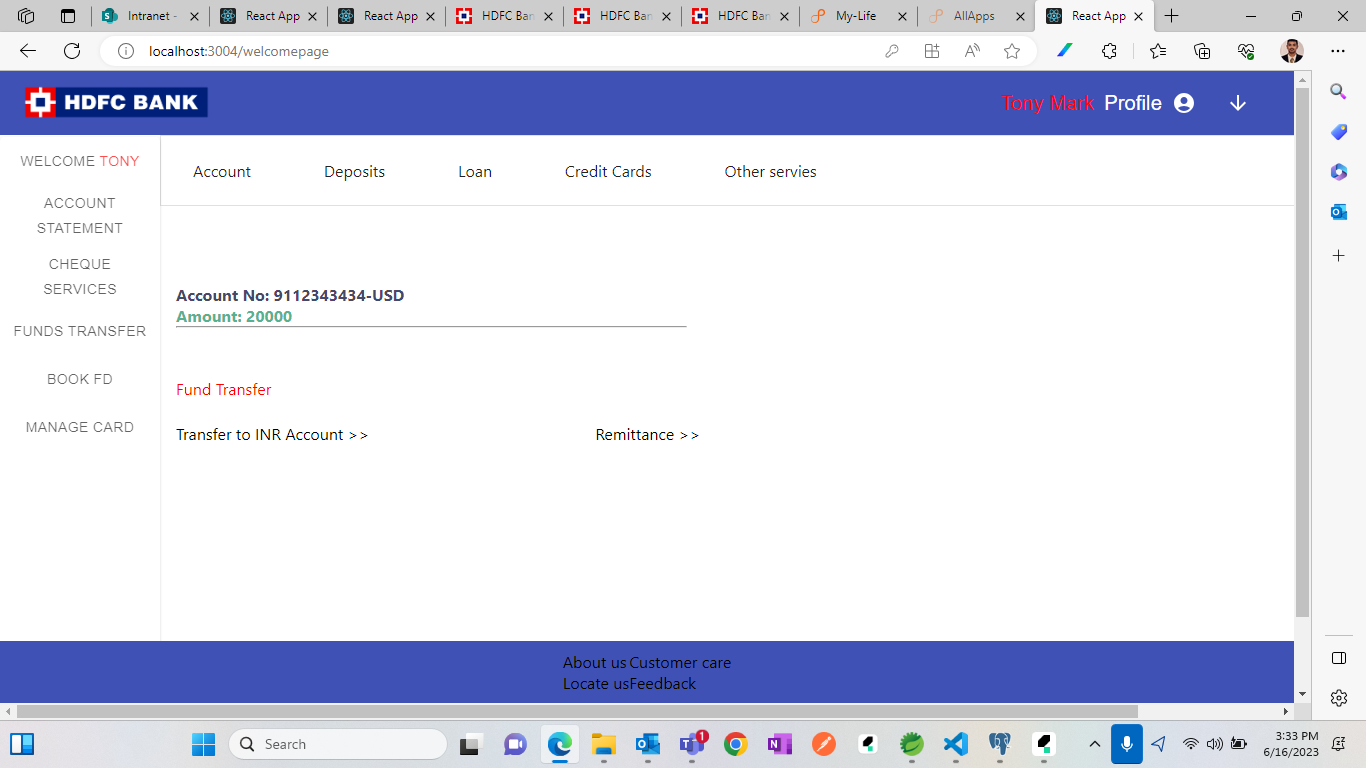
1. This is successful login screen after logged in.

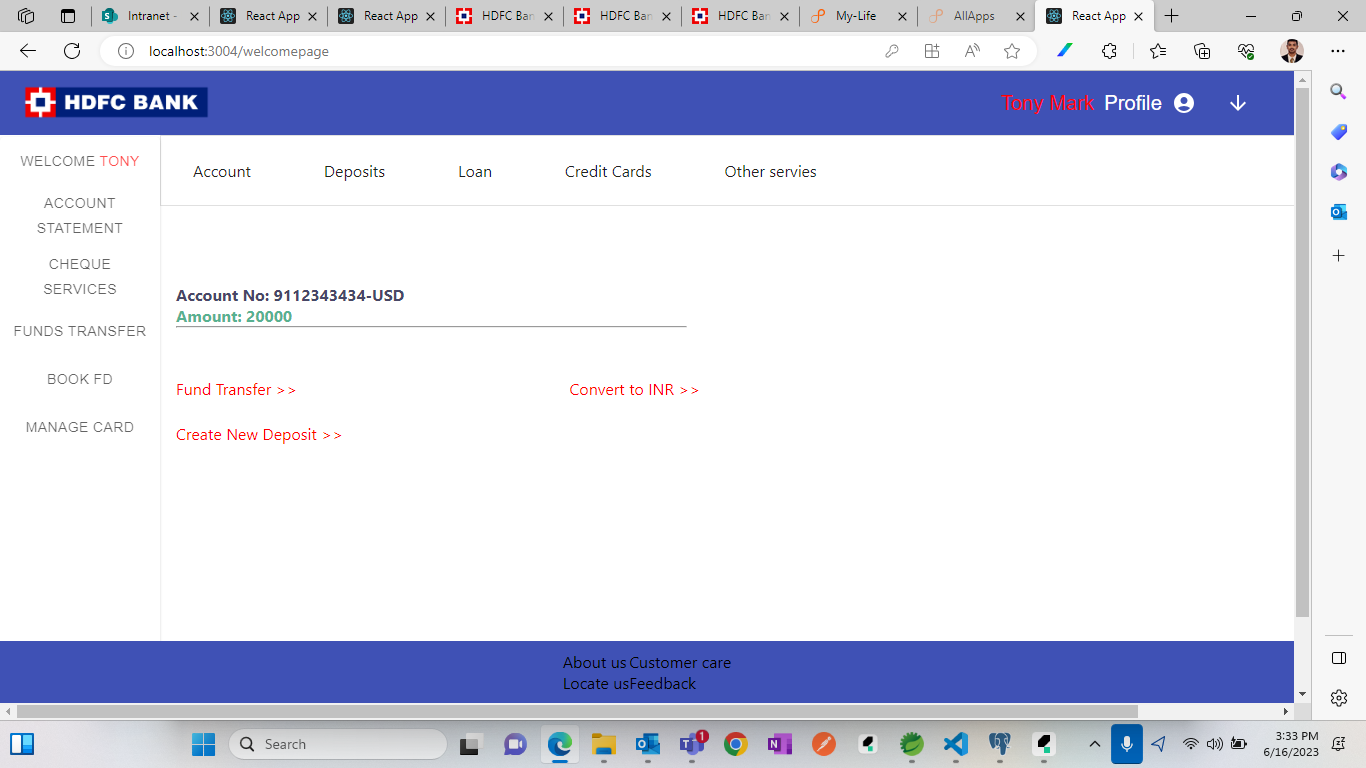


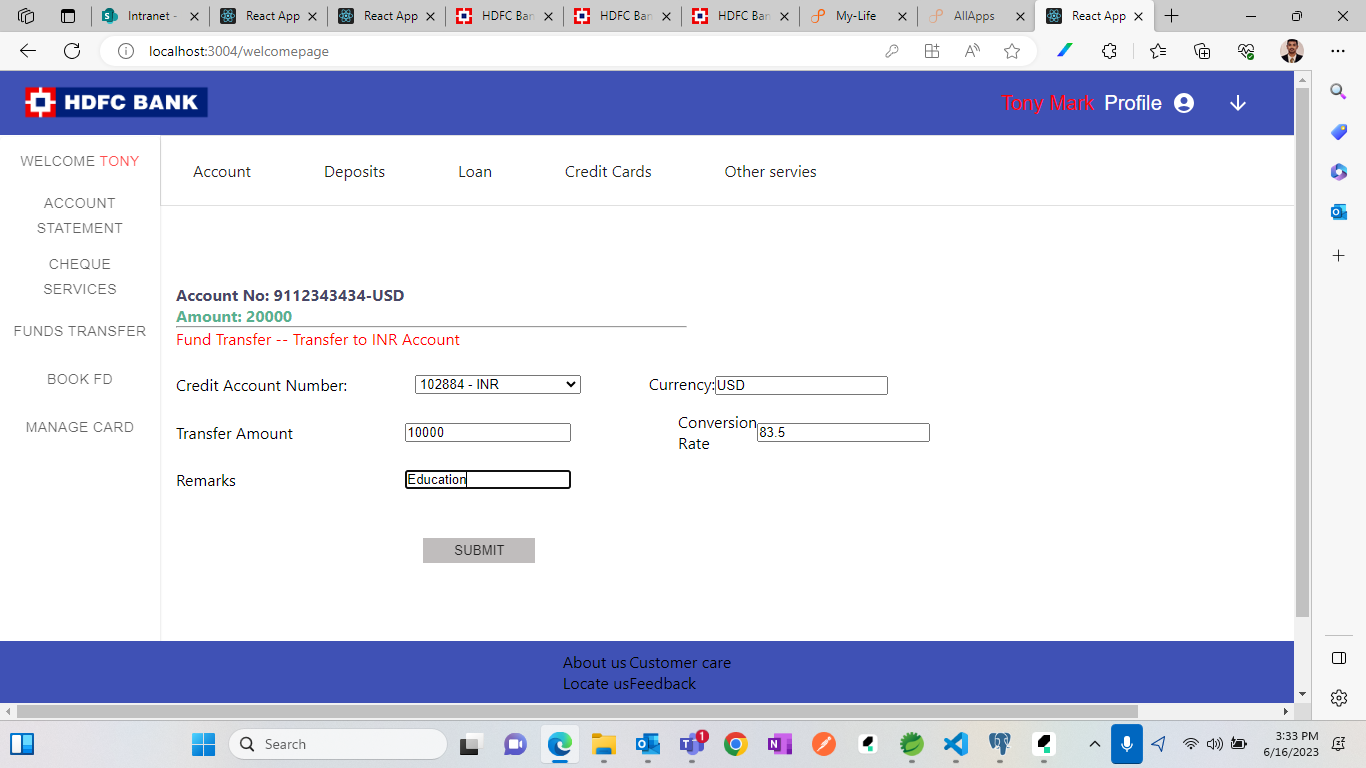
1. When we clicked on Account tab, we will get the respective account balance with currency.

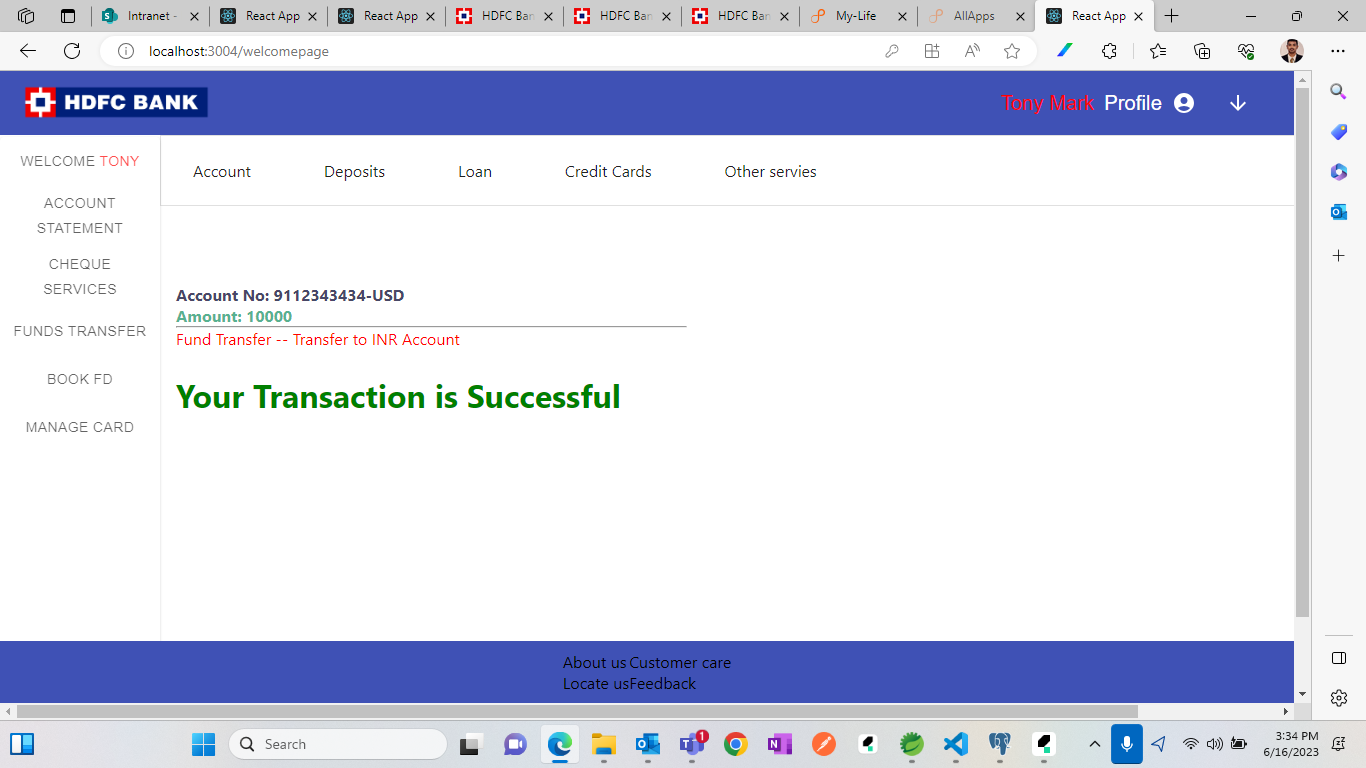


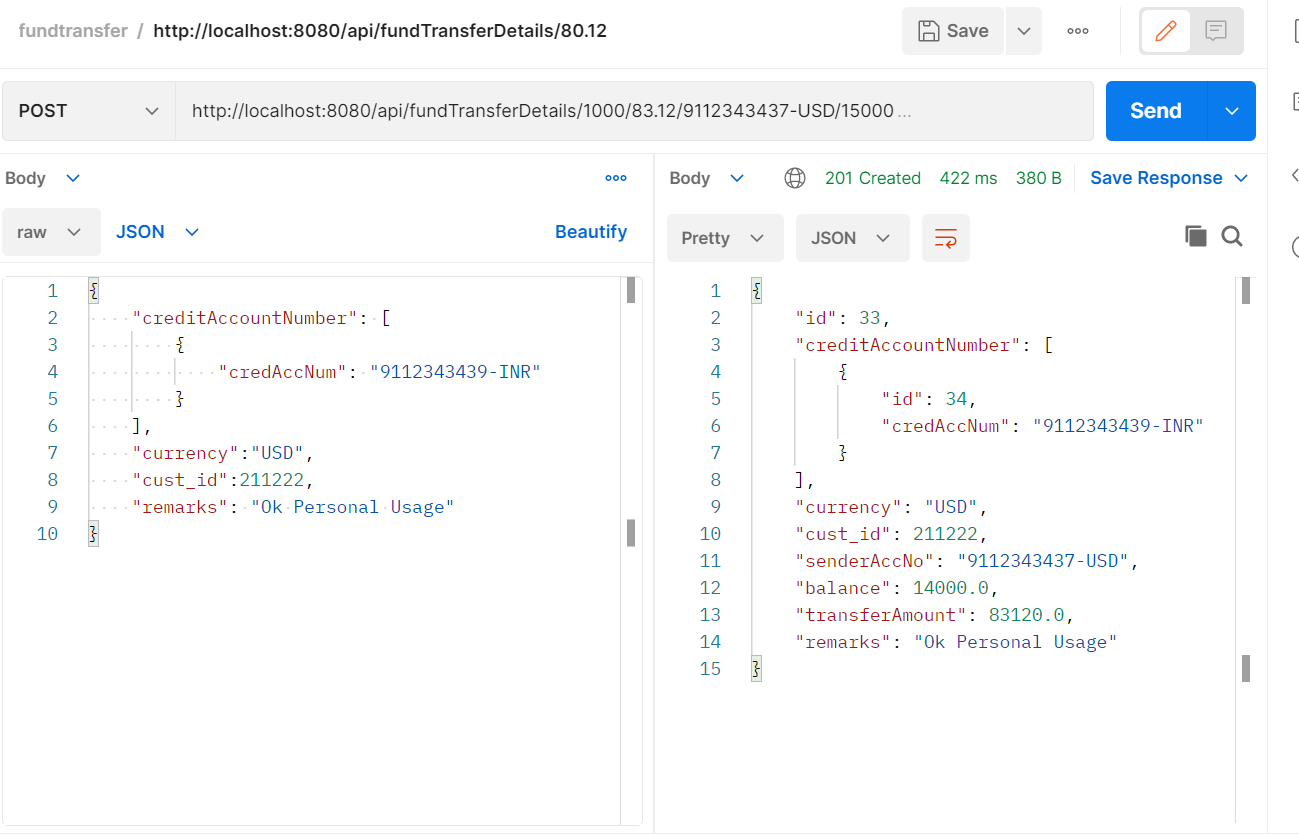
1. Select account number from which you want to transfer the fund.





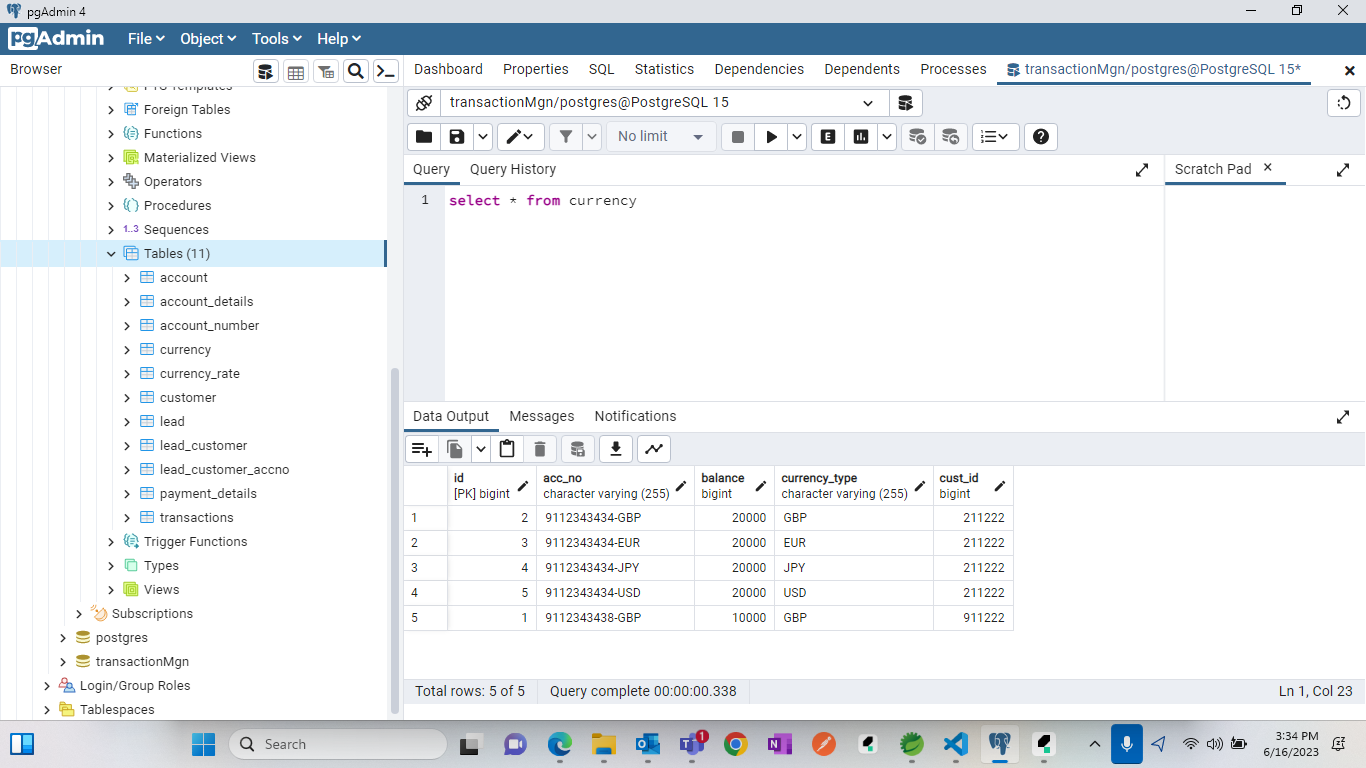


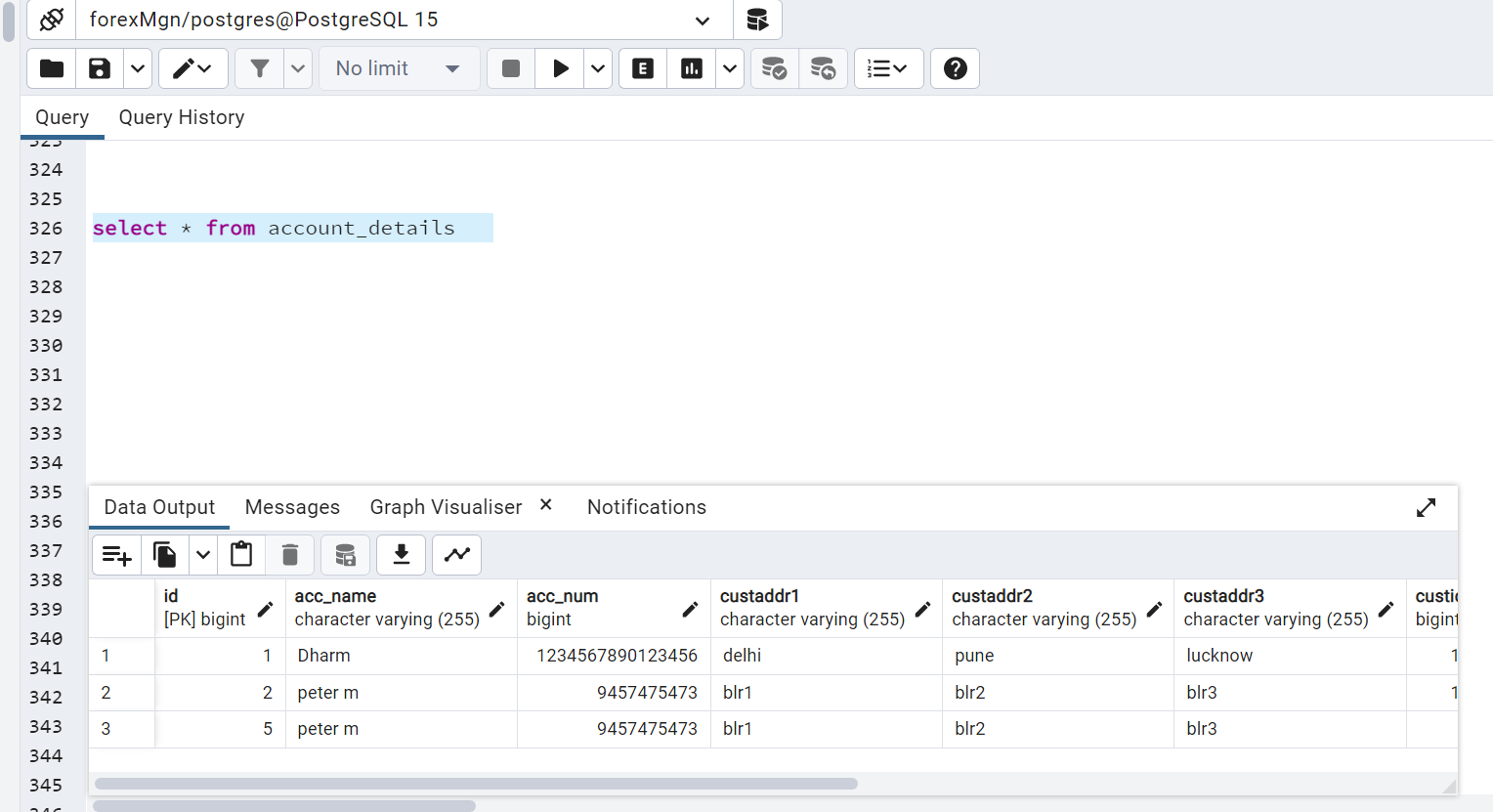


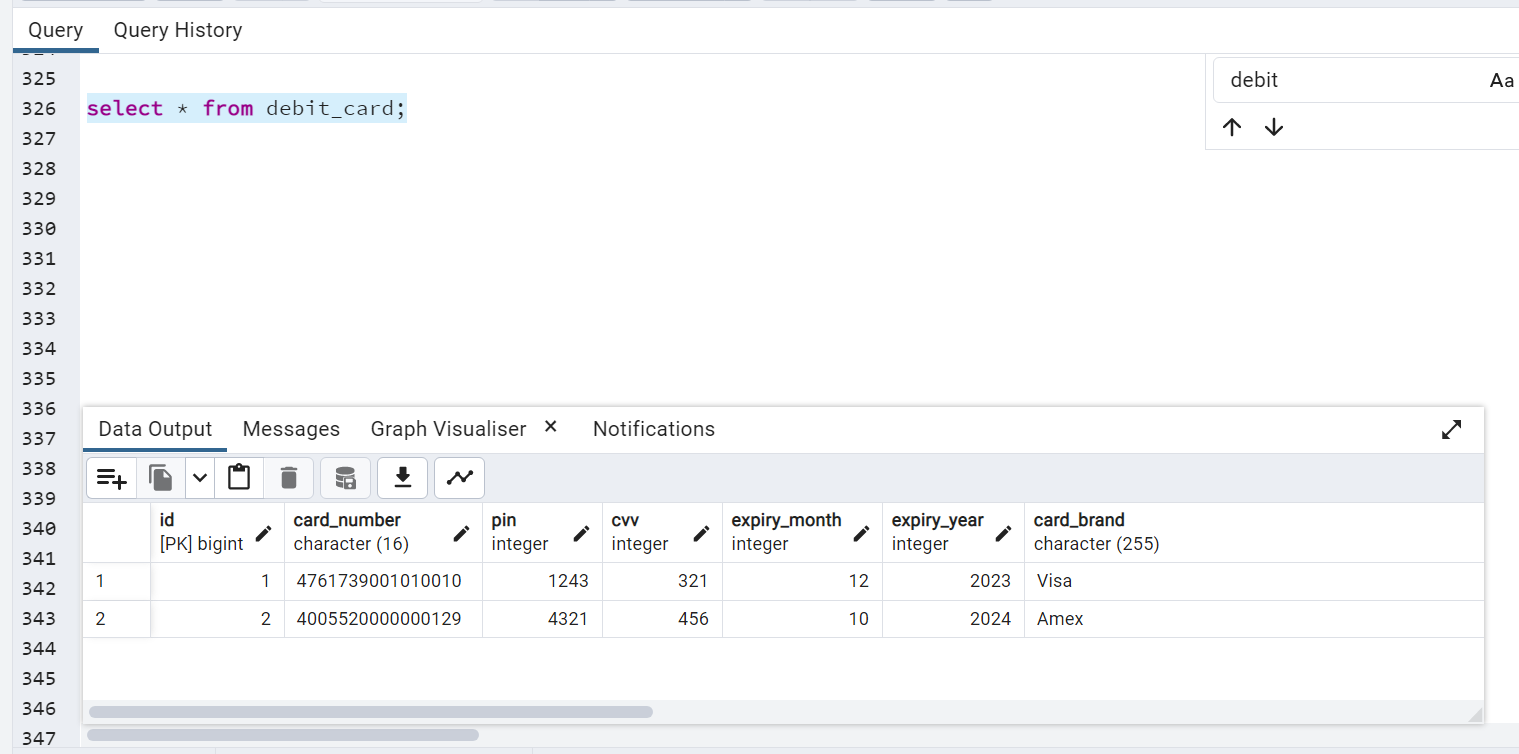


**Database Details:**

Table name: Currency







**Steps to run the application.**

1. Import project as a maven project in the IDE.

2. Create database in the PostgreSQL with the username and password.

3. Add username and password and connection URL in the application.properties file

4. Start the service.

5. Import the postman collection in the postman and run the API.