**PERSONAL DETAILS**

NAME: NANSUBUGA JOYCE EUZEBIA

Repository Url: [euzebia/accounts\_manager: Helps bank staff to manage bank customers'account information (github.com)](https://github.com/euzebia/accounts_manager)

**BRIEF DESCRIPTION OF THE DESIGN AND EVIDENCE IN SCREEN SHOTS**

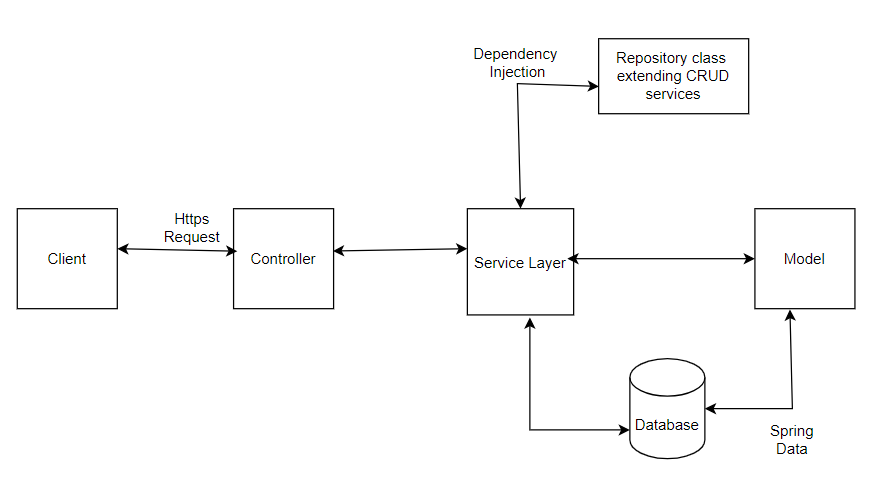
**Software description**

**Software name: accounts\_manager**

A real-world use case considered for implementation is an off-the-shelf software developed by a third party software vendor/system owner that is used in banks to help bank staff (personal banker) create bank accounts, modifying accounts based on customer preferences and managing other account details.

**Architecture**

The api design follows a layered architecture as illustrated below;



Explanation;

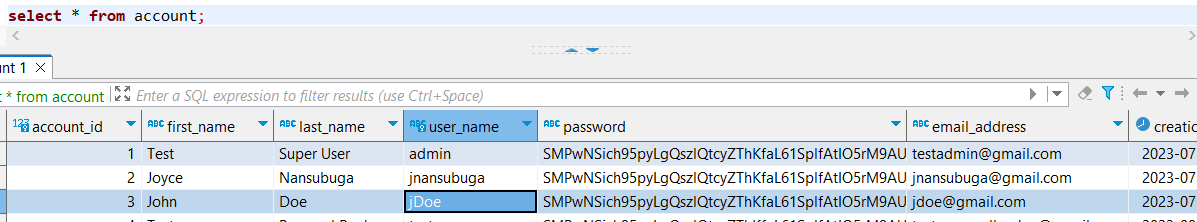
1. A client makes an http request i.e get,put,post,etc.. and is forwarded to the controller.
2. The controller maps the request and calls the service layer logic.
3. In the service layer, data validation,database calls and other business logic is where it is performed from.
4. The database acts a data repository for the data used by the api/application.

Security

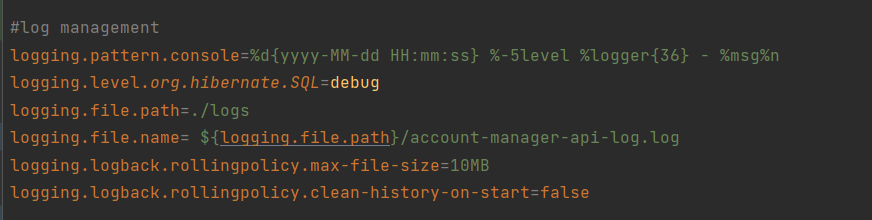
* Api credentials( basic auth)
* Csrf and xss



Encryption of user credentials ie password



Log management



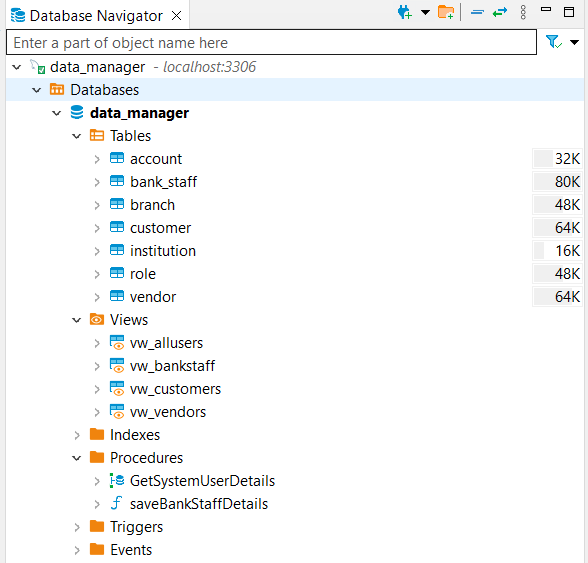
Roles and permissions (portal)

Language in which it was designed

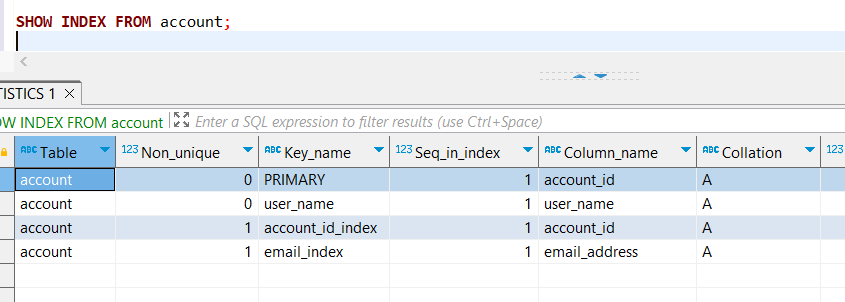
Application performance

-input validation

-stored procedures and view plus functions



Use of indices



Database design

-ER and how entities relate with each other

Api versioning and importance

Evidences to attach

-----------------------------------

1. application start up

2. log management

3. screen shots of postman when user details exist,when user details does not exist

4. Add a sequence diagram to illustrate rate limit, on the email verification service (throttling)

5. Explain where you have applied security both client and server side

6. Show stored procedures used to incorporate all the login details

7. Explain how you have used dependency injection

8. Explain the api architecture(DataAccess,service layer,presentation layer);

9. Explain how you have used circuit breakers

10. Give a brief on the client api, show evidence that it is running

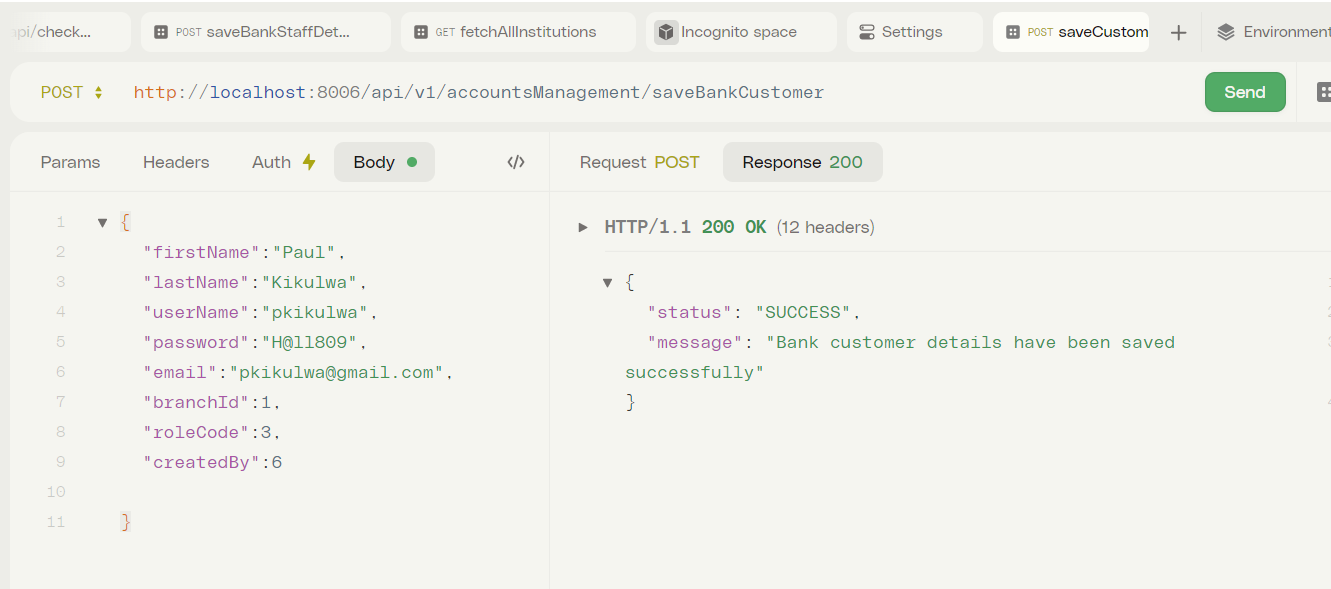
Additions

Pending items

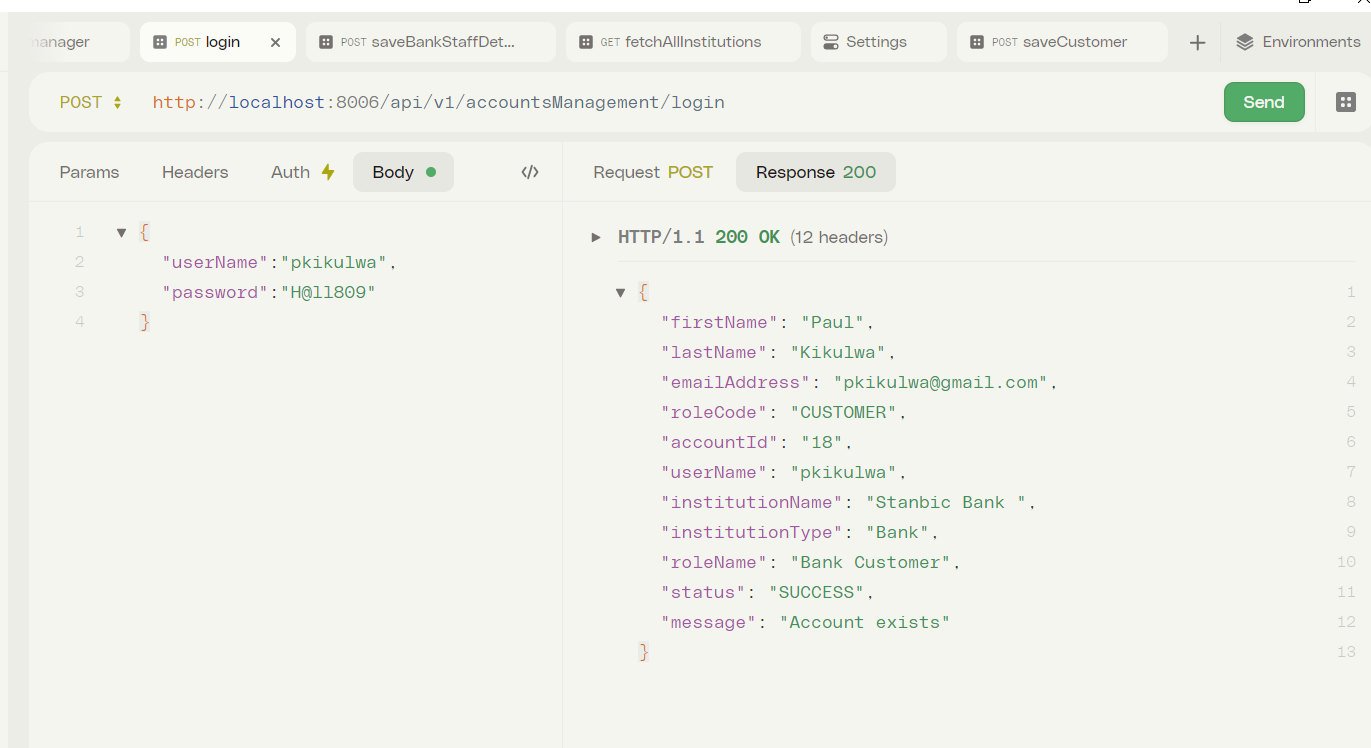
* Circuit breaker
* Caching
* Portal
* Documentation
* Testing end to end

Roles and permissions

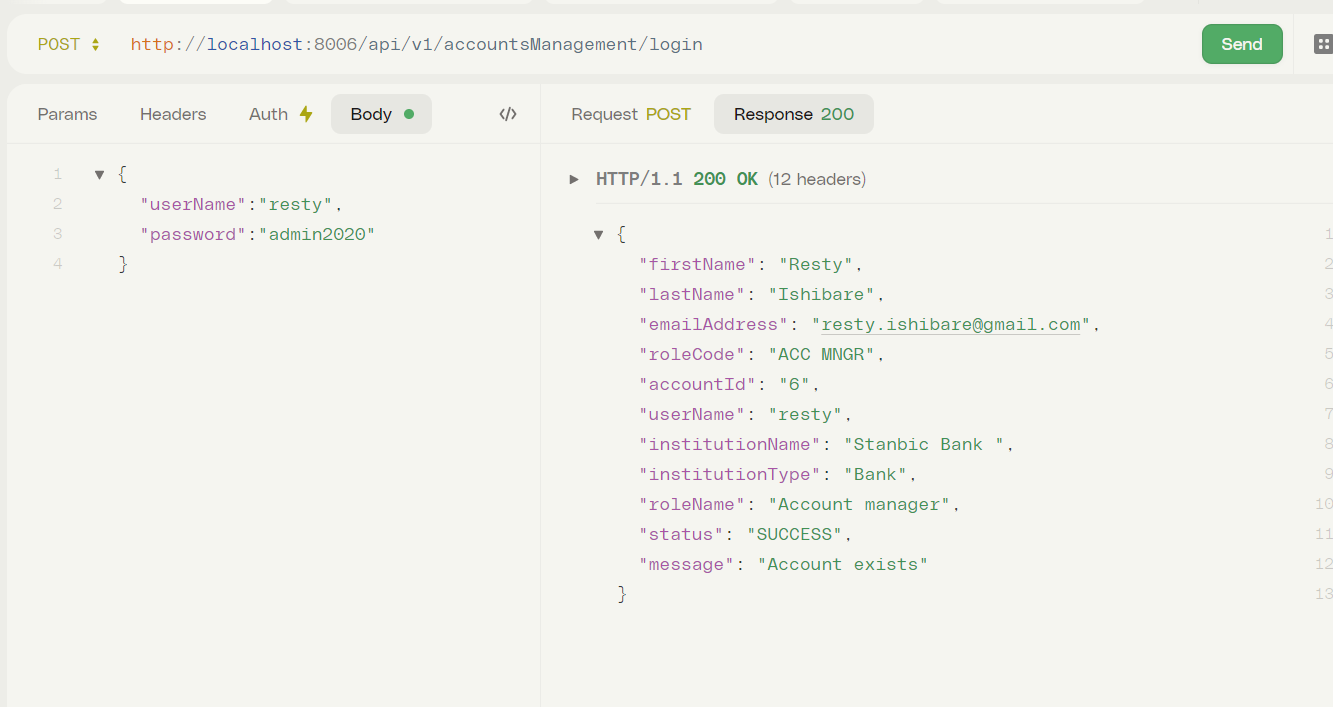
Creation of a bank customer



Newly created bank customer logs in



Bank staff access same endpoint to login.



System vendor login

