Comments:

I developed the entire project from scratch using resources like YouTube tutorials and documentation. Debugging was particularly interesting, as I learned to check API calls thoroughly, enhancing my problem-solving skills. I also faced challenges with data handling between the frontend and backend, which deepened my understanding of both Angular and Node.js.

Account Management Application

Full Stack + Angular 18 Material + Node JS + PostgreSQL

1. Database Setup

1.1. Creating the Database and Table

1. **Environment**: Use XAMPP to manage PostgreSQL.
2. **Database Creation**: Create a new PostgreSQL database to store account data.
3. **Table Structure**: Design a table to hold account details such as ID, name, email, and balance.

2. Backend Development

2.1. Setting Up Node.js

1. **Project Initialization**: Set up a Node.js project to handle server-side logic.
2. **API Implementation**: Develop RESTful APIs to manage account data. This includes endpoints for creating, reading, updating, and deleting account information.

2.2. API Testing

1. **Tool**: Use Postman to test the functionality of the implemented APIs.
2. **Database Verification**: Confirm that the changes reflect in the PostgreSQL database after API calls.

3. Frontend Development

3.1. Angular Application Setup

1. **Project Creation**: Set up an Angular standalone application using version 18.

2. **Material Design**: Install Angular Material for building a responsive and visually appealing user interface.

3.2. Proxy Configuration

1. **Setup**: Configure a proxy to allow the Angular application to communicate with the backend server running on a different port.

3.3. Component Generation

1. **Components**: Generate necessary components, services, and modules to structure the application effectively.
2. **Routing**: Implement routing logic to navigate between different components of the application.

3.4. Angular Material Table

1. **Data Display**: Create an Angular Material table to display account data, including actions for editing and deleting entries.
2. **HTTP Methods**: Implement the necessary HTTP methods (GET, POST, PUT, DELETE) in the service to manage account information.

3.5. Features Implementation

1. **Searching, Sorting, and Pagination**: Integrate functionalities to search, sort, and paginate through the account data displayed in the table.
2. **User Input Forms**: Use Angular Material form fields to capture user input for creating and editing accounts.

4. User Interaction

4.1. Form Handling

1. **Data Submission**: Capture input data from forms and submit it to the backend when creating or updating an account.
2. **Edit Functionality**: Enable users to populate form fields with existing account data for modification.

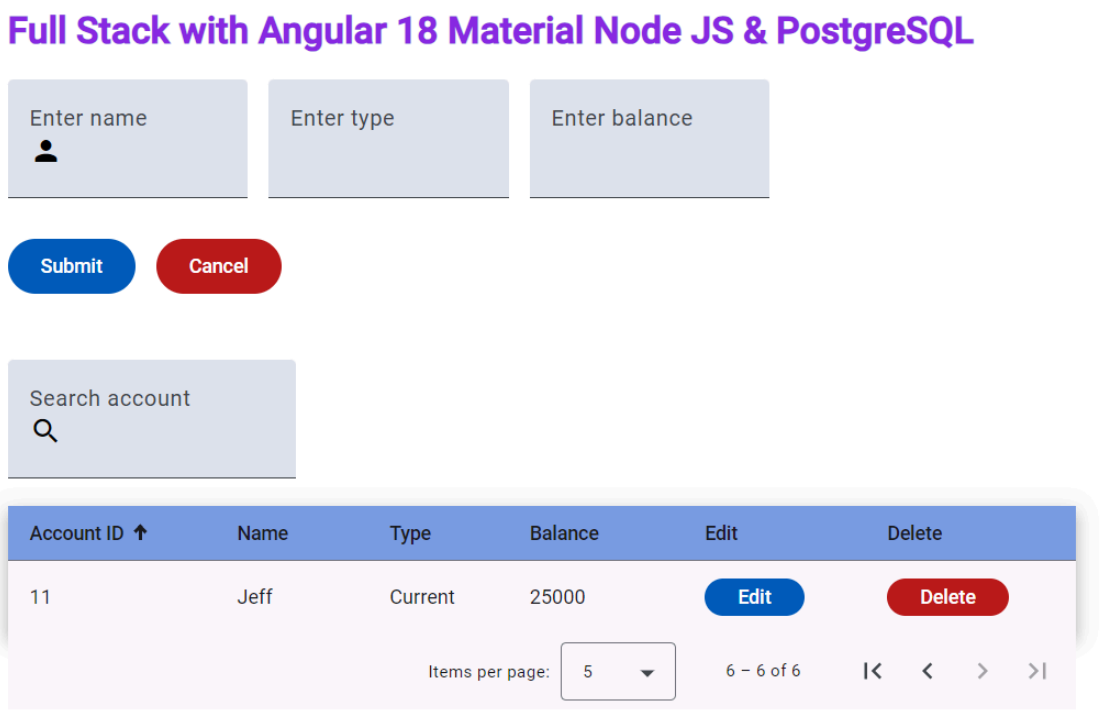
4.2. Account Deletion

1. **Removal Process**: Implement a feature to delete accounts from the database, ensuring that the table reflects the changes immediately.

References:

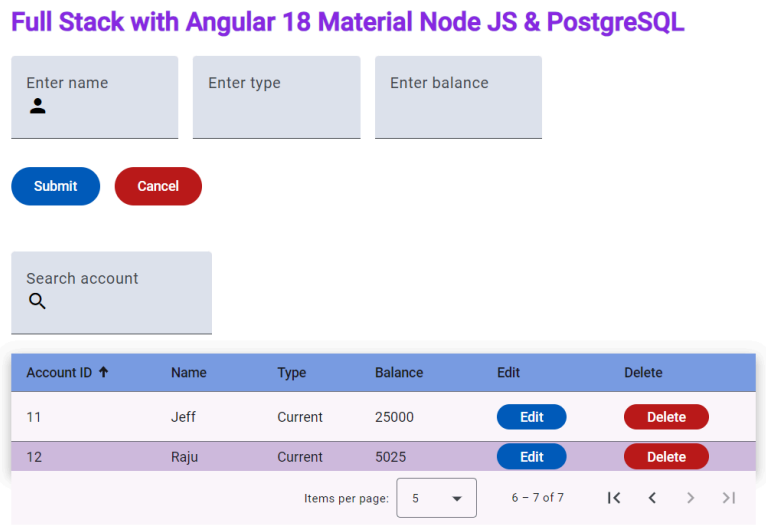
1. https://www.youtube.com/watch?v=xjgCkEfUIwE
2. https://material.angular.io/

Images:



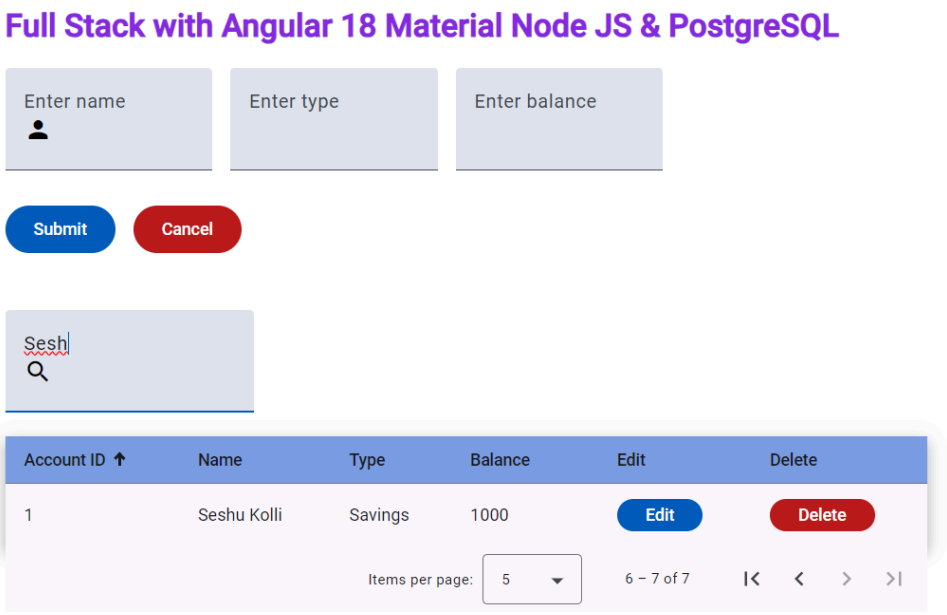
Insertion:

- Inserted Raju



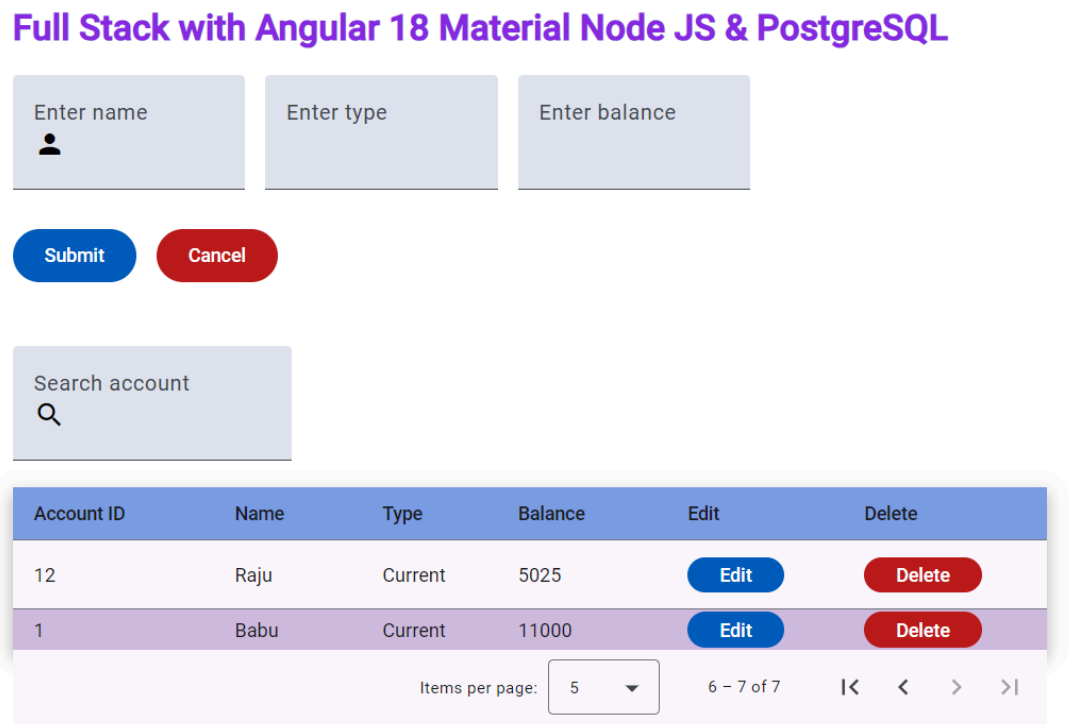
Search:

- Search seshu



Edit:

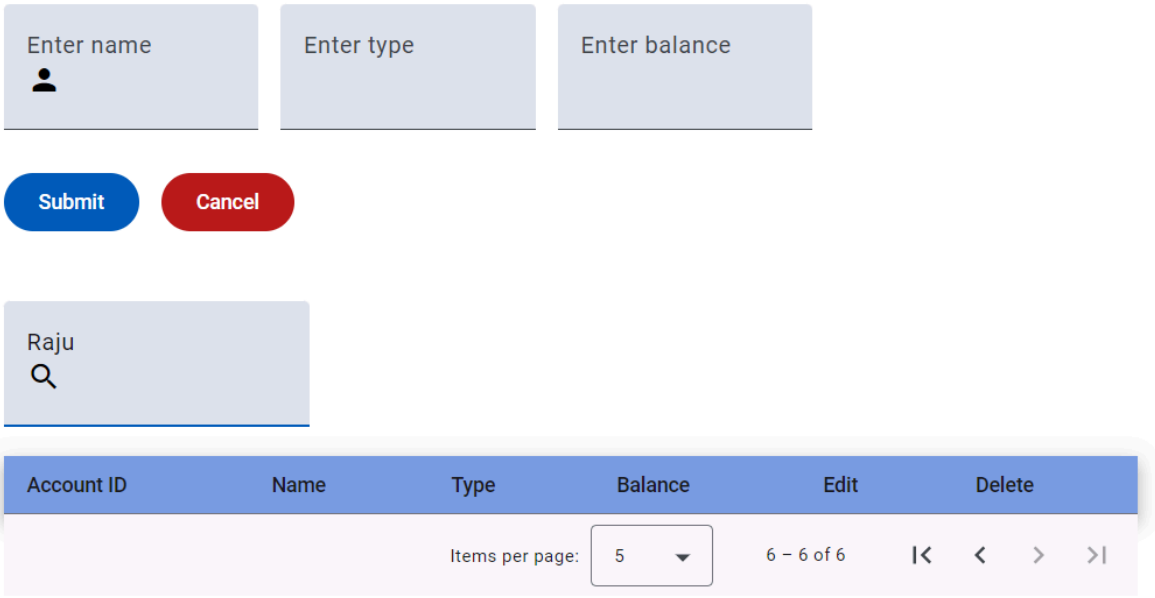
- Edited seshu kolli to Babu



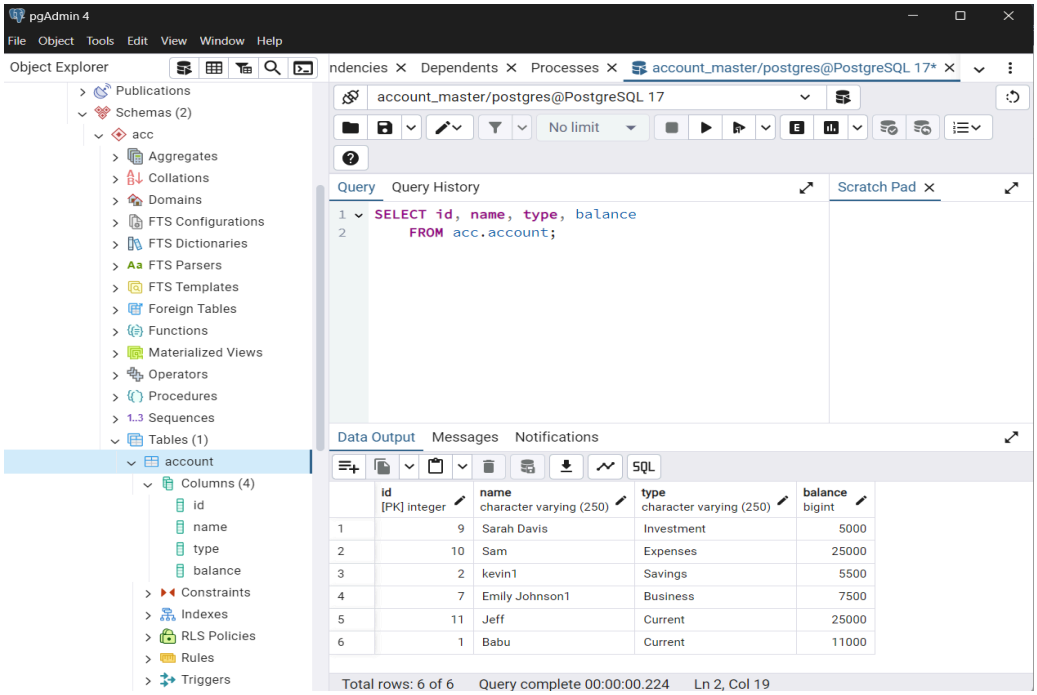
Delete:

Deleted Raju

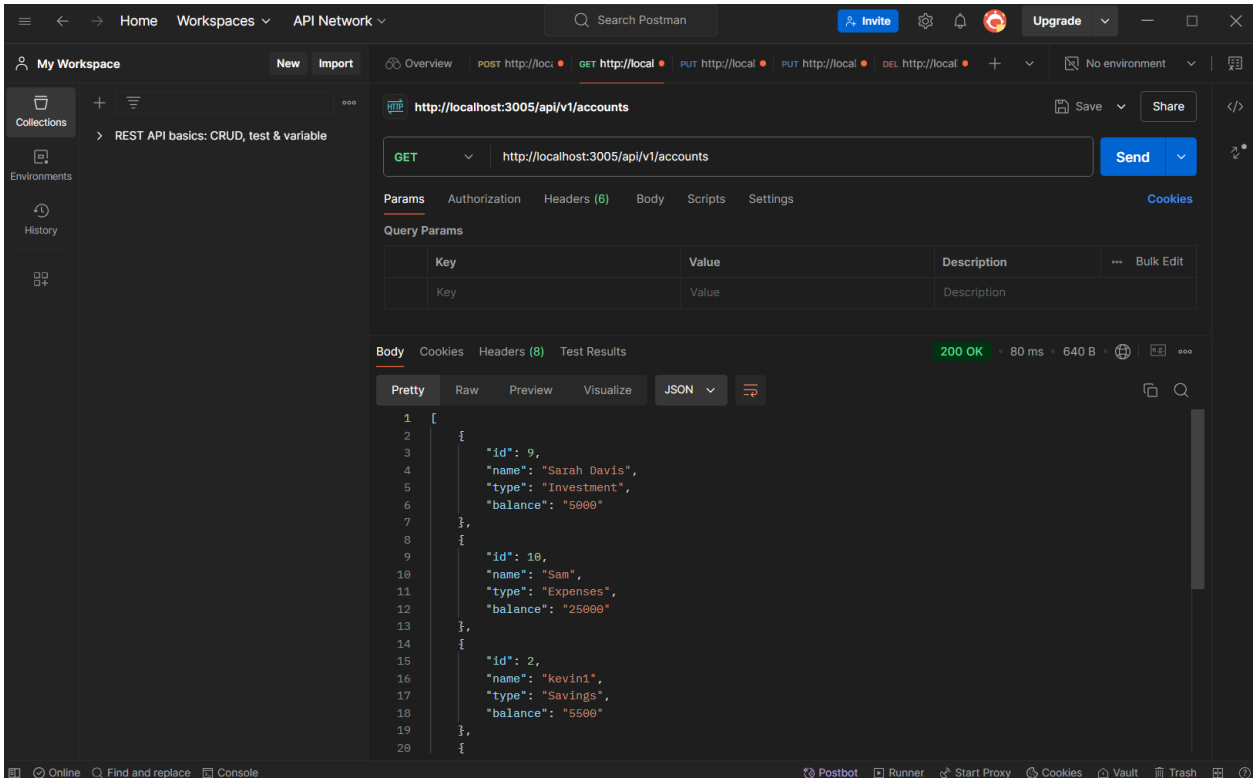
Full Stack with Angular 18 Material Node JS & PostgreSQL



Postgres DataBase:



Postman:


Home
Workspaces
API Network
Search Postman
Invite
Upgrade
My Workspace
New
Import
Collections
Environments
History
REST API basics: CRUD, test & variable
Overview
GET http://local
No environment
http://localhost:3005/api/v1/accounts
Save
Share
GET
http://localhost:3005/api/v1/accounts
Send
Params
Authorization
Headers (6)
Body
Scripts
Settings
Cookies
Query Params
Key
Value
Description
Bulk Edit
Body
Cookies
Headers (8)
Test Results
200 OK
80 ms
640 B
Pretty
Raw
Preview
Visualize
JSON
[
{
"id": 9,
"name": "Sarah Davis",
"type": "Investment",
"balance": "5000"
},
{
"id": 10,
"name": "Sam",
"type": "Expenses",
"balance": "25000"
},
{
"id": 2,
"name": "kevin1",
"type": "Savings",
"balance": "5500"
},
{
Online
Find and replace
Console
Postbot
Runner
Start Proxy
Cookies
Vault
Trash