REST API Develop using NodeJS & Backend SQLDB

28.05.2024

K. DUSHYANT REDDY

Overview

To build a REST API using NodeJS to handle customer data and store it in an SQL database.

Description

To create a POST API that accepts JSON input for first name, last name, email, mobile number (with country code), address, and pin code. Implement validation for email, mobile number, and pin code. Upon receiving a valid request, store the data in a SQL database using a stored procedure, with a table named "Customer Details". The table will include fields for a unique Customer ID (auto-generated), first name, last name, email, mobile number, address, and pin code. The API will return a success response on successful insertion or a failure response if validation fails or the insertion is unsuccessful.

Prerequisites

Basic knowledge on -

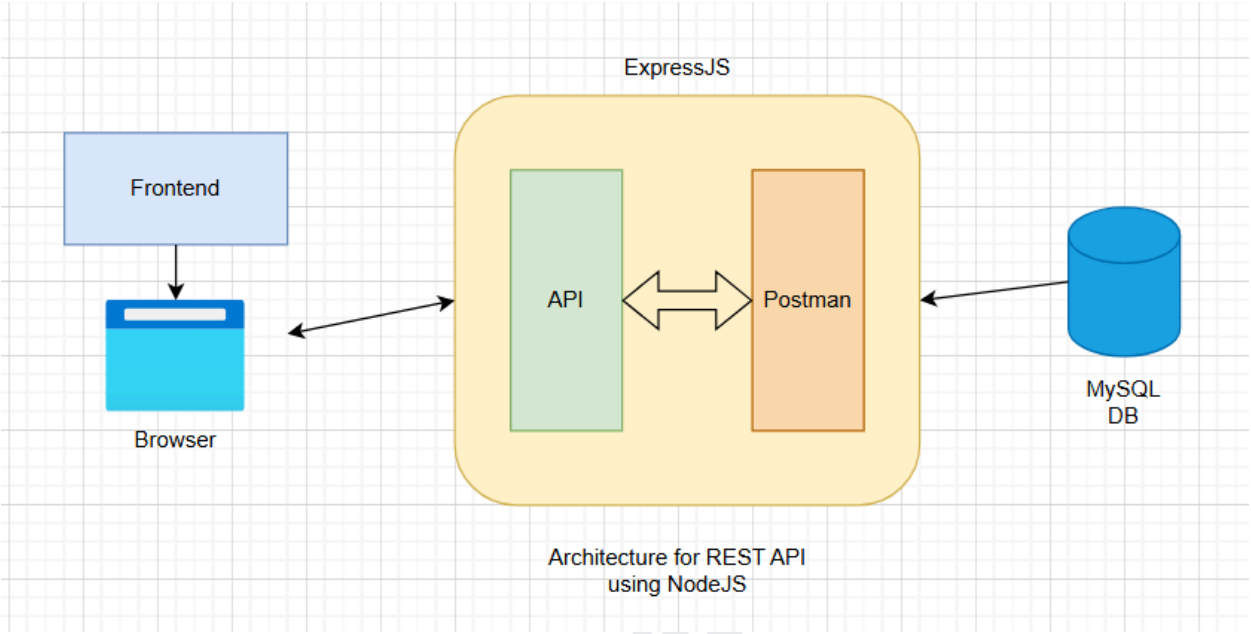
1. **Visual Studio Code** - a popular and adaptable Microsoft source code editor. Because of its reputation for being both easy to use and feature-rich, developers working with a variety of platforms and programming languages choose it over other options.

2. **NodeJS** - a free and open-source cross-platform JavaScript runtime environment that facilitates server-side scripting using JavaScript with the ultimate goal of building dynamic website content.

3. **Postman/Thunder Client** - Postman is a tool that makes it simple for us to create and use APIs. The HTTP requests that we submit to the server hosting the API are constructed using Postman. Thunder Client is a lightweight Rest API Client Extension for Visual Studio Code

4. **SQL** - acts as an essential client tool for creating and running SQL queries. It is used for connecting to the database server, which is typically a dedicated machine running SQL Server.

Architecture



Data Flow:

a. Frontend (Browser): This is where the user interacts with the application. The frontend sends HTTP requests to the API. In this case, the frontend is a browser, but it could also be a mobile app or another type of client.

b. API (Express.js and Node.js): The API receives the HTTP requests from the frontend. It's built with Express.js, a web application framework for Node.js. Node.js is a JavaScript runtime that executes JavaScript code outside a web browser. The API processes the requests, interacts with the database if necessary, and sends HTTP responses back to the frontend.

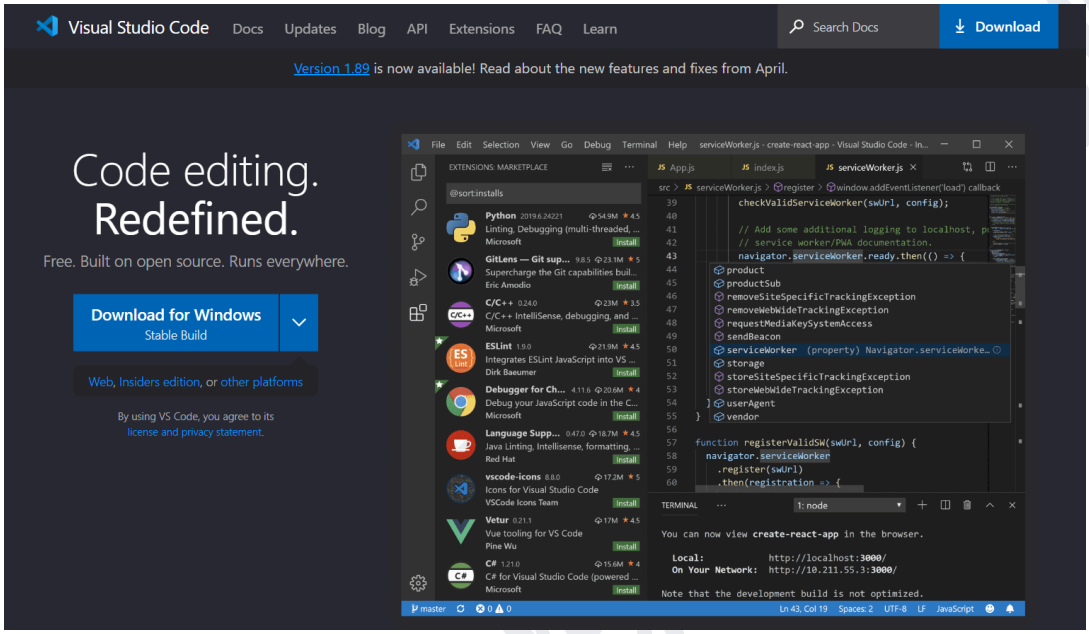
c. Postman: Postman is a tool used for API testing. It allows developers to send HTTP requests to the API and view the responses. This helps in debugging and verifying the functionality of the API.

d. MySQL DB: This is the database where the application's data is stored. The API interacts with the MySQL database to retrieve, insert, update, or delete data based on the HTTP requests it receives.

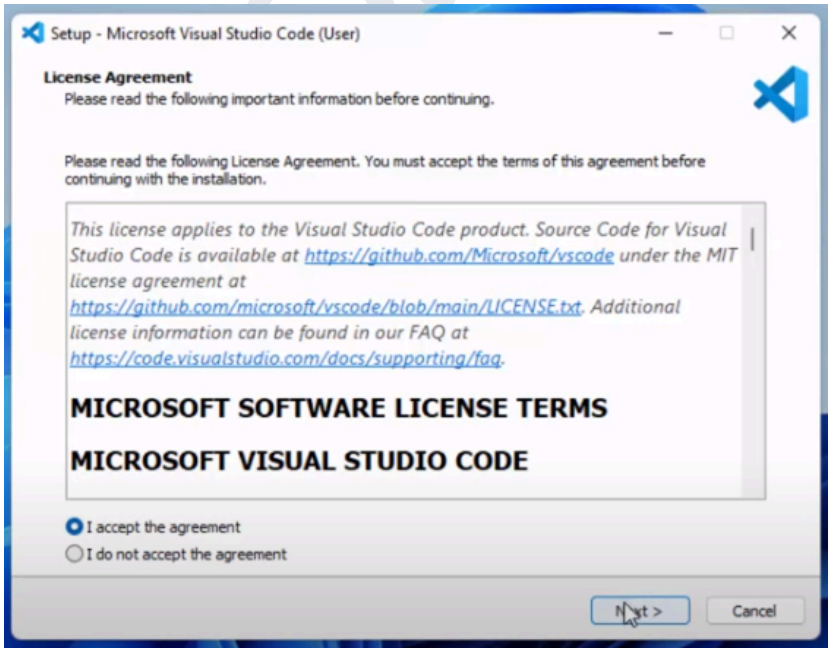
Steps

Step 1: Installation of Softwares

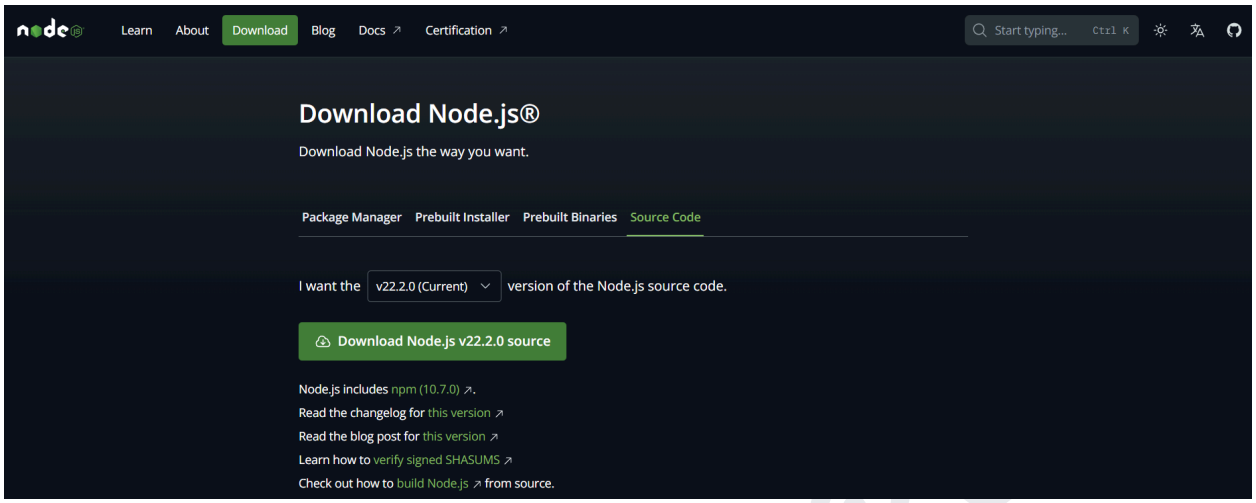
1.1 Download the VS Code windows x86 user installer.



1.2 Install the configured file

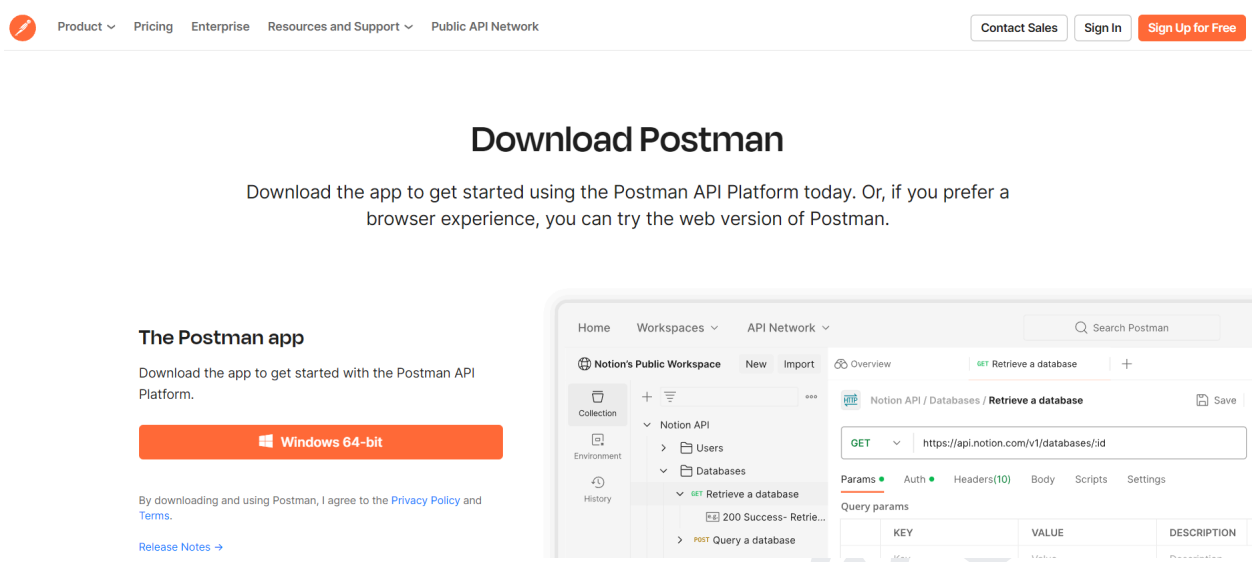


1.3 Download the NodeJS windows installer source code.

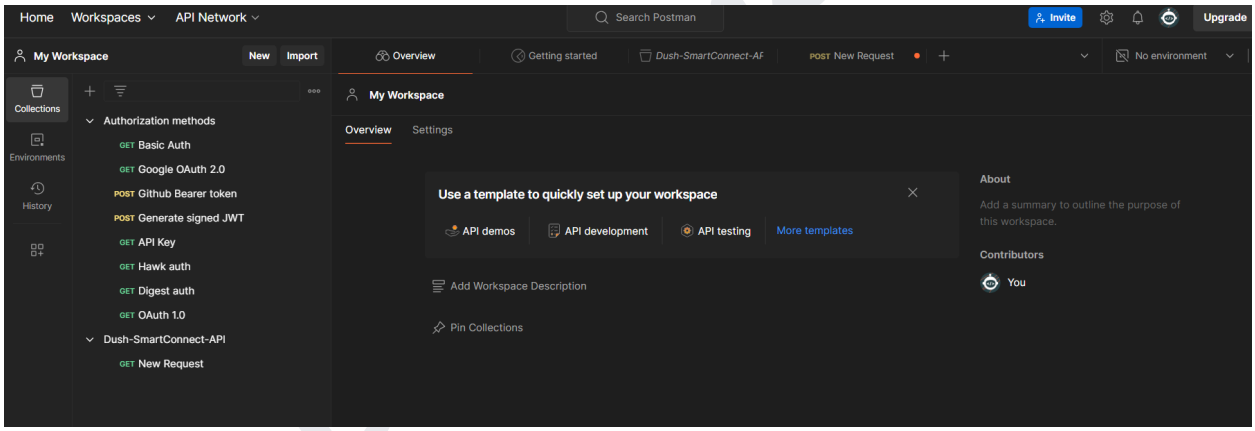


1.4 Install the .exe file that has been downloaded.

1.5 Similarly, download the Postman windows 64 bit.



1.6 After that install it and create an account to start building the custom APIs in workspace.



1.7 Similarly, download the MySQL DB msi installer.

1.8 Now, install it and click Yes on the popup modal.

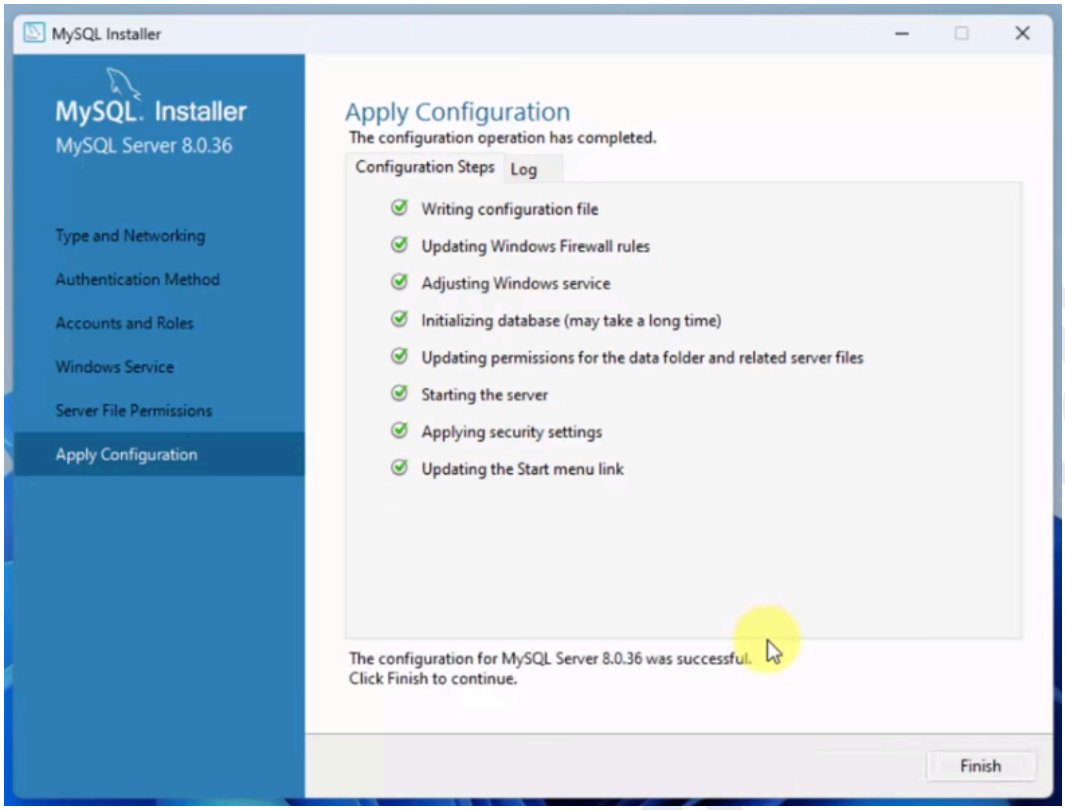
1.9 Select “Choosing setup type” as “Custom” and click on Next. Step5 - Click ‘+’ on “MySQL servers” then ‘+’ on “MySQL server <version>”, select the first product in it and click the right arrow button.

1.10 Similarly, click ‘+’ on “Applications” then ‘+’ on “MySQL workbench” then ‘+’ on “MySQL workbench <version>”, select the first product in it and click the right arrow button. Similarly, click ‘+’ on “MySQL shell” then ‘+’ on “MySQL shell <version>”, select the first product in it and click the right arrow button.

1.11 Click “Next” a couple of times and click “Execute” and then click “Next”. Step9 - In “Authentication method” click “Next” and in the “Accounts and Roles” page type the password and remember the password.

1.12 Click “Next” in the "Windows service" page. Click “Next” in "Server file permissions" page then click on "Execute" in the "Apply Configuration" window. Click on "Finish" click on "Next" click on "Finish", now it will open one black window (MySQL Shell) ignore it after some time it will open "MySQL Workbench"

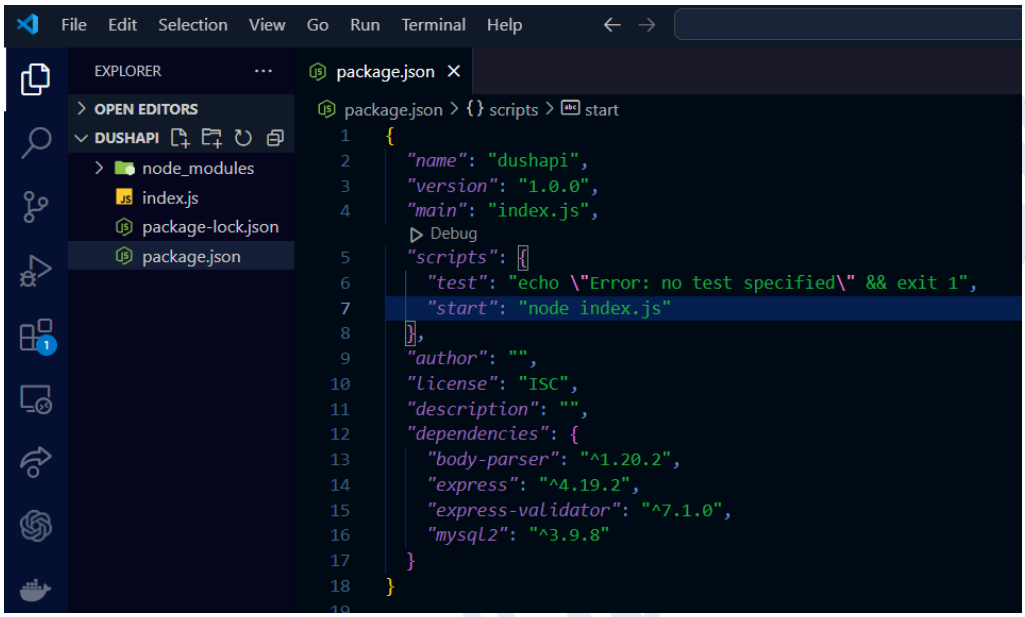
1.13 In the menu bar, go to databases, click on "Connect database" and click on OK. Then give the password which you gave in previous steps, click on "Ok”. It should connect without any error.



1.14 Note for all the above downloaded softwares copy the path of the directory and paste it in environment variables so that it can be easily accessible through CMD.

Step 2: Build Project in VS Code Console

2.1 Open VScode, create a folder named **“dushapi”** and then open the terminal and run the command **“npm init”** to initialize the project and then install the required dependencies.



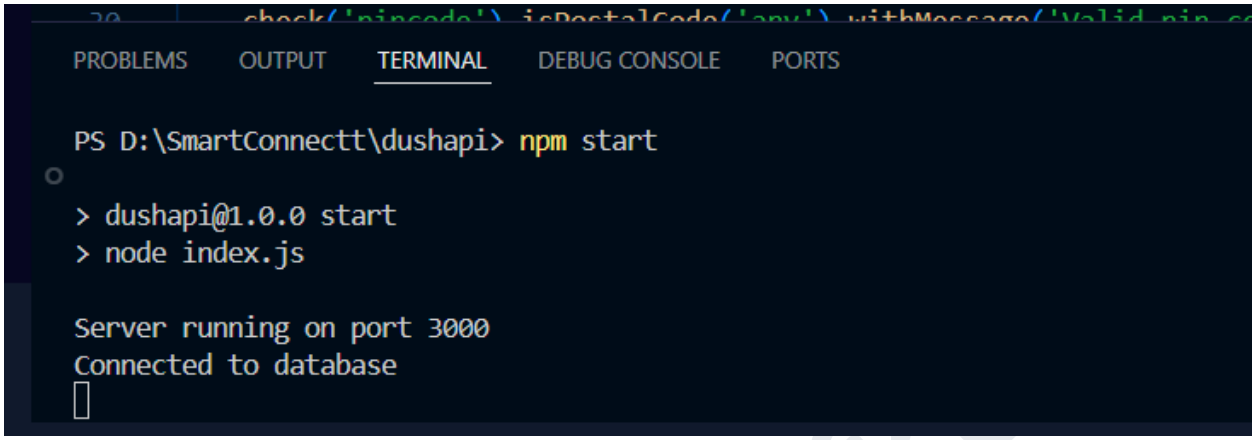
2.2 Create the **index.js** file and write the code for building the server, and configuring with database.

```
const express = require('express');

const bodyParser = require('body-parser');

const mysql = require('mysql2');

const { check, validationResult } = require('express-validator');


const app = express();

const port = 3000;


app.use(bodyParser.json());
```

```
const db = mysql.createConnection({
    host: 'localhost',
    user: 'root',
    password: 'mysql',
    database: 'dush_cust_db'
});

db.connect((err) => {
    if (err) throw err;
    console.log('Connected to database');
});

app.post('/api/customers', [
    check('firstName').notEmpty().withMessage('First name is required'),
    check('lastName').notEmpty().withMessage('Last name is required'),
    check('email').isEmail().withMessage('Valid email is required'),
    check('mobileNumber').matches(/^\+\d{1,3}\d{10}$/).withMessage('Valid
mobile number with country code is required'),
    check('address').notEmpty().withMessage('Address is required'),
    check('pincode').isPostalCode('any').withMessage('Valid pin code is
required')
], (req, res) => {
    const errors = validationResult(req);
    if (!errors.isEmpty()) {
        return res.status(400).json({ errors: errors.array() });
    }
```

```
    const { firstName, lastName, email, mobileNumber, address, pincode } =
req.body;


    const query = 'CALL AddCustomer(?, ?, ?, ?, ?, ?)';
    db.query(query, [firstName, lastName, email, mobileNumber, address,
pincode], (err, results) => {
        if (err) {
            return res.status(500).json({ message: 'Error inserting data',
error: err });
        }
        res.status(201).json({ message: 'Customer added successfully',
customerID: results.insertId });
    });
});


app.get('/api/customers', (req, res) => {
    const query = 'SELECT * FROM CustomerDetails';
    db.query(query, (err, results) => {
        if (err) {
            return res.status(500).json({ message: 'Error fetching data',
error: err });
        }
        res.status(200).json(results);
    });
});


app.listen(port, () => {
    console.log(`Server running on port ${port}`);
});
```

2.3 After writing the LOC, run the code in the terminal by **“npm start”**.



Step 3: Launch MYSQL in CMD

3.1 Login into the mysql directory by typing **“mysql -u root -p”** and then give the password which you had created while installing mysql.

```
C:\Users\dushi>mysql -u root -p
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 17
Server version: 8.0.37 MySQL Community Server - GPL

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
```

3.2 Now, create a database named **“dush_cust_db”** , the same name that is given in the index.js file.

```
mysql> create database dush_cust_db;
Query OK, 1 row affected (0.01 sec)

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| dush_cust_db       |
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.00 sec)

mysql> USE dush_cust_db;
Database changed
```

3.3 After building the DB, create a table **"CustomerDetails"** with the requirements assigned in the Description.

```
mysql> CREATE TABLE CustomerDetails (
    ->     CustomerID INT AUTO_INCREMENT PRIMARY KEY,
    ->     FirstName VARCHAR(255) NOT NULL,
    ->     LastName VARCHAR(255) NOT NULL,
    ->     Email VARCHAR(255) NOT NULL UNIQUE,
    ->     MobileNumber VARCHAR(20) NOT NULL UNIQUE,
    ->     Address TEXT NOT NULL,
    ->     Pincode VARCHAR(10) NOT NULL
    -> );
Query OK, 0 rows affected (0.17 sec)
```

3.4 Since the requirement is of stored procedure not the inline queries type the command for as.

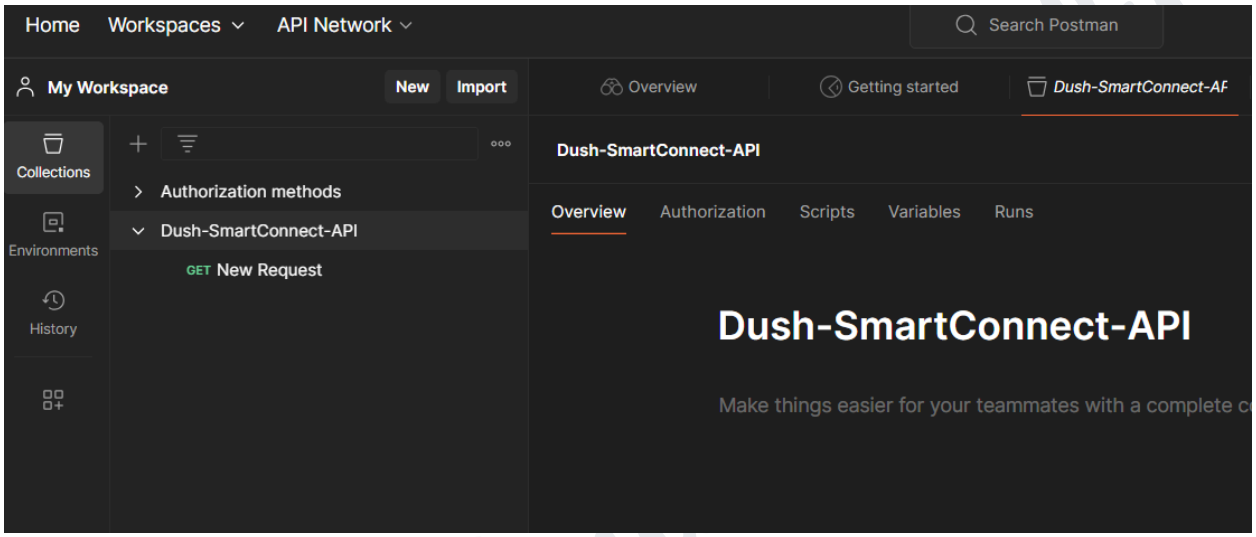
```
mysql> CREATE PROCEDURE AddCustomer(
    ->     IN firstName VARCHAR(255),
    ->     IN lastName VARCHAR(255),
    ->     IN email VARCHAR(255),
    ->     IN mobileNumber VARCHAR(20),
    ->     IN address TEXT,
    ->     IN pincode VARCHAR(10)
    -> )
    -> BEGIN
    ->     INSERT INTO CustomerDetails (FirstName, LastName, Email, MobileNumber, Address, Pincode)
    ->     VALUES (firstName, lastName, email, mobileNumber, address, pincode);
    -> END //
Query OK, 0 rows affected (0.02 sec)

mysql> DELIMITER ;
```

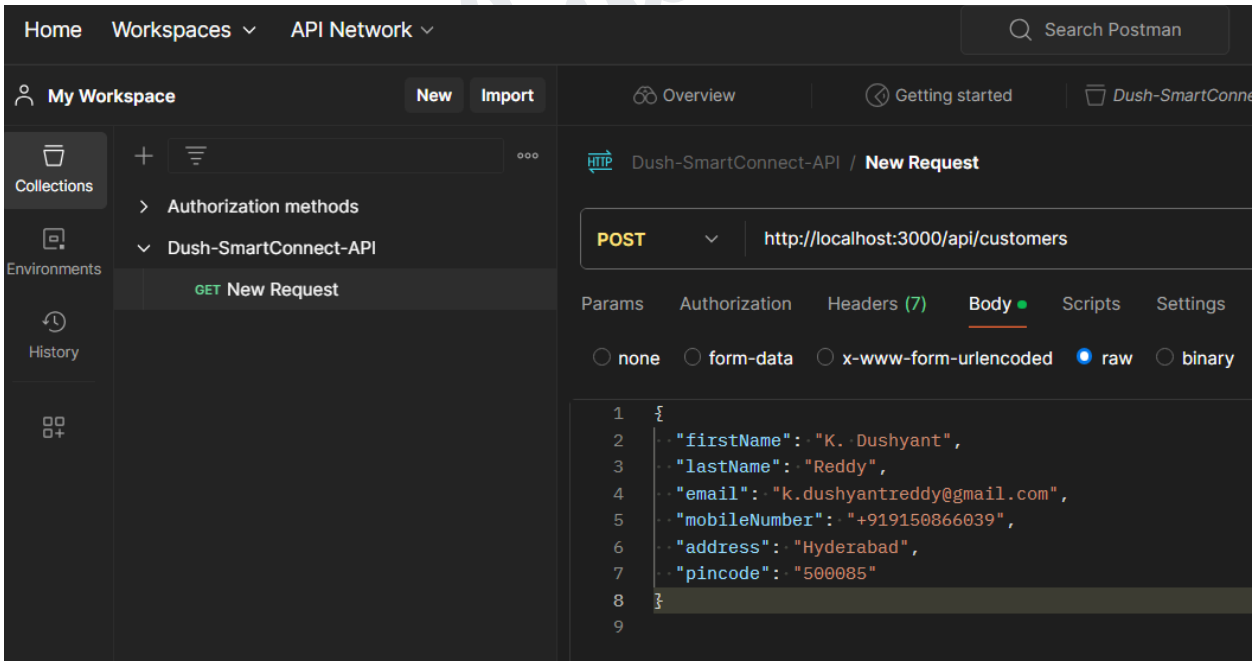
3.5 Check if the table is created or not by using the command **"show tables".**

```
mysql> show tables;
+-------------------------+
| Tables_in_dush_cust_db |
+-------------------------+
| customerdetails         |
+-------------------------+
1 row in set (0.04 sec)
```

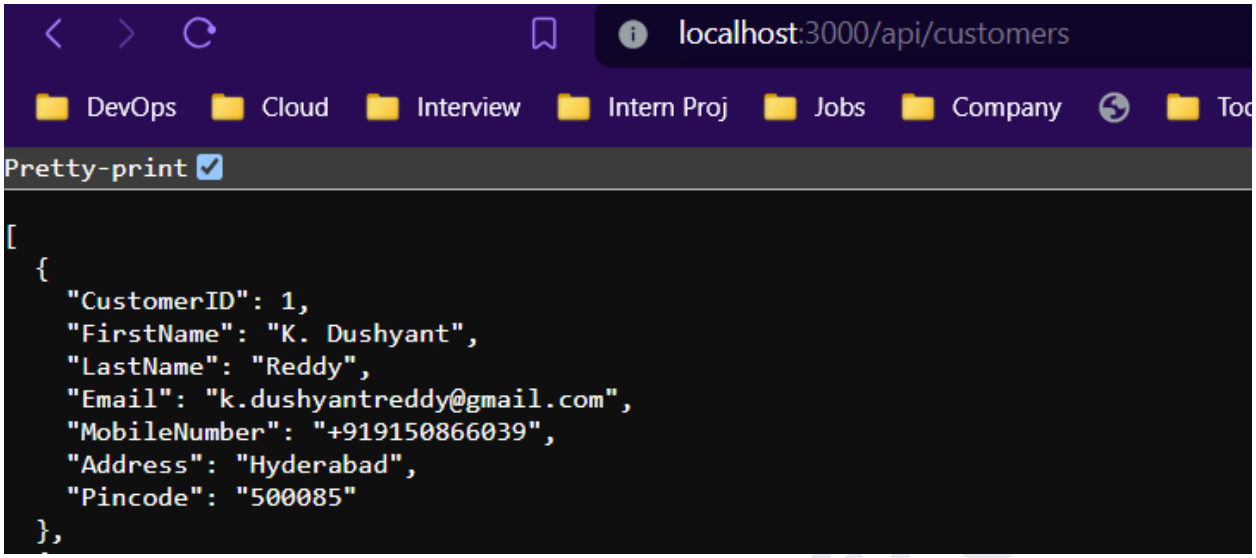
Step 4: Hit APIs in Postman

4.1 Now, create a collection **“Dush-SmartConnect-API”** and then create a new request and set the request type as **POST** and enter the URL for the endpoint **“http://localhost:3000/api/customers”** and keep Body as raw with **JSON** format. And click **Send**.

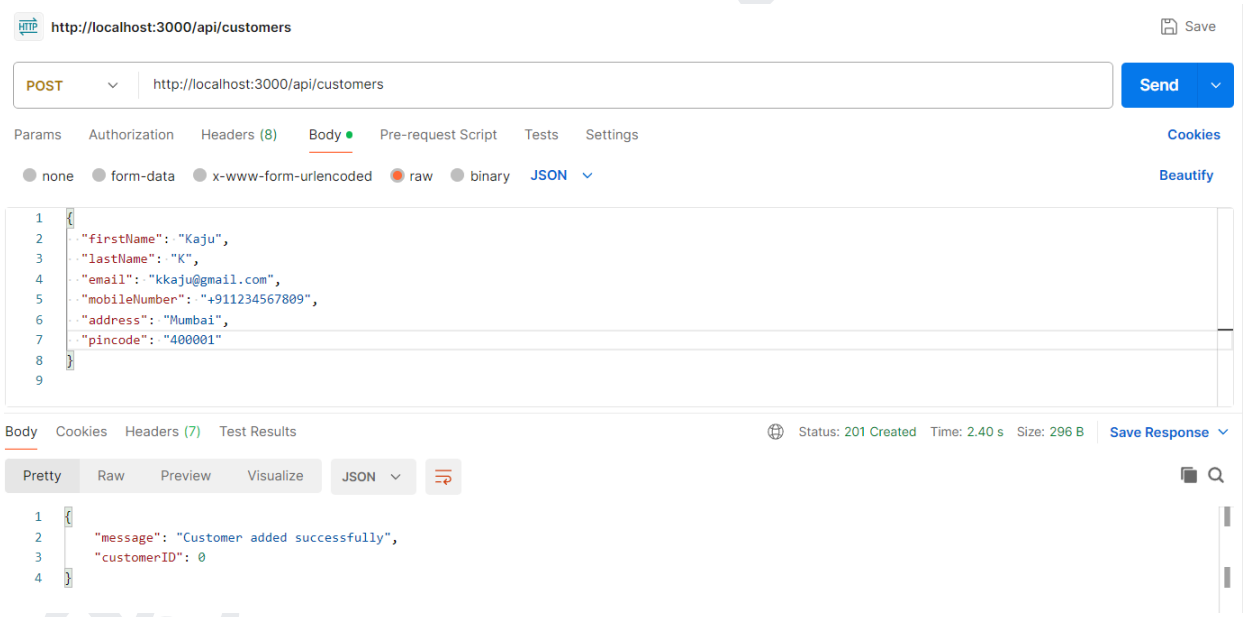


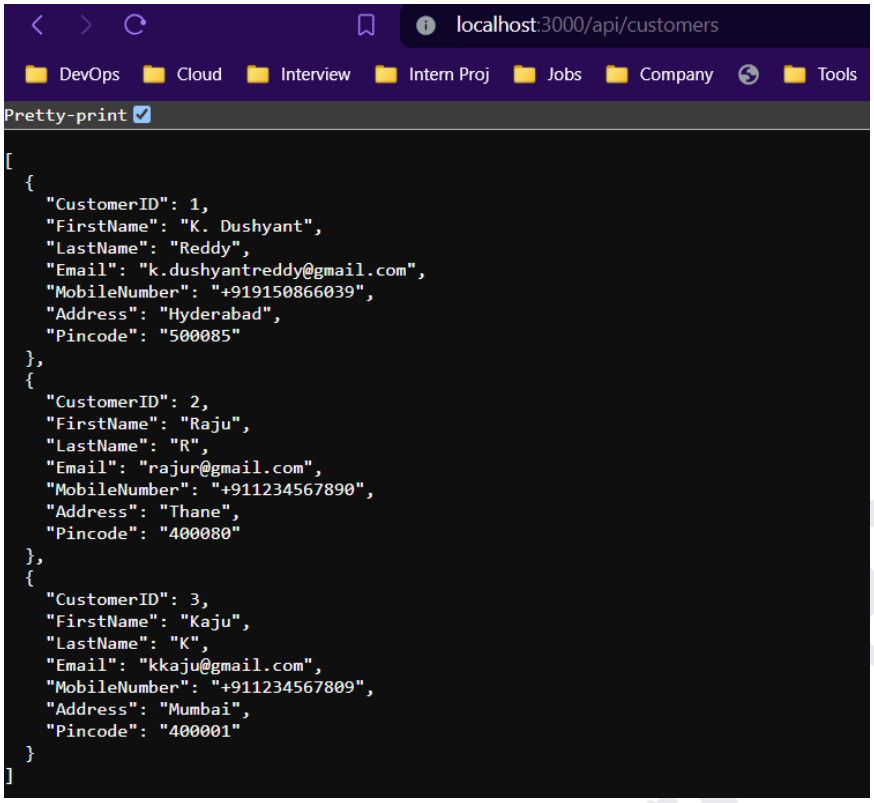


4.2 Now copy the URL and paste it in the New tab to check if the API is responded or not in JSON dataform.

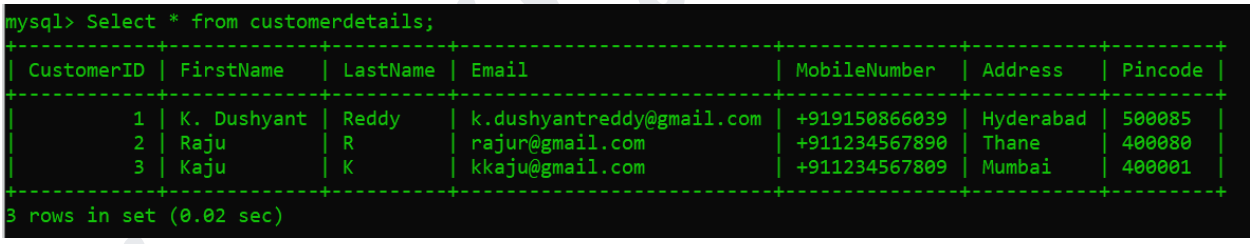


4.3 Similarly hit a couple more custom API requests.



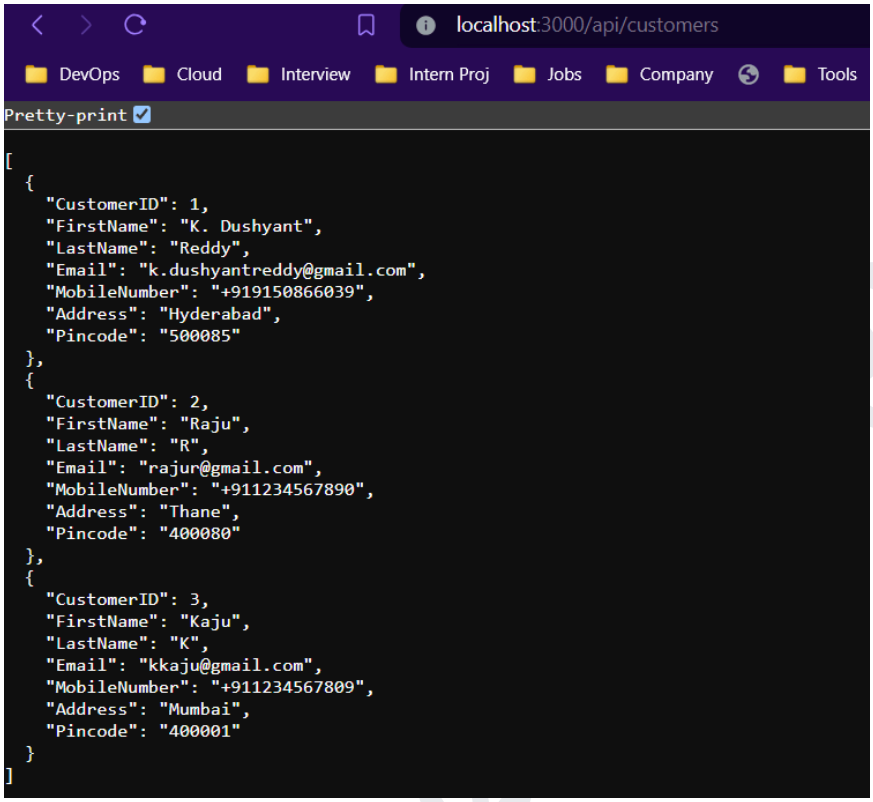


4.4 Note while we are hitting the API it's also storing the MySQL DB to check that type in the command **"Select * from customerdetails"**.



RESULT

Task on to build a REST API using NodeJS to handle customer data and store it in an SQL database successfully accomplished.




```
[
  {
    "CustomerID": 1,
    "FirstName": "K. Dushyant",
    "LastName": "Reddy",
    "Email": "k.dushyantreddy@gmail.com",
    "MobileNumber": "+919150866039",
    "Address": "Hyderabad",
    "Pincode": "500085"
  },
  {
    "CustomerID": 2,
    "FirstName": "Raju",
    "LastName": "R",
    "Email": "rajur@gmail.com",
    "MobileNumber": "+911234567890",
    "Address": "Thane",
    "Pincode": "400080"
  },
  {
    "CustomerID": 3,
    "FirstName": "Kaju",
    "LastName": "K",
    "Email": "kkaju@gmail.com",
    "MobileNumber": "+911234567809",
    "Address": "Mumbai",
    "Pincode": "400001"
  }
]
```


```
mysql> Select * from customerdetails;
+------------+-------------+----------+--------------------------+--------------+-----------+---------+
| CustomerID | FirstName   | LastName | Email                    | MobileNumber | Address   | Pincode |
+------------+-------------+----------+--------------------------+--------------+-----------+---------+
|          1 | K. Dushyant | Reddy    | k.dushyantreddy@gmail.com | +919150866039 | Hyderabad | 500085  |
|          2 | Raju        | R        | rajur@gmail.com          | +911234567890 | Thane     | 400080  |
|          3 | Kaju        | K        | kkaju@gmail.com          | +911234567809 | Mumbai    | 400001  |
+------------+-------------+----------+--------------------------+--------------+-----------+---------+
3 rows in set (0.02 sec)
```