

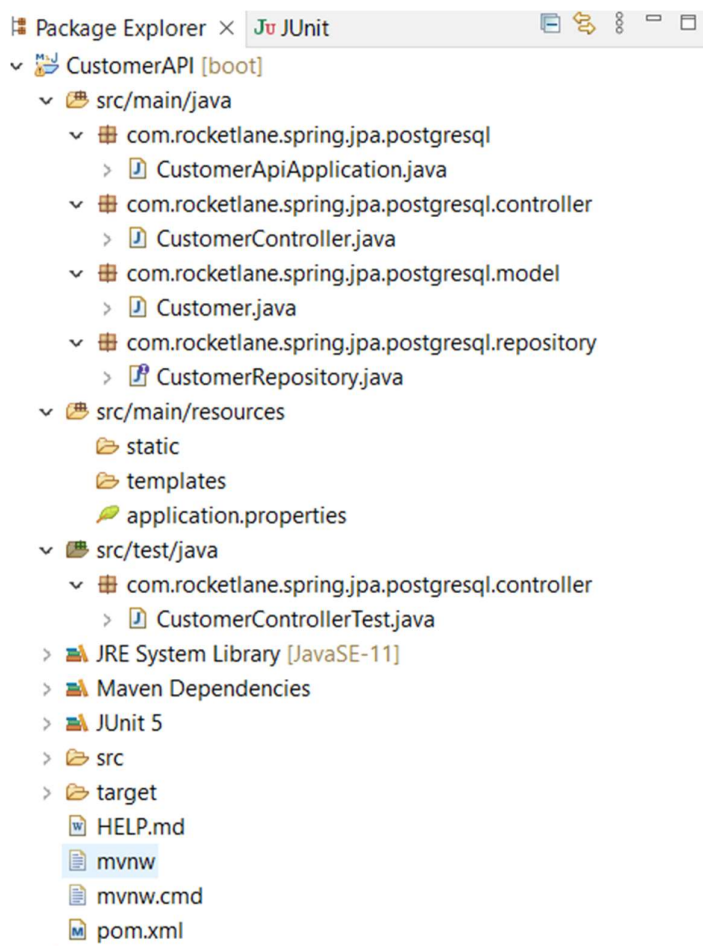
# ROCKETLANE CODING ASSIGNMENT

## SERVICE TO MANAGE CUSTOMER INFORMATION

Framework used: Spring Boot

Database: Postgresql

Project Layout:



## PACKAGES AND LAYERS IN THIS PROJECT

### 1. DOMAIN LAYER – Creating Customer Model

**Package:** COM.ROCKETLANE.SPRING.JPA.POSTGRESQL.MODEL

The data model (Customer) is defined with the fields: ID, FIRSTNAME, LASTNAME, EMAILID, MOBILENO, CITY and ADDRESS.

### 2. PERSISTENCE LAYER – Creating Customer Repository

**Package:** COM.ROCKETLANE.SPRING.JPA.POSTGRESQL.REPOSITORY

In the repository package, CustomerRepository interface extends JpaRepository .

It provides method for easy retrieval from the database without the need for manual implementation.

### 3. BUSINESS LAYER- Create Customer Controller

**Package:** COM.ROCKETLANE.SPRING.JPA.POSTGRESQL.CONTROLLER

Customer controller is the controller which exposes REST endpoints for the CRUD Operation.

#### **Create- HTTP POST METHOD**

One POST method to **create a new customer record** in the database.

#### **Read- HTTP GET METHOD**

Two GET methods to **retrieve customer records** from the database. One gets all the customer details and the other gets a single record given a customer ID.

#### **Update- HTTP PUT METHOD**

One PUT method to **update a customer record** in the database given the customer id of the record to be updated is given.

#### **Delete- HTTP DELETE METHOD**

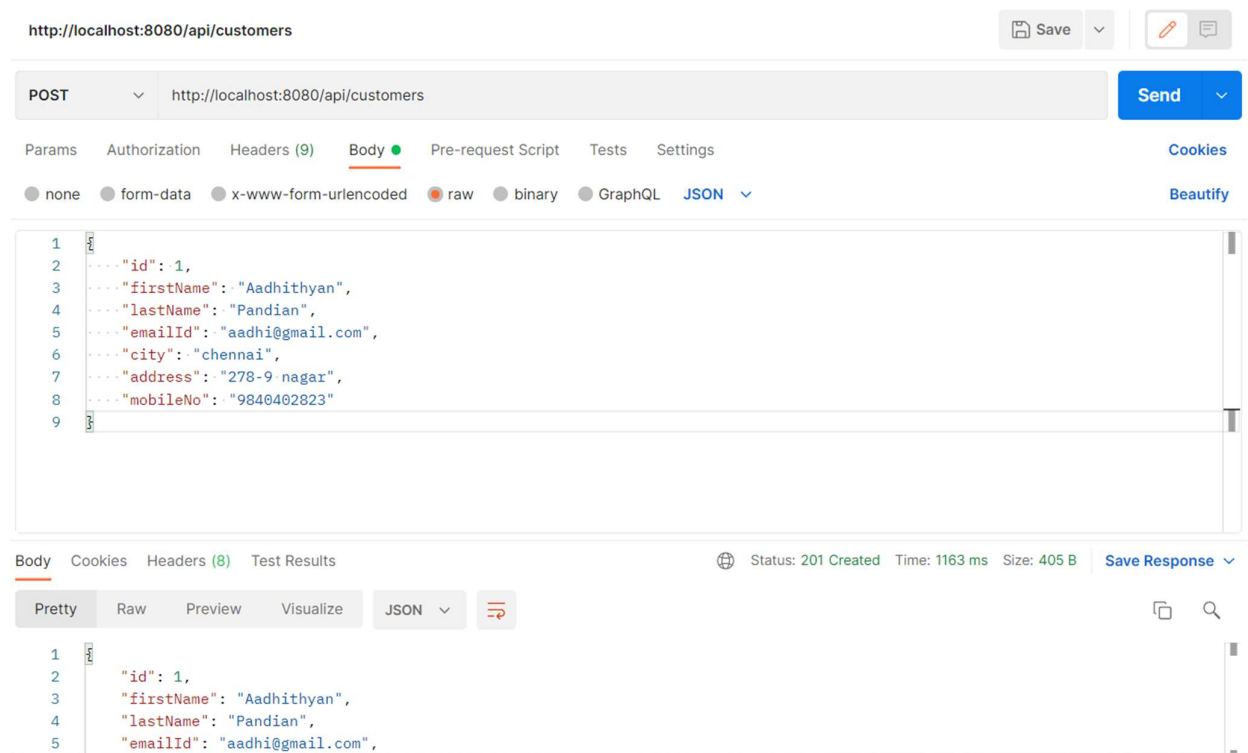
One DELETE method to **delete a customer record** in the database given the customer id of the record to be deleted.

## IMPLEMENTED API'S:

1. HTTP POST <http://localhost:8080/customers>
2. HTTP GET <http://localhost:8080/customers/2>
3. HTTP GET <http://localhost:8080/customers>
3. HTTP PUT <http://localhost:8080/customers/2>
4. HTTP DELETE <http://localhost:8080/customers/2>

## TESTING THE APIs IN POSTMAN:

Creating a new record using POST method:



Retrieving all customers detail with GET method:

http://localhost:8080/api/customers

GET http://localhost:8080/api/customers

Params Authorization Headers (9) Body Pre-request Script Tests Settings

Body Cookies Headers (8) Test Results

Status: 200 OK Time: 544 ms Size: 706 B

Save Response

```
1 {
2   {
3     "id": 1,
4     "firstName": "Aadhithyan",
5     "lastName": "Pandian",
6     "emailId": "aadhi@gmail.com",
7     "city": "chennai",
8     "address": "278-9 nagar",
9     "mobileNo": "9840402823"
10  },
11  {
12    "id": 2,
13    "firstName": "Aadil",
14    "lastName": "Rasheed",
15    "emailId": "aadil@gmail.com",
16    "city": "Srinagar",
17    "address": "Dal Street",
18    "mobileNo": "9840486534"
19  },
20  {
21    "id": 3,
22    "firstName": "Harikrishnan",
```

Retrieving customer details based on ID using GET method:

GET http://localhost:8080/api/customers/3

http://localhost:8080/api/customers/3

Params Authorization Headers (9) Body Pre-request Script Tests Settings

Body Cookies Headers (8) Test Results

Status: 200 OK Time: 54 ms Size: 413 B

Save Response

```
1 {
2   "id": 3,
3   "firstName": "Harikrishnan",
4   "lastName": "Srikrishnan",
5   "emailId": "hksh@gmail.com",
6   "city": "Dubai",
7   "address": "Vivekanandar Street",
8   "mobileNo": "342343212343"
9 }
```

Updating customer detail with given id using PUT method:

The screenshot shows a REST client interface with the following details:

- Method:** PUT
- URL:** http://localhost:8080/api/customers/2
- Body:** A JSON object representing a customer:

```
{  "id": 2,  "firstName": "Aadil",  "lastName": "Rasheed Lone",  "emailId": "arl@gmail.com",  "city": "Banglore",  "address": "Indiranagar",  "mobileNo": "9850502823"}
```
- Status:** 200 OK
- Time:** 56 ms
- Size:** 399 B
- Response:** The same JSON object as the request body is returned.

Deleting a customer detail with the given ID using DELETE method:

The screenshot shows a REST client interface with the following details:

- Method:** DELETE
- URL:** http://localhost:8080/api/customers/1
- Status:** 200 OK
- Time:** 56 ms
- Size:** 399 B
- Response:** The same JSON object as the request body is returned.

After deleting the customer with id 1:

GET http://localhost:80...

No Environment

http://localhost:8080/api/customers

Save

Send

GET http://localhost:8080/api/customers

Params Authorization Headers (9) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

Body Cookies Headers (8) Test Results

Status: 200 OK Time: 28 ms Size: 562 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 3,
3   "firstName": "Harikrishnan",
4   "lastName": "Srikrishnan",
5   "emailId": "hksk@gmail.com",
6   "city": "Dubai",
7   "address": "Vivekanandar Street",
8   "mobileNo": "342343212343"
9 },
10 {
11   "id": 2,
12   "firstName": "Aadil",
13   "lastName": "Rasheed Lone",
14   "emailId": "arl@gmail.com",
15   "city": "Bangalore",
16   "address": "Indiranagar",
17   "mobileNo": "9850502823"
18 }
```

SNAPSHOT OF THE PGADMIN IN POSTGRESQL:

PGAdmin

FileObjectToolsHelp

Browser

PostgreSQL 13

Databases (3)

customerdb

Casts

Catalogs

Event Triggers

Extensions

Foreign Data Wrappers

Languages

Publications

Schemas (1)

public

Collations

Domains

FTS Configurations

FTS Dictionaries

FTS Parsers

FTS Templates

Foreign Tables

Functions

Materialized Views

Procedures

Sequences

Tables (1)

customers

DashboardPropertiesSQLStatisticsDependenciesDependents

public.customers/customerdb/postgres@PostgreSQL 13

customerdb/postgres@PostgreSQL 13

Query EditorQuery HistoryScratch Pad

1SELECT \* FROM public.customers

2ORDER BY id ASC

Data OutputExplainMessagesNotifications

id	address	city	email_id	first_name	last_name	mobile_no
[PK] bigint	character varying (255)	character varying (255)	character varying (255)	character varying (255)	character varying (255)	character varying (255)
1	Indiranagar	Banglore	arl@gmail.com	Aadil	Rasheed Lone	9850502823
2	Vivekanandar Street	Dubai	hsk@gmail.com	Harikrishnan	Srikrishnan	342343212343