

# API Report: CustomerDetails (Spring Boot)

## Summary

This repository implements a Spring Boot REST API for managing customer details. Key points:

- Framework: Spring Boot (parent version 3.5.10).
- Java: 21.
- Data: Spring Data JPA with MySQL connector for runtime.
- Layers: controllers, DTOs, entities, repositories, services (with impl package).
- Typical controllers: `CustomerDetailsController`, `CustomerNameController`, `CustomerAddressController`, `CustomerContactController`, `CustomerIdentificationController`, `CustomerProofIdController`, `CustomerClassificationController`.
- Project built with Maven; `spring-boot-starter-web` and `spring-boot-starter-data-jpa` dependencies included.

The screenshots below show project structure, key controller and DTO classes, entity mappings, repository interfaces, and sample endpoints.

## Screenshots

(Images are embedded from the repository `s/` folder and are Postman request/response screenshots.)

### Screenshot 1

The screenshot shows a Postman collection named "Customer". It contains several requests, including a POST request to `http://localhost:8080/customers` with the following raw JSON body:

```

1 {
2     "customerIdentifier": "C500",
3     "customerFullName": "Yash Arora",
4     "customerDateOfBirth": "2003-01-01",
5     "customerGender": "Male",
6     "customerType": "Individual",
7     "customerStatus": "Active",
8     "customerCountryOfOrigin": "India",
9     "customerPreferredLanguage": "English"
10 }
11

```

The response status is 201 Created.

## Screenshot 2

The screenshot shows the Postman application interface. On the left, there's a sidebar with 'Collections' (Customer), 'Environments', 'History', 'Flows', and 'Files'. The main area shows a collection named 'Customer' with several API endpoints listed: POST CreateCust, GET GetCust, PUT PutCust, POST CustContact, GET CustContact, PUT CustContact, POST CustName, GET CustName, PUT CustName, POST CustAddr, GET CustAddr, PUT CustAddr, POST CustIden, GET CustIden, PUT CustIden, POST CustProof, GET CustProof, PUT CustProof, POST CustClass, GET CustClass, PUT CustClass. A specific endpoint, 'GET GetCust', is selected. The right panel displays the request details: method 'GET', URL 'http://localhost:8080/customers', and a table for 'Query Params'. Below this, the 'Body' tab shows a JSON response:

```

1 [ [
2   {
3     "customerIdentifier": "C500",
4     "customerFullName": "Yash Arora",
5     "customerGender": "Male",
6     "customerType": "Individual",
7     "customerDateOfBirth": "2003-01-01",
8     "customerPreferredLanguage": "English",
9     "customerStatus": "Active",
10    }
]

```

## Screenshot 3

The screenshot shows the Postman application interface. The sidebar and collection structure are identical to Screenshot 2. The selected endpoint is 'PUT PutCust'. The right panel shows the request details: method 'PUT', URL 'http://localhost:8080/customers/C500', and a table for 'Headers'. The 'Body' tab is selected, showing the raw JSON body:

```

1 {
2   "customerFullName": "Yash Updated",
3   "customerDateOfBirth": "2003-01-01",
4   "customerGender": "Male",
5   "customerType": "Individual",
6   "customerStatus": "Inactive",
7   "customerCountryOfOrigin": "India",
8   "customerPreferredLanguage": "Hindi"
9 }
10

```

## Screenshot 4

The screenshot shows the Postman application interface. On the left, there's a sidebar with 'Collections' (Customer), 'Environments', 'History', 'Flows', and 'Files (BETA)'. Below the sidebar, it says 'My Collection'. The main area has a search bar 'Search Postman' and tabs for 'POST Crea', 'GET GetC', 'PUT PutC', 'POST Cust', 'PUT PutCust', 'GET CustC', and 'PUT CustC'. The 'POST Cust' tab is selected. The URL is set to 'http://localhost:8080/customer-contacts'. The 'Body' tab is selected, showing a raw JSON payload:

```

1  {
2   "contactType": "Mobile",
3   "contactValue": "9876543210",
4   "customerIdentifier": "C500"
5 }

```

Below the body, the response status is '201 Created' with a duration of '41 ms' and a size of '256 B'. The response body is also shown in JSON format:

```

1  {
2   "id": 2,
3   "contactType": "Mobile",
4   "contactValue": "9876543210",
5   "customerIdentifier": "C500"
6 }

```

## Screenshot 5

The screenshot shows the Postman application interface. The sidebar and collection navigation are identical to Screenshot 5. The 'POST Cust' tab is still selected. The URL is now 'http://localhost:8080/customer-contacts'. The 'Params' tab is selected, showing a table for 'Query Params' with one row: 'Key' (Value) and 'Description' (Description). The 'Body' tab is selected again, showing a raw JSON response:

```

1  [
2   {
3    "id": 1,
4    "contactType": "Email",
5    "contactValue": "yash@email.com",
6    "customerIdentifier": "C500"
7   },
8   {
9    "id": 2,

```

## Screenshot 6

The screenshot shows the Postman application interface. The left sidebar displays a collection named "Customer" containing various API endpoints such as CreateCust, GetCust, PutCust, CustContact, CustName, CustAddr, CustIden, CustProof, CustClass, and CustClass. The main workspace shows a PUT request to "http://localhost:8080/customer-contacts/1". The "Body" tab is selected, showing the following raw JSON payload:

```

1 {
2   "contactType": "Email",
3   "contactValue": "yash@email.com",
4   "customerIdentifier": "C500"
5 }

```

The response section shows a 200 OK status with a response time of 24 ms and a body size of 254 B. The response JSON is identical to the request body.

Screenshot 7

The screenshot shows the Postman application interface. The left sidebar displays a collection named "Customer" containing various API endpoints such as CreateCust, GetCust, PutCust, CustContact, CustName, CustAddr, CustIden, CustProof, CustClass, and CustClass. The main workspace shows a POST request to "http://localhost:8080/customer-names". The "Body" tab is selected, showing the following raw JSON payload:

```

1 {
2   "nameType": "Legal",
3   "nameValue": "Yash Arora",
4   "customerIdentifier": "C500"
5 }

```

The response section shows a 201 Created status with a response time of 30 ms and a body size of 249 B. The response JSON is identical to the request body.

Screenshot 8

Customer / CustName

GET http://localhost:8080/customer-names

Key	Value	Description
		Description

```

1 [
2   {
3     "id": 1,
4     "nameType": "Nickname",
5     "nameValue": "Yash",
6     "customerIdentifier": "C500"
7   },
8   {
9     "id": 2,

```

## Screenshot 9

Customer / CustName

PUT http://localhost:8080/customer-names/1

Body

```

1 {
2   "nameType": "Nickname",
3   "nameValue": "Yash",
4   "customerIdentifier": "C500"
5 }

```

Key	Value	Description
		Description

```

1 [
2   {
3     "id": 1,
4     "nameType": "Nickname",
5     "nameValue": "Yash",
6     "customerIdentifier": "C500"
7   }

```

## Screenshot 10

The screenshot shows the Postman application interface. On the left, there's a sidebar with 'Collections' (Customer), 'Environments', 'History', 'Flows', and 'Files (BETA)'. Below the sidebar, it says 'My Collection'. The main area has tabs for 'Customer / CustAddr' and 'Customer / CustClass'. The 'Customer / CustAddr' tab is active. It shows a 'POST' request to 'http://localhost:8080/customer-addresses'. The 'Body' tab is selected, showing a raw JSON payload:

```

1  {
2   "addressType": "Permanent",
3   "addressLine": "123 MG Road",
4   "city": "Bangalore",
5   "state": "Karnataka",
6   "country": "India",
7   "postalCode": "560001",
8   "effectiveDate": "2024-01-01",
9   "customerIdentifier": "C500"
10 }

```

Below the request, the response is shown: '201 Created' with '38 ms' and '367 B'. The 'Body' tab shows the JSON response:

```

1  [
2   {
3     "id": 1,
4     "addressType": "Permanent",
5     "addressLine": "123 MG Road",
6     "city": "Bangalore",
7     "state": "Karnataka",
8     "country": "India",
9     "postalCode": "560001",
10    "effectiveDate": "2024-01-01",
11   }
12 ]

```

Screenshot 11

The screenshot shows the Postman application interface. The sidebar and collection structure are identical to Screenshot 11. The 'Customer / CustAddr' tab is active. It shows a 'GET' request to 'http://localhost:8080/customer-addresses'. The 'Params' tab is selected, showing a table for 'Query Params':

	Key	Value	Description	... Bulk Edit
	Key	Value	Description	

Below the request, the response is shown: '200 OK' with '22 ms' and '364 B'. The 'Body' tab shows the JSON response:

```

1  [
2   {
3     "id": 1,
4     "addressType": "Permanent",
5     "addressLine": "123 MG Road",
6     "city": "Bangalore",
7     "state": "Karnataka",
8     "country": "India",
9     "postalCode": "560001",
10    "effectiveDate": "2024-01-01",
11   }
12 ]

```

Screenshot 12

The screenshot shows the Postman interface with a collection named "Customer". A PUT request is selected with the URL `http://localhost:8080/customer-addresses/1`. The "Body" tab is active, showing the following JSON payload:

```

1  {
2   "addressType": "Temporary",
3   "addressLine": "456 New Street",
4   "city": "Mumbai",
5   "state": "Maharashtra",
6   "country": "India",
7   "postalCode": "400001",
8   "effectiveDate": "2024-02-01",
9   "customerIdentifier": "C500"
10 }

```

The response status is 200 OK with 25 ms latency and 364 B size.

## Screenshot 13

The screenshot shows the Postman interface with a collection named "Customer". A GET request is selected with the URL `http://localhost:8080/customer-identifications`. The "Params" tab is active, showing a table for query parameters:

	Key	Value	Description	... Bulk Edit
	Key	Value	Description	

The response status is 200 OK with 21 ms latency and 166 B size.

## Screenshot 14

POST <http://localhost:8080/customer-identifications>

```

1 {
2   "identificationType": "PAN",
3   "identificationNumber": "ABCDE1234F",
4   "customerIdentifier": "C500"
5 }

```

201 Created

```

1 {
2   "id": 1,
3   "identificationType": "PAN",
4   "identificationNumber": "ABCDE1234F",
5   "customerIdentifier": "C500"
6 }

```

## Screenshot 15

PUT <http://localhost:8080/customer-identifications/1>

```

1 {
2   "identificationType": "PAN",
3   "identificationNumber": "ABCDE9999Z",
4   "customerIdentifier": "C500"
5 }

```

200 OK

```

1 {
2   "id": 1,
3   "identificationType": "PAN",
4   "identificationNumber": "ABCDE9999Z",
5   "customerIdentifier": "C500"
6 }

```

## Screenshot 16

The screenshot shows the Postman application interface. On the left, there's a sidebar with 'Collections' (Customer), 'Environments', 'History', 'Flows', and 'Files (BETA)'. Below the sidebar, it says 'My Collection'. The main area has a search bar 'Search Postman' and tabs for 'Collections', 'Workspaces', and 'API Network'. A 'Customer / CustProof' collection is selected. Inside, a 'POST' request is shown with the URL 'http://localhost:8080/customer-proof-ids'. The 'Body' tab is selected, showing a raw JSON payload:

```

1  {
2    "proofType": "Passport",
3    "proofNumber": "P1234567",
4    "issueDate": "2020-01-01",
5    "expiryDate": "2030-01-01",
6    "customerIdentifier": "C500"
7  }

```

Below the request, the response is displayed: '201 Created' with a response time of '34 ms' and a size of '304 B'. The response body is also a raw JSON object:

```

1  [
2    {
3      "id": 2,
4      "proofType": "Passport",
5      "proofNumber": "P1234567",
6      "issueDate": "2020-01-01",
7      "expiryDate": "2030-01-01",
8      "customerIdentifier": "C500"
9    }

```

At the bottom, there are buttons for 'Runner', 'Start Proxy', 'Cookies', 'Vault', 'Trash', and 'AI'.

## Screenshot 17

This screenshot is similar to Screenshot 17, showing the same Postman interface and Customer collection. However, the request type is now 'GET' instead of 'POST'. The URL remains 'http://localhost:8080/customer-proof-ids'. The 'Params' tab is selected, showing a table for 'Query Params' with one row: 'Key' and 'Value'. The response is a '200 OK' status with a response time of '16 ms' and a size of '437 B'. The response body is identical to Screenshot 17:

```

1  [
2    {
3      "id": 2,
4      "proofType": "Passport",
5      "proofNumber": "P1234567",
6      "issueDate": "2020-01-01",
7      "expiryDate": "2030-01-01",
8      "customerIdentifier": "C500"
9    }

```

At the bottom, there are buttons for 'Runner', 'Start Proxy', 'Cookies', 'Vault', 'Trash', and 'AI'.

## Screenshot 18

The screenshot shows the Postman application interface. The left sidebar displays a collection named "Customer" containing various API endpoints such as CreateCust, GetCust, PutCust, etc. The main workspace shows a PUT request to "http://localhost:8080/customer-proof-ids/1". The "Body" tab is selected, showing the following raw JSON payload:

```

1  {
2    "proofType": "Passport",
3    "proofNumber": "P9999999",
4    "issueDate": "2021-01-01",
5    "expiryDate": "2031-01-01",
6    "customerIdentifier": "C500"
7  }

```

The response section shows a successful 200 OK status with 18 ms duration and 299 B size. The response body is identical to the sent payload.

Screenshot 19

The screenshot shows the Postman application interface. The left sidebar displays a collection named "Customer" containing various API endpoints such as CreateCust, GetCust, PutCust, etc. The main workspace shows a POST request to "http://localhost:8080/customer-classifications". The "Body" tab is selected, showing the following raw JSON payload:

```

1  {
2    "classificationType": "Risk",
3    "classificationValue": "Low",
4    "customerIdentifier": "C500"
5  }

```

The response section shows a successful 201 Created status with 59 ms duration and 261 B size. The response body is identical to the sent payload.

Screenshot 20

Customer / CustClass

GET http://localhost:8080/customer-classifications

Key	Value	Description	Bulk Edit
Key	Value	Description	

Body

```
{
  "id": 1,
  "classificationType": "Risk",
  "classificationValue": "High",
  "customerIdentifier": "C500"
}
```

200 OK • 16 ms • 259 B

## Screenshot 21

Customer / CustClass

PUT http://localhost:8080/customer-classifications/1

Body

```
{
  "id": 1,
  "classificationType": "Risk",
  "classificationValue": "High",
  "customerIdentifier": "C500"
}
```

Body

```
{
  "id": 1,
  "classificationType": "Risk",
  "classificationValue": "High",
  "customerIdentifier": "C500"
}
```

200 OK • 19 ms • 257 B