

Payment Application I

Problem Statement:

← Classroom

Hibernate and CRUD operations
Payment Application 1

?

V

≡

Problem Submissions Solutions Doubts

Payment Application 1

Easy • Score 80/90 • Advanced java • Spring Hibernate

Problem statement

Send feedback

You as a developer need to work on the **CNPayment Application**. The application manages payment transactions and stores payment data in a database. You need to complete the project based on the given template.

Tasks:-

1. In the **Payment** entity class, there are three attributes, **int id**, **String paymentType**, and **String description** for which you have to write the necessary methods with appropriate annotations.

2. Complete the **PaymentController** class, methods to handle HTTP requests with required annotations for the following APIs:

- GET "/payment/allPayments": It retrieves a list of all payments.
- GET "/payment/id/{id}" (@PathVariable int id) : It retrieves a specific payment by id.
- POST "/payment/save" (@RequestBody Payment payment): It saves a new payment.
- GET "/payment/paymentType/{paymentType}" (@PathVariable String paymentType): It retrieves a payment by type.
- GET "/payment/description/{keyword}" (@PathVariable String keyword): It retrieves a payment by description keyword.

3. Complete the **PaymentService** class as mentioned below:

a. Autowire PaymentDAL.

b. Complete the following methods:

- getPaymentById(int id): This method fetches payment from the dal layer for a specific id.
- getPaymentByPaymentType(String paymentType): This method fetches a list of payments from the dal layer based on the paymentType received.
- getPaymentByDescriptionKeyword(String keyword): This method fetches a list of payments from the dal layer based on the keyword received.

Step 1
Download starter kit

Step 2
Complete project on local IDE

Step 3
Export code as .zip file

Step 4
Upload .zip file max 50mb

Drag and drop .zip here or [browse](#)

Submit

Problem Submissions Solutions Doubts

- getPaymentByDescriptionKeyword(String keyword): This method fetches a list of payments from the dal layer based on the keyword received.
- getAllPayments(): This method fetches a list of payments from the dal layer.
- addPayment(Payment payment): This method saves payment entity into the database using the dal layer.

4. Complete the **PaymentDALImpl** class as mentioned below:

a. Autowire EntityManager.

b. Override the following methods:

- getById(int id): This method fetches the payment entity from the database for a specific id.
- getAllPayments(): This method fetches the list of payments from the database.
- getByPaymentType(String paymentType): This method fetches the list of payments from the database based on the paymentType received.
- getByPaymentDescription(String keyword): This method fetches the list of payments from the database based on the keyword received.
- addPayment(Payment payment): This method saves a payment entity into the database.

5. Complete the following **Custom Exceptions** classes with the required methods and their implementations in the **PaymentService** class:

a. **ElementAlreadyExistsException**: Throws an exception if "id" already exists while posting a new payment.

b. **NotFoundException**: Throws an exception if "id" is not present while fetching a payment.

c. **InvalidInputException**: Throws an exception when the given "paymentType" is invalid or does not exist. Accepted values for "paymentType" are "Credit", "Debit" & "Cash".

6. Test the application using tools such as Postman to ensure data is saved and retrieved correctly from the database.

Output:

POST ▼ http://localhost:8080/payment/save Send ▼

Params ● Authorization ● Headers (10) **Body** ● Pre-request Script Tests Settings Cookies

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary JSON ▼ Beautify

```
1 {
2   "id": 5,
3   "paymentType": "Debit Card",
4   "description": "payment done with debit card"
5 }
```

Body Cookies Headers (4) Test Results 🌐 200 OK 454 ms 123 B Save Response ▼

Pretty Raw Preview Visualize Text ▼ ↺ 📄 🔍

1

GET ▼ http://localhost:8080/payment/description/Cash Send ▼

Params ● Authorization ● Headers (8) **Body** ● Pre-request Script Tests Settings Cookies

Key	Value	Bulk Edit
file		

Body Cookies Headers (5) Test Results 🌐 200 OK 20 ms 315 B Save Response ▼

Pretty Raw Preview Visualize JSON ▼ ↺ 📄 🔍

```
1 [
2   {
3     "id": 7,
4     "paymentType": "Cash",
5     "description": "payment done with Cash"
6   },
7   {
8     "id": 8,
9     "paymentType": "Cash",
10    "description": "payment done with Cash on Delivery"
11  }
12 ]
```

GET ▼ http://localhost:8080/payment/id/5 Send ▼

Params ● Authorization ● Headers (8) **Body** ● Pre-request Script Tests Settings Cookies

☒ none ☐ form-data ☐ x-www-form-urlencoded ☐ raw ☐ binary

This request does not have a body

Body Cookies Headers (5) Test Results 🌐 200 OK 417 ms 244 B Save Response ▼

Pretty Raw Preview Visualize JSON ▼ ↺ 📄 🔍

```
1 {
2   "id": 5,
3   "paymentType": "Debit Card",
4   "description": "payment done with debit card"
5 }
```

GET

http://localhost:8080/payment/allPayments

Send

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

Cookies

Query Params

	Key	Value	Bulk Edit
--	-----	-------	-----------

Body

Cookies

Headers (5)

Test Results

200 OK

11 ms

479 B

Save Response

Pretty

Raw

Preview

Visualize

JSON

```
2      {
3        "id": 5,
4        "paymentType": "Debit Card",
5        "description": "payment done with debit card"
6      },
7      {
8        "id": 6,
9        "paymentType": "Credit Card",
10       "description": "payment done with credit card"
11     },
12     {
13       "id": 7,
14       "paymentType": "Cash",
15       "description": "payment done with Cash"
16     },
```