

Payment Application II

Problem Statement:

← Classroom

Hibernate and CRUD operations
Payment Application 2

2 V

Problem Submissions Solutions Doubts

PaymentApplication2

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

Problem statement

In continuation to your **CNPayment Application**. The application should now also allow the feature to update and delete a payment. Based on the given template complete the project.

Tasks:-

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

- PUT "/payment/update": It updates a payment record in the database.
- DELETE "/payment/delete/id/{id}": It deletes a payment record from the database.
- PUT "/payment/update/{id}/description/{description}": It updates the payment description for a specific ID.

2. Complete the **PaymentService** class methods mentioned below:

- update(Payment updatePayment): This method updates a payment with the help of dal layer.
- updateDescription(int id, String description): This method updates the description of a payment for a specific id from the dal layer.
- delete(int id): This method deletes a payment for a specific ID from the database.

3. Complete the **PaymentDALImpl** class methods as mentioned below:

a. Override the following methods:

- update(Payment updatePayment): The method updates a payment record in the database.
- updateDescription(int paymentId, String description): This method updates the payment description for a specific ID in the database.
- delete(int paymentId): This method deletes a payment for a specific ID from the database.

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

► **Special Instructions for submitting the solution:**

Send feedback

1 Step 1
Download starter kit

2 Step 2
Complete project on local IDE

3 Step 3
Export code as .zip file

4 Step 4
Upload .zip file max 50mb

Drag and drop .zip here or browse

Submit

Output:

The screenshot shows a REST client interface with a PUT request to `localhost:8080/payment/update`. The 'Body' tab is selected, showing a JSON payload:

```
1 {  
2   "id": 2,  
3   "paymentType": "Credit Payment",  
4   "description": "Sample description for credit"  
5 }
```

Below the body editor, the 'Body' tab is active, displaying the response status: `200 OK`, `26 ms`, and `123 B`. The response is shown in 'Pretty' format.

The screenshot shows a REST client interface with a DELETE request to `localhost:8080/payment/delete/id/1`. The 'Params' tab is selected, showing a table for query parameters:

Key	Value	Description
Key	Value	Description

Below the params section, the 'Body' tab is active, displaying the response status: `200 OK`, `346 ms`, and `123 B`. The response is shown in 'Pretty' format.