**Spring data JPA part 2**

**-Akanksha Tyagi**

**id-4701**

**JPQL:**

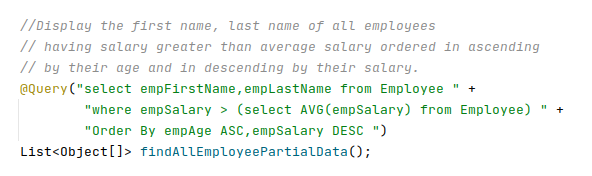
**Instructions for JPQL and Native SQL Query**

Create an employeeTable table with the following fields: empId, empFirstName, empLastName, empSalary, empAge.

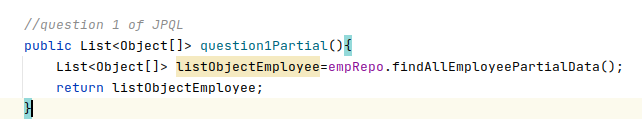
Create an Employee entity having following fields: id, firstName, lastName, salary, age which maps to the table columns given in above.

**1. Display the first name, last name of all employees having salary greater than average salary ordered in ascending by their age and in descending by their salary.**

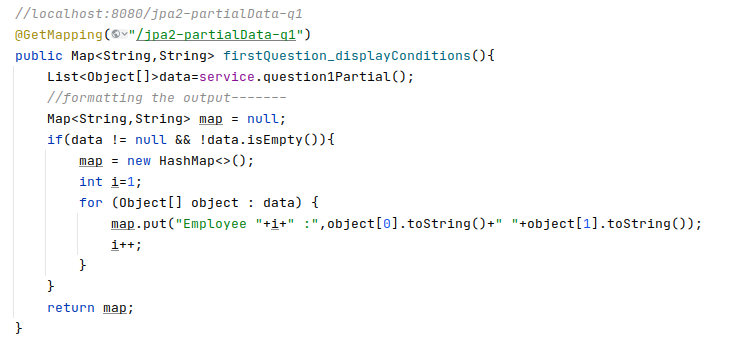
EmployeeRepository:



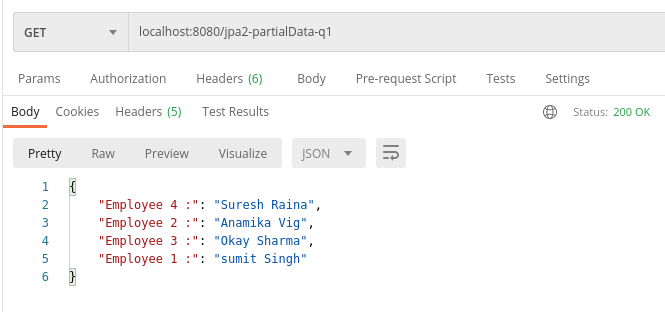
EmployeeService:



EmployeeController:



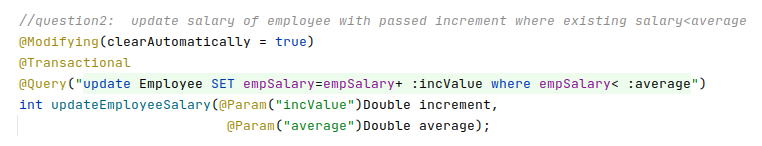
**Output:** <http://localhost:8080/jpa2-partialData-q1>



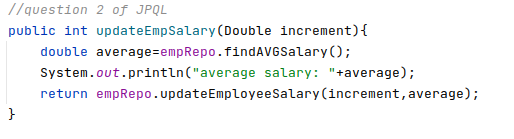
**-------------------------------------------------------------------------------------------------------------------------**

**2: Update salary of all employees by a salary passed as a parameter whose existing salary is less than the average salary.**

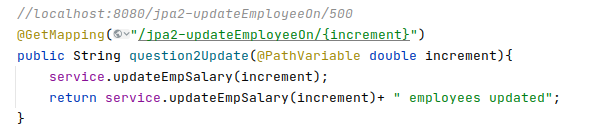
EmployeeRepository:

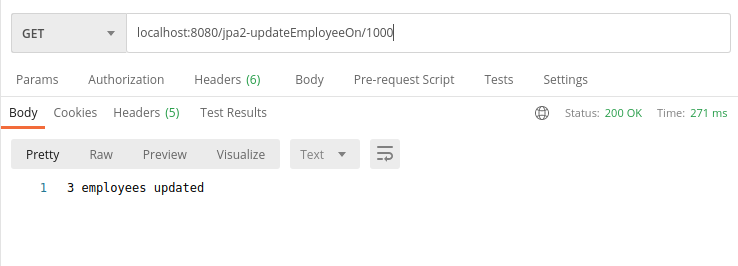


EmployeeService:

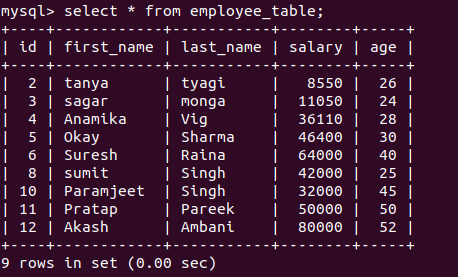


EmployeeController:

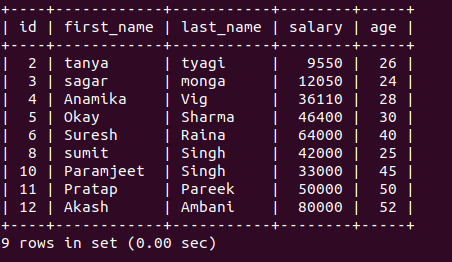


**Output:** <http://localhost:8080/jpa2-updateEmployeeOn/1000> 

Sql: before updating the salary:



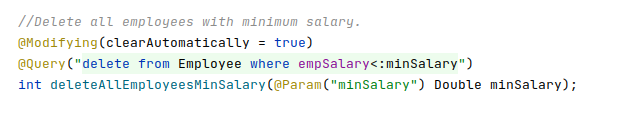
After updating the salary:



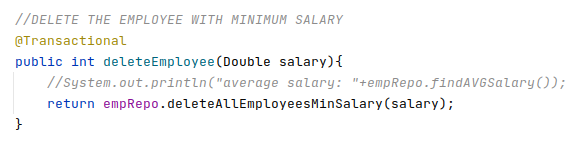
**-------------------------------------------------------------------------------------------------------------------------**

**3. Delete all employees with minimum salary.**

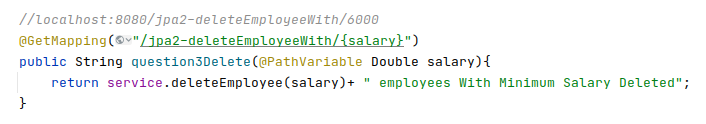
EmployeeRepository:



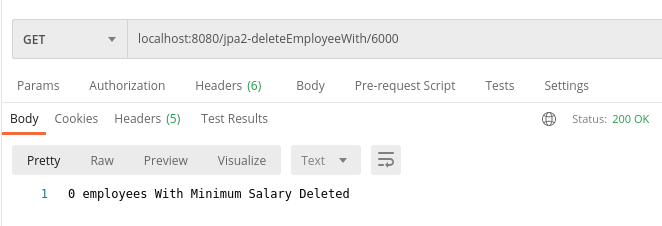
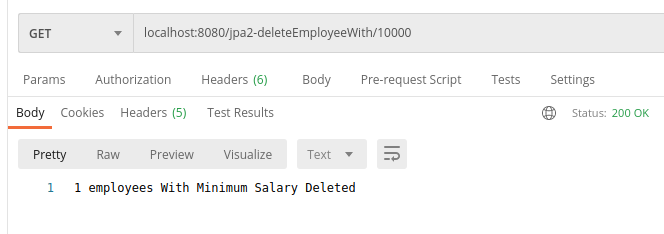
EmployeeService:



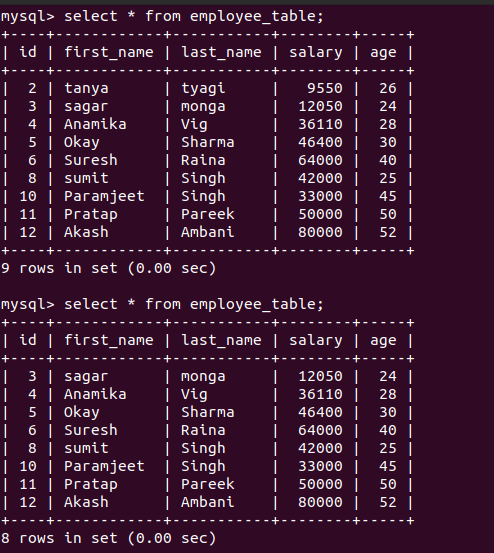
EmployeeController:

****

**Output:**

[http://localhost:8080/jpa2-deleteEmployeeWith/6000](http://localhost:8080/jpa2-deleteEmployeeWith/10000) ****<http://localhost:8080/jpa2-deleteEmployeeWith/10000> ****

**Sql:**

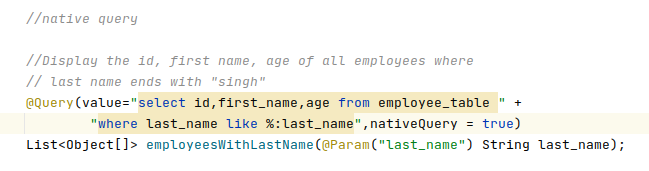
****

**-------------------------------------------------------------------------------------------------------------------------**

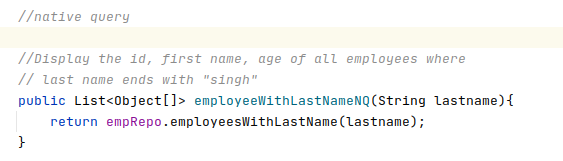
**Native Query:**

**1. Display the id, first name, age of all employees where last name ends with "singh".**

EmployeeRepository:



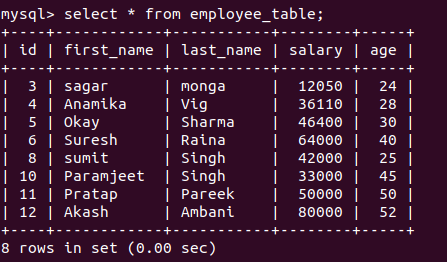
EmployeeService:

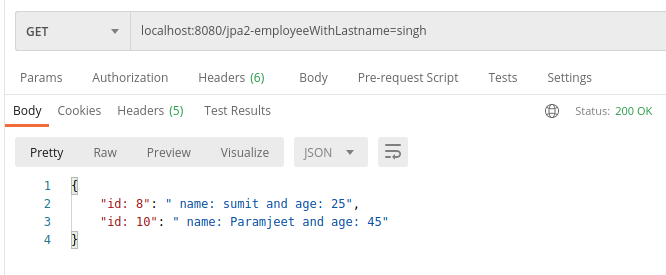


EmployeeController:

****

**Output:** <http://localhost:8080/jpa2-employeeWithLastname=singh>

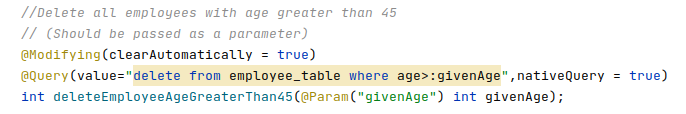
****

****

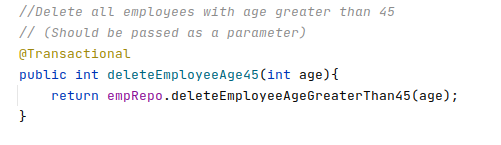
**-------------------------------------------------------------------------------------------------------------------------**

**2. Delete all employees with age greater than 45(Should be passed as a parameter).**

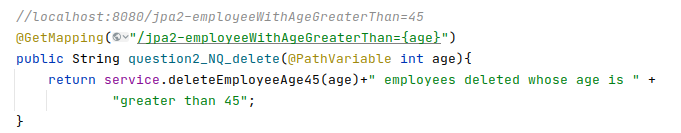
EmployeeRepository:



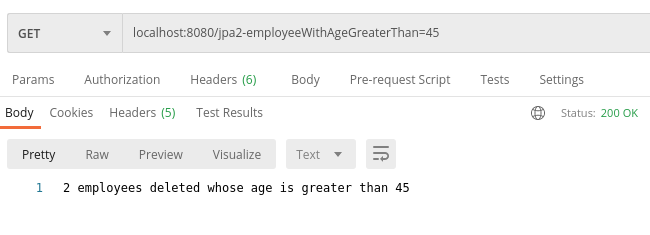
EmployeeService:



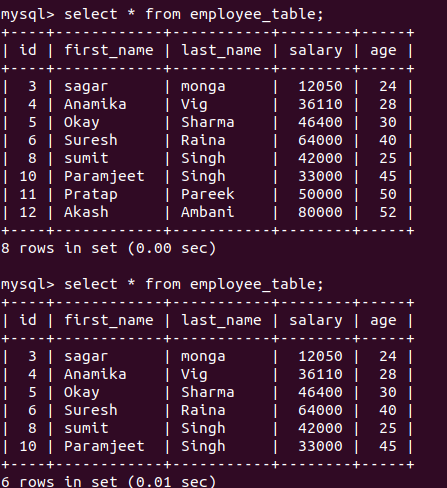
EmployeeController:

****

**Output:** <http://localhost:8080/jpa2-employeeWithAgeGreaterThan=45>



Sql :

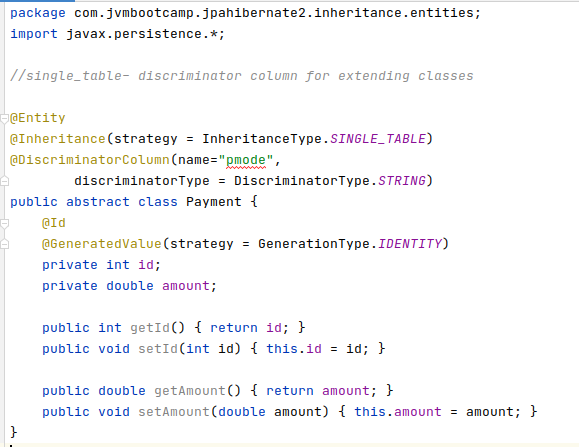


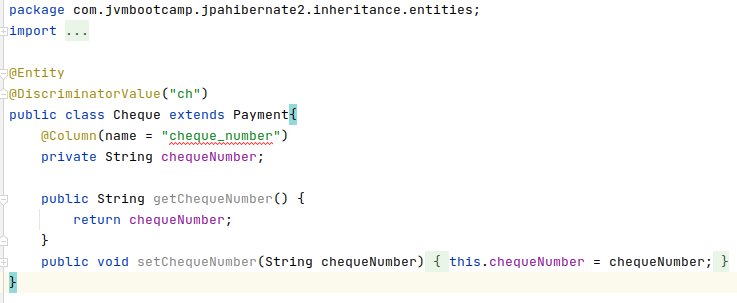
**-------------------------------------------------------------------------------------------------------------------------**

**Inheritance Mapping:**

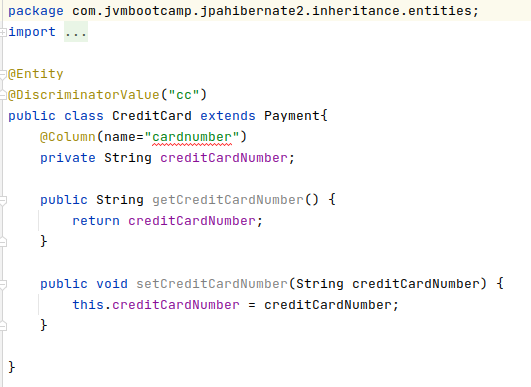
**1. Implement and demonstrate Single Table strategy.**

Payment.java

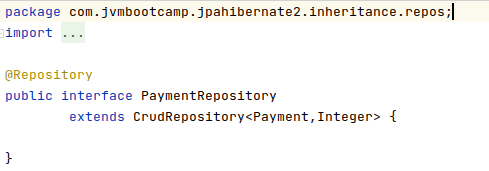
****

Cheque.java****

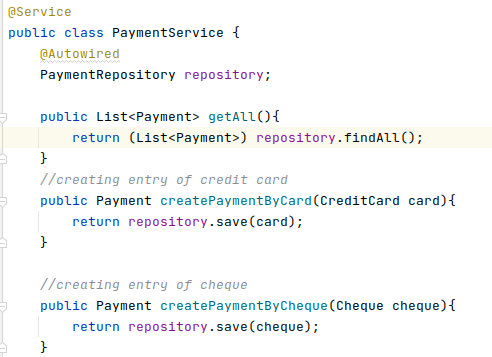
CreditCard.java

****

PaymentRepository.java



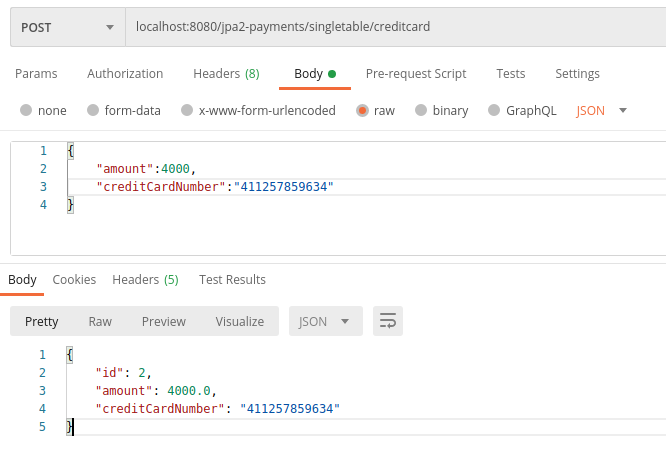
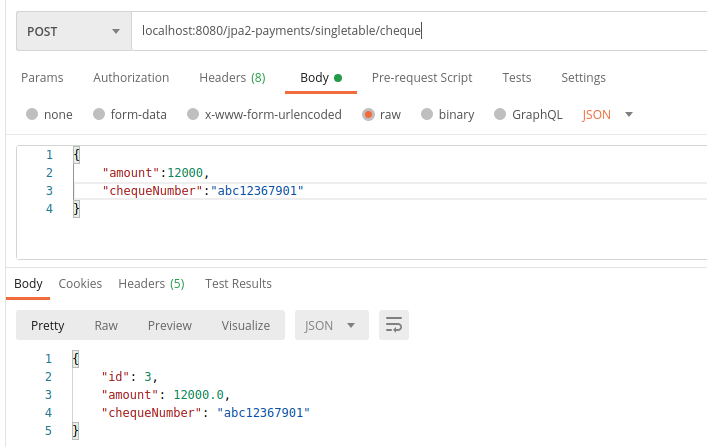
PaymentService.java

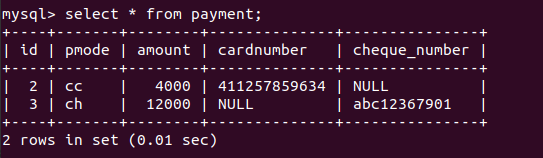


PaymentController.java



Output:

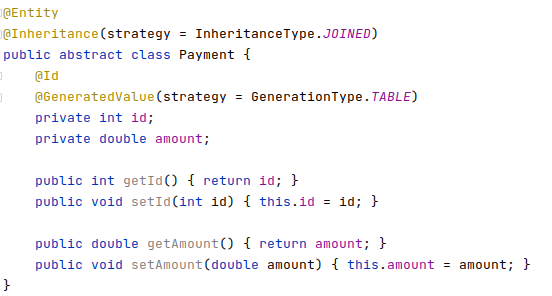
credit card: <http://localhost:8080/jpa2-payments/singletable/creditcard> (in post)Cheque:<http://localhost:8080/jpa2-payments/singletable/cheque> (in post) 

getAllData:

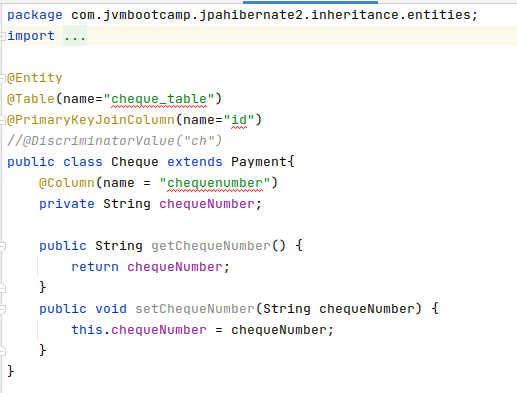
**-------------------------------------------------------------------------------------------------------------------------**

**2. Implement and demonstrate Joined strategy.**

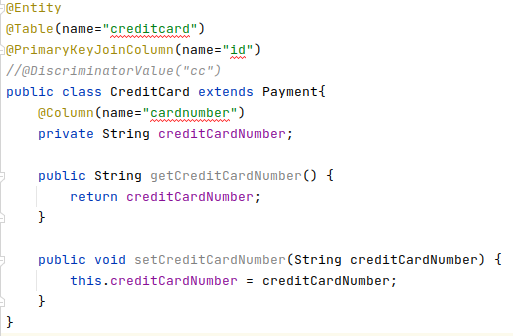
Payment.java



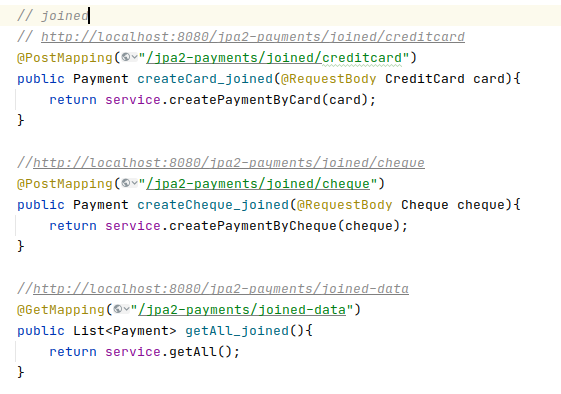
Cheque.java



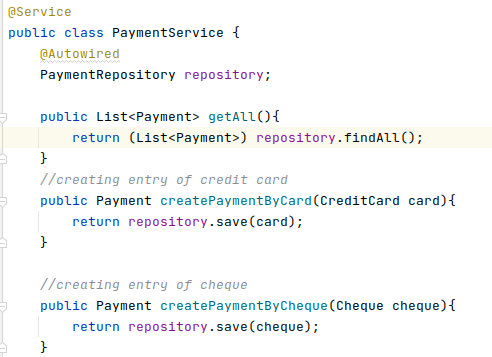
CreditCard.java

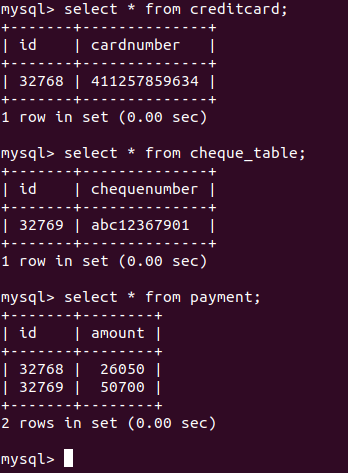


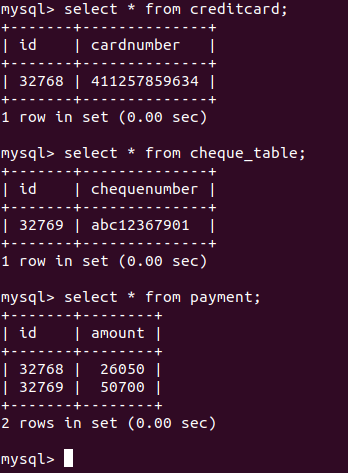
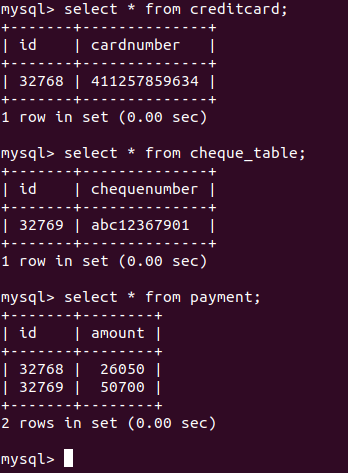
PaymentController.java:



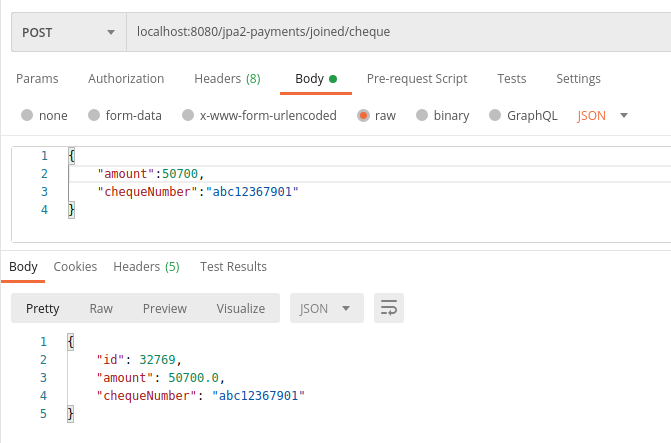
PaymentService.java

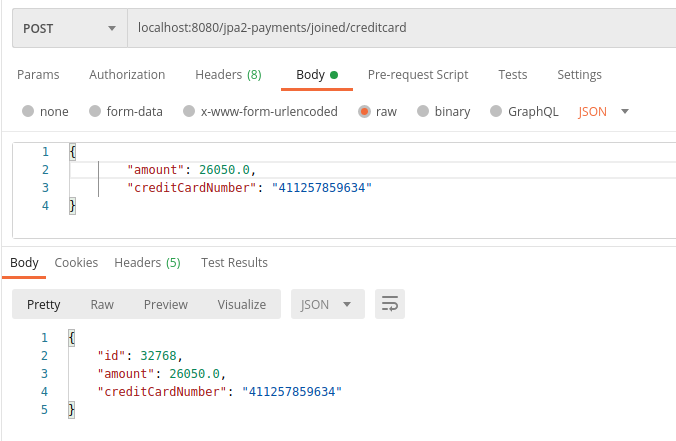


Output: 



Output: <http://localhost:8080/jpa2-payments/joined/cheque> (in post)



<http://localhost:8080/jpa2-payments/joined/creditcard> (in post)

**-------------------------------------------------------------------------------------------------------------------------**

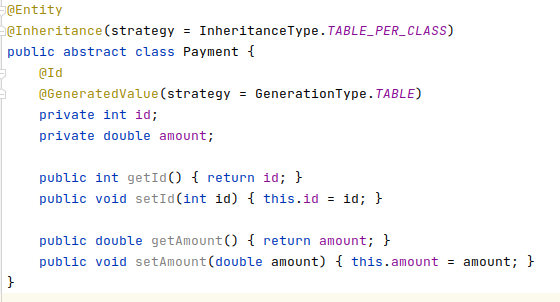
**3. Implement and demonstrate Table Per Class strategy.**

**Edit application.properties as follows:**

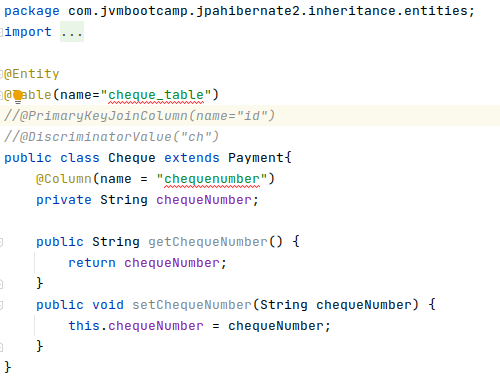
***[ #this is for table per class inheritance***

**spring.jpa.hibernate.ddl-auto = update ]**

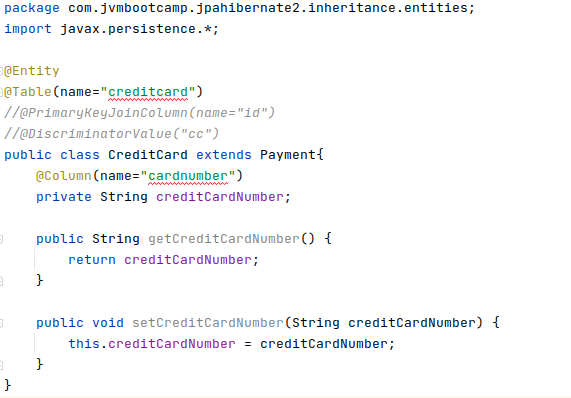
Payment.java



Cheque.java



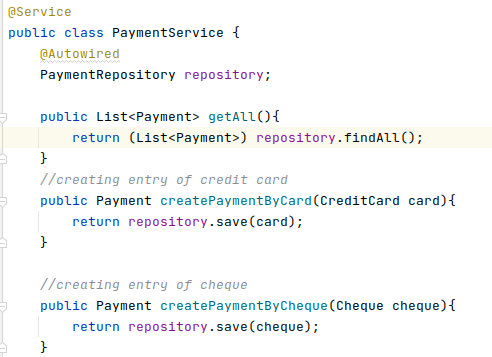
CreditCard.java



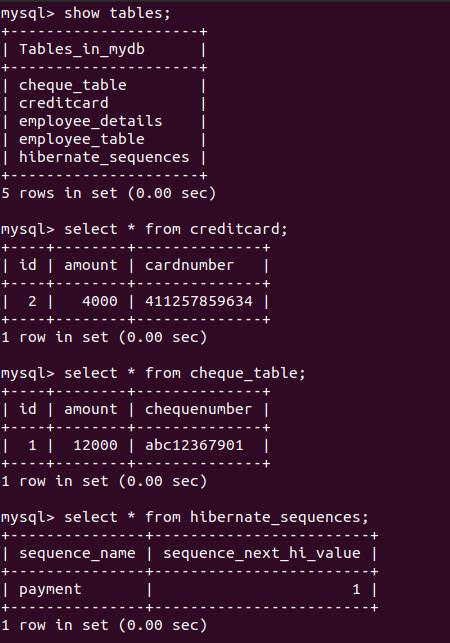
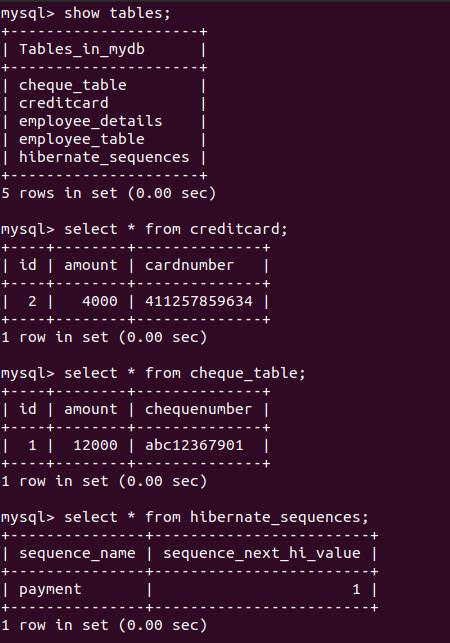
PaymentController.java

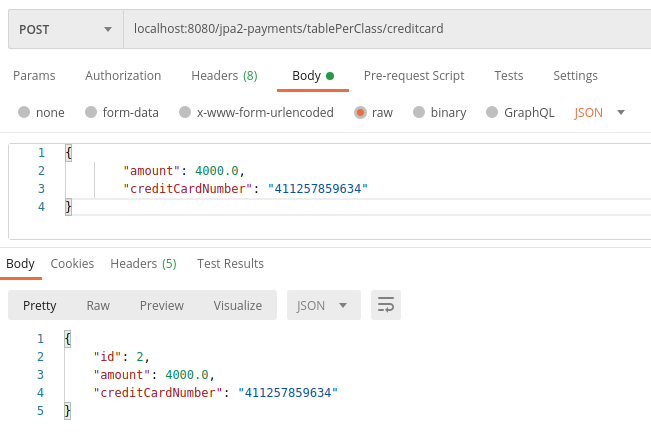
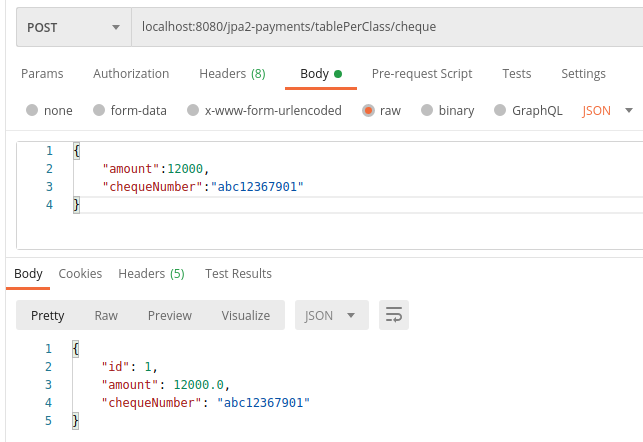


PaymentService.java



Output:



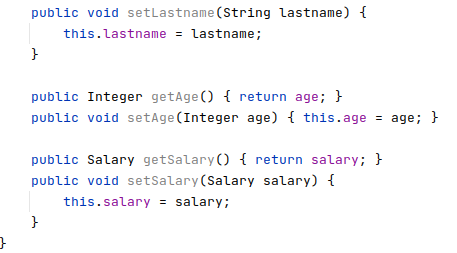
****

**-------------------------------------------------------------------------------------------------------------------------**

**Component Mapping:**

**1. Implement and demonstrate Embedded mapping using employee table having following fields: id, firstName, lastName, age, basicSalary, bonusSalary, taxAmount, specialAllowanceSalary.**

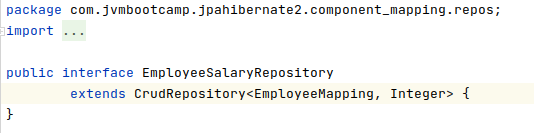
EmployeeMapping.java



Salary.java



EmployeeSalaryRepository.java



EmployeeSalaryService.java

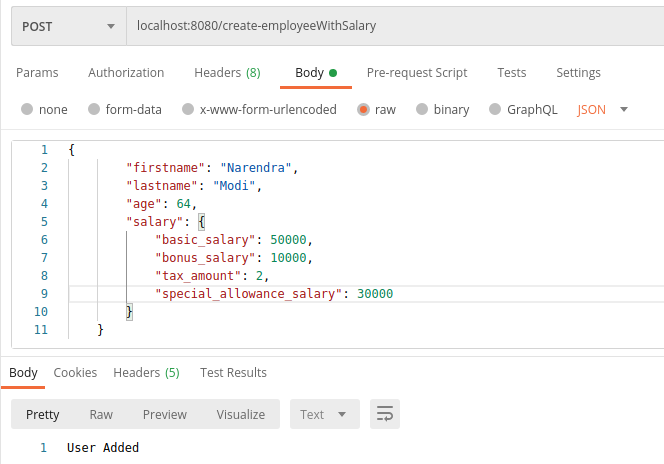


EmployeeSalaryController.java



**Output:**

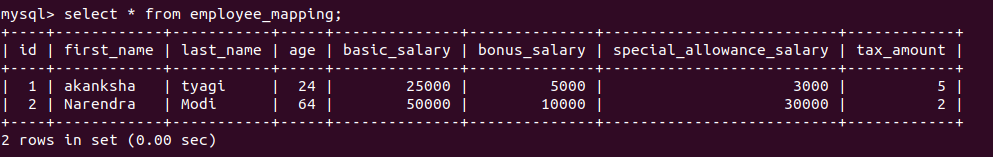
Creating employee with salary:

<https://localhost:8080/create-employeeWithSalary> (in post)****

Getting employee details with salary break down:

<http://localhost:8080/allEmployeeWithSalary>



Sql: 

**--------------------------------------------------------------------------------------------------------------------------------------**