Relationships between Entities:

OneToOne

OneToMany

ManyToOne

ManyToMany

Many To Many

Customer

Product

The relationship between these 2 entities is many to many. So the relationship will become a new entity.

Categories.java

@ManyToMany(targetEntity=Item.class,cascade=CascadeType.ALL)

@JoinTable(name="categories\_items",joinColumns=@JoinColumn(name="cat\_id\_fk",referencedColumnName="catid"),inverseJoinColumns=@JoinColumn(name="item\_id\_fk",referencedColumnName="itemid"))

private Set items;

Demo:

Many To Many

Customer

Product

package com.ust.spring.entity;

import java.util.List;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.ManyToMany;

import javax.persistence.Table;

import com.sun.istack.NotNull;

@Entity

@Table(name = "CUST")

public class Customer {

@Id

@Column(name = "id")

@GeneratedValue(strategy = GenerationType.AUTO)

private Integer id;

@NotNull

private String name;

private String address;

@ManyToMany

private List<Product> products;

public Customer() {}

public Customer(Integer id, String name, String address) {

super();

this.id = id;

this.name = name;

this.address = address;

}

public Integer getId() {

return id;

}

public void setId(Integer id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getAddress() {

return address;

}

public void setAddress(String address) {

this.address = address;

}

public List<Product> getProducts() {

return products;

}

public void setProducts(List<Product> products) {

this.products = products;

}

@Override

public String toString() {

return "Customer [id=" + id + ", name=" + name + ", address=" + address + "]";

}

}

package com.ust.spring.entity;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

@Entity

public class Product {

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

private Integer id;

private String name;

private Double price;

public Product() {}

public Product(Integer id, String name, Double price) {

super();

this.id = id;

this.name = name;

this.price = price;

}

public Integer getId() {

return id;

}

public void setId(Integer id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public Double getPrice() {

return price;

}

public void setPrice(Double price) {

this.price = price;

}

@Override

public String toString() {

return "Product [id=" + id + ", name=" + name + ", price=" + price + "]";

}

}

package com.ust.spring.repository;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.stereotype.Repository;

import com.ust.spring.entity.Customer;

@Repository

public interface CustomerRepository extends JpaRepository<Customer, Integer>

{

}

package com.ust.spring.repository;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.stereotype.Repository;

import com.ust.spring.entity.Product;

@Repository

public interface ProductRepository extends JpaRepository<Product, Integer>

{

}

package com.ust.spring.controller;

import java.util.List;

import java.util.Optional;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.DeleteMapping;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.PutMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import com.ust.spring.entity.Product;

import com.ust.spring.repository.ProductRepository;

class ProductAlreadyExistException extends RuntimeException

{

public ProductAlreadyExistException(String message)

{

super(message);

}

}

@RestController

@RequestMapping("/product")

public class ProductController {

@Autowired

private ProductRepository pr;

@PostMapping

public Product addProduct(@RequestBody Product product) {

System.out.println(product);

if(product.getId()==null)

{

return pr.save(product);

}

Product temp=findProductById(product.getId());

if(temp==null)

{

pr.save(product);

}else

{

throw new ProductAlreadyExistException("Product with ID:"+product.getId()+" Already exist");

}

return product;

}

@GetMapping("/{id}")

public Product findProductById(@PathVariable("id")Integer id) {

Optional<Product> temp=pr.findById(id);

Product product=null;

if(temp.isPresent())

{

product=temp.get();

}

return product;

}

@GetMapping

public List<Product> retrieveAllProducts() {

return pr.findAll();

}

@PutMapping("/{id}")

public Product updateProduct(@PathVariable("id") Integer id,@RequestBody Product product) {

Product existing = findProductById(id);

if(existing!=null)

{

existing=product;

pr.save(existing);

}

return existing;

}

@DeleteMapping("/{id}")

public Product delete(@PathVariable("id") Integer id) {

Product temp=findProductById(id);

if(temp!=null)

{

pr.delete(temp);

}

return temp;

}

}

package com.ust.spring.controller;

import java.util.List;

import java.util.Optional;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.DeleteMapping;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.PutMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import com.ust.spring.entity.Customer;

import com.ust.spring.repository.CustomerRepository;

class CustomerAlreadyExistException extends RuntimeException

{

public CustomerAlreadyExistException(String message)

{

super(message);

}

}

@RestController

@RequestMapping("/customer")

public class CustomerController {

@Autowired

private CustomerRepository pr;

@PostMapping

public Customer addCustomer(@RequestBody Customer customer) {

System.out.println(customer);

if(customer.getId()==null)

{

return pr.save(customer);

}

Customer temp=findCustomerById(customer.getId());

if(temp==null)

{

pr.save(customer);

}else

{

throw new CustomerAlreadyExistException("Customer with ID:"+customer.getId()+" Already exist");

}

return customer;

}

@GetMapping("/{id}")

public Customer findCustomerById(@PathVariable("id")Integer id) {

Optional<Customer> temp=pr.findById(id);

Customer customer=null;

if(temp.isPresent())

{

customer=temp.get();

}

return customer;

}

@GetMapping

public List<Customer> retrieveAllCustomers() {

return pr.findAll();

}

@PutMapping("/{id}")

public Customer updateCustomer(@PathVariable("id") Integer id,@RequestBody Customer customer) {

Customer existing = findCustomerById(id);

if(existing!=null)

{

existing=customer;

pr.save(existing);

}

return existing;

}

@DeleteMapping("/{id}")

public Customer delete(@PathVariable("id") Integer id) {

Customer temp=findCustomerById(id);

if(temp!=null)

{

pr.delete(temp);

}

return temp;

}

}

Preparation for Module 3 assessment:

1. Practise the spring boot REST api using Visual Studio code.
2. Run the spring boot project from vs code terminal.
3. Open git bash terminal in vs code and curl the URLs using GET, POST, PUT, DELETE
4. Watch WECP platform usage video