**Layers in Application**

**Service Layer:**

The Service Layer contains business logic. It acts as an intermediary between the Controller layer (which handles HTTP requests) and the Repository layer (which interacts with the database). Services encapsulate business rules and logic and often coordinate transactions.

**Repository Layer:**

The Repository Layer handles data access. This layer abstracts the data storage details and provides CRUD (Create, Read, Update, and Delete) operations. In Spring, the @Repository annotation is used, and often the Spring Data JPA JpaRepository interface is extended to provide these operations without needing to write implementation code.

**DTO (Data Transfer Object):**

DTOs are simple objects used to transfer data between different layers of the application, especially between the client and server. They are not entities and do not contain business logic. They are used to encapsulate data and reduce the number of method calls between layers.

**Logic Flow:**

1. **Controller**: Handles incoming HTTP requests and delegates business logic to the Service layer.
2. **Service**: Contains business logic. It interacts with the Repository layer to fetch or save data and may convert between DTOs and Entities.
3. **Repository**: Interacts with the database to perform CRUD operations.
4. **Entity**: Represents a table in the database.
5. **DTO**: Used to transfer data between layers, particularly between the Service layer and the Controller layer.

**Example Application:**

Let's create a simple application for managing customers.

1. **Entity:**

package com.example.medicalstore.model;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

@Entity

public class Customer {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Integer id;

private String name;

private String contact;

private String address;

// Getters and Setters

}

1. DTO:

package com.example.medicalstore.dto;

public class CustomerDTO {

private Integer id;

private String name;

private String contact;

private String address;

// Getters and Setters

}

1. Repository:

package com.example.medicalstore.repository;

import com.example.medicalstore.model.Customer;

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

public interface CustomerRepository extends JpaRepository<Customer, Integer> {

}

1. Service:

package com.example.medicalstore.service;

import com.example.medicalstore.dto.CustomerDTO;

import com.example.medicalstore.model.Customer;

import com.example.medicalstore.repository.CustomerRepository;

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

import org.springframework.stereotype.Service;

import java.util.List;

import java.util.stream.Collectors;

@Service

public class CustomerService {

@Autowired

private CustomerRepository customerRepository;

public List<CustomerDTO> getAllCustomers() {

return customerRepository.findAll().stream().map(this::convertToDTO).collect(Collectors.toList());

}

public void saveCustomer(CustomerDTO customerDTO) {

Customer customer = convertToEntity(customerDTO);

customerRepository.save(customer);

}

private CustomerDTO convertToDTO(Customer customer) {

CustomerDTO dto = new CustomerDTO();

dto.setId(customer.getId());

dto.setName(customer.getName());

dto.setContact(customer.getContact());

dto.setAddress(customer.getAddress());

return dto;

}

private Customer convertToEntity(CustomerDTO dto) {

Customer customer = new Customer();

customer.setId(dto.getId());

customer.setName(dto.getName());

customer.setContact(dto.getContact());

customer.setAddress(dto.getAddress());

return customer;

}

}

1. Controller:

package com.example.medicalstore.controller;

import com.example.medicalstore.dto.CustomerDTO;

import com.example.medicalstore.service.CustomerService;

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

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

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

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

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

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

import java.util.List;

@Controller

@RequestMapping("/customers")

public class CustomerController {

@Autowired

private CustomerService customerService;

@GetMapping("")

public String getAllCustomers(Model model) {

List<CustomerDTO> customers = customerService.getAllCustomers();

model.addAttribute("customers", customers);

return "customers";

}

@GetMapping("/new")

public String createCustomerForm(Model model) {

model.addAttribute("customer", new CustomerDTO());

return "create\_customer";

}

@PostMapping("")

public String saveCustomer(@ModelAttribute("customer") CustomerDTO customerDTO) {

customerService.saveCustomer(customerDTO);

return "redirect:/customers";

}

}