**Exercise 11: Implementing Dependency Injection**

**Scenario:**

You are developing a customer management application where the service class depends on a repository class. Use Dependency Injection to manage these dependencies.

**Steps:**

**1. Create a New Java Project:**

* **Project Name**: DependencyInjectionExample
* Use your preferred IDE to create a new Java project.

**2. Define Repository Interface:**

**CustomerRepository Interface**:

* Create an interface CustomerRepository with a method to find a customer by ID.

public interface CustomerRepository {

Customer findCustomerById(String id);

}

**3. Implement Concrete Repository:**

**CustomerRepositoryImpl Class**:

* Create a class CustomerRepositoryImpl that implements CustomerRepository. For simplicity, this implementation will use a hardcoded list of customers.

public class CustomerRepositoryImpl implements CustomerRepository {

@Override

public Customer findCustomerById(String id) {

// Simulate finding a customer by ID

if ("123".equals(id)) {

return new Customer("123", "John Doe", "john.doe@example.com");

} else {

return null;

}

}

}

**4. Define Service Class:**

**CustomerService Class**:

* Create a class CustomerService that depends on CustomerRepository for its operations.

public class CustomerService {

private CustomerRepository customerRepository;

// Constructor injection

public CustomerService(CustomerRepository customerRepository) {

this.customerRepository = customerRepository;

}

public Customer getCustomerById(String id) {

return customerRepository.findCustomerById(id);

}

}

**5. Implement Dependency Injection:**

* **Constructor Injection**: We’ll use constructor injection to provide the CustomerRepository dependency to CustomerService.

**6. Test the Dependency Injection Implementation:**

**TestDependencyInjection Class**:

* Create a main class to demonstrate creating a CustomerService with CustomerRepositoryImpl and using it to find a customer.

public class TestDependencyInjection {

public static void main(String[] args) {

// Create the repository

CustomerRepository repository = new CustomerRepositoryImpl();

// Inject the repository into the service

CustomerService service = new CustomerService(repository);

// Use the service to find a customer

Customer customer = service.getCustomerById("123");

// Display customer details

if (customer != null) {

System.out.println("Customer ID: " + customer.getId());

System.out.println("Customer Name: " + customer.getName());

System.out.println("Customer Email: " + customer.getEmail());

} else {

System.out.println("Customer not found.");

}

}

}

**Customer Class**:

* Define a simple Customer class used in the repository and service.

public class Customer {

private String id;

private String name;

private String email;

// Constructor

public Customer(String id, String name, String email) {

this.id = id;

this.name = name;

this.email = email;

}

// Getters

public String getId() {

return id;

}

public String getName() {

return name;

}

public String getEmail() {

return email;

}

}