**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:**
   * Create a new Java project named **DependencyInjectionExample**.
2. **Define Repository Interface:**
   * Create an interface **CustomerRepository** with methods like **findCustomerById()**.
3. **Implement Concrete Repository:**
   * Create a class **CustomerRepositoryImpl** that implements **CustomerRepository**.
4. **Define Service Class:**
   * Create a class **CustomerService** that depends on **CustomerRepository**.
5. **Implement Dependency Injection:**
   * Use constructor injection to inject **CustomerRepository** into **CustomerService**.
6. **Test the Dependency Injection Implementation:**
   * Create a main class to demonstrate creating a **CustomerService** with **CustomerRepositoryImpl** and using it to find a customer.

Solution:

1st file: CustomerRepository.java

public interface CustomerRepository {

String findCustomerById(String customerId);

}

2nd file: CustomerRepositoryImpl.java

public class CustomerRepositoryImpl implements CustomerRepository {

@Override

public String findCustomerById(String customerId) {

return "Customer[id=" + customerId + ", name=Alice]";

}

}

3rd file: CustomerService.java

public class CustomerService {

private CustomerRepository repository;

public CustomerService(CustomerRepository repository) {

this.repository = repository;

}

public void getCustomerDetails(String customerId) {

String customer = repository.findCustomerById(customerId);

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

}

}

4th file: DependencyInjectionExample.java

public class DependencyInjectionExample {

public static void main(String[] args) {

CustomerRepository repository = new CustomerRepositoryImpl();

CustomerService service = new CustomerService(repository);

service.getCustomerDetails("C101");

}

}

Output:

