

## Praktikum 7 - Layered Architecture Pattern

Nama : Cania Nabilatul Adawah

NIM : 2403102

Kelas : D3TI2C

Mata Kuliah : Pemrograman Berbasis Objek

1. Layered Architecture Pattern adalah pola arsitektur perangkat lunak yang memisahkan sistem menjadi beberapa lapisan logis (layers) dengan prinsip Separation of Concerns (pemisahan tanggung jawab).

Tujuannya: meningkatkan keteraturan, kemudahan pemeliharaan, dan skalabilitas.

Empat lapisan utama:

### 1. Presentation Layer (UI Layer)

- Menampilkan data ke pengguna.
- Menerima input dan melakukan validasi.
- Pola umum: MVC Pattern, MVP, MVVM.

### 2. Application Layer (Business Logic Layer)

- Menangani aturan bisnis dan alur kerja.
- Pola umum: Service Layer Pattern, Domain Logic Pattern.

### 3. Data Access Layer (Persistence Layer)

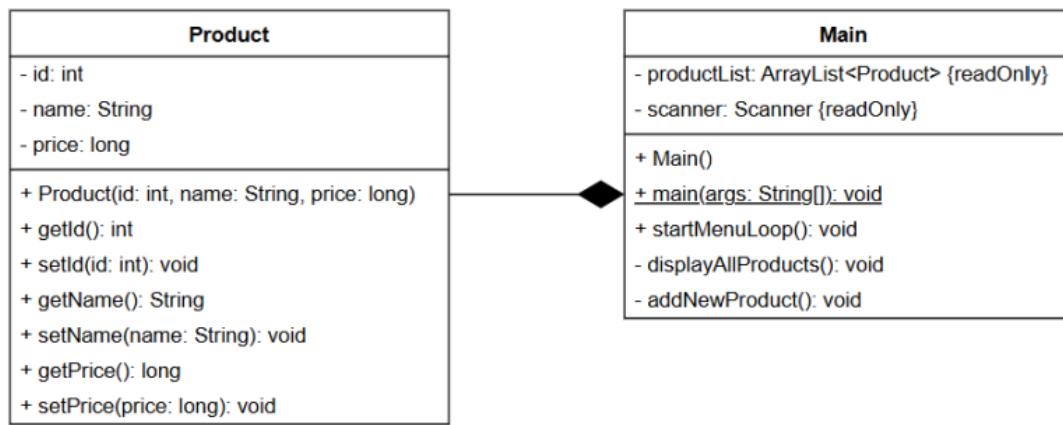
- Mengatur komunikasi dengan penyimpanan data.
- Pola umum: DAO Pattern, Repository Pattern, Data Mapper Pattern.

### 4. Data Source Layer

- Lapisan fisik penyimpanan data (database, file system, cloud storage).
- Tidak berinteraksi langsung dengan UI, hanya melalui DAO.

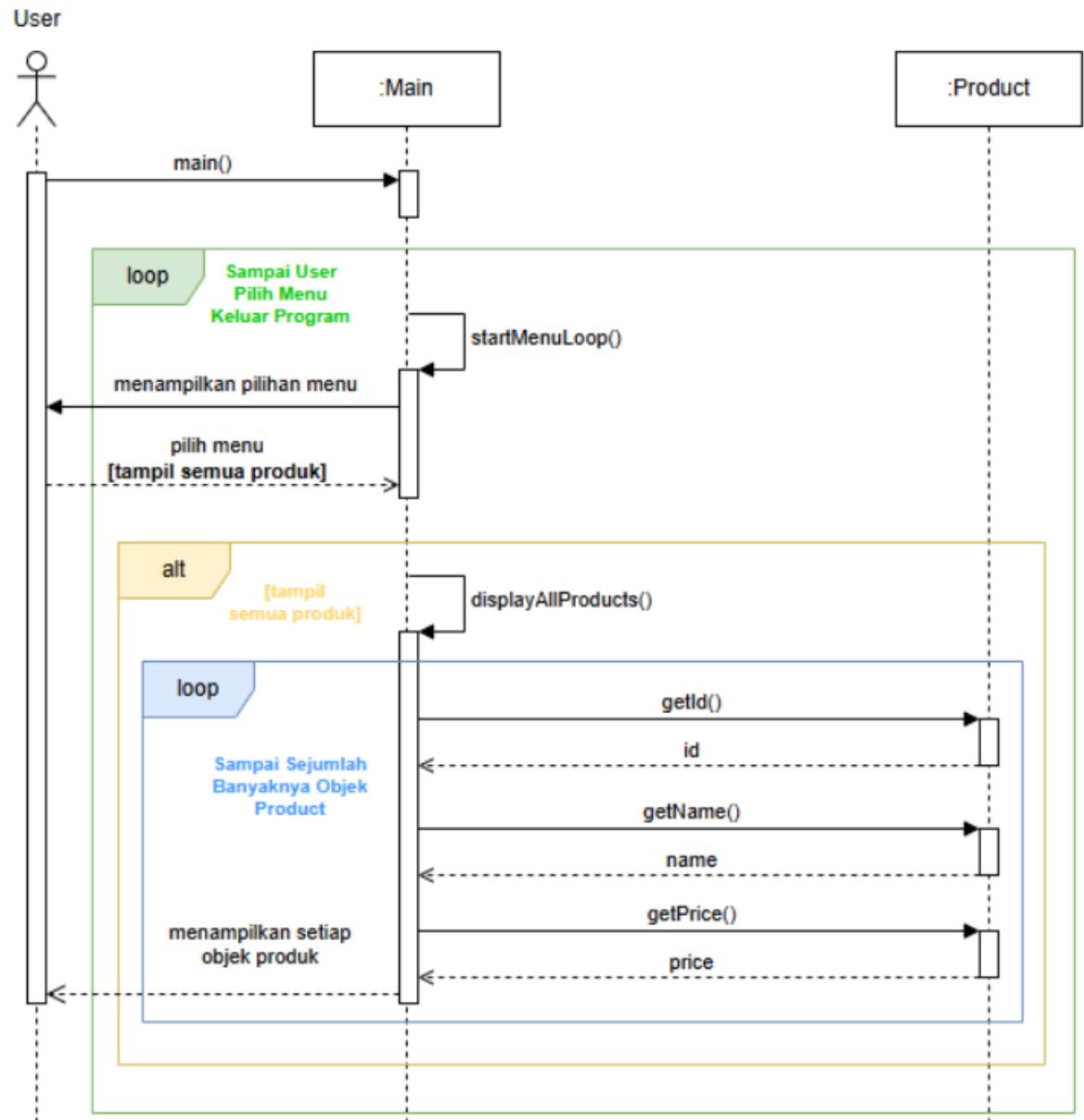
## 2. Tanpa Pattern

### a) Class Diagram

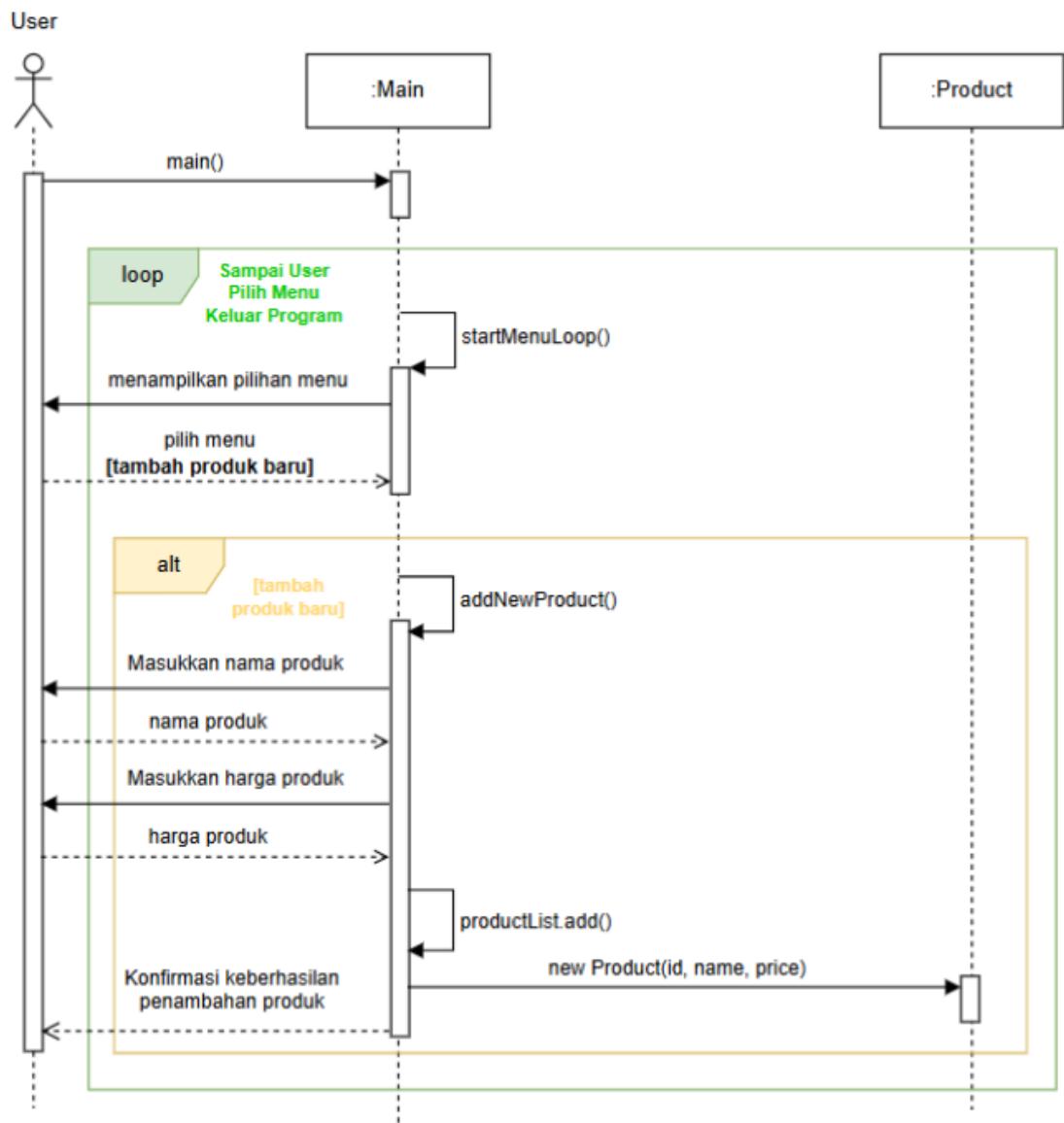


b) Sequence Diagram

- 1) Menampilkan semua daftar produk



2) Menambah produk baru



- c) Kode Program  
 1) Product.java

LayeredArchitecturePattern > TanpaPattern > model > **J** Product.java > Product > getId()

```
1 package model;
2
3 public class Product {
4     private int id;
5     private String name;
6     private long price;
7
8     public Product(int id, String name, long price) {
9         this.id = id;
10        this.name = name;
11        this.price = price;
12    }
13
14    public int getId() {
15        return id;
16    }
17
18    public void setId(int id) {
19        this.id = id;
20    }
21
22    public String getName() {
23        return name;
24    }
25
26    public void setName(String name) {
27        this.name = name;
28    }
```

```
26 ˇ     public void setName(String name) {  
27      |       this.name = name;  
28    }  
29  
30 ˇ     public long getPrice() {  
31      |       return price;  
32    }  
33 ˇ     public void setPrice(long price) {  
34      |       this.price = price;  
35    }  
36  
37  }  
38
```

2) Main.java

```
LayeredArchitecturePattern > TanpaPattern > J Main.java > Main
1 import java.util.ArrayList;
2 import java.util.Scanner;
3
4 import model.Product;
5
6 public class Main {
7
8     private final ArrayList<Product> productList = new ArrayList<>();
9     private final Scanner scanner = new Scanner(System.in);
10
11    public Main() {
12        productList.add(new Product(id: 1, name: "Laptop ASUS", price: 9500000));
13        productList.add(new Product(id: 2, name: "Monitor Dell", price: 2500000));
14    }
15
16    public static void main(String[] args) {
17        Main app = new Main();
18        app.startMenuLoop();
19        app.scanner.close();
20    }
21
22
23    public void startMenuLoop() {
24        boolean running = true;
25        while (running) {
26            System.out.println(x: "\n--- APLIKASI TANPA PATTERN ---");
27            System.out.println(x: "1. Tampilkan Semua Produk");
28            System.out.println(x: "2. Tambah Produk Baru");
```

```
28 |         System.out.println(x: "2. Tambah Produk Baru");
29 |         System.out.println(x: "3. Keluar");
30 |         System.out.print(s: "Pilih opsi: ");
31 |
32 |     try {
33 |         int choice = Integer.parseInt(scanner.nextLine());
34 |         switch (choice) {
35 |             case 1:
36 |                 displayAllProducts();
37 |                 break;
38 |             case 2:
39 |                 addNewProduct();
40 |                 break;
41 |             case 3:
42 |                 running = false;
43 |                 System.out.println(x: "Terima kasih! Program selesai.");
44 |                 break;
45 |             default:
46 |                 System.out.println(x: "Opsi tidak valid.");
47 |             }
48 |         } catch (NumberFormatException e) {
49 |             System.out.println(x: "Input tidak valid. Masukkan angka.");
50 |         }
51 |     }
52 | }
```

d) Bukti Running

```

PS C:\Users\user\OneDrive\Documents\projectpbo\PraktikumPBO\tugaspraktikum\praktikum7> & 'C:\Program Files\Java\jdk-21\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\user\AppData\Roaming\Code\User\workspaceStorage\b32c409f27f0f6b8f20f47117824d4ae\redhat.java\jdt_ws\praktikum7_a47e36f4\bin' 'Main'

--- APLIKASI TANPA PATTERN ---
1. Tampilkan Semua Produk
2. Tambah Produk Baru
3. Keluar
Pilih opsi: 1

--- Daftar Produk ---
1 - Laptop ASUS | Rp 9500000
2 - Monitor Dell | Rp 2500000

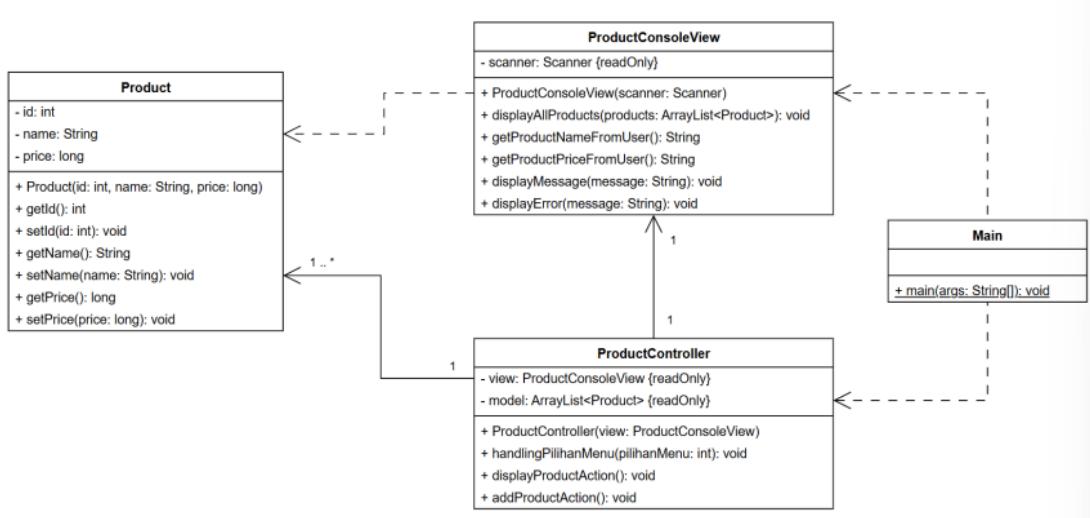
--- APLIKASI TANPA PATTERN ---
1. Tampilkan Semua Produk
2. Tambah Produk Baru
3. Keluar
Pilih opsi: 2
Masukkan Nama Produk: Laptop Lenovo
Masukkan Harga Produk: 4000000
Produk berhasil ditambahkan!

--- APLIKASI TANPA PATTERN ---
1. Tampilkan Semua Produk
2. Tambah Produk Baru
3. Keluar
Pilih opsi: 3
Terima kasih! Program selesai.
PS C:\Users\user\OneDrive\Documents\projectpbo\PraktikumPBO\tugaspraktikum\praktikum7>

```

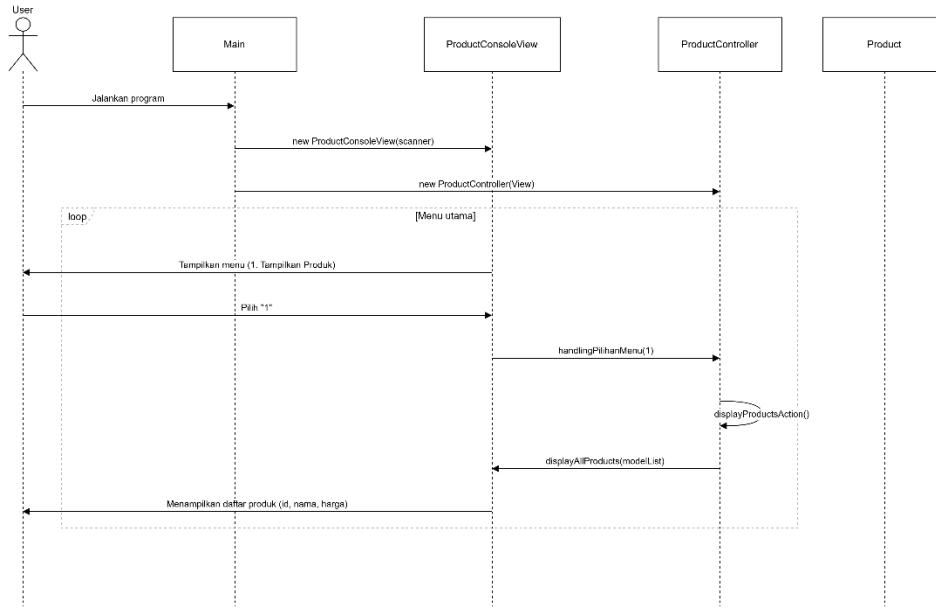
### 3. MVC Pattern

#### a) Class Diagram

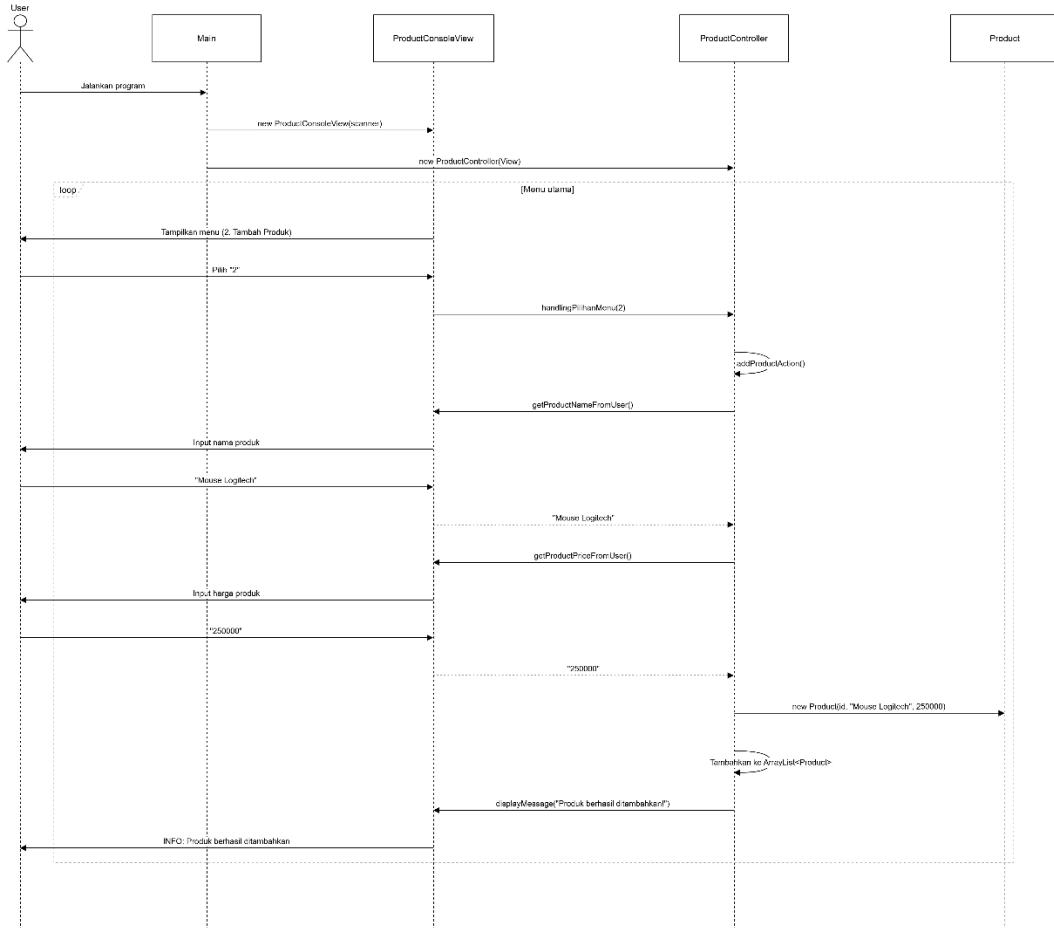


#### b) Sequence Diagram

- 1) Menampilkan semua daftar produk



## 2) Menambah produk baru



## c) Kode Program

### 1) ProductController.java

```
LayeredArchitecturePattern > MVC_Pattern > controller > ProductController.java > ProductController > handlingPilihanMenu(int)
1  package MVC_Pattern.controller;
2
3  import java.util.ArrayList;
4
5  import MVC_Pattern.model.Product;
6  import MVC_Pattern.view.ProductConsoleView;
7
8  public class ProductController {
9      private final ProductConsoleView view;
10     private final ArrayList<Product> model = new ArrayList<>();
11
12     public ProductController(ProductConsoleView view) {
13         this.view = view;
14         model.add(new Product(id: 1, name: "Laptop ASUS", price: 9500000));
15         model.add(new Product(id: 2, name: "Monitor Dell", price: 2500000));
16     }
17
18     public void handlingPilihanMenu(int pilihanMenu) {
19         switch (pilihanMenu) {
20             case 1:
21                 displayProductsAction();
22                 break;
23             case 2:
24                 addProductAction();
25                 break;
26             case 3:
27                 view.displayMessage(message: "Keluar dari aplikasi.");
28                 break;
29             default:
30                 view.displayError(message: "Opsi tidak valid.");
31         }
32     }
33
34     private void displayProductsAction() {
35         view.displayAllProducts(model);
36     }
37
38     private void addProductAction() {
39         String name = view.getProductFromUser();
40         String priceStr = view.getProductPriceFromUser();
41         try {
42             long price = Long.parseLong(priceStr);
43             if (price <= 0) {
44                 throw new IllegalArgumentException(s: "Harga harus angka positif lebih dari 0.");
45             }
46             int newId = model.size() + 1;
47             model.add(new Product(newId, name, price));
48             view.displayMessage(message: "Produk berhasil ditambahkan!");
49         } catch (IllegalArgumentException e) {
50             view.displayError("Gagal menambah produk: " + e.getMessage());
51         }
52     }
53 }
54
```

## 2) Product.java

LayeredArchitecturePattern > MVC\_Pattern > model >  Product.java >  Product

```
1 package MVC_Pattern.model;
2
3 public class Product {
4
5     private int id;
6     private String name;
7     private long price;
8
9     public Product(int id, String name, long price) {
10         this.id = id;
11         this.name = name;
12         this.price = price;
13     }
14
15     public int getId(){
16         return id;
17     }
18
19     public void setId(int id){
20         this.id = id;
21     }
22
23     public String getName(){
24         return name;
25     }
26
27     public void setName(String name){
28         this.name = name;
29     }
```

```
26
27     public void setName(String name){
28         this.name = name;
29     }
30
31     public long getPrice(){
32         return price;
33     }
34
35     public void setPrice(long price){
36         this.price = price;
37     }
38
39 }
40
41
```

3) ProductConsoleView.java

```
LayeredArchitecturePattern > MVC_Pattern > view > ProductConsoleView.java > ...
1  package MVC_Pattern.view;
2
3  import java.util.ArrayList;
4  import java.util.Scanner;
5  import MVC_Pattern.model.Product;
6
7  public class ProductConsoleView {
8      private final Scanner scanner;
9
10     public ProductConsoleView(Scanner scanner) {
11         this.scanner = scanner;
12     }
13
14     public void displayAllProducts(ArrayList<Product> products) {
15         System.out.println("DAFTAR PRODUK");
16         for (Product p : products) {
17             System.out.println(p.getId() + ". " + p.getName() + " - Rp" + p.getPrice());
18         }
19     }
20
21     public String getProductNameFromUser() {
22         System.out.print("Masukkan nama produk: ");
23         return scanner.nextLine();
24     }
25
26     public String getProductPriceFromUser() {
27         System.out.print("Masukkan harga produk: ");
28         return scanner.nextLine();
29     }
30
31     public void displayMessage(String message) {
32         System.out.println(message);
33     }
34
35     public void displayError(String message) {
36         System.err.println("Error: " + message);
37     }
38 }
39
```

4) Main.java

LayeredArchitecturePattern > MVC\_Pattern >  Main.java >  Main

```
1 package MVC_Pattern;
2
3 import MVC_Pattern.controller.ProductController;
4 import MVC_Pattern.view.ProductConsoleView;
5 import java.util.Scanner;
6
7 public class Main {
8     Run | Debug
9     public static void main(String[] args) {
10         Scanner scanner = new Scanner(System.in);
11         ProductConsoleView view = new ProductConsoleView(scanner);
12         ProductController controller = new ProductController(view);
13
14         while(true){
15             System.out.println(":\n--- APLIKASI MVC PATTERN ---");
16             System.out.println("1. Tampilkan Produk");
17             System.out.println("2. Tambah Produk");
18             System.out.println("3. Keluar");
19             System.out.print("Pilih: ");
20
21             try {
22                 int pilihan = Integer.parseInt(scanner.nextLine());
23                 if (pilihan == 3) break;
24                 controller.handlingPilihanMenu(pilihan);
25             } catch (NumberFormatException e) {
26                 view.displayError(message: "Input tidak valid. Masukkan angka.");
27             }
28         }
29     }
30 }
```

```

13     while(true){
14         System.out.println(x: "\n--- APLIKASI MVC PATTERN ---");
15         System.out.println(x: "1. Tampilkan Produk");
16         System.out.println(x: "2. Tambah Produk");
17         System.out.println(x: "3. Keluar");
18         System.out.print(s: "Pilih: ");
19
20         try {
21             int pilihan = Integer.parseInt(scanner.nextLine());
22             if (pilihan == 3) break;
23             controller.handlingPilihanMenu(pilihan);
24         } catch (NumberFormatException e) {
25             view.displayError(message: "Input tidak valid. Masukkan angka.");
26         }
27     }
28
29     scanner.close();
30 }
31
32

```

d) Bukti Running

The screenshot shows the Eclipse IDE interface with the terminal window active. The terminal output is as follows:

```

Microsoft Windows [Version 10.0.26100.6899]
(c) Microsoft Corporation. All rights reserved.

C:\Users\user\OneDrive\Documents\projectpbo\PraktikumPBO\tugaspraktikum\praktikum7> cmd /C ""C:\Program Files\Java\jdk-21\bin\java.exe" -XX:+ShowCodeDetailsInExceptionMessages -cp C:\Users\user\AppData\Roaming\Code\User\workspaceStorage\b32c409f27f0f6b8f20f47117824d4ae\redhat.java\jdt_ws\praktikum7_a47e36f4\bin MVC_Pattern.Main"

--- APLIKASI MVC PATTERN ---
1. Tampilkan Produk
2. Tambah Produk
3. Keluar
Pilih: 1

==== DAFTAR PRODUK ====
1. Laptop ASUS - Rp9500000
2. Monitor Dell - Rp2500000

--- APLIKASI MVC PATTERN ---
1. Tampilkan Produk
2. Tambah Produk
3. Keluar
Pilih: 2
Masukkan nama produk: Laptop Advan
Masukkan harga produk: 200000
Produk berhasil ditambahkan!

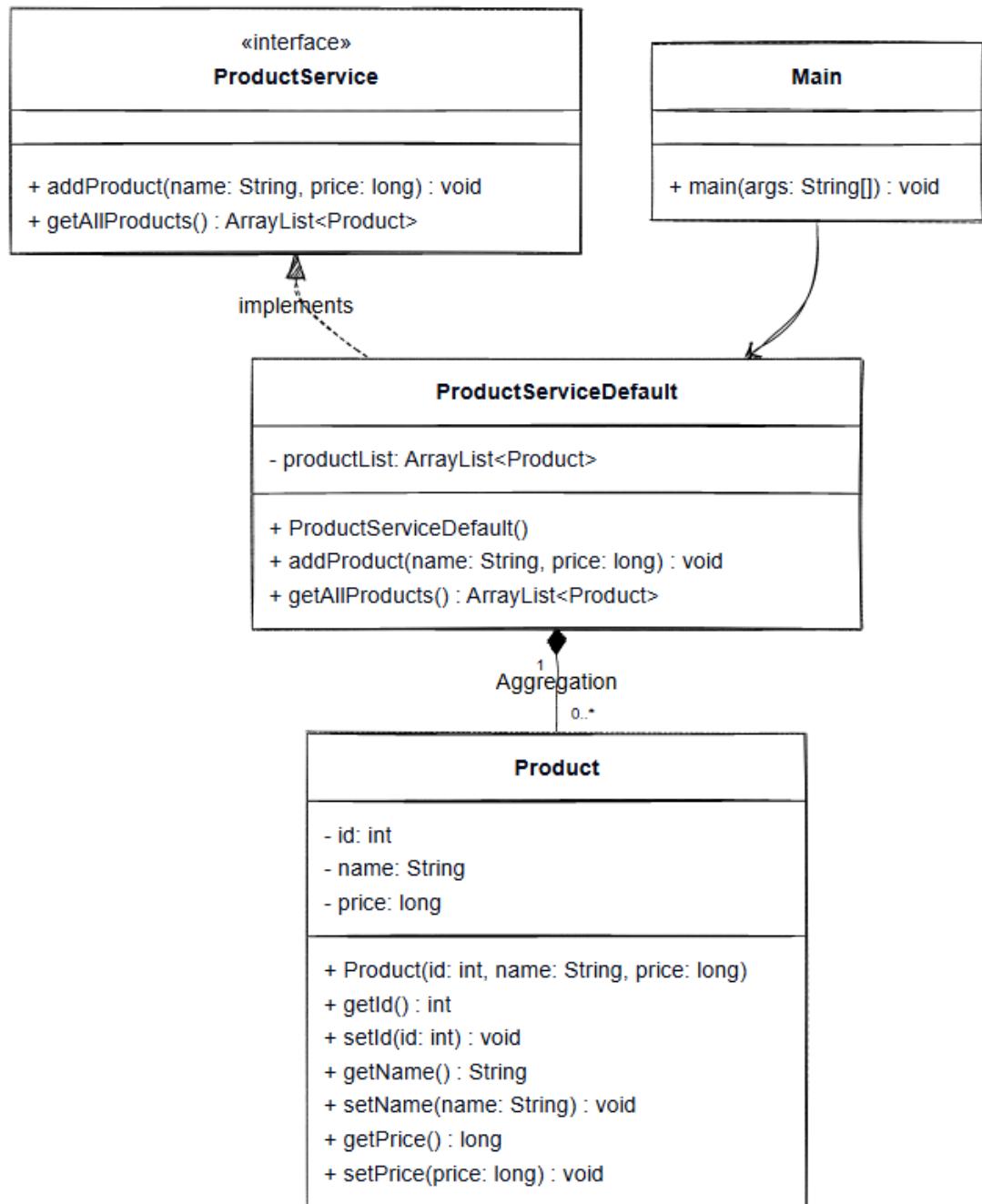
--- APLIKASI MVC PATTERN ---
1. Tampilkan Produk
2. Tambah Produk
3. Keluar
Pilih: 3

C:\Users\user\OneDrive\Documents\projectpbo\PraktikumPBO\tugaspraktikum\praktikum7>

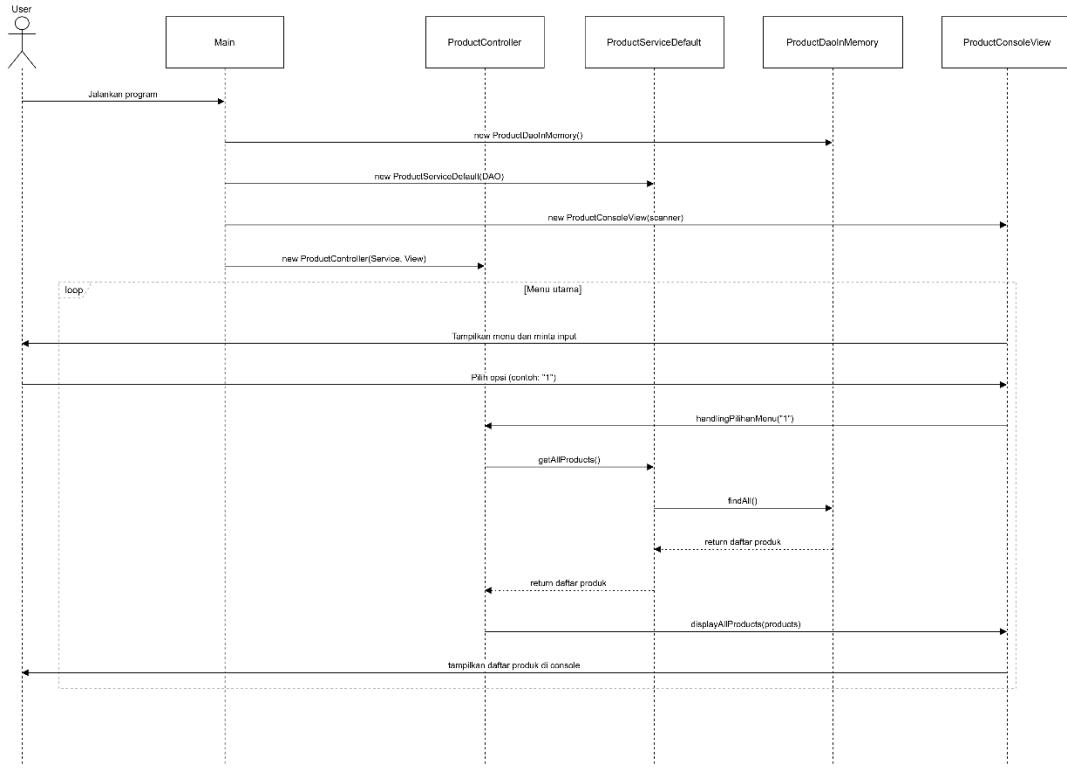
```

4. Service Layer Pattern

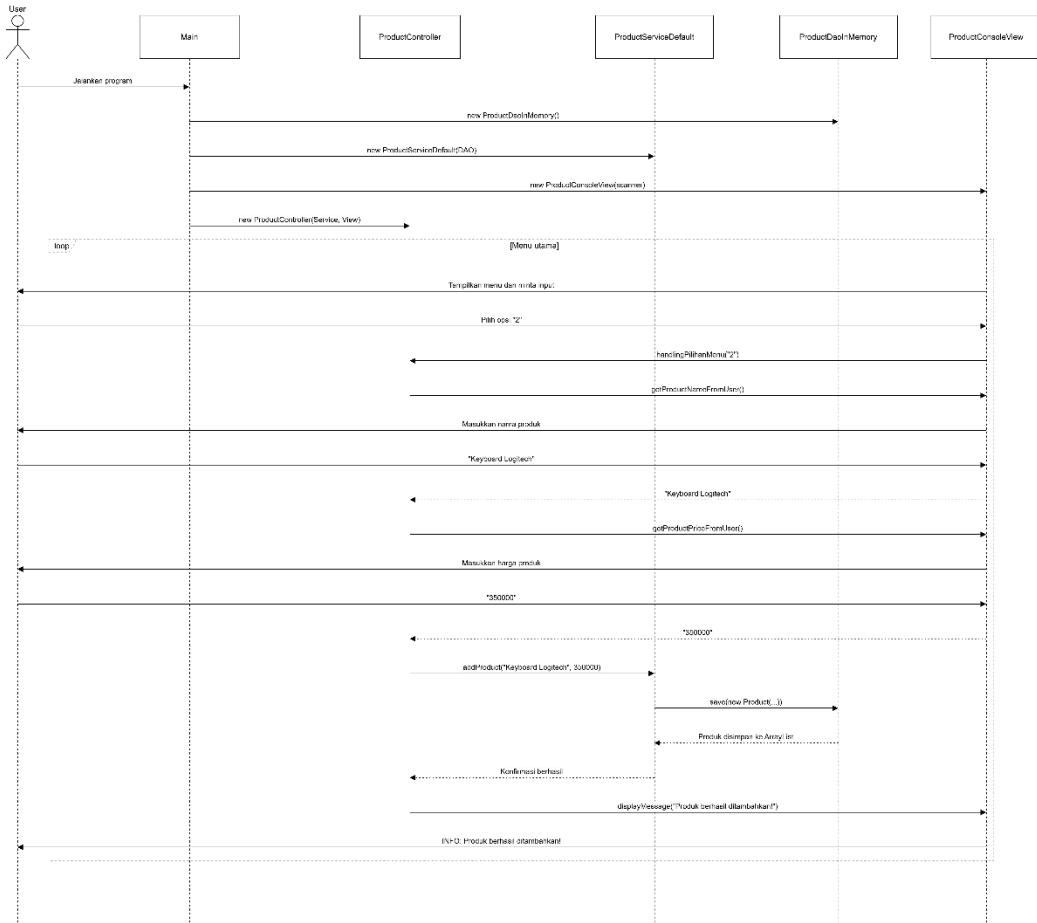
a) Class Diagram



- b) Sequence Diagram
- 1) Menampilkan semua daftar produk



2) Menambah produk baru



c) Kode Program  
1) Product.java

```
LayeredArchitecturePattern > ServiceLayer_Pattern > model > J Product.java > 📁 Product
1  package ServiceLayer_Pattern.model;
2
3  public class Product {
4      private int id;
5      private String name;
6      private long price;
7
8      public Product(int id, String name, long price) {
9          this.id = id;
10         this.name = name;
11         this.price = price;
12     }
13
14     public int getId() { return id; }
15     public String getName() { return name; }
16     public long getPrice() { return price; }
17 }
18
```

## 2) ProductService.java

```
LayeredArchitecturePattern > ServiceLayer_Pattern > service > J ProductService.java >
1  package ServiceLayer_Pattern.service;
2
3  import java.util.ArrayList;
4  import ServiceLayer_Pattern.model.Product;
5
6  public interface ProductService {
7      void addProduct(String name, long price);
8      ArrayList<Product> getAllProducts();
9  }
10
```

## 3) ProductServiceDefault.java

```
LayeredArchitecturePattern > ServiceLayer_Pattern > service > J ProductServiceDefault.java > ...
1 package ServiceLayer_Pattern.service;
2
3 import java.util.ArrayList;
4 import ServiceLayer_Pattern.model.Product;
5
6 public class ProductServiceDefault implements ProductService {
7
8     private final ArrayList<Product> productList = new ArrayList<>();
9
10    public ProductServiceDefault() {
11        productList.add(new Product(id: 1, name: "Laptop ASUS", price: 9500000));
12        productList.add(new Product(id: 2, name: "Monitor Dell", price: 2500000));
13    }
14
15    @Override
16    public void addProduct(String name, long price) {
17        if (price <= 0) {
18            throw new IllegalArgumentException(s: "Harga harus angka positif lebih dari 0.");
19        }
20        int newId = productList.size() + 1;
21        productList.add(new Product(newId, name, price));
22    }
23
24    @Override
25    public ArrayList<Product> getAllProducts() {
26        return productList;
27    }
28}
29
```

4) Main.java

```
LayeredArchitecturePattern > ServiceLayer_Pattern > J Main.java > ...
1  package ServiceLayer_Pattern;
2
3  import java.util.List;
4  import java.util.Scanner;
5
6  import ServiceLayer_Pattern.model.Product;
7  import ServiceLayer_Pattern.service.ProductServiceDefault;
8
9  public class Main {
10
11     private final ProductServiceDefault productService = new ProductServiceDefault();
12     private final Scanner scanner = new Scanner(System.in);
13
14     Run | Debug
15     public static void main(String[] args) {
16         Main app = new Main();
17         app.startMenuLoop();
18         app.scanner.close();
19     }
20
21     public void startMenuLoop() {
22         boolean running = true;
23         while (running) {
24             System.out.println(x: "\n--- APLIKASI SERVICE LAYER PATTERN ---");
25             System.out.println(x: "1. Tampilkan Semua Produk");
26             System.out.println(x: "2. Tambah Produk Baru");
27             System.out.println(x: "3. Keluar");
28             System.out.print(s: "Pilih opsi: ");
```

```
28     try {
29         int choice = Integer.parseInt(scanner.nextLine());
30         switch (choice) {
31             case 1:
32                 displayAllProducts();
33                 break;
34             case 2:
35                 addNewProduct();
36                 break;
37             case 3:
38                 running = false;
39                 break;
40             default:
41                 System.out.println("Opsi tidak valid.");
42             }
43         } catch (NumberFormatException e) {
44             System.out.println("Input tidak valid. Masukkan angka.");
45         }
46     }
47 }
48
49
50 private void displayAllProducts() {
51     System.out.println("\n--- Daftar Produk ---");
52     List<Product> products = productService.getAllProducts();
53     for (Product product : products) {
54         System.out.println(product.getId() + " - " + product.getName());
```

```

54     |         System.out.println(product.getId() + " - "
55     |             + product.getName() + " Rp. " + product.getPrice());
56     |
57 }
58
59 private void addNewProduct() {
60     System.out.print(s: "Masukkan Nama Produk: ");
61     String name = scanner.nextLine();
62     System.out.print(s: "Masukkan Harga Produk: ");
63     String priceString = scanner.nextLine();
64
65     try {
66         long price = Long.parseLong(priceString);
67         productService.addProduct(name, price);
68         System.out.println(x: "Produk berhasil ditambahkan!");
69     } catch (NumberFormatException e) {
70         System.out.println(x: "Error: Harga tidak valid.");
71     } catch (IllegalArgumentException e) {
72         System.out.println("Error: " + e.getMessage());
73     }
74 }
75
76

```

#### d) Bukti Running

The screenshot shows the Eclipse IDE interface with the 'PRAKTIKUM' project selected in the left sidebar. The terminal window displays the following output:

```

Microsoft Windows [Version 10.0.26100.6899]
(c) Microsoft Corporation. All rights reserved.

C:\Users\user\OneDrive\Documents\projectpbo\PraktikumPBO\tugaspraktikum\praktikum7> cmd /C ""C:\Program Files\Java\jdk-21\bin\java
a.exe" -XX:+ShowCodeDetailsInExceptionMessages -cp C:\Users\user\AppData\Roaming\Code\User\workspaceStorage\b32c409f27f0fb8f20f4
7117824d4ae\redhat.java\jdt_ws\praktikum7_a47e36f4\bin ServiceLayer_Pattern.Main"

--- APLIKASI SERVICE LAYER PATTERN ---
1. Tampilkan Semua Produk
2. Tambah Produk Baru
3. Keluar
Pilih opsi: 1

--- Daftar Produk ---
1 - Laptop ASUS Rp. 9500000
2 - Monitor Dell Rp. 2500000

--- APLIKASI SERVICE LAYER PATTERN ---
1. Tampilkan Semua Produk
2. Tambah Produk Baru
3. Keluar
Pilih opsi: 2
Masukkan Nama Produk: Laptop Acer
Masukkan Harga Produk: 3000000
Produk berhasil ditambahkan!

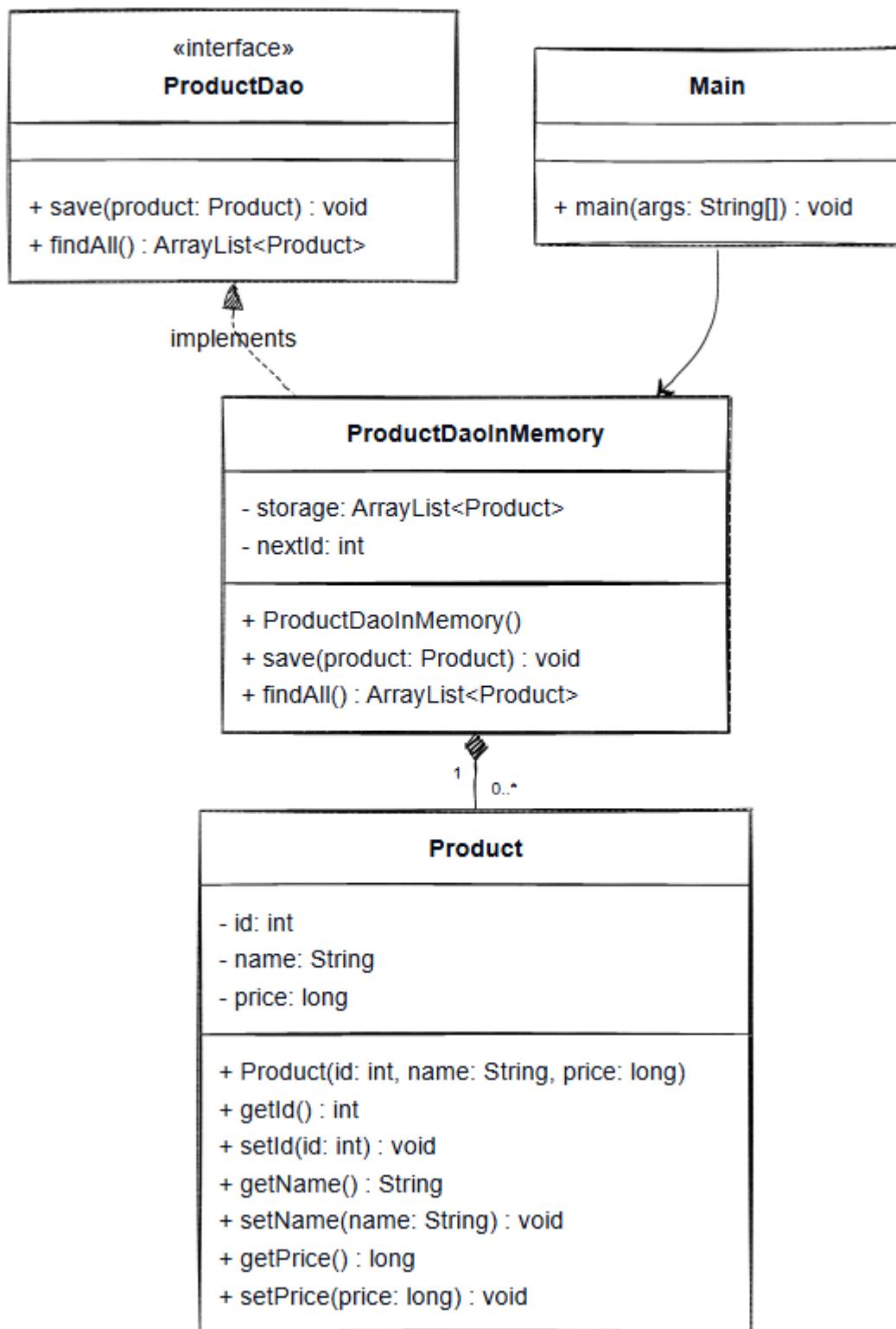
--- APLIKASI SERVICE LAYER PATTERN ---
1. Tampilkan Semua Produk
2. Tambah Produk Baru
3. Keluar
Pilih opsi: 3

C:\Users\user\OneDrive\Documents\projectpbo\PraktikumPBO\tugaspraktikum\praktikum7>

```

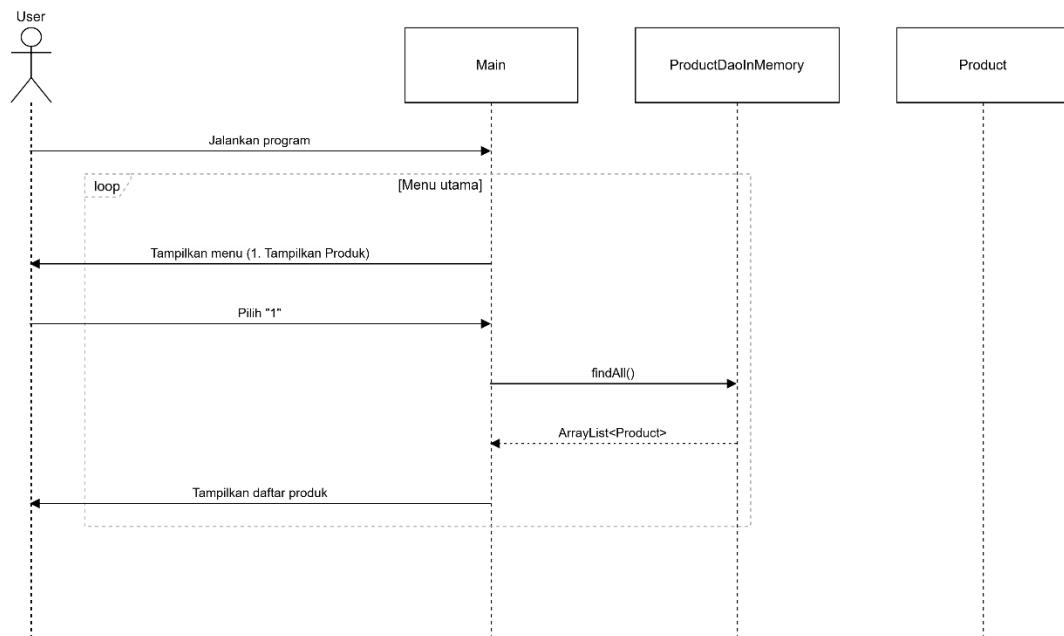
#### 5. DAO Pattern

##### a) Class Diagram

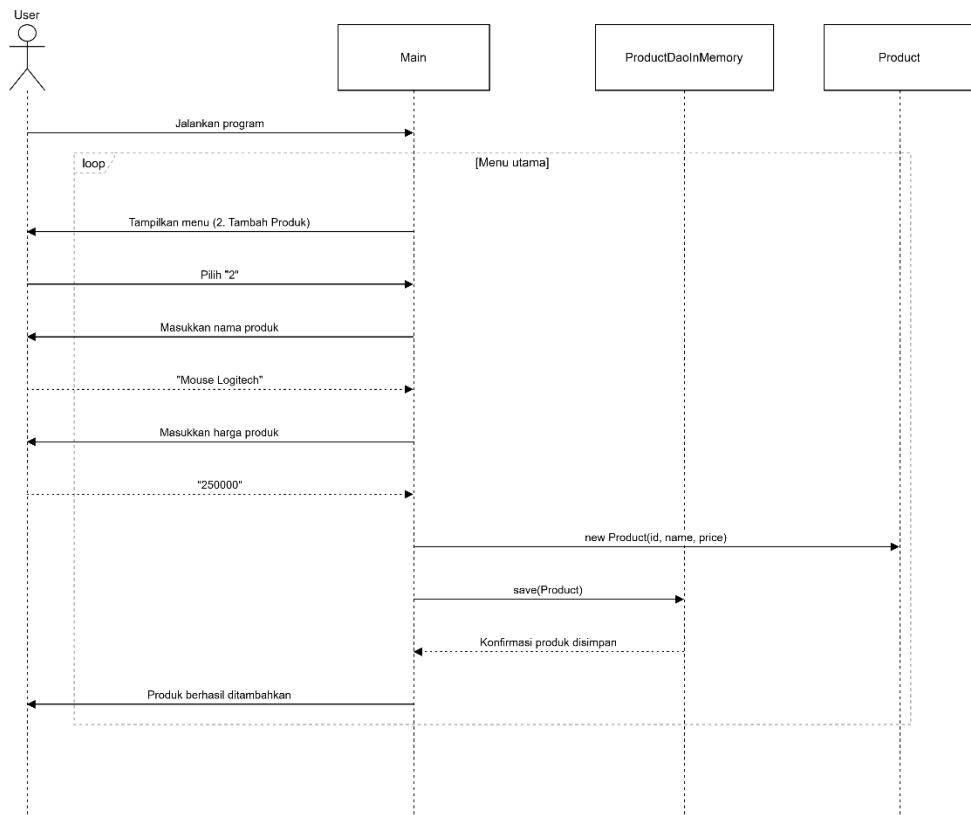


b) Sequence Diagram

### 1) Menampilkan semua daftar produk



### 2) Menambah produk baru



### c) Kode Program

#### 1) ProductDaoInMemory.java

```
LayeredArchitecturePattern > DAO_Pattern > dao > memory > J ProductDaoInMemory.java > ...
1 package dao.memory;
2
3 import java.util.ArrayList;
4 import dao.ProductDao;
5 import model.Product;
6
7 public class ProductDaoInMemory implements ProductDao {
8
9     private final ArrayList<Product> storage = new ArrayList<>();
10
11    public ProductDaoInMemory() {
12        // Data awal (contoh)
13        storage.add(new Product(id: 1, name: "Laptop ASUS", price: 9500000));
14        storage.add(new Product(id: 2, name: "Monitor Dell", price: 2500000));
15    }
16
17    @Override
18    public void save(Product product) {
19        storage.add(product);
20    }
21
22    @Override
23    public ArrayList<Product> findAll() {
24        return storage;
25    }
26}
27
```

## 2) ProductDao.java

```
LayeredArchitecturePattern > DAO_Pattern > dao > J ProductDao.java > ...
1 package dao;
2
3 import java.util.ArrayList;
4 import model.Product;
5
6 public interface ProductDao {
7     void save(Product product);
8     ArrayList<Product> findAll();
9 }
10
```

3) Product.java

```
LayeredArchitecturePattern > DAO_Pattern > model > J Product.java > ...
1  package model;
2
3  public class Product {
4      private int id;
5      private String name;
6      private long price;
7
8      public Product(int id, String name, long price){
9          this.id = id;
10         this.name = name;
11         this.price = price;
12     }
13
14     public int getId() { return id; }
15     public void setId(int id){ this.id = id; }
16
17     public String getName() { return name; }
18     public void setName(String name) { this.name = name; }
19
20     public long getPrice() { return price; }
21     public void setPrice(long price) { this.price = price; }
22
23     @Override
24     public String toString() {
25         return id + " - " + name + " Rp. " + price;
26     }
27 }
28 }
```

4) Main.java

```
LayeredArchitecturePattern > DAO_Pattern > J Main.java > ...
1 import java.util.ArrayList;
2 import java.util.Scanner;
3
4 import dao.ProductDao;
5 import dao.memory.ProductDaoInMemory;
6 import model.Product;
7
8 public class Main {
9     private final ProductDao productDao = new ProductDaoInMemory();
10    private final Scanner scanner = new Scanner(System.in);
11
12    Run | Debug
13    public static void main(String[] args) {
14        Main app = new Main();
15        app.startMenuLoop();
16        app.scanner.close();
17    }
18
19    public void startMenuLoop() {
20        boolean running = true;
21        while (running) {
22            System.out.println(x: "\n--- APLIKASI DAO PATTERN ---");
23            System.out.println(x: "1. Tampilkan Semua Produk");
24            System.out.println(x: "2. Tambah Produk Baru");
25            System.out.println(x: "3. Keluar");
26            System.out.print(s: "Pilih opsi: ");
27            try {
28                int choice = Integer.parseInt(scanner.nextLine());
29                switch (choice) {
```

```
28     switch (choice) {
29         case 1:
30             displayProducts();
31             break;
32         case 2:
33             addNewProduct();
34             break;
35         case 3:
36             running = false;
37             System.out.println(x: "Keluar dari aplikasi...");
38             break;
39         default:
40             System.out.println(x: "Opsi tidak valid.");
41     }
42 } catch (NumberFormatException e) {
43     System.out.println(x: "Input tidak valid. Masukkan angka 1-3.");
44 }
45 }
46 }
47
48 private void displayProducts() {
49     System.out.println(x: "\n--- Daftar Produk ---");
50     ArrayList<Product> products = productDao.findAll();
51     if (products.isEmpty()) {
52         System.out.println(x: "Belum ada produk yang tersimpan.");
53     } else {
54         for (Product product : products) {
```

```

55     |           System.out.println(product);
56     |
57     }
58 }
59
60 private void addNewProduct() {
61     System.out.print("Nama Produk: ");
62     String name = scanner.nextLine();
63
64     System.out.print("Harga Produk: ");
65     try {
66         long price = Long.parseLong(scanner.nextLine());
67         if (price <= 0) {
68             System.out.println("Error: Harga harus lebih besar dari 0.");
69             return;
70         }
71         int newId = productDao.findAll().size() + 1;
72         Product新产品 = new Product(newId, name, price);
73         productDao.save(新产品);
74         System.out.println("Produk berhasil ditambahkan!");
75     } catch (NumberFormatException e) {
76         System.out.println("Input harga tidak valid. Harus berupa angka.");
77     }
78 }
79
80

```

d) Bukti Running

```

--- APLIKASI DAO PATTERN ---
1. Tampilkan Semua Produk
2. Tambah Produk Baru
3. Keluar
Pilih opsi: 1

--- Daftar Produk ---
1 - Laptop ASUS Rp. 9500000
2 - Monitor Dell Rp. 2500000

--- APLIKASI DAO PATTERN ---
1. Tampilkan Semua Produk
2. Tambah Produk Baru
3. Keluar
Pilih opsi: 2
Nama Produk: Laptop Msi
Harga Produk: 4000000
Produk berhasil ditambahkan!

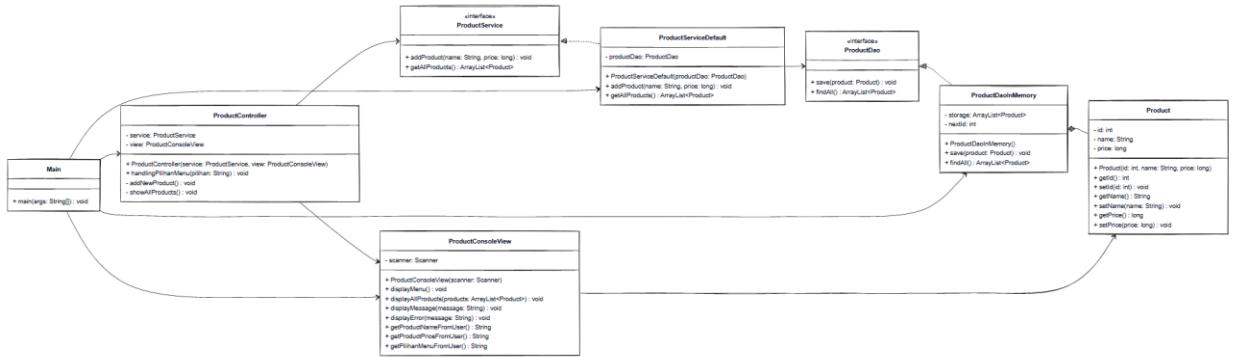
--- APLIKASI DAO PATTERN ---
1. Tampilkan Semua Produk
2. Tambah Produk Baru
3. Keluar
Pilih opsi: 3
Keluar dari aplikasi...

```

C:\Users\user\OneDrive\Documents\projectpbo\PraktikumPBO\tugaspraktikum\praktikum7>

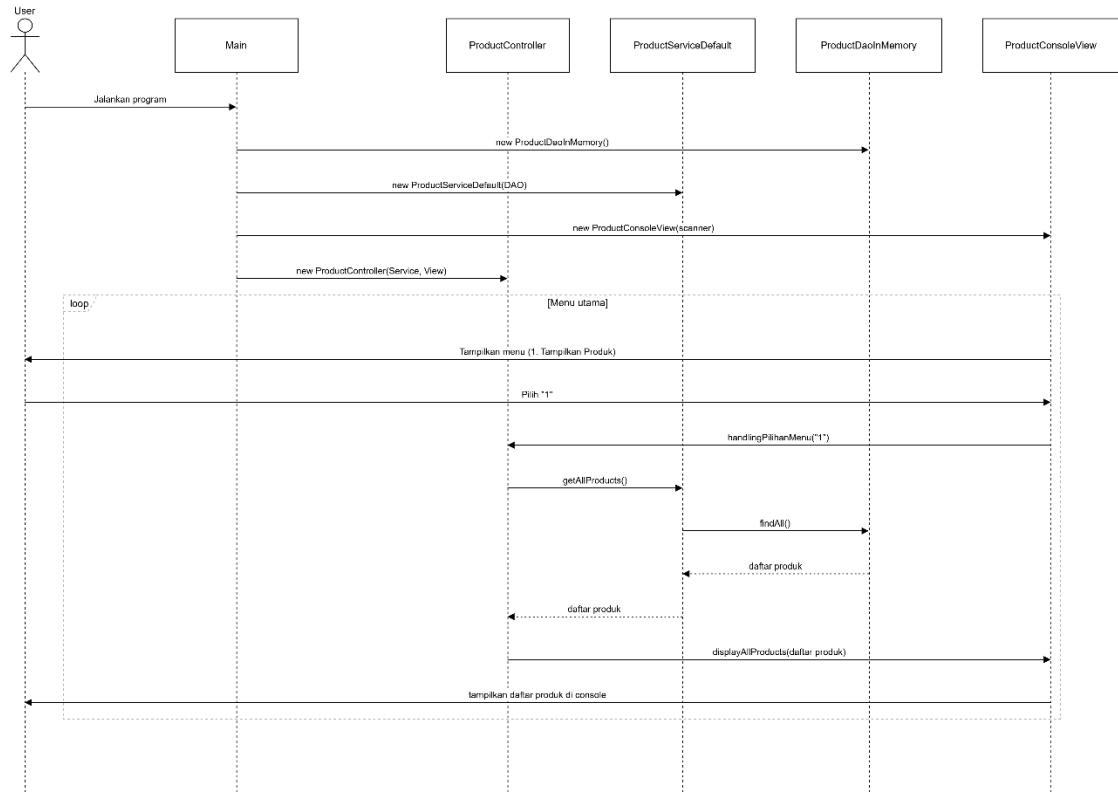
## 6. MVC + Service Layer + DAO Pattern

### a) Class Diagram

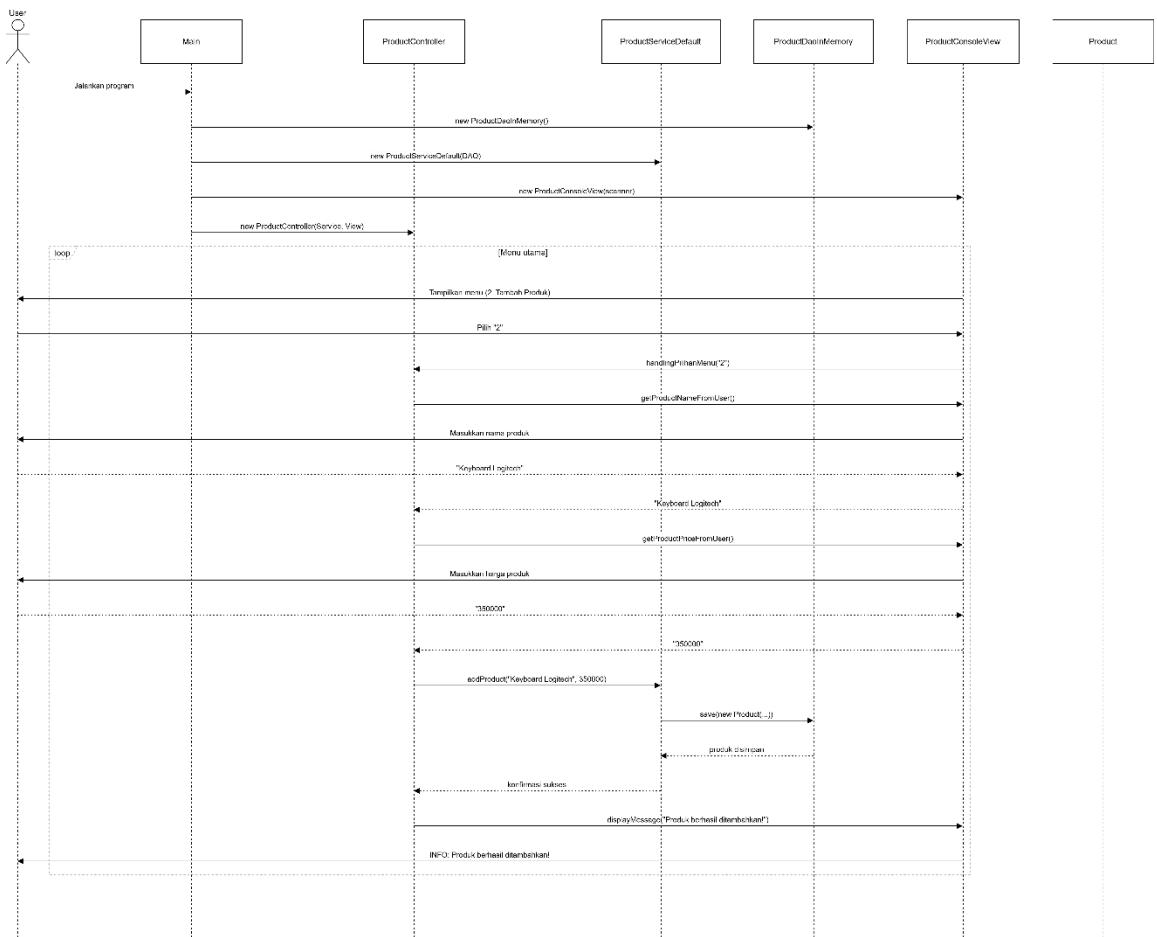


### b) Sequence Diagram

1) Menampilkan semua daftar produk



2) Menambah produk baru



### c) Kode Program

#### 1) ProductController.java

LayeredArchitecturePattern > MVC\_ServiceLayer.DAO > controller > **J** ProductController.java > ProductController

```
1 package MVC_ServiceLayer.DAO.controller;
2
3 import MVC_ServiceLayer.DAO.service.ProductService;
4 import MVC_ServiceLayer.DAO.view.ProductConsoleView;
5
6 public class ProductController {
7
8     private final ProductService service;
9     private final ProductConsoleView view;
10
11    public ProductController(ProductService service, ProductConsoleView view) {
12        this.service = service;
13        this.view = view;
14    }
15
16    public void handlingPilihanMenu(String pilihanMenu) {
17        try {
18            int menu = Integer.parseInt(pilihanMenu);
19            switch (menu) {
20                case 1:
21                    showAllProducts();
22                    break;
23                case 2:
24                    addNewProduct();
25                    break;
26                case 3:
27                    view.displayMessage(message: "Aplikasi ditutup.");
28                    System.exit(status: 0);
29                    break;
30            }
31        } catch (Exception e) {
32            view.displayMessage(message: "Terjadi kesalahan pada sistem.");
33        }
34    }
35}
```

```
29         break;
30     default:
31         view.displayError(message: "Opsi tidak valid.");
32         break;
33     }
34 } catch (NumberFormatException e) {
35     view.displayError(message: "Input tidak valid. Masukkan angka.");
36 }
37 }
38
39 private void addNewProduct() {
40     String name = view.getProductFromUser();
41     String priceStr = view.getProductPriceFromUser();
42     try {
43         long price = Long.parseLong(priceStr);
44         service.addProduct(name, price);
45         view.displayMessage(message: "Produk berhasil ditambahkan!");
46     } catch (Exception e) {
47         view.displayError("Gagal menambah produk: " + e.getMessage());
48     }
49 }
50
51 private void showAllProducts() {
52     view.displayAllProducts(service.getAllProducts());
53 }
54 }
55 }
```

2) ProductDaoInMemory.java

```
LayeredArchitecturePattern > MVC_ServiceLayer.DAO > dao > memory > J ProductDaoInMemory.java > P  
1 package MVC_ServiceLayer.DAO.dao.memory;  
2  
3 import java.util.ArrayList;  
4 import MVC_ServiceLayer.DAO.dao.ProductDao;  
5 import MVC_ServiceLayer.DAO.model.Product;  
6  
7 public class ProductDaoInMemory implements ProductDao {  
8  
9     private final ArrayList<Product> storage = new ArrayList<>();  
10    private int nextId = 1;  
11  
12    public ProductDaoInMemory() {  
13        storage.add(new Product(id: 1, name: "Laptop ASUS", price: 9500000));  
14        storage.add(new Product(id: 2, name: "Monitor Dell", price: 2500000));  
15        nextId = 3;  
16    }  
17  
18    @Override  
19    public void save(Product p) {  
20        if (p.getId() == 0) {  
21            p.setId(nextId++);  
22        }  
23        storage.add(p);  
24    }  
25  
26    @Override  
27    public ArrayList<Product> findAll() {  
28        return storage;  
29    }  
30  
31
```

3) ProductDao.java

```
LayeredArchitecturePattern > MVC_ServiceLayer.DAO > dao > J ProductDao.java > ...
1 package MVC_ServiceLayer.DAO.dao;
2
3 import java.util.ArrayList;
4 import MVC_ServiceLayer.DAO.model.Product;
5
6 public interface ProductDao {
7     void save(Product product);
8     ArrayList<Product> findAll();
9 }
10
```

4) Product.java

```
LayeredArchitecturePattern > MVC_ServiceLayer.DAO > model > J Product.java > ...
1 package MVC_ServiceLayer.DAO.model;
2
3 public class Product {
4
5     private int id;
6     private String name;
7     private long price;
8
9     public Product(int id, String name, long price) {
10         this.id = id;
11         this.name = name;
12         this.price = price;
13     }
14
15     public int getId() { return id; }
16     public void setId(int id) { this.id = id; }
17
18     public String getName() { return name; }
19     public void setName(String name) { this.name = name; }
20
21     public long getPrice() { return price; }
22     public void setPrice(long price) { this.price = price; }
23 }
24
```

5) ProductService.java

```
LayeredArchitecturePattern > MVC_ServiceLayer.DAO > service > J ProductService.java > ...
1 package MVC_ServiceLayer.DAO.service;
2
3 import java.util.ArrayList;
4 import MVC_ServiceLayer.DAO.model.Product;
5
6 public interface ProductService {
7     void addProduct(String name, long price);
8     ArrayList<Product> getAllProducts();
9 }
10
```

6) ProductServiceDefault.java

```
LayeredArchitecturePattern > MVC_ServiceLayer.DAO > service > J ProductServiceDefault.java > ...
1 package MVC_ServiceLayer.DAO.service;
2
3 import java.util.ArrayList;
4 import MVC_ServiceLayer.DAO.dao.ProductDao;
5 import MVC_ServiceLayer.DAO.model.Product;
6
7 public class ProductServiceDefault implements ProductService {
8
9     private final ProductDao productDao;
10
11     public ProductServiceDefault(ProductDao productDao) {
12         this.productDao = productDao;
13     }
14
15     @Override
16     public void addProduct(String name, long price) {
17         if (name == null || name.trim().isEmpty()) {
18             throw new IllegalArgumentException(s: "Nama tidak boleh kosong.");
19         }
20         if (price <= 0) {
21             throw new IllegalArgumentException(s: "Harga harus > 0");
22         }
23         productDao.save(new Product(id: 0, name, price));
24     }
25
26     @Override
27     public ArrayList<Product> getAllProducts() {
28         return productDao.findAll();
29     }
30 }
```

```
24     }
25
26     @Override
27     public ArrayList<Product> getAllProducts() {
28         return productDao.findAll();
29     }
30 }
31
```

## 7) ProductConsoleView.java

```
LayeredArchitecturePattern > MVC_ServiceLayer.DAO > view > ProductConsoleView.java > ...
1 package MVC_ServiceLayer.DAO.view;
2
3 import java.util.ArrayList;
4 import java.util.Scanner;
5 import MVC_ServiceLayer.DAO.model.Product;
6
7 public class ProductConsoleView {
8
9     private final Scanner scanner;
10
11    public ProductConsoleView(Scanner scanner) {
12        this.scanner = scanner;
13    }
14
15    public void displayMenu() {
16        System.out.println(" --- APLIKASI MVC + SERVICE LAYER + DAO PATTERN ---");
17        System.out.println(" 1. Tampilkan Semua Produk");
18        System.out.println(" 2. Tambah Produk Baru");
19        System.out.println(" 3. Keluar");
20        System.out.print("Pilih opsi: ");
21    }
22
23    public void displayAllProducts(ArrayList<Product> products) {
24        System.out.println(" --- Daftar Produk ---");
25        if (products.isEmpty()) {
26            System.out.println("Tidak ada produk tersedia.");
27        } else {
28            for (Product product : products) {
29                System.out.println(product.getId() + " - " + product.getName())
30            }
31        }
32    }
33}
```

```
30         |         |         |         |         + " Rp. " + product.getPrice());
31     }
32 }
33 }
34
35 public void displayMessage(String message) {
36     System.out.println("INFO: " + message);
37 }
38
39 public void displayError(String message) {
40     System.out.println("ERROR: " + message);
41 }
42
43 public String getProductNameFromUser() {
44     System.out.print(s: "Masukkan Nama Produk: ");
45     return scanner.nextLine();
46 }
47
48 public String getProductPriceFromUser() {
49     System.out.print(s: "Masukkan Harga Produk: ");
50     return scanner.nextLine();
51 }
52
53 public String getPilihanMenuFromUser() {
54     return scanner.nextLine();
55 }
56 }
```

8) Main.java

```
LayeredArchitecturePattern > MVC_ServiceLayer.DAO > J Main.java > ...
1 package MVC_ServiceLayer.DAO;
2
3 import java.util.Scanner;
4
5 import MVC_ServiceLayer.DAO.controller.ProductController;
6 import MVC_ServiceLayer.DAO.dao.ProductDao;
7 import MVC_ServiceLayer.DAO.dao.memory.ProductDaoInMemory;
8 import MVC_ServiceLayer.DAO.service.ProductService;
9 import MVC_ServiceLayer.DAO.service.ProductServiceDefault;
10 import MVC_ServiceLayer.DAO.view.ProductConsoleView;
11
12 public class Main {
    Run | Debug
13     public static void main(String[] args) {
14
15         // Layer DAO
16         ProductDao dao = new ProductDaoInMemory();
17
18         // Layer Service
19         ProductService service = new ProductServiceDefault(dao);
20
21         // Layer View
22         Scanner scanner = new Scanner(System.in);
23         ProductConsoleView view = new ProductConsoleView(scanner);
24
25         // Layer Controller
26         ProductController controller = new ProductController(service, view);
27
28         // Loop Menu Utama
29     }
30     // Loop Menu Utama
31     while (true) {
32         view.displayMenu();
33         String pilihanMenu = view.getPilihanMenuFromUser();
34         controller.handlingPilihanMenu(pilihanMenu);
35     }
36 }
```

d) Bukti Running

```

> MVC_Pattern
└─ MVC_ServiceLayer.DAO
  └─ controller
    J ProductController.class
    J ProductController.java
  └─ dao
    └─ memory
      J ProductDaoInMemory.class
      J ProductDaoInMemory.java
      J ProductDao.class
      J ProductDao.java
  └─ model
    J Product.class
    J Product.java
  └─ service
    J ProductService.class
    J ProductService.java
    J ProductServiceDefault.class
    J ProductServiceDefault.java
  └─ view
    L ProductControllerView.java

C:\Users\user\OneDrive\Documents\projectpbo\PraktikumPBO\tugaspraktikum\praktikum7>cd LayeredArchitecturePattern\MVC_ServiceLayer.DAO
C:\Users\user\OneDrive\Documents\projectpbo\PraktikumPBO\tugaspraktikum\praktikum7\LayeredArchitecturePattern\MVC_ServiceLayer.DAO>javac Main.java controller/*.java dao/*.java memory/*.java model/*.java service/*.java view/*.java
C:\Users\user\OneDrive\Documents\projectpbo\PraktikumPBO\tugaspraktikum\praktikum7\LayeredArchitecturePattern\MVC_ServiceLayer.DAO>java MVC_ServiceLayer.DAO.Main
Error: Could not find or load main class MVC_ServiceLayer.DAO.Main
Caused by: java.lang.ClassNotFoundException: MVC_ServiceLayer.DAO.Main
C:\Users\user\OneDrive\Documents\projectpbo\PraktikumPBO\tugaspraktikum\praktikum7\LayeredArchitecturePattern\MVC_ServiceLayer.DAO>cd ..
C:\Users\user\OneDrive\Documents\projectpbo\PraktikumPBO\tugaspraktikum\praktikum7\LayeredArchitecturePattern>java MVC_ServiceLayer.DAO.Main
--- APLIKASI MVC + SERVICE LAYER + DAO PATTERN ---
1. Tampilkan Semua Produk
2. Tambah Produk Baru
3. Keluar
Pilih opsi: 1

--- Daftar Produk ---
1 - Laptop ASUS Rp. 9500000
2 - Monitor Dell Rp. 2500000

--- APLIKASI MVC + SERVICE LAYER + DAO PATTERN ---
1. Tampilkan Semua Produk
2. Tambah Produk Baru
3. Keluar
Pilih opsi: 2
Masukkan Nama Produk: Laptop Imac
Masukkan Harga Produk: 6000000
INFO: Produk berhasil ditambahkan!

--- APLIKASI MVC + SERVICE LAYER + DAO PATTERN ---
1. Tampilkan Semua Produk
2. Tambah Produk Baru
3. Keluar
Pilih opsi: 1

--- Daftar Produk ---
1 - Laptop ASUS Rp. 9500000
2 - Monitor Dell Rp. 2500000
3 - Laptop Imac Rp. 6000000

--- APLIKASI MVC + SERVICE LAYER + DAO PATTERN ---
1. Tampilkan Semua Produk

```

```

--- APLIKASI MVC + SERVICE LAYER + DAO PATTERN ---
1. Tampilkan Semua Produk
2. Tambah Produk Baru
3. Keluar
Pilih opsi: 1

--- Daftar Produk ---
1 - Laptop ASUS Rp. 9500000
2 - Monitor Dell Rp. 2500000

--- APLIKASI MVC + SERVICE LAYER + DAO PATTERN ---
1. Tampilkan Semua Produk

```

```
J ProductServiceDefault.class
J ProductServiceDefault.java U
< view ●
J ProductConsoleView.class
J ProductConsoleView.java U
J Main.class
J Main.java U
> ServiceLayer_Pattern ●
> TanpaPattern ●
OUTLINE
TIMELINE
```

2. Tambah Produk Baru  
3. Keluar  
Pilih opsi: 1

--- Daftar Produk ---  
1 - Laptop ASUS Rp. 9500000  
2 - Monitor Dell Rp. 2500000  
3 - Laptop Imac Rp. 6000000

--- APLIKASI MVC + SERVICE LAYER + DAO PATTERN ---  
1. Tampilkan Semua Produk  
2. Tambah Produk Baru  
3. Keluar

Pilih opsi: 3  
INFO: Aplikasi ditutup.