Praktikum 7 - Layered Architecture Pattern

Nama : Cania Nabilatul Adawah

NIM : 2403102

Kelas : D3TI2C

Mata Kuliah : Pemrograman Berbasis Objek

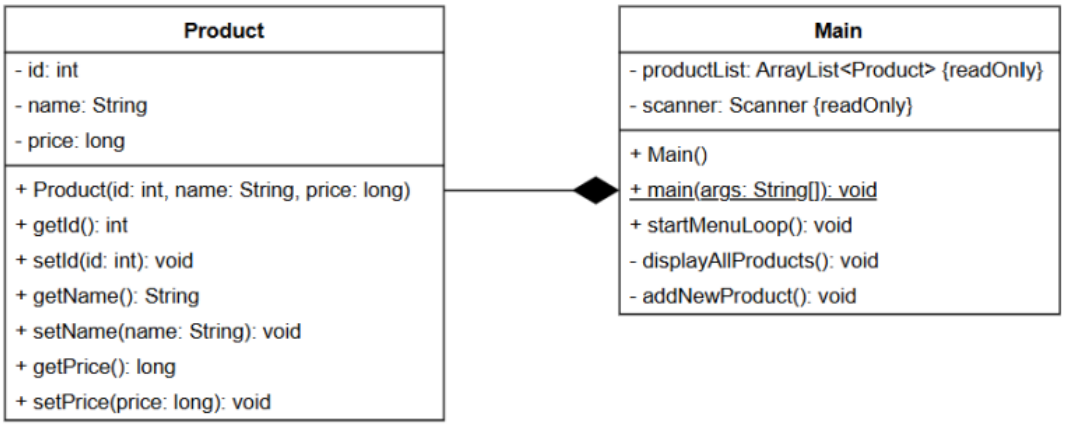
1. Layered Architecture Pattern adalah pola arsitektur perangkat lunak yang memisahkan sistem menjadi beberapa lapisan logis (layers) dengan prinsip Separation of Concerns (pemisahan tanggung jawab).
Tujuannya: meningkatkan keteraturan, kemudahan pemeliharaan, dan skalabilitas.

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.

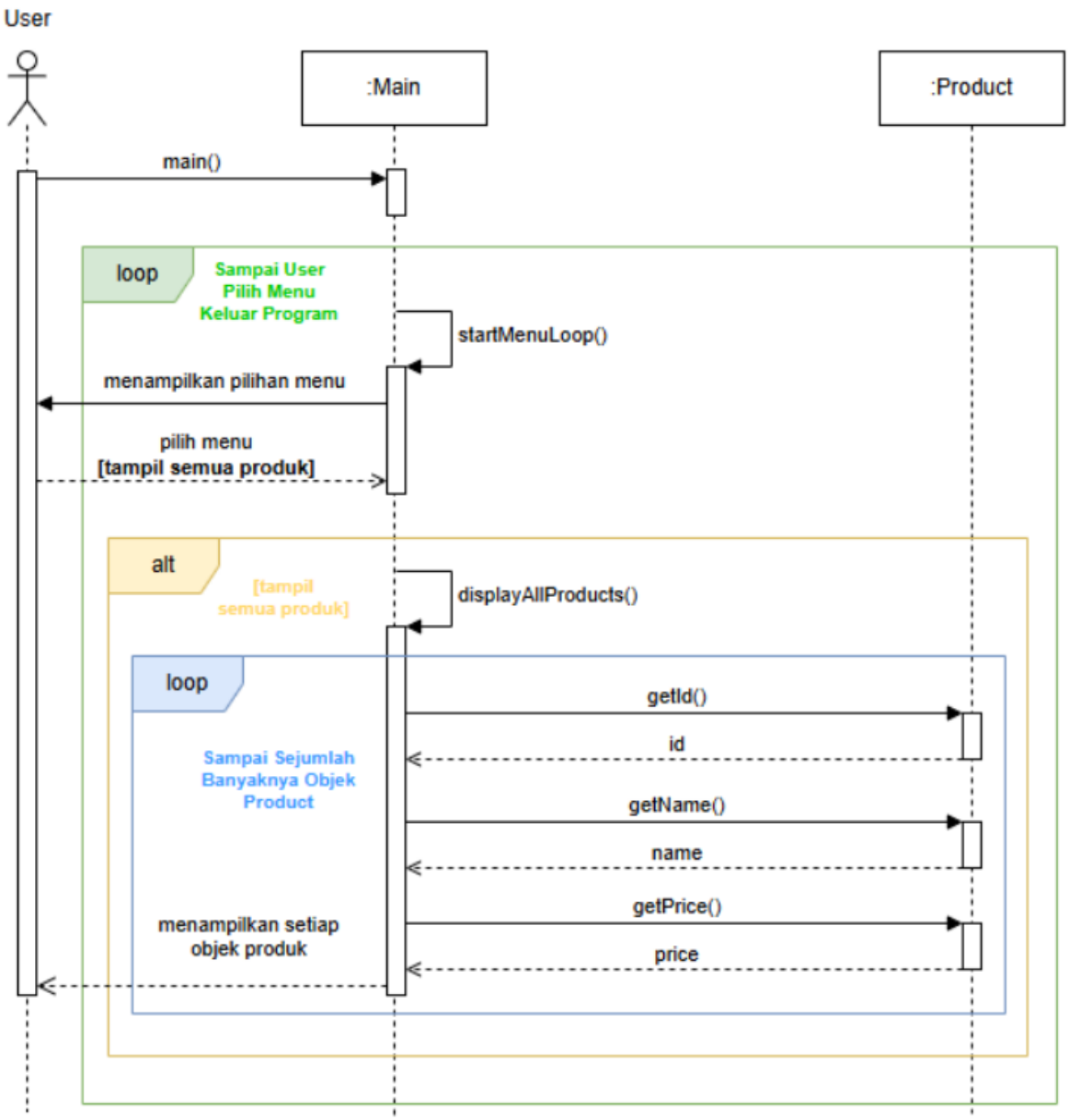
2. Tanpa Pattern

a) Class Diagram

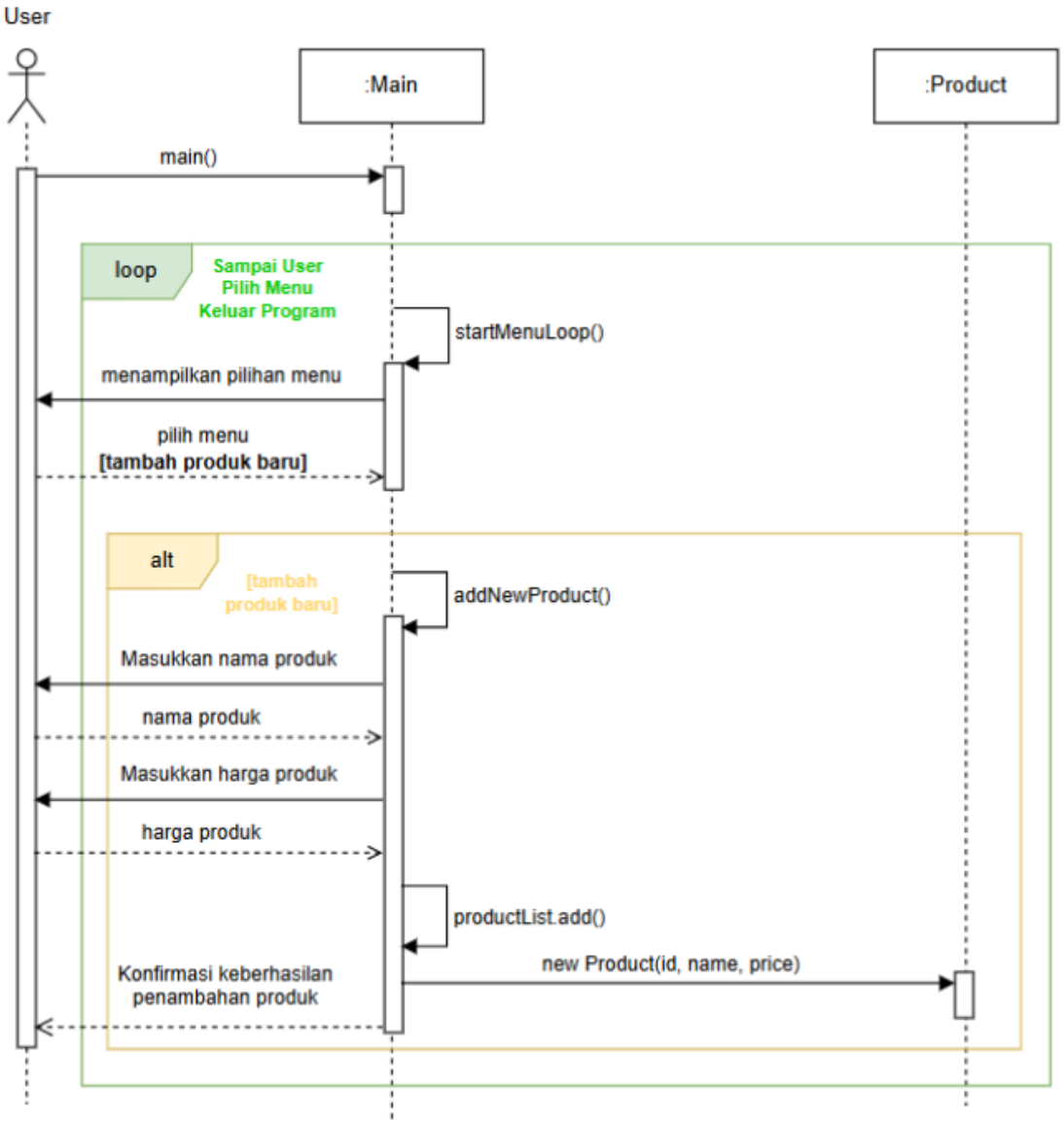


b) Sequence Diagram

1) Menampilkan semua daftar produk

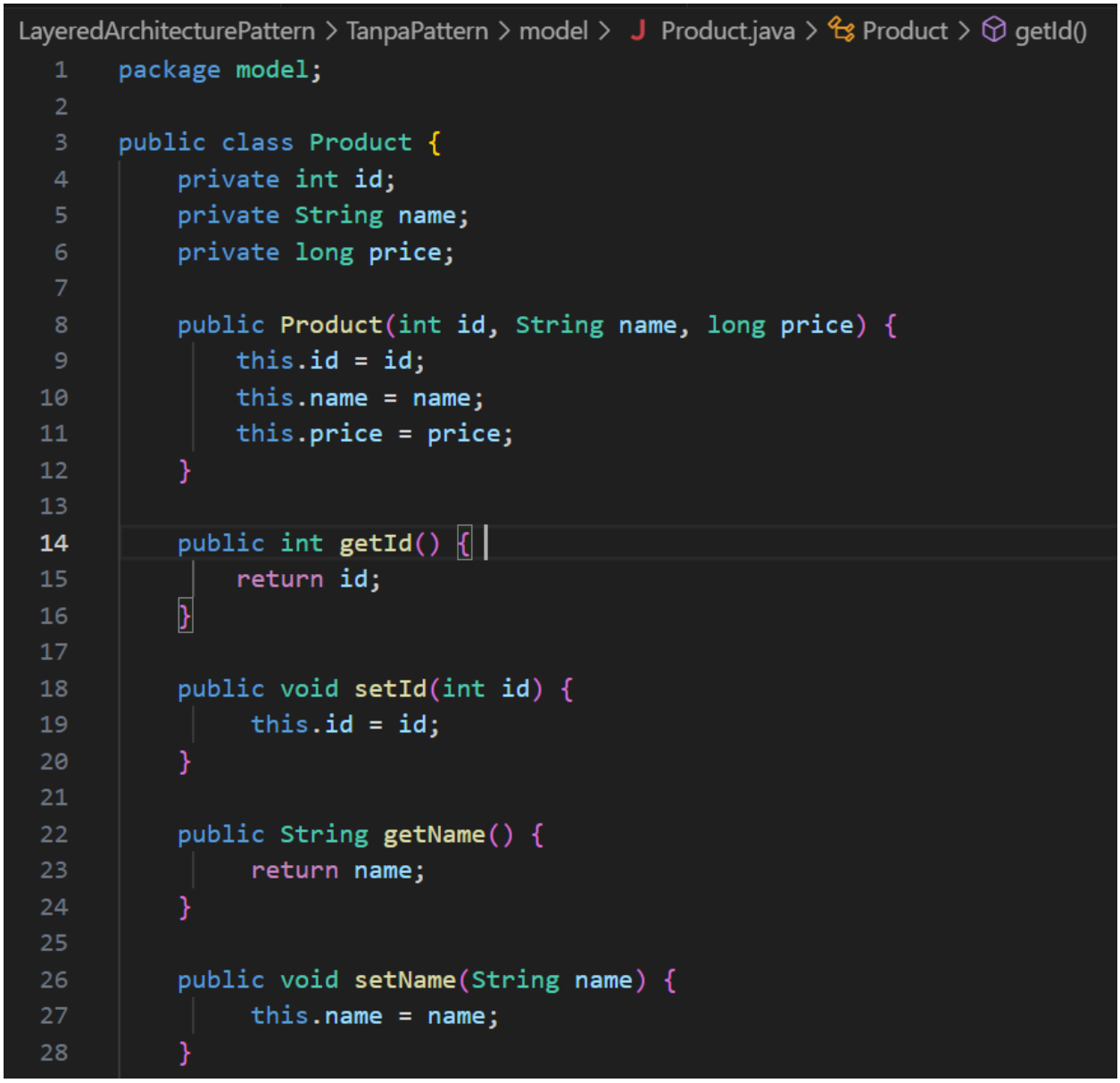


2) Menambah produk baru



c) Kode Program

1) Product.java


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

2) Main.java

LayeredArchitecturePattern > TanpaPattern > Main.java > Main

```java
import java.util.ArrayList;
import java.util.Scanner;

import model.Product;

public class Main {

    private final ArrayList<Product> productList = new ArrayList<>();
    private final Scanner scanner = new Scanner(System.in);

    public Main() {
        productList.add(new Product(id: 1, name: "Laptop ASUS", price: 9500000));
        productList.add(new Product(id: 2, name: "Monitor Dell", price: 2500000));
    }

    Run | Debug
    public static void main(String[] args) {
        Main app = new Main();
        app.startMenuLoop();
        app.scanner.close();
    }


    public void startMenuLoop() {
        boolean running = true;
        while (running) {
            System.out.println(x: "\n--- APLIKASI TANPA PATTERN ---");
            System.out.println(x: "1. Tampilkan Semua Produk");
            System.out.println(x: "2. Tambah Produk Baru");
```

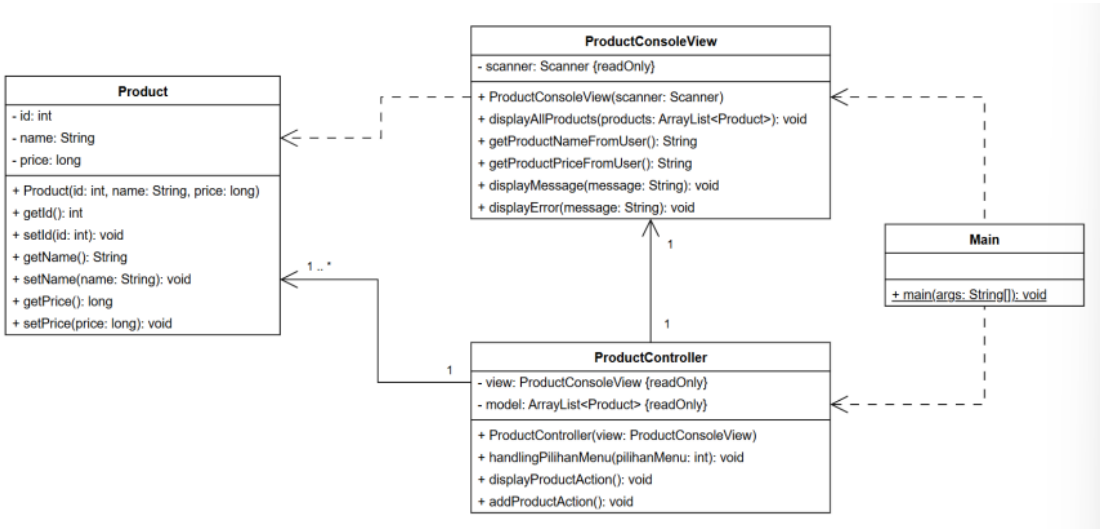
```
28              System.out.println(x: "2. Tambah Produk Baru");
29              System.out.println(x: "3. Keluar");
30              System.out.print(s: "Pilih opsi: ");
31
32              try {
33                  int choice = Integer.parseInt(scanner.nextLine());
34                  switch (choice) {
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      }
53
```

d) Bukti Running

```
PS C:\Users\user\OneDrive\Documents\projectpbo\PraktikumPBO\tugaspraktikum\praktikum7>  & 'C:\Program Files\Java\jdk-21\bi
n\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\user\AppData\Roaming\Code\User\workspaceStorage\b32c
409f27f0f6b8f20f47117824d4ae\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>
```

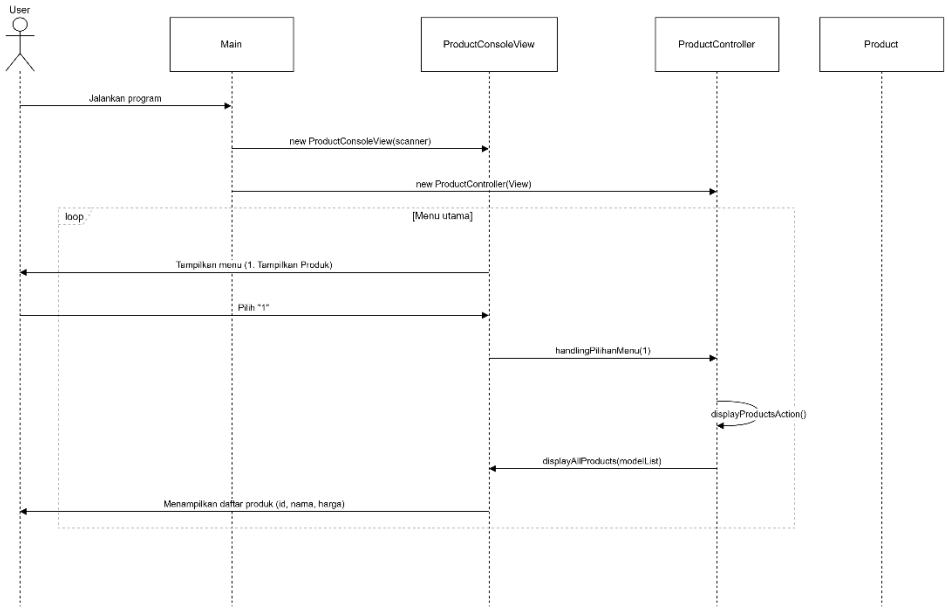
3. MVC Pattern

a) Class Diagram

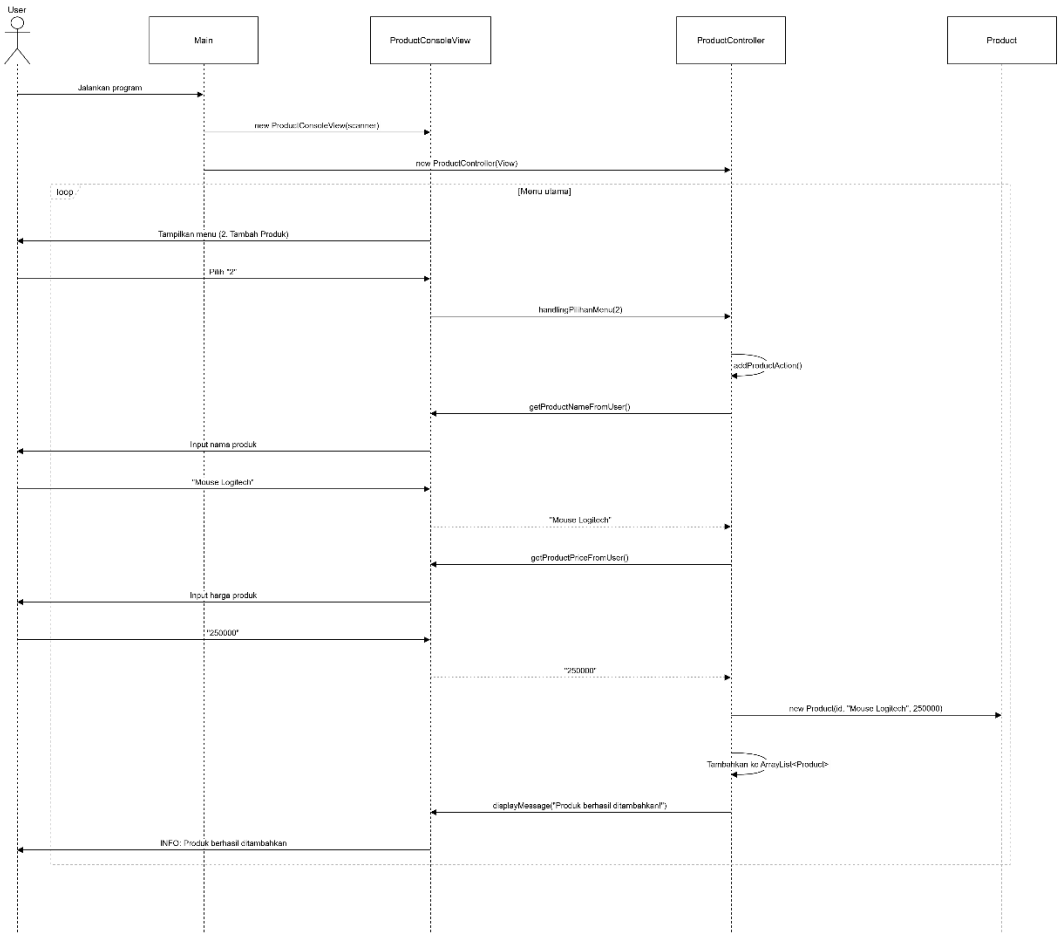


b) Sequence Diagram

1) Menampilkan semua daftar produk



2) Menambah produk baru



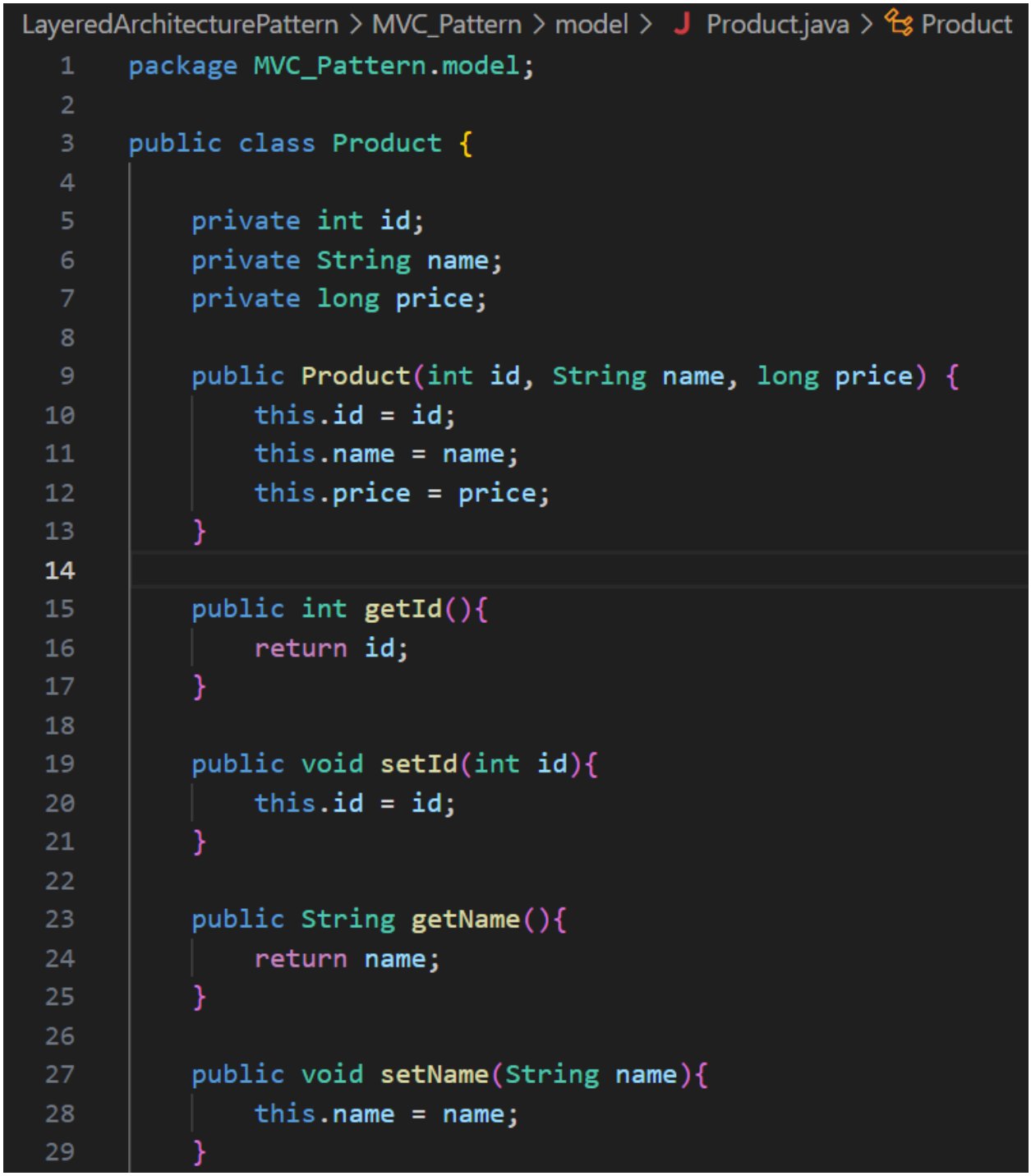
c) Kode Program

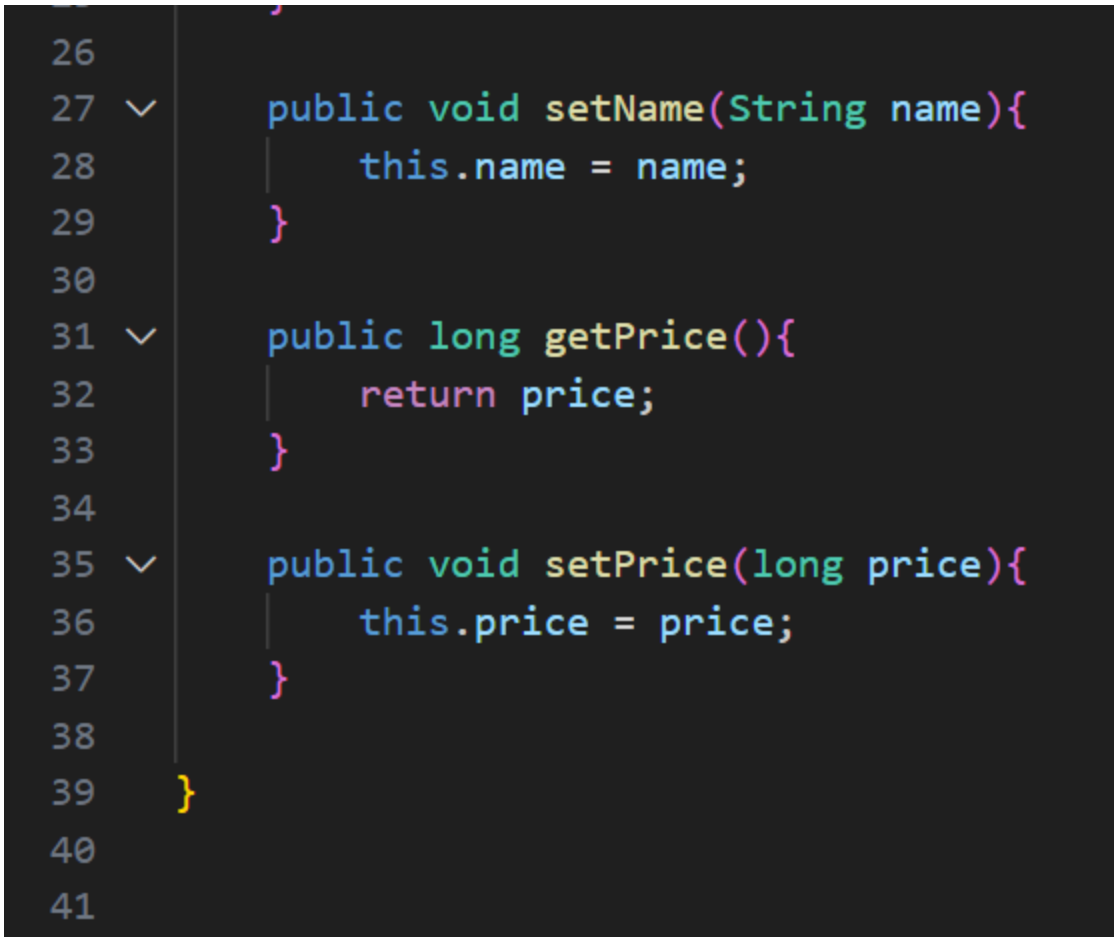
1) ProductController.java

LayeredArchitecturePattern > MVC_Pattern > controller > ProductController.java > ProductController > handlingPilihanMenu(int)

```java
package MVC_Pattern.controller;

import java.util.ArrayList;

import MVC_Pattern.model.Product;
import MVC_Pattern.view.ProductConsoleView;

public class ProductController {
    private final ProductConsoleView view;
    private final ArrayList<Product> model = new ArrayList<>();

    public ProductController(ProductConsoleView view) {
        this.view = view;
        model.add(new Product(id: 1, name: "Laptop ASUS", price: 9500000));
        model.add(new Product(id: 2, name: "Monitor Dell", price: 2500000));
    }

    public void handlingPilihanMenu(int pilihanMenu) {
        switch (pilihanMenu) {
            case 1:
                displayProductsAction();
                break;
            case 2:
                addProductAction();
                break;
            case 3:
                view.displayMessage(message: "Keluar dari aplikasi.");
                break;
            default:
                view.displayError(message: "Opsi tidak valid.");
        }
    }

    private void displayProductsAction() {
        view.displayAllProducts(model);
    }

    private void addProductAction() {
        String name = view.getProductNameFromUser();
        String priceStr = view.getProductPriceFromUser();
        try {
            long price = Long.parseLong(priceStr);
            if (price <= 0) {
                throw new IllegalArgumentException(s: "Harga harus angka positif lebih dari 0.");
            }
            int newId = model.size() + 1;
            model.add(new Product(newId, name, price));
            view.displayMessage(message: "Produk berhasil ditambahkan!");
        } catch (IllegalArgumentException e) {
            view.displayError("Gagal menambah produk: " + e.getMessage());
        }
    }
}

```

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 > J ProductConsoleView.java > ...

```java
package MVC_Pattern.view;

import java.util.ArrayList;
import java.util.Scanner;
import MVC_Pattern.model.Product;

public class ProductConsoleView {
    private final Scanner scanner;

    public ProductConsoleView(Scanner scanner) {
        this.scanner = scanner;
    }

    public void displayAllProducts(ArrayList<Product> products) {
        System.out.println(x: "\n=== DAFTAR PRODUK ===");
        for (Product p : products) {
            System.out.println(p.getId() + ". " + p.getName() + " - Rp" + p.getPrice());
        }
    }

    public String getProductNameFromUser() {
        System.out.print(s: "Masukkan nama produk: ");
        return scanner.nextLine();
    }

    public String getProductPriceFromUser() {
        System.out.print(s: "Masukkan harga produk: ");
        return scanner.nextLine();
    }
```

```java

    public void displayMessage(String message) {
        System.out.println(message);
    }

    public void displayError(String message) {
        System.err.println("Error: " + message);
    }
}

```

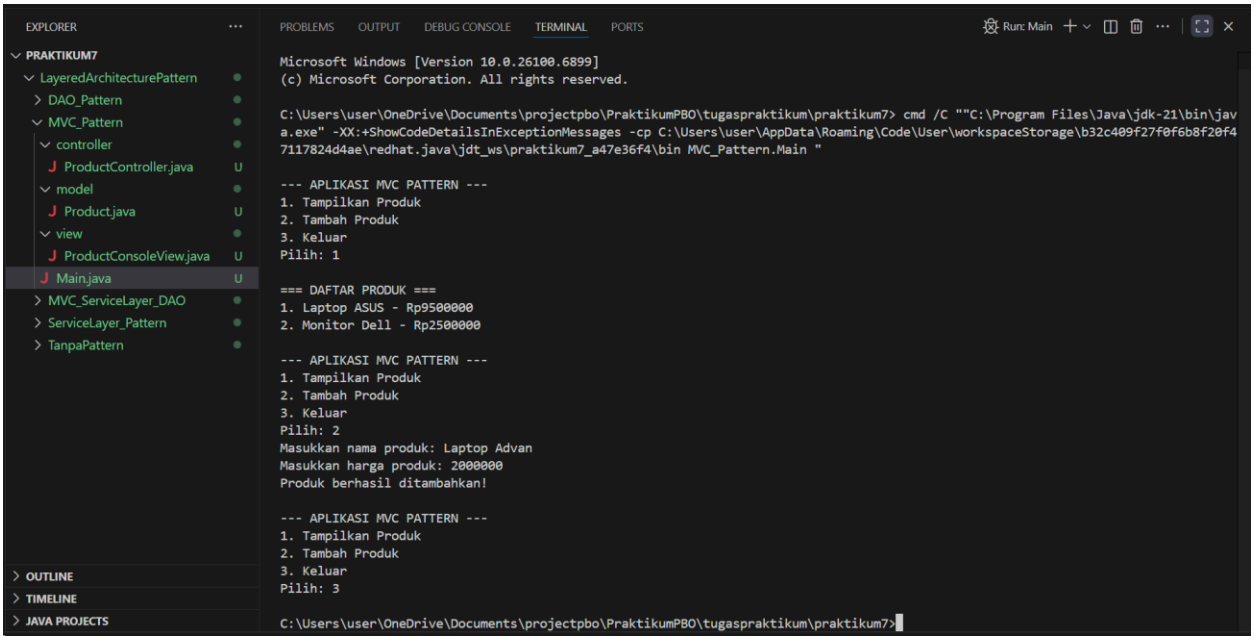
4) Main.java

LayeredArchitecturePattern > MVC_Pattern > Main.java > Main

```java
package MVC_Pattern;

import MVC_Pattern.controller.ProductController;
import MVC_Pattern.view.ProductConsoleView;
import java.util.Scanner;

public class Main {
    Run | Debug
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        ProductConsoleView view = new ProductConsoleView(scanner);
        ProductController controller = new ProductController(view);

        while(true){
            System.out.println(x: "\n--- APLIKASI MVC PATTERN ---");
            System.out.println(x: "1. Tampilkan Produk");
            System.out.println(x: "2. Tambah Produk");
            System.out.println(x: "3. Keluar");
            System.out.print(s: "Pilih: ");

            try {
                int pilihan = Integer.parseInt(scanner.nextLine());
                if (pilihan == 3) break;
                controller.handlingPilihanMenu(pilihan);
            } catch (NumberFormatException e) {
                view.displayError(message: "Input tidak valid. Masukkan angka.");
            }
        }

```

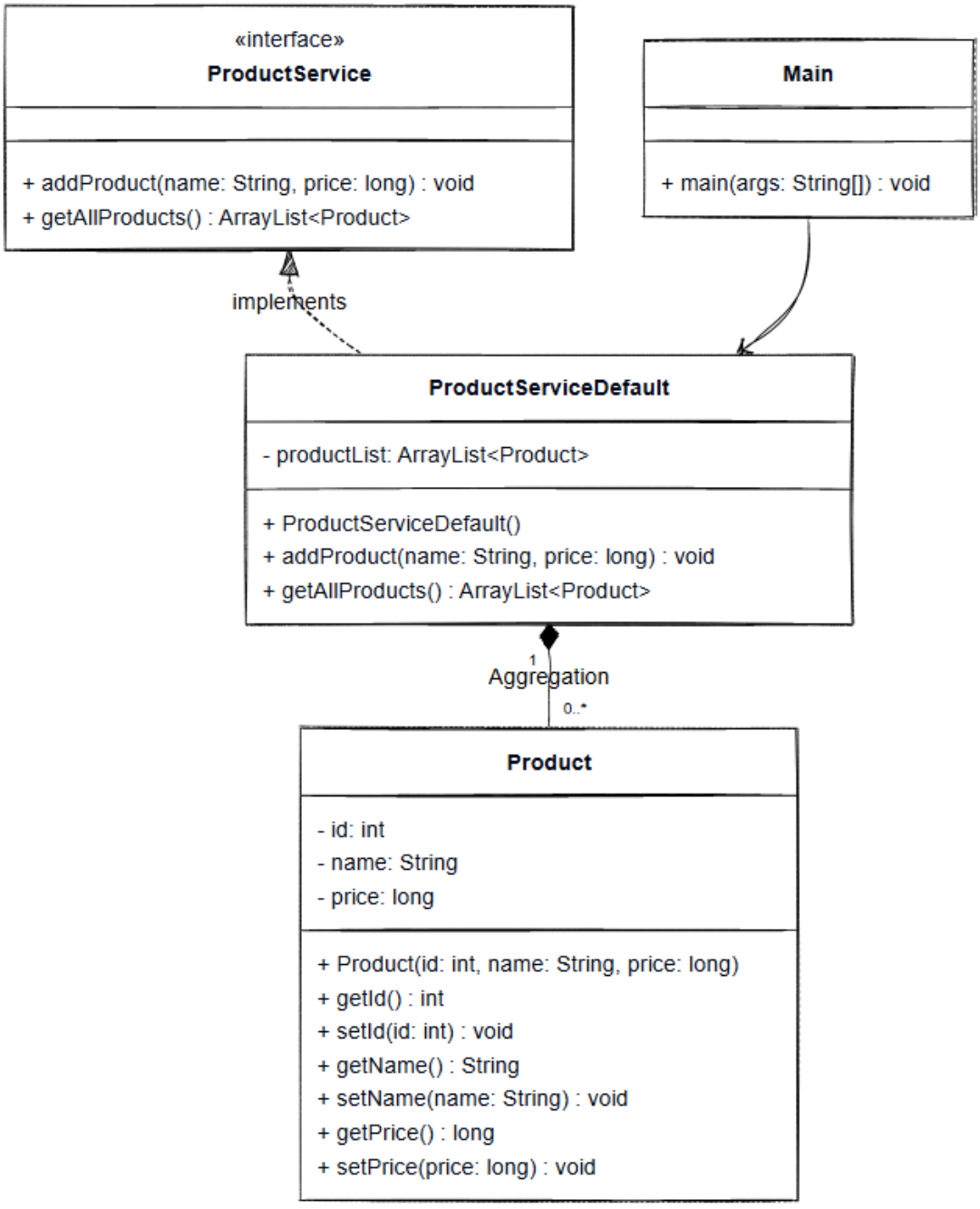
```java
        while(true){
            System.out.println("\n--- APLIKASI MVC PATTERN ---");
            System.out.println("1. Tampilkan Produk");
            System.out.println("2. Tambah Produk");
            System.out.println("3. Keluar");
            System.out.print("Pilih: ");

            try {
                int pilihan = Integer.parseInt(scanner.nextLine());
                if (pilihan == 3) break;
                controller.handlingPilihanMenu(pilihan);
            } catch (NumberFormatException e) {
                view.displayError("Input tidak valid. Masukkan angka.");
            }
        }

        scanner.close();
    }
}

```

d) Bukti Running



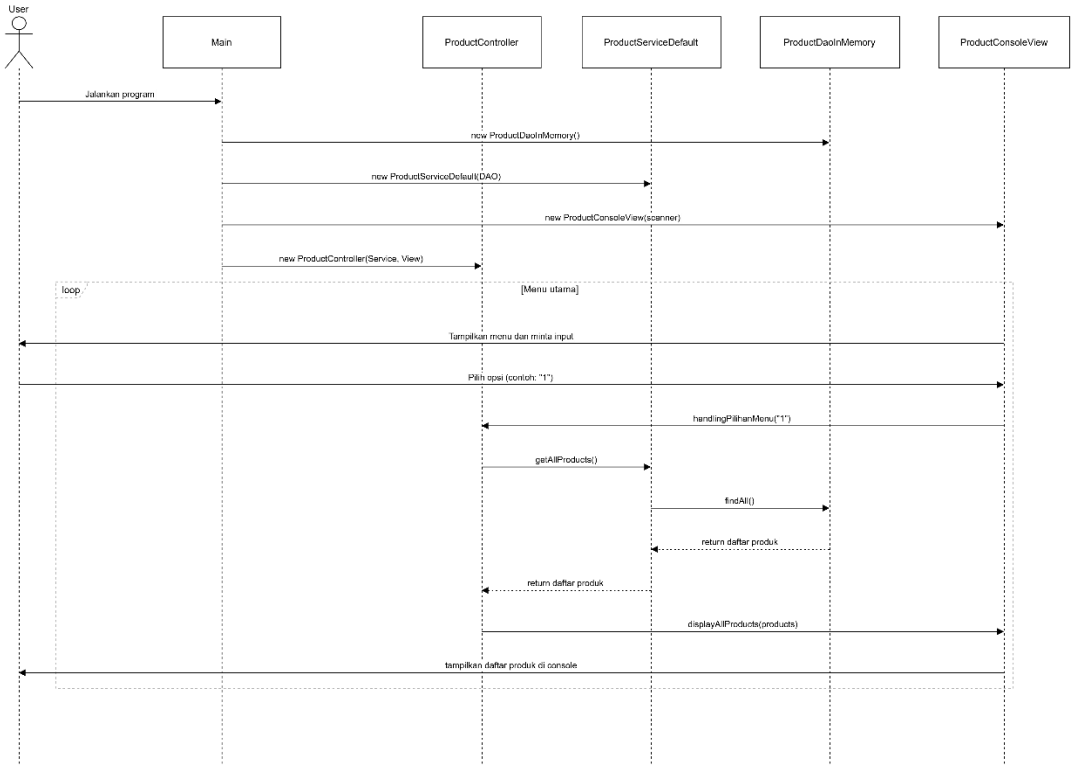
4. Service Layer Pattern

a) Class Diagram

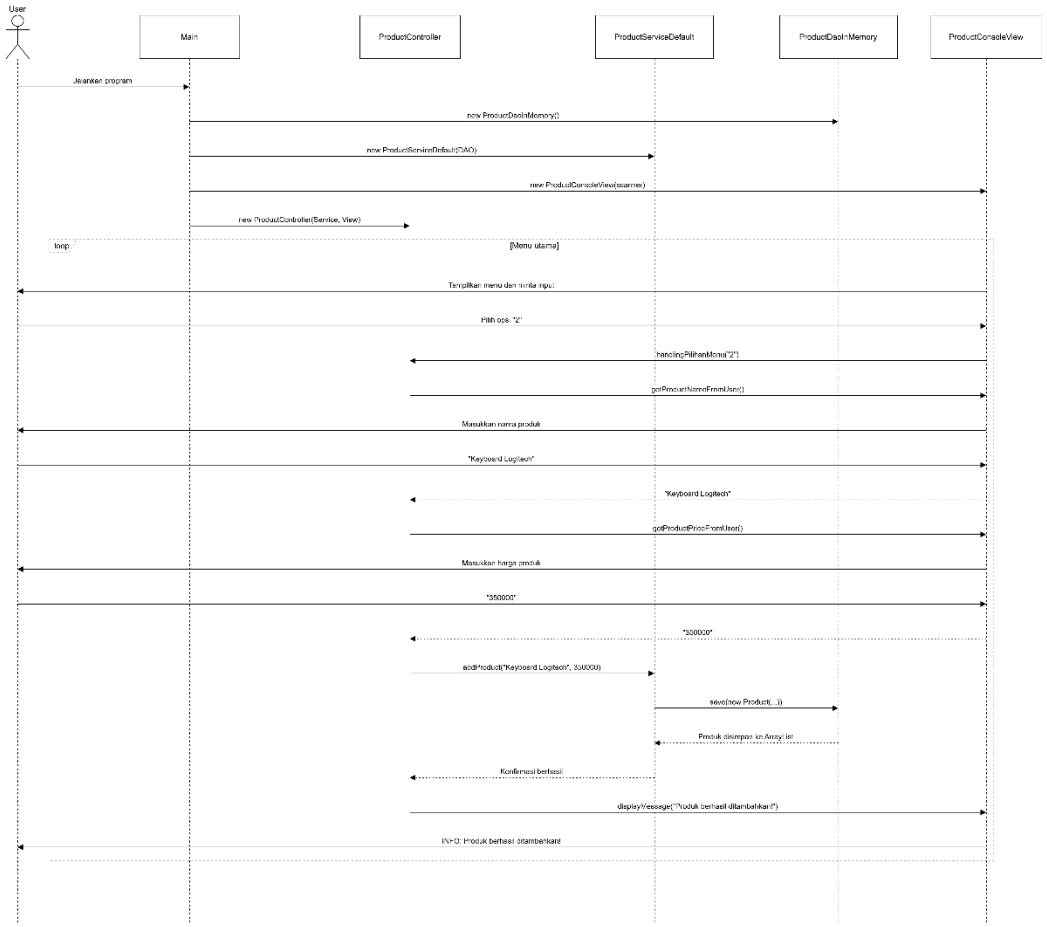


b) Sequence Diagram

1) Menampilkan semua daftar produk



2) Menambah produk baru



c) Kode Program

1) Product.java

LayeredArchitecturePattern > ServiceLayer_Pattern > model > Product.java > Product

```java
package ServiceLayer_Pattern.model;

public class Product {
    private int id;
    private String name;
    private long price;

    public Product(int id, String name, long price) {
        this.id = id;
        this.name = name;
        this.price = price;
    }

    public int getId() { return id; }
    public String getName() { return name; }
    public long getPrice() { return price; }
}

```

2) ProductService.java

LayeredArchitecturePattern > ServiceLayer_Pattern > service > ProductService.java >

```java
package ServiceLayer_Pattern.service;

import java.util.ArrayList;
import ServiceLayer_Pattern.model.Product;

public interface ProductService {
    void addProduct(String name, long price);
    ArrayList<Product> getAllProducts();
}

```

3) ProductServiceDefault.java

LayeredArchitecturePattern > ServiceLayer_Pattern > service > J ProductServiceDefault.java > ...

```java
package ServiceLayer_Pattern.service;

import java.util.ArrayList;
import ServiceLayer_Pattern.model.Product;

public class ProductServiceDefault implements ProductService {

    private final ArrayList<Product> productList = new ArrayList<>();

    public ProductServiceDefault() {
        productList.add(new Product(id: 1, name: "Laptop ASUS", price: 9500000));
        productList.add(new Product(id: 2, name: "Monitor Dell", price: 2500000));
    }

    @Override
    public void addProduct(String name, long price) {
        if (price <= 0) {
            throw new IllegalArgumentException(s: "Harga harus angka positif lebih dari 0.");
        }
        int newId = productList.size() + 1;
        productList.add(new Product(newId, name, price));
    }

    @Override
    public ArrayList<Product> getAllProducts() {
        return productList;
    }
}

```

4) Main.java

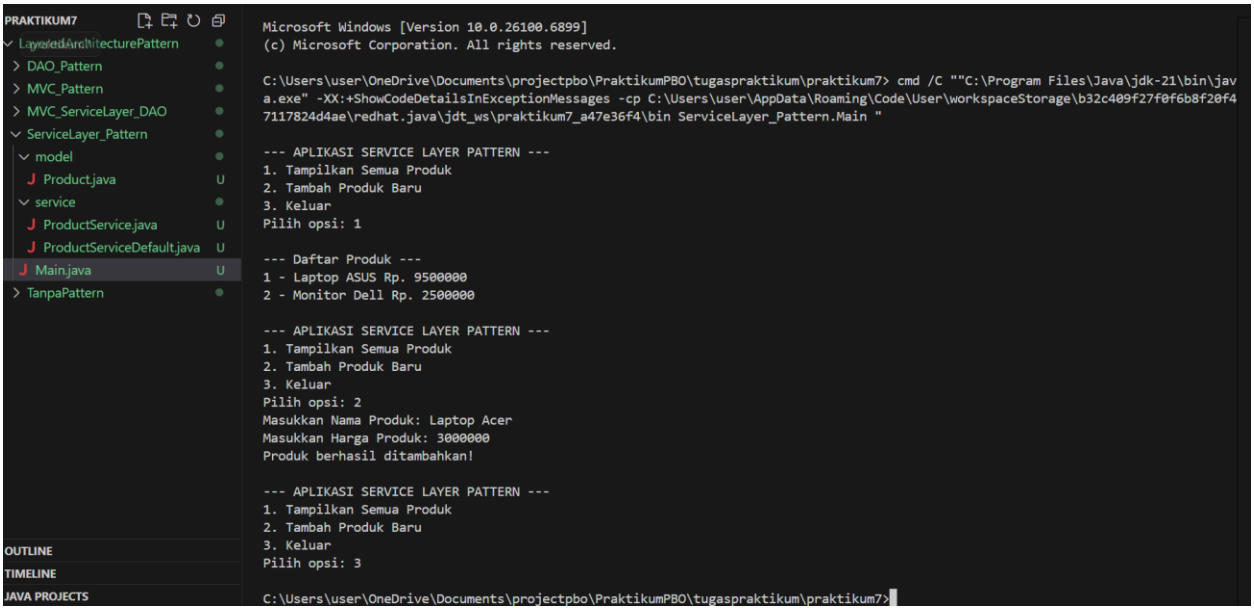
LayeredArchitecturePattern > ServiceLayer_Pattern > Main.java > ...

```java
package ServiceLayer_Pattern;

import java.util.List;
import java.util.Scanner;

import ServiceLayer_Pattern.model.Product;
import ServiceLayer_Pattern.service.ProductServiceDefault;

public class Main {

    private final ProductServiceDefault productService = new ProductServiceDefault();
    private final Scanner scanner = new Scanner(System.in);

    Run | Debug
    public static void main(String[] args) {
        Main app = new Main();
        app.startMenuLoop();
        app.scanner.close();
    }

    public void startMenuLoop() {
        boolean running = true;
        while (running) {
            System.out.println(x: "\n--- APLIKASI SERVICE LAYER PATTERN ---");
            System.out.println(x: "1. Tampilkan Semua Produk");
            System.out.println(x: "2. Tambah Produk Baru");
            System.out.println(x: "3. Keluar");
            System.out.print(s: "Pilih opsi: ");
```

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

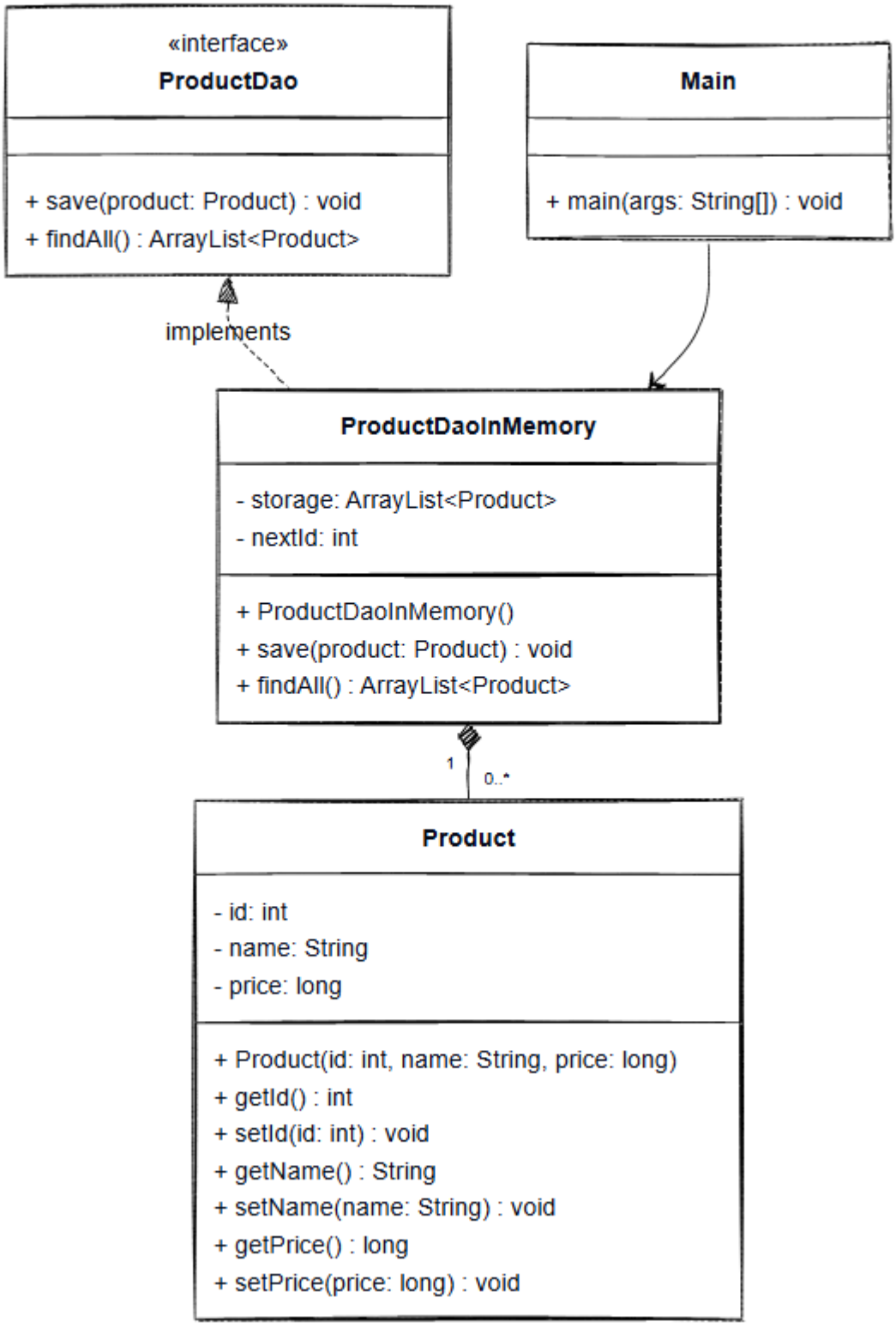
```
54                  System.out.println(product.getId() + " - "
55                          + product.getName() + " Rp. " + product.getPrice());
56              }
57          }
58
59          private void addNewProduct() {
60              System.out.print(s: "Masukkan Nama Produk: ");
61              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



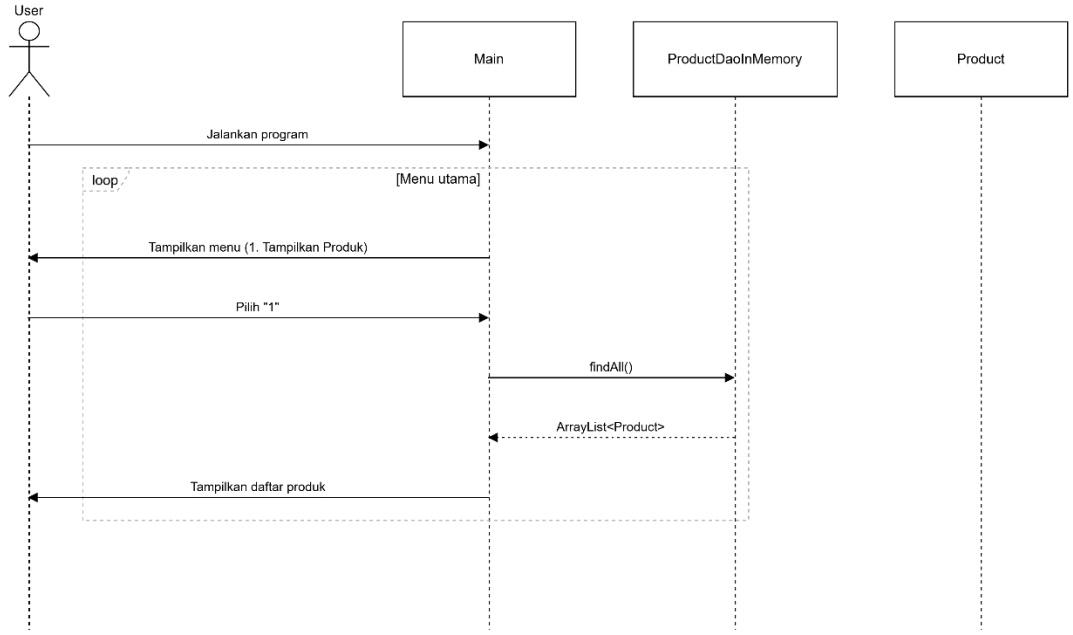
5. DAO Pattern

a) Class Diagram

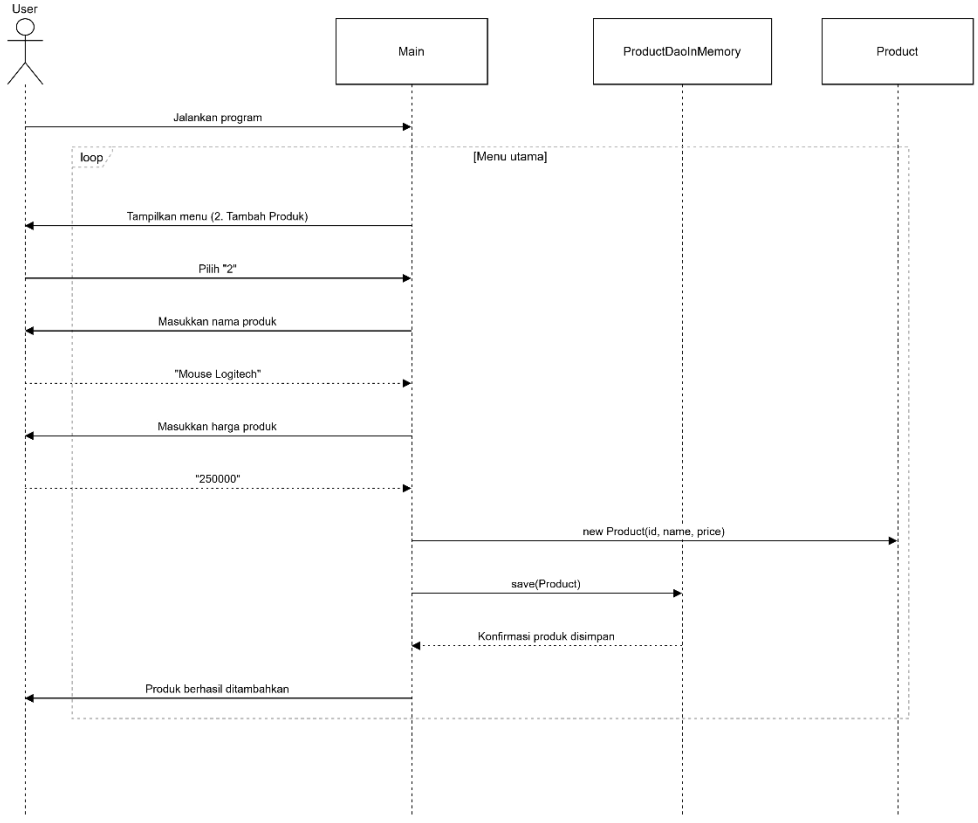


b) Sequence Diagram

1) Menampilkan semua daftar produk



2) Menambah produk baru



c) Kode Program

1) ProductDaoInMemory.java

LayeredArchitecturePattern > DAO_Pattern > dao > memory > ProductDaoInMemory.java > ...

```java
package dao.memory;

import java.util.ArrayList;
import dao.ProductDao;
import model.Product;

public class ProductDaoInMemory implements ProductDao {

    private final ArrayList<Product> storage = new ArrayList<>();

    public ProductDaoInMemory() {
        // Data awal (contoh)
        storage.add(new Product(id: 1, name: "Laptop ASUS", price: 9500000));
        storage.add(new Product(id: 2, name: "Monitor Dell", price: 2500000));
    }

    @Override
    public void save(Product product) {
        storage.add(product);
    }

    @Override
    public ArrayList<Product> findAll() {
        return storage;
    }
}

```

2) ProductDao.java

LayeredArchitecturePattern > DAO_Pattern > dao > ProductDao.java >

```java
package dao;

import java.util.ArrayList;
import model.Product;

public interface ProductDao {
    void save(Product product);
    ArrayList<Product> findAll();
}

```

3) Product.java

LayeredArchitecturePattern > DAO_Pattern > model > Product.java > ...

```java
package model;

public class Product {
    private int id;
    private String name;
    private long price;

    public Product(int id, String name, long price){
        this.id = id;
        this.name = name;
        this.price = price;
    }

    public int getId() { return id; }
    public void setId(int id){ this.id = id; }

    public String getName() { return name; }
    public void setName(String name) { this.name = name; }

    public long getPrice() { return price; }
    public void setPrice(long price) { this.price = price; }

    @Override
    public String toString() {
        return id + " - " + name + " Rp. " + price;
    }
}

```

4) Main.java

LayeredArchitecturePattern > DAO_Pattern > Main.java > ...

```java
import java.util.ArrayList;
import java.util.Scanner;

import dao.ProductDao;
import dao.memory.ProductDaoInMemory;
import model.Product;

public class Main {
    private final ProductDao productDao = new ProductDaoInMemory();
    private final Scanner scanner = new Scanner(System.in);

    Run | Debug
    public static void main(String[] args) {
        Main app = new Main();
        app.startMenuLoop();
        app.scanner.close();
    }

    public void startMenuLoop() {
        boolean running = true;
        while (running) {
            System.out.println(x: "\n--- APLIKASI DAO PATTERN ---");
            System.out.println(x: "1. Tampilkan Semua Produk");
            System.out.println(x: "2. Tambah Produk Baru");
            System.out.println(x: "3. Keluar");
            System.out.print(s: "Pilih opsi: ");
            try {
                int choice = Integer.parseInt(scanner.nextLine());
                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) {
```

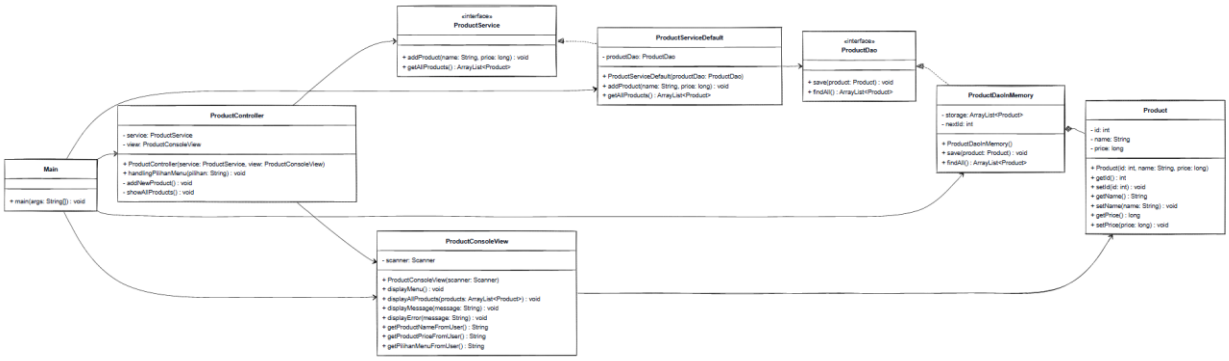
```
55                  System.out.println(product);
56              }
57          }
58      }
59
60      private void addNewProduct() {
61          System.out.print(s: "Nama Produk: ");
62          String name = scanner.nextLine();
63
64          System.out.print(s: "Harga Produk: ");
65          try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

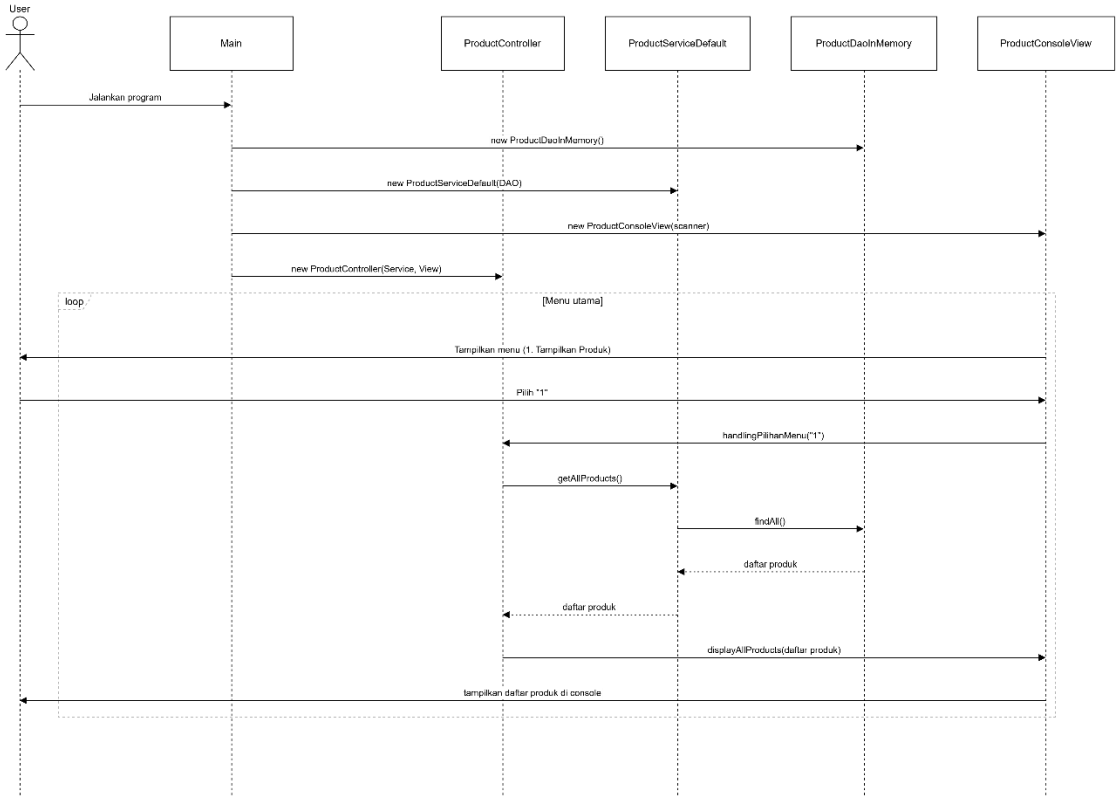
6. MVC + Service Layer + DAO Pattern

a) Class Diagram

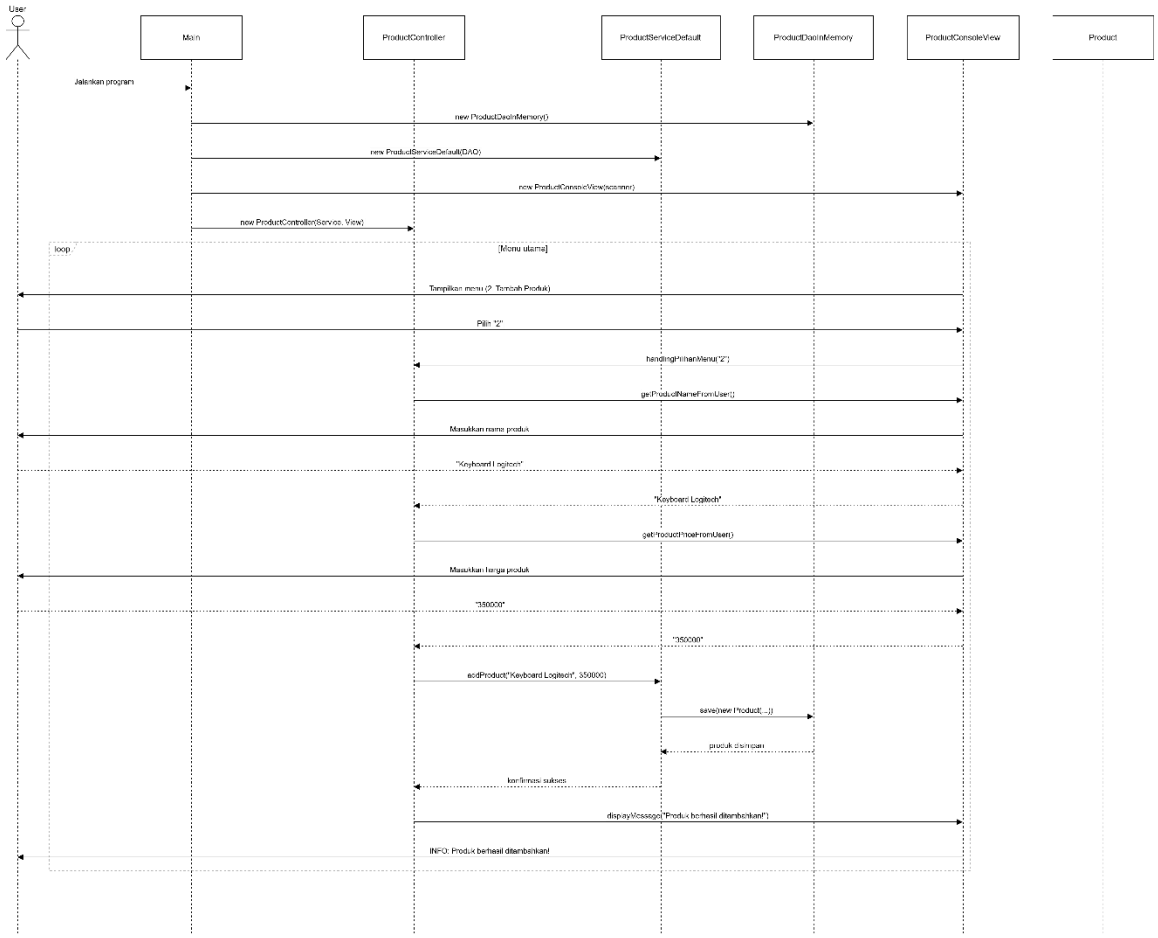


b) Sequence Diagram

1) Menampilkan semua daftar produk



2) Menambah produk baru



c) Kode Program

1) ProductController.java

LayeredArchitecturePattern > MVC_ServiceLayer_DAO > controller > ProductController.java > ProductController

```java
package MVC_ServiceLayer_DAO.controller;

import MVC_ServiceLayer_DAO.service.ProductService;
import MVC_ServiceLayer_DAO.view.ProductConsoleView;

public class ProductController {

    private final ProductService service;
    private final ProductConsoleView view;

    public ProductController(ProductService service, ProductConsoleView view) {
        this.service = service;
        this.view = view;
    }

    public void handlingPilihanMenu(String pilihanMenu) {
        try {
            int menu = Integer.parseInt(pilihanMenu);
            switch (menu) {
                case 1:
                    showAllProducts();
                    break;
                case 2:
                    addNewProduct();
                    break;
                case 3:
                    view.displayMessage(message: "Aplikasi ditutup.");
                    System.exit(status: 0);
                    break;
```

```
                    break;
                default:
                    view.displayError(message: "Opsi tidak valid.");
                    break;
            }
        } catch (NumberFormatException e) {
            view.displayError(message: "Input tidak valid. Masukkan angka.");
        }
    }

    private void addNewProduct() {
        String name = view.getProductNameFromUser();
        String priceStr = view.getProductPriceFromUser();
        try {
            long price = Long.parseLong(priceStr);
            service.addProduct(name, price);
            view.displayMessage(message: "Produk berhasil ditambahkan!");
        } catch (Exception e) {
            view.displayError("Gagal menambah produk: " + e.getMessage());
        }
    }

    private void showAllProducts() {
        view.displayAllProducts(service.getAllProducts());
    }
}

```

2) ProductDaoInMemory.java

LayeredArchitecturePattern > MVC_ServiceLayer_DAO > dao > memory > ProductDaoInMemory.java > Pro

```java
package MVC_ServiceLayer_DAO.dao.memory;

import java.util.ArrayList;
import MVC_ServiceLayer_DAO.dao.ProductDao;
import MVC_ServiceLayer_DAO.model.Product;

public class ProductDaoInMemory implements ProductDao {

    private final ArrayList<Product> storage = new ArrayList<>();
    private int nextId = 1;

    public ProductDaoInMemory() {
        storage.add(new Product(id: 1, name: "Laptop ASUS", price: 9500000));
        storage.add(new Product(id: 2, name: "Monitor Dell", price: 2500000));
        nextId = 3;
    }

    @Override
    public void save(Product p) {
        if (p.getId() == 0) {
            p.setId(nextId++);
        }
        storage.add(p);
    }

    @Override
    public ArrayList<Product> findAll() {
        return storage;
    }
```

```java
    @Override
    public ArrayList<Product> findAll() {
        return storage;
    }
}
```

3) ProductDao.java

LayeredArchitecturePattern > MVC_ServiceLayer_DAO > dao > ProductDao.java > ...

```java
package MVC_ServiceLayer_DAO.dao;

import java.util.ArrayList;
import MVC_ServiceLayer_DAO.model.Product;

public interface ProductDao {
    void save(Product product);
    ArrayList<Product> findAll();
}

```

4) Product.java

LayeredArchitecturePattern > MVC_ServiceLayer_DAO > model > Product.java > ...

```java
package MVC_ServiceLayer_DAO.model;

public class Product {

    private int id;
    private String name;
    private long price;

    public Product(int id, String name, long price) {
        this.id = id;
        this.name = name;
        this.price = price;
    }

    public int getId() { return id; }
    public void setId(int id) { this.id = id; }

    public String getName() { return name; }
    public void setName(String name) { this.name = name; }

    public long getPrice() { return price; }
    public void setPrice(long price) { this.price = price; }
}

```

5) ProductService.java

LayeredArchitecturePattern > MVC_ServiceLayer_DAO > service > ProductService.java > ...

```java
package MVC_ServiceLayer_DAO.service;

import java.util.ArrayList;
import MVC_ServiceLayer_DAO.model.Product;

public interface ProductService {
    void addProduct(String name, long price);
    ArrayList<Product> getAllProducts();
}

```

6) ProductServiceDefault.java

LayeredArchitecturePattern > MVC_ServiceLayer_DAO > service > ProductServiceDefault.java > ...

```java
package MVC_ServiceLayer_DAO.service;

import java.util.ArrayList;
import MVC_ServiceLayer_DAO.dao.ProductDao;
import MVC_ServiceLayer_DAO.model.Product;

public class ProductServiceDefault implements ProductService {

    private final ProductDao productDao;

    public ProductServiceDefault(ProductDao productDao) {
        this.productDao = productDao;
    }

    @Override
    public void addProduct(String name, long price) {
        if (name == null || name.trim().isEmpty()) {
            throw new IllegalArgumentException(s: "Nama tidak boleh kosong.");
        }
        if (price <= 0) {
            throw new IllegalArgumentException(s: "Harga harus > 0");
        }
        productDao.save(new Product(id: 0, name, price));
    }

    @Override
    public ArrayList<Product> getAllProducts() {
        return productDao.findAll();
    }
}
```

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

7) ProductConsoleView.java

LayeredArchitecturePattern > MVC_ServiceLayer_DAO > view > J 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(x: "\n--- APLIKASI MVC + SERVICE LAYER + DAO PATTERN ---");
17          System.out.println(x: "1. Tampilkan Semua Produk");
18          System.out.println(x: "2. Tambah Produk Baru");
19          System.out.println(x: "3. Keluar");
20          System.out.print(s: "Pilih opsi: ");
21      }
22
23      public void displayAllProducts(ArrayList<Product> products) {
24          System.out.println(x: "\n--- Daftar Produk ---");
25          if (products.isEmpty()) {
26              System.out.println(x: "Tidak ada produk tersedia.");
27          } else {
28              for (Product product : products) {
29                  System.out.println(product.getId() + " - " + product.getName()
```

```
                    + " Rp. " + product.getPrice());
            }
        }
    }

    public void displayMessage(String message) {
        System.out.println("INFO: " + message);
    }

    public void displayError(String message) {
        System.out.println("ERROR: " + message);
    }

    public String getProductNameFromUser() {
        System.out.print(s: "Masukkan Nama Produk: ");
        return scanner.nextLine();
    }

    public String getProductPriceFromUser() {
        System.out.print(s: "Masukkan Harga Produk: ");
        return scanner.nextLine();
    }

    public String getPilihanMenuFromUser() {
        return scanner.nextLine();
    }
}
```

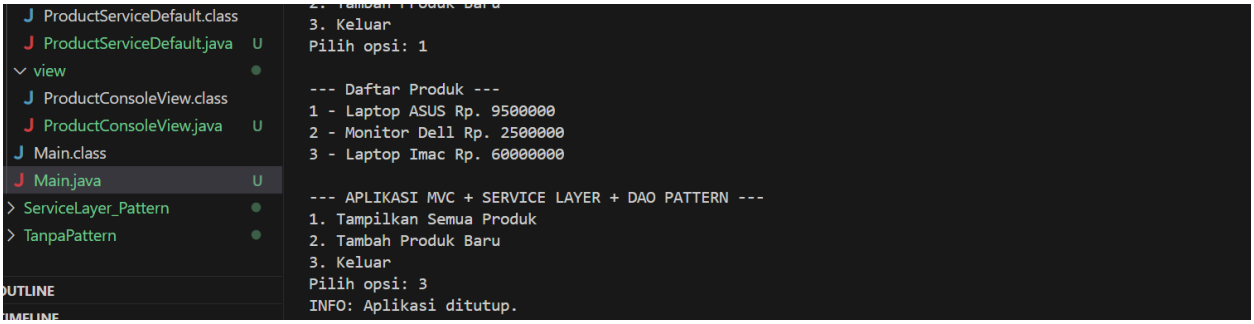
8) Main.java

LayeredArchitecturePattern > MVC_ServiceLayer_DAO > Main.java > ...

```java
package MVC_ServiceLayer_DAO;

import java.util.Scanner;

import MVC_ServiceLayer_DAO.controller.ProductController;
import MVC_ServiceLayer_DAO.dao.ProductDao;
import MVC_ServiceLayer_DAO.dao.memory.ProductDaoInMemory;
import MVC_ServiceLayer_DAO.service.ProductService;
import MVC_ServiceLayer_DAO.service.ProductServiceDefault;
import MVC_ServiceLayer_DAO.view.ProductConsoleView;

public class Main {
    Run | Debug
    public static void main(String[] args) {

        // Layer DAO
        ProductDao dao = new ProductDaoInMemory();

        // Layer Service
        ProductService service = new ProductServiceDefault(dao);

        // Layer View
        Scanner scanner = new Scanner(System.in);
        ProductConsoleView view = new ProductConsoleView(scanner);

        // Layer Controller
        ProductController controller = new ProductController(service, view);

        // Loop Menu Utama
        // Loop Menu Utama
        while (true) {
            view.displayMenu();
            String pilihanMenu = view.getPilihanMenuFromUser();
            controller.handlingPilihanMenu(pilihanMenu);
        }
    }
}
```

d) Bukti Running


```
--- 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: 60000000
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. 60000000

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


ProductServiceDefault.class
ProductServiceDefault.java U
view
ProductConsoleView.class
ProductConsoleView.java U
Main.class
Main.java U
ServiceLayer_Pattern
TanpaPattern
OUTLINE
3. Keluar
Pilih opsi: 1
--- Daftar Produk ---
1 - Laptop ASUS Rp. 9500000
2 - Monitor Dell Rp. 2500000
3 - Laptop Imac Rp. 60000000
--- APLIKASI MVC + SERVICE LAYER + DAO PATTERN ---
1. Tampilkan Semua Produk
2. Tambah Produk Baru
3. Keluar
Pilih opsi: 3
INFO: Aplikasi ditutup.