



A. IDENTITAS

NIM : 2403001
Nama Lengkap : Siti Sa'adah
Kelas : D3 TI 2A
Program Studi : D3 TEKNIK INFORMATIKA
Jurusan : TEKNIK INFORMATIKA

B. HASIL PRAKTIKUM/ TUGAS

1. SRP → Setiap Kelas hanya boleh memiliki satu tanggung jawab utama, atau satu tujuan perubahan.
2. Code Program SRP

Case Study - 1 Single Responsibility Principle

Buatlah PROGRAM
perhitungan luas persegi
dengan menerapkan
prinsip Single
Responsibility!



a. Persegi.java

```
package Srp;

public class Persegi {
    private int sisi;

    public Persegi(int sisi) {
        this.sisi = sisi;
    }

    public int getSisi() {
        return sisi;
    }
}
```



b. PenghitungPersegi.java

```
package Srp;

public class PenghitungPersegi {

    public int hitungLuas(Persegi persegi) {
        return persegi.getSisi() * persegi.getSisi();
    }
}
```



c. Main.java

```
package Srp;

import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);

        System.out.print("Masukkan panjang sisi persegi: ");
        int sisi = input.nextInt();

        Persegi persegi = new Persegi(sisi);
        PenghitungPersegi penghitung = new PenghitungPersegi();

        int luas = penghitung.hitungLuas(persegi);
        System.out.println("Luas persegi = " + luas);

        input.close();
    }
}
```

d. Run

```
D:\Tisa\Tisa\solid-principle>javac Srp/*.java

D:\Tisa\Tisa\solid-principle>java Srp/Main.java
Masukkan panjang sisi persegi: 13
Luas persegi = 169

D:\Tisa\Tisa\solid-principle>
```



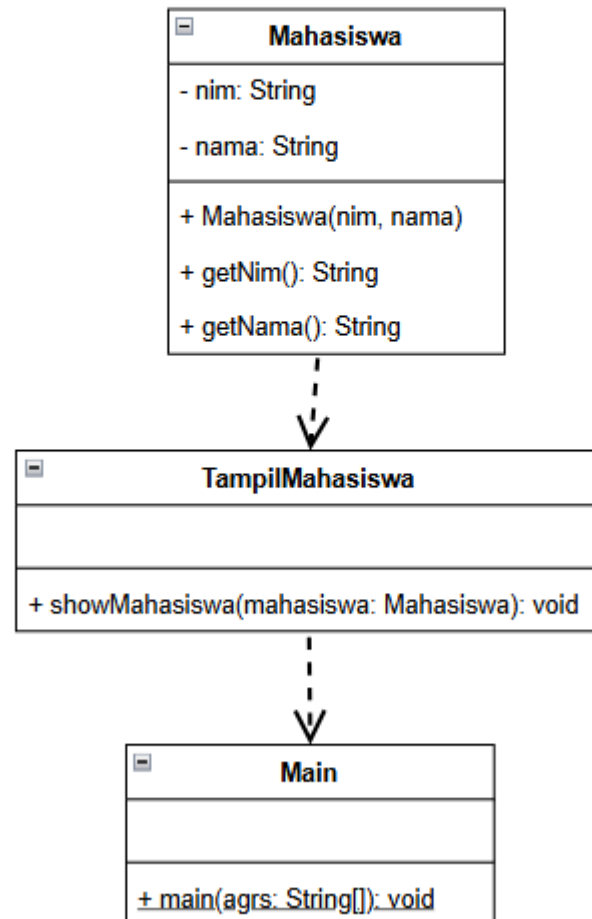
3. Class Diagram SRP

Case Study - 2 Single Responsibility Principle

Buatlah CLASS DIAGRAM
dari studi kasus
pengelolaan data
mahasiswa dengan
menerapkan prinsip Single
Responsibility!



DEPEDENCY?





4. OCP → dapat ditambah fungsi baru tanpa mengubah yang lama
5. Code Program OCP

Case Study - 1

Open Closed Principle

Buatlah PROGRAM
**pemilihan metode
pembayaran customer**
dengan menerapkan
prinsip Open closed!

27

- a. TipePembayaran.java

```
package Ocp;

public interface TipePembayaran {
    void memprosesPembayaran();
}
```



b. Debit.java

```
package Ocp;

public class Debit implements TipePembayaran {
    @Override
    public void memprosesPembayaran() {
        System.out.println("Memproses pembayaran menggunakan kartu Debit...");
    }
}
```

c. Cash.java

```
package Ocp;

public class Cash implements TipePembayaran {
    @Override
    public void memprosesPembayaran() {
        System.out.println("Memproses pembayaran menggunakan uang Cash...");
    }
}
```

d. Kredit.java

```
package Ocp;

public class Kredit implements TipePembayaran {
    @Override
    public void memprosesPembayaran() {
        System.out.println("Memproses pembayaran menggunakan kartu Kredit...");
    }
}
```




e. PembayaranCustomer.java

```
package Ocp;

public class PembayaranCustomer {

    public void menerimaPembayaran(TipePembayaran tipe) {
        tipe.memprosesPembayaran();
    }
}
```



f. Main.java

```
package Ocp;

import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        PembayaranCustomer customer = new PembayaranCustomer();

        System.out.println("=== Pilih Metode Pembayaran ===");
        System.out.println("1. Debit");
        System.out.println("2. Kredit");
        System.out.println("3. Cash");
        System.out.print("Pilihan Anda: ");
        int pilihan = input.nextInt();

        TipePembayaran tipe = null;

        switch (pilihan) {
            case 1:
                tipe = new Debit();
                break;
            case 2:
                tipe = new Kredit();
                break;
            case 3:
                tipe = new Cash();
                break;
            default:
                System.out.println("Pilihan tidak valid!");
                System.exit(0);
        }

        customer.menerimaPembayaran(tipe);
        input.close();
    }
}
```



g. Run

```
D:\Tisa\Tisa\solid-principle>javac Ocp/*.java
```

```
D:\Tisa\Tisa\solid-principle>java Ocp/Main.java
```

```
=== Pilih Metode Pembayaran ===
```

1. Debit
2. Kredit
3. Cash

```
Pilihan Anda: 1
```

```
Memproses pembayaran menggunakan kartu Debit...
```

```
D:\Tisa\Tisa\solid-principle>java Ocp/Main.java
```

```
=== Pilih Metode Pembayaran ===
```

1. Debit
2. Kredit
3. Cash

```
Pilihan Anda: 2
```

```
Memproses pembayaran menggunakan kartu Kredit...
```

```
D:\Tisa\Tisa\solid-principle>java Ocp/Main.java
```

```
=== Pilih Metode Pembayaran ===
```

1. Debit
2. Kredit
3. Cash

```
Pilihan Anda: 3
```

```
Memproses pembayaran menggunakan uang Cash...
```

```
D:\Tisa\Tisa\solid-principle>java Ocp/Main.java
```

```
=== Pilih Metode Pembayaran ===
```

1. Debit
2. Kredit
3. Cash

```
Pilihan Anda: 4
```

```
Pilihan tidak valid!
```

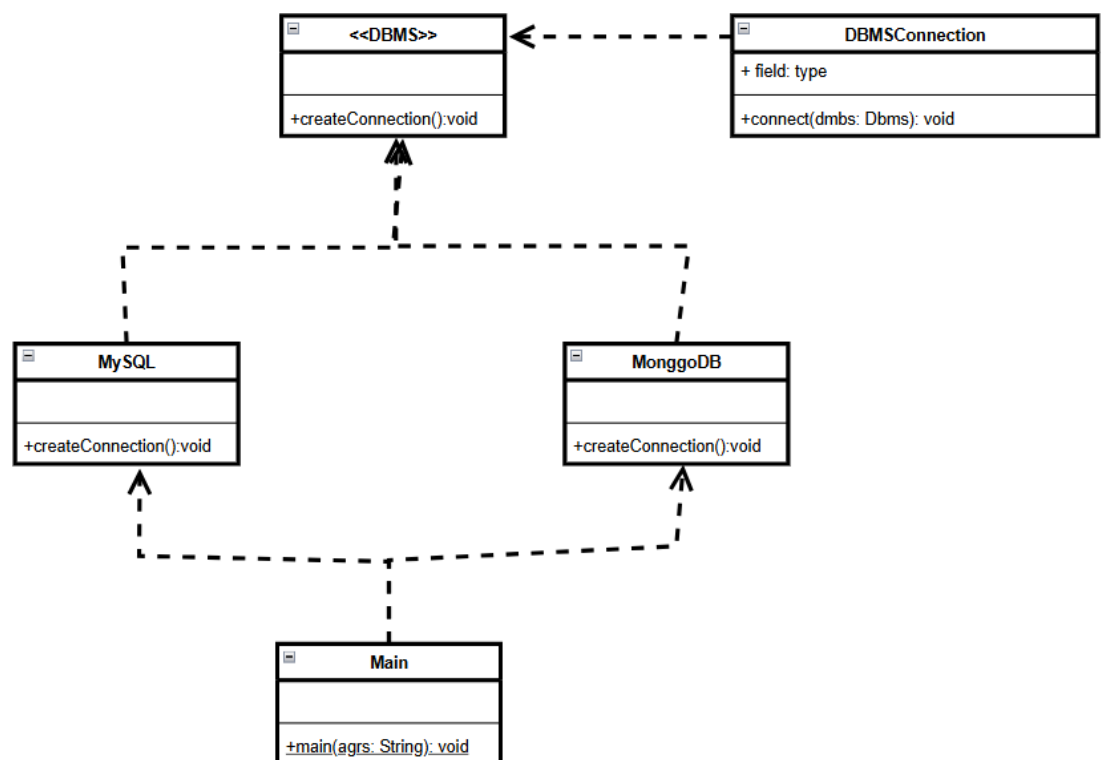


6. Class Diagram OCP

Case Study - 2 Open Closed Principle

Buatlah CLASS DIAGRAM
dari studi kasus **koneksi
DBMS** dengan
menerapkan prinsip Open
closed!

30





7. LSP → Objek dari subclass harus dapat menggantikan superclass – nya, tanpa mengubah program nya.
8. Kode Program LSP

Case Study - 1

Liskov Substitution Principle

Buatlah PROGRAM pewarisan fitur aplikasi **sosial media** dengan menerapkan prinsip Liskov substitution!

42

- a. SosialMedia.java

```
package Lsp;

public abstract class SocialMedia {
    abstract void chat();
    abstract void sendPhotosAndVideos();
}
```



b. PostMediaManager.java



```
package Lsp;  
  
abstract class PostMediaManager extends SocialMedia {  
    abstract void publishPost();  
}
```

c. VideoGroupManager.java



```
package Lsp;  
  
abstract class VideoGroupManager extends SocialMedia {  
    abstract void callGroupVideo();  
}
```



d. WhatsApp.java

```
package Lsp;

class WhatsApp extends VideoGroupManager {

    @Override
    void chat() {
        System.out.println("Mengirim pesan via WhatsApp...");
    }

    @Override
    void sendPhotosAndVideos() {
        System.out.println("Mengirim foto dan video di WhatsApp...");
    }

    @Override
    void callGroupVideo() {
        System.out.println("Melakukan panggilan video grup di WhatsApp...");
    }
}
```

e. Instagram.java

```
package Lsp;

class Instagram extends PostMediaManager {

    @Override
    void chat() {
        System.out.println("Mengirim pesan via Instagram DM...");
    }

    @Override
    void sendPhotosAndVideos() {
        System.out.println("Mengirim foto/video story atau DM di Instagram...");
    }

    @Override
    void publishPost() {
        System.out.println("Mempublikasikan post baru di Instagram...");
    }
}
```



f. Main.java

```
package Lsp;

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

        SocialMedia wa = new WhatsApp();
        SocialMedia ig = new Instagram();

        System.out.println("=== WhatsApp ===");
        wa.chat();
        wa.sendPhotosAndVideos();
        if (wa instanceof VideoGroupManager) {
            ((VideoGroupManager) wa).callGroupVideo();
        }

        System.out.println("\n=== Instagram ===");
        ig.chat();
        ig.sendPhotosAndVideos();
        if (ig instanceof PostMediaManager) {
            ((PostMediaManager) ig).publishPost();
        }
    }
}
```




g. Run

```
D:\Tisa\Tisa\solid-principle>javac Lsp/*.java

D:\Tisa\Tisa\solid-principle>java Lsp/Main.java
=== WhatsApp ===
Mengirim pesan via WhatsApp...
Mengirim foto dan video di WhatsApp...
Mengirim foto dan video di WhatsApp...
Melakukan panggilan video grup di WhatsApp...

=== Instagram ===
Mengirim pesan via Instagram DM...
Mengirim foto/video story atau DM di Instagram...
Mempublikasikan post baru di Instagram...

D:\Tisa\Tisa\solid-principle>
```

Ln 7, Col 1



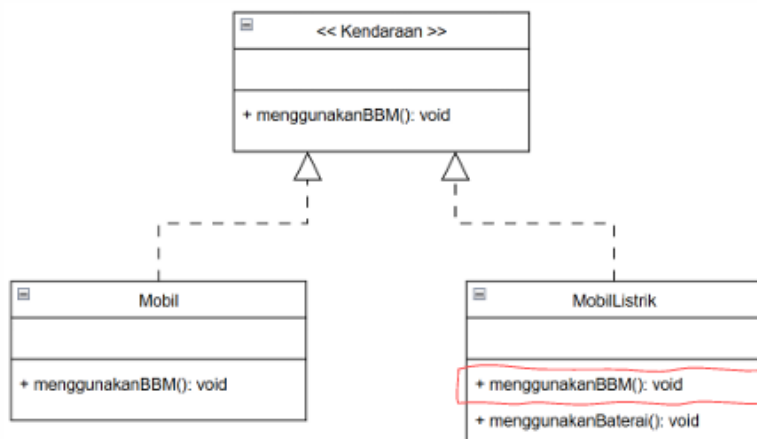
9. Class Diagram LSP

Case Study - 2 Liskov Substitution Principle

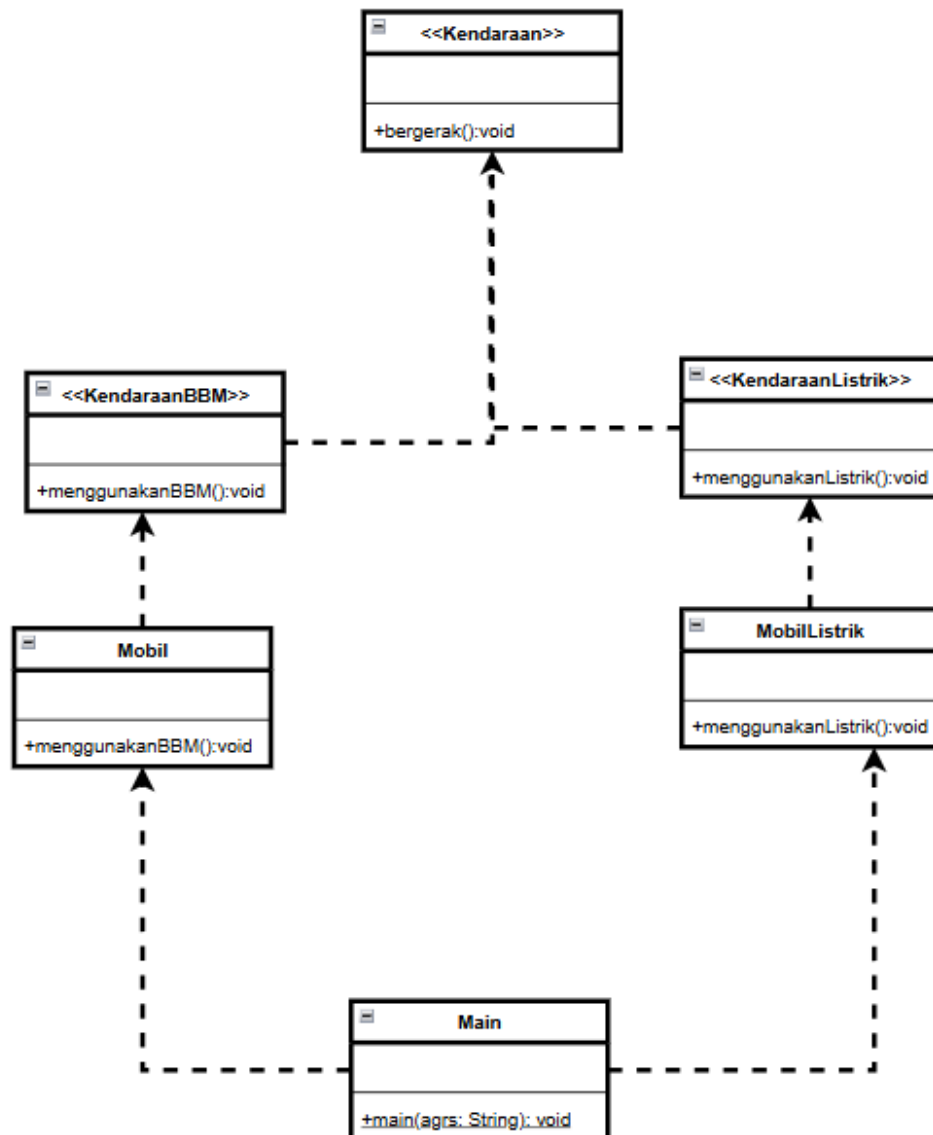
Buatlah CLASS DIAGRAM dari studi kasus **pewarisan sifat kendaraan** dengan menerapkan prinsip Liskov substitution!

45

Liskov Substitution - **Case Study 2 (BAD)**



46





10. ISP → Tidak membuat satu interface besar, yang mengakibatkan pemaksaan terhadap implementasi method yang tidak relevan.

11. Kode Program ISP

Case Study - 1

Interface Segregation Principle

Buatlah PROGRAM perhitungan luas kubus dan persegi serta volume kubus dengan menerapkan prinsip Interface segregation!

63

a. Bentuk3Dimensi.java

```
package Isp;

public interface Bentuk3Dimensi {
    double hitungVolume();
}
```



b. Bentuk2Dimensi.java

```
package Isp;

public interface Bentuk2Dimensi {
    double hitungLuas();
}
```

c. Kubus.java

```
package Isp;

public class Kubus implements Bentuk2Dimensi, Bentuk3Dimensi {
    private double sisi;

    public Kubus(double sisi) {
        this.sisi = sisi;
    }

    @Override
    public double hitungLuas() {
        return 6 * sisi * sisi;
    }

    @Override
    public double hitungVolume() {
        return sisi * sisi * sisi;
    }
}
```



d. Persegi.java

```
package Isp;

public class Persegi implements Bentuk2Dimensi {
    private double sisi;

    public Persegi(double sisi) {
        this.sisi = sisi;
    }

    @Override
    public double hitungLuas() {
        return sisi * sisi;
    }
}
```

e. Main.java

```
package Isp;

public class Main {
    public static void main(String[] args) {
        Persegi persegi = new Persegi(5);
        Kubus kubus = new Kubus(5);

        System.out.println("=== Program Perhitungan Bangun ===");
        System.out.println("Luas Persegi: " + persegi.hitungLuas());
        System.out.println("Luas Permukaan Kubus: " + kubus.hitungLuas());
        System.out.println("Volume Kubus: " + kubus.hitungVolume());
    }
}
```



f. Run

```
D:\Tisa\Tisa\solid-principle>javac Isp/*.java

D:\Tisa\Tisa\solid-principle>java Isp/Main.java
=== Program Perhitungan Bangun ===
Luas Persegi: 25.0
Luas Permukaan Kubus: 150.0
Volume Kubus: 125.0

D:\Tisa\Tisa\solid-principle>
```



12. Class Diagram ISP

Case Study - 2 Interface Segregation Principle

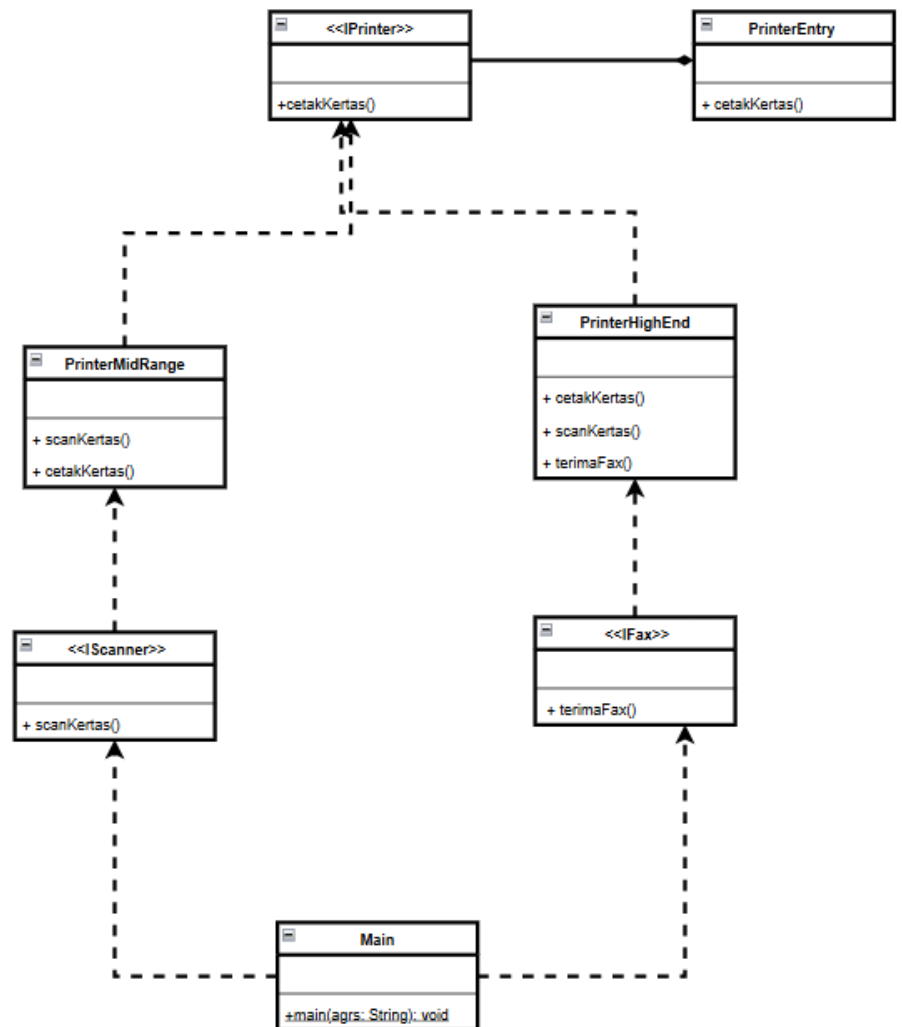
Buatlah CLASS DIAGRAM
dari studi kasus
pembagian fitur printer
dengan menerapkan
prinsip Interface
segregation!

66

Interface Segregation - Case Study 2 (BAD)



67





13. DIP → Class tingkat tinggi tidak boleh bergantung pada class konkret, tapi pada interface.

14. Code Program DIP

Case Study - 1

Dependency Inversion Principle

Buatlah PROGRAM
Konektivitas Multi DBMS
dengan menerapkan
prinsip Dependency
inversion!

82

a. Database.java

```
package Dib;

public interface Database {
    void insert(String data);
}
```



b. MySQL.java

```
package Dib;

public class MySQL implements Database {
    @Override
    public void insert(String data) {
        System.out.println("Data '" + data + "' berhasil dimasukkan ke database MySQL.");
    }
}
```

c. PostgreSQL.java

```
package Dib;

public class PostgreSQL implements Database {
    @Override
    public void insert(String data) {
        System.out.println("Data '" + data + "' berhasil dimasukkan ke database PostgreSQL.");
    }
}
```



d. UserService.java

```
package Dib;

public class UserService {
    private Database database;
    public UserService(Database database) {
        this.database = database;
    }
    public void daftarPengguna(String namaPengguna) {
        System.out.println("Mendaftarkan pengguna: " + namaPengguna);
        database.insert(namaPengguna);
    }
}
```

e. Main.java

```
package Dib;
public class Main {
    public static void main(String[] args) {
        Database mysql = new MySQL();
        UserService layananMySQL = new UserService(mysql);
        layananMySQL.daftarPengguna("Kudo Shinichi");
        System.out.println();
        Database postgre = new PostgreSQL();
        UserService layananPostgre = new UserService(postgre);
        layananPostgre.daftarPengguna("Kuroba Kaito");
    }
}
```



f. Run

```
D:\Tisa\Tisa\solid-principle>javac Dib/*.java

D:\Tisa\Tisa\solid-principle>java Dib/Main.java
Mendaftarkan pengguna: Kudo Shinichi
Data 'Kudo Shinichi' berhasil dimasukkan ke database MySQL.

Mendaftarkan pengguna: Kuroba Kaito
Data 'Kuroba Kaito' berhasil dimasukkan ke database PostgreSQL.

D:\Tisa\Tisa\solid-principle>
```



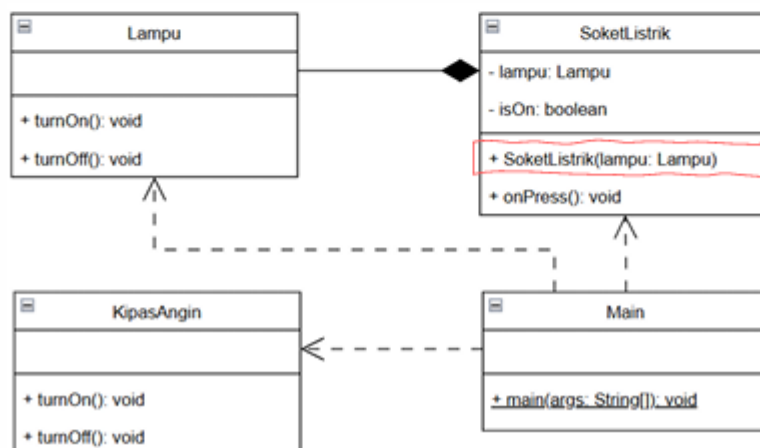
15. Class Diagram DIP

Case Study - 2 Dependency Inversion Principle

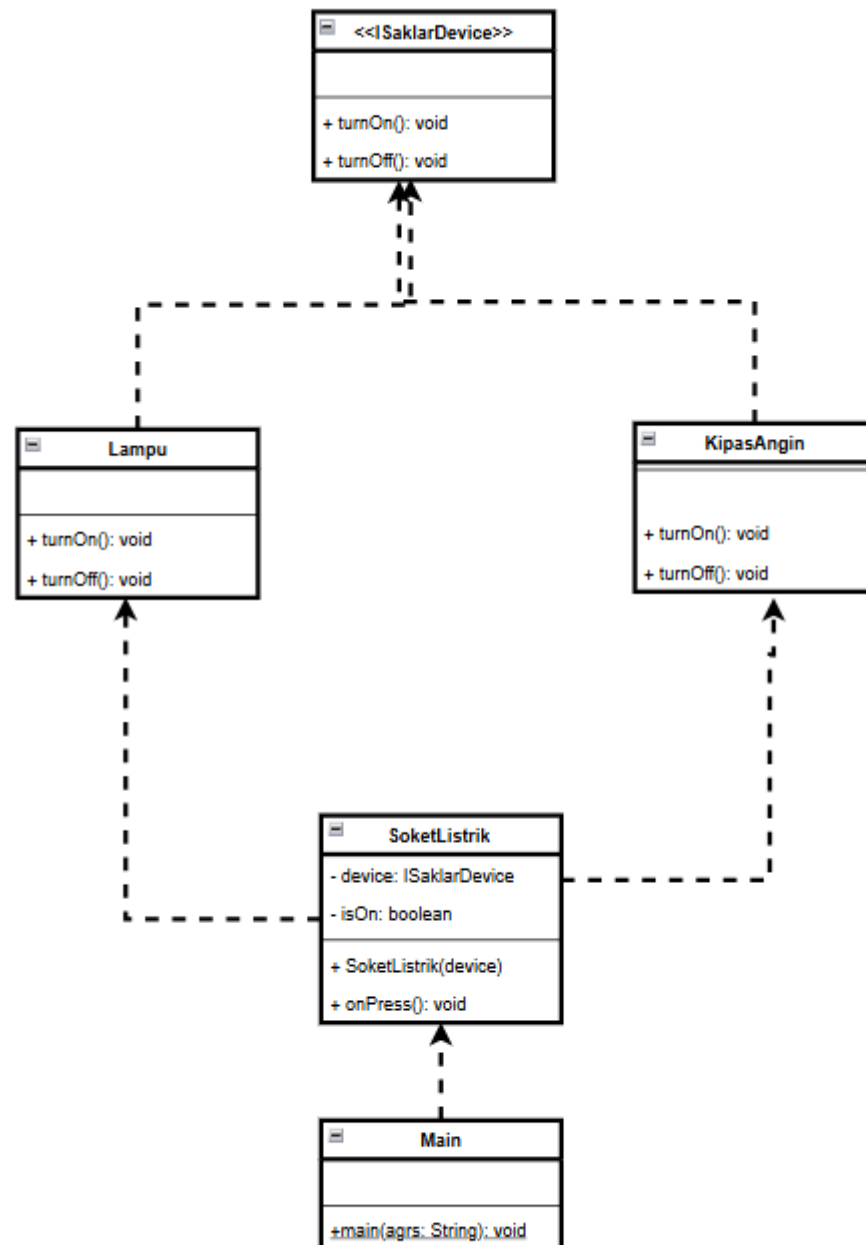
Buatlah CLASS DIAGRAM
dari studi kasus **Saklar
Lampu dan Saklar Kipas
Angin** dengan
menerapkan prinsip
Dependency inversion!

85

Dependency Inversion - **Case Study 2 (BAD)**



86





16. Kesimpulan

- a. S → Memisahkan Tanggung Jawab
- b. O → Hindari Modifikasi Berulang
- c. L → Pewarisan Harus Logis
- d. I → Interface Kecil dan Spesifik
- e. D → Gunakan Abstraksi

17. Link Repository

https://github.com/Tisa-34/solid-principle_PBO