Basic Concepts

1. Contoh code smell yang sering terjadi di OOP yaitu shotgun surgery. Shotgun surgery merupakan suatu keadaan dimana apabila developer ingin mengubah sedikit pada suatu bagian, ia harus mengganti banyak class dan method dan kebanyakan harus membuat code yang sama. Shotgun surgery terjadi karena:

1. proses separasi yang buruk.
2. gagal dalam memahami prinsip single responsibility.
3. tidak mengidentifikasi common behavior.
4. Gagal dalam memperkenalkan design pattern yang baik.

contoh Shotgun Surgery:

public void debit(int debit) throws Exception

{

if(amount <= 500)

{

throw new Exception("Mininum balance shuold be over 500");

}

amount = amount-debit;

System.out.println("Now amount is" + amount);

}

public void transfer(Account from,Account to,int cerditAmount) throws Exception

{

if(from.amount <= 500)

{

throw new Exception("Mininum balance shuold be over 500");

}

to.amount = amount+cerditAmount;

}

public void sendWarningMessage()

{

if(amount <= 500)

{

System.out.println("amount should be over 500");

}

}

Jika kita perhatikan setiap operasi debit(), transfer(), sendWarningMessage() memiliki 1 validasi yaitu: account balance harus melebihi 500. Kita mengcopy validasi yang sama pada setiap metode karena kita tidak dapat mengidentifikasi common validation, sehingga menghasilkan shotgun surgery code smell. Permasalahan muncul ketika kita menambah kriteria validasinya misalnya: Jika tipe account personal dan balance harus melebihi 500, maka kita bisa melakukan operasi diatas. Pada skenario ini, kita harus mengubah seluruh metode dimana kita bisa mengatasi dengan membuat common method yaitu isAccount Underflow() dimana semua validasi akan ditaruh disana.

private boolean isAccountUnderflow()

{

return amount<=500;

}

public void debit(int debit) throws Exception

{

if(isAccountUnderflow())

{

throw new Exception("Mininum balance shuold be over 500");

}

amount = amount-debit;

System.out.println("Now amount is" + amount);

}

public void transfer(AcountRefactored from,AcountRefactored to,int cerditAmount) throws Exception

{

if(isAccountUnderflow())

{

throw new Exception("Mininum balance shuold be over 500");

}

to.amount = amount+cerditAmount;

}

public void sendWarningMessage()

{

if(isAccountUnderflow())

{

System.out.println("amount should be over 500");

}

}

}

2.

Get

* Get request dapat digunakan pada bookmark, cache
* Get request tidak dianjurkan pada data yang bersifat sensitif

Post

* Post request tidak memiliki halangan(*restriction*) maupun panjang data
* Post request tidak dapat dibookmark dan tidak pernah dicache