# Feedback voor Groep 5

**Notitie:**

Tijdens het schrijven van deze feedback is de code op SonarCloud zeer verouderd.

Van alle projecten behalve BifiData is de scan 3 weken oud.

Hierom hebben wij SonarLint ter ondersteuning gebruikt.

### BifiData

DAO interface:

* DAO<T> is niet descriptive.

Deze klassen moeten een logger gebruiken i.p.v. een exceptie afvangen met e.printStackTrace():

* AbstractGenericDAO.java
* HibernateUtil.java
* KlantDAOImpl.java

De conventie is om variabele namen met een kleine letter te beginnen. Dit klopt nog niet in:

* AdresID in AdresDTO.java
* Vrijwel elk veld in KlantDTO.java

HibernateUtil.java moet een private constructor krijgen want je mag deze klas niet instantiëren.

Er zijn twee verantwoordelijkheden in dit project: Data uit database ophalen, en data bewaren in een model. De code zou overzichtelijker zijn, als deze verantwoordelijkheden duidelijk gescheiden zijn. Één manier om dit te doen, is om een package voor iedere verantwoordelijkheid te hebben. Binnen de "data uit database halen" package zet je de verbinding met Hibernate en MongoDB, en alle functies die die verbindingen aanroepen.

Een test moet falen om 1 reden. Daarom is het is een good practice om zo min mogelijk (1) asserts in je test te hebben. Dit dwingt je om meer testen te schrijven. Zie bijvoorbeeld ValidationTestUtils.java.

### BifiUI

GetFactuur() heeft geen functionaliteit.

De conventie is om methode namen met een kleine letter te beginnen. Dit geldt ook voor packages.

Hardcoded code in GetFactuur(), je krijgt nu altijd maandnummer 1 en jaar 1.

In CommandInterpreter.java wordt een TerminalScreen gemaakt, maar deze wordt nooit ge-closed. Dit kan het beste in een "finally".

### FactuurService

KlantController.java en FactuurContoller.java van project FactuurService gebruiken objecten uit project BifiData. De twee projecten herkennen elkaar niet, en hierdoor worden objecten uit BifiData niet herkend. Om dit op te lossen, kan je de dependency opnemen via Maven (lokaal of via de cloud).

SonarLint geeft ongeveer 15 keer aan in FactuurControllerTest.java dat er beter een constante kan worden gebruikt dan code duplication.

Ook hier is het handig om te streven naar zo min mogelijk asserts. Zo heeft testPersoonFromKlantConversion() 7 asserts. Wel heel goed dat er overal een duidelijke message bij staat!

Ik zie dat er IDE-specifieke bestanden zoals .classpath aanwezig zijn. Ik neem aan dat dit niet mee moet worden gestuurd. Hier kan je bijvoorbeeld een .gitignore bestand voor gebruiken.

Bij resources.hibernate.cfg.xml staan alleen gegevens zoals username en password. Je kan hier config files voor gebruiken (of systeemvariabelen), zodat de gegevens niet voor andere mensen zichtbaar zijn. Ook deze kan je toevoegen aan de .gitignore.

De structuur bij src.main.java.nl… is vrij onduidelijk. Domeinklassen (die ook bij BifiData.java staan) en controllers staan door elkaar.

### IEFService

In de tests van dit project worden engelse en nederlandse methodenamen gebruikt. Het is wenselijk om consistente methodenamen te hebben. Gebruik desnoods conventies.

Verwijder commentaar zoals in IEFservice.java.

Vrij veel tests hebben geen assert(), dus ze testen niets. Er zijn wel veel tests geschreven in losse testmethoden. Netjes.

Gebruik constanten i.p.v. telkens hetzelfde te typen, zoals result in KlantTest.java (wordt 9x gebruikt en opnieuw aangemaakt).

Ook wordt er vaak een final variabele aangemaakt, bijvoorbeeld expectedResult. Die wordt dan één of twee zinnen later gebruikt in een assert. Hier is het duidelijker om direct expectedResult als parameter mee te geven i.p.v. het eerst toewijzen.

Er ontbreken een aantal tests. Bijvoorbeeld de test van zetKlantOmNaarIefKlantStructuurInBifiFormaat() in IEFBiFiConverter.java.