OAK 3 ACADEMY	**Datum**: 25/09/2017 **Opleiding**: Java Developer **Lesmodule**: JAVA SE 8
	Naam: .....

Instructies:

De bedoeling is dat je deze test volledig zelfstandig kunt uitvoeren, alle geziene leerstof gebruiken en combineren in één project.

Volgende leerstof komt aan bod:

Java Fundamentals
GIT
Maven
JUnit
MySQL – SQL
JDBC

Afspraak:

Voorziene tijd: 4 dagen
Als je klaar bent, stuur je de link door van jouw git repository.

IntelliJ Maven Project:

Naam: E-shop
Volgende packages maak je zeker aan:

- **be.vdab.entiteiten**: hierin komen al je klassen (zie Opdracht 1);
- **be.vdab.dao:** hierin komen al je Data Access Objects;
- **be.vdab.dao.impl**: hierin komen al je implementaties van je dao's.
- **be.vdab.ui**: hierin komt alles wat je nodig hebt voor de User Interface van je project.

Andere nuttige packages mogen aangemaakt worden, wanneer je deze nodig acht.

Aandachtspunten:

- Geen dode code + geen code in commentaar achterlaten;
- Je project moet runnen zonder foutmeldingen;
- Formateer je code (CTRL + ALT + L) in elke klasse;
- Houd rekening met de Naming Conventions van Java. Gebruik ook goede/duidelijke naamgevingen voor je variabelen – methoden …
- **Schrijf Unit testen waar nodig;**
- Gebruik maken van andere libraries wordt gestimuleerd;
- Maak een remote repository aan voor dit project waar je regelmatig je code update.
- **Deze opdracht gaan we later ook uitwerken via een webapplicatie, zorg er dus voor dat je volledige backend herbruikbaar is.**

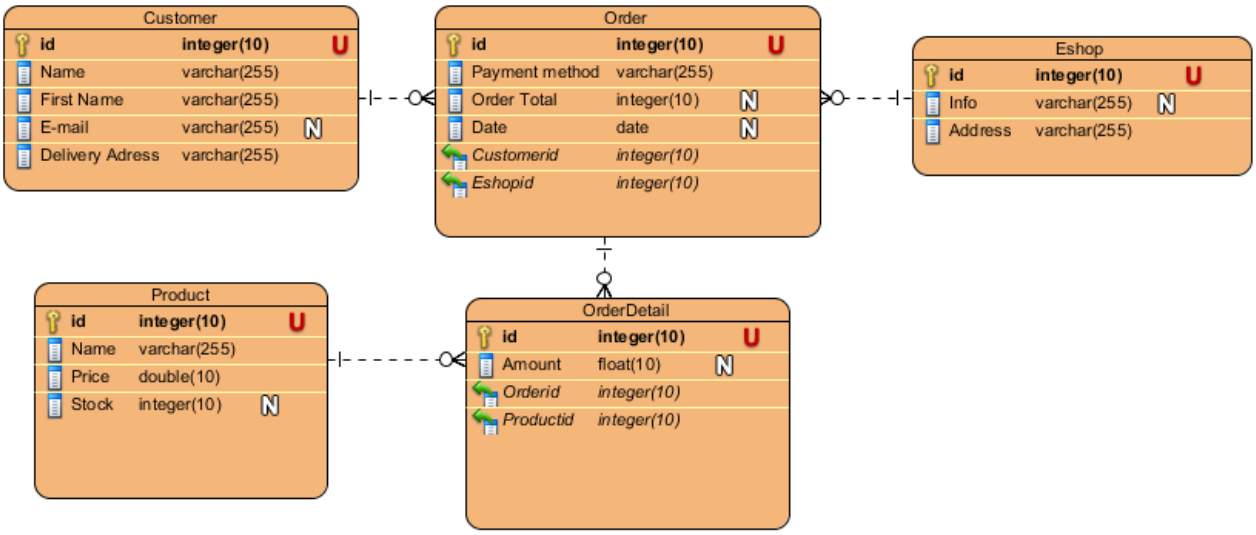
Opdrachten:

Voor deze opdracht maak je een prototype voor een online shop. De uitwerking zal beperkt worden tot het winkelmandje.

- We voorzien een model waarbij een klant bestellingen kan hebben, elke order bevat één of meer producten. Voor elke order moet je kunnen bijhouden hoeveel we van elke product willen bestellen, en het totaalbedrag (product prijs maal het aantal).
- Elke bestelling is gelinkt aan een klant, een klant kan geen of meer bestellingen hebben. Een order is ook gelinkt aan een shop.
- Het idee is dat verschillende shops kunnen aangemaakt worden om gebruik te maken van dit model. Alle shops delen dezelfde klanten, de bestellingen zijn gelinkt aan één shop. Daarom voorzien we geen directe relatie tussen klant en shop.

Opdracht1: Klassen aanmaken + database

Maak onderstaande tabellen met de nodige relaties aan in MySQL:



Voorzie een minimum aan test data. Eén shop, één customer, en twee producten.

Maak nu in Java de nodige entiteiten aan voor de net aangemaakte tabellen om optimaal te kunnen werken. De klant moet kunnen inloggen op je applicatie met een gebruikersnaam en een wachtwoord. Maak hiervoor een klasse User aan.
Denk eerst goed na:

- Welke entiteiten heb je nodig?
- Kan ik gebruik maken van superklassen/ subklassen/ Interfaces …?
- Welke extra's heb ik nodig?

Opdracht2: Data access functionaliteiten

De gegeven functionaliteiten per DAO zijn een minimum - indien nodig kan je uitbreiden - implementeer de nodige functionaliteiten.
Voorzie de volgende DAO's:

ShopDao

List<Eshop> listAllShops();

CustomerDao

Customer findCustomers(String name, String firstname, String username);

User findByLoginAndUsername (String username, String password);

ProductDao

List<Product> findProducts(String productname);

OrderDao

List<Order> findOrdersForCustomer(Customer customer);

void saveOrder(Order order);

BasketDao

void saveOrUpdateBasket(Basket basket);

void addProductToBasket(Product product)

void removeProductFromBasket(Product product)

Basket getBasket();

void clearBasket();

Opdracht3: Gebruiksvriendelijkheid

Een gebruiker kan op elk moment zijn winkelmandje raadplegen. Voorzie een link voor de gebruiker die op elk ogenblik zijn winkelmandje kan bekijken. Als het winkelmandje leeg is toon je een gebruiksvriendelijke tekst anders de gekozen producten. De gebruiker kan altijd kiezen om verder te winkelen of om af te rekenen.

Opdracht4: User Interface

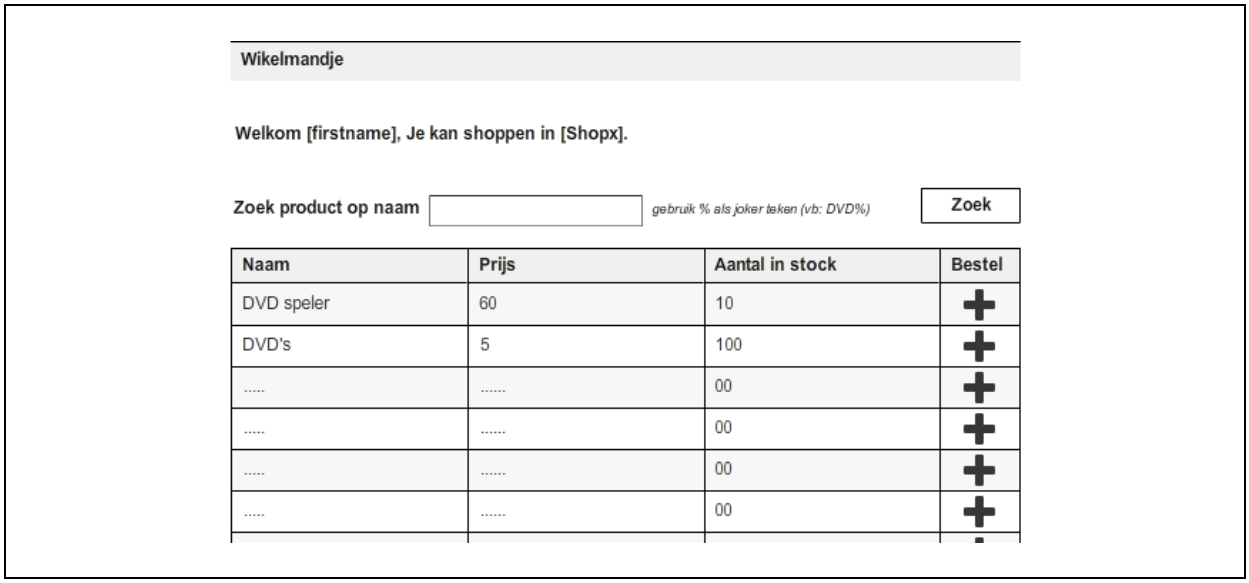
DIT IS ONDERGESCHIKT TEN OPZICHTE VAN DE VORIGE OPDRACHTEN. ZORG EERST EN VOORAL DAT ALLE BOVENSTAANDE KLASSEN enz… In ORDE ZIJN!

Maak een prototype swing applicatie als presentatie laag. De gebruikers krijgt voor de eerste keer alle producten te zien. De gebruiker heeft de mogelijkheid om te zoeken in de lijst van producten en eventueel een joker teken te gebruiken. Indien niets in het zoekveld wordt ingevoerd geef je alle producten terug.
Wanneer een klant een product kiest wordt deze in het winkelmandje gestopt. Er wordt telkens maar één product toegevoegd. Bij het kiezen om af te rekenen worden de producten uit winkelmandje bewaard in de database en wordt het winkelmandje leeg gemaakt. Toon een gebruiksvriendelijke bericht aan de gebruiker dat de order is afgehandeld.
Voor de architectuur houd je rekening met volgende zaken:

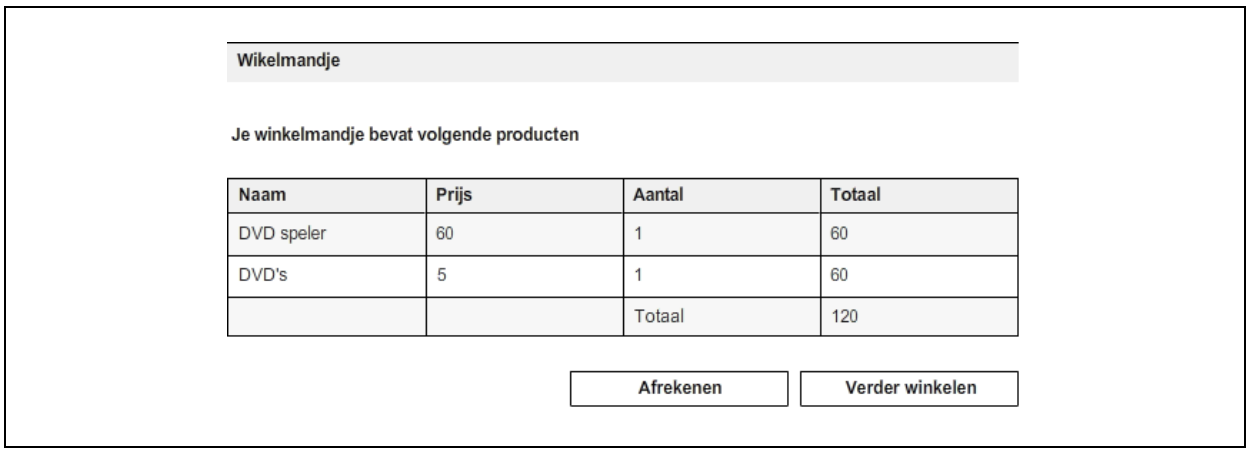
- Presentatie laag heeft weet van Business laag maar kan nooit rechtstreeks de Data laag aanspreken.
- In de business laag voorzie je GEEN data access maar gebruik je de Data laag die de nodige data access functionaliteiten voorziet.

Volgende schermindelingen zijn niet verplicht en dienen enkel om de vraag te verduidelijken

Shop keuze scherm



Product scherm



Winkelmandje