VDAB

samen sterk voor werk

JPA met Hibernate Takenbundel

Inhoudsopgave

1 TAKEN

1.1 Alles voor de keuken

Je voert het script `allesvoordekeuken.sql` uit.
Dit script maakt een database `allesvoordekeuken`.
Je kan deze database openen met de gebruiker `cursist`, paswoord `cursist`

Deze database bevat een lege table `artikels`
Je voegt met de MySQL Workbench enkele records (uit de categorieën `fruit`, `groente`, `keukenapparaten`) toe aan deze table (later komt er een gerelateerde table bij met deze categorieën)

Je maakt een website.

- Je definieert een connection pool in `context.xml`
- Je configureert JPA in `persistence.xml`
- Je maakt een filter die JPA initialiseert
- De beginpagina bevat een menu met volgende hyperlinks
  - Artikel zoeken op nummer (leidt naar `/artikels/zoekenopnummer.htm`)
  - Artikel toevoegen (leidt naar `/artikels/toevoegen.htm`)
  - Artikels zoeken op naam (leidt naar `/artikels/zoekenopnaam.htm`)
  - Prijsverhoging (leidt naar `/artikels/prijsverhoging.htm`)
  - Artikelkortingen (leidt naar `/artikels/kortingen.htm`)
  - Artikels per artikelgroep (leidt naar `/artikels/perartikelgroep.htm`)
- Je toont onder dit menu at random één van de logo afbeeldingen uit het takenmateriaal.

1.2 Artikel

Je maakt een entity class `Artikel`, die bij de table `artikels` hoort.

1.3 Zoeken op nummer

Je maakt een pagina waarin de gebruiker een artikelnummer intikt.
Je zoekt dit artikel en toont de naam, aankoopprijs, verkoopprijs en % winst (verkoopprijs – aankoopprijs) / aankoopprijs * 100

1.4 JPA project

Maak van je project een JPA project

1.5 Toevoegen

Je maakt een pagina om een artikel toe te voegen.

- De aankoopprijs moet minstens 0.01 zijn.
- De verkoopprijs kan niet kleiner zijn dan de aankoopprijs.

1.6 ThreadLocal

Je beheert de EntityManager met een ThreadLocal variabele.

Je leidt je repository class af van een base class AbstractRepository.

1.7 Zoeken op naam

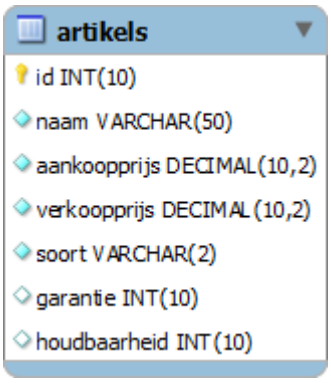
De gebruiker tikt een woord. Je toont de artikels die dit woord hebben in hun naam.
Je sorteert de artikels op naam.
Je toont per artikel het nummer, de naam, de aankoopprijs en de verkoopprijs

1.8 Prijsverhoging

De gebruiker tikt een percentage.
Je verhoogt de verkoopprijs van alle artikels met dit percentage.

1.9 Food en Non-Food artikels

Je voert het script *FoodNonFood.sql* uit.
Dit script voegt aan de table `artikels` drie kolommen toe



- `soort`
  bevat `F` bij food artikels
  bevat `NF` bij non-food artikels
- `garantie`
  blijft leeg bij food artikels
  bevat het aantal maanden garantie bij non-food artikels
- `houdbaarheid`
  bevat de houdbaarheid (in dagen) bij food artikels.
  blijft leeg bij non-food artikels

Je doet via de `MySQL Workbench` volgende aanpassingen

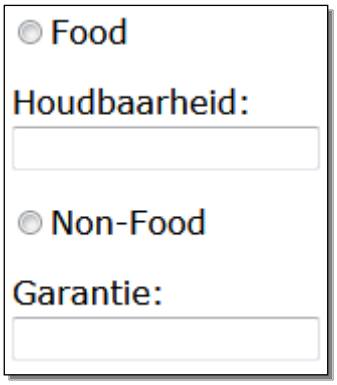
1. Je vult bij de bestaande records de kolom `soort` met `F` of `NF`
2. Je vult de kolom `houdbaarheid` in bij records met `F` in in de kolom `soort`
3. Je vult de kolom `garantie` in bij records met `NF` in de kolom `soort`

Je maakt via inheritance een onderscheid tussen food en non-food artikels.

Je maakt de class `Artikel` abstract en maak je twee derived classes: `FoodArtikel` en `NonFoodArtikel`

- De class `FoodArtikel` heeft een `int` attribuut `houdbaarheid`
- De class `NonFoodArtikel` heeft een `int` attribuut `garantie`

Je breidt de pagina om een nieuw artikel toe te voegen uit



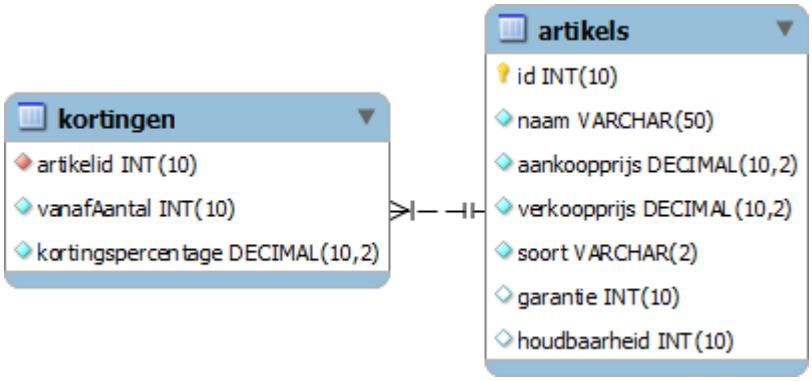
Je geeft de 1° radio button als id `food`.
Je geeft de 2° radio button als id `nonfood`
Je geeft het 1) extra tekstvak als id `houdbaarheid`
Je geeft het 1) extra tektsvak als id `garantie`

Je voegt volgende code toe aan het `<script>` tag. De code enabelt het tekstvak `houdbaarheid` als de gebruiker `Food` kiest en enabeld het tekstvak `Garantie` als de gebruiker `Non-Food` kiest.

```
document.getElementById('food').onclick =
  enableDisableInputs;
document.getElementById('nonfood').onclick =
  enableDisableInputs;
enableDisableInputs();
function enableDisableInputs() {
  document.getElementById('houdbaarheid').disabled =
    ! document.getElementById('food').checked;
    document.getElementById('garantie').disabled =
    ! document.getElementById('nonfood').checked;
}
```

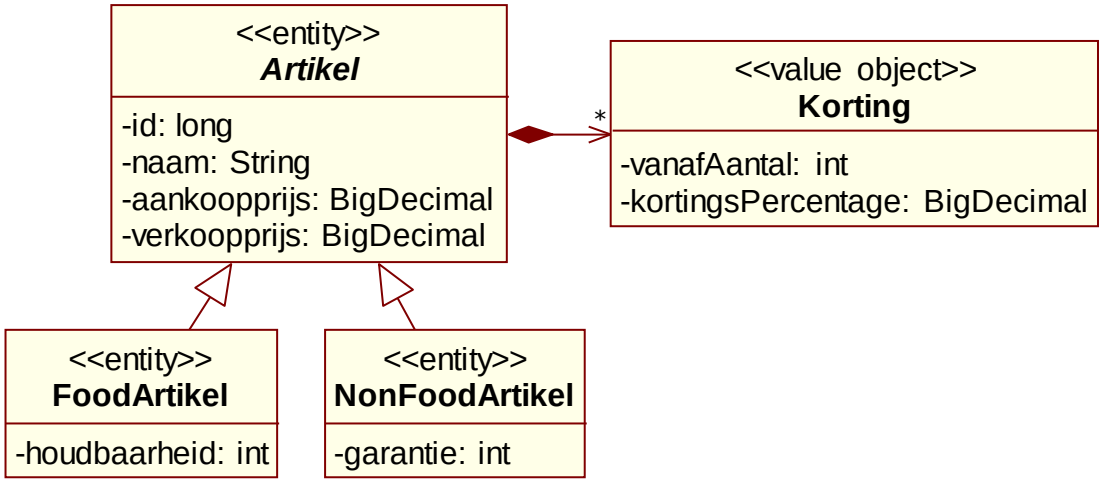
1.10 Kortingen

Je voert het script *Kortingen.sql* uit.
Dit script voegt een table `kortingen` toe



Deze table bevat hoeveelheidskortingen per artikel.
Een record met `artikelid` 1, `vanafaantal` 5 en `kortingspercentage` 2 betekent:
bij het kopen van artikel 1 krijg je 2% korting vanaf 5 stuks.

Je maakt een bijbehorende value object class `Korting`.
De class `Artikel` bevat een `Set` met een verzameling `Korting` objecten



Je maakt een pagina waarmee je de kortingen van een artikel kan zien.
De gebruiker ziet een alfabetische lijst van artikelnamen.
Iedere artikelnaam is een hyperlink.
Als de gebruiker zo'n hyperlink klikt, toon je in dezelfde pagina de kortingen van dat artikel.
Je toont per korting beide kortingseigenschappen.

1.11 Artikelgroepen

De database `artikels` bevat een table `artikelgroepen`

Je doet via de `MySQL Workbench` volgende aanpassingen

1. Je voegt enkele records toe aan deze table.
2. Je voert het script `ArtikelgroepIdToevoegenAanArtikels.sql` uit.
   De table `artikels` bevat nu een extra kolom: `artikelgroepId`.
3. Je vult bij elk record die kolom met een `artikelgroepid` uit de tabel `artikelgroepen`
4. Je voert het script `ForeignKeyArtikelgroepenToevoegenAanArtikels.sql` uit. Dit script maakt van de kolom `artikelgroepid` in de table `artikels` een foreign key die verwijst naar de primary key kolom `id` in de table `artikelgroepen`.

Je maakt een nieuwe entity class `Artikelgroep`. Je definieert ook een bidirectionele één op veel associatie tussen deze class en de class `Artikel`.

Eerste programmauitbreiding

Als de gebruiker een artikel toevoegt, vraag je een artikelgroep voor dit artikel.

Je voegt daartoe een lijst met de namen van artikelgroepen toe aan de pagina:

Tweede programmauitbreiding

Je voegt een programmaonderdeel toe: Artikels per artikelgroep.

De gebruiker ziet in de pagina de namen van de artikelgroepen als hyperlinks, alfabetisch gesorteerd

Als hij één van deze hyperlinks aanklikt, ziet hij de artikels van die artikelgroep, alfabetisch gesorteerd. Hij ziet per artikel nummer, naam en verkoopprijs

1.12 Artikellijst

Je maakt een pagina met een artikellijst.

De lijst bevat per artikel de artikelnaam en de naam van de artikelgroep.

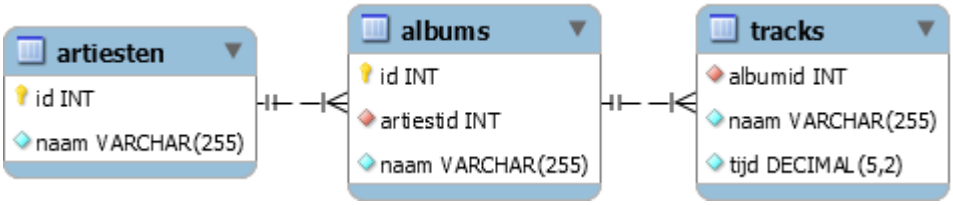
De lijst is gesorteerd op de artikelnaam.

Je bepaalt zelf de layout van de pagina.

Lees de informatie performant uit de database.

1.13 Muziek

Je voert het script `muziek.sql` uit. Dit script maakt een database `muziek`.
Je kan deze database openen met de gebruiker `cursist`, paswoord `cursist`



Je maakt een website.

Je toont op de welkompagina een alfabetische lijst van alle albums.
Je toont per album de naam van het album en de naam van de bijbehorende artiest.

Elke albumnaam is een hyperlink. Als de gebruiker deze aanklikt, toon je op een nieuwe pagina de albumtitel en de bijbehorende artiest. Je toont daar onder een lijst met de tracks van dat album. Je toont per track de naam en de tijd. Je toont onder de tracks de tijd van het album (de totale tijd van de tracks van dat album).

2 VOORBEELDOPLOSSINGEN

2.1 Alles voor de keuken

2.1.1 context.xml

Zelfde als in theorie, maar `fietsacademy` wordt drie keer `allesvoordekeuken`

2.1.2 persistence.xml

Zelfde als in theorie, maar `fietsacademy` wordt twee keer `allesvoordekeuken`

2.1.3 JPAFilter

Zelfde als in theorie, maar `fietsacademy` wordt `allesvoordekeuken`

2.1.4 head.tag

Zelfde als in theorie

2.1.5 menu.tag

```
<%@ tag language="java" pageEncoding="ISO-8859-1"%>
<%@taglib prefix='c' uri='http://java.sun.com/jsp/jstl/core'%>
<nav>
  <ul>
    <li><a href="<c:url value='/artikels/zoekenopnummer.htm'/>">Artikel zoeken
    op nummer</a></li>
    <li><a href="<c:url value='/artikels/toevoegen.htm'/>">Artikel toevoegen
      </a></li>
    <li><a href="<c:url value='/artikels/zoekenopnaam.htm'/>">Artikels zoeken
    op naam</a></li>
    <li><a href="<c:url value='/artikels/prijsverhoging.htm'/>">Prijsverhoging
    </a></li>
    <li><a href="<c:url value='/artikels/kortingen.htm'/>">Artikelkortingen
    </a></li>
    <li><a href="<c:url value='/artikels/perartikelgroep.htm'/>">Artikels per
    artikelgroep</a></li>
    <li><a href="<c:url value='/artikels.htm'/>">Artikellijst</a></li>
  </ul>
</nav>
```

2.1.6 default.css

Zelfde als in theorie

2.1.7 IndexServlet

```
package be.vdab.servlets;
// enkele imports
@WebServlet("/index.htm")
public class IndexServlet extends HttpServlet {
  private static final long serialVersionUID = 1L;
  private static final String VIEW = "/WEB-INF/JSP/index.jsp";
  private final Random random = new Random();
  @Override
  protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    request.setAttribute("random", 1 + random.nextInt(4));
    request.getRequestDispatcher(VIEW).forward(request, response);
  }
}
```

2.1.8 index.jsp

```
<%@ page contentType='text/html' pageEncoding='UTF-8' session='false'%>
<%@taglib prefix='c' uri='http://java.sun.com/jsp/jstl/core'%>
<%@taglib prefix="v" uri='http://vdab.be/tags'%>
<!doctype html>
```

```
<html lang='nl'>
<v:head title='Alles voor de keuken'/>
<body>
<header>
<v:menu />
<h1>Alles voor de keuken</h1>
<img src='<c:url value="/images/logo${random}.jpg"/>' alt='logo' id='logo'>
</header>
</body>
</html>
```

2.2 Artikel

2.2.1 De class Artikel

```
package be.vdab.entities;
// enkele imports
@Entity
@Table(name = "artikels")
public class Artikel implements Serializable {
  private static final long serialVersionUID = 1L;
  @Id
  private long id;
  private String naam;
  private BigDecimal aankoopprijs;
  private BigDecimal verkoopprijs;
  // je maakt getters voor id, naam, aankoopprijs en verkoopprijs
}
```

2.2.2 Uitbreiding in persistence.xml

```
<class>be.vdab.entities.Artikel</class>
```

2.3 Zoeken op nummer

2.3.1 Extra method in de class Artikel

```
public BigDecimal getWinstPercentage() {
  return verkoopprijs.subtract(aankoopprijs).divide(aankoopprijs, 2,
    RoundingMode.HALF_UP).multiply(BigDecimal.valueOf(100));
}
```

2.3.2 Extra method in de class JPAFilter

```
public static EntityManager getEntityManager() {
  return entityManagerFactory.createEntityManager();
}
```

2.3.3 ArtikelRepository

```
package be.vdab.repositories;
// enkele imports
public class ArtikelRepository {
  public Optional<Artikel> read(long id) {
    EntityManager entityManager = JPAFilter.getEntityManager();
    try {
      return Optional.ofNullable(entityManager.find(Artikel.class, id));
    } finally {
      entityManager.close();
    }
  }
}
```

2.3.4 StringUtils: zelfde als in theorie

2.3.5 ZoekenOpNummerServlet

```
package be.vdab.servlets;
// enkele imports ...
@WebServlet("/artikels/zoekenopnummer.htm")
public class ZoekenOpNummerServlet extends HttpServlet {
  private static final long serialVersionUID = 1L;
  private static final String VIEW = "/WEB-INF/JSP/artikels/zoekenopnummer.jsp";
  private final transient ArtikelRepository artikelRepository
    = new ArtikelRepository();
  @Override
  protected void doGet(HttpServletRequest request,  HttpServletResponse response)
    throws ServletException, IOException {
    if (request.getQueryString() != null) {
      String idString = request.getParameter("id");
      if (StringUtils.isLong(idString)) {
        artikelRepository.read(Long.parseLong(idString))
          .ifPresent(artikel -> request.setAttribute("artikel", artikel));
      } else {
        request.setAttribute("fouten",
          Collections.singletonMap("id", "tik een getal"));
      }
    }
    request.getRequestDispatcher(VIEW).forward(request, response);
  }
}
```

2.3.6 zoekenopnummer.jsp

```
<%@page contentType='text/html' pageEncoding='UTF-8' session='false'%>
<%@taglib uri='http://java.sun.com/jsp/jstl/core' prefix='c'%>
<%@taglib uri='http://java.sun.com/jsp/jstl/fmt' prefix='fmt'%>
<%@taglib uri='http://vdab.be/tags' prefix='v'%>
<!doctype html>
<html lang='nl'>
<head>
<v:head title='${empty artikel ? "Artikel zoeken" : artikel.naam}'/>
</head>
<body>
  <v:menu/>
  <h1>Artikel zoeken</h1>
  <form>
    <label>Nummer:<span>${fouten.id}</span>
    <input name='id' value='${param.id}'
      required autofocus type='number' min='1'></label>
    <input type='submit' value='Zoeken'>
  </form>
  <c:if test='${not empty param and empty fouten and empty artikel}'>
  Artikel niet gevonden
  </c:if>
  <c:if test='${not empty artikel}'>
    <dl><dt>Naam</dt>
    <dd>${artikel.naam}</dd>
        <dt>Aankoopprijs</dt>
    <dd><fmt:formatNumber value='${artikel.aankoopprijs}'/></dd>
        <dt>Verkoopprijs</dt>
    <dd><fmt:formatNumber value='${artikel.verkoopprijs}'/></dd>
        <dt>Winst</dt>
    <dd><fmt:formatNumber value='${artikel.winstPercentage}'/>%</dd>
    </dl>
  </c:if></body></html>
```

2.4 Toevoegen

2.4.1 Artikel: uitbreiding

Voor de variabele id:

```
@GeneratedValue(strategy = GenerationType.IDENTITY)
```

Extra code:

```
private static final BigDecimal MINIMUM_AANKOOPPRIJS = BigDecimal.valueOf(0.01);
protected Artikel() {}
public Artikel(String naam, BigDecimal aankoopprijs, BigDecimal verkoopprijs) {
  setNaam(naam);
  setAankoopprijs(aankoopprijs);
  setVerkoopprijs(verkoopprijs);
}
public static boolean isNaamValid(String naam) {
  return naam != null && !naam.trim().isEmpty();
}
public static boolean isAankoopprijsValid(BigDecimal aankoopprijs) {
  return aankoopprijs != null
    && aankoopprijs.compareTo(MINIMUM_AANKOOPPRIJS) >= 0;
}
public static boolean isVerkoopprijsValid(BigDecimal verkoopprijs,
  BigDecimal aankoopprijs) {
  return verkoopprijs != null && aankoopprijs != null
    && verkoopprijs.compareTo(aankoopprijs) >= 0;
}
public void setNaam(String naam) {
  if (!isNaamValid(naam)) {
    throw new IllegalArgumentException();
  }
  this.naam = naam;
}
public void setAankoopprijs(BigDecimal aankoopprijs) {
  if ( ! isAankoopprijsValid(aankoopprijs)) {
    throw new IllegalArgumentException();
  }
  this.aankoopprijs = aankoopprijs;
}
public void setVerkoopprijs(BigDecimal verkoopprijs) {
  if ( ! isVerkoopprijsValid(verkoopprijs ,aankoopprijs) {
    throw new IllegalArgumentException();
  }
  this.verkoopprijs = verkoopprijs;
}
```

2.4.2 ArtikelRepository

```
package be.vdab.repositories;
// enkele imports ...
public class ArtikelRepository {
  public Optional<Artikel> read(long id, EntityManager entityManager) {
    return Optional.ofNullable(entityManager.find(Artikel.class, id));
  }
  public void create(Artikel artikel, EntityManager entityManager) {
      entityManager.persist(artikel);
  }
}
```

2.4.3 ArtikelService

```
package be.vdab.services;
// enkele imports ...
public class ArtikelService {
  private final ArtikelRepository artikelRepository = new ArtikelRepository();
  public Optional<Artikel> read(long id) {
    EntityManager entityManager = JPAFilter.getEntityManager();
    try {
      return artikelRepository.read(id, entityManager);
    } finally {
      entityManager.close();
    }
  }
  public void create(Artikel artikel) {
    EntityManager entityManager = JPAFilter.getEntityManager();
    entityManager.getTransaction().begin();
    try {
      artikelRepository.create(artikel, entityManager);
      entityManager.getTransaction().commit();
    } catch (PersistenceException ex) {
      entityManager.getTransaction().rollback();
      throw ex;
    } finally {
      entityManager.close();
    }
  }
}
```

2.4.4 ZoekenOpNummerServlet

`ArtikelRepository` wordt `ArtikelService` en `artikelRepository` wordt `artikelService`

2.4.5 ToevoegenServlet

```
package be.vdab.servlets.artikels;
// enkele imports ...
@WebServlet("/artikels/toevoegen.htm")
public class ToevoegenServlet extends HttpServlet {
  private static final long serialVersionUID = 1L;
  private static final String VIEW = "/WEB-INF/JSP/artikels/toevoegen.jsp";
  private static final String REDIRECT_URL
    = "%s/artikels/zoekenopnummer.htm?id=%d";
  private final transient ArtikelService artikelService = new ArtikelService();
  @Override
  protected void doGet(HttpServletRequest request,
      HttpServletResponse response) throws ServletException, IOException {
    request.getRequestDispatcher(VIEW).forward(request, response);
  }
  @Override
  protected void doPost(HttpServletRequest request,
      HttpServletResponse response) throws ServletException, IOException {
    Map<String, String> fouten = new HashMap<>();
    String naam = request.getParameter("naam");
    if (!Artikel.isNaamValid(naam)) {
      fouten.put("naam", "verplicht");
    }
    String aankoopprijsString = request.getParameter("aankoopprijs");
    BigDecimal aankoopprijs = null;
    if (StringUtils.isBigDecimal(aankoopprijsString)) {
      aankoopprijs = new BigDecimal(aankoopprijsString));
      if (!Artikel.isAankoopprijsValid(aankoopprijs)) {
        fouten.put("aankoopprijs", "tik een positief getal of 0");
```

```
      }
    } else {
      fouten.put("aankoopprijs", "tik een positief getal of 0");
    }
    String verkoopprijsString = request.getParameter("verkoopprijs");
    BigDecimal verkoopprijs = null;
    if (StringUtils.isBigDecimal(verkoopprijsString)) {
      verkoopprijs = new BigDecimal(verkoopprijsString));
      if (!Artikel.isVerkoopprijsValid(verkoopprijs, aankoopprijs)) {
        fouten.put("verkoopprijs",
          "tik een positief getal groter of gelijk aan de aankoopprijs");
      }
    } else {
      fouten.put("verkoopprijs", "tik een positief getal of 0");
    }
    if (fouten.isEmpty()) {
      Artikel artikel = new Artikel(naam, aankoopprijs, verkoopprijs);
      artikelService.create(artikel);
      response.sendRedirect(response.encodeRedirectURL(String.format(
          REDIRECT_URL, request.getContextPath(), artikel.getId())));
    }
    if ( ! fouten.isEmpty()) {
      request.setAttribute("fouten", fouten);
      request.getRequestDispatcher(VIEW).forward(request, response);
    }
  }
}
```

2.4.6 toevoegen.jsp

```
<%@page contentType='text/html' pageEncoding='UTF-8' session='false'%>
<%@taglib uri='http://java.sun.com/jsp/jstl/core' prefix='c'%>
<%@taglib uri='http://vdab.be/tags' prefix='v'%>
<!doctype html>
<html lang='nl'>
  <head>
  <v:head title='Artikel toevoegen'/>
  </head>
  <body>
    <v:menu/>
    <h1>Artikel toevoegen</h1>
    <form method='post' id='toevoegform'>
      <label>Naam:<span>${fouten.naam}</span>
      <input name='naam' value='${param.naam}' autofocus required></label>
      <label>Aankoopprijs:<span>${fouten.aankoopprijs}</span>
      <input name='aankoopprijs' value='${param.aankoopprijs}' required
        type='number' min='0' step='0.01'></label>
      <label>Verkoopprijs:<span>${fouten.verkoopprijs}</span>
       <input name='verkoopprijs' value='${param.verkoopprijs}' required
         type='number' min='0' step='0.01'></label>
      <input type='submit' value='Toevoegen' id='toevoegknop'>
    </form>
    <script>
      document.getElementById('toevoegform').onsubmit= function() {
        document.getElementById('toevoegknop').disabled=true;
      };
    </script>
  </body>
</html>
```

2.5 ThreadLocal

2.5.1 JPAFilter

Zelfde als in theorie, maar `fietsacademy` wordt `allesvoordekeuken`

2.5.2 AbstractRepository

Zelfde als in theorie

2.5.3 ArtikelRepository

```
package be.vdab.repositories;
import be.vdab.entities.Artikel;
public class ArtikelRepository extends AbstractRepository {
  public Optional<Artikel> read(long id) {
    return Optional.ofNullable(getEntityManager().find(Artikel.class, id));
  }
  public void create(Artikel artikel) {
    getEntityManager().persist(artikel);
  }
}
```

2.5.4 AbstractService

Zelfde als in theorie

2.5.5 ArtikelService

```
package be.vdab.services;
import be.vdab.repositories.ArtikelRepository;
import be.vdab.entities.Artikel;
public class ArtikelService extends AbstractService {
  private final ArtikelRepository artikelRepository = new ArtikelRepository();
  public Optional<Artikel> read(long id) {
    return artikelRepository.read(id);
  }
  public void create(Artikel artikel) {
    beginTransaction();
    try {
      artikelRepository.create(artikel);
      commit();
    } catch (PersistenceException ex) {
      rollback();
      throw ex;
    }
  }
}
```

2.6 Zoeken op naam

2.6.1 orm.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<entity-mappings xmlns="http://xmlns.jcp.org/xml/ns/persistence/orm"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/persistence/orm
  http://xmlns.jcp.org/xml/ns/persistence/orm_2_1.xsd" version="2.1">
  <named-query name='Artikel.findByNaamContains'>
    <query>
      select a from Artikel a where a.naam like :zoals order by a.naam
    </query>
  </named-query>
</entity-mappings>
```

2.6.2 ArtikelRepository: extra method

```
public List<Artikel> findByNaamContains(String woord) {
```

```
  return getEntityManager()
    .createNamedQuery("Artikel.findByNaamContains", Artikel.class)
    .setParameter("zoals", '%' + woord + '%').getResultList();
}
```

2.6.3 ArtikelService: extra method

```
public List<Artikel> findByNaamContains(String woord) {
  return artikelRepository.findByNaamContains(woord);
}
```

2.6.4 ZoekenOpNaamServlet

```
package be.vdab.servlets.artikels;
// enkele imports ...
@WebServlet("/artikels/zoekenopnaam.htm")
public class ZoekenOpNaamServlet extends HttpServlet {
  private static final long serialVersionUID = 1L;
  private final static String VIEW = "/WEB-INF/JSP/artikels/zoekenopnaam.jsp";
  private final transient ArtikelService artikelService = new ArtikelService();
  @Override
  protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    String woord = request.getParameter("woord");
    if (woord != null) {
      if (woord.trim().isEmpty()) {
        request.setAttribute("fouten",
          Collections.singletonMap("woord", "verplicht"));
      } else {
        request.setAttribute("artikels",
          artikelService.findByNaamContains(woord));
      }
    }
    request.getRequestDispatcher(VIEW).forward(request, response);
  }
}
```

2.6.5 zoekenopnaam.jsp

```
<%@page contentType='text/html' pageEncoding='UTF-8' session='false'%>
<%@taglib prefix='c' uri='http://java.sun.com/jsp/jstl/core'%>
<%@taglib uri='http://java.sun.com/jsp/jstl/fmt' prefix='fmt'%>
<%@taglib uri='http://vdab.be/tags' prefix='v'%>
<!doctype html>
<html lang="nl">
<head>
  <v:head title='Artikels zoeken op naam'/>
  <style>
    td {
      text-align:right;
    }
    td:nth-child(2) {
      text-align:left;
    }
  </style>
</head>
<body>
  <v:menu/>
  <h1>Artikels zoeken op naam</h1>
  <form>
    <label>Woord:<span>${fouten.woord}</span>
    <input name='woord' value='${param.woord}' autofocus required
      type='search'></label>
    <input type='submit' value='Zoeken'>
```

```
  </form>
  <c:if test='${not empty param and empty fouten and empty artikels}'>
    Geen artikels gevonden
  </c:if>
  <c:if test='${not empty artikels}'>
    <table>
      <thead>
        <tr>
          <th>Nummer</th>
          <th>Naam</th>
          <th>Aankoopprijs</th>
          <th>Verkoopprijs</th>
        </tr>
      </thead>
      <tbody>
        <c:forEach items='${artikels}' var='artikel'>
          <tr>
           <td>${artikel.id}</td>
           <td>${artikel.naam}</td>
           <td><fmt:formatNumber value='${artikel.aankoopprijs}'
             minFractionDigits='2' maxFractionDigits='2'/></td>
           <td><fmt:formatNumber value='${artikel.verkoopprijs}'
             minFractionDigits='2' maxFractionDigits='2'/></td>
          </tr>
        </c:forEach>
      </tbody>
    </table>
  </c:if>
</body>
</html>
```

2.7 Prijsverhoging

2.7.1 orm.xml: extra named query

```
<named-query name="Artikel.prijsverhoging">
  <query>
    update Artikel a
    set a.verkoopprijs = a.verkoopprijs * :factor
  </query>
</named-query>
```

2.7.2 ArtikelRepository: extra method

```
public void prijsverhoging(BigDecimal factor) {
  getEntityManager().createNamedQuery("Artikel.prijsverhoging")
    .setParameter("factor", factor)
    .executeUpdate();
}
```

2.7.3 ArtikelService: extra method

```
public void prijsverhoging(BigDecimal percentage) {
  BigDecimal factor
    = BigDecimal.ONE.add(percentage.divide(BigDecimal.valueOf(100)));
  beginTransaction();
  try {
    artikelRepository.prijsverhoging(factor);
    commit();
  } catch (PersistenceException ex) {
    rollback();
    throw ex;
  }
}
```

2.8 PrijsverhogingServlet

```
package be.vdab.servlets.artikels;
// enkele imports ...
@WebServlet("/artikels/prijsverhoging.htm")
public class PrijsverhogingServlet extends HttpServlet {
  private static final long serialVersionUID = 1L;
  private static final String VIEW = "/WEB-INF/JSP/artikels/prijsverhoging.jsp";
  private final transient ArtikelService artikelService = new ArtikelService();
  @Override
  protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    request.getRequestDispatcher(VIEW).forward(request, response);
  };
  @Override
  protected void doPost(HttpServletRequest request,
    HttpServletResponse response) throws ServletException, IOException {
    Map<String, String> fouten = new HashMap<>();
    String percentageString = request.getParameter("percentage");
    BigDecimal percentage = null;
    if (StringUtils.isBigDecimal(percentageString)) {
      percentage = new BigDecimal(percentageString));
      if (percentage.compareTo(BigDecimal.ZERO) <= 0) {
        fouten.put("percentage", "tik een positief getal");
      }
    } else {
      fouten.put("percentage", "tik een positief getal");
    }
    if (fouten.isEmpty()) {
      artikelService.prijsverhoging(percentage);
      response.sendRedirect(request.getContextPath()));
    } else {
      request.setAttribute("fouten", fouten);
      request.getRequestDispatcher(VIEW).forward(request, response);
    }
  }
}
```

2.8.1WEB-INF/JSP/artikels/prijsverhoging.jsp

```
<%@page contentType='text/html' pageEncoding='UTF-8' session='false'%>
<%@taglib  prefix='c' uri='http://java.sun.com/jsp/jstl/core'%>
<%@taglib uri='http://vdab.be/tags' prefix='v'%>
<!doctype html>
<html lang='nl'>
<head>
<v:head title='Prijsverhoging'/>
</head>
<body>
<v:menu/>
  <h1>Prijsverhoging</h1>
  <form method='post' id='verhogingform'>
    <label>Percentage:<span>${fouten.percentage}</span>
    <input name='percentage' value='${param.percentage}' type='number'
      min='0.01' step='0.01' autofocus required>
    </label>
    <input type='submit' value='Verhoog prijzen' id='verhogingknop'>
  </form>
  <script>
    document.getElementById('verhogingform').onsubmit= function() {
      document.getElementById('verhogingknop').disabled=true;
    };
```

```
</script>
  </body>
</html>
```

2.9 Food en non-food artikels

2.9.1 Artikel

```
package be.vdab.entities;
// enkele imports ...
@Entity
@Table(name = "artikels")
@Inheritance(strategy = InheritanceType.SINGLE_TABLE)
@DiscriminatorColumn(name = "soort")
public abstract class Artikel implements Serializable {
  ...
}
```

2.9.2 FoodArtikel

```
package be.vdab.entities;
// enkele imports ...
@Entity
@DiscriminatorValue("F")
public class FoodArtikel extends Artikel {
  private static final long serialVersionUID = 1L;
  private int houdbaarheid;
  public FoodArtikel(String naam, BigDecimal aankoopprijs,
    BigDecimal verkoopprijs, int houdbaarheid) {
    super(naam, aankoopprijs, verkoopprijs);
    setHoudbaarheid(houdbaarheid);
  }
  public static boolean isHoudbaarheidValid(int houdbaarheid) {
    return houdbaarheid >= 1;
  }
  public int getHoudbaarheid() {
    return houdbaarheid;
  }
  public void setHoudbaarheid(int houdbaarheid) {
    if (!isHoudbaarheidValid(houdbaarheid)) {
      throw new IllegalArgumentException();
    }
    this.houdbaarheid = houdbaarheid;
  }
  protected FoodArtikel() {}
}
```

2.9.3 NonFoodArtikel

```
package be.vdab.entities;
// enkele imports ...
@Entity
@DiscriminatorValue("NF")
public class NonFoodArtikel extends Artikel {
  private static final long serialVersionUID = 1L;
  private int garantie;
  public NonFoodArtikel(String naam, BigDecimal aankoopprijs,
    BigDecimal verkoopprijs, int garantie) {
    super(naam, aankoopprijs, verkoopprijs);
    setGarantie(garantie);
  }
  public static boolean isGarantieValid(int garantie) {
    return garantie >= 0;
```

```
  }
  public int getGarantie() {
    return garantie;
  }
  public void setGarantie(int garantie) {
    if (!isGarantieValid(garantie)) {
      throw new IllegalArgumentException();
    }
    this.garantie = garantie;
  }
  protected NonFoodArtikel() {}
}
```

2.9.4 persistence.xml: uitbreiding

```
<class>be.vdab.entities.FoodArtikel</class>
<class>be.vdab.entities.NonFoodArtikel</class>
```

2.9.5 StringUtils: extra method

```
public static boolean isInt(String string) {
  if (string == null) {
    return false;
  }
  try {
    Integer.parseInt(string);
    return true;
  } catch (NumberFormatException ex) {
    return false;
  }
}
```

2.9.6 ToevoegenServlet: doPost method

Extra code voor `if (fouten.isEmpty()) {`

```
int houdbaarheid = 0;
int garantie = 0;
String soort = request.getParameter("soort");
if (soort == null) {
  fouten.put("soort", "maak een keuze");
} else {
  switch (soort) {
    case "F":
      String houdbaarheidString = request.getParameter("houdbaarheid");
      if (StringUtils.isInt(houdbaarheidString)) {
        houdbaarheid = Integer.parseInt(houdbaarheidString);
        if ( ! FoodArtikel.isHoudbaarheidValid(houdbaarheid)) {
          fouten.put("houdbaarheid", "tik een positief getal");
        }
      } else {
        fouten.put("houdbaarheid", "tik een positief getal");
      }
      break;
    case "NF":
      String garantieString = request.getParameter("garantie");
      if (StringUtils.isInt(garantieString)) {
        garantie = Integer.parseInt(garantieString);
        if (!NonFoodArtikel.isGarantieValid(garantie)) {
          fouten.put("garantie", "tik een positief getal of 0");
        }
      } else {
        fouten.put("garantie", "tik een positief getal of 0");
      }
      break;
```

```
    default:
      fouten.put("soort", "maak een keuze");
  }
}
```

Regel `Artikel artikel = new Artikel(naam, aankoopprijs, verkoopprijs);` wordt

```
Artikel artikel;
if ("F".equals(soort)) {
  artikel = new FoodArtikel(naam, aankoopprijs, verkoopprijs, houdbaarheid);
} else {
  artikel = new NonFoodArtikel(naam, aankoopprijs, verkoopprijs, garantie);
}
```

2.9.7 toevoegen.jsp

Extra code toevoegen voor submit button

```
<div><span>${fouten.soort}</span><label>
<input name='soort' value='F' type='radio' id='food'
  ${param.soort == "F" ? "checked" : "" }>Food</label></div>
<label>Houdbaarheid: <span>${fouten.houdbaarheid}</span>
<input name='houdbaarheid' value='${param.houdbaarheid}' type='number' min='1'
  id='houdbaarheid'></label>
<div><label><input name='soort' value='NF' type='radio' id='nonfood'
  ${param.soort == "NF" ? "checked" : ""}>Non-Food</label></div>
<label>Garantie: <span>${fouten.garantie}</span>
<input name='garantie' value='${param.garantie}' type='number' min='0'
  id='garantie'></label>
```

2.10 Kortingen

2.10.1 Korting

```
package be.vdab.valueobjects;
// enkele imports
@Embeddable
public class Korting implements Serializable {
  private static final long serialVersionUID = 1L;
  private int vanafAantal;
  private BigDecimal kortingsPercentage;
  public int getVanafAantal() {
    return vanafAantal;
  }
  public BigDecimal getKortingsPercentage() {
    return kortingsPercentage;
  }
  @Override
  public int hashCode() {
    return vanafAantal;
  }
  @Override
  public boolean equals(Object object) {
    if (!(object instanceof Korting)) {
      return false;
    }
    Korting andereKorting = (Korting) object;
    return vanafAantal == andereKorting.vanafAantal;
  }
}
```

2.10.2 Artikel: uitbreiding

```
@ElementCollection @OrderBy("vanafAantal")
@CollectionTable(name = "kortingen",
  joinColumns = @JoinColumn(name = "artikelid"))
private Set<Korting> kortingen;
public Set<Korting> getKortingen() {
  return Collections.unmodifiableSet(kortingen);
}
```

2.10.3 orm.xml: extra named query

```
<named-query name="Artikel.findAll">
  <query>
    select a from Artikel a
    order by a.naam
  </query>
</named-query>
```

2.10.4 ArtikelRepository: uitbreiding

```
public List<Artikel> findAll() {
  return getEntityManager().createNamedQuery("Artikel.findAll", Artikel.class)
    .getResultList();
}
```

2.10.5 ArtikelService: uitbreiding

```
public List<Artikel> findAll() {
  return artikelRepository.findAll();
}
```

2.10.6 KortingenServlet

```
package be.vdab.servlets.artikels;
// enkele imports
@WebServlet("/artikels/kortingen.htm")
public class KortingenServlet extends HttpServlet {
  private static final long serialVersionUID = 1L;
  private static final String VIEW = "/WEB-INF/JSP/artikels/kortingen.jsp";
  private final transient ArtikelService artikelService = new ArtikelService();
  @Override
  protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    request.setAttribute("artikels", artikelService.findAll());
    String id = request.getParameter("id");
    if (id != null) {
      artikelService.read(Long.parseLong(id))
        .ifPresent(artikel -> request.setAttribute("artikel", artikel));
    }
    request.getRequestDispatcher(VIEW).forward(request, response);
  }
}
```

2.10.7 kortingen.jsp

```
<%@page contentType='text/html' pageEncoding='UTF-8' session='false'%>
<%@taglib uri='http://java.sun.com/jsp/jstl/core' prefix='c'%>
<%@taglib uri='http://java.sun.com/jsp/jstl/fmt' prefix='fmt'%>
<%@taglib uri='http://vdab.be/tags' prefix='v'%>
<!doctype html>
<html lang='nl'>
<head>
  <v:head title='Artikelkortingen'/>
  <style>
  td {
  text-align: right:
```

```
  }
  </style>
</head>
<body>
  <v:menu/>
  <h1>Artikelkortingen</h1>
  <ul class='zonderbolletjes'>
  <c:forEach items='${artikels}' var='artikel'>
    <c:url value='' var='url'>
      <c:param name='id' value='${artikel.id}'/>
    </c:url>
    <li><a href='${url}'>${artikel.naam}</a></li>
   </c:forEach>
  </ul>
  <c:if test='${not empty artikel}'>
    <h2>${artikel.naam}</h2>
    <table>
      <thead>
        <tr>
          <th>Vanaf aantal</th>
          <th>% korting</th>
        </tr>
      </thead>
      <tbody>
      <c:forEach items='${artikel.kortingen}' var='korting'>
        <tr>
          <td>${korting.vanafAantal }</td>
          <td><fmt:formatNumber
            value='${korting.kortingsPercentage}' minFractionDigits='2'
            maxFractionDigits='2'/></td>
        </tr>
      </c:forEach>
      </tbody>
    </table>
  </c:if>
</body>
</html>
```

2.11 Artikelgroepen

2.11.1 Artikelgroep

```
package be.vdab.entities;
// enkele imports
@Entity
@Table(name = "artikelgroepen")
public class Artikelgroep implements Serializable {
  private static final long serialVersionUID = 1L;
  @Id
  @GeneratedValue(strategy = GenerationType.IDENTITY)
  private long id;
  private String naam;
  @OneToMany(mappedBy = "artikelgroep")
  private Set<Artikel> artikels;
  // Je maakt een getter voor id en naam
  public Set<Artikel> getArtikels() {
    return Collections.unmodifiableSet(artikels);
  }
```

```
  public void add(Artikel artikel) {
    artikels.add(artikel);
    if (artikel.getArtikelgroep() != this) {
      artikel.setArtikelgroep(this);
    }
  }
  public void remove(Artikel artikel) {
    artikels.remove(artikel);
    if (artikel.getArtikelgroep() == this) {
      artikel.setArtikelgroep(null);
      }
  }
  @Override
  public boolean equals(Object object) {
    if (!(object instanceof Artikelgroep)) {
      return false;
    }
    Artikelgroep andereArtikelgroep = (Artikelgroep) object;
    return naam.equalsIgnoreCase(andereArtikelgroep.naam);
  }
  @Override
  public int hashCode() {
    return naam.toUpperCase().hashCode();
  }
}
```

2.11.2 persistence.xml: uitbreiding

```
<class>be.vdab.entities.Artikelgroep</class>
```

2.11.3 Artikel: uitbreiding en aangepaste constructor

```
@ManyToOne(fetch = FetchType.LAZY, optional = false)
@JoinColumn(name = "artikelgroepid")
private Artikelgroep artikelgroep;
public Artikelgroep getArtikelgroep() {
  return artikelgroep;
}
public void setArtikelgroep(Artikelgroep artikelgroep) {
  if (this.artikelgroep != null
    && this.artikelgroep.getArtikels().contains(this)) {
    this.artikelgroep.remove(this);
  }
  this.artikelgroep = artikelgroep;
  if (artikelgroep != null && ! artikelgroep.getArtikels().contains(this)) {
    artikelgroep.add(this);
  }
}
@Override
public boolean equals(Object object) {
  if (!(object instanceof Artikel)) {
    return false;
  }
  Artikel anderArtikel = (Artikel) object;
  return naam.equalsIgnoreCase(anderArtikel.naam);
}
@Override
public int hashCode() {
  return naam.toUpperCase().hashCode();
}
```

```
public Artikel(String naam, BigDecimal aankoopprijs, BigDecimal verkoopprijs,
Artikelgroep artikelgroep) {
  if (!isAankoopprijsVerkoopprijsValid(aankoopprijs, verkoopprijs)) {
    throw new IllegalArgumentException();
  }
  setNaam(naam);
  setAankoopprijs(aankoopprijs);
  setVerkoopprijs(verkoopprijs);
  setArtikelgroep(artikelgroep);
}
```

2.11.4 FoodArtikel: aangepaste constructor

```
public FoodArtikel(String naam, BigDecimal aankoopprijs,
  BigDecimal verkoopprijs, int houdbaarheid, Artikelgroep artikelgroep) {
  super(naam, aankoopprijs, verkoopprijs, artikelgroep);
  setHoudbaarheid(houdbaarheid);
}
```

2.11.5 NonFoodArtikel: aangepaste constructor

```
public NonFoodArtikel(String naam, BigDecimal aankoopprijs,
    BigDecimal verkoopprijs, int garantie, Artikelgroep artikelgroep) {
  super(naam, aankoopprijs, verkoopprijs, artikelgroep);
  setGarantie(garantie);
}
```

2.11.6 orm.xml: extra named query

```
<named-query name="Artikelgroepen.findAll">
  <query>select a
         from Artikelgroep a
         order by a.naam
  </query>
</named-query>
```

2.11.7 ArtikelgroepRepository

```
package be.vdab.repositories;
// enkele imports
public class ArtikelgroepRepository extends AbstractRepository {
  public List<Artikelgroep> findAll() {
    return getEntityManager().createNamedQuery("Artikelgroepen.findAll",
      Artikelgroep.class).getResultList();
  }
  public Optional<Artikelgroep> read(long id) {
    return Optional.ofNullable(getEntityManager().find(Artikelgroep.class, id));
  }
}
```

2.11.8 ArtikelgroepService

```
package be.vdab.services;
// enkele imports
public class ArtikelgroepService extends AbstractService {
  private final ArtikelgroepRepository artikelgroepRepository
    = new ArtikelgroepRepository();
  public List<Artikelgroep> findAll() {
    return artikelgroepRepository.findAll();
  }
  public Optional<Artikelgroep> read(long id) {
    return artikelgroepRepository.read(id);
  }
}
```

2.11.9 ToevoegenServlet: uitbreiding

Een extra private variabele

```
private final transient ArtikelgroepService artikelgroepService
  = new ArtikelgroepService();
```

doGet method

```
@Override
protected void doGet(HttpServletRequest request, HttpServletResponse response)
  throws ServletException, IOException {
  request.setAttribute("artikelgroepen", artikelgroepService.findAll());
  request.getRequestDispatcher(VIEW).forward(request, response);
}
```

doPost method: extra code voor `if (fouten.isEmpty()) {`

```
String artikelgroepId = request.getParameter("artikelgroepid");
if (artikelgroepId== null) {
  fouten.put("artikelgroepid", "verplicht");
}
```

doPost method: extra code na `Artikel artikel;`

```
Artikelgroep artikelgroep
  = artikelgroepService.read(Long.parseLong(artikelgroepId)).get();
```

doPost method: extra parameter in constructor oproep van `FoodArtikel` en `NonFoodArtikel`

```
, artikelgroep
```

2.11.10 toevoegen.jsp

Extra code voor de submit button

```
<label>Artikelgroep:<span>${fouten.artikelgroepid}</span>
<select name='artikelgroepid' size='${artikelgroepen.size()}' required>
  <c:forEach items='${artikelgroepen}' var='artikelgroep'>
    <option value='${artikelgroep.id}'
      ${artikelgroep.id == param.artikelgroepid ? 'selected' : ''}>
      ${artikelgroep.naam}</option>
  </c:forEach>
</select></label>
```

2.11.11 PerArtikelgroepServlet

```
package be.vdab.servlets.artikels;
// enkele imports
@WebServlet("/artikels/perartikelgroep.htm")
public class PerArtikelgroepServlet extends HttpServlet {
  private static final long serialVersionUID = 1L;
  private static final String VIEW =
    "/WEB-INF/JSP/artikels/perartikelgroep.jsp";
  private final transient ArtikelgroepService artikelgroepService
    = new ArtikelgroepService();
  @Override
  protected void doGet(HttpServletRequest request,  HttpServletResponse response)
    throws ServletException, IOException {
    request.setAttribute("artikelgroepen", artikelgroepService.findAll());
    String id = request.getParameter("id");
    if (id != null) {
      artikelgroepService.read(Long.parseLong(id))
        .ifPresent(artikelgroep ->
          request.setAttribute("artikelgroep", artikelgroep));
    }
    request.getRequestDispatcher(VIEW).forward(request, response);
  }
```

```
}
```

2.11.12 perartikelgroep.jsp

```
<%@page contentType='text/html' pageEncoding='UTF-8' session='false'%>
<%@taglib uri='http://java.sun.com/jsp/jstl/core' prefix='c'%>
<%@taglib uri='http://java.sun.com/jsp/jstl/fmt' prefix='fmt'%>
<%@taglib uri='http://vdab.be/tags' prefix='v'%>
<!doctype html>
<html lang='nl'>
<head>
  <v:head title='Artikels per artikelgroep'/>
  <style>
    td:first-child, td:last-child {
      text-align:right;
    }
  </style>
</head>
<body>
  <v:menu/>
  <h1>Artikels per artikelgroep</h1>
  <ul class='zonderbolletjes'>
  <c:forEach items='${artikelgroepen}' var='artikelgroep'>
    <c:url value='' var='url'>
      <c:param name='id' value='${artikelgroep.id}'/>
    </c:url>
    <li><a href='${url}'>${artikelgroep.naam}</a></li>
  </c:forEach>
  </ul>
  <c:if test='${not empty artikelgroep}'>
    <h2>${artikelgroep.naam}</h2>
    <table>
      <thead>
        <tr>
          <th>Nummer</th>
          <th>Naam</th>
          <th>Verkoopprijs</th>
        </tr>
      </thead>
      <tbody>
      <c:forEach items='${artikelgroep.artikels}' var='artikel'>
        <tr>
          <td>${artikel.id}</td>
          <td>${artikel.naam}</td>
          <td><fmt:formatNumber value='${artikel.verkoopprijs}'
            minFractionDigits='2' maxFractionDigits='2'/></td>
        </tr>
      </c:forEach>
    </tbody>
    </table>
  </c:if>
</body>
</html>
```

2.12 Artikellijst

2.12.1 Artikel: extra annotation

```
@NamedEntityGraph(name = "Artikel.metArtikelgroep",
  attributeNodes = @NamedAttributeNode("artikelgroep"))
```

2.12.2 ArtikelRepository: nieuwe method

```
public List<Artikel> findAllMetArtikelGroep() {
  return getEntityManager()
    .createNamedQuery("Artikel.findAll", Artikel.class)
    .setHint("javax.persistence.loadgraph",
       getEntityManager().createEntityGraph("Artikel.metArtikelgroep"))
    .getResultList();
}
```

2.12.3 ArtikelService: nieuwe method

```
public List<Artikel> findAllMetArtikelgroep() {
  return artikelRepository.findAllMetArtikelGroep();
}
```

2.12.4 ArtikelLijstServlet

```
package be.vdab.servlets.artikels;
// enkele imports ...
@WebServlet("/artikels.htm")
public class ArtikelLijstServlet extends HttpServlet {
  private static final long serialVersionUID = 1L;
  private final static String VIEW = "/WEB-INF/JSP/artikels/artikels.jsp";
  private final transient ArtikelService artikelService = new ArtikelService();
  @Override
  protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    request.setAttribute("artikels", artikelService.findAllMetArtikelgroep());
    request.getRequestDispatcher(VIEW).forward(request, response);
  }
}
```

2.12.5 artikels.jsp

```
<%@page contentType='text/html' pageEncoding='UTF-8' session='false'%>
<%@taglib uri='http://java.sun.com/jsp/jstl/core' prefix='c'%>
<%@taglib uri='http://vdab.be/tags' prefix='v'%>
<!doctype html>
<html lang='nl'>
<head>
  <v:head title='Artikels'/>
</head>
<body>
  <v:menu/>
  <h1>Artikels</h1>
  <ul>
  <c:forEach items='${artikels}' var='artikel'>
    <li>${artikel.naam} (${artikel.artikelgroep.naam})</li>
  </c:forEach>
  </ul>
  <body>
</html>
```

2.13 Muziek

2.13.1 context.xml

Zelfde als in theorie, maar `fietsacademy` wordt drie keer `muziek`

2.13.2 persistence.xml

Zelfde als in theorie, maar `fietsacademy` wordt twee keer `muziek`

en extra regels

```
<class>be.vdab.entities.Artiest</class>
<class>be.vdab.entities.Album</class>
<class>be.vdab.valueobjects.Track</class>
```

2.13.3 orm.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<entity-mappings xmlns="http://xmlns.jcp.org/xml/ns/persistence/orm"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/persistence/orm
    http://xmlns.jcp.org/xml/ns/persistence/orm_2_1.xsd"
  version="2.1">
  <named-query name='Album.findAll'>
    <query>
      select a from Album a
      order by a.naam
    </query>
  </named-query>
</entity-mappings>
```

2.13.4 JPAFilter

Zelfde als in theorie, maar `fietsacademy` wordt `muziek`

2.13.5 Artiest

```
package be.vdab.entities;
// enkele imports ...
@Entity
@Table(name = "artiesten")
public class Artiest implements Serializable {
  private static final long serialVersionUID = 1L;
  @Id
  @GeneratedValue(strategy = GenerationType.IDENTITY)
  private long id;
  private String naam;
  // en getters voor id en naam
}
```

2.13.6 Album

```
package be.vdab.entities;
// enkele imports ...
@Entity
@Table(name = "albums")
@NamedEntityGraph(name =Album. MET_ARTIEST,
  attributeNodes = @NamedAttributeNode("artiest") )
public class Album implements Serializable {
  private static final long serialVersionUID = 1L;
  public static final String MET_ARTIEST = "Album.metArtiest";
  @Id
  @GeneratedValue(strategy = GenerationType.IDENTITY)
  private long id;
  private String naam;
  @ManyToOne(optional = false, fetch = FetchType.LAZY)
  @JoinColumn(name = "artiestid")
  private Artiest artiest;
  @ElementCollection
  @CollectionTable(name="tracks", joinColumns=@JoinColumn(name = "albumid"))
  private Set<Track> tracks;
```

```
  // en getters voor id, naam, artiest en tracks
  public BigDecimal getTijd() {
    BigDecimal tijd = BigDecimal.ZERO;
    for (Track track : tracks) {
      tijd=tijd.add(track.getTijd());
    }
    return tijd;
  }
}
```

2.13.7 Track

```
package be.vdab.valueobjects;
// enkele imports ...
@Embeddable
public class Track implements Serializable {
  private static final long serialVersionUID = 1L;
  private String naam;
  private BigDecimal tijd;
  // en getters voor naam en tijd
  @Override
  public boolean equals(Object object) {
    if (! (object instanceof Track)) {
      return false;
    }
    return naam.equalsIgnoreCase(((Track) object).naam);
  }
  @Override
  public int hashCode() {
    return naam.toLowerCase().hashCode();
  }
}
```

2.13.8 AbstractRepository

zelfde als in theorie

2.13.9 AlbumRepository

```
package be.vdab.repositories;
// enkele imports ...
public class AlbumRepository extends AbstractRepository {
  public List<Album> findAll() {
    return getEntityManager().createNamedQuery("Album.findAll", Album.class)
      .setHint("javax.persistence.loadgraph",
        getEntityManager().createEntityGraph(Album.MET_ARTIEST))
      .getResultList();
  }
  public Optional<Album> read(long id) {
    return Optional.ofNullable(getEntityManager().find(Album.class, id));
  }
}
```

2.13.10 AbstractService

zelfde als in theorie

2.13.11 AlbumService

```
package be.vdab.services;
// enkele imports ...
public class AlbumService extends AbstractService {
  private final AlbumRepository albumRepository = new AlbumRepository();
  public List<Album> findAll(){
    return albumRepository.findAll();
  }
```

```
  public Optional<Album> read(long id){
    return albumRepository.read(id);
  }
}
```

2.13.12 IndexServlet

```
package be.vdab.servlets;
// enkele imports ...
@WebServlet("/index.htm")
public class IndexServlets extends HttpServlet {
  private static final long serialVersionUID = 1L;
  private static final String VIEW = "/WEB-INF/JSP/index.jsp";
  private final transient AlbumService albumService = new AlbumService();
  @Override
  protected void doGet(HttpServletRequest request, HttpServletResponse response)
  throws ServletException, IOException {
    request.setAttribute("albums", albumService.findAll());
    request.getRequestDispatcher(VIEW).forward(request, response);
  }
}
```

2.13.13 index.jsp

```
<%@ page contentType='text/html' pageEncoding='UTF-8' session='false'%>
<%@taglib prefix='c' uri='http://java.sun.com/jsp/jstl/core'%>
<!doctype html>
<html lang='nl'>
<head>
<title>Albums</title>
</head>
<body>
  <h1>Albums</h1>
  <ul>
  <c:forEach var='album' items='${albums}'>
    <c:url var='url' value='/album.htm'>
      <c:param name='id' value='${album.id}'/>
    </c:url>
    <li><a href='${url}'>${album.naam}</a> ${album.artiest.naam}</li>
  </c:forEach>
  </ul>
</body>
</html>
```

2.13.14 StringUtils: zelfde als in theorie

2.13.15 AlbumServlet

```
package be.vdab.servlets.albums;
// enkele imports ...
@WebServlet("/album.htm")
public class AlbumServlet extends HttpServlet {
  private static final long serialVersionUID = 1L;
  private static final String VIEW = "/WEB-INF/JSP/album.jsp";
  private final transient AlbumService albumService = new AlbumService();
  @Override
  protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    String idString = request.getParameter("id");
    if (Stringutils.isLong(idString)){
        Optional<Album> optionalAlbum =
          albumService.read(Long.parseLong(idString));
        if (optionalAlbum.isPresent()) {
          request.setAttribute("album", optionalAlbum.get());
          request.getRequestDispatcher(VIEW).forward(request, response);
```

```
        } else {
          naarHomePage(request, response);
        }
    } else  {
      naarHomePage(request, response);
    }
  }
  private void naarHomePage(HttpServletRequest request,
    HttpServletResponse response) throws IOException {
    response.sendRedirect(response.encodeRedirectURL(request.getContextPath()));
  }
}
```

2.13.16 album.jsp

```
<%@ page contentType='text/html' pageEncoding='UTF-8' session='false'%>
<%@taglib prefix='c' uri='http://java.sun.com/jsp/jstl/core'%>
<!doctype html>
<html lang='nl'>
<head>
<title>${album.naam} - ${album.artiest.naam}</title>
</head>
<body>
  <h1>${album.naam} - ${album.artiest.naam}</h1>
  <ul>
  <c:forEach var="track" items="${album.tracks}">
    <li>${track.naam} ${track.tijd}</li>
  </c:forEach>
  </ul>
  Totale tijd: ${album.tijd}
</body>
</html>
```

3 COLOFON

Domeinexpertisemanager: Jean Smits

Moduleverantwoordelijke: Hans Desmet

Medewerkers: Hans Desmet

Versie: 16/8/2017

Nummer dotatielijst: