



Sagesse Devoir
Ecole Polytechnique Thiès



Sagesse Devoir
Ecole Polytechnique Thiès



République du Sénégal
Un Peuple – Un But – Une Foi
Ministère l'Enseignement Supérieur de la Recherche et
de l'Innovation
École Polytechnique de Thiès
B.P.A 10 Thiès
Tél: (221) 76 223 61 74 – Fax: (221) 33 951 14 67

RAPPORT SUR JAKARTAEE

Présenté par :

Mohamed Massamba SENE

Professeur
Dr. Samba Sidibé

Matière
Développement Web 3

Table des matières

I.	Configuration des serveurs	3
1.	Installation serveur d'application	3
2.	Installation SGBD.....	6
3.	Création source de données	12
II.	Mapping JPA.....	14
1.	Création nouveau projet	14
2.	Création des entités	17
III.	Développement des EJB et des façades	27
1.	Création des Façades	27
2.	Initialisation de la BD	34
3.	Session Bean pour l'envoi d'email	38
4.	Envoi d'alertes automatique.....	43
IV.	Développement des interfaces web JSF.....	48
1.	CRUD sur les entités	48
2.	Créer un menu	61
3.	Filtrage, triage et pagination	64
4.	Recherche avancée	68
V.	Déploiement dans un environnement de production	70
1.	Installation de l'environnement de production.....	70
2.	Déployer l'application à la racine du serveur	71

I. Configuration des serveurs

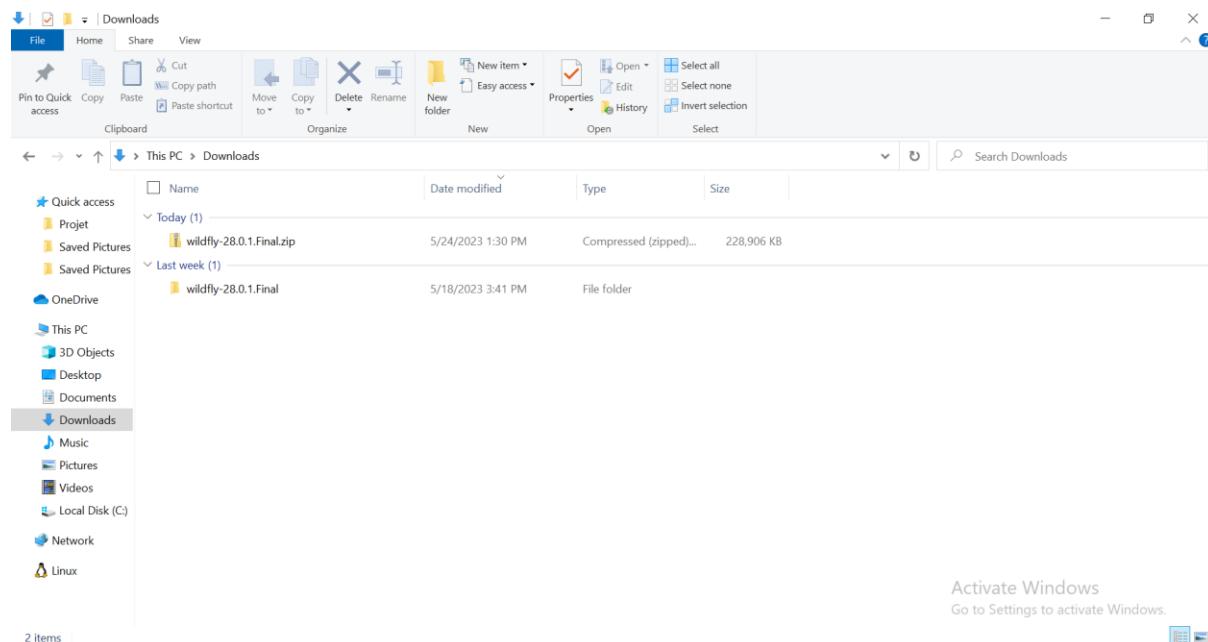
1. Installation serveur d'application

Nous choisissons WildFly autrefois connu sous le nom de JBoss Application Server qui est un serveur d'applications open source, léger et flexible qui prend en charge la plate-forme Jakarta EE (anciennement Java EE). Il est développé par la communauté JBoss et maintenu par Red Hat.

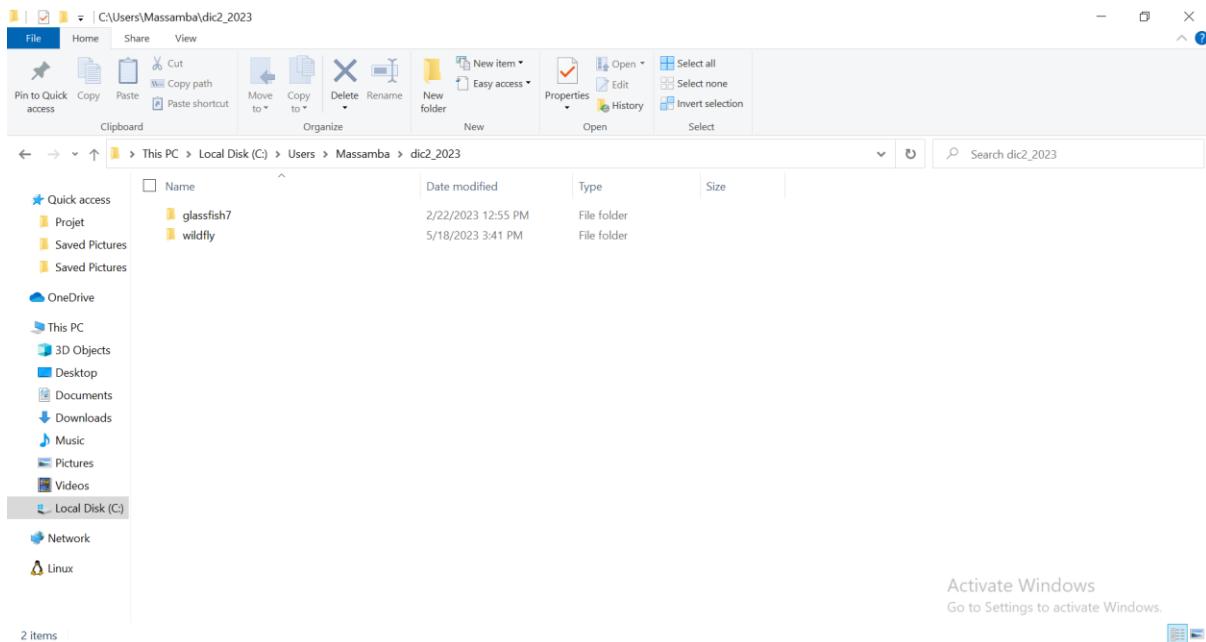
Il est entièrement compatible avec les spécifications Jakarta EE, ce qui en fait une option fiable pour le développement d'applications Java d'entreprise.

Télécharger et dézipper du dossier d'installation

On télécharge le dossier zippé au niveau du site <https://www.wildfly.org/> que l'on dézippe ensuite



On déplace le dossier dézippé que l'on a renommé au niveau du répertoire C:/Users/Massamba/dic2_2023



Créer un utilisateur

On crée un utilisateur qui sera utilisé pour accéder à la console de management afin de pouvoir gérer et configurer le serveur. Pour cela on ouvre un terminal en mode administrateur. Ensuite on se déplace au niveau de l'emplacement du dossier bin de WildFly et on exécute la commande *add-user.bat*

```
Administrator: Command Prompt
Microsoft Windows [Version 10.0.19044.2965]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\system32>cd C:\Users\Massamba\dic2_2023\wildfly\bin

C:\Users\Massamba\dic2_2023\wildfly\bin>java18
Java 18 activated.

C:\Users\Massamba\dic2_2023\wildfly\bin>add-user.bat

What type of user do you wish to add?
  a) Management User ('mgmt-users.properties')
  b) Application User ('application-users.properties')
(a):

Enter the details of the new user to add.
Using realm 'ManagementRealm' as discovered from the existing property files.
Username : massamba
Password recommendations are listed below. To modify these restrictions edit the add-user.properties configuration file.
- The password should be different from the username
- The password should not be one of the following restricted values (root, admin, administrator)
- The password should contain at least 8 characters, 1 alphabetic character(s), 1 digit(s), 1 non-alphanumeric symbol(s)
Password :
Re-enter Password :
What groups do you want this user to belong to? (Please enter a comma separated list, or leave blank for none)[ ]:
About to add user 'massamba' for realm 'ManagementRealm'
Is this correct yes/no? yes
Added user 'massamba' to file 'C:\Users\Massamba\dic2_2023\wildfly\standalone\configuration\mgmt-users.properties'
Added user 'massamba' to file 'C:\Users\Massamba\dic2_2023\wildfly\domain\configuration\mgmt-users.properties'
Added user 'massamba' with groups to file 'C:\Users\Massamba\dic2_2023\wildfly\standalone\configuration\mgmt-groups.properties'
Added user 'massamba' with groups to file 'C:\Users\Massamba\dic2_2023\wildfly\domain\configuration\mgmt-groups.properties'
Press any key to continue . . .

C:\Users\Massamba\dic2_2023\wildfly\bin>
```

Démarrage de WildFly

On exécute la commande *standalone.bat*

```

Administrator: Command Prompt - standalone.bat
C:\Users\Massamba\dic2_2023\wildfly\bin>standalone.bat
Calling "C:\Users\Massamba\dic2_2023\wildfly\bin\standalone.conf.bat"
Setting JAVA property to "C:\Program Files\Java\jdk-18.0.2.1\bin\java"
=====
JBoss Bootstrap Environment
JBoss_HOME: "C:\Users\Massamba\dic2_2023\wildfly"
JAVA: "C:\Program Files\Java\jdk-18.0.2.1\bin\java"

JAVA_OPTS: "-Dprogram.name=standalone.bat -Xms64M -Xmx512M -XX:MaxMetaspaceSize=96M -XX:MaxMetaspaceSize=256M -Djava.net.preferIPv4Stack=true -Djboss.modules.system.pkgs=org.jboss byteman -Djava.awt.headless=true --add-exports=java.desktop/sun.awt=ALL-UNNAMED --add-exports=java.naming/com.sun.jndi.ldap=ALL-UNNAMED --add-exports=java.naming/com.sun.jndi.url.ldap=ALL-UNNAMED --add-exports=java.naming/com.sun.jndi.url.ldap=ALL-UNNAMED --add-exports=jdk.naming.dns=com.sun.jndi.dns=ALL-UNNAMED --add-exports=java.base/java.lang.invoke=ALL-UNNAMED --add-opens=java.base/java.lang.invoke=ALL-UNNAMED --add-opens=java.base/java.lang.reflect=ALL-UNNAMED --add-opens=java.base/java.io=ALL-UNNAMED --add-opens=java.base/java.net=ALL-UNNAMED --add-opens=java.base/java.security=ALL-UNNAMED --add-opens=java.base/java.util=ALL-UNNAMED --add-opens=java.base/java.util.concurrent=ALL-UNNAMED --add-opens=java.management/javax.management=ALL-UNNAMED --add-opens=java.naming/javax.naming=ALL-UNNAMED -Djava.security.manager=allow"
=====

14:08:32,823 INFO [org.jboss.modules] (main) JBoss Modules version 2.1.0.Final
14:08:33,743 INFO [org.jboss.msc] (main) JBoss MSC version 1.5.0.Final
14:08:33,761 INFO [org.jboss.threads] (main) JBoss Threads version 2.4.0.Final
14:08:33,771 INFO [org.wildfly.security] (ServerService Thread Pool -- 28) ELY00001: WildFly Elytron version 2.1.0.Final
14:08:36,512 INFO [org.jboss.as.controller.management-deprecated] (ServerService Thread Pool -- 12) WFLYCTL0028: Attribute 'cluster' in the resource at address '/subsystem=ejb3/service=remote' is deprecated, and may be removed in a future version. See the attribute description in the output of the read-resource-description operation to learn more about the deprecation.
14:08:36,628 INFO [org.jboss.as.server] (Controller Boot Thread) WFLYSRV0039: Creating http management service using socket-binding (management-http)
14:08:36,661 INFO [org.xnio] (MSC service thread 1-5) XNIO version 3.8.9.Final
14:08:36,677 INFO [org.xnio.nio] (MSC service thread 1-5) XNIO NIO Implementation Version 3.8.9.Final
14:08:36,751 INFO [org.jboss.as.jaxrs] (ServerService Thread Pool -- 57) WFLYRS0016: RESTEasy version 6.2.4.Final
14:08:36,766 INFO [org.jboss.as.txn] (ServerService Thread Pool -- 74) WFLYT00013: The node-identifier attribute on the /subsystem=transactions is set to the default value . This is a danger for environments running multiple servers. Please make sure the attribute value is unique.
14:08:36,819 INFO [org.wildfly.extension.microprofile.jwt.smallrye] (ServerService Thread Pool -- 66) WFLYWT00001: Activating MicroProfile JWT Subsystem
14:08:36,820 INFO [org.jboss.as.clustering.infinispan] (ServerService Thread Pool -- 65) WFLYCLINF00001: Activating Infinispan subsystem.
14:08:36,823 INFO [org.wildfly.extension.elytron.oidc._private] (ServerService Thread Pool -- 53) WFLYOIDC00001: Activating WildFly Elytron OIDC Subsystem
14:08:36,821 INFO [org.wildfly.extension.microprofile.config.smallrye] (ServerService Thread Pool -- 65) WFLYCONF00001: Activating MicroProfile Config Subsystem
14:08:36,861 INFO [org.wildfly.extension.health] (ServerService Thread Pool -- 54) WFLYHEALTH0001: Activating Base Health Subsystem
14:08:36,876 INFO [org.jboss.as.jsf] (ServerService Thread Pool -- 62) WFLYJSF0007: Activated the following Jakarta Server Faces Implementations: [main]
=====
Administrator: Command Prompt - standalone.bat
14:08:36,861 INFO [org.wildfly.extension.health] (ServerService Thread Pool -- 54) WFLYHEALTH0001: Activating Base Health Subsystem
14:08:36,876 INFO [org.jboss.as.jsf] (ServerService Thread Pool -- 62) WFLYJSF0007: Activated the following Jakarta Server Faces Implementations: [main]
14:08:36,932 INFO [org.wildfly.extension.metrics] (ServerService Thread Pool -- 64) WFLYMETRICS00001: Activating Base Metrics Subsystem
14:08:36,936 INFO [org.jboss.as.webservices] (ServerService Thread Pool -- 76) WFLYWS00002: Activating WebServices Extension
14:08:37,162 INFO [org.jboss.as.naming] (MSC service thread 1-1) WFLYNAM00003: Starting Naming Service
14:08:37,170 INFO [org.jboss.as.connector] (MSC service thread 1-1) WFLYJCA00009: Starting Jakarta Connectors Subsystem (WildFly/IronJacamar 3.0.2.Final)
14:08:37,262 INFO [org.wildfly.extension.undertow] (MSC service thread 1-3) WFLYUT00003: Undertow 2.3.6.Final starting
14:08:37,286 INFO [org.jboss.as.connector.subsystems.datasources] (ServerService Thread Pool -- 44) WFLYJCA0004: Deploying JDBC-compliant driver class org.h2.Driver (version 2.1)
14:08:37,416 INFO [org.jboss.as.connector.deployers.jdbc] (MSC service thread 1-3) WFLYJCA0018: Started Driver service with driver-name = h2
14:08:37,522 WARN [org.wildfly.extension.elytron] (MSC service thread 1-3) WFLYELEY00023: KeyStore file 'C:\Users\Massamba\dic2_2023\wildfly\standalone\configuration\application.keystore' does not exist. Used blank.
14:08:37,567 INFO [org.wildfly.extension.undertow] (ServerService Thread Pool -- 75) WFLYUT0014: Creating file handler for path 'C:\Users\Massamba\dic2_2023\wildfly\welcome-content' with options [directory-listing: 'false', follow-symlink: 'false', case-sensitive: 'true', safe-symlink-paths: '[ ]']'
14:08:37,658 WARN [org.wildfly.extension.elytron] (MSC service thread 1-3) WFLYELEY0084: KeyStore C:\Users\Massamba\dic2_2023\wildfly\standalone\configuration\application.keystore not found, it will be auto-generated on first use with a self-signed certificate for host localhost
14:08:37,927 INFO [org.jboss.as.mail.extension] (MSC service thread 1-6) WFLYMAIL00001: Bound mail session [java:jboss/mail/Default]
14:08:38,128 INFO [org.wildfly.extension.io] (ServerService Thread Pool -- 56) WFLYI00001: Worker 'default' has auto-configured to 16 IO threads with 128 max task threads based on your 8 available processors
14:08:38,231 INFO [org.jboss.as.ejb3] (MSC service thread 1-6) WFLYEJB0482: Strict pool mdb-strict-max-pool is using a max instance size of 32 (per class), which is derived from the number of CPUs on this host.
14:08:38,231 INFO [org.jboss.as.ejb3] (MSC service thread 1-4) WFLYEJB0481: Strict pool slsb-strict-max-pool is using a max instance size of 128 (per class), which is derived from the thread worker pool sizing.
14:08:38,492 INFO [org.wildfly.extension.undertow] (MSC service thread 1-7) WFLYUT0012: Started server default-server.
14:08:38,497 INFO [org.wildfly.extension.undertow] (MSC service thread 1-5) Queuing requests.
14:08:38,510 INFO [org.jboss.remoting] (MSC service thread 1-2) JBoss Remoting version 5.0.27.Final
14:08:38,512 INFO [org.wildfly.extension.undertow] (MSC service thread 1-5) WFLYUT0018: Host default-host starting
14:08:38,791 INFO [org.wildfly.extension.undertow] (MSC service thread 1-4) WFLYUT0006: Undertow HTTP listener default listening on 127.0.0.1:8080
14:08:38,793 INFO [org.wildfly.extension.undertow] (MSC service thread 1-6) WFLYUT0006: Undertow HTTPS listener https listening on 127.0.0.1:8443
14:08:39,640 INFO [org.jboss.as.ejb3] (MSC service thread 1-6) WFLYEJB0493: Jakarta Enterprise Beans subsystem suspension complete
14:08:39,121 INFO [org.jboss.as.patching] (MSC service thread 1-8) WFLYPAT0050: WildFly Full cumulative patch ID is: base, one-off patches include: none
14:08:39,138 INFO [org.jboss.as.server.deployment.scanner] (MSC service thread 1-8) WFLYDS0013: Started FileSystemDeploymentService for directory C:\Users\Massamba\dic2_2023\wildfly\standalone\deployments
14:08:39,189 INFO [org.jboss.as.connector.subsystems.datasources] (MSC service thread 1-6) WFLYJCA00001: Bound data source [java:jboss/datasources/ExampleDS]
14:08:39,259 INFO [org.jboss.ws.common.management] (MSC service thread 1-3) JBSW02052: Starting JBossWS 6.2.0.Final (Apache CXF 4.0.0)
14:08:39,523 INFO [org.jboss.as.server] (Controller Boot Thread) WFLYSRV0212: Resuming server
14:08:39,542 INFO [org.jboss.as] (Controller Boot Thread) WFLYSRV0060: Http management interface listening on http://127.0.0.1:9990
14:08:39,543 INFO [org.jboss.as] (Controller Boot Thread) WFLYSRV0051: Admin console listening on http://127.0.0.1:9990
14:08:39,546 INFO [org.jboss.as] (Controller Boot Thread) WFLYSRV0025: WildFly Full 28.0.1.Final (WildFly Core 20.0.2.Final) started in @342ms@ [Started: 280ms@/522ms] services (317 services are lazy, passive or on-demand) - Server configuration file in use: standalone.xml
=====

Activate Windows

```

On peut ensuite accéder à la console de management où l'on peut gérer et configurer le serveur à l'adresse : <http://127.0.0.1:9990/>

The screenshot shows two browser windows side-by-side. The left window is titled 'HAL Management Console' and displays a login dialog box. The right window is titled 'desktop-tio1vcf | Management C' and displays the WildFly Application Server management interface.

HAL Management Console (Left Window):

- Title bar: HAL Management Console
- Address bar: localhost:9990/console/index.html
- Content: A 'Se connecter' (Connect) dialog box with fields for 'Nom d'utilisateur' (massamba) and 'Mot de passe' (redacted). Buttons: 'Se connecter' (Connect) and 'Annuler' (Cancel).
- Bottom right corner: 'Activate Windows' message with a link to 'Go to Settings to activate Windows.'

WildFly Application Server (Right Window):

- Title bar: desktop-tio1vcf | Management C
- Address bar: localhost:9990/console/index.html
- Content: A navigation menu with tabs: Homepage, Deployments, Configuration, Runtime, Patching, Access Control. The 'Homepage' tab is selected.
- Sub-sections:
 - Deployments:** Add and manage deployments. Sub-links: Deploy an Application (with Start button), Monitor the Server (with Start button), and View log files or JVM usage.
 - Configuration:** Configure subsystem settings. Sub-links: Create a Data source (with Start button), Assign User Roles (with Start button), and Manage user and group permissions for management operations.
 - Runtime:** Monitor server status. Sub-links: Monitor the Server (with Start button) and View runtime information such as server status, JVM status, and server log files.
 - Access Control:** Assign roles to users or groups to determine access to system resources. Sub-links: Assign User Roles (with Start button) and Manage user and group permissions for management operations.
- Bottom right corner: 'Activate Windows' message with a link to 'Go to Settings to activate Windows.'
- Bottom right corner: Version information: 3.6.5.Final, Tools, and Settings.

2. Installation SGBD

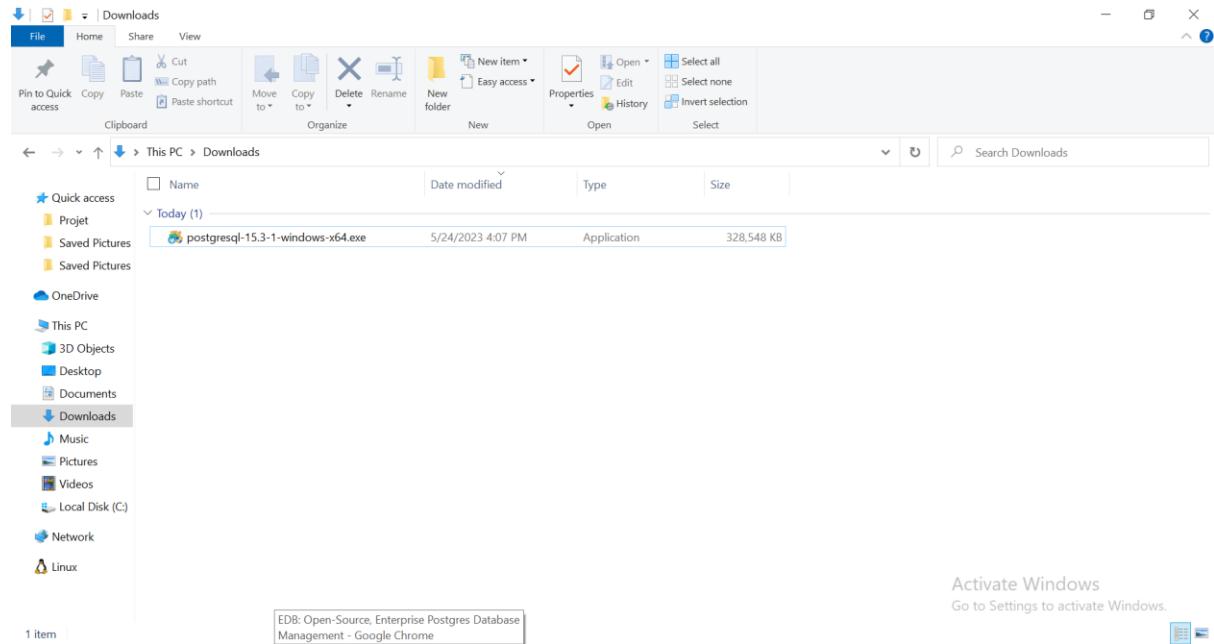
Nous installons donc installer Postgres. PostgreSQL, également connu sous le nom de Postgres, est un puissant système de gestion de base de données relationnelle objet open source (ORDBMS). Il a été développé à l'origine à l'Université de Californie à Berkeley dans les années 1980 et est depuis devenu l'un des systèmes de base de données les plus populaires et les plus utilisés.

PostgreSQL est un système de base de données mature et riche en fonctionnalités qui excelle en termes de fiabilité, d'extensibilité et de performances. Sa combinaison de fonctionnalités avancées, d'évolutivité et de

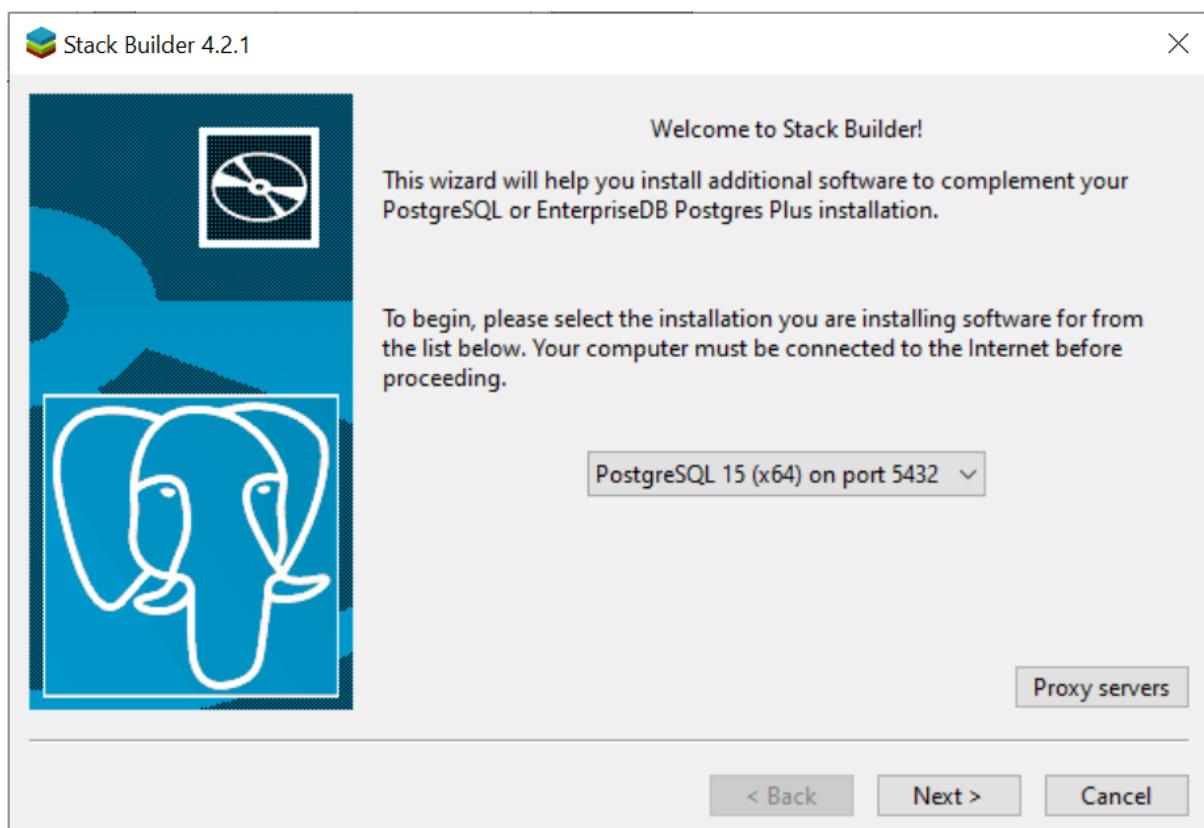
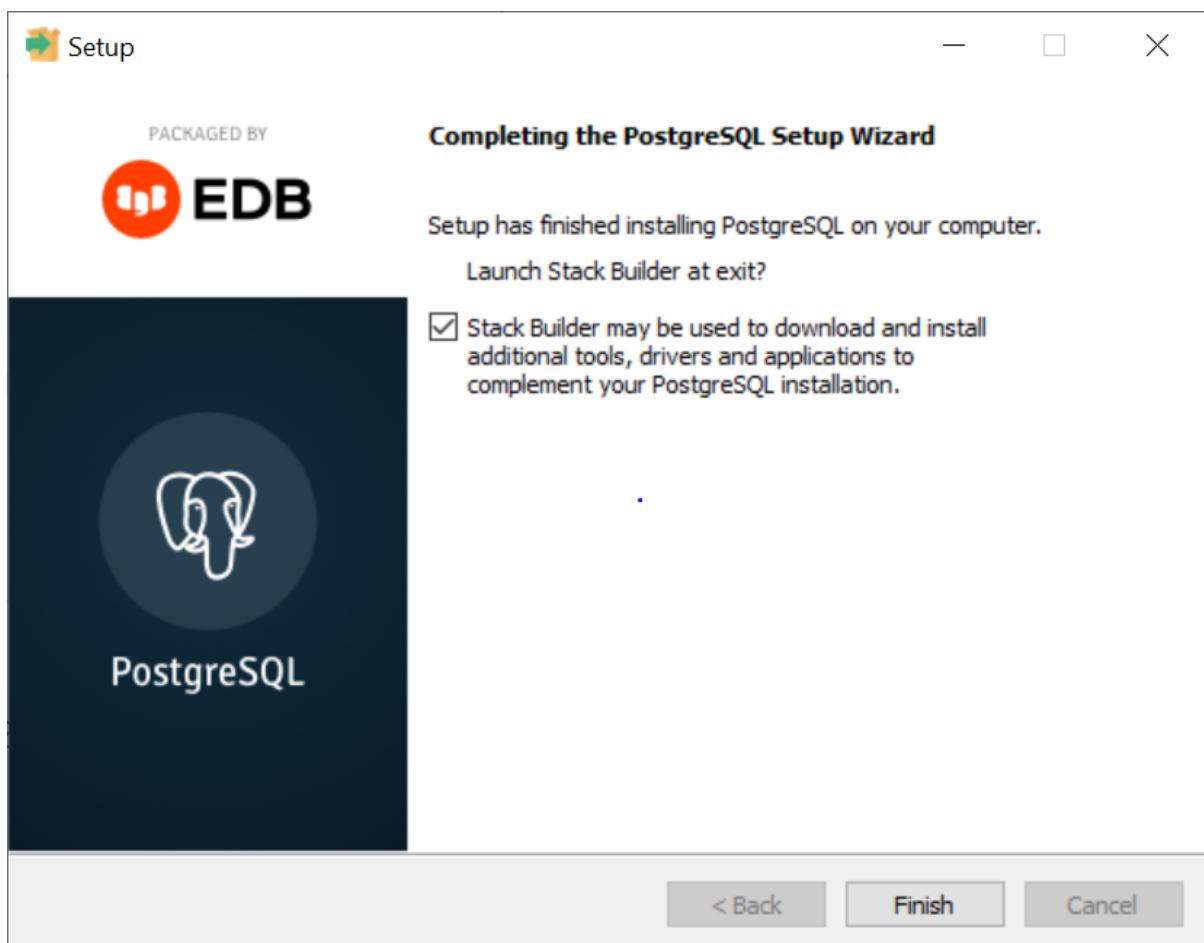
nature open source en fait un choix populaire pour une large gamme d'applications, des petits projets aux déploiements au niveau de l'entreprise.

On télécharge l'installer pour Postgres au niveau du site

<https://www.enterprisedb.com/downloads/postgres-postgresql-downloads>



Une fois téléchargée il ne reste plus qu'à l'exécuter et suivre les étapes afin d'installer Postgres sur notre ordinateur.





X

Please select the applications you would like to install.

- Categories
 - + Add-ons, tools and utilities
 - + Database Drivers
 - Npgsql v3.2.6-3
 - pgJDBC v42.5.1-1
 - psqlODBC (32 bit) v13.02.0000-1
 - psqlODBC (64 bit) v13.02.0000-1
 - + Database Server
 - + Registration-required and trial products
 - + Spatial Extensions
 - + Web Development

The official PostgreSQL ODBC driver (64bit version). Packaged by EnterpriseDB.

< Back

Next >

Cancel



X

All the installation files have now been successfully downloaded.

Please click the "Next" button to start the installations.

Note: You must allow all installations to run to completion. If you are prompted to restart the computer, click "No" or "Restart Later" and manually restart your computer when all the installation have finished.

Skip Installation

< Back

Next >

Cancel



Setup

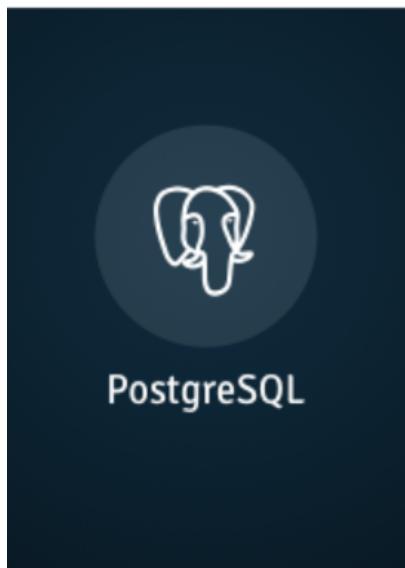


PACKAGED BY



Setup psqlODBC

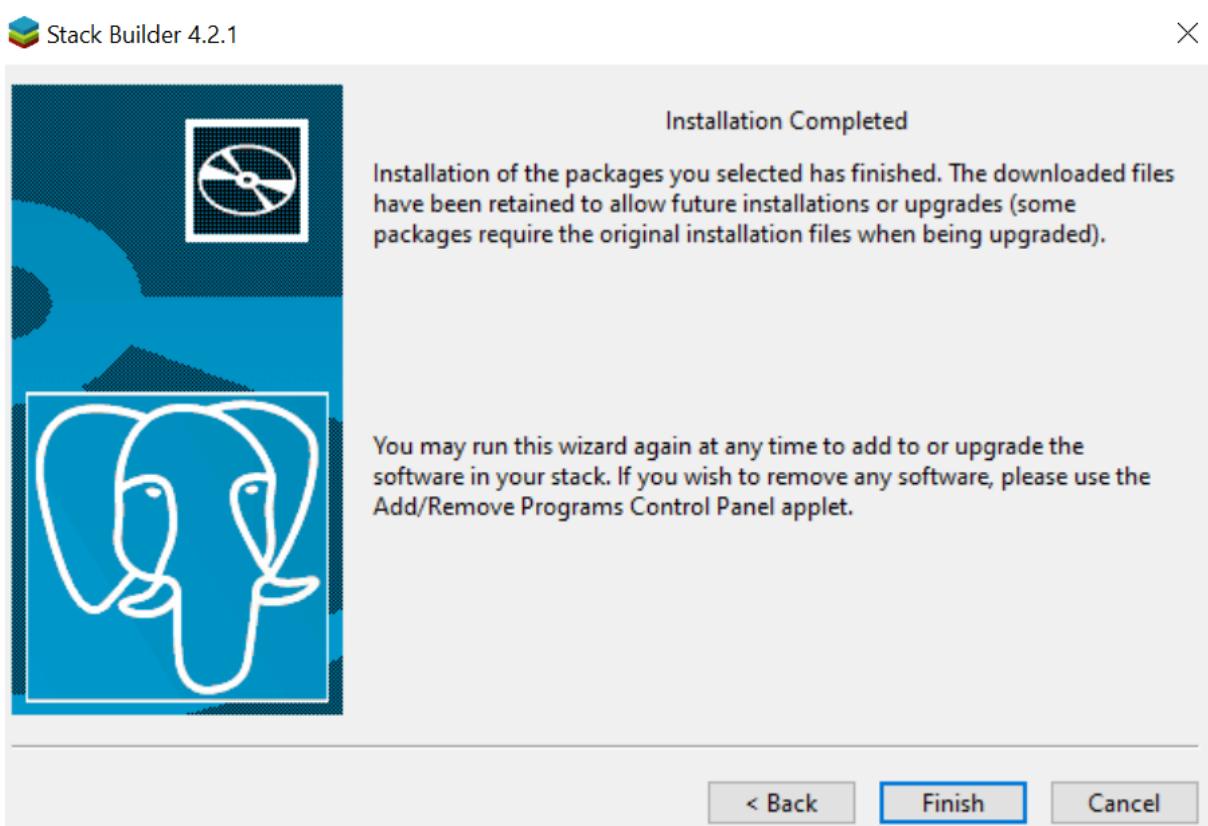
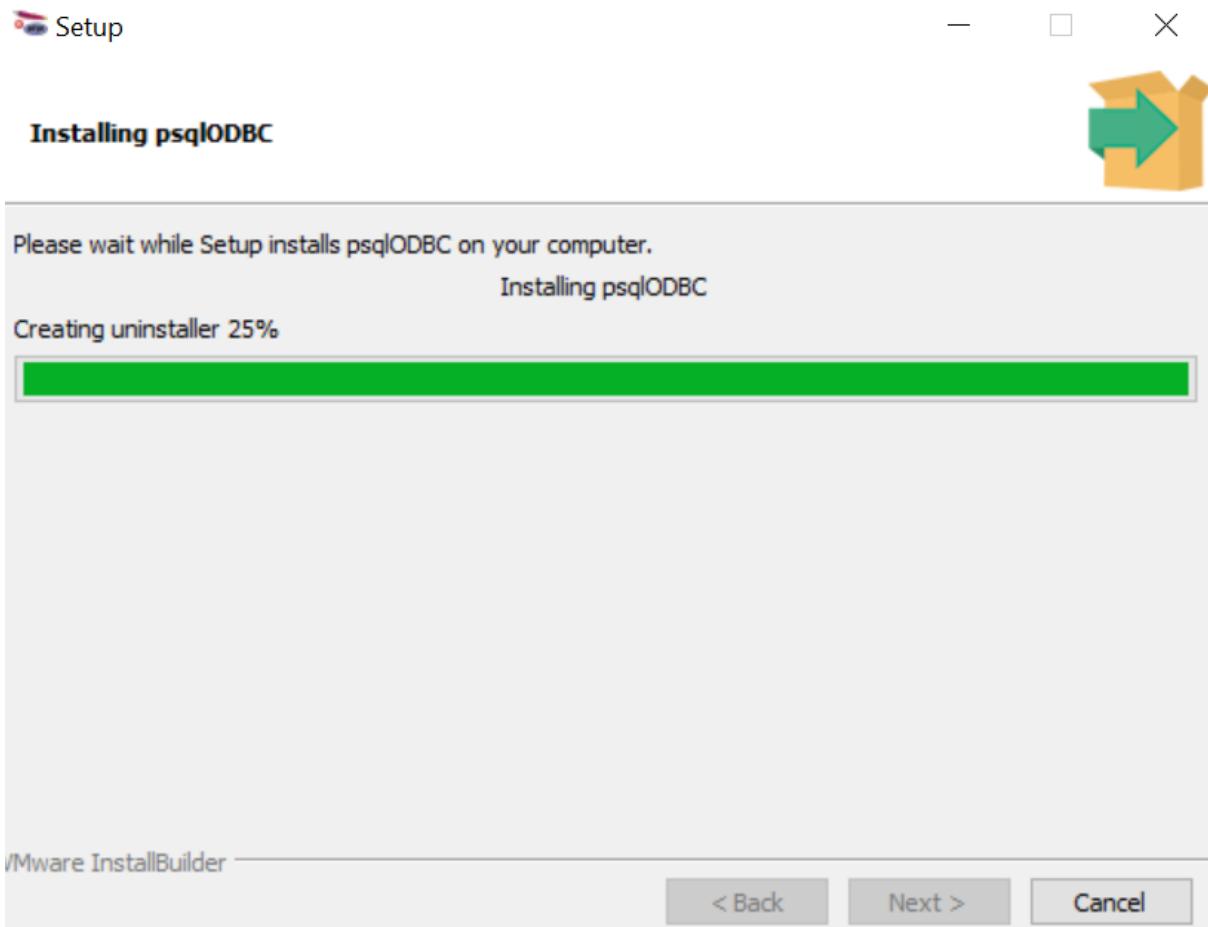
Welcome to the psqlODBC Setup Wizard.



< Back

Next >

Cancel



```

SQL Shell (psql)
Database [postgres]:
Port [5432]:
Username [postgres]:
Password for user postgres:
pgsql (15.3)
WARNING: Console code page (850) differs from Windows code page (1252)
         8-bit characters might not work correctly. See psql reference
         page "Notes for Windows users" for details.
Type "help" for help.

postgres=# \l
      List of databases
   Name | Owner | Encoding |           Collate           |           Ctype           | ICU Locale | Locale Provide
   +---+-----+-----+-----+-----+-----+-----+
postgres | postgres | UTF8    | English_United States.1252 | English_United States.1252 |          | libc
|       |
template0 | postgres | UTF8    | English_United States.1252 | English_United States.1252 |          | libc
| =c/postgres      +
|       |
| postgres=CTc/postgres
template1 | postgres | UTF8    | English_United States.1252 | English_United States.1252 |          | libc
| =c/postgres      +
|       |
| postgres=CTc/postgres
(3 rows)

postgres=

```

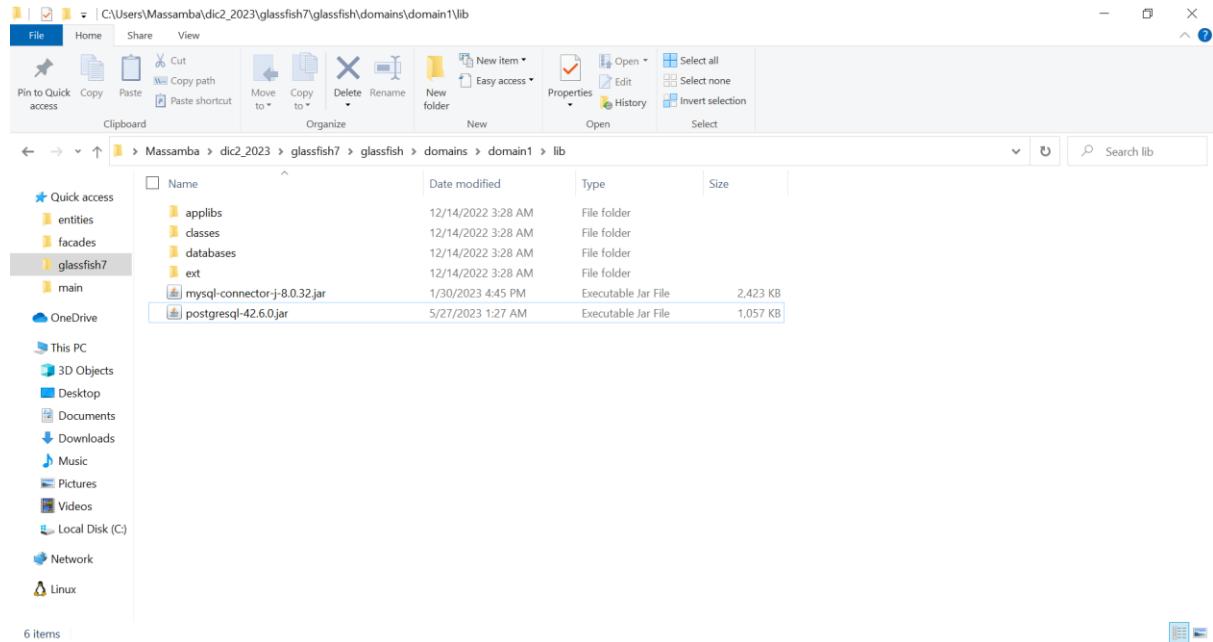
3. Création source de données

Pour créer une source de données en utilisant Postgres et Glassfish on procède en suivant les étapes :

Télécharger le PostgreSQL JDBC driver

On télécharge le driver à partir du site <https://jdbc.postgresql.org/download/> et on le place dans le fichier .jar au niveau de

C:\Users\Massamba\dic2_2023\glassfish7\glassfish\domains\domain1\lib



Créer une base de données sur Postgres pour la vente de vélos

```
SQL Shell (psql)
psql (15.3)
WARNING: Console code page (850) differs from Windows code page (1252)
          8-bit characters might not work correctly. See psql reference
          page "Notes for Windows users" for details.
Type "help" for help.

postgres=# CREATE DATABASE ventes_velos;
CREATE DATABASE
postgres=# \l
      List of databases
   Name    | Owner     | Encoding | Collate           | Ctype            | ICU Locale | Locale Prov
   ider | Access privileges
-----+-----+-----+-----+-----+-----+-----+
 postgres | postgres | UTF8   | English_United States.1252 | English_United States.1252 | libc        |
 |       |          |          |          |          |          |
 template0 | postgres | UTF8   | English_United States.1252 | English_United States.1252 | libc        |
 | =c/postgres |          +          |          |          |          |          |
 |          |          |          |          |          |          |
 |          |          |          |          |          |          |
 template1 | postgres | UTF8   | English_United States.1252 | English_United States.1252 | libc        |
 | =c/postgres |          +          |          |          |          |          |
 |          |          |          |          |          |          |
 |          |          |          |          |          |          |
 ventes_velos | postgres | UTF8   | English_United States.1252 | English_United States.1252 | libc        |
 |          |          |          |          |          |          |
(4 rows)

postgres=#

```

Créer la source de données

```
C:\Users\Massamba\dic2_2023\glassfish7\glassfish\bin>asadmin create-jdbc-connection-pool --datasourceclassname org.postgresql.ds.PGConnectionPoolDataSource --restype javax.sql.ConnectionPoolDataSource --property user=postgres:password=password:DatabaseName=ventes_velos:ServerName=localhost:port=5432:allowPublicKeyRetrieval=true:useSSL=false veloPool
JDBC connection pool veloPool created successfully.
Command create-jdbc-connection-pool executed successfully.

C:\Users\Massamba\dic2_2023\glassfish7\glassfish\bin>asadmin ping-connection-pool veloPool
CLT031: Warning: Option "target" is obsolete and will be ignored.
CLT031: Warning: Option "target" is obsolete and will be ignored.
Command ping-connection-pool executed successfully.

C:\Users\Massamba\dic2_2023\glassfish7\glassfish\bin>
C:\Users\Massamba\dic2_2023\glassfish7\glassfish\bin>asadmin create-jdbc-resource --connectionpoolid veloPool jdbc/velo
JDBC resource jdbc/velo created successfully.
Command create-jdbc-resource executed successfully.

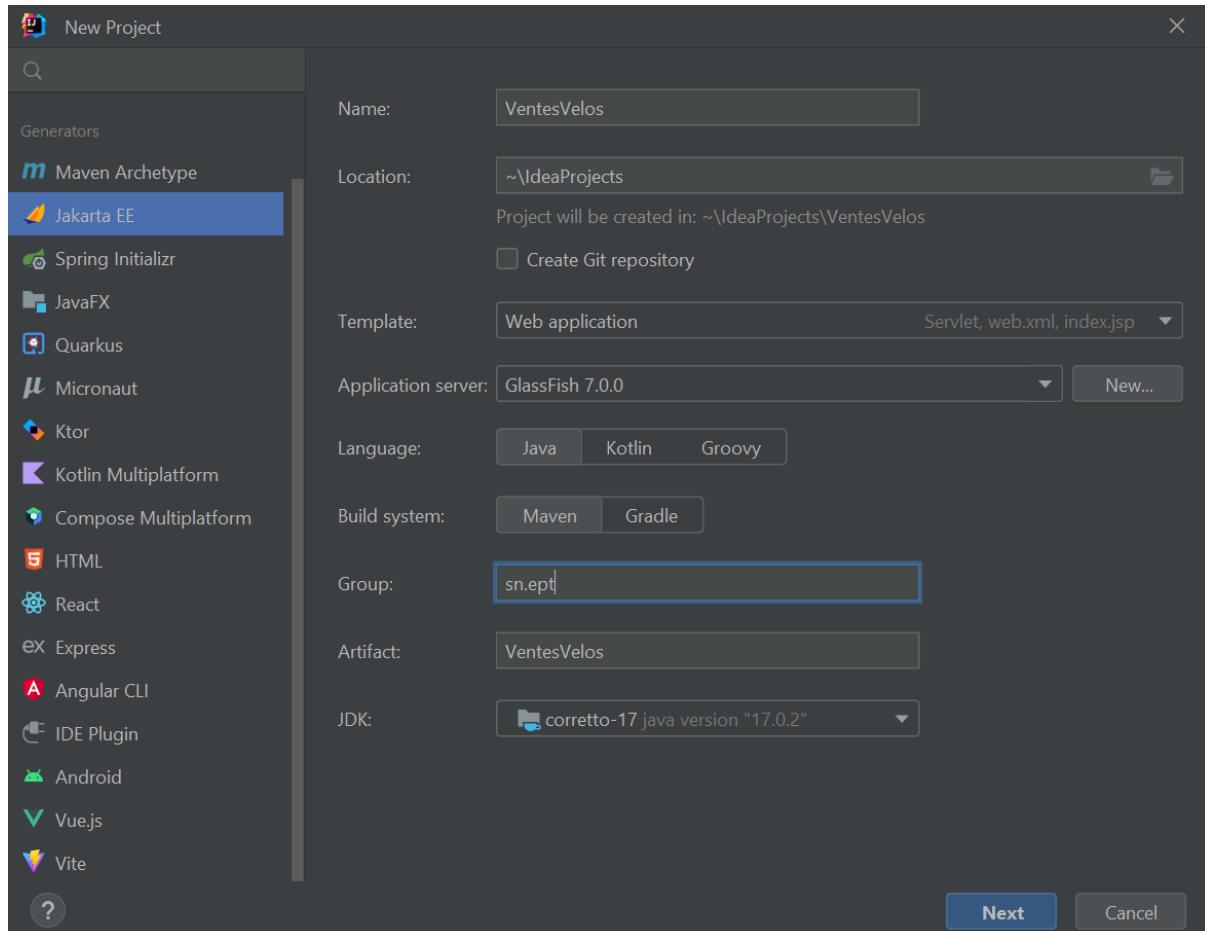
C:\Users\Massamba\dic2_2023\glassfish7\glassfish\bin>
```

On voit donc que le pool est correctement créé et que la connexion est effectivement établie étant donné que le ping fonctionne donc on pourra l'utiliser dans notre projet.

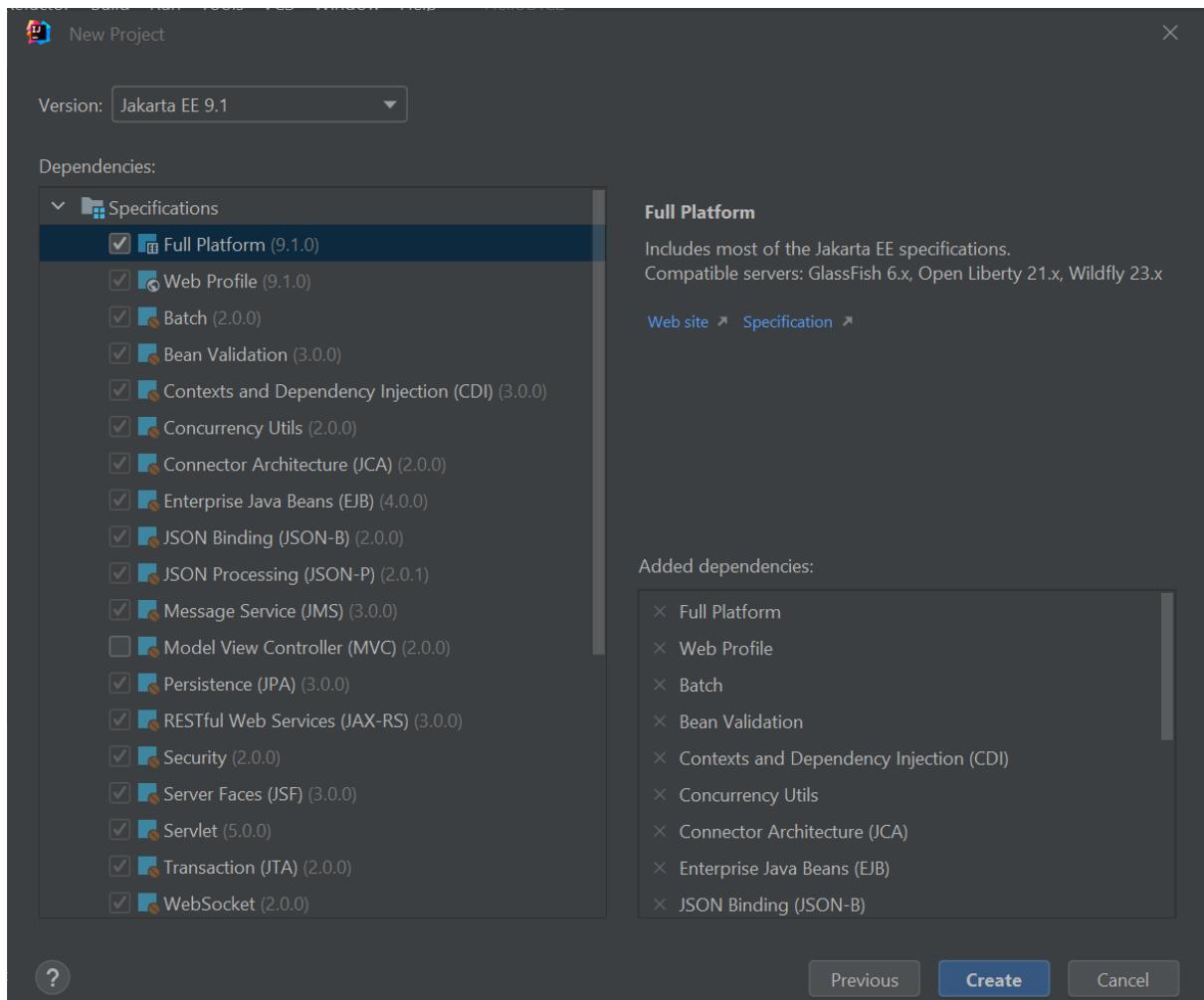
II. Mapping JPA

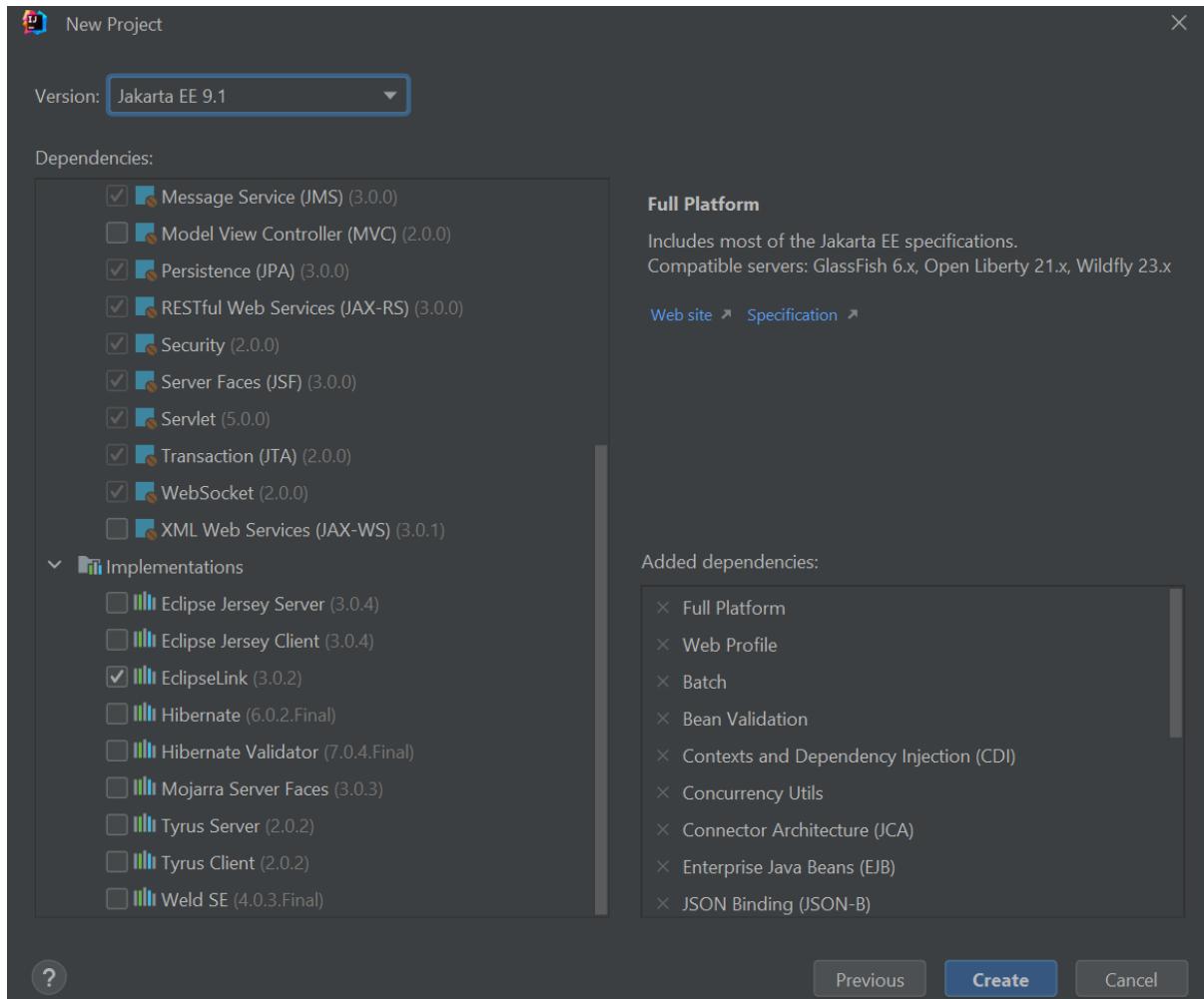
1. Crédation nouveau projet

On crée un nouveau projet JakartaEE intitulé VentesVelos en choisissant Glassfish comme serveur d'application.



On choisit l'option full platform pour disposer de toutes les spécifications JakartaEE ainsi que EclipseLink





Ajout du driver dans le pom.xml

Pour établir la connexion à la base de données, nous avons besoin du driver postgres ainsi nous ajoutons la dépendance dans notre pom.xml

```

<dependency>
    <groupId>org.junit.jupiter</groupId>
    <artifactId>junit-jupiter-engine</artifactId>
    <version>$junit.version</version>
    <scope>test</scope>
</dependency>
<dependency>
    <groupId>org.postgresql</groupId>
    <artifactId>postgresql</artifactId>
    <version>42.6.0</version>
</dependency>
</dependencies>

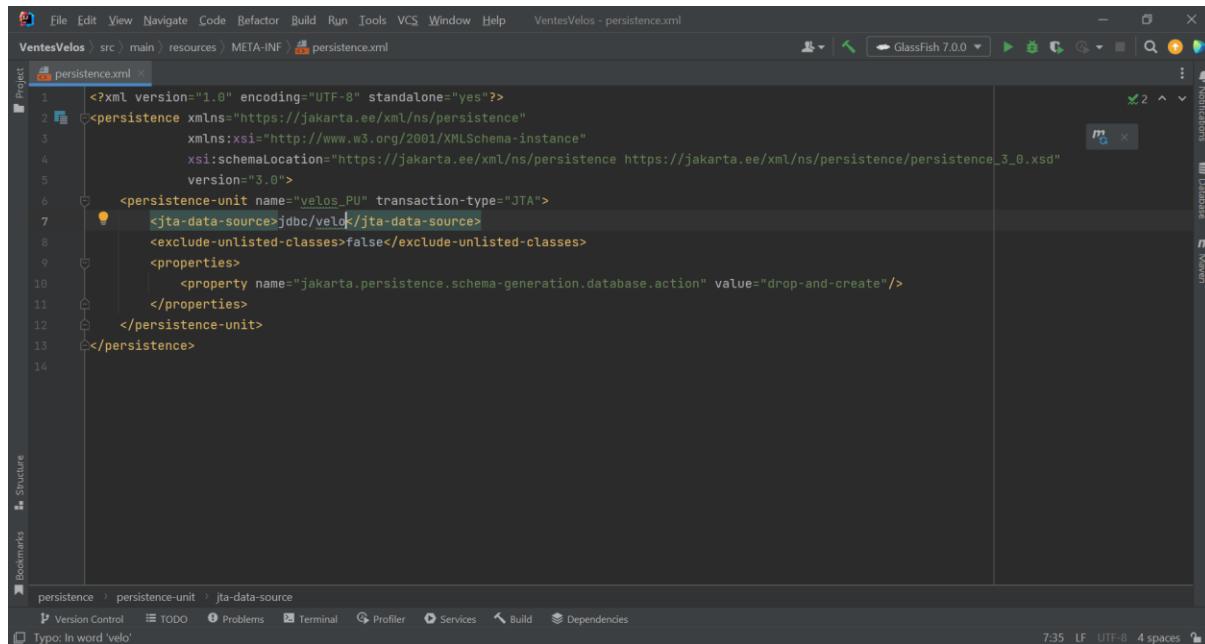
<build>
    <plugins>
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-war-plugin</artifactId>
            <version>3.3.2</version>
        </plugin>
    </plugins>
</build>
</project>

```

Ajout de la persistence unit dans persistence.xml

Nous créons une persistence unit dans le fichier persistence.xml qui va être utilisé pour le mappage des entités, la connexion à la base de données, les transactions...

On renseigne notre pool précédemment créé au niveau de l'élément *jta-data-source* et on utilise le *drop-and-create* afin de réinitialiser la base de données à chaque déploiement de notre application étant donné qu'on est en environnement de développement.

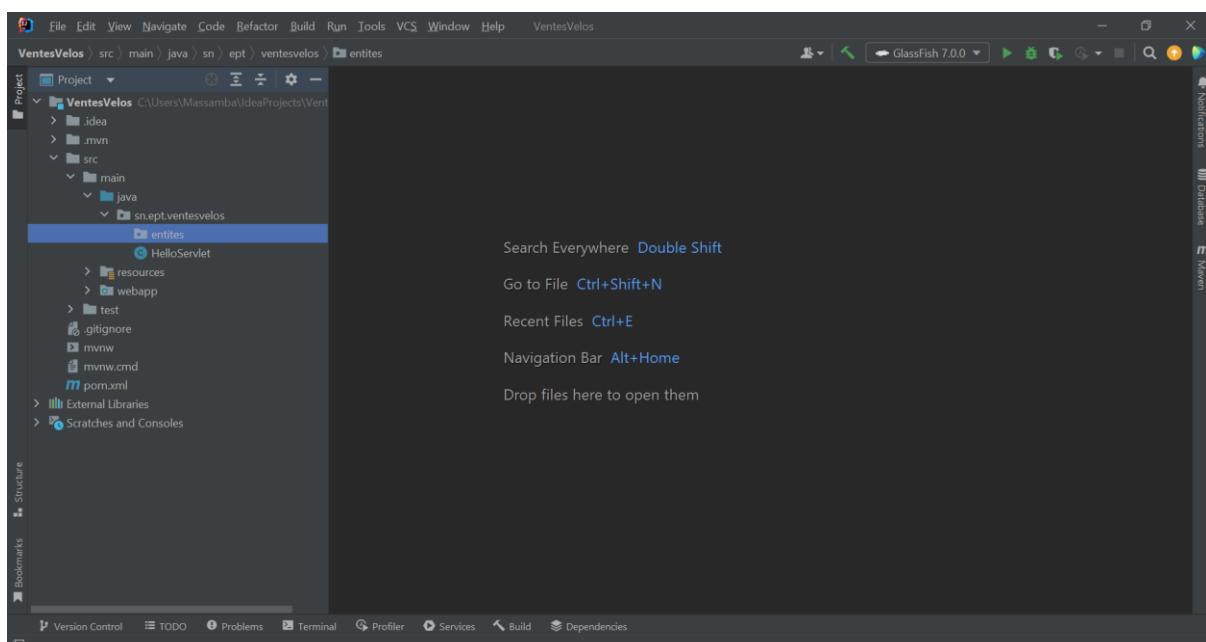
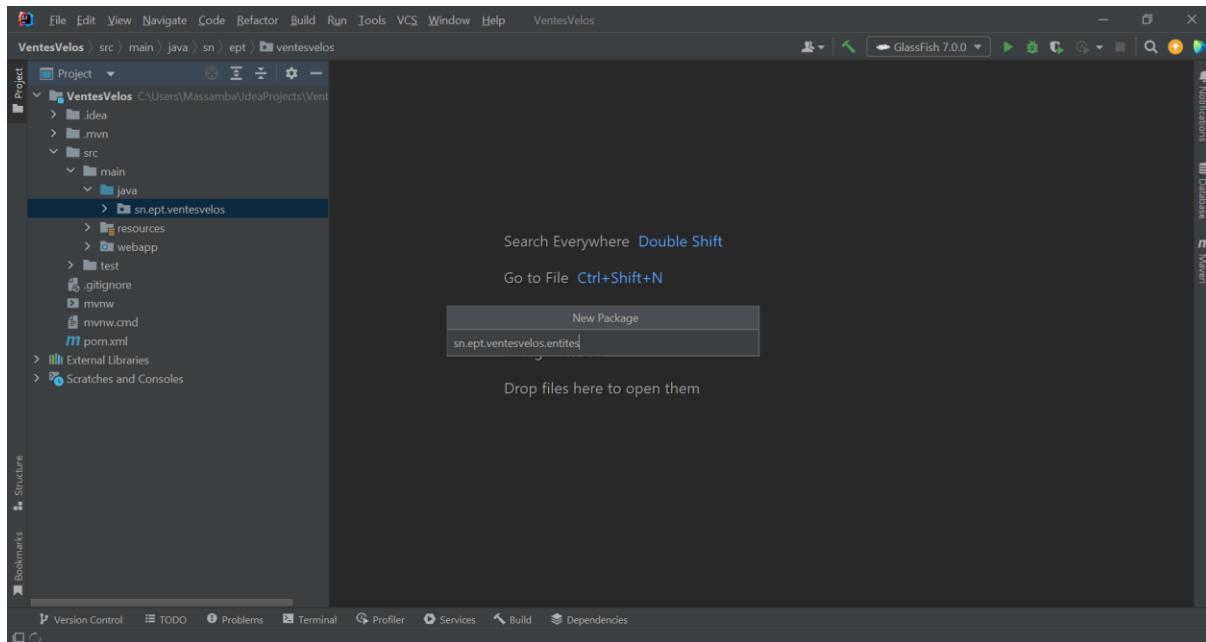


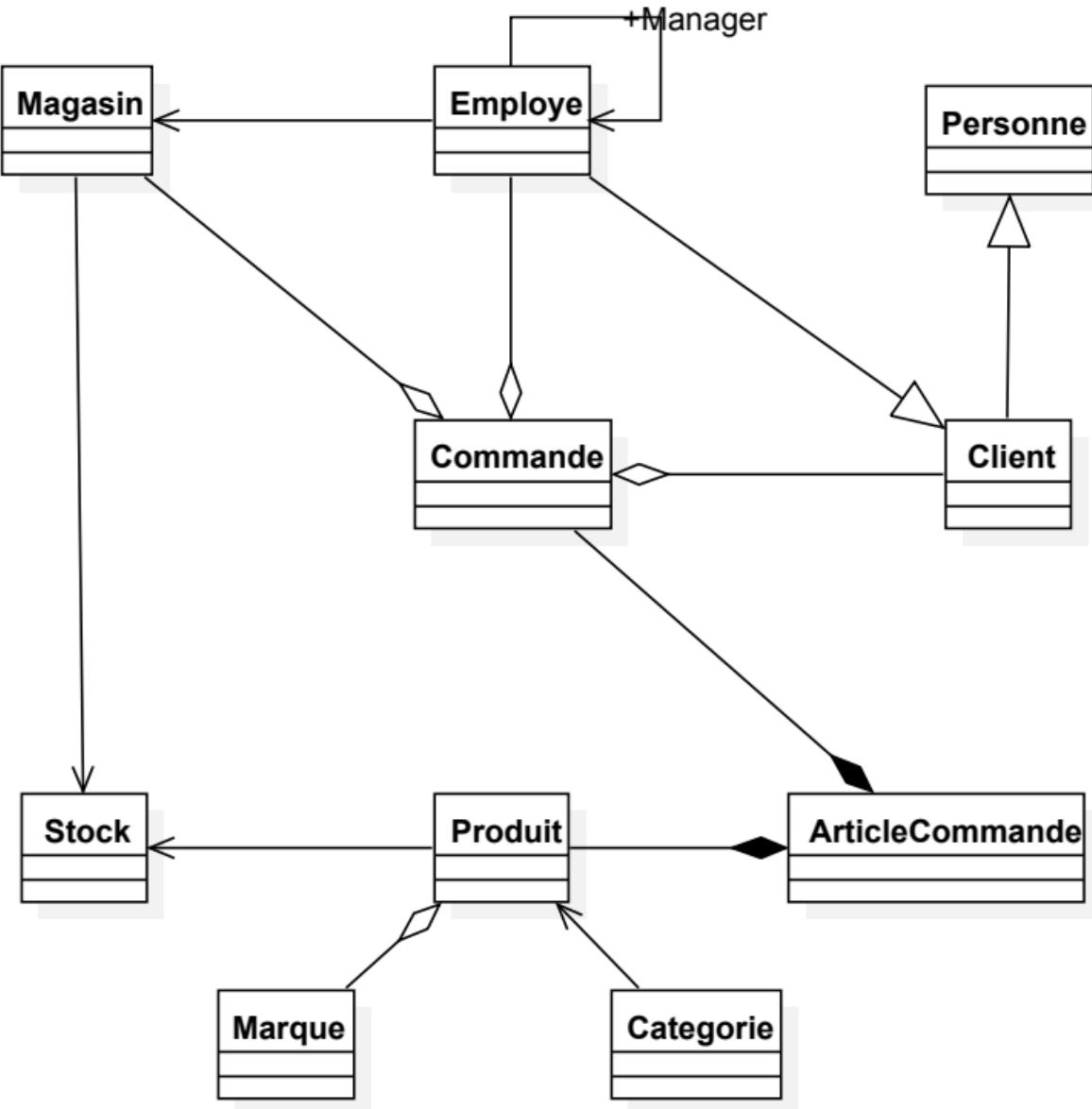
```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<persistence xmlns="https://jakarta.ee/xml/ns/persistence"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="https://jakarta.ee/xml/ns/persistence https://jakarta.ee/xml/ns/persistence_3_0.xsd"
    version="3.0">
    <persistence-unit name="velos_PU" transaction-type="JTA">
        <jta-data-source>jdbc/velo</jta-data-source>
        <exclude-unlisted-classes>false</exclude-unlisted-classes>
        <properties>
            <property name="jakarta.persistence.schema-generation.database.action" value="drop-and-create"/>
        </properties>
    </persistence-unit>
</persistence>
```

2. Création des entités

Création d'un package entities

On crée un package entities où nous allons mettre nos différentes entités



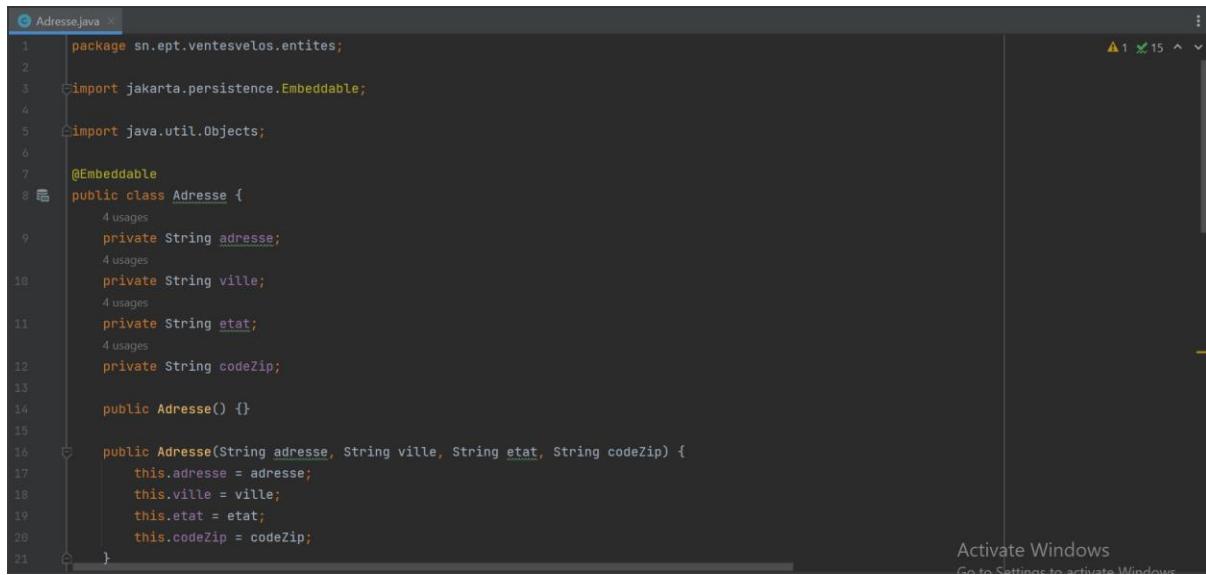


Nous suivons donc le diagramme de classe ci-dessus.

Création d'une classe Adresse

On utilise l'annotation `@Embeddable` pour spécifier qu'il s'agit d'une classe qui sera embarqué dans nos entités. On crée les attributs adresse, ville, etat et codeZip avec les getters et setters correspondants.

On ajoute également les constructeurs avec et sans arguments de même que les méthodes `equals()`, `hashCode()` et `toString()`.



The screenshot shows a code editor window for a Java file named 'Adresse.java'. The code defines a class 'Adresse' with four private string fields: 'adresse', 'ville', 'etat', and 'codeZip'. It includes a constructor that initializes these fields and a default constructor. The code uses annotations such as @Entity, @Table, @SequenceGenerator, @Id, @GeneratedValue, @Basic, and @Column. A tooltip 'Activate Windows' is visible in the bottom right corner of the IDE interface.

```
1 package sn.ept.ventesvelos.entites;
2
3 import jakarta.persistence.Embeddable;
4
5 import java.util.Objects;
6
7 @Embeddable
8 public class Adresse {
9     private String adresse;
10    private String ville;
11    private String etat;
12    private String codeZip;
13
14    public Adresse() {}
15
16    public Adresse(String adresse, String ville, String etat, String codeZip) {
17        this.adresse = adresse;
18        this.ville = ville;
19        this.etat = etat;
20        this.codeZip = codeZip;
21    }
22}
```

Pour créer les entités on utilise les annotations suivantes sur les classes :

- `@Entity` : permet de spécifier qu'il s'agit des informations de la classe et qu'on aimerait le stocker dans la base de données
- `@Table` : permet de spécifier le nom de la table dans la base de données
- `@SequenceGenerator` : permet de spécifier la stratégie de génération de la clé en conjonction avec `@GeneratedValue` sur l'id où on passe le générateur. On prend comme valeur initiale le numéro du dernier enregistrement de la base
- On crée l'attribut id avec les annotations `@Id` et `@GeneratedValue` pour spécifier l'attribut utilisé comme clé primaire de la table. On utilise également `@Basic` avec `option=false` et `@Column` pour spécifier le nom de la colonne et qu'elle est obligatoire

Création de l'entité Personne

Etant donné que la classe Client doit hériter d'elle, nous utilisons l'annotation `@Inheritance` avec la stratégie JOINED qui permet de faire de chaque entité une table séparée. On ajoute les attributs prenom, nom, telephone et email en ajoutant `@Basic` et `@Column` ainsi que les getters et setters pour le nom de la colonne ainsi que si elle est obligatoire ou non.

On ajoute enfin les constructeurs avec et sans arguments, les méthodes `toString()`, `equals()` et `hashCode()`.

```

1 package sn.ept.ventesvelos.entites;
2
3 import ...
4
5
6 @Entity
7 @Table(name="personne")
8 @Inheritance(strategy = InheritanceType.JOINED)
9 @SequenceGenerator(name = "personne_seq", sequenceName = "personne_seq", allocationSize = 1, initialValue = 1456)
10 public class Personne implements Serializable {
11
12     @Id
13     @GeneratedValue(strategy = GenerationType.SEQUENCE, generator = "personne_seq")
14     @Basic(optional = false)
15     @Column(name = "ID")
16     private Integer id;
17
18
19     @Basic(optional = false)
20     @Column(name = "PRENOM")
21     private String prenom;
22
23     @Basic(optional = false)
24     @Column(name = "NOM")
25     private String nom;

```

Création de l'entité Client

Vu qu'elle hérite de la classe Personne, on étend la classe Personne. Ensuite étant donné que la classe Employe hérite de Client, nous utilisons l'annotation `@Inheritance` avec la stratégie JOINED. On ajoute ensuite l'attribut adresse de la classe précédemment créée en utilisant l'annotation `@Embedded` pour l'embarquer dans la classe ainsi que `@AttributeOverrides` afin de redéfinir les propriétés des colonnes.

La classe Client possède une relation d'aggrégation avec Commande donc on la traduit en créant un champ commandeCollection avec l'annotation `@OneToMany` avec `CascadeType.REMOVE` et `orphanRemoval=true` pour stocker toutes les commandes du client et cascader la suppression.

On ajoute enfin les getters, setters, les constructeurs avec et sans arguments, les méthodes `toString()`, `equals()` et `hashCode()`.

```

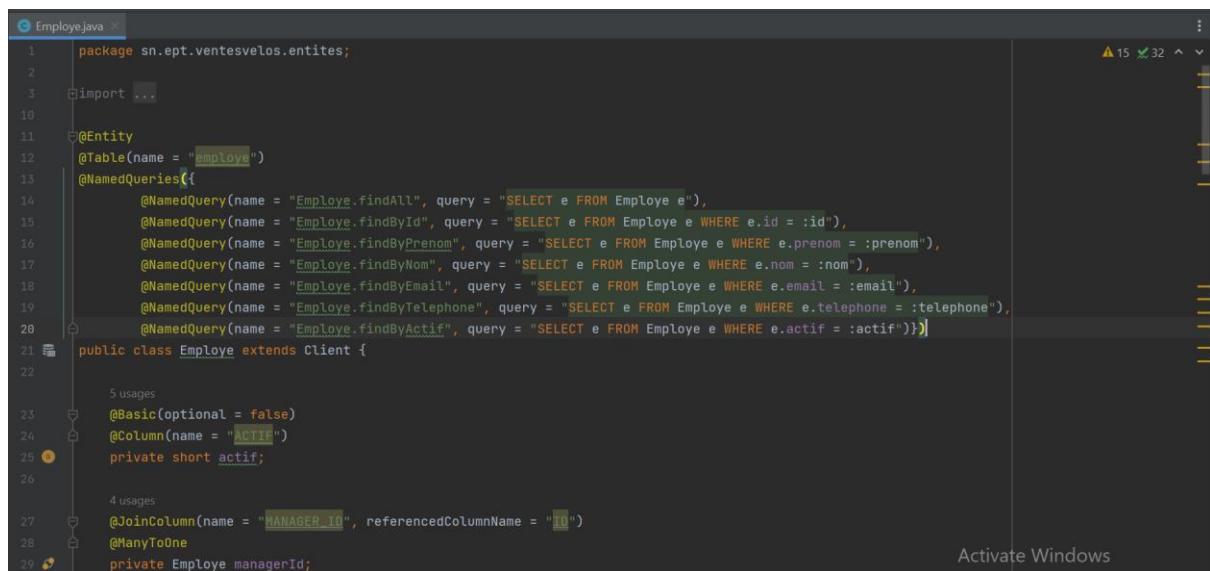
1 package sn.ept.ventesvelos.entites;
2
3 import ...
4
5
6 @Entity
7 @Inheritance(strategy = InheritanceType.JOINED)
8 @NamedQueries({
9     @NamedQuery(name = "Client.findAll", query = "SELECT c FROM Client c"),
10    @NamedQuery(name = "Client.findById", query = "SELECT c FROM Client c WHERE c.id = :id"),
11    @NamedQuery(name = "Client.findByPrenom", query = "SELECT c FROM Client c WHERE c.prenom = :prenom"),
12    @NamedQuery(name = "Client.findByName", query = "SELECT c FROM Client c WHERE c.nom = :nom"),
13    @NamedQuery(name = "Client.findByTelephone", query = "SELECT c FROM Client c WHERE c.telephone = :telephone"),
14    @NamedQuery(name = "Client.findByEmail", query = "SELECT c FROM Client c WHERE c.email = :email"),
15    @NamedQuery(name = "Client.findByAdresse", query = "SELECT c FROM Client c WHERE c.adresse = :adresse")})
16
17 @Table(name = "client")
18 public class Client extends Personne {
19
20
21     @Embedded
22     @AttributeOverrides({
23         @AttributeOverride(name="adresse", column = @Column(name="ADRESSE")),
24         @AttributeOverride(name="ville", column = @Column(name="VILLE")),
25         @AttributeOverride(name="etat", column = @Column(name="ETAT")),
26         @AttributeOverride(name="codeZip", column = @Column(name="CODE_ZIP"))})

```

Création de l'entité Employe

On traduit l'héritage en étendant la classe Client ainsi que l'attribut actif. Pour traduire la relation réflexive Manager et l'association avec Magasin on ajoute les attributs managerId et magasinId avec l'annotation @ManyToOne et @JoinColumn. On traduit l'aggrégation avec Commande et l'opposé de la relation réflexive par les attributs commandeCollection et employeCollection avec l'annotation @OneToMany avec avec *CascadeType.REMOVE* et *orphanRemoval=true* pour stocker toutes les commandes de l'employé, tous les employés dont il est manager et cascader la suppression.

On ajoute enfin les getters, setters, les constructeurs avec et sans arguments, les méthodes `toString()`, `equals()` et `hashCode()`.



```

1 package sn.ept.ventesvelos.entites;
2
3 import ...
4
5 @Entity
6 @Table(name = "employe")
7 @NamedQueries({
8     @NamedQuery(name = "Employe.findAll", query = "SELECT e FROM Employe e"),
9     @NamedQuery(name = "Employe.findById", query = "SELECT e FROM Employe e WHERE e.id = :id"),
10    @NamedQuery(name = "Employe.findByPrenom", query = "SELECT e FROM Employe e WHERE e.prenom = :prenom"),
11    @NamedQuery(name = "Employe.findByNom", query = "SELECT e FROM Employe e WHERE e.nom = :nom"),
12    @NamedQuery(name = "Employe.findByEmail", query = "SELECT e FROM Employe e WHERE e.email = :email"),
13    @NamedQuery(name = "Employe.findByTelephone", query = "SELECT e FROM Employe e WHERE e.telephone = :telephone"),
14    @NamedQuery(name = "Employe.findByActif", query = "SELECT e FROM Employe e WHERE e.actif = :actif"))})
15
16 public class Employe extends Client {
17
18     5 usages
19     @Basic(optional = false)
20     @Column(name = "ACTIF")
21     private short actif;
22
23     4 usages
24     @JoinColumn(name = "MANAGER_10", referencedColumnName = "ID")
25     @ManyToOne
26     private Client managerId;
27
28 }

```

Création de l'entité Magasin

On ajoute les attributs nom, telephone, email, l'attribut adresse de la classe précédemment créée en utilisant l'annotation @Embedded pour l'embarquer dans la classe ainsi que @AttributeOverrides afin de redéfinir les propriétés des colonnes.

Pour traduire l'aggrégation avec Commande, les associations avec Employe et Stock, on ajoute les attributs employeCollection, commandeCollection et stockCollection avec l'annotation @OneToMany avec *CascadeType.REMOVE* et *orphanRemoval=true* pour stocker tous les employés, les commandes effectuées au magasin et les stocks disponibles ainsi que cascader la suppression.

On ajoute enfin les getters, setters, les constructeurs avec et sans arguments, les méthodes `toString()`, `equals()` et `hashCode()`.

```

1 package sn.ept.ventesvelos.entites;
2
3 import ...
4
5
6 @Entity
7 @Table(name = "magasin")
8 @NamedQueries({
9     @NamedQuery(name = "Magasin.findAll", query = "SELECT m FROM Magasin m"),
10    @NamedQuery(name = "Magasin.findById", query = "SELECT m FROM Magasin m WHERE m.id = :id"),
11    @NamedQuery(name = "Magasin.findByName", query = "SELECT m FROM Magasin m WHERE m.nom = :nom"),
12    @NamedQuery(name = "Magasin.findByTelephone", query = "SELECT m FROM Magasin m WHERE m.telephone = :telephone"),
13    @NamedQuery(name = "Magasin.findByEmail", query = "SELECT m FROM Magasin m WHERE m.email = :email"),
14    @NamedQuery(name = "Magasin.findByAdresse", query = "SELECT m FROM Magasin m WHERE m.adresse = :adresse")})
15 @SequenceGenerator(name = "magasin_seq", sequenceName = "magasin_seq", allocationSize = 1, initialValue = 4)
16
17 public class Magasin implements Serializable {
18
19     13 usages
20
21     @Id
22     @GeneratedValue(strategy = GenerationType.SEQUENCE, generator = "magasin_seq")
23     @Basic(optional = false)
24     @Column(name = "ID")
25     private Integer id;
26
27     5 usages
28     @Basic(optional = false)
29     @Column(name = "NOM")

```

Création de l'entité Commande

On ajoute les attributs statut, dateCommande, dateLivraisonVoulue, dateLivraison.

Pour traduire l'aggrégation avec Magasin, Employe et Client, nous créons les attributs magasinId, vendeurId et clientId avec l'annotation @ManyToOne et @JoinColumn.

Pour traduire la composition avec ArticleCommande, on crée l'attribut articleCommandeCollection avec l'annotation @OneToMany avec *CascadeType.REMOVE* et *orphanRemoval=true* pour stocker toutes les articles commandés de la commande et cascader la suppression.

On ajoute enfin les getters, setters, les constructeurs avec et sans arguments, les méthodes `toString()`, `equals()` et `hashCode()`.

```

1 package sn.ept.ventesvelos.entites;
2
3 import ...
4
5
6 @Entity
7 @Table(name = "commande")
8 @NamedQueries({
9     @NamedQuery(name = "Commande.findAll", query = "SELECT c FROM Commande c"),
10    @NamedQuery(name = "Commande.findByNumero", query = "SELECT c FROM Commande c WHERE c.numero = :numero"),
11    @NamedQuery(name = "Commande.findByStatut", query = "SELECT c FROM Commande c WHERE c.statut = :statut"),
12    @NamedQuery(name = "Commande.findByDateCommande", query = "SELECT c FROM Commande c WHERE c.dateCommande = :dateCommande"),
13    @NamedQuery(name = "Commande.findByDateLivraisonVoulue", query = "SELECT c FROM Commande c WHERE c.dateLivraisonVoulue = :dateLivraisonVoulue"),
14    @NamedQuery(name = "Commande.findByDateLivraison", query = "SELECT c FROM Commande c WHERE c.dateLivraison = :dateLivraison")})
15 @SequenceGenerator(name = "commande_seq", sequenceName = "commande_seq", allocationSize = 1, initialValue = 1616)
16
17 public class Commande implements Serializable {
18
19     13 usages
20
21     @Id
22     @GeneratedValue(strategy = GenerationType.SEQUENCE, generator = "commande_seq")
23     @Basic(optional = false)
24     @Column(name = "NUMERO")
25     private Integer numero;
26
27     5 usages
28     @Basic(optional = false)
29     @Column(name = "STATUT")
30     private short statut;

```

Créer l'entité ArticleCommande

On crée d'abord une classe pour la clé primaire d'ArticleCommande, **ArticleCommandePK**. On utilise l'annotation @Embeddable pour spécifier qu'il s'agit d'une classe qui sera embarqué dans une entité. On ajoute les attributs numeroCommande et ligne en spécifiant qu'ils sont obligatoire puis on crée les getters, setters, les constructeurs avec et sans arguments ainsi que les méthodes `toString()`, `equals()` et `hashCode()`.

```
ArticleCommandePK.java x
1 package sn.ept.ventesvelos.entites;
2
3 import ...
4
5 12 usages
6
7 @Embeddable
8 public class ArticleCommandePK implements Serializable {
9
10     4 usages
11     @Basic(optional = false)
12     @Column(name="NUMERO_COMMANDE")
13     private int numeroCommande;
14     4 usages
15     @Basic(optional = false)
16     @Column(name = "LIGNE")
17     private int ligne;
18
19     1 usage
20     public ArticleCommandePK() {
21     }
22
23     2 usages
24     public ArticleCommandePK(int numeroCommande, int ligne) {
25         this.numeroCommande = numeroCommande;
26         this.ligne = ligne;
27     }
28
29 }
```

On crée la classe ArticleCommande en utilisant l'annotation @EmbeddedId pour embarquer la clé primaire dans la classe. On ajoute les attributs quantite, prixDepart et remise. Pour traduire la composition de Produit et Commande on ajoute les attributs produitId et commandeId avec les annotations @ManyToOne et @JoinColumn.

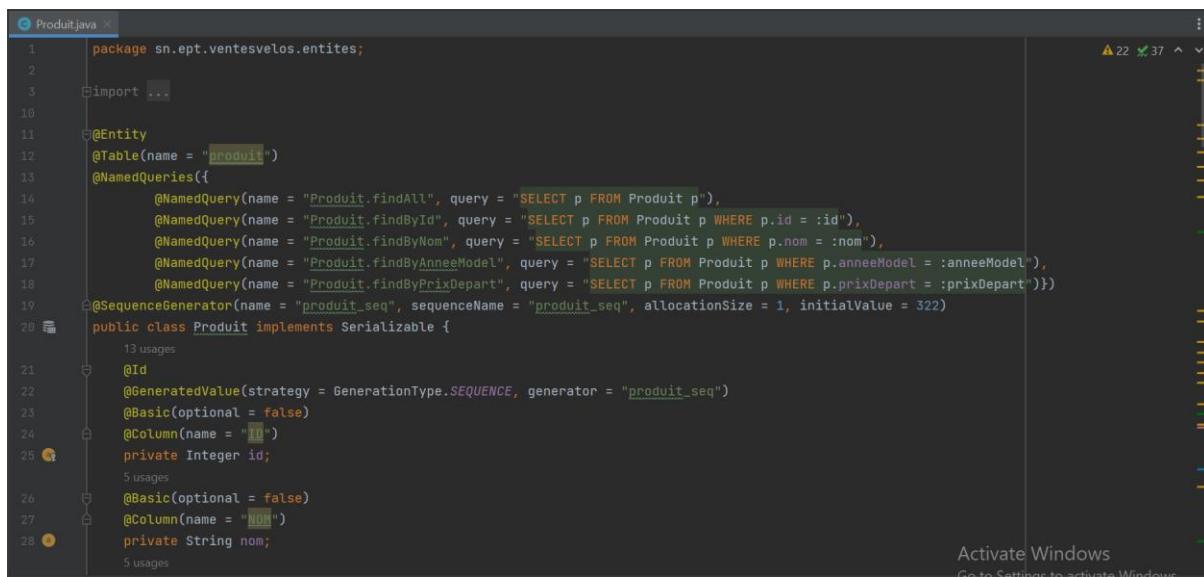
On ajoute enfin les getters, setters, les constructeurs avec et sans arguments, les méthodes `toString()`, `equals()` et `hashCode()`.

```
1 package sn.ept.ventesvelos.entites;
2
3 import ...
4
5
6 @Entity
7 @Table(name = "article_commande")
8 @NamedQueries({
9     @NamedQuery(name = "ArticleCommande.findAll", query = "SELECT a FROM ArticleCommande a"),
10    @NamedQuery(name = "ArticleCommande.findByNumeroCommande", query = "SELECT a FROM ArticleCommande a WHERE a.articleCommandePK.numeroCommande = :numeroCommande"),
11    @NamedQuery(name = "ArticleCommande.findByLigne", query = "SELECT a FROM ArticleCommande a WHERE a.articleCommandePK.ligne = :ligne"),
12    @NamedQuery(name = "ArticleCommande.findByProduitId", query = "SELECT a FROM ArticleCommande a WHERE a.produitId = :produitId"),
13    @NamedQuery(name = "ArticleCommande.findByQuantite", query = "SELECT a FROM ArticleCommande a WHERE a.quantite = :quantite"),
14    @NamedQuery(name = "ArticleCommande.findByPrixDepart", query = "SELECT a FROM ArticleCommande a WHERE a.prixDepart = :prixDepart"),
15    @NamedQuery(name = "ArticleCommande.findByRemise", query = "SELECT a FROM ArticleCommande a WHERE a.remise = :remise"))
16
17
18 public class ArticleCommande implements Serializable {
19
20     @EmbeddedId
21     private ArticleCommandePK articleCommandePK;
22
23     @Basic(optional = false)
24     @Column(name = "QUANTITE")
25     private int quantite;
26
27     @Basic(optional = false)
28     @Column(name = "PRIX_DEPART")
```

Créer l'entité Produit

On crée les attributs nom, anneeModel et prixDepart. Pour traduire l'aggrégation Marque et l'association Categorie on crée les attributs marqueId et categorieId avec les annotations @ManyToOne et @JoinColumn. On traduit ensuite la composition ArticleCommande et l'association Stock avec les attributs articleCommandeCollection et stockCollection en utilisant l'annotation @OneToMany avec *CascadeType.REMOVE* et *orphanRemoval=true* pour stocker toutes les articles commandes où il figure ainsi que les stocks et cascader la suppression.

On ajoute enfin les getters, setters, les constructeurs avec et sans arguments, les méthodes `toString()`, `equals()` et `hashCode()`.



```

1 package sn.ept.ventesvelos.entites;
2
3 import ...
4
5 @Entity
6 @Table(name = "produit")
7 @NamedQueries({
8     @NamedQuery(name = "Produit.findAll", query = "SELECT p FROM Produit p"),
9     @NamedQuery(name = "Produit.findById", query = "SELECT p FROM Produit p WHERE p.id = :id"),
10    @NamedQuery(name = "Produit.findByNom", query = "SELECT p FROM Produit p WHERE p.nom = :nom"),
11    @NamedQuery(name = "Produit.findByAnneeModel", query = "SELECT p FROM Produit p WHERE p.anneeModel = :anneeModel"),
12    @NamedQuery(name = "Produit.findByPrixDepart", query = "SELECT p FROM Produit p WHERE p.prixDepart = :prixDepart"))
13    @SequenceGenerator(name = "produit_seq", sequenceName = "produit_seq", allocationSize = 1, initialValue = 322)
14
15 public class Produit implements Serializable {
16     @Id
17     @GeneratedValue(strategy = GenerationType.SEQUENCE, generator = "produit_seq")
18     @Basic(optional = false)
19     @Column(name = "ID")
20     private Integer id;
21     @Basic(optional = false)
22     @Column(name = "NOM")
23     private String nom;
24 }
25
26
27
28

```

Créer l'entité Categorie

On ajoute l'attribut nom et on traduit l'association Produit avec l'attribut produitCollection avec l'annotation @OneToMany avec *CascadeType.REMOVE* et *orphanRemoval=true* pour stocker tous les produits de cette catégorie et cascader la suppression.

On ajoute enfin les getters, setters, les constructeurs avec et sans arguments, les méthodes `toString()`, `equals()` et `hashCode()`.

```

1 package sn.ept.ventesvelos.entites;
2
3 import ...
4
5 @Entity
6     @Table(name = "categorie")
7     @NamedQueries({
8         @NamedQuery(name = "Categorie.findAll", query = "SELECT c FROM Categorie c"),
9         @NamedQuery(name = "Categorie.findById", query = "SELECT c FROM Categorie c WHERE c.id = :id"),
10        @NamedQuery(name = "Categorie.findByNom", query = "SELECT c FROM Categorie c WHERE c.nom = :nom")})
11    @SequenceGenerator(name = "categorie_seq", sequenceName = "categorie_seq", allocationSize = 1, initialValue = 8)
12    public class Categorie implements Serializable {
13
14        13 usages
15        @Id
16            @GeneratedValue(strategy = GenerationType.SEQUENCE, generator = "categorie_seq")
17            @Basic(optional = false)
18            @Column(name = "ID")
19            private Integer id;
20
21        5 usages
22        @Basic(optional = false)
23        @Column(name = "NOM")
24        private String nom;
25
26        3 usages
27        @OneToMany(cascade = CascadeType.ALL, mappedBy = "categorieId", orphanRemoval = true)

```

Activate Windows
Go to Settings to activate Windows

Créer l'entité Marque

On ajoute l'attribut nom et on traduit l'aggrégation Produit avec l'attribut produitCollection avec l'annotation @OneToMany avec *CascadeType.REMOVE* et *orphanRemoval=true* pour stocker toutes les et cascader la suppression.

On ajoute enfin les getters, setters, les constructeurs avec et sans arguments, les méthodes `toString()`, `equals()` et `hashCode()`.

```

1 package sn.ept.ventesvelos.entites;
2
3 import ...
4
5 @Entity
6     @Table(name = "marque")
7     @NamedQueries({
8         @NamedQuery(name = "Marque.findAll", query = "SELECT m FROM Marque m"),
9         @NamedQuery(name = "Marque.findById", query = "SELECT m FROM Marque m WHERE m.id = :id"),
10        @NamedQuery(name = "Marque.findByNom", query = "SELECT m FROM Marque m WHERE m.nom = :nom")})
11    @SequenceGenerator(name = "marque_seq", sequenceName = "marque_seq", allocationSize = 1, initialValue = 10)
12    public class Marque implements Serializable {
13
14        13 usages
15        @Id
16            @GeneratedValue(strategy = GenerationType.SEQUENCE, generator = "marque_seq")
17            @Basic(optional = false)
18            @Column(name = "ID")
19            private Integer id;
20
21        5 usages
22        @Basic(optional = false)
23        @Column(name = "NOM")
24        private String nom;
25
26        3 usages
27        @OneToMany(cascade = CascadeType.ALL, mappedBy = "marqueId", orphanRemoval = true)

```

Activate Windows
Go to Settings to activate Windows

Créer l'entité Stock

On crée d'abord une classe pour la clé primaire d'ArticleCommande, **StockPK**. On utilise l'annotation @Embeddable pour spécifier qu'il s'agit d'une classe qui sera embarqué dans une entité. On ajoute les attributs magasinId et produitId en spécifiant qu'ils sont obligatoire puis on crée les getters, setters, les constructeurs avec et sans arguments ainsi que les méthodes `toString()`, `equals()` et `hashCode()`.

```

1 package sn.ept.ventesvelos.entites;
2
3 import ...
4
5 13 usages
6
7 @Embeddable
8 public class StockPK implements Serializable {
9     7 usages
10    @Basic(optional = false)
11    @Column(name = "MAGASIN_ID")
12    private int magasinId;
13    7 usages
14    @Basic(optional = false)
15    @Column(name = "PRODUIT_ID")
16    private int produitId;
17
18    no usages
19    public StockPK() {
20
21    4 usages
22    public StockPK(int magasinId, int produitId) {
23        this.magasinId = magasinId;
24        this.produitId = produitId;
25

```

On crée la classe Stock en utilisant l'annotation @EmbeddedId pour embarquer la clé primaire dans la classe. On ajoute l'attribut quantite. Pour traduire l'association de Produit et Magasin on ajoute les attributs produitId et magasinId avec les annotations @ManyToOne et @JoinColumn.

On ajoute enfin les getters, setters, les constructeurs avec et sans arguments, les méthodes `toString()`, `equals()` et `hashCode()`.

```

1 package sn.ept.ventesvelos.entites;
2
3 import ...
4
5 @Entity
6 @Table(name = "stock")
7 @NamedQueries({
8     @NamedQuery(name = "Stock.findAll", query = "SELECT s FROM Stock s"),
9     @NamedQuery(name = "Stock.findByMagasinId", query = "SELECT s FROM Stock s WHERE s.stockPK.magasinId = :magasinId"),
10    @NamedQuery(name = "Stock.findByProduitId", query = "SELECT s FROM Stock s WHERE s.stockPK.produitId = :produitId"),
11    @NamedQuery(name = "Stock.findByQuantite", query = "SELECT s FROM Stock s WHERE s.quantite = :quantite")})
12
13 public class Stock implements Serializable {
14     15 usages
15     @EmbeddedId
16     protected StockPK stockPK;
17     5 usages
18     @Basic(optional = false)
19     @Column(name = "QUANTITE")
20     private int quantite;
21
22     3 usages
23     @JoinColumn(name = "PRODUIT_ID", referencedColumnName = "ID", insertable = false, updatable = false)
24     @ManyToOne(optional = false)
25     private Produit produitId;
26

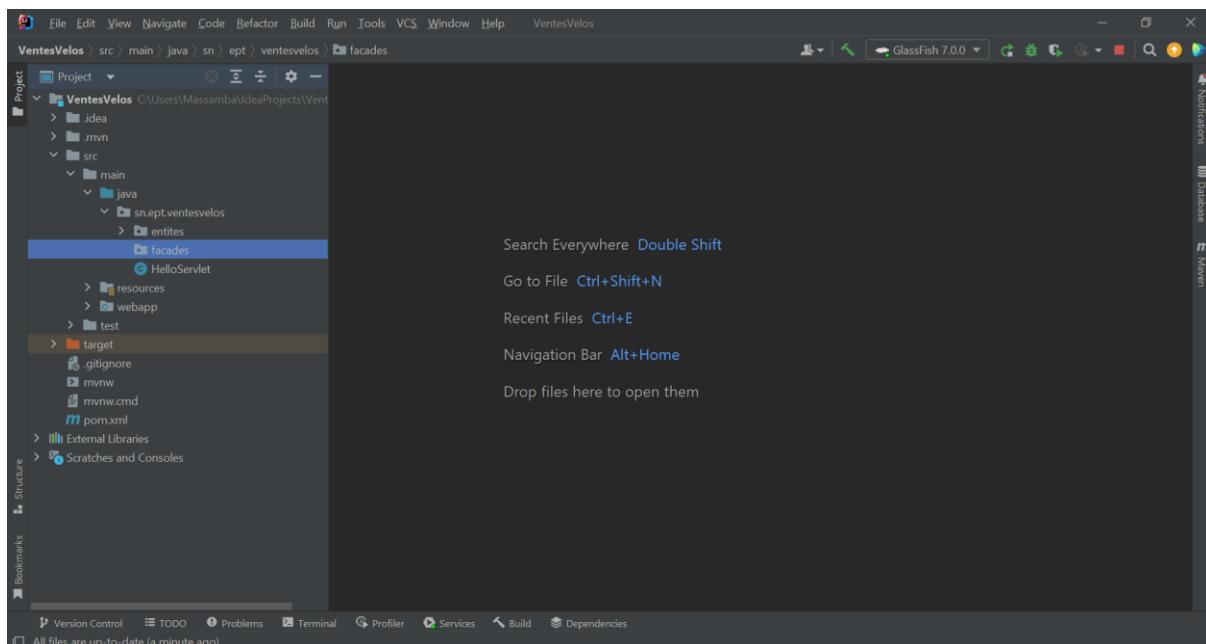
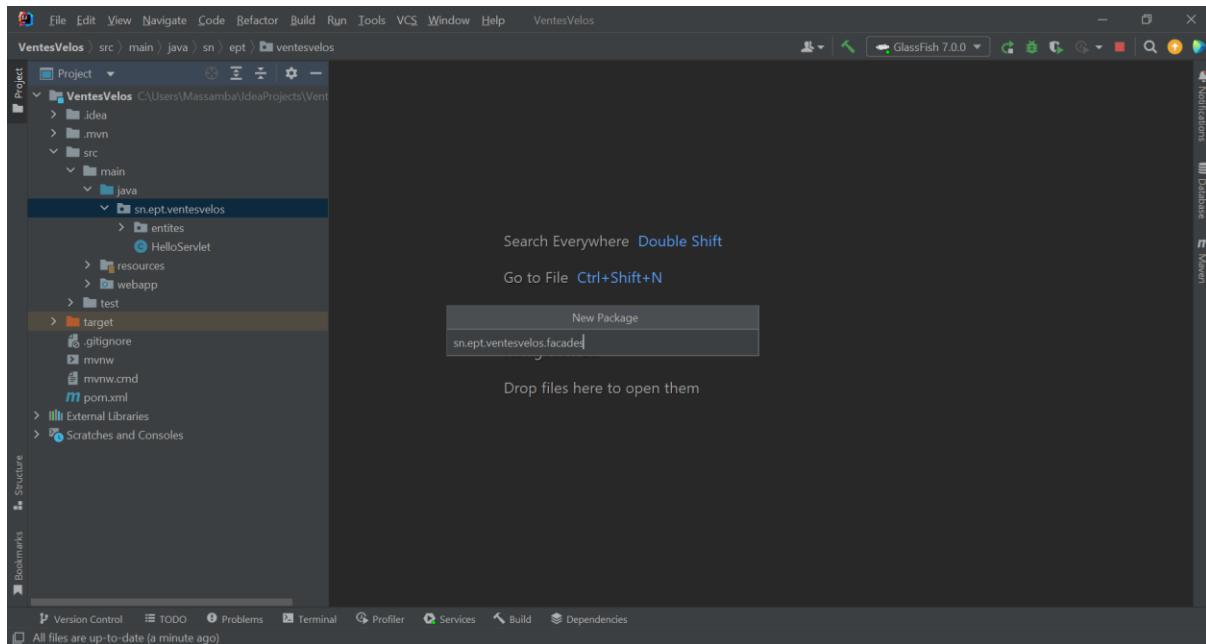
```

III. Développement des EJB et des façades

1. Création des Façades

Création d'un package facades

On crée un package facades où nous allons mettre nos différentes façades



Création de la classe AbstractFacade

La classe AbstractFacade qui définit un ensemble de méthodes pour créer, supprimer, mettre à jour les entités. C'est d'elle que toutes nos façades hériteront.

The screenshot shows the AbstractFacade.java file in an IDE. The code defines an abstract facade class that extends AbstractFacade<T>. It includes a private EntityManager variable and methods for creating, editing, and removing entities. The IDE interface includes tabs for Version Control, TODO, Problems, Terminal, Profiler, Services, Build, and Dependencies. A status bar at the bottom indicates the file is up-to-date and shows the time as 7:23.

```
1 package sn.ept.ventesvelos.facades;
2
3 import ...
4
5 10 inheritors
6
7 public abstract class AbstractFacade<T> {
8
9     4 usages
10    private Class<T> entityClass;
11
12    public AbstractFacade(Class<T> entityClass) { this.entityClass = entityClass; }
13
14    10 implementations
15    protected abstract EntityManager getEntityManager();
16
17    no usages
18    public void create(T entity) { getEntityManager().persist(entity); }
19
20    no usages
21    public void edit(T entity) { getEntityManager().merge(entity); }
22
23    no usages
24    public void remove(T entity) { getEntityManager().remove(getEntityManager().merge(entity)); }
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
209
210
211
212
213
214
215
216
217
217
218
219
219
220
221
222
223
224
225
226
227
227
228
229
229
230
231
232
233
233
234
235
235
236
236
237
237
238
238
239
239
240
240
241
241
242
242
243
243
244
244
245
245
246
246
247
247
248
248
249
249
250
250
251
251
252
252
253
253
254
254
255
255
256
256
257
257
258
258
259
259
260
260
261
261
262
262
263
263
264
264
265
265
266
266
267
267
268
268
269
269
270
270
271
271
272
272
273
273
274
274
275
275
276
276
277
277
278
278
279
279
280
280
281
281
282
282
283
283
284
284
285
285
286
286
287
287
288
288
289
289
290
290
291
291
292
292
293
293
294
294
295
295
296
296
297
297
298
298
299
299
300
300
301
301
302
302
303
303
304
304
305
305
306
306
307
307
308
308
309
309
310
310
311
311
312
312
313
313
314
314
315
315
316
316
317
317
318
318
319
319
320
320
321
321
322
322
323
323
324
324
325
325
326
326
327
327
328
328
329
329
330
330
331
331
332
332
333
333
334
334
335
335
336
336
337
337
338
338
339
339
340
340
341
341
342
342
343
343
344
344
345
345
346
346
347
347
348
348
349
349
350
350
351
351
352
352
353
353
354
354
355
355
356
356
357
357
358
358
359
359
360
360
361
361
362
362
363
363
364
364
365
365
366
366
367
367
368
368
369
369
370
370
371
371
372
372
373
373
374
374
375
375
376
376
377
377
378
378
379
379
380
380
381
381
382
382
383
383
384
384
385
385
386
386
387
387
388
388
389
389
390
390
391
391
392
392
393
393
394
394
395
395
396
396
397
397
398
398
399
399
400
400
401
401
402
402
403
403
404
404
405
405
406
406
407
407
408
408
409
409
410
410
411
411
412
412
413
413
414
414
415
415
416
416
417
417
418
418
419
419
420
420
421
421
422
422
423
423
424
424
425
425
426
426
427
427
428
428
429
429
430
430
431
431
432
432
433
433
434
434
435
435
436
436
437
437
438
438
439
439
440
440
441
441
442
442
443
443
444
444
445
445
446
446
447
447
448
448
449
449
450
450
451
451
452
452
453
453
454
454
455
455
456
456
457
457
458
458
459
459
460
460
461
461
462
462
463
463
464
464
465
465
466
466
467
467
468
468
469
469
470
470
471
471
472
472
473
473
474
474
475
475
476
476
477
477
478
478
479
479
480
480
481
481
482
482
483
483
484
484
485
485
486
486
487
487
488
488
489
489
490
490
491
491
492
492
493
493
494
494
495
495
496
496
497
497
498
498
499
499
500
500
501
501
502
502
503
503
504
504
505
505
506
506
507
507
508
508
509
509
510
510
511
511
512
512
513
513
514
514
515
515
516
516
517
517
518
518
519
519
520
520
521
521
522
522
523
523
524
524
525
525
526
526
527
527
528
528
529
529
530
530
531
531
532
532
533
533
534
534
535
535
536
536
537
537
538
538
539
539
540
540
541
541
542
542
543
543
544
544
545
545
546
546
547
547
548
548
549
549
550
550
551
551
552
552
553
553
554
554
555
555
556
556
557
557
558
558
559
559
560
560
561
561
562
562
563
563
564
564
565
565
566
566
567
567
568
568
569
569
570
570
571
571
572
572
573
573
574
574
575
575
576
576
577
577
578
578
579
579
580
580
581
581
582
582
583
583
584
584
585
585
586
586
587
587
588
588
589
589
590
590
591
591
592
592
593
593
594
594
595
595
596
596
597
597
598
598
599
599
600
600
601
601
602
602
603
603
604
604
605
605
606
606
607
607
608
608
609
609
610
610
611
611
612
612
613
613
614
614
615
615
616
616
617
617
618
618
619
619
620
620
621
621
622
622
623
623
624
624
625
625
626
626
627
627
628
628
629
629
630
630
631
631
632
632
633
633
634
634
635
635
636
636
637
637
638
638
639
639
640
640
641
641
642
642
643
643
644
644
645
645
646
646
647
647
648
648
649
649
650
650
651
651
652
652
653
653
654
654
655
655
656
656
657
657
658
658
659
659
660
660
661
661
662
662
663
663
664
664
665
665
666
666
667
667
668
668
669
669
670
670
671
671
672
672
673
673
674
674
675
675
676
676
677
677
678
678
679
679
680
680
681
681
682
682
683
683
684
684
685
685
686
686
687
687
688
688
689
689
690
690
691
691
692
692
693
693
694
694
695
695
696
696
697
697
698
698
699
699
700
700
701
701
702
702
703
703
704
704
705
705
706
706
707
707
708
708
709
709
710
710
711
711
712
712
713
713
714
714
715
715
716
716
717
717
718
718
719
719
720
720
721
721
722
722
723
723
724
724
725
725
726
726
727
727
728
728
729
729
730
730
731
731
732
732
733
733
734
734
735
735
736
736
737
737
738
738
739
739
740
740
741
741
742
742
743
743
744
744
745
745
746
746
747
747
748
748
749
749
750
750
751
751
752
752
753
753
754
754
755
755
756
756
757
757
758
758
759
759
760
760
761
761
762
762
763
763
764
764
765
765
766
766
767
767
768
768
769
769
770
770
771
771
772
772
773
773
774
774
775
775
776
776
777
777
778
778
779
779
780
780
781
781
782
782
783
783
784
784
785
785
786
786
787
787
788
788
789
789
790
790
791
791
792
792
793
793
794
794
795
795
796
796
797
797
798
798
799
799
800
800
801
801
802
802
803
803
804
804
805
805
806
806
807
807
808
808
809
809
810
810
811
811
812
812
813
813
814
814
815
815
816
816
817
817
818
818
819
819
820
820
821
821
822
822
823
823
824
824
825
825
826
826
827
827
828
828
829
829
830
830
831
831
832
832
833
833
834
834
835
835
836
836
837
837
838
838
839
839
840
840
841
841
842
842
843
843
844
844
845
845
846
846
847
847
848
848
849
849
850
850
851
851
852
852
853
853
854
854
855
855
856
856
857
857
858
858
859
859
860
860
861
861
862
862
863
863
864
864
865
865
866
866
867
867
868
868
869
869
870
870
871
871
872
872
873
873
874
874
875
875
876
876
877
877
878
878
879
879
880
880
881
881
882
882
883
883
884
884
885
885
886
886
887
887
888
888
889
889
890
890
891
891
892
892
893
893
894
894
895
895
896
896
897
897
898
898
899
899
900
900
901
901
902
902
903
903
904
904
905
905
906
906
907
907
908
908
909
909
910
910
911
911
912
912
913
913
914
914
915
915
916
916
917
917
918
918
919
919
920
920
921
921
922
922
923
923
924
924
925
925
926
926
927
927
928
928
929
929
930
930
931
931
932
932
933
933
934
934
935
935
936
936
937
937
938
938
939
939
940
940
941
941
942
942
943
943
944
944
945
945
946
946
947
947
948
948
949
949
950
950
951
951
952
952
953
953
954
954
955
955
956
956
957
957
958
958
959
959
960
960
961
961
962
962
963
963
964
964
965
965
966
966
967
967
968
968
969
969
970
970
971
971
972
972
973
973
974
974
975
975
976
976
977
977
978
978
979
979
980
980
981
981
982
982
983
983
984
984
985
985
986
986
987
987
988
988
989
989
990
990
991
991
992
992
993
993
994
994
995
995
996
996
997
997
998
998
999
999
1000
1000
1001
1001
1002
1002
1003
1003
1004
1004
1005
1005
1006
1006
1007
1007
1008
1008
1009
1009
1010
1010
1011
1011
1012
1012
1013
1013
1014
1014
1015
1015
1016
1016
1017
1017
1018
1018
1019
1019
1020
1020
1021
1021
1022
1022
1023
1023
1024
1024
1025
1025
1026
1026
1027
1027
1028
1028
1029
1029
1030
1030
1031
1031
1032
1032
1033
1033
1034
1034
1035
1035
1036
1036
1037
1037
1038
1038
1039
1039
1040
1040
1041
1041
1042
1042
1043
1043
1044
1044
1045
1045
1046
1046
1047
1047
1048
1048
1049
1049
1050
1050
1051
1051
1052
1052
1053
1053
1054
1054
1055
1055
1056
1056
1057
1057
1058
1058
1059
1059
1060
1060
1061
1061
1062
1062
1063
1063
1064
1064
1065
1065
1066
1066
1067
1067
1068
1068
1069
1069
1070
1070
1071
1071
1072
1072
1073
1073
1074
1074
1075
1075
1076
1076
1077
1077
1078
1078
1079
1079
1080
1080
1081
1081
1082
1082
1083
1083
1084
1084
1085
1085
1086
1086
1087
1087
1088
1088
1089
1089
1090
1090
1091
1091
1092
1092
1093
1093
1094
1094
1095
1095
1096
1096
1097
1097
1098
1098
1099
1099
1100
1100
1101
1101
1102
1102
1103
1103
1104
1104
1105
1105
1106
1106
1107
1107
1108
1108
1109
1109
1110
1110
1111
1111
1112
1112
1113
1113
1114
1114
1115
1115
1116
1116
1117
1117
1118
1118
1119
1119
1120
1120
1121
1121
1122
1122
1123
1123
1124
1124
1125
1125
1126
1126
1127
1127
1128
1128
1129
1129
1130
1130
1131
1131
1132
1132
1133
1133
1134
1134
1135
1135
1136
1136
1137
1137
1138
1138
1139
1139
1140
1140
1141
1141
1142
1142
1143
1143
1144
1144
1145
1145
1146
1146
1147
1147
1148
1148
1149
1149
1150
1150
1151
1151
1152
1152
1153
1153
1154
1154
1155
1155
1156
1156
1157
1157
1158
1158
1159
1159
1160
1160
1161
1161
1162
1162
1163
1163
1164
1164
1165
1165
1166
1166
1167
1167
1168
1168
1169
1169
1170
1170
1171
1171
1172
1172
1173
1173
1174
1174
1175
1175
1176
1176
1177
1177
1178
1178
1179
1179
1180
1180
1181
1181
1182
1182
1183
1183
1184
1184
1185
1185
1186
1186
1187
1187
1188
1188
1189
1189
1190
1190
1191
1191
1192
1192
1193
1193
1194
1194
1195
1195
1196
1196
1197
1197
1198
1198
1199
1199
1200
1200
1201
1201
1202
1202
1203
1203
1204
1204
1205
1205
1206
1206
1207
1207
1208
1208
1209
1209
1210
1210
1211
1211
1212
1212
1213
1213
1214
1214
1215
1215
1216
1216
1217
1217
1218
1218
1219
1219
1220
1220
1221
1221
1222
1222
1223
1223
1224
1224
1225
1225
1226
1226
1227
1227
1228
1228
1229
1229
1230
1230
1231
1231
1232
1232
1233
1233
1234
1234
1235
1235
1236
1236
1237
1237
1238
1238
1239
1239
1240
1240
1241
1241
1242
1242
1243
1243
1244
1244
1245
1245
1246
1246
1247
1247
1248
1248
1249
1249
1250
1250
1251
1251
1252
1252
1253
1253
1254
1254
1255
1255
1256
1256
1257
1257
1258
1258
1259
1259
1260
1260
1261
1261
1262
1262
1263
1263
1264
1264
1265
1265
1266
1266
1267
1267
1268
1268
1269
1269
1270
1270
1271
1271
1272
1272
1273
1273
1274
1274
1275
1275
1276
1276
1277
1277
1278
1278
1279
1279
1280
1280
1281
1281
1282
1282
1283
1283
1284
1284
1285
1285
1286
1286
1287
1287
1288
1288
1289
1289
1290
1290
1291
1291
1292
1292
1293
1293
1294
1294
1295
1295
1296
1296
1297
1297
1298
1298
1299
1299
1300
1300
1301
1301
1302
1302
1303
1303
1304
1304
1305
1305
1306
1306
1307
1307
1308
1308
1309
1309
1310
1310
1311
1311
1312
1312
1313
1313
1314
1314
1315
1315
1316
1316
1317
1317
1318
1318
1319
1319
1320
1320
1321
1321
1322
1322
1323
1323
1324
1324
1325
1325
1326
1326
1327
1327
1328
1328
1329
1329
1330
1330
1331
1331
1332
1332
1333
1333
1334
1334
1335
1335
1336
1336
1337
1337
1338
1338
1339
1339
1340
1340
1341
1341
1342
1342
1343
1343
1344
1344
1345
1345
1346
1346
1347
1347
1348
1348
1349
1349
1350
1350
1351
1351
1352
1352
1353
1353
1354
1354
1355
1355
1356
1356
1357
1357
1358
1358
1359
1359
1360
1360
1361
1361
1362
1362
1363
1363
1364
1364
1365
1365
1366
1366
1367
1367
1368
1368
1369
1369
1370
1370
1371
1371
1372
1372
1373
1373
1374
1374
1375
1375
1376
1376
1377
1377
1378
1378
1379
1379
1380
1380
1381
1381
1382
1382
1383
1383
1384
1384
1385
1385
1386
1386
1387
1387
1388
1388
1389
1389
1390
1390
1391
1391
1392
1392
```

CategorieFacade

The screenshot shows the NetBeans IDE interface with the following details:

- Title Bar:** VentesVelos - CategorieFacade.java
- Toolbar:** File, Edit, View, Navigate, Code, Refactor, Build, Run, Tools, VCS, Window, Help, GlassFish 7.0.0
- Project Tree:** VentesVelos / src / main / java / sn / ept / ventesvelos / facades / CategorieFacade
- Code Editor:** CategorieFacade.java

```
1 package sn.ept.ventesvelos.facades;
2
3 import ...
4
5 no usages
6
7 @Stateless
8 public class CategorieFacade extends AbstractFacade{
9     1 usage
10    @PersistenceContext(name="velos_PU")
11    private EntityManager em;
12
13    no usages
14    public CategorieFacade() { super(Categorie.class); }
15
16    @Override
17    protected EntityManager getEntityManager() { return this.em; }
18 }
19
20
21
22
```

- Toolbars:** Version Control, TODO, Problems, Terminal, Profiler, Services, Build, Dependencies
- Status Bar:** All files are up-to-date (3 minutes ago), 3:8, CRLF, UTF-8, 4 spaces

MarqueFacade

The screenshot shows the NetBeans IDE interface with the following details:

- Title Bar:** VentesVelos - MarqueFacade.java
- Toolbar:** File, Edit, View, Navigate, Code, Refactor, Build, Run, Tools, VCS, Window, Help, GlassFish 7.0.0
- Project Tree:** VentesVelos / src / main / java / sn / ept / ventesvelos / facades / MarqueFacade
- Code Editor:** MarqueFacade.java

```
1 package sn.ept.ventesvelos.facades;
2
3 import ...
4
5 no usages
6
7 @Stateless
8 public class MarqueFacade extends AbstractFacade{
9     1 usage
10    @PersistenceContext(name="velos_PU")
11    private EntityManager em;
12
13    no usages
14    public MarqueFacade() { super(Marque.class); }
15
16    @Override
17    protected EntityManager getEntityManager() { return this.em; }
18 }
19
20
21
22
```

- Toolbars:** Version Control, TODO, Problems, Terminal, Profiler, Services, Build, Dependencies
- Status Bar:** All files are up-to-date (3 minutes ago), 3:8, CRLF, UTF-8, 4 spaces

ProduitFacade

```
package sn.ept.ventesvelos.facades;
import ...;
no usages
@Stateless
public class ProduitFacade extends AbstractFacade{
    1 usage
    @PersistenceContext(unitName = "velos_PU")
    private EntityManager em;
    no usages
    public ProduitFacade() { super(Produit.class); }
    @Override
    protected EntityManager getEntityManager() { return this.em; }
}
```

ClientFacade

```
package sn.ept.ventesvelos.facades;
import ...;
no usages
@Stateless
public class ClientFacade extends AbstractFacade{
    1 usage
    @PersistenceContext(unitName = "velos_PU")
    private EntityManager em;
    no usages
    public ClientFacade() {super(Client.class);}
    @Override
    protected EntityManager getEntityManager() {return this.em;}
}
```

CommandeFacade

The screenshot shows the IntelliJ IDEA interface with the file `CommandeFacade.java` open. The code implements a facade pattern:

```
1 package sn.ept.ventesvelos.facades;
2
3 import ...
4
5 no usages
6
7 @Stateless
8 public class CommandeFacade extends AbstractFacade{
9     1 usage
10    @PersistenceContext(unitName = "velos_PU")
11    private EntityManager em;
12
13    no usages
14    public CommandeFacade() {super(Commande.class);}
15
16    @Override
17    protected EntityManager getEntityManager() {return this.em;}
18}
```

The code editor has syntax highlighting for Java and annotations. The status bar at the bottom indicates the file is up-to-date and shows the encoding as CRLF.

EmployeFacade

The screenshot shows the IntelliJ IDEA interface with the file `EmployeFacade.java` open. The code implements a facade pattern:

```
1 package sn.ept.ventesvelos.facades;
2
3 import ...
4
5 no usages
6
7 public class EmployeFacade extends AbstractFacade{
8
9     1 usage
10    @PersistenceContext(unitName = "velos_PU")
11    private EntityManager em;
12
13    no usages
14    public EmployeFacade() {super(Employe.class);}
15
16    @Override
17    protected EntityManager getEntityManager() {return this.em;}
18}
```

The code editor has syntax highlighting for Java and annotations. The status bar at the bottom indicates the file is up-to-date and shows the encoding as CRLF.

MagasinFacade

The screenshot shows the MagasinFacade.java file in an IDE. The code defines a stateless bean named MagasinFacade that extends AbstractFacade. It uses a PersistenceContext with unitName "velos_PU" and a private EntityManager em. The constructor calls super(Magasin.class). The getEntityManager method returns the EntityManager.

```
1 package sn.ept.ventesvelos.facades;
2
3 import ...
4
5 no usages
6
7 @Stateless
8 public class MagasinFacade extends AbstractFacade{
9
10     1 usage
11     @PersistenceContext(unitName = "velos_PU")
12     private EntityManager em;
13
14     no usages
15     public MagasinFacade() {super(Magasin.class);}
16
17     @Override
18     protected EntityManager getEntityManager() {return this.em;}
19 }
```

StockFacade

The screenshot shows the StockFacade.java file in an IDE. The code defines a stateless bean named StockFacade that extends AbstractFacade. It uses a PersistenceContext with unitName "velos_PU" and a private EntityManager em. The constructor calls super(Stock.class). The getEntityManager method returns the EntityManager.

```
1 package sn.ept.ventesvelos.facades;
2
3 import ...
4
5 no usages
6
7 @Stateless
8 public class StockFacade extends AbstractFacade{
9
10     1 usage
11     @PersistenceContext(unitName = "velos_PU")
12     private EntityManager em;
13
14     no usages
15     public StockFacade() {super(Stock.class);}
16
17     @Override
18     protected EntityManager getEntityManager() {return this.em;}
19 }
```

PersonneFacade

```

1 package sn.ept.ventesvelos.facades;
2
3 import jakarta.ejb.Stateless;
4 import jakarta.persistence.EntityManager;
5 import jakarta.persistence.PersistenceContext;
6 import sn.ept.ventesvelos.entities.Personne;
7
8 no usages
9 @Stateless
10 public class PersonneFacade extends AbstractFacade{
11
12     @PersistenceContext(unitName = "velos_PU")
13     private EntityManager em;
14
15     no usages
16     public PersonneFacade() {super(Personne.class);}
17
18     @Override
19     protected EntityManager getEntityManager() {return this.em;}
}

```

2. Initialisation de la BD

Création du script pour l'initialisation de la base de données

On place le fichier vente_velos.sql contenant le script qui insérant les données dans les tables étant donné qu'une fois le projet lancé notre base et les tables sont créées. On place le fichier au niveau du répertoire ressources/sql.

On modifie le script afin qu'il respecte la syntaxe et les fonctionnalités de Postgres.

```

1 -- Insertion des données dans la table categorie
2 INSERT INTO categorie (ID, NOM) VALUES (1, 'Children Bicycles'), (2, 'Comfort Bicycles'), (3, 'Cruisers Bicycles'), (4, 'Cyclocross Bicycles'),
3 (5, 'Electric Bikes'), (6, 'Mountain Bikes'), (7, 'Road Bikes');
4
5 -- Insertion des données dans la table marque
6 INSERT INTO marque (ID, NOM) VALUES (1, 'Electra'), (2, 'Haro'), (3, 'Heller'), (4, 'Pure Cycles'), (5, 'Ritchey'), (6, 'Strider'), (7, 'Sun
7 Bicycles'), (8, 'Surly'), (9, 'Trek');
8
9 -- Insertion des données dans la table magasin
10 INSERT INTO magasin (ID, NOM, TELEPHONE, EMAIL, ADRESSE, VILLE, ETAT, CODE_ZIP) VALUES (1, 'Santa Cruz Bikes', '(831) 476-4321', 'santacruz@bikes
11 .shop', '3700 Portola Drive', 'Santa Cruz', 'CA', '95060'), (2, 'Baldwin Bikes', '(516) 379-8888', 'baldwin@bikes.shop', '4200 Chestnut Lane',
12 'Baldwin', 'NY', '11432'), (3, 'Rowlett Bikes', '(972) 530-5555', 'rowlett@bikes.shop', '8000 Fairway Avenue', 'Rowlett', 'TX', '75088');
13
14 -- Insertion des données dans la table produit
15 INSERT INTO produit (ID, NOM, MARQUE_ID, CATEGORIE_ID, ANNEE_MODEL, PRIX_DEPART) VALUES (1, 'Trek 820 - 2016', 9, 6, 2016, 379.99), (2, 'Ritchey
16 Timberwolf Frameset - 2016', 5, 6, 2016, 749.99), (3, 'Surly Wednesday Frameset - 2016', 8, 6, 2016, 999.99), (4, 'Trek Fuel EX 8 29 - 2016', 9, 6, 2016, 2899
17 .99), (5, 'Heller Shagamaw Frame - 2016', 3, 6, 2016, 1328.99), (6, 'Surly Ice Cream Truck Frameset - 2016', 8, 6, 2016, 469.99), (7, 'Trek Slash 8 27.5 - 2016', 9, 6, 2016, 3999.99), (8, 'Trek Remedy 29 Carbon Frameset - 2016', 9, 6, 2016, 1799.99), (9, 'Trek Conduit+ - 2016', 9, 5, 2016, 2999.99), (10, 'Surly
18 Straggler - 2016', 8, 4, 2016, 1549.00), (11, 'Surly Straggler 650b - 2016', 8, 4, 2016, 1680.99), (12, 'Electra Townie Original 21D - 2016', 1, 3, 2016, 549
19 .99), (13, 'Electra Cruiser 1 (24-Inch) - 2016', 1, 3, 2016, 529.99), (14, 'Electra Girl''s Hawaii 1 (16-inch) - 2015/2016', 1, 3, 2016, 269.99), (15,
20 'Electra Mote 1 - 2016', 1, 3, 2016, 529.99), (16, 'Electra Townie Original 7D EQ - 2016', 1, 3, 2016, 599.99), (17, 'Pure Cycles Vine 8-Speed - 2016', 4, 3, 2016, 429.00), (18, 'Pure Cycles Western 3-Speed - Women''s - 2015/2016', 4, 3, 2016, 449.00), (19, 'Pure Cycles William 3-Speed - 2016', 4, 3, 2016, 449.00), (20, 'Electra Townie Original 7D EQ - Women''s - 2016', 1, 3, 2016, 599.99), (21, 'Electra Cruiser 1 (24-Inch) - 2016', 1, 1, 2016, 269.99), (22, 'Electra Girl''s Hawaii 1 (20-inch) - 2015/2016', 1, 1, 2016, 269.99), (23, 'Electra Girl''s Hawaii 1 (20-inch) - 2015/2016', 1, 1, 2016, 269.99)

```

Création d'une classe InitDB

Nous créons une classe InitBD qui servira à initialiser la base de données à partir du fichier vente_velos.sql obtenu après modification.

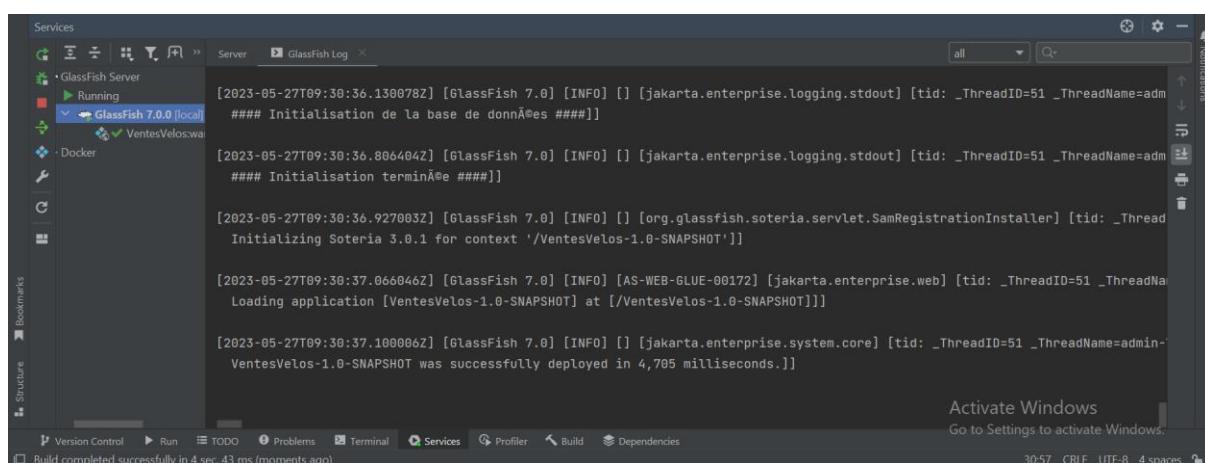
On utilise les annotations `@Singleton` et `@Startup` pour nous assurer qu'un seul objet partagé par tout le monde sera créée et qu'il sera créé au démarrage de l'application.

On crée une variable dataSource avec l'annotation @Resource qui injecte la ressource dataSource responsable de la connexion à la base de données en spécifiant la source de données avec l'argument lookup.

On crée une méthode init avec l'annotation @PostConstruct qui nous permet de demander à l'application d'exécuter cette méthode après la création de la classe. Dans cette méthode on crée un objet Connection pour accéder à la base de données ainsi qu'un objet Statement pour exécuter nos requêtes. On lit le fichier avec InputStream et BufferedReader. Chaque ligne du fichier est exécutée à partir de l'objet Statement.

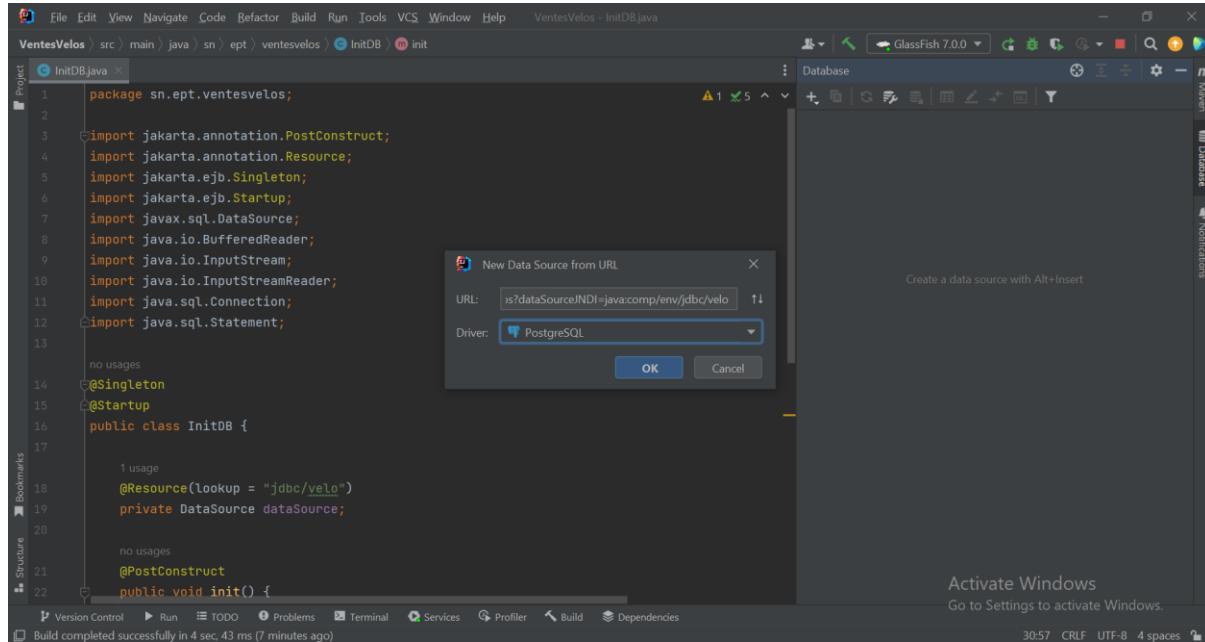
Chargement de la base de données

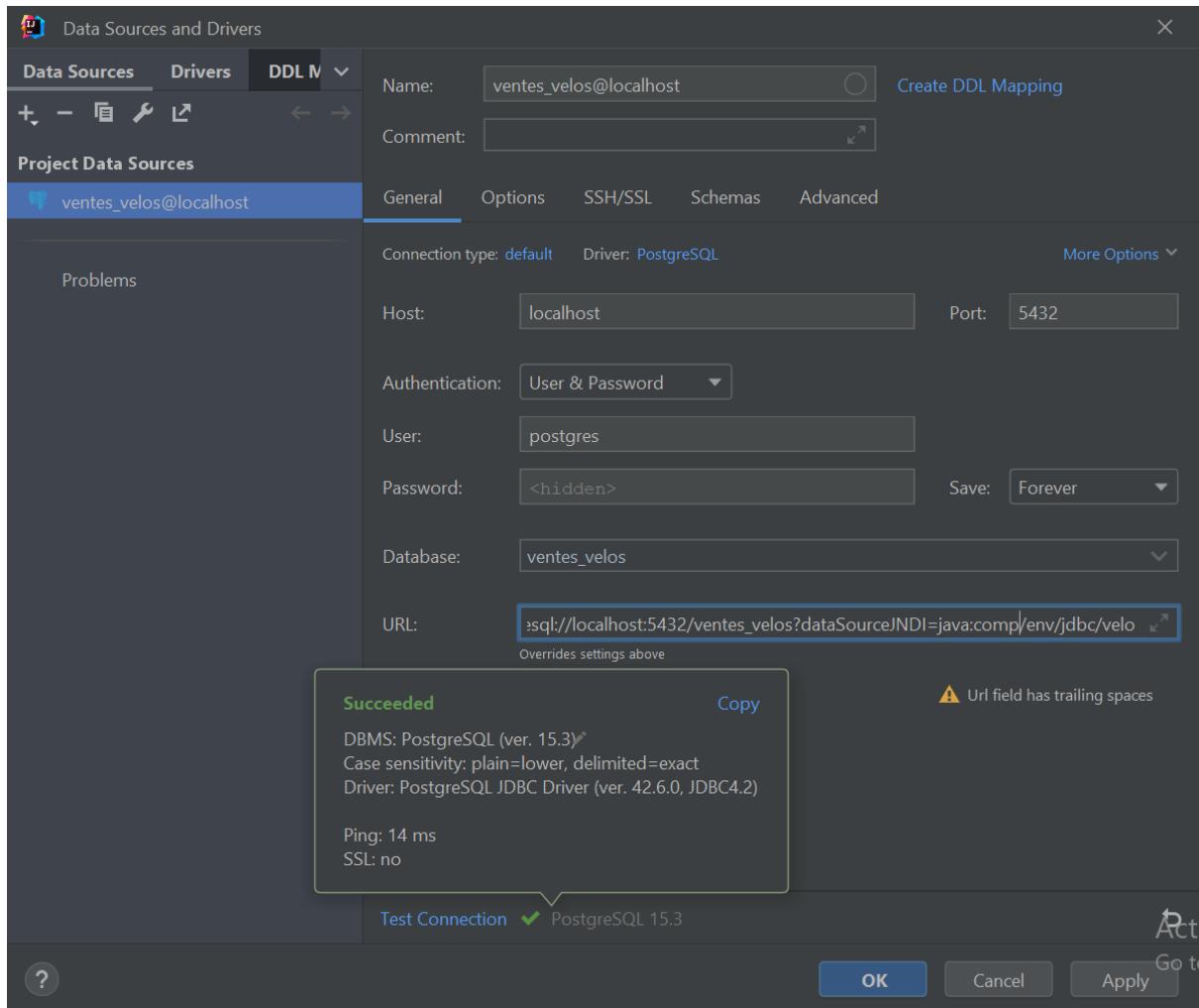
Lorsqu'on déploie le projet, la base de données est créée et remplie.



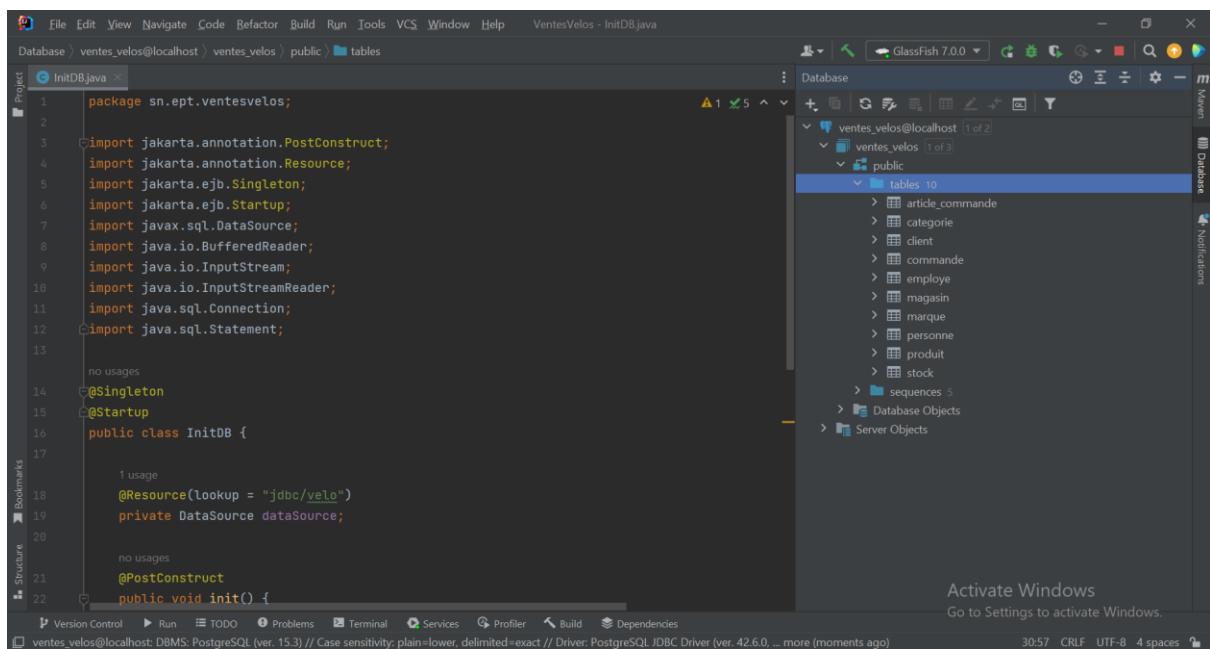
Nous pouvons nous connecter à la base de données à partir d'IntelliJ afin de visualiser les résultats.

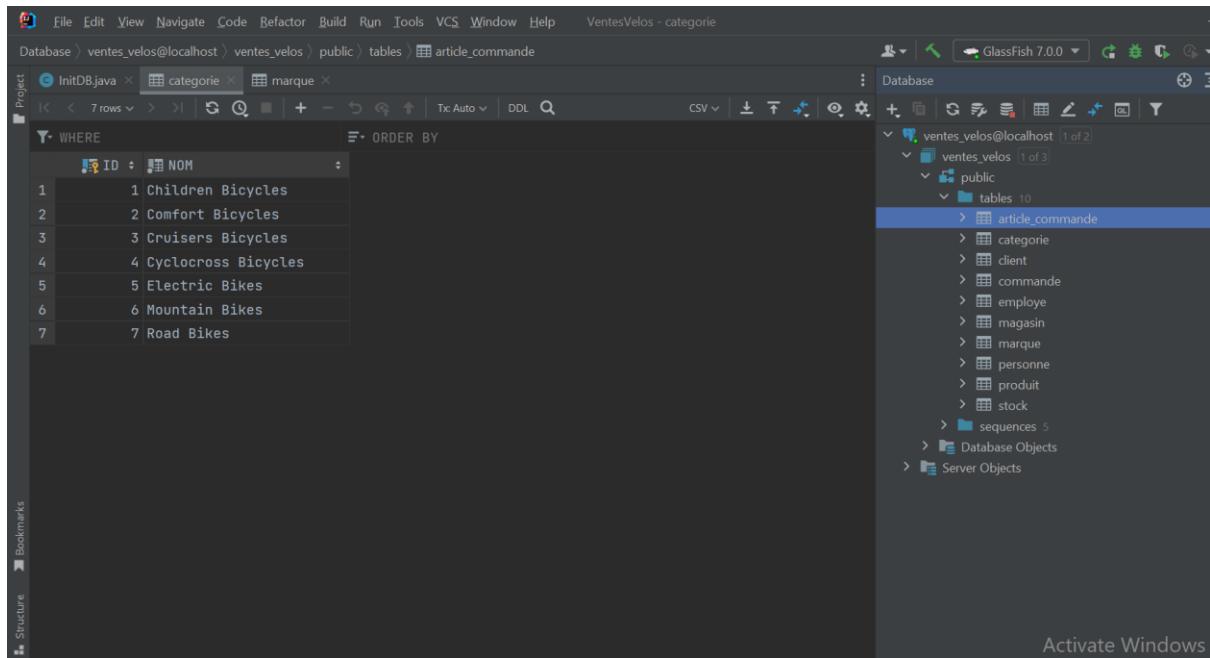
Pour cela ajoute le datasource au niveau de la section Database en utilisant l'option Data Source From URL où l'on spécifie l'emplacement de notre pool précédemment créé.





Nous voyons donc qu'une table a été créée pour chacune de nos entités et que les données du TP précédent ont été insérées.





3. Session Bean pour l'envoi d'email

Etant donné que nous n'avons pas un serveur SMTP local, nous allons utiliser gmail qui offre un serveur SMTP pour l'envoi d'email.

Configurer Gmail comme serveur SMTP

Pour cela nous allons dans les paramètres de notre compte Gmail et allons dans l'onglet *Sécurité*. Ici nous activons la validation en deux étapes :



Activer la validation en deux étapes ?

Deuxième étape : [Invite Google \(par défaut\)](#)

Option de secours : [Message vocal ou SMS](#)

Vous serez toujours connecté au compte tunknowed@gmail.com sur les appareils suivants :
iPhone, Google Intel Apollo Lake Chromebook et Google Intel Apollo Lake Chromebook.

Il se peut que vous soyez déconnecté de votre compte sur vos autres appareils. Pour vous reconnecter, vous devrez saisir votre mot de passe et suivre la deuxième étape de validation.

[ACTIVER](#)

Nous allons ensuite dans la partie *Mot de passe des applications* et nous spécifions l'application et l'appareil pour lequel nous voulons le mot de passe.

← Mots de passe des applications

Les mots de passe d'application vous permettent de vous connecter à votre compte Google à partir d'applications sur des appareils non compatibles avec la validation en deux étapes. Comme vous ne devez saisir le mot de passe qu'une fois, vous n'avez pas besoin de le mémoriser. [En savoir plus](#)

Vous n'avez aucun mot de passe d'application.

Sélectionnez l'application et l'appareil pour lesquels vous souhaitez générer le mot de passe d'application.

Messagerie



Ordinateur Windows



[GÉNÉRER](#)

Mot de passe d'application généré

Votre mot de passe d'application pour ordinateur Windows

lvdg rtzp svcj ptzg

Comment l'utiliser ?

Add your Google account

Enter the information below to connect to your Google account.

Email address

Password

Include your Google contacts and calendars

1. Ouvrez l'application Courrier.
2. Ouvrez le menu "Paramètres".
3. Sélectionnez "Comptes", puis votre compte Google.
4. Remplacez le mot de passe par celui de 16 caractères indiqué ci-dessus.

Tout comme votre mot de passe classique, ce mot de passe spécifique à une application permet d'accorder un accès complet à votre compte Google. Étant donné que vous n'avez pas besoin de le mémoriser, ne le notez nulle part ni ne le partagez avec personne.

[En savoir plus](#)

OK

Nous pouvons tester sur le site <https://www.gmass.co/smtp-test>

SMTP Server	Port	Security
<input type="text" value="smtp.gmail.com"/> Sendgrid Mailgun SMTP2GO sendinblue JangoSMTP GMass	<input type="text" value="587"/> 25 2525 465 587	<input type="text" value="Tls"/>
Username	Password	
<input type="text" value="tunknowed@gmail.com"/>	<input type="password" value="*****"/>	
From email address	To email address	
<input type="text" value="tunknowed@gmail.com"/>	<input type="text" value="tunknowed@gmail.com"/>	
Test it		Activate Windows Go to Settings to activate Windows.

SMTP test from smtp.gmail.com



Téléchargement et ajout du certificat SSL au keystore

```
MINGW64:/c/Users/Massamba/OneDrive/Documents/DIC2/Jakarta/Projet
Massamba@DESKTOP-TT01VCF: MINGW64 ~/OneDrive/Documents/DIC2/Jakarta/Projet
$ openssl s_client -connect smtp.gmail.com:587 -starttls smtp | openssl x509 -outform DER > smtp_gmail_com.crt

Administrator: Command Prompt
Microsoft Windows [Version 10.0.19044.2965]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\system32>cd C:\Users\Massamba\OneDrive\Documents\dic2\jakarta\Projet

C:\Users\Massamba\OneDrive\Documents\dic2\jakarta\Projet>keytool -import -alias smtp.gmail.com -keystore "C:/Program Files/Amazon Corretto/jdk17.0.2_8/lib/security/cacerts" -file smtp_gmail_com.crt
Warning: use -cacerts option to access cacerts keystore
Enter keystore password:
Owner: CN=smtp.gmail.com
Issuer: CN=GTS CA 1C3, O=Google Trust Services LLC, C=US
Serial number: 8b4634b92ba40a330a24067f95349db5
Valid from: Mon May 08 08:24:44 UTC 2023 until: Mon Jul 31 08:24:43 UTC 2023
Certificate fingerprints:
    SHA1: 68:60:0A:80:E1:BA:B0:AC:E5:92:14:4C:3B:F4:3C:09:63:C4:48:8B
    SHA256: D5:79:FB:63:7A:89:BA:1C:9D:0C:14:E2:1E:C4:8D:DC:28:F6:F8:ED:17:75:48:0B:F4:45:72:D8:D4:A6:BA:B2
Signature algorithm name: SHA256withRSA
Subject Public Key Algorithm: 256-bit EC (secp256r1) key
Version: 3

Extensions:

#1: ObjectId: 1.3.6.1.4.1.11129.2.4.2 Criticality=false
0000: 04 81 F2 00 F0 00 77 00 AD F7 BE FA 7C FF 10 C8 .....w.....
0010: 8B 9D 3D 9C 1E 3E 18 6A B4 67 29 5D CF B1 0C 24 ...=>.j.g)]...$ 
0020: CA 85 86 34 EB DC 82 8A 00 00 01 87 FA AD 7B 79 ...4.....y
0030: 00 00 04 03 00 48 30 46 02 21 00 A3 7D C5 87 6E .....H0F.!....n
0040: 03 85 34 21 0B 33 E8 41 10 4A F4 FD 09 E2 6E 54 ..4!.3.A.J....nT
0050: CD A5 37 AB 37 06 75 91 C4 4F 24 02 21 00 D8 80 ..7.7.u..0$.!...
0060: FB 5F 98 E5 92 36 BA 3D 0E E2 B4 3C 5F 78 44 06 .._.6.=...<_xD.
```

Ajout de la dépendance jakarta mail

Elle est utilisée pour gérer l'envoi des emails. Pour s'assurer que les bibliothèques nécessaires sont dans notre classpath, nous ajoutons donc la dépendance dans notre pom.xml

```

<dependency>
    <groupId>org.junit.jupiter</groupId>
    <artifactId>junit-jupiter-engine</artifactId>
    <version>${junit.version}</version>
    <scope>test</scope>
</dependency>
<dependency>
    <groupId>org.postgresql</groupId>
    <artifactId>postgresql</artifactId>
    <version>42.6.0</version>
</dependency>
<dependency>
    <groupId>com.sun.mail</groupId>
    <artifactId>jakarta.mail</artifactId>
    <version>1.6.7</version>
</dependency>
</dependencies>

<build>
    <plugins>
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-war-plugin</artifactId>
            <version>3.3.2</version>
        </plugin>
    </plugins>
</build>

```

Création du Bean pour l'envoi d'email

On crée une classe SendEmailBean avec l'annotation @Stateless qui permet de signifier que l'objet peut être détruit ou redonné après utilisation mais sans conservation d'information. On crée également les variables pour les paramètres d'accès au serveur SMTP.

On ajoute une méthode sendEmail qui sera utilisé pour l'envoi des emails. Dans celle-ci on effectue la configuration du serveur SMTP à l'aide de l'objet Properties.

Ensuite on crée une session avec authentification et on met en place la logique pour créer et envoyer le message.

```

@Stateless
public class SendEmailBean {
    private static final String smtpHost = "smtp.gmail.com";
    private static final int smtpPort = 587;
    private static final String username = "tunknowed@gmail.com";
    private static final String password = "lvdgrtzpsvcjptza";

    public void sendEmail(String destinataire, String sujet, String contenu){
        // Ajout des paramètres du serveur SMTP
        Properties properties = new Properties();
        properties.put("mail.smtp.host", smtpHost);
        properties.put("mail.smtp.port", smtpPort);
        properties.put("mail.smtp.auth", "true");
        properties.put("mail.smtp.starttls.enable", "true");
        properties.put("mail.smtp.ssl.trust", smtpHost);

        // Création d'une session avec authentification
        Session session = Session.getInstance(properties, new Authenticator() {

```

The screenshot shows the NetBeans IDE interface with the following details:

- Top Bar:** File, Edit, View, Navigate, Code, Refactor, Build, Run, Tools, VCS, Window, Help, VentesVelos - SendEmailBean.java
- Toolbar:** GlassFish 7.0.0, various icons for file operations.
- Project Tree:** VentesVelos > src > main > java > sn > ept > ventesvelos > SendEmailBean.java
- Code Editor:** The current file is SendEmailBean.java, showing the following code:

```
2 usages
29 @Override
30     protected PasswordAuthentication getPasswordAuthentication() {
31         return new PasswordAuthentication(username, password);
32     };
33
34     try {
35         // Création du message
36         Message message = new MimeMessage(session);
37         message.setFrom(new InternetAddress(username));
38         message.setRecipients(Message.RecipientType.TO, InternetAddress.parse(destinataire));
39         message.setSubject(sujet);
40         message.setText(contenu);
41         // Envoi du message
42         Transport.send(message);
43         System.out.println("Le message a été envoyé avec succès");
44     } catch (MessagingException e) {
45         // System.out.println("Le message n'a pas pu être envoyé " + e.getMessage());
46         e.printStackTrace();
47     }
48
49
50 }
```

The code implements a Java Bean for sending emails using JavaMail. It overrides the getPasswordAuthentication method to return a PasswordAuthentication object. The main logic is in a try block where it creates a MimeMessage, sets the recipient, subject, and text, then sends it using the Transport class. A catch block handles MessagingException.

4. Envoi d'alertes automatique

Créer un EJB qui envoie un e-mail à chaque démarrage et arrêt de l'application

Nous créons une classe `NotifyApplicationStatusBean` avec l'annotation `@Singleton` pour nous assurer qu'un seul objet partagé par tout le monde sera créé et `@Startup` qui permet de l'initialiser au démarrage de l'application.

On utilise l'annotation `@Inject` pour obtenir une instance de `SendEmailBean` qui sera utilisée pour l'envoi des emails.

On crée une méthode appStart avec l'annotation @PostConstruct ce qui signifie qu'elle sera exécutée automatiquement après création de l'instance et dans celle-ci on envoie un mail qui informe du démarrage de l'application.

On crée une méthode appStop avec l'annotation @PreDestroy ce qui signifie qu'elle sera exécutée avant la destruction de l'EJB lors de l'arrêt de l'application et dans celle-ci on envoie un email qui informe de l'arrêt.

- Démarrage de l'application
En démarrant l'application à 14h41 nous voyons qu'un email a au même moment été envoyé au niveau de notre adresse

```
C:\Users\Massamba\dic2_2023\glassfish7\glassfish\bin\asadmin.bat start-domain domain1
"C:\Program Files\Amazon Corretto\jdk17.0.2_8\bin\java.exe" -Dfile.encoding=windows-1252 -classpath C:\Users\Massamba\A
[2023-05-27 02:41:14,100] Artifact VentesVelos:war exploded: Waiting for server connection to start artifact deployment
Detected server admin port: 4848
Detected server http port: 8080
Attempting to start domain1.... Please look at the server log for more details.....
Connected to server
[2023-05-27 02:41:32,418] Artifact VentesVelos:war exploded: Artifact is being deployed, please wait...
[2023-05-27 02:41:58,714] Artifact VentesVelos:war exploded: Artifact is deployed successfully
[2023-05-27 02:41:58,715] Artifact VentesVelos:war exploded: Deploy took 26,297 milliseconds
```

Démarrage de l'application Boîte de réception x

tunknowed@gmail.com 14:41 (il y a 0 minute) ★ ↵ :

À moi ▾

L'application a été démarrée

...

[Message tronqué] [Afficher l'intégralité du message](#)

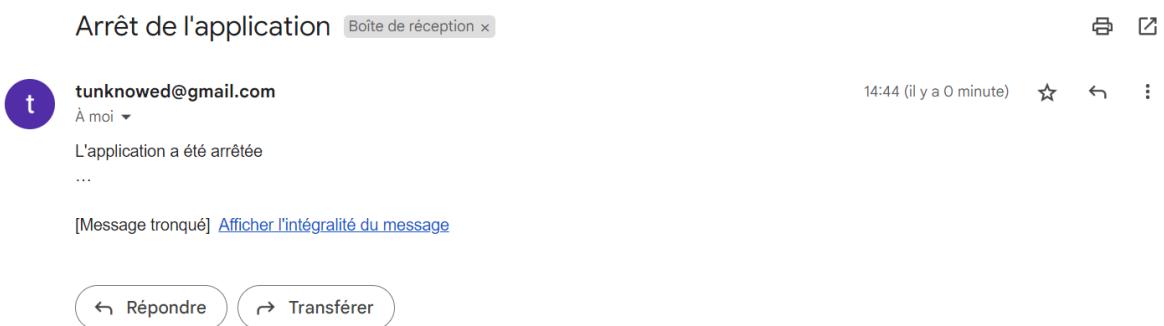
Répondre Transférer

- Arrêt de l'application

En démarrant l'application à 14h44 nous voyons qu'un email a au même moment été envoyé au niveau de notre adresse

```
Connected to server
[2023-05-27 02:41:32,418] Artifact VentesVelos:war exploded: Artifact is being deployed, please wait...
[2023-05-27 02:41:58,714] Artifact VentesVelos:war exploded: Artifact is deployed successfully
[2023-05-27 02:41:58,715] Artifact VentesVelos:war exploded: Deploy took 26,297 milliseconds
C:\Users\Massamba\dic2_2023\glassfish7\glassfish\bin\asadmin.bat stop-domain domain1
[2023-05-27 02:44:58,604] Artifact VentesVelos:war exploded: Artifact is not deployed. Press 'Deploy' to start deployment

Process finished with exit code 0
Waiting for the domain to stop .
Waiting finished after 33 ms.
Command stop-domain executed successfully.
Disconnected from server
```



Créez un EJB qui envoie périodiquement par e-mail l'état des stocks des produits. Vous pouvez utiliser les Timers avec les annotations @Schedules et/ou @Schedule pour le faire (voir références [3, 4]).

Nous créons une classe StockNotificationBean qui sera utilisée pour l'envoi des emails.

On utilise l'annotation `@Singleton` pour nous assurer qu'un seul objet partagé par tout le monde sera créé et `@Startup` qui permet de l'initialiser au démarrage de l'application.

On utilise l'annotation `@Inject` pour obtenir une instance de `SendEmailBean` qui sera utilisée pour l'envoi des emails ainsi qu'une variable pour vérifier si c'est l'envoie initial a été fait. On augmente également la variable `stockFacade` avec l'annotation `@EJB` que l'on va utiliser pour l'accès aux opérations de manipulation des stocks afin de récupérer l'état des stocks des produits.

On crée une méthode `getStockInformation` dans laquelle nous récupérons tous les objets de l'entité `Stock` grâce à la méthode `findAll()` de `stockFacade` et on les met dans un paragraphe chacun d'eux étant séparé par un retour à la ligne. Ceci nous permet de connaître l'état actuel des stocks.

On crée une méthode `periodicStockEmail` annotée avec `@Schedule` qui permet de réexécuter la méthode par la période spécifiée et on envoie le mail avec comme contenu le résultat de l'appel de la méthode `getStockInformation()`.

Pour tester nous avons mis la période à 1 minute et nous voyons donc qu'un mail est envoyé chaque minute avec l'état des stocks.

```
1 usage
14     @Inject
15     private SendEmailBean emailSender;
16
17     @EJB
18     private StockFacade stockFacade;
19
20     1 usage
21     public String getStockInformation() {
22         String stockInformation = "L'état actuel des stocks est le suivant: \n\n";
23         List<Stock> stockList = stockFacade.findAll();
24         for (Stock e: stockList) {
25             stockInformation += "Le stock au magasin" + e.getMagasinId().getNom() + " pour le produit " + e.getProduitId().getNom() + " est de " + e.getQuantite();
26         }
27         return stockInformation;
28     }
29
30     no usages
31     @Schedule(hour="*", minute = "*/1", persistent = false)
32     public void periodicStockEmail(Timer timer) {
33         emailSender.sendEmail( destinataire: "tunknowed@gmail.com", sujet: "Etat des stocks",
34                               contenu: "Voici l'état actuel des stocks " + getStockInformation());
35     }
36 }
```

Gmail

Rechercher dans les messages

Nouveau message

Boîte de réception... 1115

Messages suivis

En attente

Messages envoyés

Brouillons 1

Plus

Libellés +

Bénéficiez d'une protection supplémentaire contre l'hameçonnage

Continuer Non, merci

Principale Promotions 50 nouveaux Réseaux sociaux 50 nouveaux

Etat des stocks - Voici l'état actuel des stocks L'état actuel des stocks est le suivant: ... 08:29

Etat des stocks - Voici l'état actuel des stocks L'état actuel des stocks est le suivant: ... 08:28

Démarrage de l'application - L'application a été démarrée 08:26

Why do some people compare Ronaldo with Messi? Ronaldo is a striker and Messi the ... 18 juil.

Why is Messi so slow in adjusting to his god-like level in PSG unlike other elite str... 18 juil.

Etat des stocks Boîte de réception x

tunknowed@gmail.com A moi

Voici l'état actuel des stocks L'état actuel des stocks est le suivant:

Le stock au magasin Santa Cruz Bikes pour le produit Trek 820 - 2016 est de 27

Le stock au magasin Santa Cruz Bikes pour le produit Ritchey Timberwolf Frameset - 2016 est de 5

Le stock au magasin Santa Cruz Bikes pour le produit Surly Wednesday Frameset - 2016 est de 6

Le stock au magasin Santa Cruz Bikes pour le produit Trek Fuel EX 29 - 2016 est de 23

Le stock au magasin Santa Cruz Bikes pour le produit Heller Shagamaw Frame - 2016 est de 22

Le stock au magasin Santa Cruz Bikes pour le produit Surly Ice Cream Truck Frameset - 2016 est de 0

Le stock au magasin Santa Cruz Bikes pour le produit Trek Slash 8.27.5 - 2016 est de 8

Le stock au magasin Santa Cruz Bikes pour le produit Trek Remedy 29 Carbon Frameset - 2016 est de 0

Le stock au magasin Santa Cruz Bikes pour le produit Trek Conduit+ - 2016 est de 11

Le stock au magasin Santa Cruz Bikes pour le produit Surly Straggler - 2016 est de 15

Nous voyons donc que chaque minute un mail est envoyé avec l'état des stocks. Ceci serait encombrant donc on met la période à chaque heure.

The screenshot shows a Java code editor within an IDE. The file being edited is `StockNotificationBean.java`. The code implements a singleton bean for stock notifications.

```
1.0  package com.ventesvelos;
1.1  import javax.annotation.PostConstruct;
1.2  import javax.ejb.Singleton;
1.3  import javax.ejb.Startup;
1.4  import javax.inject.Inject;
1.5  import com.ventesvelos.model.SendEmailBean;
1.6  import com.ventesvelos.model.StockFacade;
1.7
1.8  public class StockNotificationBean {
1.9
1.10     @Inject
1.11     private SendEmailBean emailSender;
1.12
1.13     @Startup
1.14     @Singleton
1.15     public void onStart() {
1.16
1.17         String stockInformation = "";
1.18         List<Stock> stockList = stockFacade.findAll();
1.19         for (Stock e: stockList) {
1.20             stockInformation += e.toString() + "\n";
1.21         }
1.22         return stockInformation;
1.23     }
1.24
1.25     public String getStockInformation() {
1.26         String stockInformation = "";
1.27         List<Stock> stockList = stockFacade.findAll();
1.28         for (Stock e: stockList) {
1.29             stockInformation += e.toString() + "\n";
1.30         }
1.31         return stockInformation;
1.32     }
1.33
1.34     @Schedule(hour="*", minute = "0", persistent = false)
1.35     public void periodicStockEmail(Timer timer) {
1.36         emailSender.sendEmail(_destinataire: "tunknowed@gmail.com", sujet: "Etat des stocks", contenu: "Voici l'état actuel des stocks " + getStockInformation());
1.37     }
1.38 }
```

The code includes annotations for `@Singleton`, `@Startup`, `@Inject`, and `@Schedule`. It uses `SendEmailBean` and `StockFacade` from the `com.ventesvelos` package. The `getStockInformation` method retrieves all stock data from the facade and returns it as a string. The scheduled task sends an email to a specified recipient with the current stock status.

IV. Développement des interfaces web JSF

1. CRUD sur les entités

A. Création des MBeans

Avant de créer les pages nous devons d'abord créer les Managed Beans qui permettront d'effectuer les opérations CRUD sur les entités.

On utilise `@Named` et `@ViewScopped` pour marquer le Mbean comme un composant JSF nommé et définir l'étendue du Mbean comme étant la vue JSF.

On injecte la façade correspondant à la classe avec l'annotation `@EJB`.

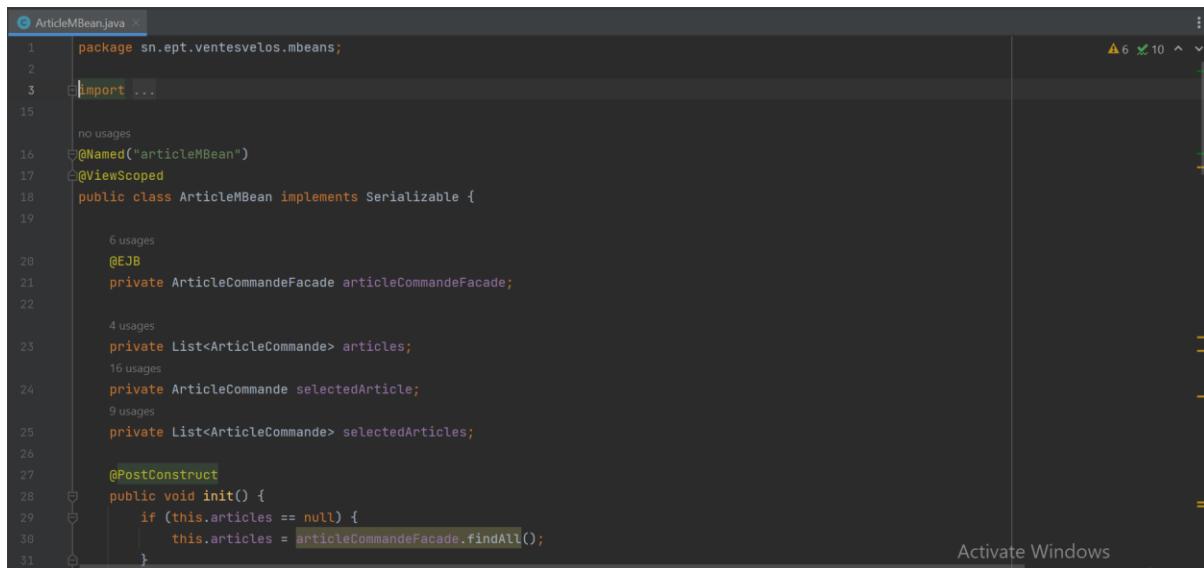
On crée les attributs pour avoir l'ensemble des entités, une entité spécifique sélectionnée ainsi que l'ensemble des entités sélectionnés. On ajoute des getters et setters pour ces attributs.

On ajoute une méthode `init()` avec l'annotation `@PostConstruct` pour spécifier qu'elle doit être exécutée après création de l'objet où on initialise l'ensemble des entités.

On finit par créer les méthodes pour modifier, enregistrer et supprimer les entités sélectionnées tout en envoyant des messages au niveau de la page JSF.

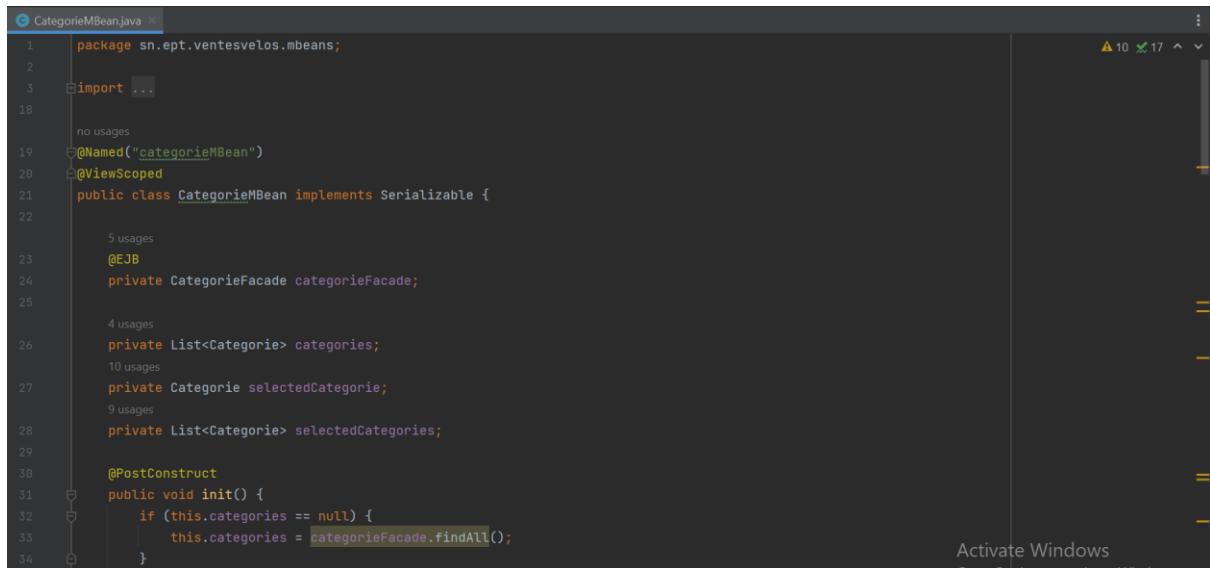
Ici on ajoute un aperçu de chaque MBean et vous pourrez voir plus en détails au niveau du code

ArticleMBean



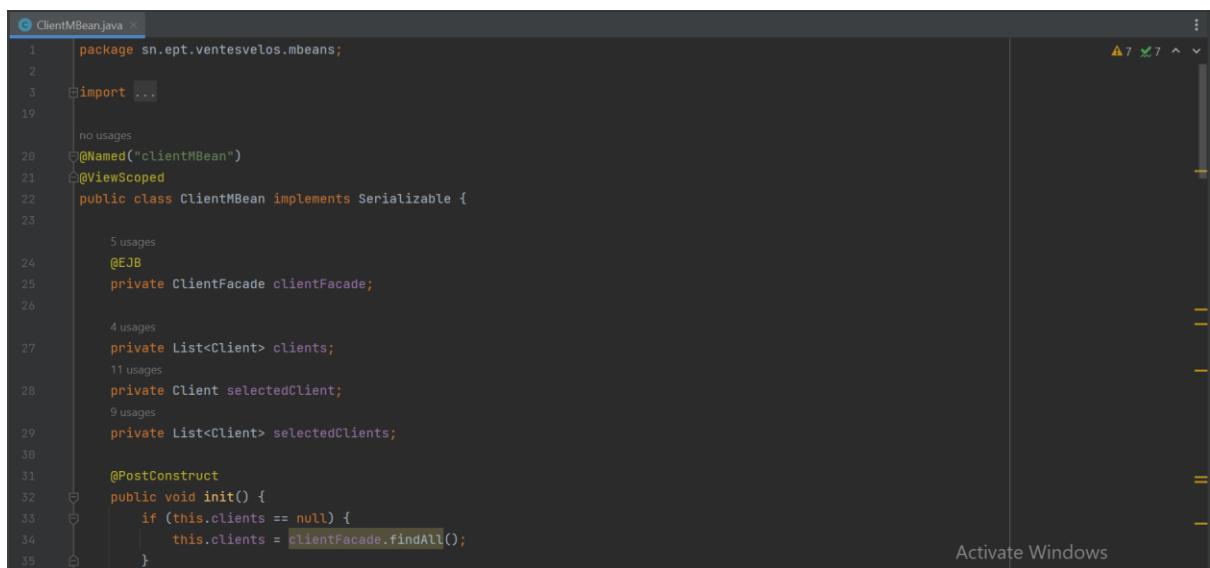
```
ArticleMBean.java
1 package sn.ept.ventesvelos.mbeans;
2
3 import ...
4
5 no usages
6
7 @Named("articleMBean")
8 @ViewScoped
9 public class ArticleMBean implements Serializable {
10
11     6 usages
12     @EJB
13     private ArticleCommandeFacade articleCommandeFacade;
14
15     4 usages
16     private List<ArticleCommande> articles;
17     16 usages
18     private ArticleCommande selectedArticle;
19     9 usages
20     private List<ArticleCommande> selectedArticles;
21
22     @PostConstruct
23     public void init() {
24         if (this.articles == null) {
25             this.articles = articleCommandeFacade.findAll();
26         }
27     }
28 }
```

CategorieMBean



```
1 package sn.ept.ventesvelos.mbeans;
2
3 import ...
4
5 no usages
6
7 @Named("categorieMBean")
8 @ViewScoped
9 public class CategorieMBean implements Serializable {
10
11     5 usages
12     @EJB
13     private CategorieFacade categorieFacade;
14
15     4 usages
16     private List<Categorie> categories;
17     10 usages
18     private Categorie selectedCategorie;
19     9 usages
20     private List<Categorie> selectedCategories;
21
22     @PostConstruct
23     public void init() {
24         if (this.categories == null) {
25             this.categories = categorieFacade.findAll();
26         }
27     }
28 }
```

ClientMBean



```
1 package sn.ept.ventesvelos.mbeans;
2
3 import ...
4
5 no usages
6
7 @Named("clientMBean")
8 @ViewScoped
9 public class ClientMBean implements Serializable {
10
11     5 usages
12     @EJB
13     private ClientFacade clientFacade;
14
15     4 usages
16     private List<Client> clients;
17     11 usages
18     private Client selectedClient;
19     9 usages
20     private List<Client> selectedClients;
21
22     @PostConstruct
23     public void init() {
24         if (this.clients == null) {
25             this.clients = clientFacade.findAll();
26         }
27     }
28 }
```

CommandeMBean

```
15 no usages
16 @Named("commandeMBean")
17 @ViewScoped
18 public class CommandeMBean implements Serializable {
19
20     5 usages
21     @EJB
22     private CommandeFacade commandeFacade;
23
24     4 usages
25     private List<Commande> commandes;
26     10 usages
27     private Commande selectedCommande;
28     9 usages
29     private List<Commande> selectedCommandes;
30
31     @PostConstruct
32     public void init() {
33         if (this.commandes == null) {
34             this.commandes = commandeFacade.findAll();
35         }
36     }
37
38     2 usages
39 }
```

EmployeeMBean

The screenshot shows an IDE interface with a code editor and various status indicators. The code editor displays the `EmployeeMBean.java` file. The code itself is as follows:

```
EmployeeMBean.java ×  
no usages  
17     @Named("employeeMBean")  
18     @ViewScoped  
19     public class EmployeeMBean implements Serializable {  
20  
    5 usages  
21        @EJB  
22        private EmployeFacade employeFacade;  
23  
    4 usages  
24        private List<Employe> employees;  
    13 usages  
25        private Employe selectedEmploye;  
    9 usages  
26        private List<Employe> selectedEmployees;  
27  
28        @PostConstruct  
29        public void init() {  
30            if (this.employees == null) {  
31                this.employees = employeFacade.findAll();  
32            }  
33        }  
34  
    3 usages  
35        public List<Employe> getEmployees() { return employees; }
```

Annotations and variables are color-coded: `@Named`, `@ViewScoped`, `@EJB`, `employeFacade`, `employees`, `selectedEmploye`, `selectedEmployees`, and `getEmployees()` are highlighted in orange or purple. The code editor has a dark theme with light-colored syntax highlighting. On the right side, there are several status icons: a green checkmark, a yellow warning triangle, and a red error circle, followed by the numbers "6", "33", and "1". Below the code editor, a status bar displays the text "Activate Windows".

MagasinMBean

```
17 no usages
18 @Named("magasinMBean")
19 @ViewScoped
20 public class MagasinMBean implements Serializable {
21
22     5 usages
23     @EJB
24     private MagasinFacade magasinFacade;
25
26     4 usages
27     private List<Magasin> magasins;
28     11 usages
29     private Magasin selectedMagasin;
30     9 usages
31     private List<Magasin> selectedMagasins;
32
33     @PostConstruct
34     public void init() {
35         if (this.magasins == null) {
36             this.magasins = magasinFacade.findAll();
37         }
38     }
39
40     4 usages
41     public List<Magasin> getMagasins() { return magasins; }
```

MarqueMBean

```
19 no usages
20 @Named("marqueMBean")
21 @ViewScoped
22 public class MarqueMBean implements Serializable {
23
24     5 usages
25     @EJB
26     private MarqueFacade marqueFacade;
27
28     4 usages
29     private List<Marque> marques;
30     10 usages
31     private Marque selectedMarque;
32     9 usages
33     private List<Marque> selectedMarques;
34
35     @PostConstruct
36     public void init() {
37         if (this.marques == null) {
38             this.marques = marqueFacade.findAll();
39         }
40     }
41
42     2 usages
43     public List<Marque> getMarques() { return marques; }
```

ProduitMBean

```
15  no usages
16  @Named("produitMBean")
17  @ViewScoped
18  public class ProduitMBean implements Serializable {
19
20      5 usages
21      @EJB
22      private ProduitFacade produitFacade;
23
24      4 usages
25      private List<Produit> produits;
26      10 usages
27      private Produit selectedProduit;
28      9 usages
29      private List<Produit> selectedProduits;
30
31      @PostConstruct
32      public void init() {
33          if (this.produits == null) {
34              this.produits = produitFacade.findAll();
35          }
36      }
37
38      3 usages
39      public List<Produit> getProduits() { return produits; }
```

StockMBean

```
16  no usages
17  @Named("stockMBean")
18  @ViewScoped
19  public class StockMBean implements Serializable {
20
21      6 usages
22      @EJB
23      private StockFacade stockFacade;
24
25      4 usages
26      private List<Stock> stocks;
27      14 usages
28      private Stock selectedStock;
29      9 usages
30      private List<Stock> selectedStocks;
31
32      @PostConstruct
33      public void init() {
34          if (this.stocks == null) {
35              this.stocks = stockFacade.findAll();
36          }
37      }
38
39      1 usage
40      public List<Stock> getStocks() { return stocks; }
```

B. Création des converters

Etant donné que pour les clés étrangères nous devrons enregistrer des entités, nous utilisons les converters avec l'annotation `@FacesConverter` qui est responsable de la conversion des valeurs entre les objets de modèle et les représentations de chaîne utilisées dans l'interface utilisateur.

Dans JSF, les convertisseurs sont utilisés pour gérer la conversion des valeurs d'entrée fournies par l'utilisateur en types de données correspondants dans le modèle de l'application, et vice versa.

Dans ces classes nous injectons les façades correspondantes en utilisant `@EJB`, ensuite nous redéfinissons les méthodes `getAsString()` et `getAsObject()` qui permettent d'obtenir leur représentation en chaîne et en objet.

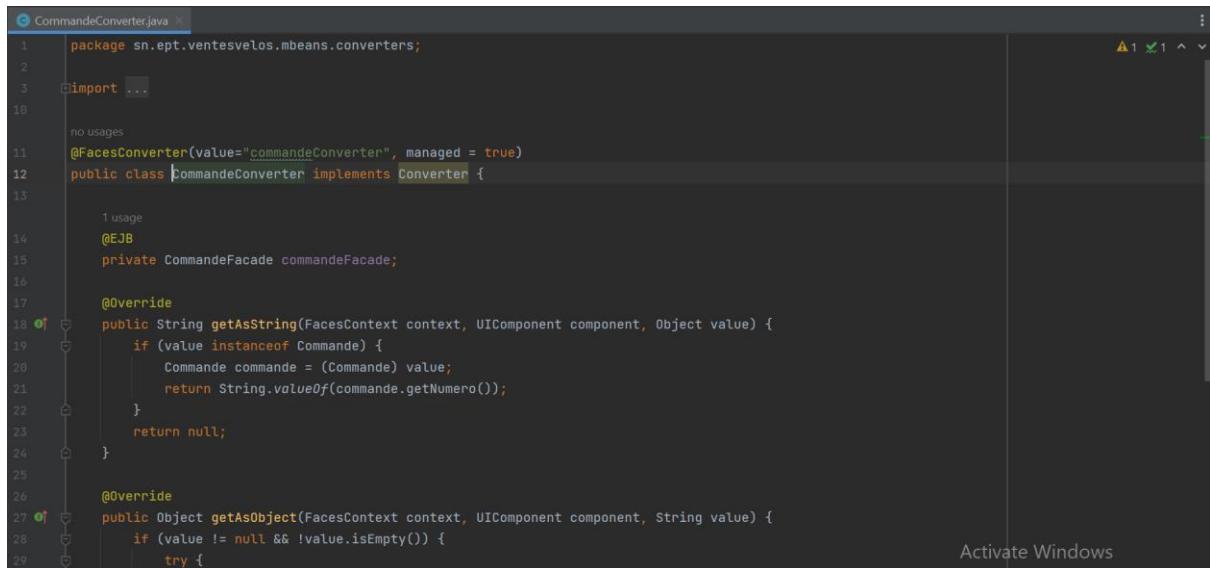
CategorieConverter

```
CategorieConverter.java
1 package sn.ept.ventesvelos.mbeans.converters;
2
3 import ...
4
5 no usages
6 @FacesConverter(value = "categorieConverter", managed = true)
7 public class CategorieConverter implements Converter {
8
9     1 usage
10    @EJB
11    private CategorieFacade categorieFacade;
12
13    @Override
14    public String getAsString(FacesContext context, UIComponent component, Object value) {
15        if (value instanceof Categorie) {
16            Categorie categorie = (Categorie) value;
17            return String.valueOf(categorie.getId());
18        }
19        return null;
20    }
21
22    @Override
23    public Object getAsObject(FacesContext context, UIComponent component, String value) {
24        if (value != null && !value.isEmpty()) {
25            try {
26                Activate Windows
```

ClientConverter

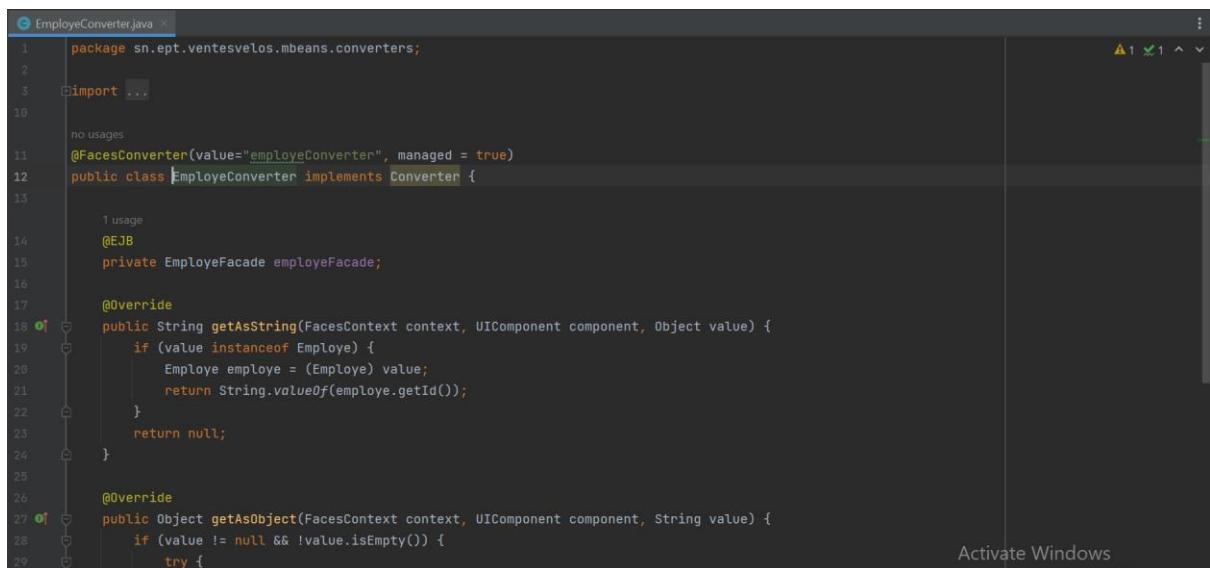
```
ClientConverter.java
1 package sn.ept.ventesvelos.mbeans.converters;
2
3 import ...
4
5 no usages
6 @FacesConverter(value="clientConverter", managed = true)
7 public class ClientConverter implements Converter {
8
9     1 usage
10    @EJB
11    private ClientFacade clientFacade;
12
13    @Override
14    public String getAsString(FacesContext context, UIComponent component, Object value) {
15        if (value instanceof Client) {
16            Client client = (Client) value;
17            return String.valueOf(client.getId());
18        }
19        return null;
20    }
21
22    @Override
23    public Object getAsObject(FacesContext context, UIComponent component, String value) {
24        if (value != null && !value.isEmpty()) {
25            try {
26                Activate Windows
```

CommandeConverter



```
1 package sn.ept.ventesvelos.mbeans.converters;
2
3 import ...
4
5 no usages
6
7 @FacesConverter(value="commandeConverter", managed = true)
8 public class CommandeConverter implements Converter {
9
10     1 usage
11     @EJB
12     private CommandeFacade commandeFacade;
13
14     @Override
15     public String getAsString(FacesContext context, UIComponent component, Object value) {
16         if (value instanceof Commande) {
17             Commande commande = (Commande) value;
18             return String.valueOf(commande.getNumero());
19         }
20         return null;
21     }
22
23     @Override
24     public Object getAsObject(FacesContext context, UIComponent component, String value) {
25         if (value != null && !value.isEmpty()) {
26             try {
27                 ...
28             }
29         }
30     }
31 }
```

EmployeConverter



```
1 package sn.ept.ventesvelos.mbeans.converters;
2
3 import ...
4
5 no usages
6
7 @FacesConverter(value="employeConverter", managed = true)
8 public class EmployeConverter implements Converter {
9
10     1 usage
11     @EJB
12     private EmployeFacade employeFacade;
13
14     @Override
15     public String getAsString(FacesContext context, UIComponent component, Object value) {
16         if (value instanceof Employe) {
17             Employe employe = (Employe) value;
18             return String.valueOf(employe.getId());
19         }
20         return null;
21     }
22
23     @Override
24     public Object getAsObject(FacesContext context, UIComponent component, String value) {
25         if (value != null && !value.isEmpty()) {
26             try {
27                 ...
28             }
29         }
30     }
31 }
```

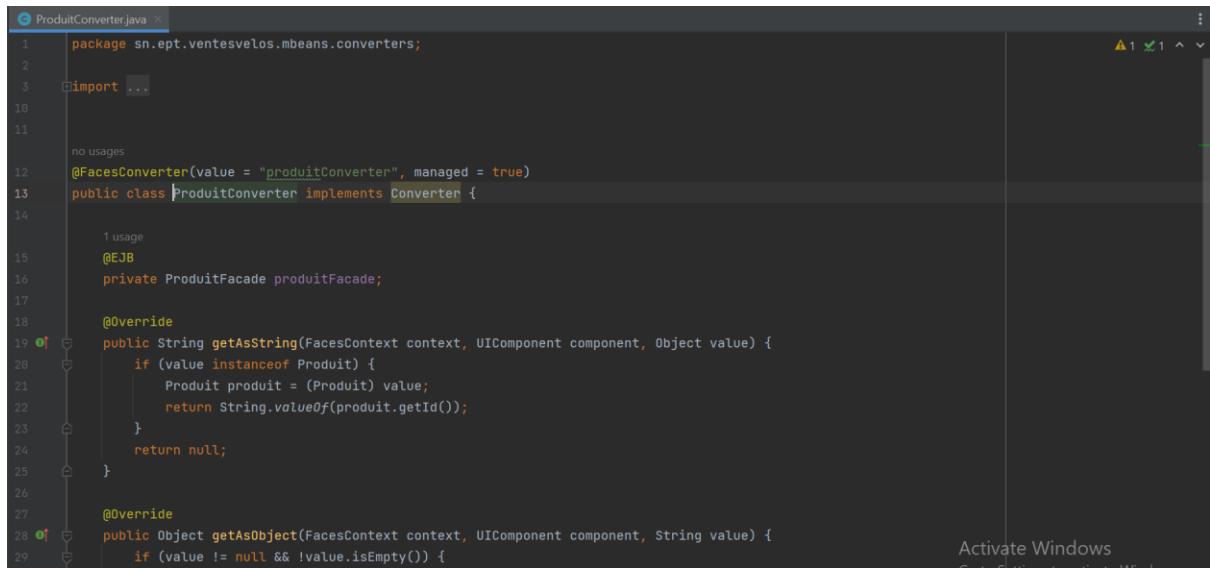
MagasinConverter

```
MagasinConverter.java ×
1 package sn.ept.ventesvelos.mbeans.converters;
2
3 import ...
4
5 no usages
6 @FacesConverter(value = "magasinConverter", managed = true)
7 public class MagasinConverter implements Converter {
8
9     1 usage
10    @EJB
11    private MagasinFacade magasinFacade;
12
13    @Override
14    public String getAsString(FacesContext context, UIComponent component, Object value) {
15        if (value instanceof Magasin) {
16            Magasin magasin = (Magasin) value;
17            return String.valueOf(magasin.getId());
18        }
19        return null;
20    }
21
22    @Override
23    public Object getAsObject(FacesContext context, UIComponent component, String value) {
24        if (value != null && !value.isEmpty()) {
25            try {
26                Activate Windows
27                MagasinFacade magasinFacade = ...
28                Magasin magasin = magasinFacade.find(Integer.parseInt(value));
29                return magasin;
29            } catch (Exception e) {
29        }
29    }
29}
```

MarqueConverter

```
MarqueConverter.java ×
1 package sn.ept.ventesvelos.mbeans.converters;
2
3 import ...
4
5 no usages
6 @FacesConverter(value="marqueConverter", managed = true)
7 public class MarqueConverter implements Converter {
8
9     1 usage
10    @EJB
11    private MarqueFacade marqueFacade;
12
13    @Override
14    public String getAsString(FacesContext context, UIComponent component, Object value) {
15        if (value instanceof Marque) {
16            Marque marque = (Marque) value;
17            return String.valueOf(marque.getId());
18        }
19        return null;
20    }
21
22    @Override
23    public Object getAsObject(FacesContext context, UIComponent component, String value) {
24        if (value != null && !value.isEmpty()) {
25            try {
26                Activate Windows
27                MarqueFacade marqueFacade = ...
28                Marque marque = marqueFacade.find(Integer.parseInt(value));
29                return marque;
29            } catch (Exception e) {
29        }
29    }
29}
```

ProduitConverter

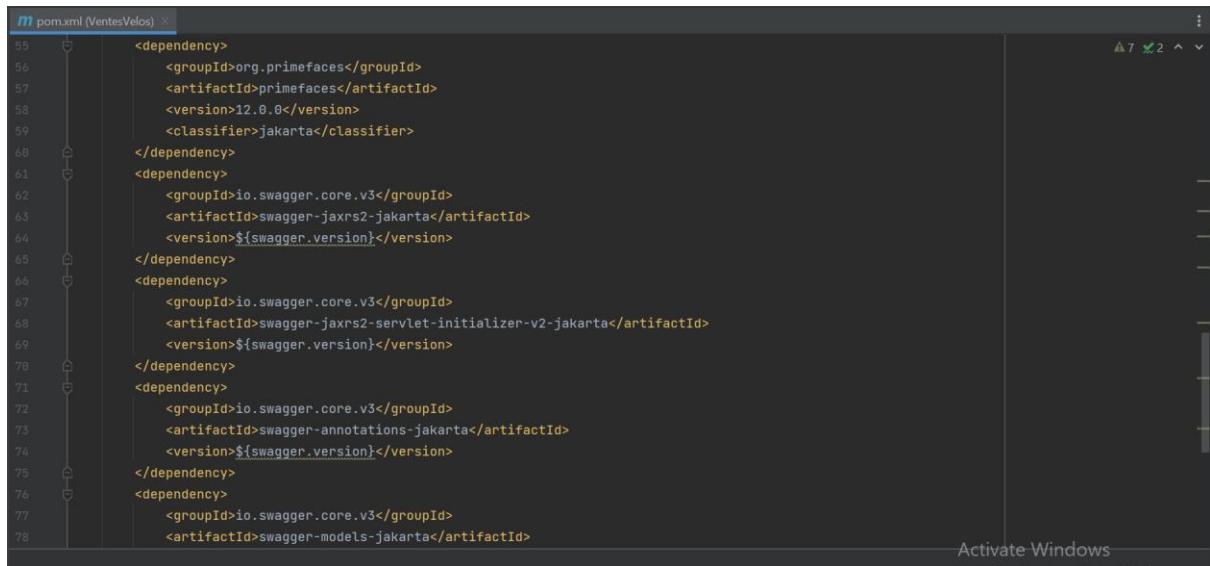


```
ProductConverter.java
1 package sn.ept.ventesvelos.mbeans.converters;
2
3 import ...
10
11 no usages
12 @FacesConverter(value = "produitConverter", managed = true)
13 public class ProduitConverter implements Converter {
14
15     1 usage
16     @EJB
17     private ProduitFacade produitFacade;
18
19     @Override
20     public String getAsString(FacesContext context, UIComponent component, Object value) {
21         if (value instanceof Produit) {
22             Produit produit = (Produit) value;
23             return String.valueOf(produit.getId());
24         }
25         return null;
26     }
27
28     @Override
29     public Object getAsObject(FacesContext context, UIComponent component, String value) {
30         if (value != null && !value.isEmpty()) {
```

C. Création des pages JSF

Pour chacune de nos entités nous créons les pages JSF pour l'ajout, la modification et la lecture des données.

Pour cela on ajoute Primefaces et un fichier *welcome.xhtml* où on a un template pour nos pages dans un tag *welcome-file-list*



```
pom.xml (VentesVelos)
55     <dependency>
56         <groupId>org.primefaces</groupId>
57         <artifactId>primefaces</artifactId>
58         <version>12.0.0</version>
59         <classifier>jakarta</classifier>
60     </dependency>
61     <dependency>
62         <groupId>io.swagger.core.v3</groupId>
63         <artifactId>swagger-jaxrs2-jakarta</artifactId>
64         <version>${swagger.version}</version>
65     </dependency>
66     <dependency>
67         <groupId>io.swagger.core.v3</groupId>
68         <artifactId>swagger-jaxrs2-servlet-initializer-v2-jakarta</artifactId>
69         <version>${swagger.version}</version>
70     </dependency>
71     <dependency>
72         <groupId>io.swagger.core.v3</groupId>
73         <artifactId>swagger-annotations-jakarta</artifactId>
74         <version>${swagger.version}</version>
75     </dependency>
76     <dependency>
77         <groupId>io.swagger.core.v3</groupId>
78         <artifactId>swagger-models-jakarta</artifactId>
```

```

<!-- web.xml -->
<servlet>
    <servlet-name>Faces Servlet</servlet-name>
    <servlet-class>jakarta.faces.webapp.FacesServlet</servlet-class>
    <load-on-startup>1</load-on-startup>
</servlet>

<context-param>
    <param-name>javax.faces.STATE_SAVING_METHOD</param-name>
    <param-value>server</param-value>
</context-param>

<servlet-mapping>
    <servlet-name>Faces Servlet</servlet-name>
    <url-pattern>*.xhtml</url-pattern>
</servlet-mapping>

<context-param>
    <param-name>primefaces.THEME</param-name>
    <param-value>saga</param-value>
</context-param>

<welcome-file-list>
    <welcome-file>welcome.xhtml</welcome-file>
</welcome-file-list>

```

Nous utilisons le composant DataTable avec CrudView ce qui nous permet de gérer toutes les opérations CRUD sur une même page.

Nous avons donc les fichiers xhtml qui sont comme tel et donne le rendu que vous verrez plus bas.

Ici on montre l'exemple pour la page catégorie mais le principe reste le même pour toutes les autres pages, vous pourrez voir plus en détails au niveau de la démo et du code.

```

<!-- categorie-pf.xhtml -->
<h:form id="form">
    <p:growl id="messages" showDetail="true"/>

    <p:toolbar style="margin-bottom: 1rem;">
        <p:toolbarGroup>
            <p:commandButton value="Ajouter" icon="pi pi-plus" actionListener="#{categorieMBean.openNew}"
                update=":dialogs:manage-categorie-content" oncomplete="PF('manageCategorieDialog').show()"
                styleClass="ui-button-success" style="margin-right: .5rem">
                <p:resetInput target=":dialogs:manage-categorie-content" />
            </p:commandButton>
            <p:commandButton id="delete-categories-button" value="#{categorieMBean.deleteButtonMessage}"
                icon="pi pi-trash" actionListener="#{categorieMBean.deleteSelectedCategories}"
                styleClass="ui-button-danger" disabled="#{!categorieMBean.hasSelectedCategories()}" update="@this">
                <p:confirm header="Confirmation" message="Supprimer les catégories sélectionnées?">
                    icon="pi pi-exclamation-triangle" />
            </p:commandButton>
        </p:toolbarGroup>
    </p:toolbar>

    <p:dataTable id="dt-categories" widgetVar="dtCategories" var="categorie" value="#{categorieMBean.categories}"
        reflow="true" styleClass="categories-table" selection="#{categorieMBean.selectedCategories}"
        rowKey="#{categorie.id}" paginator="true" rows="10" rowSelectMode="add" paginatorPosition="bottom">
        <f:facet name="header">

```

Lister

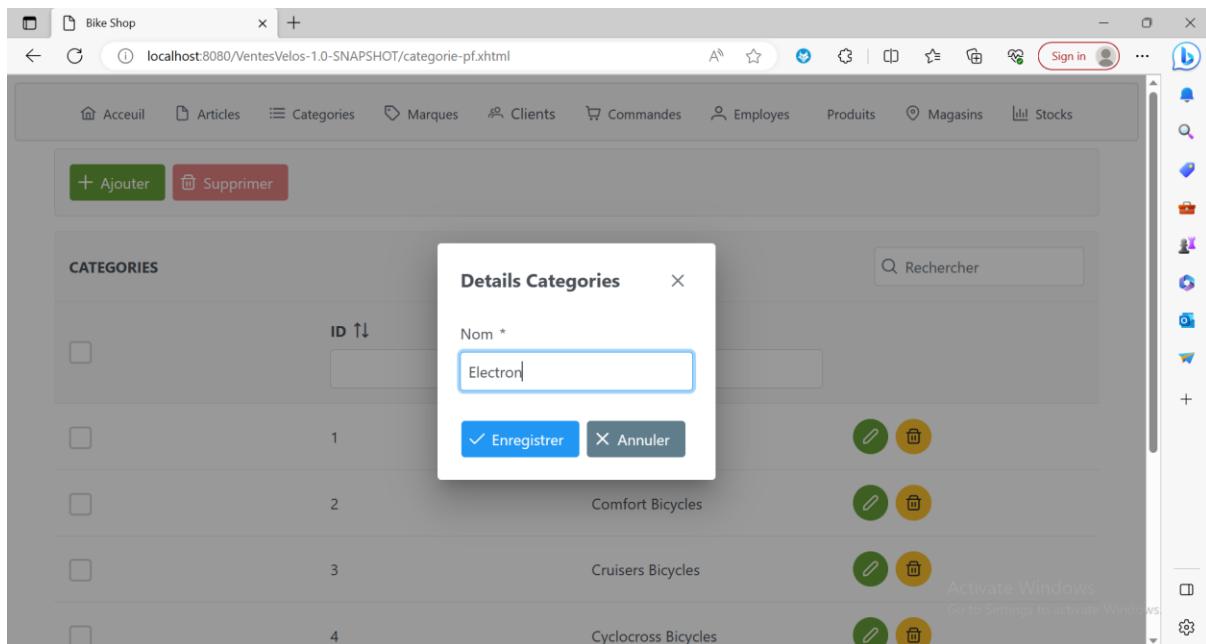
The screenshot shows a web browser window titled "Bike Shop" with the URL "localhost:8080/VentesVelos-1.0-SNAPSHOT/categorie-pf.xhtml". The page displays a table of categories:

ID	Nom	Action
1	Children Bicycles	
2	Comfort Bicycles	
3	Cruisers Bicycles	
4	Cyclocross Bicycles	
5	Electric Bikes	

At the top left, there are buttons for "+ Ajouter" (Add) and "Supprimer" (Delete). A search bar labeled "Rechercher" is at the top right. The sidebar on the right includes icons for file operations, user management, and system settings.

Ajouter

The screenshot shows the same web browser window with an "Ajouter" (Add) dialog box overlaid on the category list. The dialog is titled "Details Categories" and contains a single input field labeled "Nom *" with a placeholder value. At the bottom are two buttons: "Enregistrer" (Register) with a checkmark icon and "Annuler" (Cancel) with a cross icon.



ID	Nom	Actions
1	Children Bicycles	
2	Comfort Bicycles	
3	Cruisers Bicycles	
4	Cyclocross Bicycles	
5	Electric Bikes	
6	Mountain Bikes	
7	Road Bikes	
9	Electron	

Modifier

The screenshot shows a web browser window titled "Bike Shop" with the URL "localhost:8080/VentesVelos-1.0-SNAPSHOT/categorie-pf.xhtml". The main content is a table listing bicycle categories:

		Category	Action
1		Children Bicycles	
2		Comfort Bicycles	
3		Cruisers Bicycles	
4		Electra	
5		Road Bikes	
6		Electron	

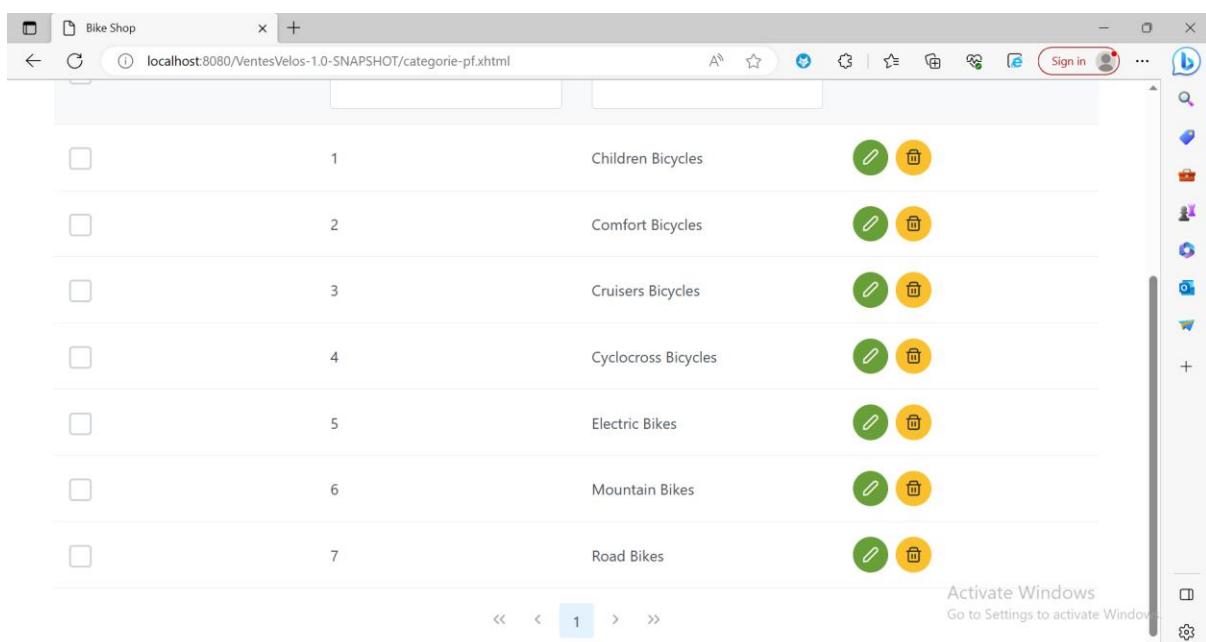
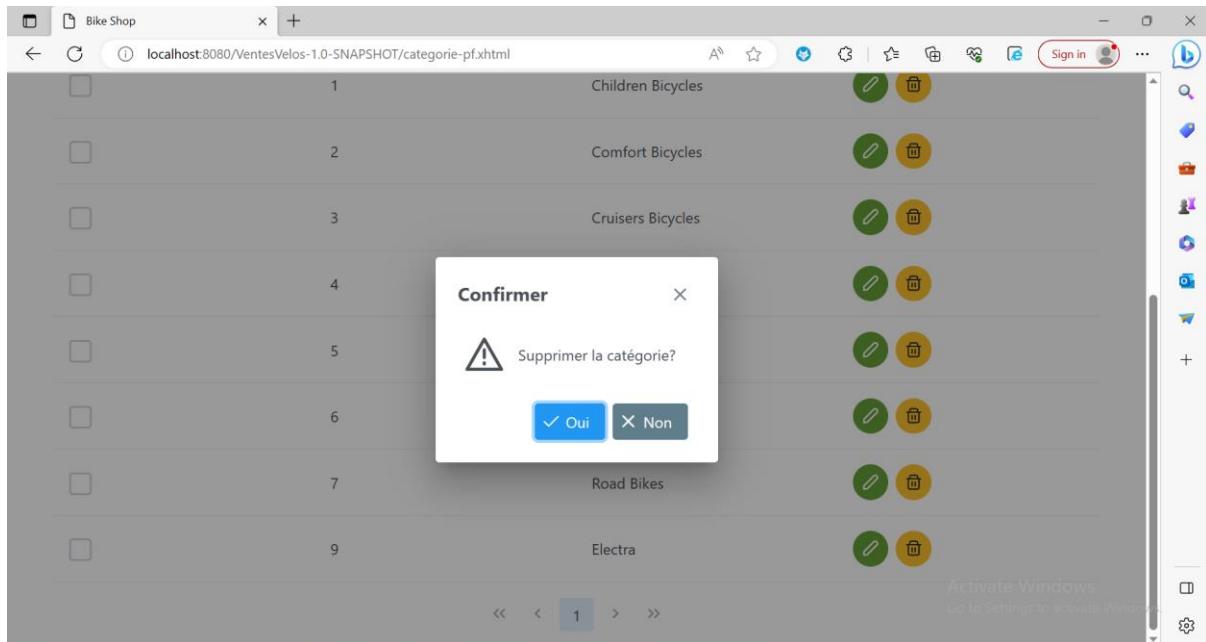
A modal dialog titled "Details Categories" is open, prompting for a category name. The input field contains "Electra", and there are "Enregistrer" and "Annuler" buttons.

The screenshot shows the same web browser window after the new category has been saved. The table now includes the new entry:

		Category	Action
1		Children Bicycles	
2		Comfort Bicycles	
3		Cruisers Bicycles	
4		Cyclocross Bicycles	
5		Electric Bikes	
6		Mountain Bikes	
7		Road Bikes	
9		Electra	

A blue success message box is displayed in the center of the screen, stating "Catégorie Modifiée".

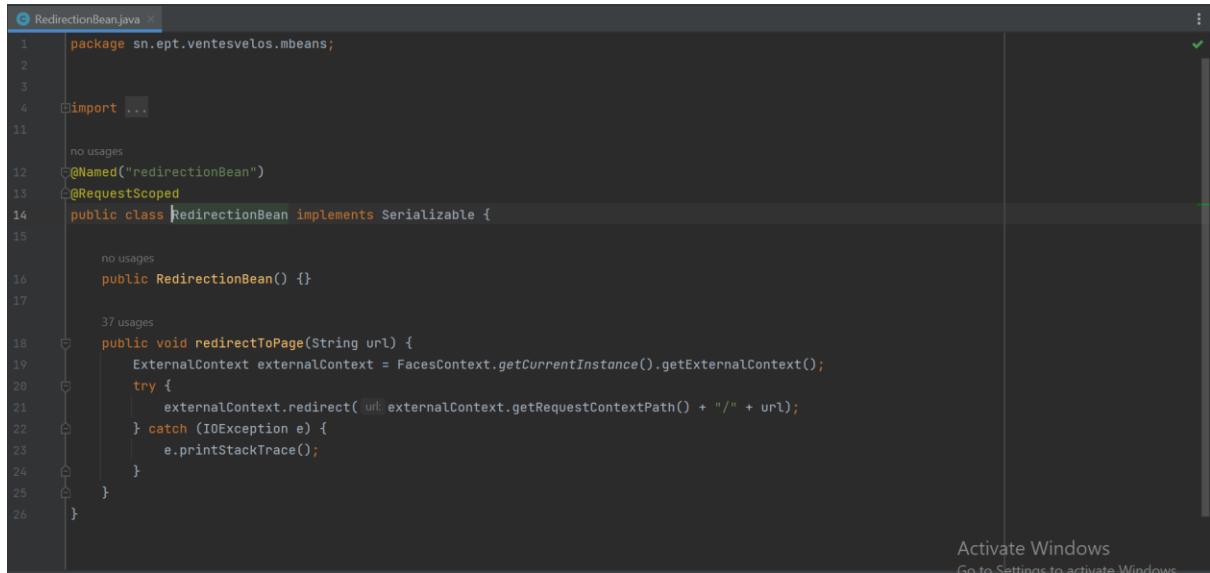
Supprimer



2. Créer un menu

Etant donné que le menu doit permettre la redirection au niveau de l'application, nous créons d'abord un bean pour la redirection. On utilise `@Named` et `@RequestScopped` pour marquer le Mbean comme un composant JSF nommé et définir l'étendue du Mbean.

On définit la méthode `redirectToPage()` pour gérer la redirection vers les autres pages.



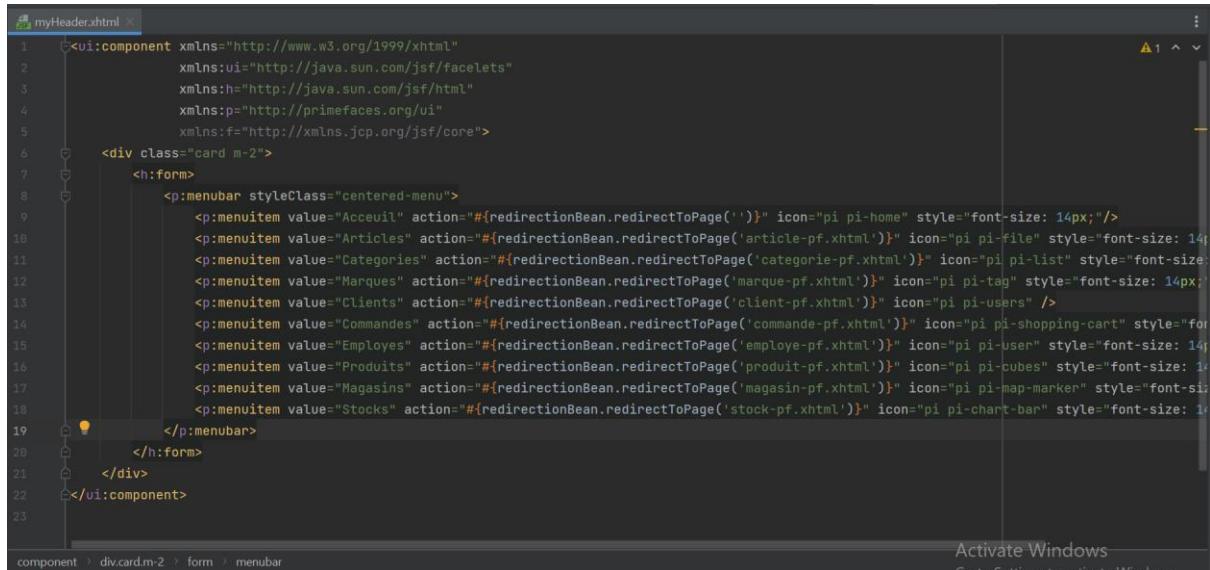
```

1 package sn.ept.ventesvelos.mbeans;
2
3
4 import ...
5
6 no usages
7
8 @Named("redirectionBean")
9 @RequestScoped
10 public class RedirectionBean implements Serializable {
11
12     no usages
13     public RedirectionBean() {}
14
15     37 usages
16     public void redirectToPage(String url) {
17         ExternalContext externalContext = FacesContext.getCurrentInstance().getExternalContext();
18         try {
19             externalContext.redirect(url: externalContext.getRequestContextPath() + "/" + url);
20         } catch (IOException e) {
21             e.printStackTrace();
22         }
23     }
24 }
25
26

```

Activate Windows
Go to Settings to activate Windows

On crée le composant pour le menu que l'on va ajouter à notre page d'accueil en ajoutant un insert qui nous permettra d'utiliser le contenu de la page d'accueil comme template et avoir le menu sur toute nos pages.



```

1 <ui:component xmlns="http://www.w3.org/1999/xhtml"
2                 xmlns:ui="http://java.sun.com/jsf/facelets"
3                 xmlns:h="http://java.sun.com/jsf/html"
4                 xmlns:p="http://primefaces.org/ui"
5                 xmlns:f="http://xmlns.jcp.org/jsf/core">
6     <div class="card m-2">
7         <h:form>
8             <p:menubar styleClass="centered-menu">
9                 <p:menuitem value="Accueil" action="#{redirectionBean.redirectToPage('')}" icon="pi pi-home" style="font-size: 14px;"/>
10                <p:menuitem value="Articles" action="#{redirectionBean.redirectToPage('article-pf.xhtml')}" icon="pi pi-file" style="font-size: 14px;"/>
11                <p:menuitem value="Categories" action="#{redirectionBean.redirectToPage('categorie-pf.xhtml')}" icon="pi pi-list" style="font-size: 14px;"/>
12                <p:menuitem value="Marques" action="#{redirectionBean.redirectToPage('marque-pf.xhtml')}" icon="pi pi-tag" style="font-size: 14px;"/>
13                <p:menuitem value="Clients" action="#{redirectionBean.redirectToPage('client-pf.xhtml')}" icon="pi pi-users" />
14                <p:menuitem value="Commandes" action="#{redirectionBean.redirectToPage('commande-pf.xhtml')}" icon="pi pi-shopping-cart" style="font-size: 14px;"/>
15                <p:menuitem value="Employes" action="#{redirectionBean.redirectToPage('employe-pf.xhtml')}" icon="pi pi-user" style="font-size: 14px;"/>
16                <p:menuitem value="Produits" action="#{redirectionBean.redirectToPage('produit-pf.xhtml')}" icon="pi pi-cubes" style="font-size: 14px;"/>
17                <p:menuitem value="Magasins" action="#{redirectionBean.redirectToPage('magasin-pf.xhtml')}" icon="pi pi-map-marker" style="font-size: 14px;"/>
18                <p:menuitem value="Stocks" action="#{redirectionBean.redirectToPage('stock-pf.xhtml')}" icon="pi pi-chart-bar" style="font-size: 14px;"/>
19            </p:menubar>
20        </h:form>
21    </div>
22 </ui:component>
23

```

Activate Windows
Go to Settings to activate Windows

```
File welcome.xhtml Line 1 of 42
```

```
20     }
21   </style>
22 </h:head>
23
24 <h:body>
25   <ui:include src="myHeader.xhtml"/>
26   <div class="container mt-2">
27     <ui:insert name="content">
28       <div class="d-flex flex-column justify-content-center align-items-center">
29         <h1 class="text-center">Bienvenue au service de ventes de vélos</h1>
30         <h3>Ici vous pourrez créer, supprimer, modifier et voir tous les objets de ce service</h3>
31         <div class="image-container">
32           <h:graphicImage value="https://www.sefiles.net/merchant/5611/images/site/PXL_20221204_204356273-slimC.jpg?t=1674157163799" styleClass="img-fluid" />
33         </div>
34       </div>
35     </ui:insert>
36   </div>
37 </h:body>
38
39 </html>
```

Nous pouvons donc voir le menu qui permet de gérer la redirection

Bienvenue au service de ventes de vélos

Ici vous pourrez créer, supprimer, modifier et voir tous les objets de ce service



Activate Windows
Go to Settings to activate Windows

ARTICLES COMMANDES

	Ligne ↑	Numero ↑	Produit ↑	Remise ↑	Quantite ↑	Prix Depart ↑	
<input type="checkbox"/>	1	1	Electra Townie Original 7D EQ - Women's - 2016	0	1	600	
<input type="checkbox"/>	2	1	Trek Remedy 29 Carbon Frameset - 2016	0	2	1800	 Activate Windows Go to Settings to activate Windows

MARQUES

	ID ↑	Nom ↑	
<input type="checkbox"/>	1	Electra	
<input type="checkbox"/>	2	Haro	
<input type="checkbox"/>	3	Heller	
<input type="checkbox"/>	4	Pure Cycles	 Activate Windows Go to Settings to activate Windows

3. Filtrage, triage et pagination

Pour ajouter le filtrage, le triage et la pagination ; nous utilisons les attributs de DataTable.

Pour la pagination on utilise les attributs *paginator*, *rows*, *paginatorPosition*.

Pour le filtrage et le triage on ajoute les attributs *sortBy* et *filterBy* au niveau des composants column.

```
product-pf.xhtml
30
31 <p:dataTable id="dt-produits" widgetVar="dtProduits" var="produit" value="#{produitMBean.produits}"
32     reflow="true" styleClass="produits-table" selection="#{produitMBean.selectedProduits}"
33     rowKey="#{produit.id}" paginator="true" rows="10" rowSelectMode="add" paginatorPosition="bottom">
34     <f:facet name="header">
35         <div style="display: flex; justify-content: space-between; align-items: center;">
36             <span style="font-weight: bold">PRODUITS</span>
37             <span class="filter-container ui-input-icon-left"> <i class="pi pi-search"></i>
38                 <p:inputText id="globalFilter" onkeyup="PF('dtProduits').filter()" placeholder="Rechercher" />
39             </span>
40         </div>
41     </f:facet>
42
43     <p:ajax event="rowSelect" update=":form:delete-produits-button" />
44     <p:ajax event="rowUnselect" update=":form:delete-produits-button" />
45     <p:ajax event="rowSelectCheckbox" update=":form:delete-produits-button" />
46     <p:ajax event="rowUnselectCheckbox" update=":form:delete-produits-button" />
47     <p:ajax event="toggleSelect" update=":form:delete-produits-button" />
48
49     <p:column selectionMode="multiple" exportable="false"></p:column>
50
51     <p:column headerText="ID" sortBy="#{produit.id}" filterBy="#{produit.id}">
52         <h:outputText value="#{produit.id}" />
53     </p:column>
54
55     <p:column headerText="Nom" sortBy="#{produit.nom}" filterBy="#{produit.nom}">
56         <h:outputText value="#{produit.nom}" />
57     </p:column>
58     <p:column headerText="Annee" sortBy="#{produit.anneeModel}" filterBy="#{produit.anneeModel}">
59         <h:outputText value="#{produit.anneeModel}" />
60     </p:column>
61     <p:column headerText="Prix" sortBy="#{produit.prixDepart}" filterBy="#{produit.prixDepart}">
62         <h:outputText value="#{produit.prixDepart}" />
63     </p:column>
64     <p:column headerText="Categorie" sortBy="#{produit.categorieId.nom}" filterBy="#{produit.categorieId.nom}">
65         <h:outputText value="#{produit.categorieId.nom}" />
66     </p:column>
67     <p:column headerText="Marque" sortBy="#{produit.marqueId.nom}" filterBy="#{produit.marqueId.nom}">
68         <h:outputText value="#{produit.marqueId.nom}" />
69     </p:column>
70     <p:column exportable="false">
71         <p:commandButton icon="pi pi-pencil" update=":dialogs:manage-produit-content"
72             onComplete="PF('manageProduitDialog').show()"
73             styleClass="edit-button rounded-button ui-button-success" process="@this" style="...">
74             <f:setPropertyActionListener value="#{produit}" target="#{produitMBean.selectedProduit}" />
75         </p:commandButton>
76     </p:column>
77
78 </p:dataTable>
```

Activate Windows
Go to Settings to activate Windows

```
product-pf.xhtml
50
51 <p:column headerText="ID" sortBy="#{produit.id}" filterBy="#{produit.id}">
52     <h:outputText value="#{produit.id}" />
53 </p:column>
54 <p:column headerText="Nom" sortBy="#{produit.nom}" filterBy="#{produit.nom}">
55     <h:outputText value="#{produit.nom}" />
56 </p:column>
57 <p:column headerText="Annee" sortBy="#{produit.anneeModel}" filterBy="#{produit.anneeModel}">
58     <h:outputText value="#{produit.anneeModel}" />
59 </p:column>
60 <p:column headerText="Prix" sortBy="#{produit.prixDepart}" filterBy="#{produit.prixDepart}">
61     <h:outputText value="#{produit.prixDepart}" />
62 </p:column>
63 <p:column headerText="Categorie" sortBy="#{produit.categorieId.nom}" filterBy="#{produit.categorieId.nom}">
64     <h:outputText value="#{produit.categorieId.nom}" />
65 </p:column>
66 <p:column headerText="Marque" sortBy="#{produit.marqueId.nom}" filterBy="#{produit.marqueId.nom}">
67     <h:outputText value="#{produit.marqueId.nom}" />
68 </p:column>
69 <p:column exportable="false">
70     <p:commandButton icon="pi pi-pencil" update=":dialogs:manage-produit-content"
71         onComplete="PF('manageProduitDialog').show()"
72         styleClass="edit-button rounded-button ui-button-success" process="@this" style="...">
73         <f:setPropertyActionListener value="#{produit}" target="#{produitMBean.selectedProduit}" />
74     </p:commandButton>
75 </p:column>
76
77 </p:dataTable>
```

Activate Windows
Go to Settings to activate Windows

Triage

Bike Shop

localhost:8080/VentesVélos-1.0-SNAPSHOT/produit-pf.xhtml

PRODUITS

ID ↓ Nom ↑ Année ↑ Prix ↑ Categorie ↑ Marque ↑

Trek Checkpoint ALR Frameset - 2019 2019 3200 Road Bikes Trek

Trek Checkpoint SL 6 - 2019 2019 3800 Road Bikes Trek

Trek Checkpoint SL 5 Women's - 2019 2019 2800 Road Bikes Trek

Trek Checkpoint

Activate Windows Go to Settings to activate Windows

Bike Shop

localhost:8080/VentesVélos-1.0-SNAPSHOT/produit-pf.xhtml

PRODUITS

ID ↑ Nom ↑ Année ↑ Prix ↑ Categorie ↑ Marque ↑

1 Trek 820 - 2016 2016 380 Mountain Bikes Trek

2 Ritchey Timberwolf Frameset - 2016 2016 750 Mountain Bikes Ritchey

3 Surly Wednesday Frameset - 2016 2016 1000 Mountain Bikes Surly

4 Trek Fuel EX 8 29 - 2016 2016 2900 Mountain Bikes Trek

Activate Windows Go to Settings to activate Windows

Filtrage

Bike Shop

localhost:8080/VentesVelos-1.0-SNAPSHOT/produit-pf.xhtml

Sign in

Acceuil Articles Categories Marques Clients Commandes Employes Produits Magasins Stocks

+ Ajouter Supprimer

PRODUITS

ID ↑ Nom ↑ Année ↑ Prix ↑ Catégorie ↑ Marque ↑

	ID	Nom	Année	Prix	Catégorie	Marque	
<input type="checkbox"/>	48	Trek Emonda S 4 - 2017	2017	1500	Road Bikes	Trek	
<input type="checkbox"/>	49	Trek Domane SL 6 - 2017	2017	3500	Road Bikes	Trek	
<input type="checkbox"/>	50	Trek Silque SLR 7 Women's - 2017	2017	6000	Road Bikes	Trek	Activate Windows Go to Settings to activate Windows

PRODUITS

ID ↑ Nom ↑ Année ↑ Prix ↑ Catégorie ↑ Marque ↑

	ID	Nom	Année	Prix	Catégorie	Marque	
<input type="checkbox"/>	3	Surly Wednesday Frameset - 2016	2016	1000	Mountain Bikes	Surly	
<input type="checkbox"/>	6	Surly Ice Cream Truck Frameset - 2016	2016	470	Mountain Bikes	Surly	Activate Windows Go to Settings to activate Windows

Pagination

			Frame - 2016				
	6	Surly Ice Cream Truck Frameset - 2016	2016	470	Mountain Bikes	Surly	
	7	Trek Slash 8 27.5 - 2016	2016	4000	Mountain Bikes	Trek	
	8	Trek Remedy 29 Carbon Frameset - 2016	2016	1800	Mountain Bikes	Trek	
	9	Trek Conduit+ - 2016	2016	3000	Electric Bikes	Trek	
	10	Surly Straggler - 2016	2016	1549	Cyclocross Bicycles	Surly	

Activate Windows
Go to Settings to activate Windows

4. Recherche avancée

Pour effectuer la recherche avancée on utilise les filtres globaux sur le datafram au niveau de la barre de recherche

```

22 <p:commandButton id="delete-produits-button" value="#{produitMBean.deleteButtonMessage}"
23   icon="pi pi-trash" actionListener="#{produitMBean.deleteSelectedProduits}"
24   styleClass="ui-button-danger" disabled="#{!produitMBean.hasSelectedProduits()}" update="@all"
25   <p:confirm header="Confirmation" message="Supprimer les marques sélectionnées?"
26   icon="pi pi-exclamation-triangle" />
27 </p:commandButton>
28 </p:toolbarGroup>
29 </p:toolbar>
30
31 <p:dataTable id="dt-produits" widgetVar="dtProduits" var="produit" value="#{produitMBean.produits}"
32   reflow="true" styleClass="produits-table" selection="#{produitMBean.selectedProduits}"
33   rowKey="#{produit.id}" paginator="true" rows="10" rowSelectMode="add" paginatorPosition="bottom">
34   <f:facet name="header">
35     <div style="...">
36       <span style="...>PRODUITS</span>
37       <span class="filter-container ui-input-icon-left"> <i class="pi pi-search"></i>
38       <p:inputText id="globalFilter" onkeyup="PF('dtProduits').filter()" placeholder="Rechercher" />
39     </div>
40   </f:facet>
41
42   <p:ajax event="rowSelect" update=".form:delete-produits-button" />
43   <p:ajax event="rowUnselect" update=".form:delete-produits-button" />
44   <p:ajax event="rowSelectCheckbox" update=".form:delete-produits-button" />

```

Activate Windows
Go to Settings to activate Windows

Recherche de la marque Surly

Bike Shop

localhost:8080/VentesVelos-1.0-SNAPSHOT/produit-pf.xhtml

PRODUITS

Surly

<input type="checkbox"/>	ID ↑↓	Nom ↑↓	Année ↑↓	Prix ↑↓	Categorie ↑↓	Marque ↑↓	
<input type="checkbox"/>	3	Surly Wednesday Frameset - 2016	2016	1000	Mountain Bikes	Surly	
<input type="checkbox"/>	6	Surly Ice Cream Truck Frameset - 2016	2016	470	Mountain Bikes	Surly	
<input type="checkbox"/>	10	Surly Straggler - 2016	2016	1549	Cyclocross Bicycles	Surly	
<input type="checkbox"/>	11	Surly Straggler 650b - 2016	2016	1681	Cyclocross Bicycles	Surly	

Activate Windows
Go to Settings to activate Windows

Recherche de l'année 2017

Bike Shop

localhost:8080/VentesVelos-1.0-SNAPSHOT/produit-pf.xhtml

PRODUITS

2017

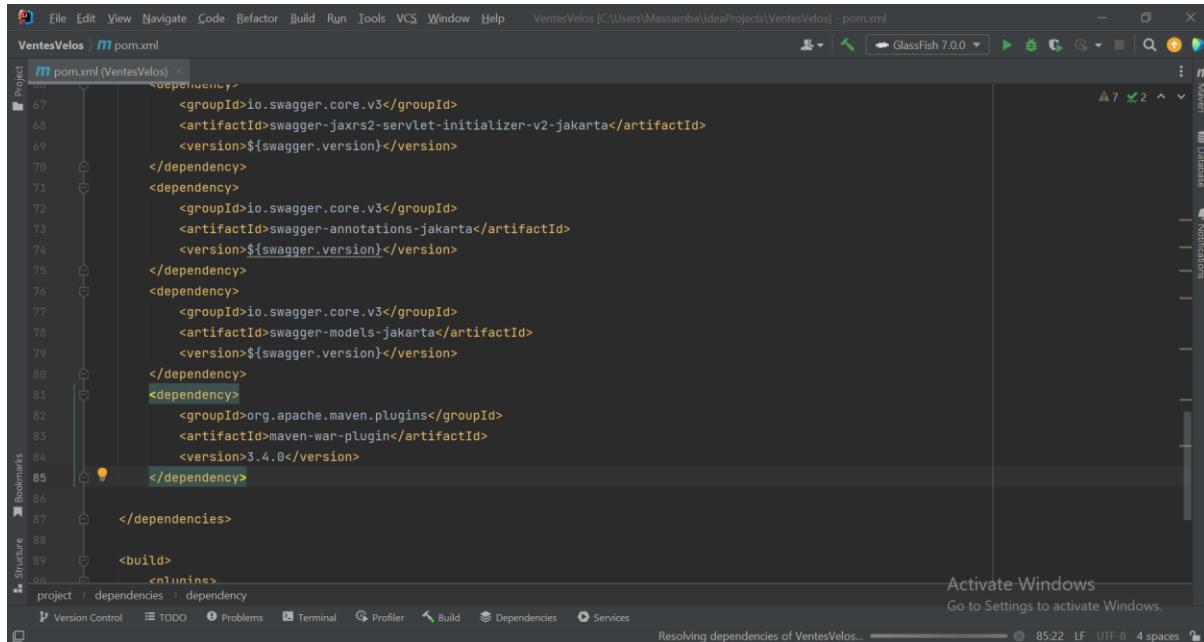
<input type="checkbox"/>	ID ↑↓	Nom ↑↓	Année ↑↓	Prix ↑↓	Categorie ↑↓	Marque ↑↓	
<input type="checkbox"/>	27	Surly Big Dummy Frameset - 2017	2017	1000	Mountain Bikes	Surly	
<input type="checkbox"/>	28	Surly Karate Monkey 27.5+ Frameset - 2017	2017	2500	Mountain Bikes	Surly	
<input type="checkbox"/>	29	Trek X-Caliber 8 - 2017	2017	1000	Mountain Bikes	Trek	
<input type="checkbox"/>	30	Surly Ice Cream Truck Frameset -	2017	1000	Mountain Bikes	Surly	

Activate Windows
Go to Settings to activate Windows

V. Déploiement dans un environnement de production

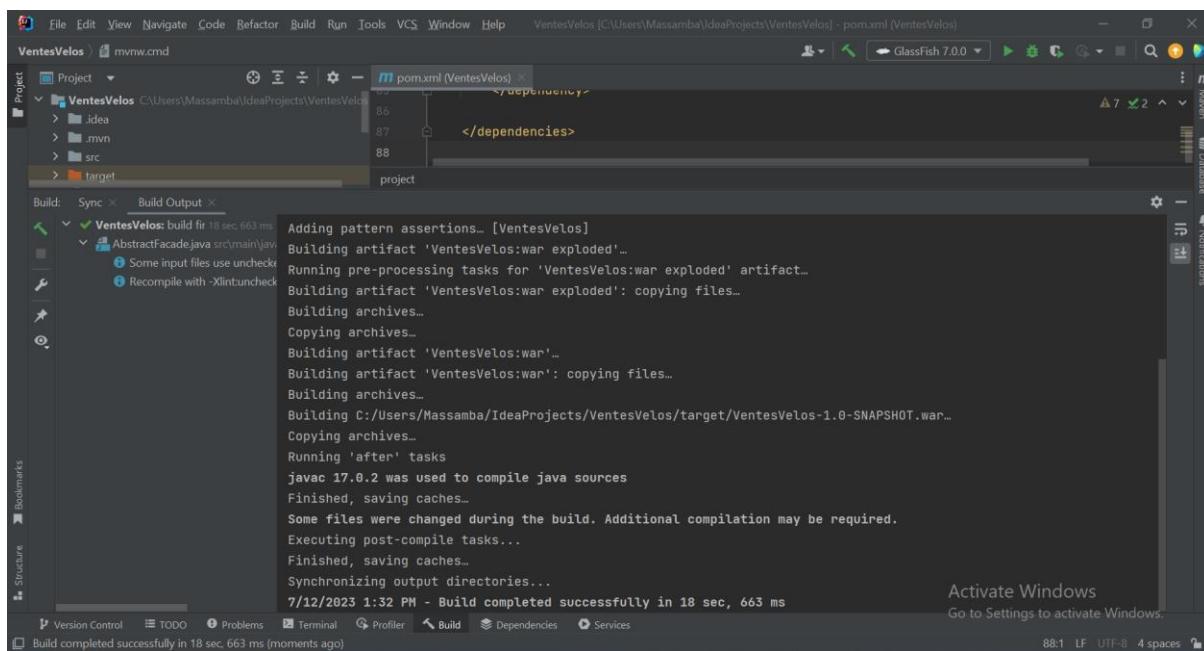
1. Installation de l'environnement de production

Pour déployer notre application, il nous faudra d'abord créer un artefact à en packageant l'application. Nous allons donc commencer par installer la dépendance *maven-war-plugin*.



```
<dependency>
    <groupId>io.swagger.core.v3</groupId>
    <artifactId>swagger-jaxrs2-servlet-initializer-v2-jakarta</artifactId>
    <version>${swagger.version}</version>
</dependency>
<dependency>
    <groupId>io.swagger.core.v3</groupId>
    <artifactId>swagger-annotations-jakarta</artifactId>
    <version>${swagger.version}</version>
</dependency>
<dependency>
    <groupId>io.swagger.core.v3</groupId>
    <artifactId>swagger-models-jakarta</artifactId>
    <version>${swagger.version}</version>
</dependency>
<dependency>
    <groupId>org.apache.maven.plugins</groupId>
    <artifactId>maven-war-plugin</artifactId>
    <version>3.4.0</version>
</dependency>
</dependencies>
<build>
    <plugins>
```

Nous créons ensuite l'artefact



```
Adding pattern assertions... [VentesVelos]
Building artifact 'VentesVelos:war exploded'...
Running pre-processing tasks for 'VentesVelos:war exploded' artifact...
Building artifact 'VentesVelos:war exploded': copying files...
Building archives...
Copying archives...
Building artifact 'VentesVelos:war'...
Building artifact 'VentesVelos:war': copying files...
Building archives...
Building C:/Users/Massamba/IdeaProjects/VentesVelos/target/VentesVelos-1.0-SNAPSHOT.war...
Copying archives...
Running 'after' tasks
javac 17.0.2 was used to compile java sources
Finished, saving caches...
Some files were changed during the build. Additional compilation may be required.
Executing post-compile tasks...
Finished, saving caches...
Synchronizing output directories...
7/12/2023 1:32 PM - Build completed successfully in 18 sec, 663 ms
```

```
</dependency>
<dependency>
    <groupId>org.apache.maven.plugins</groupId>
    <artifactId>maven-war-plugin</artifactId>
    <version>3.4.0</version>
</dependency>
</dependencies>
<build>
    <plugins>
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-war-plugin</artifactId>
            <version>3.3.2</version>
        </plugin>
    </plugins>
</build>
</project>
```

2. Déployer l'application à la racine du serveur