

Technologie JEE

Rapport TP N°4 : EJB

Réalisé par :


Abdelghani MOUSSAID

Encadré par :

Pr. Noredine GHERABI

1- Installation JBoss :

- a. Dans Eclipse, Accéder au l'onglet « Help/Eclipse Marketplace » puis installer l'outils JBossAS Tools de JBoss.



JBoss Tools 4.14.0.Final

JBoss Tools is an umbrella project for a set of Eclipse plugins that includes support for JBoss and related technologies, such as Hibernate, JBoss AS / WildFly,...


[more info](#)

by [Red Hat, Inc.](#), EPL

[openshift](#) [WildFly](#) [jbosstools](#) [maven](#) [hibernate](#)

★ 1099
🔄 Installs: **1.16M** (10,782 last month)
Install

- b. Dans l'onglet « Window/preferences », ajouter un nouveau serveur JBoss, et spécifier la version de JBoss « dans notre cas 7.1.1 » et le chemin du dossier JBoss (le dossier est disponible dans le pack JBoss)

JBoss
Application
Server 7

Edit Server Runtime Environment

JBoss Runtime

JBoss Application Server 7.1

A JBoss Server runtime references a JBoss installation directory. It can be used to set up classpaths for projects which depend on this runtime, as well as by a "server" which will be able to start and stop instances of JBoss.

Name

Home Directory
 [Download and install runtime...](#)


Runtime JRE

☒ Execution Environment:

☐ Alternate JRE:

Server base directory:

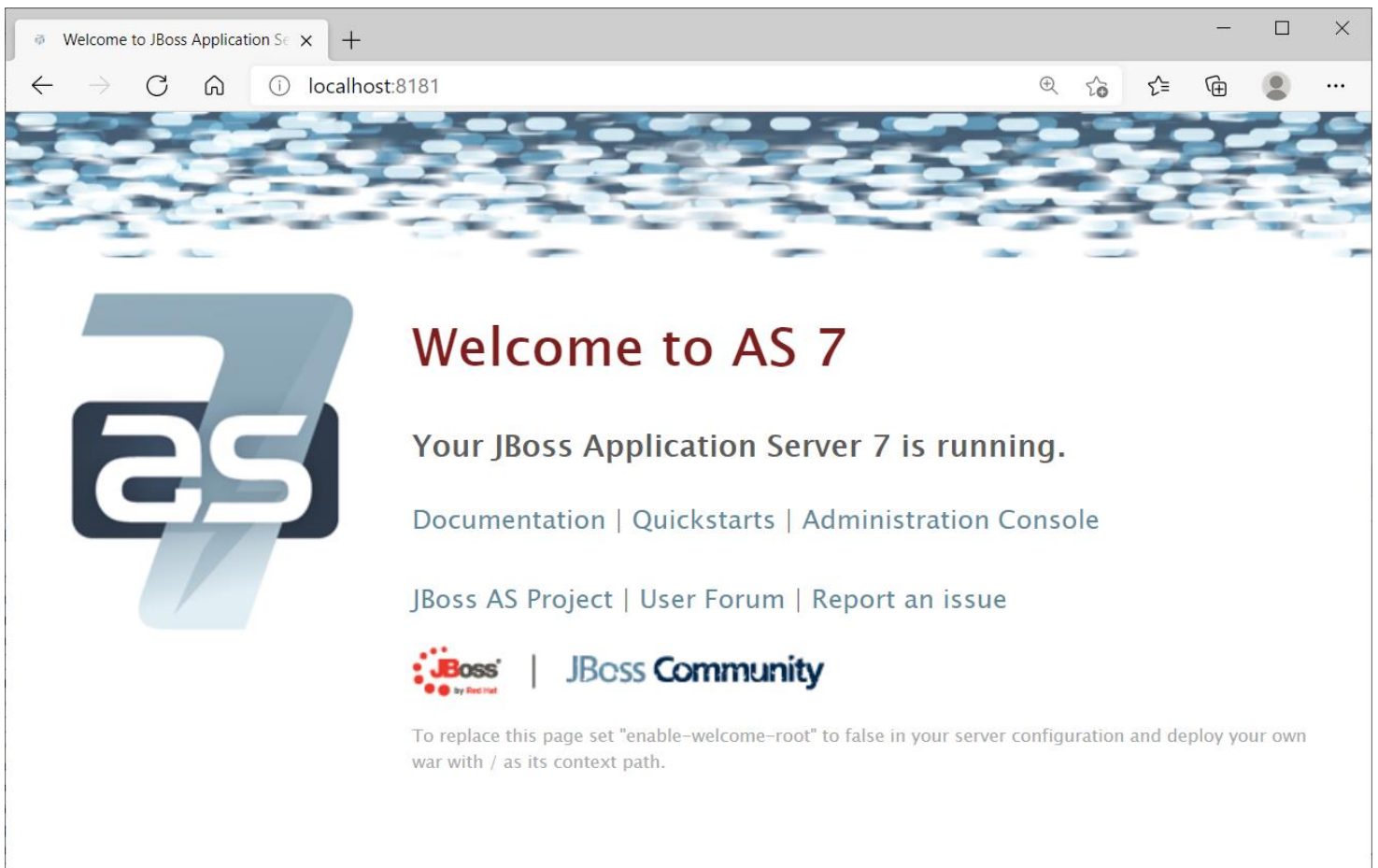
Configuration file:



c. Après l'installation du serveur JBoss , démarrer et vérifier l'état de votre serveur. (port utilisé ?, chemin de démarrage ...)

```
[org.jboss.as.server.deployment.scanner] (MSC service thread 1-3) JBAS015012: Started FileSystemDeploymentService for directory C:\Software\jboss-as-7.1.1.Final\standalone\deployments
[org.jboss.as.remoting] (MSC service thread 1-1) JBAS017100: Listening on localhost/127.0.0.1:4447
[org.jboss.as.remoting] (MSC service thread 1-8) JBAS017100: Listening on localhost/127.0.0.1:9999
[org.jboss.as.connector.subsystems.datasources] (MSC service thread 1-5) JBAS010400: Bound data source [java:jboss/datasources/ExampleDS]
[org.jboss.as] (Controller Boot Thread) JBAS015951: Admin console listening on http://127.0.0.1:9990
[org.jboss.as] (Controller Boot Thread) JBAS015874: JBoss AS 7.1.1.Final "Brontes" started in 4257ms - Started 133 of 208 services (74 services are passive or on-demand)
```

d. En utilisant le navigateur, vérifier que le serveur est bien installé



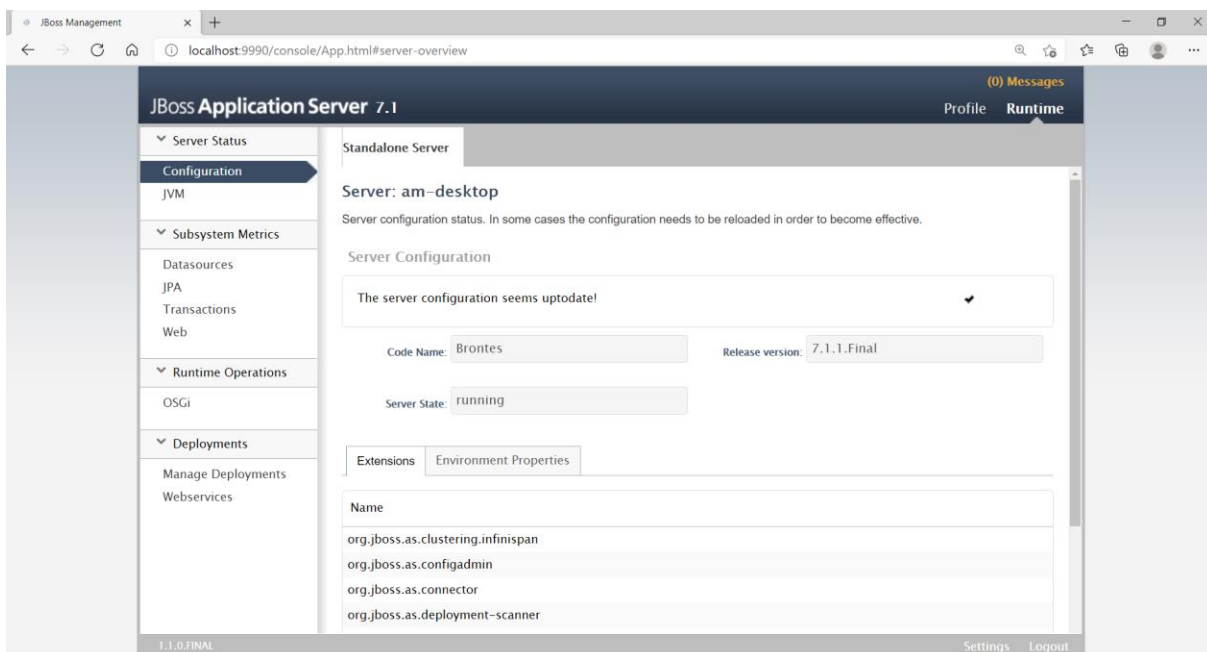
e. A l'aide de l'outil « ADD-USER » et dans la console CMD, ajouter un nouvel utilisateur « admin »

```
C:\WINDOWS\System32\cmd.exe

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

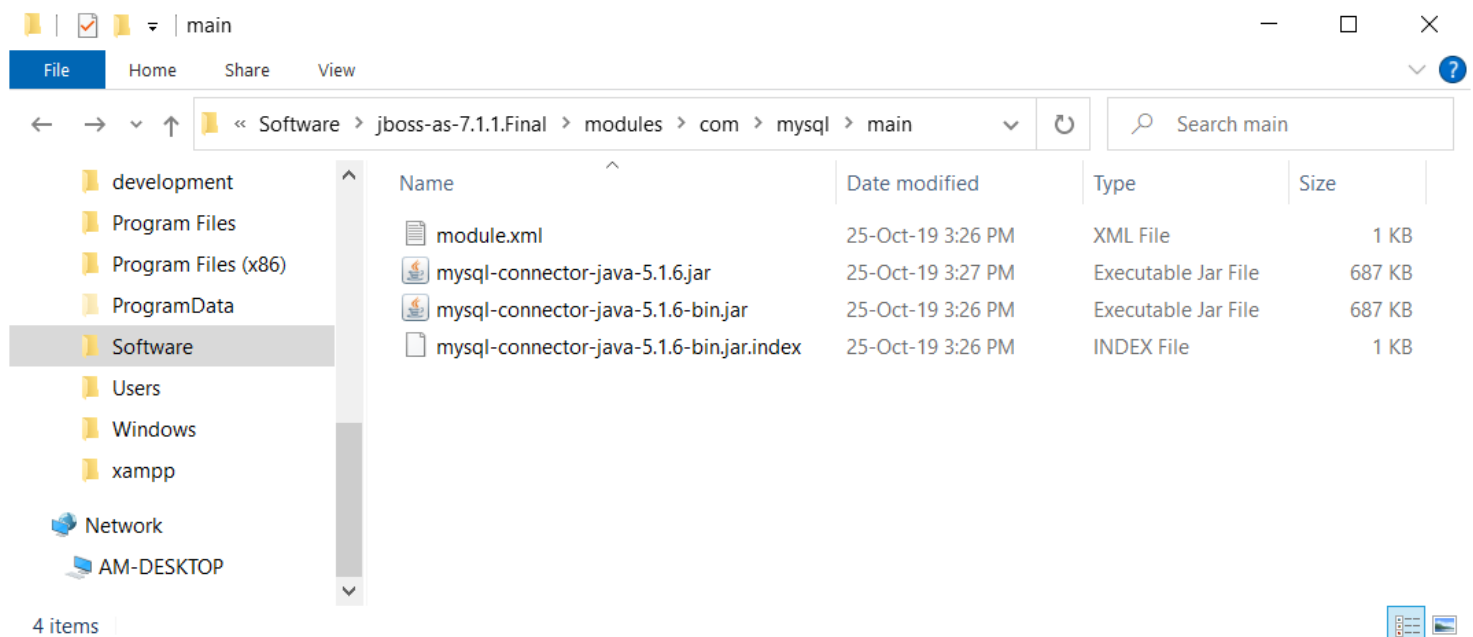
Enter the details of the new user to add.
Realm (ManagementRealm) :
Username : admin
Password :
Re-enter Password :
The username 'admin' is easy to guess
Are you sure you want to add user 'admin' yes/no? yes
About to add user 'admin' for realm 'ManagementRealm'
Is this correct yes/no? yes
Added user 'admin' to file 'C:\Software\jboss-as-7.1.1.Final\standalone\configuration\mgmt-users.properties'
Added user 'admin' to file 'C:\Software\jboss-as-7.1.1.Final\domain\configuration\mgmt-users.properties'
Press any key to continue . . .
```

f. Avec l'utilisateur « admin » accéder à la plate forme de JBoss.



2- Création d'une nouvelle source de données

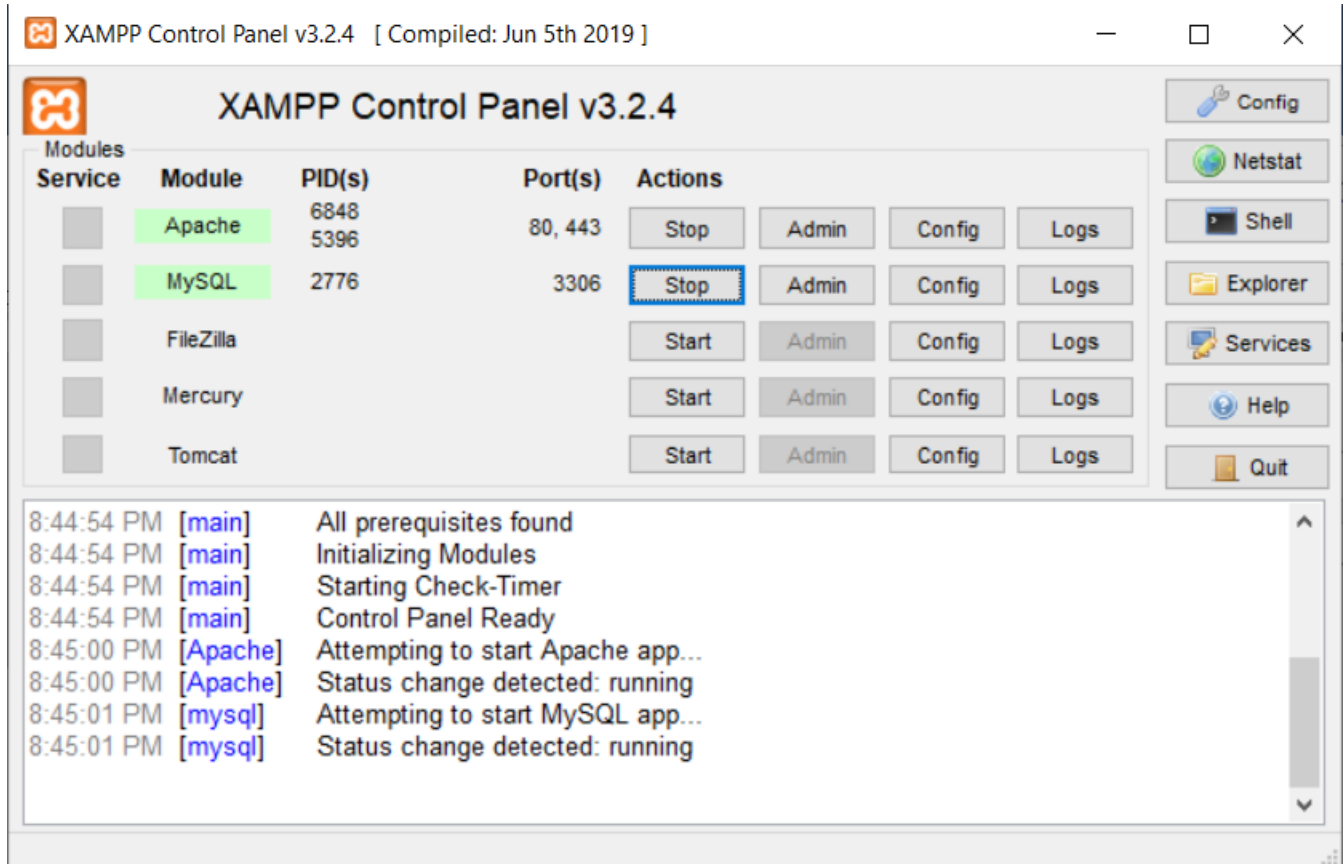
g. Dans le répertoire « modules/com » de JBoss ajouter le répertoire « mysql » contenant le driver Mysql et le fichier de configuration module.xml « le répertoire disponible dans le pack JBoss »



h. Dans le répertoire « /Jboss/standalone/configuration/ » editer le fichier standalone.xml et ajouter le driver de mysql :

```
<driver name="mysql" module="com.mysql">
  <driver-class>com.mysql.jdbc.Driver</driver-class>
  <xa-datasource-class>com.mysql.jdbc.jdbc2.optional.MysqlXADataSource</xa-datasource-class>
</driver>
```

i. Dans votre machine vérifier que XAMPSERVER est installé.



j. En utilisant PHPMyAdmin, créer une base de données vierge appelée « db_Avion».

Databases

Create database ?

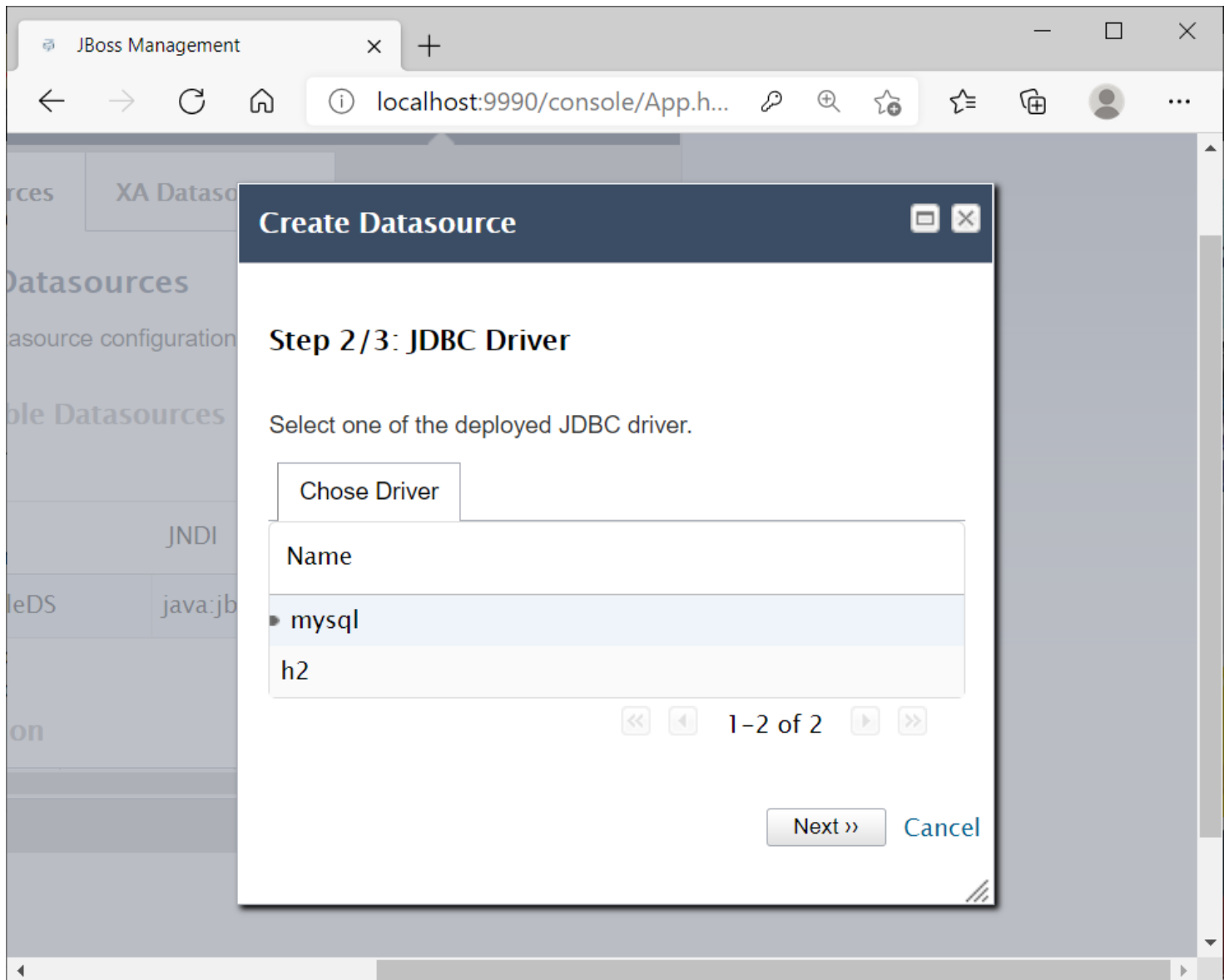
db_Avion

utf8mb4_general_ci

Create

k. Dans l'interface de Jboss, essayer de créer une nouvelle source de données « Datasources » avec ces paramètres :

The screenshot shows a web browser window with the title 'JBoss Management'. The address bar shows 'localhost...'. The main content area displays a 'Create Datasource' dialog box. The dialog has a dark blue header with the title 'Create Datasource' and a close button. Below the header, it says 'Step 1/3: Datasource Attributes'. There are two input fields: 'Name' with the value 'dsAvion' and 'JNDI Name' with the value 'java:/dsAvion'. At the bottom right, there are 'Next >>' and 'Cancel' buttons. The background of the browser shows a sidebar with various navigation options like 'Datasources', 'Configuration', 'JNDI', and 'Settings'.



Connection URL: jdbc:mysql://localhost:3306/db_Avion

I. Activer le datasource créé puis tester la connexion avec la base de données.



3- Création de la couche métier

- Architecture :

```
▼ 📁 ejbModule
  ▼ 📁 com.entities
    > 📄 Avion.java
    > 📄 Horaire.java
    > 📄 Passager.java
    > 📄 Vol.java
```

- Vol.java

```
@Entity
@Table(name="VOL")
public class Vol {

    @Id
    @GeneratedValue(strategy=GenerationType.IDENTITY)
    @Column(name="NUMERO_VOL")
    private int numeroV;

    private String jourSem;
    private String jour;

    @Column(name="PLACES_LIBRES")
    private int placesLibres;

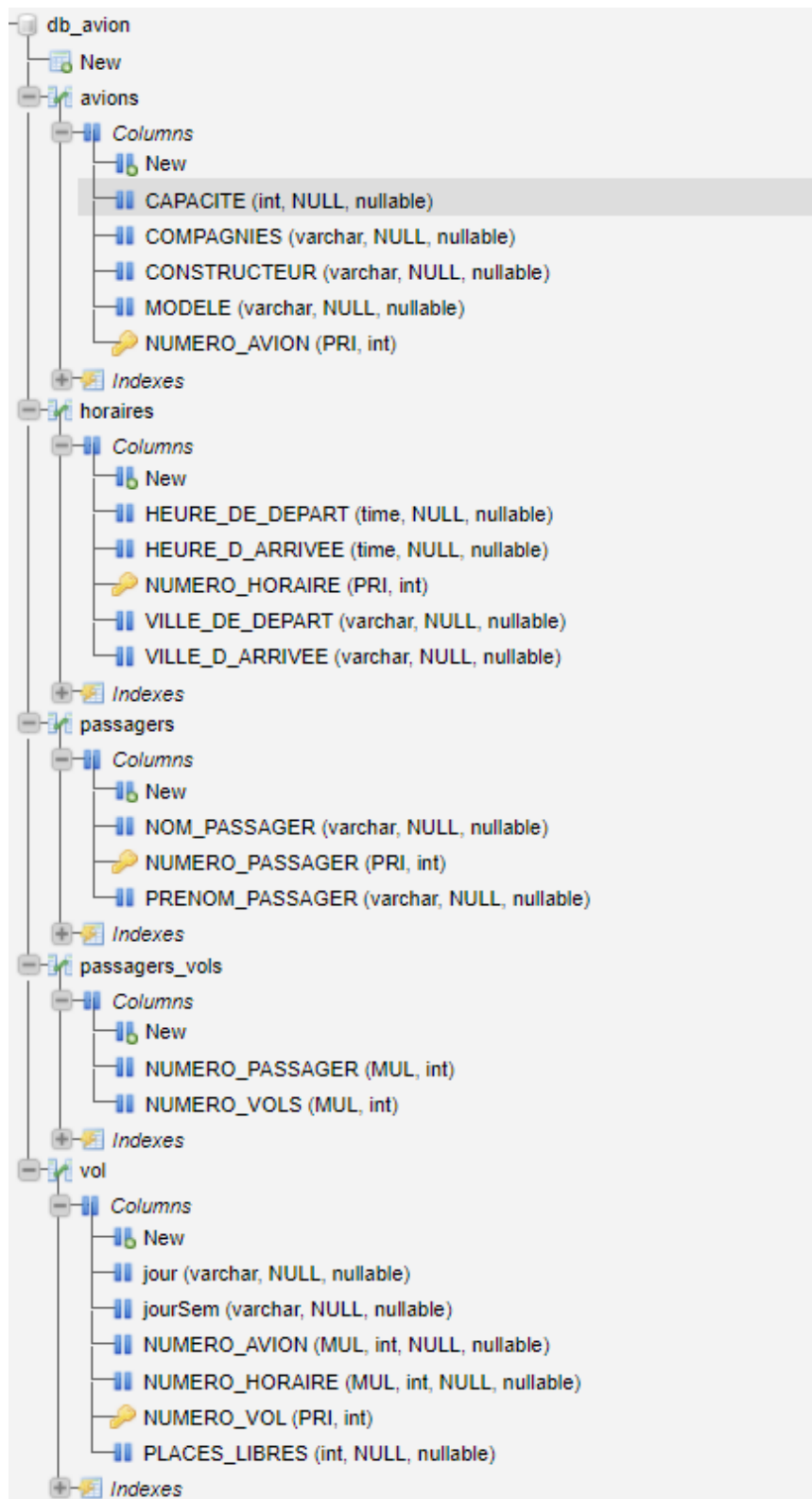
    @ManyToOne
    @JoinColumn(name="NUMERO_HORAIRE")
    private Horaire horaire;

    @ManyToOne
    @JoinColumn(name="NUMERO_AVION")
    private Avion avion;

    @ManyToMany
    @JoinTable(name="PASSAGERS_VOLS",
        joinColumns=@JoinColumn(name="NUMERO_PASSAGER"),inverseJoinColumns=@JoinColumn(name="NUMERO_VOLS"))
    private List<Passager> passagers;

    public int getNumeroV() {
        return numeroV;
    }
    public void setNumeroV(int numeroV) {
        this.numeroV = numeroV;
    }
    public String getJourSem() {
        return jourSem;
    }
    public void setJourSem(String jourSem) {
        this.jourSem = jourSem;
    }
    public String getJour() {
        return jour;
    }
    public void setJour(String jour) {
        this.jour = jour;
    }
    public int getPlacesLibres() {
        return placesLibres;
    }
    public void setPlacesLibres(int placesLibres) {
        this.placesLibres = placesLibres;
    }
    public List<Passager> getPassagers() {
        return passagers;
    }
    public void setPassagers(List<Passager> passagers) {
        this.passagers = passagers;
    }
    public Horaire getHoraire() {
        return horaire;
    }
    public void setHoraire(Horaire horaire) {
        this.horaire = horaire;
    }
    public Avion getAvion() {
        return avion;
    }
    public void setAvion(Avion avion) {
        this.avion = avion;
    }
}
```

- création du schéma dans la base de données «db_Avion »



4- Couche Session :

- Architecture :

```
▼ com.session
  > IRemote_Avion.java
  > IRemote_Horaire.java
  > IRemote_Passager.java
  > IRemote_Vol.java
```

- IRemote_Vol.java :

```
package com.session;

import java.util.List;

import javax.ejb.Remote;

import com.entities.Vol;

@Remote
public interface IRemote_Vol {

    public void addVol(Vol v);
    public void deleteVol(Vol v);
    public void updateVol(Vol v);
    public Vol getVol(int id);
    public List<Vol> getAllVol();
}
```

5- Couche Service :

- Architecture :

```
▼ com.service
  > ServiceAvion.java
  > ServiceHoraire.java
  > ServicePassager.java
  > ServiceVol.java
```

- ServiceVol.java

```
package com.service;

import java.util.List;

import javax.ejb.Stateless;
import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;
import javax.persistence.Query;

import com.entities.Vol;
import com.session.IRemote_Vol;

@Stateless
public class ServiceVol implements IRemote_Vol{

    @PersistenceContext(unitName="UPAvion")
    EntityManager Em;

    @Override
    public void addVol(Vol v) {
        Em.persist(v);
    }

    @Override
    public void deleteVol(Vol v) {
        Em.remove(v);
    }

    @Override
    public void updateVol(Vol v) {
        Em.persist(v);
    }

    @Override
    public Vol getVol(int id) {
        return Em.find( Vol.class, id );
    }

    @Override
    public List<Vol> getAllVol() {
        Query query = Em.createQuery("select v from Vol v");
        return query.getResultList();
    }
}
```

6- Persistence.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<persistence:persistence version="2.0" xmlns:persistence="http://java.sun.com/xml/ns/persistence"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/persistence http://java.sun.com/xml/ns/persistence/persistence_2_0.xsd"
  <persistence:persistence-unit name="UP_AVION" transaction-type="JTA">
    <persistence:jta-data-source>java:/dsAvion</persistence:jta-data-source>
    <persistence:class>com.entities.Avion</persistence:class>
    <persistence:class>com.entities.Horaire</persistence:class>
    <persistence:class>com.entities.Passager</persistence:class>
    <persistence:class>com.entities.Vol</persistence:class>
    <persistence:properties>
      <persistence:property name="hibernate.hbm2ddl.auto" value="create"/>
      <persistence:property name="hibernate.dialect" value="org.hibernate.dialect.MySQLDialect"/>
    </persistence:properties>
  </persistence:persistence-unit>
</persistence:persistence>
```