

Développement d'un site web

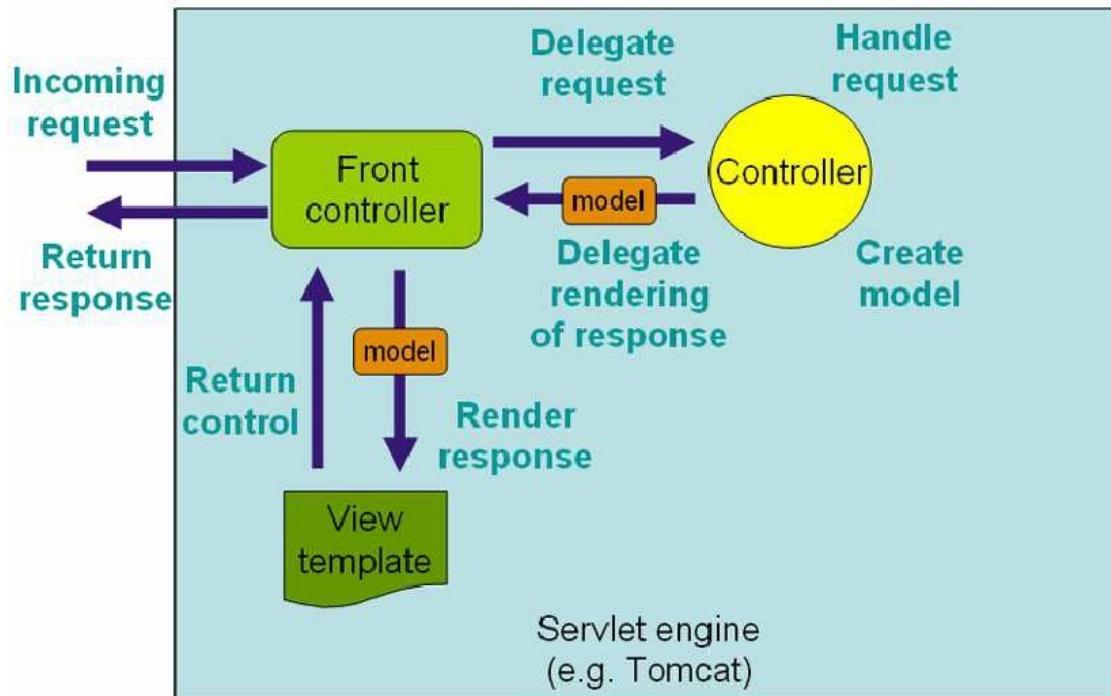
De E-CommerceE-

Basé sur JEE (Part 1)

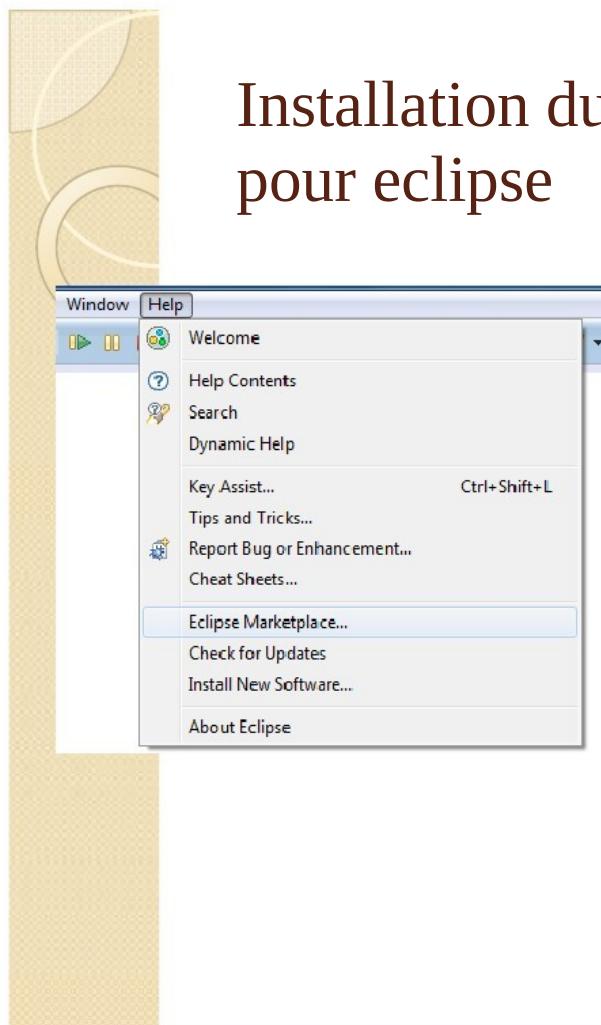
- Eclipse, Maven, JUnitMaven,
- Spring IOC
- Spring MVC
- Spring Security
- JPA, Hibernate

Spring MVC

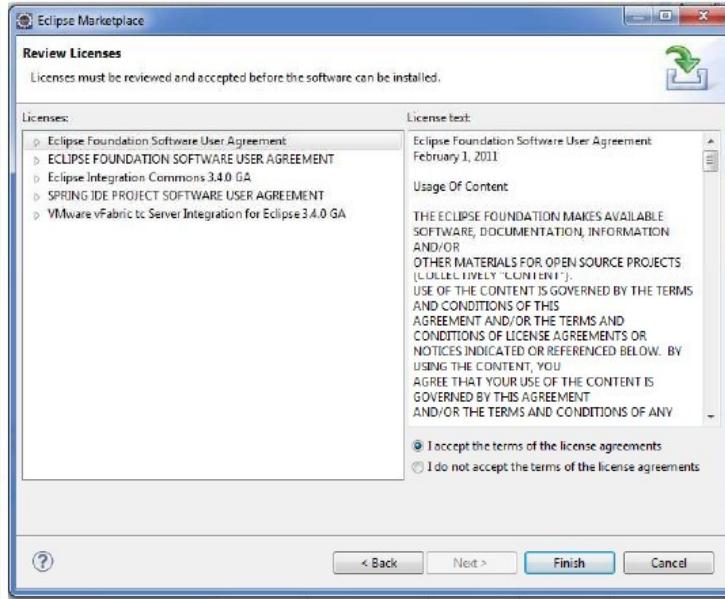
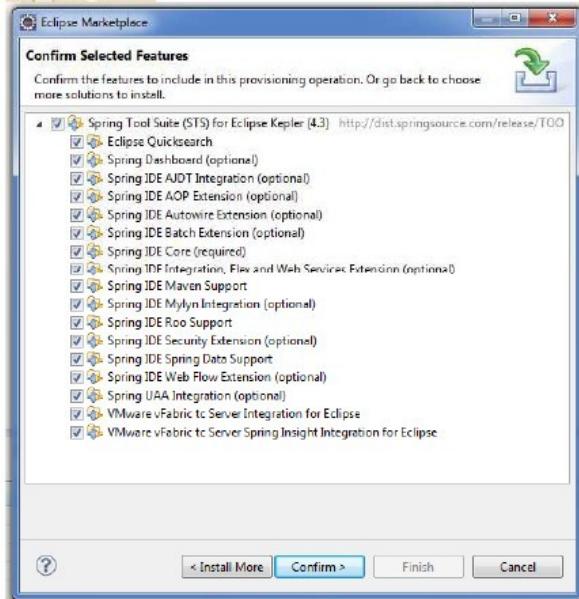
Spring MVC



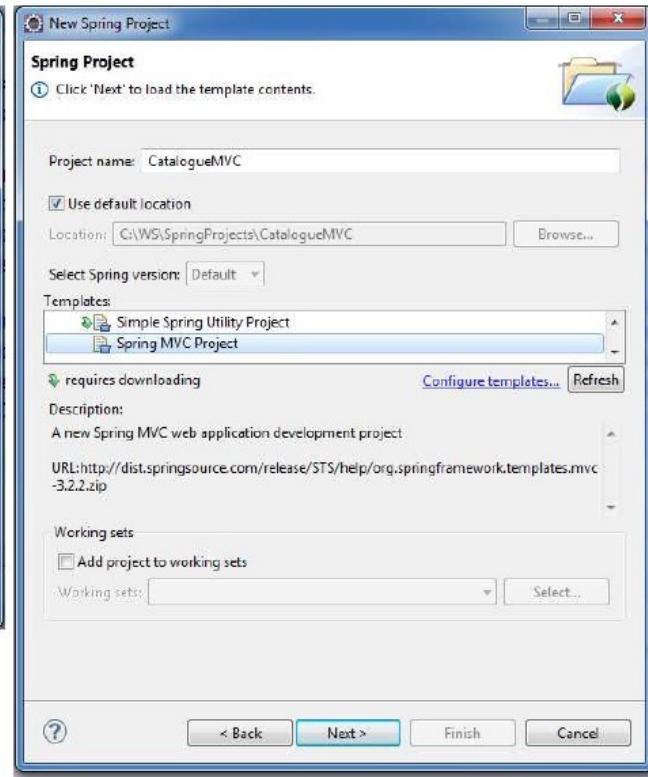
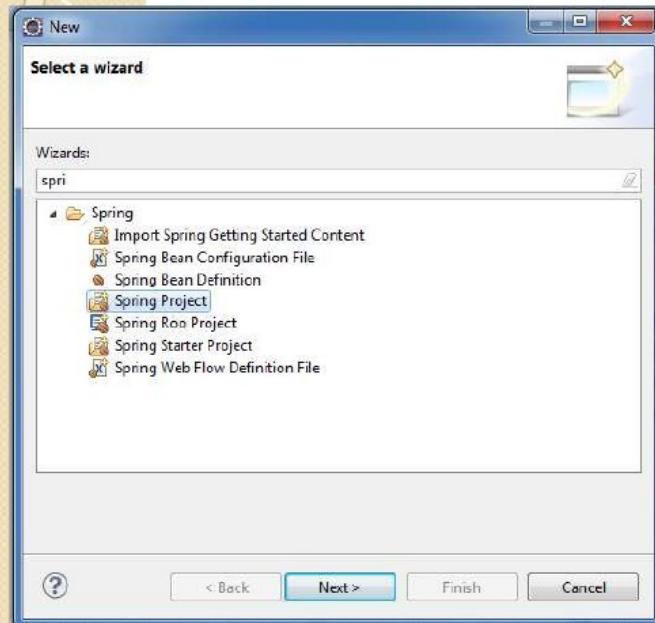
Installation du plugin : spring tools pour eclipse



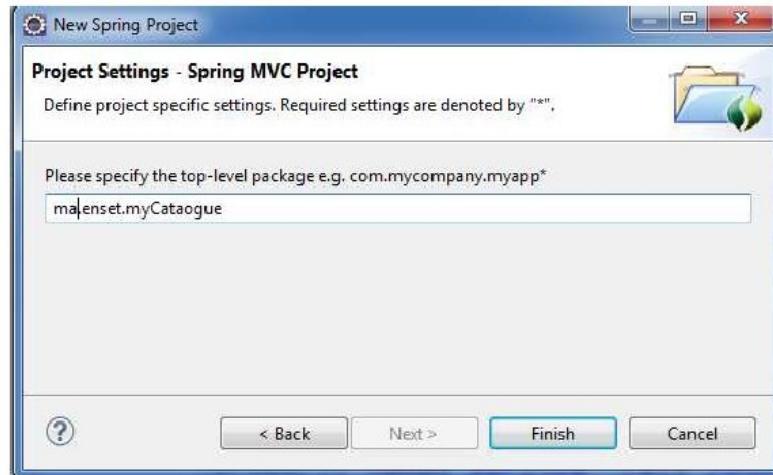
Installation du plugin : spring tools pour eclipse



Création d'un projet Spring

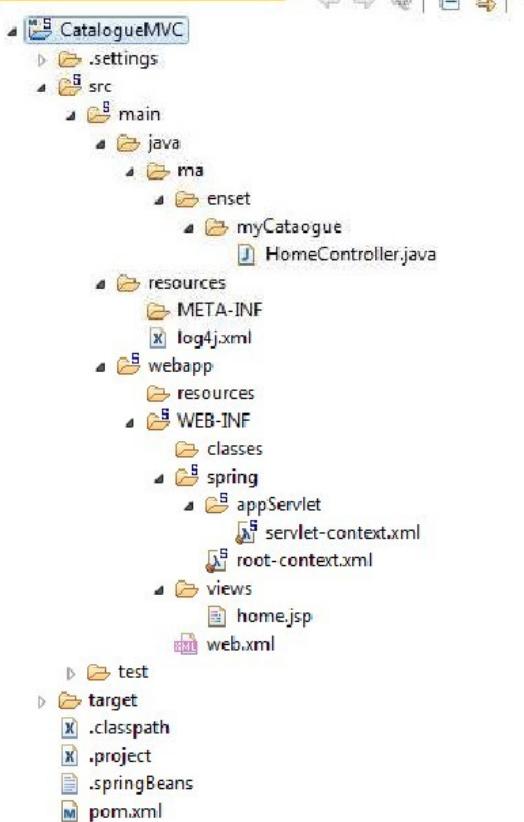


Création d'un projet Spring



Structure du projet

Navigator Explorer



web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="2.5" xmlns="http://java.sun.com/xml/ns/javaee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
    http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd">
  <!-- The definition of the Root Spring Container shared by all Servlets and
      Filters -->
  <context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>/WEB-INF/spring/root-context.xml</param-value>
  </context-param>
  <!-- Creates the Spring Container shared by all Servlets and Filters -->
  <listener>
    <listener-
      class>org.springframework.web.context.ContextLoaderListener</listener-
      class>
  </listener>
```

web.xml

```
<!-- Processes application requests -->
<servlet>
    <servlet-name>appServlet</servlet-name>
    <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-
class>
    <init-param>
        <param-name>contextConfigLocation</param-name>
        <param-value>/WEB-INF/spring/appServlet/servlet-context.xml</param-value>
    </init-param>
    <load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
    <servlet-name>appServlet</servlet-name>
    <url-pattern>/</url-pattern>
</servlet-mapping>
</web-app>
```

>/WEB-INF/spring/root-context.xml/WEB-INF/spring/root-

- Ce fichier est lu par ContextLoaderListener, au démarrage du serveur .
- C'est un fichier dans lequel contexte de l'application sera construit
- ContextLoaderListener représente Spring IOC
- c'est donc un fichier pour l'injection des dépendances
- Pour le moment, il est vide

```
<?xml version="1.0" encoding="UTF-8"?>
<beans
    xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
        http://www.springframework.org/schema/beans/spring-
        beans.xsd">

    <!-- Root Context: defines shared resources visible
        to all other web components -->

</beans>
```

>/WEB-INF/spring/appServlet/**dispatcher-servlet.xml**/WEB-INF/spring/appServlet/**dispatcher-**

- Ce fichier est lu par DispatcherServlet qui représente le contrôleur web de l'application

```
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans    xmlns="http://www.springframework.org/schema/mvc"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:beans="http://www.springframework.org/schema/beans"
    xmlns:context="http://www.springframework.org/schema/context"
    xsi:schemaLocation="http://www.springframework.org/schema/mvc http://www.springframework.org/schema/mvc/spring-mvc.xsd
        http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd
        http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context.xsd">
    <!-- Enables the Spring MVC @Controller programming model -->
    <annotation-driven />
    <!-- Handles HTTP GET requests for /resources/** by efficiently serving up static
        resources in the ${webappRoot}/resources directory -->
    <resources mapping="/resources/**" location="/resources/" />
    <!-- Resolves views selected for rendering by @Controllers to .jsp resources in the
        /WEB-INF/views directory -->
    <beans:bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">
        <beans:property name="prefix" value="/WEB-INF/views/" />
        <beans:property name="suffix" value=".jsp" />
    </beans:bean>
    <context:component-scan base-package="ma.ensembl.myCatalogue" />
</beans:beans>
```

Un exemple de contrôleur Spring MVC

```
package ma.enset.myCataogue;

import java.text.*;import java.util.*;import org.slf4j.*;import
org.springframework.stereotype.Controller;

import org.springframework.ui.Model; import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;

/** Handles requests for the application home page. */

@Controller
public class HomeController {

    private static final Logger logger = LoggerFactory.getLogger(HomeController.class);

    /** Simply selects the home view to render by returning its name. */

@RequestMapping(value = "/", method = RequestMethod.GET)
public String home(Locale locale, Model model) {

    logger.info("Welcome home! The client locale is {}.", locale);
    Date date = new Date();
    DateFormat dateFormat = DateFormat.getDateInstance(DateFormat.LONG, DateFormat.LONG, locale);
    String formattedDate = dateFormat.format(date);
    model.addAttribute("serverTime", formattedDate );
    return "home";
}
}
```

Un exemple de vue JSP

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<%@ page session="false" %>

<html>
    <head>
        <title>Home</title>
    </head>
    <body>
        <h1> Hello world!      </h1>
        <P> The time on the server is ${serverTime}. </P>
    </body>
</html>
```

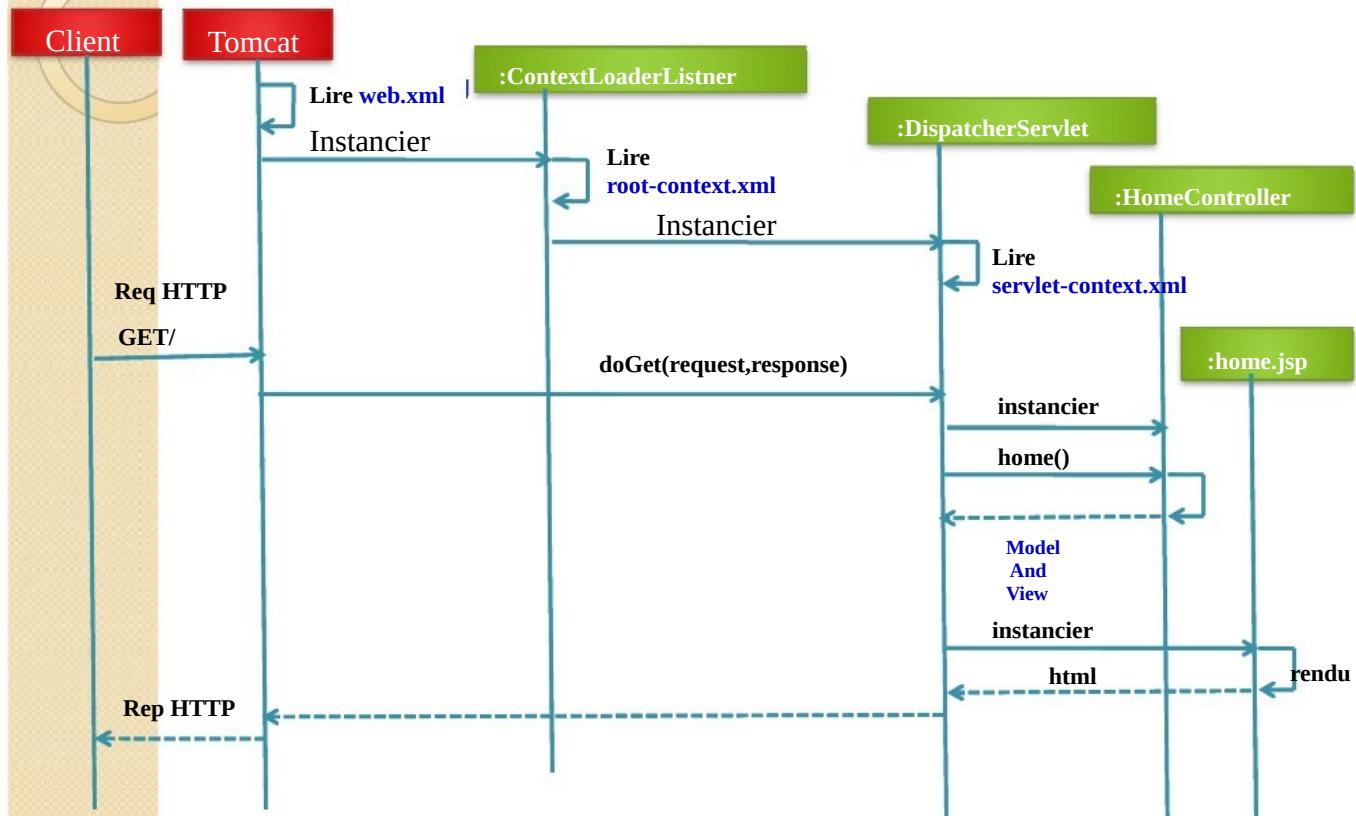


<http://localhost:8080/myCataogue/>

Hello world!

The time on the server is 23 décembre 2013 13:47:12 WET.

Functionnement





- Site de E-Commerce avec E-Spring Framework

Problème

On souhaite créer un site web de commerce électronique qui se compose de deux parties :

- Une partie back office qui nécessite une authentification et qui permet de gérer les produits et les catégories

Nous définissons deux rôles pour cette partie:

ROLE_ADMIN_CAT : l'utilisateur ayant ce rôle a la possibilité de gérer les catégories et les produits (Ajout, suppression, Edition, Modification et consultation) ainsi que les droits d'accès

ROLE_ADMIN_PROD : l'utilisateur ayant ce rôle a la possibilité de gérer uniquement les produits (Ajout, suppression, Edition, Modification et consultation)

- Une partie front office qui représentent la boutique virtuelle qui ne nécessite pas d'authentification. Dans cette partie l'utilisateur a la possibilité de :

Consulter toutes les catégories

Consulter les produits d'une catégorie

Consulter les produits sélectionnés

Chercher des produits par mot clé

Ajouter un produit avec une quantité au panier

Supprimer un produit du panier

Enregistrer le client et la commande des produits de son panier.

Gestion des catégories

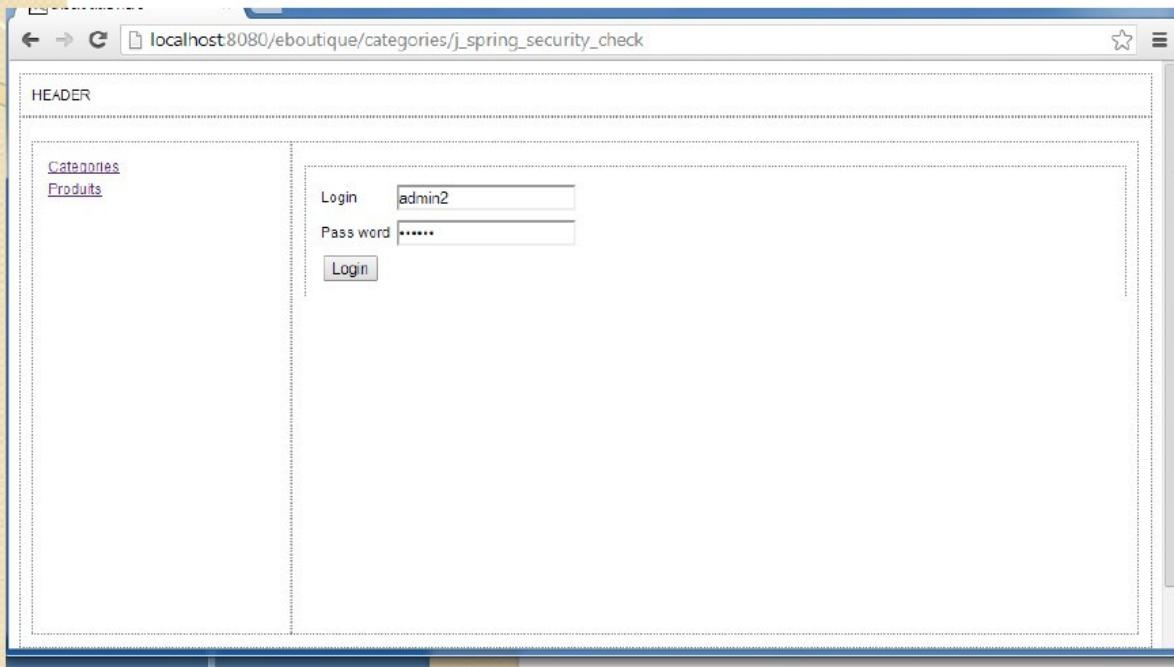
A screenshot of a web browser window titled "localhost:8080/eboutique/categories/index". The browser interface includes standard navigation buttons (back, forward, search) and a star icon. The main content area is a login form. On the left, there is a sidebar with links for "Categories" and "Produits". The main content area contains fields for "Login" and "Pass word", each with an associated input field, and a "Login" button.

HEADER

[Categories](#)
[Produits](#)

Login
Pass word

Gestion des catégories



Gestion des catégories

Insert title here

localhost:8080/eboutique/categories/index

HEADER

Catégories

Produits

ID Catégorie:

Nom Catégorie:

Description:

Photo: Choisissez un fichier Aucun fichier choisi

ID	NOM CAT	Description	Photo		
1	Ordinateurs	Ordinateurs		Supprimer	Edit
2	Imprimantes	Imprimantes		Supprimer	Edit

Gestion des produits

HEADER

[Categories](#)
[Products](#)

Logout Chercher

ID Produit
Catégorie

Désignation
Description
Prix
Sélectionné
Quantité
Photo Aucun fichier choisi

ID	Désignation	Prix	Sélectionnée	Quantité	Photo		
1	HP5600	8000.0	true	12		Supp	Edit
2	Compaq 7600	9000.0	true	80		Supp	Edit

Partie Front Office

localhost:8080/eboutique/ajouterAuPanier?idProduit=2&quantite=3

Nom de produit Total 41000,0

ID	Désignation	Prix	Quantité	Montant
1	HP600	8000,0	1	8000,0
2	Compaq 7600	9000,0	4	36000,0
	Total			44000,0

Outils de recherche

Chercher

Outils de recherche

Désignation : HP600
Prix : 8000,00
Stock : 12

Ajouter au panier

Désignation : Compaq 7600
Prix : 9000,00
Stock : 60

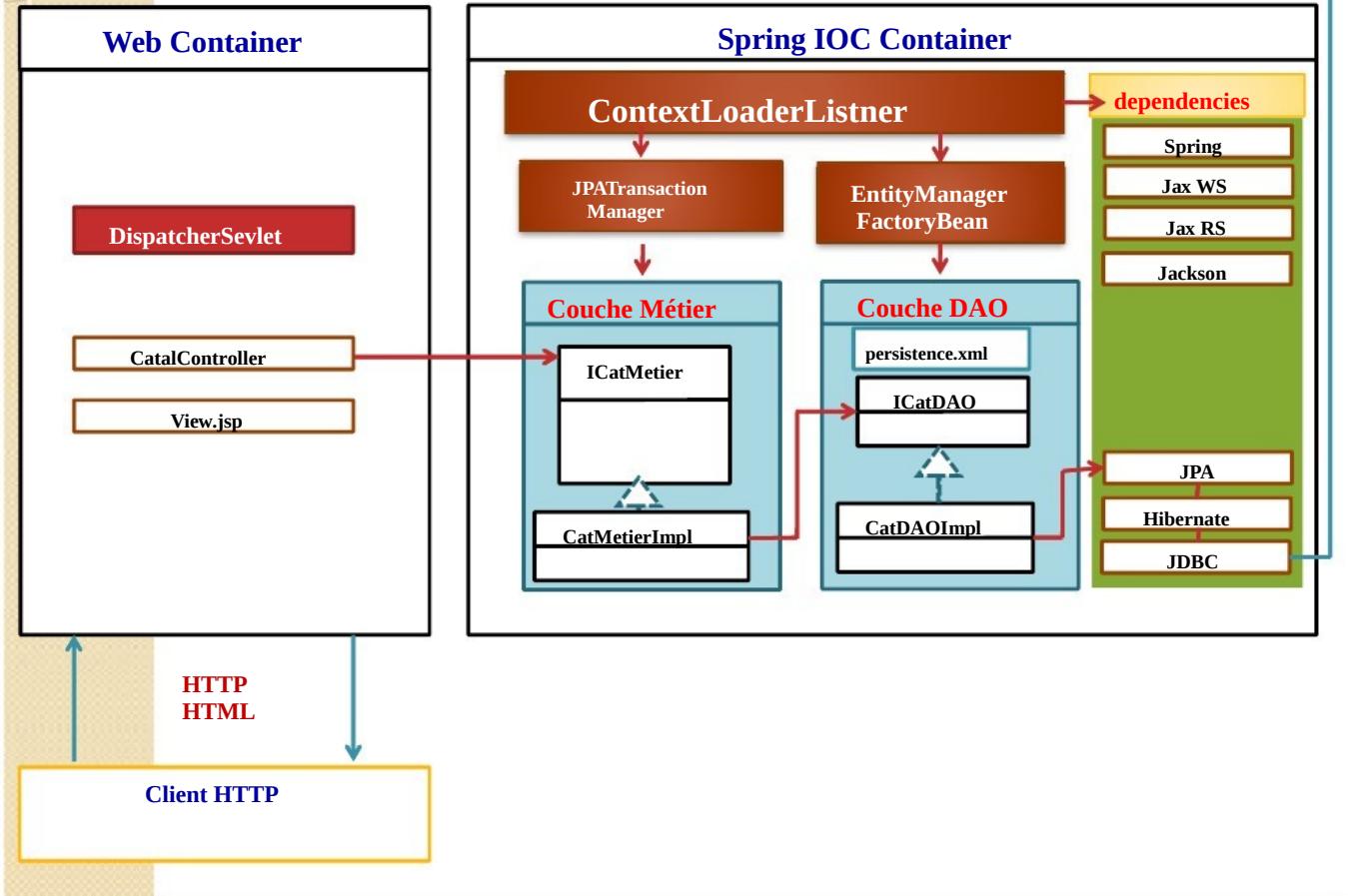
Ajouter au panier

Désignation : HP Jet 5700
Prix : 6500,00
Stock : 3

Ajouter au panier

FOOTER

Architecture



Use Case

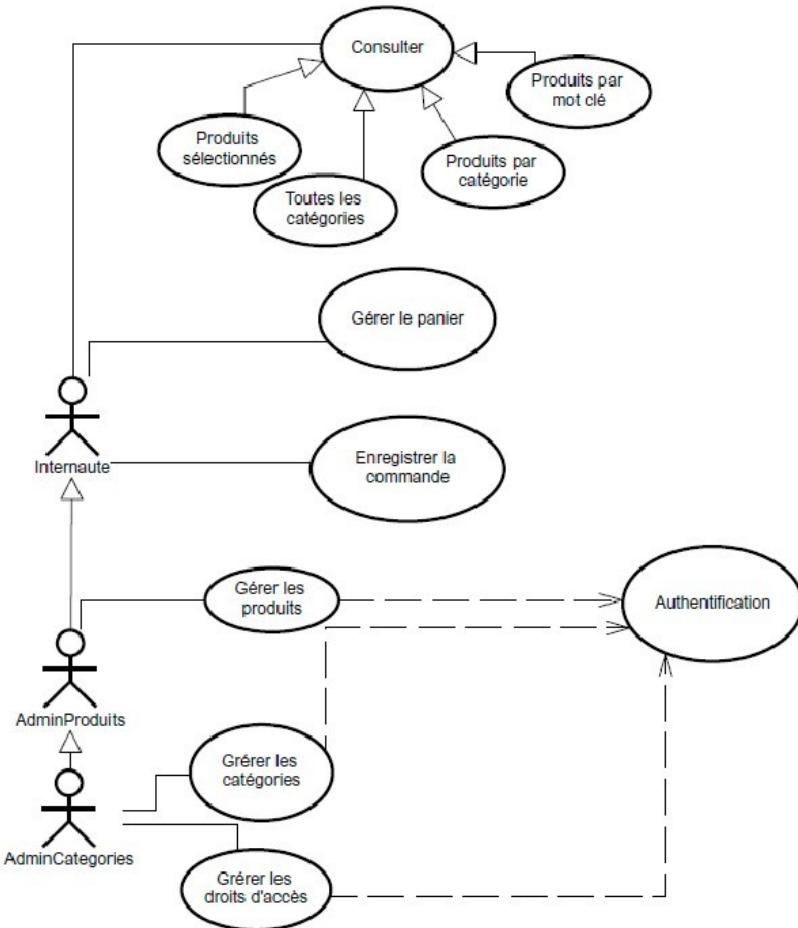


Diagramme de classes des entités

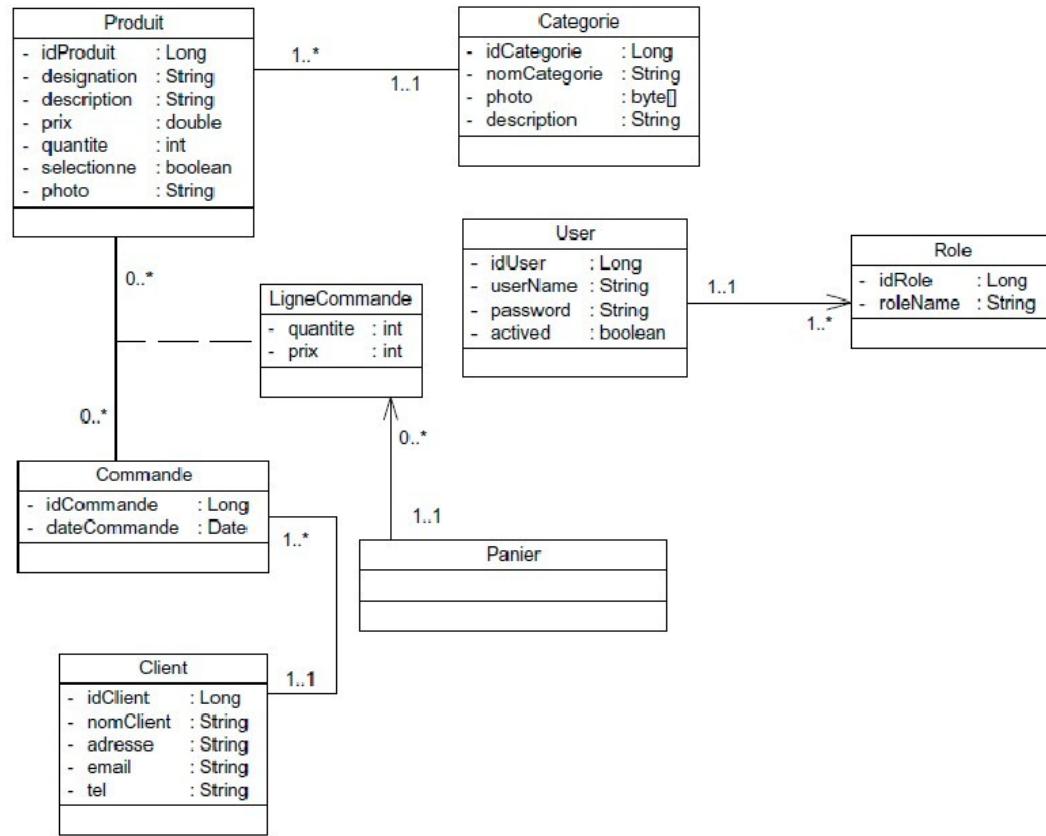
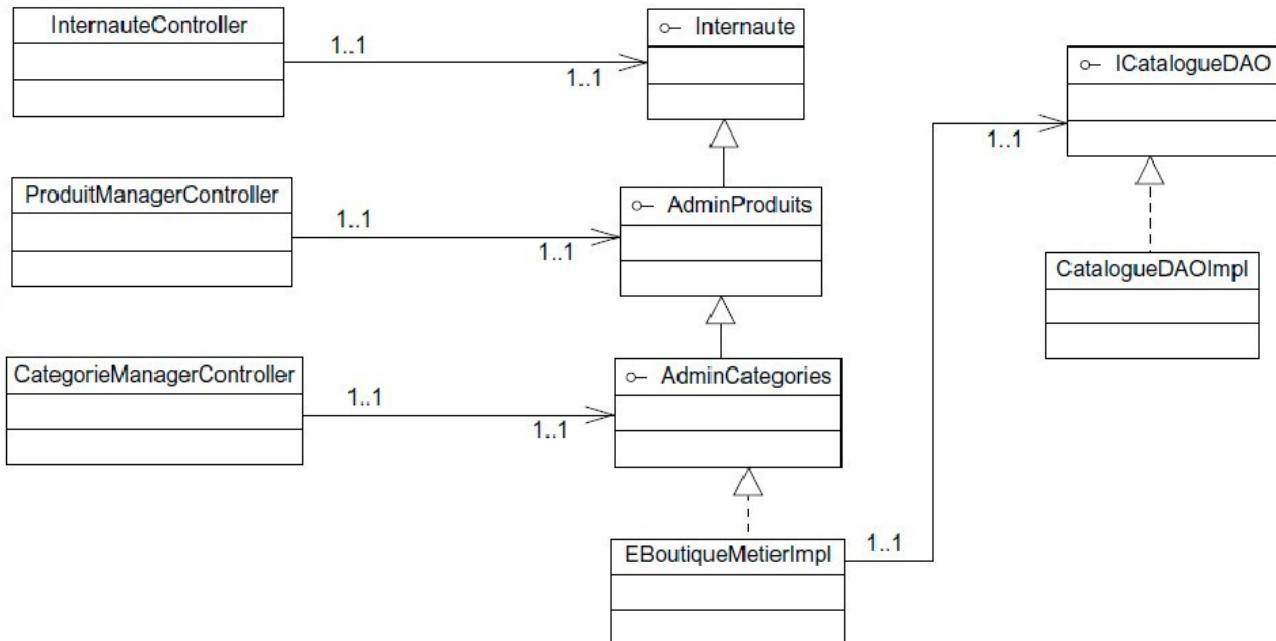


Diagramme de classes des traitements



Création des entités



Maven Dependencies

```
<!-- Hibernate-->
<dependency>
    <groupId>org.hibernate</groupId>
    <artifactId>hibernate-entitymanager</artifactId>
    <version>3.6.0.Final</version>
</dependency>
<dependency>
    <groupId>org.hibernate</groupId>
    <artifactId>hibernate-validator</artifactId>
    <version>4.1.0.Final</version>
</dependency>
<!-- MySQL Connector-->
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>5.1.6</version>
</dependency>
```

Categorie.java

```
package net.youssfieboutique.entities;

import java.io.Serializable; import java.util.*;

import javax.persistence.*; import javax.validation.constraints.Size;

import org.hibernate.validator.constraints.NotEmpty;

@Entity

public class Categorie implements Serializable {

    @Id

    @GeneratedValue

    private Long idCategorie;

    @NotEmpty

    @Size(min=4,max=20)

    private String nomCategorie; private String description;

    private String nomPhoto;

    @Lob

    private byte[] photo;

    @OneToMany(mappedBy="categorie")

    private Collection<Produit> produits=new ArrayList<Produit>();

    // Getters et Setters

    // Constructeurs sans paramètre et avec paramètres

}
```

Produit.java

```
package net.youssfi.eboutique.entities;

import java.io.Serializable; import javax.persistence.*;
import org.hibernate.validator.constraints.NotEmpty;

@Entity
public class Produit implements Serializable {

    @Id
    @GeneratedValue(strategy=GenerationType.IDENTITY)
    private Long idProduit;

    @NotEmpty
    private String designation;      private String description;
    private double prix;           private String photo;
    private int quantite;          private boolean selectionne;

    @ManyToOne
    @JoinColumn(name="ID_CAT")
    private Categorie categorie;

    // Getters et Setters
    // Constructeur sans paramètre
    // Constructeur avec paramètres

}
```

Client.java

```
package net.youssfieboutique.entities;

import java.io.Serializable;
import java.util.Collection;
import javax.persistence.*;

@Entity
public class Client implements Serializable {

    @Id
    @GeneratedValue(strategy=GenerationType.AUTO)
    private Long idClient;

    private String nomClient;

    private String adresse;

    private String email;

    private String tel;

    @OneToMany(mappedBy="client")
    private Collection<Commande> commandes;

    // Getters et Setters
    // Constructeur sans paramètre
    // Constructeur avec paramètres

}
```

Commande.java

```
package net.youssfieboutique.entities;

import java.io.Serializable;
import java.util.*;
import javax.persistence.*;

@Entity
public class Commande implements Serializable {

    @Id
    @GeneratedValue(strategy=GenerationType.AUTO)
    private Long idCommande;
    private Date dateCommande;

    @ManyToOne
    @JoinColumn(name="idClient")
    private Client client;

    @OneToMany
    @JoinColumn(name="idCommande")
    private Collection<LigneCommande> ligneCommandes;

    // Getters et Setters
    // Constructeur sans paramètre
    // Constructeur avec paramètres
}
```

LigneCommande.java

```
package net.youssfieboutique.entities;  
import java.io.Serializable;  
import javax.persistence.*;  
  
@Entity  
public class LigneCommande implements Serializable {  
  
    @Id  
    @GeneratedValue(strategy=GenerationType.AUTO)  
    private Long id;  
  
    @ManyToOne  
    @JoinColumn(name="idproduit")  
    private Produit produit;  
    private int quantite;  
    private double prix;  
    // Getters et Setters  
    // Constructeur sans paramètre  
    // Constructeur avec paramètres  
}
```

Panier.java

```
package net.youssfieboutique.entities;  
import java.io.Serializable;  
import java.util.*;  
  
public class Panier implements Serializable {  
    private Map<Long, LigneCommande>      items=new HashMap<Long, LigneCommande>();  
    public void addItem(Produit p, int quantite){  
        LigneCommande lc=items.get(p.getIdProduit());  
        if(lc==null){  
            LigneCommande art=new LigneCommande();  
            art.setProduit(p);  
            art.setQuantite(quantite);  
            art.setPrix(p.getPrix());  
            items.put(p.getIdProduit(), art);  
        }  
        else{  
            lc.setQuantite(lc.getQuantite()+quantite);  
        }  
    }  
}
```

Panier.java

```
public Collection<LigneCommande> getItems(){
    return items.values();
}

public int getSize(){
    return items.size();
}

public double getTotal(){
    double total=0;
    for(LigneCommande lc:items.values()){
        total+=lc.getPrix()*lc.getQuantite();
    }
    return total;
}

public void deleteItem(Long idproduit){
    items.remove(idproduit);
}

}
```



Tester les entités

/

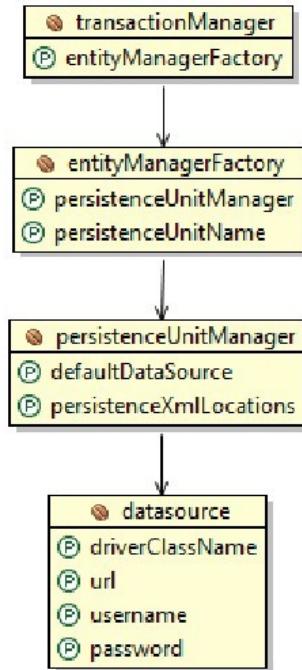
EBoutiqueV2/src/main/resources/META-INF/EBoutiqueV2/src/main/resources/META-INF/persistence.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<persistence version="2.0" xmlns="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-unit name="UP_EBOUTIQUE" transaction-type="RESOURCE_LOCAL">
    <provider>org.hibernate.ejb.HibernatePersistence</provider>
    <properties>
      <property name="hibernate.show_sql" value="true"/>
      <property name="hibernate.hbm2ddl.auto" value="update"/>
      <property name="hibernate.dialect"
        value="org.hibernate.dialect.MySQLDialect"/>
    </properties>
  </persistence-unit>
</persistence>
```

Configuration JPA avec Spring IOC

Spring Beans



Maven Properties

```
<properties>
<java-version>1.6</java-version>
<org.springframework-
    version>3.2.2.RELEASE</org.springframework-version>
<org.aspectj-version>1.6.10</org.aspectj-version>
<org.slf4j-version>1.6.6</org.slf4j-version>

</properties>
```

Maven Dependencies

```
<!-- Spring -->
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
    <version>${org.springframework-version}</version>
    <exclusions>
        <!-- Exclude Commons Logging in favor of SLF4j -->
        <exclusion>
            <groupId>commons-logging</groupId>
            <artifactId>commons-logging</artifactId>
        </exclusion>
    </exclusions>
</dependency>
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-webmvc</artifactId>
    <version>${org.springframework-version}</version>
</dependency>
```

Maven Dependencies

```
<!-- Spring -->
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-tx</artifactId>
    <version>>${org.springframework-version}</version>
</dependency>
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-orm</artifactId>
    <version>>${org.springframework-version}</version>
</dependency>
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-beans</artifactId>
    <version>>${org.springframework-version}</version>
</dependency>
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-core</artifactId>
    <version>>${org.springframework-version}</version>
</dependency>
```

Configuration JPA avec Spring IOC :

```
<?xml version="1.0" encoding="UTF-8"?>
<!--EBoutiqueV2/src/main/resources/applicationContext.xml/EBoutique
V2/src/main/resources/applicationContext.xml-->

<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:context="http://www.springframework.org/schema/context"
       xmlns:tx="http://www.springframework.org/schema/tx"
       xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd
                           http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-3.2.xsd
                           http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx-3.2.xsd">

    <bean id="datasource"
          class="org.springframework.jdbc.datasource.DriverManagerDataSource">
        <property name="driverClassName" value="com.mysql.jdbc.Driver"></property>
        <property name="url" value="jdbc:mysql://localhost:3306/eboutique"></property>
        <property name="username" value="root"></property>
        <property name="password" value=""></property>
    </bean>

```

Configuration JPA avec Spring IOC :

/EBoutiqueV2/src/main/resources/applicationContext.xml/EBoutiqueV2/src/main/resources/applicationContext.xml

```
<!-- application context -->
<bean id="persistenceUnitManager"
      class="org.springframework.orm.jpa.persistenceunit.DefaultPersistenceUnitManager">

    <property name="defaultDataSource" ref="datasource"></property>

    <property name="persistenceXmlLocations">
        <list>
            <value>classpath*:META-INF/persistence.xml</value>
        </list>
    </property>
</bean>

<bean id="entityManagerFactory"
      class="org.springframework.orm.jpa.LocalContainerEntityManagerFactoryBean">

    <property name="persistenceUnitManager" ref="persistenceUnitManager"></property>
    <property name="persistenceUnitName" value="UP_EBOUTIQUE"></property>
</bean>

<bean id="transactionManager"
      class="org.springframework.orm.jpa.JpaTransactionManager">

    <property name="entityManagerFactory" ref="entityManagerFactory"></property>
</bean>

<tx:annotation-driven transaction-manager="transactionManager"/>
<context:annotation-config></context:annotation-config>
</beans>
```

JUnit Test :

src/test/java/TestDao.java

```
package net.youssfi.eboutique;
import static org.junit.Assert.*;
import org.junit.*;
import org.springframework.context.support.ClassPathXmlApplicationContext;
public class TestDAO {
    @Before
    public void setUp() throws Exception {
    }
    @Test
    public void test() {
        try {
            ClassPathXmlApplicationContext app=
new ClassPathXmlApplicationContext(new String[]{"applicationContext.xml"});
            assertTrue(true);
        } catch (Exception e) {
            assertTrue(e.getMessage(),false);
        }
    }
}
```

Tables générées

Deuxième Partie



COUCHES DAO ET MÉTIER

Inteface IBoutiqueDao

```
package net.youssfi.eboutique.dao;
import java.util.List;
import net.youssfi.eboutique.entities.*;
public interface IBoutiqueDao {
    public Long ajouterCategorie(Categorie c);
    public List<Categorie> listCategories();
    public Categorie getCategorie(Long idCat);
    public void supprimerCategrorie(Long idcat);
    public void modifierCategorie(Categorie c);

    public Long ajouterProduit(Produit p, Long idCat);
    public List<Produit> listproduits();
    public List<Produit> produitsParMotCle(String mc);
    public List<Produit> produitsParCategorie(Long idCat);
    public List<Produit> produitsSelectionnes();
    public Produit getProduit(Long idP);
    public void supprimerProduit(Long idP);
    public void modifierProduit(Produit p);

    public void ajouterUser(User u);
    public void attribuerRole(Role r,Long userID);
    public Commande enregistrerCommande(Panier p,Client c);
}
```

BoutiqueDaoImpl

```
package net.youssfi.eboutique.dao;

import java.util.List;
import javax.persistence.*;
import javax.persistence.Query;
import net.youssfi.eboutique.entities.*;

public class BoutiqueDaoImpl implements IBoutiqueDao {

    @PersistenceContext
    private EntityManager em;

    @Override
    public Long ajouterCategorie(Categorie c) {
        em.persist(c);
        return c.getIdCategorie();
    }

    @Override
    public List<Categorie> listCategories() {
        Query req=em.createQuery("select c from Categorie c");
        return req.getResultList();
    }
}
```

BoutiqueDaoImpl

```
@Override
public Categorie getCategorie(Long idCat) {
    return em.find(Categorie.class, idCat);
}

@Override
public void supprimerCategorie(Long idcat) {
    Categorie c=em.find(Categorie.class, idcat);
    em.remove(c);
}

@Override
public void modifierCategorie(Categorie c) {
    em.merge(c);
}

@Override
public Long ajouterProduit(Produit p, Long idCat) {
    Categorie c=getCategorie(idCat);
    p.setCategorie(c);em.persist(p);
    return p.getIdProduit();
}
```

BoutiqueDaoImpl

```
@Override  
public List<Produit> listproduits() {  
    Query req=em.createQuery("select p from Produit p");  
    return req.getResultList();  
}  
  
@Override  
public List<Produit> produitsParMotCle(String mc) {  
    Query req=em.createQuery("select p from Produit p where p.designation like  
        :x or p.description like:x");  
    req.setParameter("x", "%" + mc + "%");  
    return req.getResultList();  
}  
  
@Override  
public List<Produit> produitsParCategorie(Long idCat) {  
    Query req=em.createQuery("select p from Produit p where  
        p.categorie.idCategorie=:x");  
    req.setParameter("x", idCat);  
    return req.getResultList();  
}
```

BoutiqueDaoImpl

```
@Override  
public List<Produit> produitsSelectionnes() {  
    Query req=em.createQuery("select p from Produit p where  
        p.selectionne=true");  
    return req.getResultList();  
}  
  
@Override  
public Produit getProduit(Long idP) {  
    return em.find(Produit.class, idP);  
}  
@Override  
public void supprimerProduit(Long idP) {  
    Produit p=getProduit(idP);  
    em.remove(p);  
}
```

BoutiqueDaoImpl

```
@Override  
public void modifierProduit(Produit p) {  
    em.merge(p);  
}  
@Override  
public void ajouterUser(User u) {  
    em.persist(u);  
}  
@Override  
public void attribuerRole(Role r, Long userID) {  
    User u=em.find(User.class, userID);  
    r.setUser(u);  
    em.persist(r);  
}
```

BoutiqueDaoImpl

```
@Override  
public Commande enregistrerCommande(Panier panier, Client c) {  
    em.persist(c);  
    Commande cmd=new Commande();  
    cmd.setClient(c);  
    cmd.setLigneCommandes(panier.getItems());  
    em.persist(cmd);  
    return cmd;  
}  
}
```



COUCHE MÉTIER

Interface InternauteBoutiqueMetier

```
package net.youssfi.eboutique.metier;
import java.util.List;
import net.youssfi.eboutique.entities.*;
public interface InternauteBoutiqueMetier {
    public List<Categorie> listCategories();
    public Categorie getCategorie(Long idCat);
    public List<Produit> listproduits();
    public List<Produit> produitsParMotCle(String mc);
    public List<Produit> produitsParCategorie(Long idCat);
    public List<Produit> produitsSelectionnes();
    public Produit getProduit(Long idP);
    public Commande enregistrerCommande(Panier p,Client c);
}
```

Interface IAdminProduitsMetier

```
package net.youssfi.eboutique.metier;
import net.youssfi.eboutique.entities.Produit;
public interface IAdminProduitMetier extends
    InternauteBoutiqueMetier {
    public Long ajouterProduit(Produit p, Long idCat);
    public void supprimerProduit(Long idP);
    public void modifierProduit(Produit p);
}
```

Interface IAdminCategoriesMetier

```
package net.youssfi.eboutique.metier;  
  
import net.youssfi.eboutique.entities.Categorie;  
import net.youssfi.eboutique.entities.Role;  
import net.youssfi.eboutique.entities.User;  
  
public interface IAdminCategoriesMetier extends IAdminProduitMetier {  
    public Long ajouterCategorie(Categorie c);  
    public void supprimerCategrorie(Long idcat);  
    public void modifierCategorie(Categorie c);  
    public void ajouterUser(User u);  
    public void attribuerRole(Role r,Long userID);  
}
```

Implémentation BoutiqueMetierImpl

```
package net.youssfi.eboutique.metier;
import java.util.List;
import org.springframework.transaction.annotation.Transactional;
import net.youssfi.eboutique.dao.*;
import net.youssfi.eboutique.entities.*;
@Transactional
public class BoutiqueMetierImpl implements IAdminCategoriesMetier{
    private IBoutiqueDao dao;
    public void setDao(IBoutiqueDao dao) {
        this.dao = dao;
    }
    @Override
    public Long ajouterProduit(Produit p, Long idCat) {
        return dao.ajouterProduit(p, idCat);
    }
}
```

Implémentation BoutiqueMetierImpl

```
@Override
public void supprimerProduit(Long idP) {
    dao.supprimerProduit(idP);
}

@Override
public void modifierProduit(Produit p) {
    dao.modifierProduit(p);
}

@Override
public List<Categorie> listCategories() {
    return dao.listCategories();
}

@Override
public Categorie getCategorie(Long idCat) {
    return dao.getCategorie(idCat);
}
```

Implémentation BoutiqueMetierImpl

```
@Override
public List<Produit> listproduits() {
    return dao.listproduits();
}

@Override
public List<Produit> produitsParMotCle(String mc) {
    return dao.produitsParMotCle(mc);
}

@Override
public List<Produit> produitsParCategorie(Long idCat) {
    return dao.produitsParCategorie(idCat);
}

@Override
public List<Produit> produitsSelectionnes() {
    return dao.produitsSelectionnes();
}
```

Implémentation BoutiqueMetierImpl

```
@Override
public Produit getProduit(Long idP) {
    return dao.getProduit(idP);
}

@Override
public Commande enregistrerCommande(Panier p, Client c) {
    return dao.enregistrerCommande(p, c);
}

@Override
public Long ajouterCategorie(Categorie c) {
    return dao.ajouterCategorie(c);
}

@Override
public void supprimerCategorie(Long idcat) {
    dao.supprimerCategorie(idcat);
}
```

Implémentation BoutiqueMetierImpl

```
@Override
public void modifierCategorie(Categorie c) {
    dao.modifierCategorie(c);
}

@Override
public void ajouterUser(User u) {
    dao.ajouterUser(u);
}

@Override
public void attribuerRole(Role r, Long userID) {
    dao.attribuerRole(r, userID);
}
```



Tester les deux couches métiers et dao

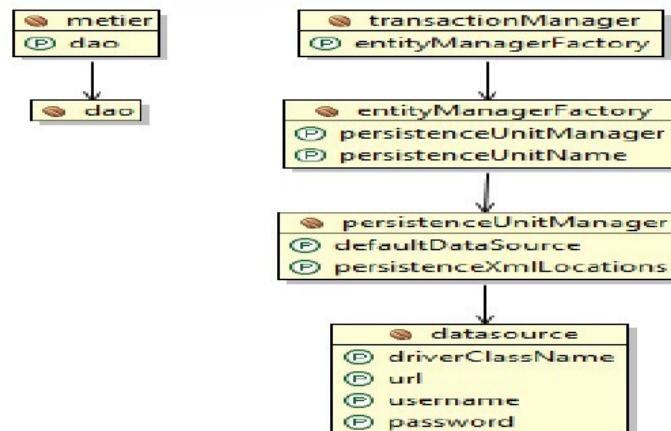
Injection des dépendances :

/

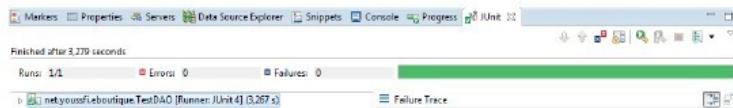
EBoutiqueV2/src/main/resources/applicationContext.xml/EBoutique
V2/src/main/resources/applicationContext.xml

```
<bean id="dao" class="net.youssfi.eboutique.dao.BoutiqueDaoImpl"></bean>  
  
<bean id="metier" class="net.youssfi.eboutique.metier.BoutiqueMetierImpl">  
    <property name="dao" ref="dao"></property>  
</bean>
```

Spring Beans



JUnit Test

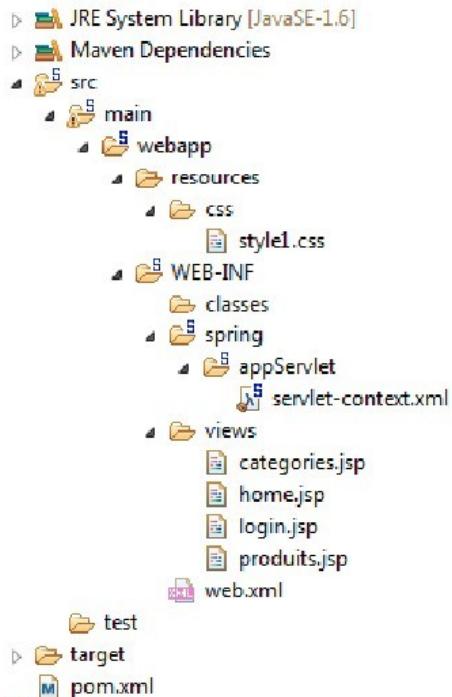
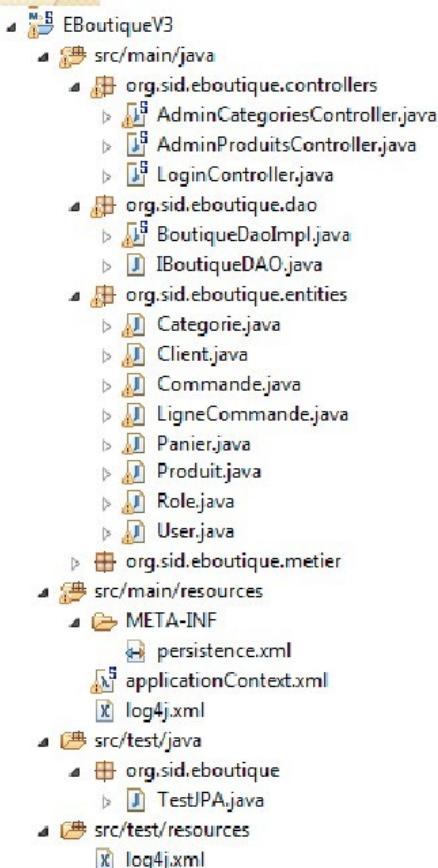


```
package net.youssfi.eboutique;
import static org.junit.Assert.*;import java.util.List;import net.youssfi.eboutique.entities.*;
import net.youssfi.eboutique.metier.IAdminCategoriesMetier;
import org.junit.*; import org.springframework.context.support.*;
public class TestDAO {
    @Before
    public void setUp() throws Exception { }
    @Test
    public void test() {
        try {
            ClassPathXmlApplicationContext context=_____
            new ClassPathXmlApplicationContext(new String[]{"applicationContext.xml"});
            IAdminCategoriesMetier metier=(IAdminCategoriesMetier) context.getBean("metier");
            List<Categorie> cats1=metier.listCategories();
            metier.ajouterCategorie(new Categorie("Ordinateur", "Ordinateurs", "", null));
            metier.ajouterCategorie(new Categorie("Imprimantes", "Imprimantes", "", null));
            List<Categorie> cats2=metier.listCategories();
            assertTrue(cats2.size()==cats1.size()+2);
        } catch (Exception e) { assertTrue(e.getMessage(),false); }
    }
}
```

Troisième Partie

• COUCHE WEB

Structure du projet



web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="2.5" xmlns="http://java.sun.com/xml/ns/javaee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-
    app_2_5.xsd">

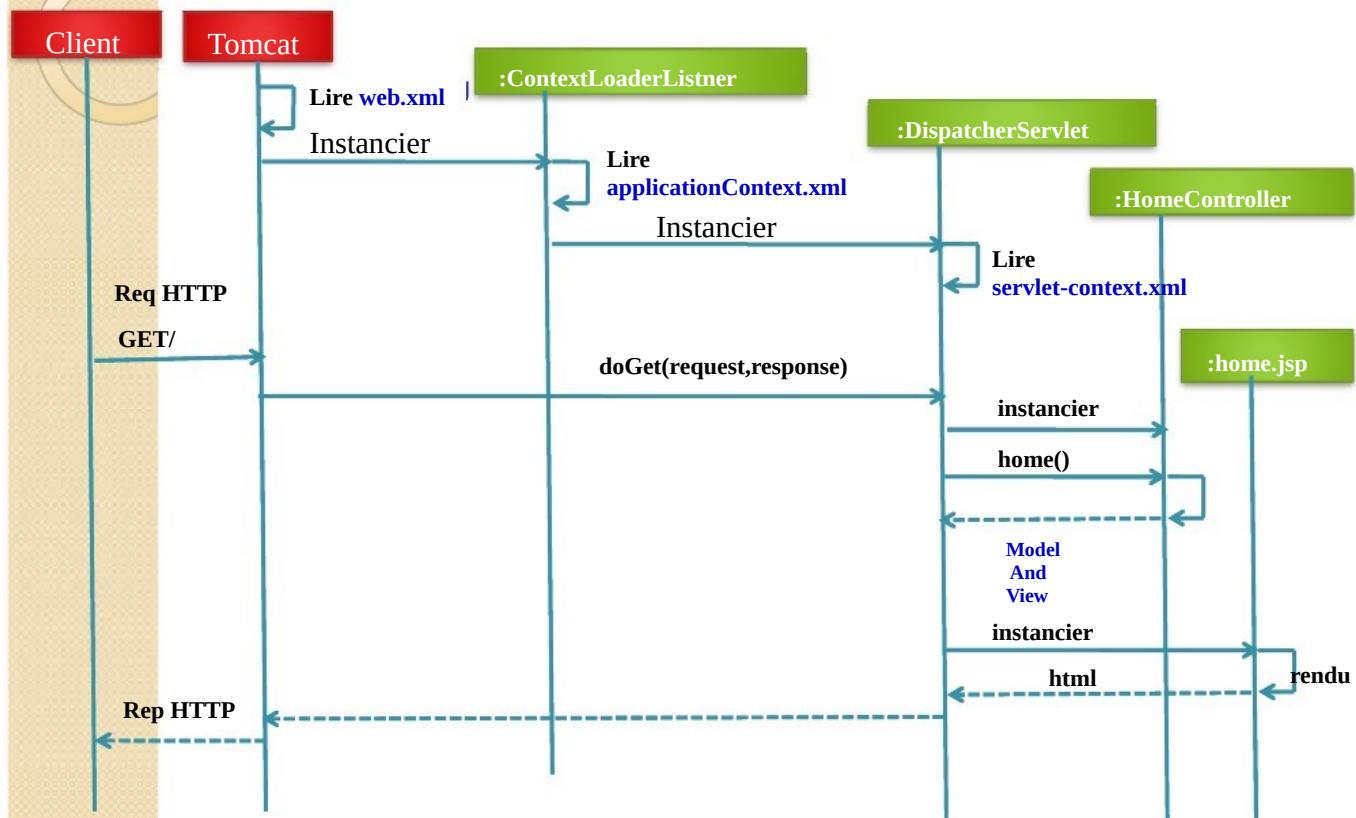
  <!-- The definition of the Root Spring Container shared by all Servlets and Filters -
      ->

  <context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>classpath*:applicationContext.xml</param-value>
  </context-param>
  <!-- Creates the Spring Container shared by all Servlets and Filters -->
  <listener>
    <listener-class>
      org.springframework.web.context.ContextLoaderListener
    </listener-class>
  </listener>
```

web.xml

```
<!-- Processes application requests -->
<servlet>
    <servlet-name>appServlet</servlet-name>
    <servlet-
        class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
        <init-param>
            <param-name>contextConfigLocation</param-name>
            <param-value>/WEB-INF/spring/appServlet/servlet-context.xml</param-value>
        </init-param>
        <load-on-startup>1</load-on-startup>
    </servlet>
    <servlet-mapping>
        <servlet-name>appServlet</servlet-name>
        <url-pattern>/</url-pattern>
    </servlet-mapping>
</web-app>
```

Fonctionnement



Maven Dependencies

```
<!-- Apache Commons Upload -->
<dependency>
    <groupId>commons-fileupload</groupId>
    <artifactId>commons-fileupload</artifactId>
    <version>1.2.2</version>
</dependency>
<dependency>
    <groupId>org.apache.commons</groupId>
    <artifactId>commons-io</artifactId>
    <version>1.3.2</version>
</dependency>
```

/WEB-INF/spring/appServlet/servlet-/WEB-INF/spring/appServlet/servlet-context.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans xmlns="http://www.springframework.org/schema/mvc"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:beans="http://www.springframework.org/schema/beans"
    xmlns:context="http://www.springframework.org/schema/context"
    xsi:schemaLocation="http://www.springframework.org/schema/mvc
        http://www.springframework.org/schema/mvc/spring-mvc.xsd
        http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-
        beans.xsd
        http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-
        context.xsd">

    <!-- DispatcherServlet Context: defines this servlet's request-
        processing infrastructure -->
    <!-- Enables the Spring MVC @Controller programming model -->
    <annotation-driven />
    <!-- Handles HTTP GET requests for /resources/** by efficiently
        serving up static resources in the ${webappRoot}/resources
        directory -->
    <resources mapping="/resources/**" location="/resources/" />
```

/WEB-INF/spring/appServlet/servlet-/WEB-INF/spring/appServlet/servlet-context.xml

```
<beans:bean  
    class="org.springframework.web.servlet.view.InternalResourceViewResolver">  
    <beans:property name="prefix" value="/WEB-INF/views/" />  
    <beans:property name="suffix" value=".jsp" />  
</beans:bean>  
<context:component-scan base-package="net.youssfi.eboutique" />  
</beans:beans>  
  
<context:component-scan base-package="net.youssfi.eboutique" />  
  
<!-- Configuration Upload-->  
  
<beans:bean name="multipartResolver"  
    class="org.springframework.web.multipart.commons.CommonsMultipartResolver">  
    <beans:property name="maxUploadSize" value="100000"/></beans:property>  
</beans:bean>
```



Gestion des catégories

AdminCategoriesController

```
package net.youssf.eboutique.controllers; import java.io.*;  
import javax.validation.Valid;  
import net.youssf.eboutique.entities.Categorie;  
import net.youssf.eboutique.metier.IAdminCategoriesMetier;  
import org.apache.commons.io.IOUtils;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.http.MediaType;  
import org.springframework.stereotype.Controller;  
import org.springframework.ui.Model;  
import org.springframework.validation.BindingResult;  
import org.springframework.web.bind.annotation.RequestMapping;  
import org.springframework.web.bind.annotation.ResponseBody;  
import org.springframework.web.bind.annotation.SessionAttributes;  
import org.springframework.web.multipart.MultipartFile;
```

AdminCategoriesController

```
@Controller  
@RequestMapping("/adminCat")  
@SessionAttributes("editedCat")  
public class AdminCategoriesController {  
    @Autowired  
    private IAdminCategoriesMetier metier;  
  
    @RequestMapping("/index")  
    public String index(Model model){  
        model.addAttribute("categorie", new Categorie());  
        model.addAttribute("categories", metier.listCategories());  
        return "categories";  
    }  
}
```

AdminCategoriesController

```
@RequestMapping("/saveCat")
public String saveCat(@Valid Categorie c,BindingResult bindingResult,
    Model model,MultipartFile file) throws Exception{
    if(bindingResult.hasErrors()){
        model.addAttribute("categories", metier.listCategories());
        return "categories";
    }
    if(!file.isEmpty())c.setPhoto(file.getBytes());
    else{
        if(c.getIdCategorie()!=null){
            Categorie cat=(Categorie) model.asMap().get("editedCat");
            c.setPhoto(cat.getPhoto());
        }
    }
    if(c.getIdCategorie()==null) metier.ajouterCategorie(c);
    else metier.modifierCategorie(c);
    model.addAttribute("categorie", new Categorie());
    model.addAttribute("categories", metier.listCategories());
    return "categories";
}
```

AdminCategoriesController

```
@RequestMapping(value="/photoCat",produces=MediaType.IMAGE_JPEG_VALUE)
@ResponseBody
public byte[] getPhoto(Long idCat) throws IOException{
    Categorie c=metier.getCategorie(idCat);
    if(c.getPhoto()==null) return new byte[0];
    else return IOUtils.toByteArray(new ByteArrayInputStream(c.getPhoto()));
}

@RequestMapping(value="/suppCat")
public String suppCat(Long idCat,Model model){
    metier.supprimerCategrorie(idCat);
    model.addAttribute("categorie", new Categorie());
    model.addAttribute("categories", metier.listCategories());
    return "categories";
}

@RequestMapping(value="/editCat")
public String editCat(Long idCat,Model model){
    Categorie c=metier.getCategorie(idCat);
    model.addAttribute("editedCat", c);model.addAttribute("categorie",c );
    model.addAttribute("categories", metier.listCategories());
    return "categories";}
```

Vue : categories.jsp

Screenshot of a web browser showing the 'categories.jsp' page. The URL in the address bar is 'localhost:8080/eboutique/adminCat/saveCat'. The page displays a form for adding a category and a table listing existing categories.

Form Fields:

- ID Catégorie: [Input field]
- Nom Catégorie: [Input field]
- Description: [Input field]
- Photo: [File input field] Choisissez un fichier | Aucun fichier choisi
- Save: [Submit button]

Table Data:

ID	NOM CAT	DESCRIPTION	PHOTO		
11	Ordinateurs	mmmmmmmmmmmm		Supprimer	Edit
12	aaaaaaa	mmmmmmmm		Supprimer	Edit

Vue : categories.jsp

```
<%@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<%@taglib uri="http://www.springframework.org/tags/form" prefix="f"%>
<head>
    <link rel="stylesheet" type="text/css" href="<%=request.getContextPath()
    %>/resources/css/style.css">
</head>
<div id="formCat" class="cadre">
    <f:form modelAttribute="categorie" action="saveCat"           method="post"
    enctype="multipart/form-data">
        <table>
            <tr>
                <td>ID Catégorie:</td>
                <td>${categorie.idCategorie}<f:input type="hidden" path="idCategorie"/></td>
                <td><f:errors path="idCategorie"></f:errors> </td>
            </tr>
            <tr>
                <td>Nom Catégorie</td><td><f:input path="nomCategorie"/></td>
                <td><f:errors path="nomCategorie"></f:errors> </td>
            </tr>
        </table>
    </f:form>
</div>
```

Vue : categories.jsp

```
<tr>
    <td>Description</td>
    <td><f:textarea path="description"/></td>
    <td><f:errors path="description"></f:errors> </td>
</tr>
<tr>
    <td>Photo</td>
    <c:if test="${categorie.idCategorie!=null}">
        <td></td>
    </c:if>
    <td>
        <input type="file" name="file"></td>
    </tr>
    <tr>
        <td><input type="submit" value="Save"></td>
    </tr>
</table>
</f:form>
</div>
```

Vue : categories.jsp

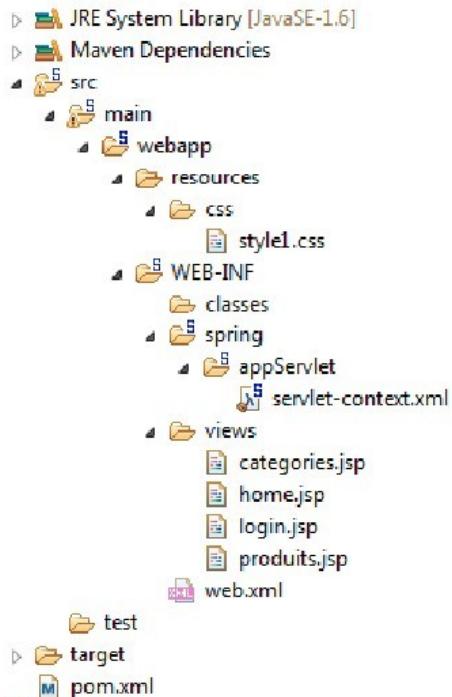
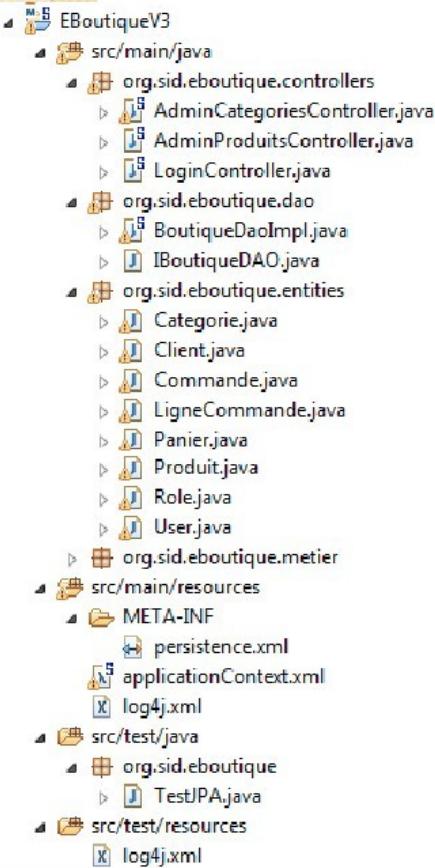
```
<div id="tabCategories" class="cadre">
<table class="tabStyle1">
<tr>
    <th>ID</th><th>NOM CAT</th><th>DESCRIPTION</th><th>PHOTO</th>
    <th></th><th></th>
</tr>
<c:forEach items="${categories}" var="cat">
    <tr>
        <td>${cat.idCategorie}</td>
        <td>${cat.nomCategorie}</td>
        <td>${cat.description}</td>
        <td></td>
        <td><a href="suppCat?idCat=${cat.idCategorie}">Supprimer</a></td>
        <td><a href="editCat?idCat=${cat.idCategorie}">Edit</a></td>
    </tr>
</c:forEach>
</table>
</div>
```

Feuille de style CSS :

/

```
EBoutiqueV2/src/main/webapp/resources/css/style.css/EBoutique  
Y2/src/main/webapp/resources/css/style.css  
div.cadre{  
    border: 1px dotted gray;  
    border-radius:10px;  
    margin: 10px;  
    padding: 10px;  
}  
.tabStyle1 th{  
    border: 1px dotted gray;  
    background: pink;  
    padding: 5px;  
}  
.tabStyle1 td{  
    border: 1px dotted gray;  
    background: white;  
    padding: 5px;  
}
```

Structure du projet





Gestion des produits

AdminProduitsController

```
package org.sid.eboutique.controllers;  
import java.io.*;  
import javax.validation.Valid;  
import org.apache.commons.io.IOUtils;  
import org.sid.eboutique.entities.Produit;  
import org.sid.eboutique.metier.IAdminProduitsMetier;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.http.MediaType;  
import org.springframework.stereotype.Controller;  
import org.springframework.ui.Model;  
import org.springframework.validation.BindingResult;  
import org.springframework.web.bind.annotation.RequestMapping;  
import org.springframework.web.bind.annotation.ResponseBody;  
import org.springframework.web.multipart.MultipartFile;
```

AdminProduitsController

```
@Controller  
@RequestMapping(value="/adminProd")  
public class AdminProduitsController {  
    @Autowired  
    private IAdminProduitsMetier metier;  
  
    @RequestMapping(value="/index")  
    public String index(Model model){  
        model.addAttribute("produit",new Produit());  
        model.addAttribute("produits",metier.listproduits());  
        model.addAttribute("categories", metier.listCategories());  
        return "produits";  
    }  
}
```

AdminProduitsController

```
@RequestMapping(value="/saveProd")
public String saveProd(@Valid Produit p,BindingResult bindingResult,
Model model,MultipartFile file) throws IOException{
if(bindingResult.hasErrors()){
    model.addAttribute("categories",metier.listCategories());
    model.addAttribute("produits",metier.listproduits());
    return("produits");}
if(!file.isEmpty()) {
    String path=System.getProperty("java.io.tmpdir");
    p.setPhoto(file.getOriginalFilename());
    Long idP=null;
    if(p.getIdProduit()==null){
        idP=metier.ajouterProduit(p,p.getCategorie().getIdcategorie());
        else{ metier.modifierProduit(p);idP=p.getIdProduit();}
        file.transferTo(new File(path+"/"+"PROD_"+idP+"_"+file.getOriginalFilename()));
    }else{ if(p.getIdProduit()==null)
        metier.ajouterProduit(p, p.getCategorie().getIdcategorie());
        else metier.modifierProduit(p);
    }
    model.addAttribute("produit",new Produit());
    model.addAttribute("produits",metier.listproduits());
    model.addAttribute("categories",metier.listCategories());
    return "produits";
}
```

AdminProduitsController

```
@RequestMapping(value="photoProd", produces=MediaType.IMAGE_JPEG_VALUE)
@ResponseBody
public byte[] photCat(Long idProd) throws IOException{
    Produit p=metier.getProduit(idProd);
    File f=new
    File(System.getProperty("java.io.tmpdir")+"/PROD_"+idProd+"_"+p.getPhoto());
    return IOUtils.toByteArray(new FileInputStream(f));
}

@RequestMapping(value="/suppProd")
public String supp(Long idProd,Model model){
    metier.supprimerProduit(idProd);
    model.addAttribute("produit",new Produit());
    model.addAttribute("produits",metier.listproduits());
    model.addAttribute("categories", metier.listCategories());
    return "produits";
}
```

AdminProduitsController

```
@RequestMapping(value="/editProd")
public String edit(Long idProd,Model model){
    Produit p=metier.getProduit(idProd);
    model.addAttribute("produit",p);
    model.addAttribute("produits",metier.listproduits());
    model.addAttribute("categories", metier.listCategories());
    return "produits";
}
}
```

Vue : produits.jsp

localhost:8080/eboutique x

localhost:8080/eboutique/adminProd/index

[Logout](#)

ID Produit
Désignation
Categorie ▾
Description
Prix
Quantité
Sélectionner?
Fichier Aucun fichier choisi

ID	Désignation	Description	Categorie	Prix	Quantité	Selected	Photo		
3	HP546789	aaaaaaaaaa	Ordinateurs	9000.0	6	true		Supp	Edit

Vue : produits.jsp

localhost:8080/eboutique x localhost:8080/eboutique/adminProd/saveProd

[Logout](#)

ID Produit la taille doit être entre 4 et 15
ne peut pas être vide.

Désignation la taille doit être entre 4 et 15
ne peut pas être vide.

Catégorie

Description

Prix

Quantité

Sélectionner?

Photo Aucun fichier choisi

ID	Désignation	Description	Catégorie	Prix	Quantité	Selected	Photo
3	HP540789	aaaaaaaaaa	Ordinateurs	9000.0	0	true	 Supp Edit

Vue : produits.jsp

localhost8080/eboutique x
localhost8080/eboutique/adminProd/editProd?idProd=3

Logout

ID Produit: 3
Désignation: HP546789
Catégorie: Ordinateurs
Description:aaaaaaaaaa
Prix: 9000.0
Quantité: 6
Sélectionner?

Photo:  Choisissez un fichier Aucun fichier choisi

Save

ID	Désignation	Description	Catégorie	Prix	Quantité	Selected	Photo
3	HP546789	aaaaaaaaaa	Ordinateurs	9000.0	6	true	 Supp Edit

Vue : produits.jsp

```
<%@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<%@taglib uri="http://www.springframework.org/tags/form" prefix="f"%>
<head>
    <link rel="stylesheet" type="text/css"
        href="<%=request.getContextPath()%>/resources/css/style1.css">
</head>
<a href="
```

Vue : produits.jsp

```
<tr>

    <td>Désignation</td>
    <td><f:input path="designation"/></td>
    <td><f:errors path="designation" cssClass="errors"></f:errors></td>
</tr>

<tr>

    <td>Cétegorie</td>
    <td><f:select path="categorie.idcategorie" items="${categories}"
itemValue="idcategorie" itemLabel="nomCategorie"></f:select></td>
    <td><f:errors path="designation" cssClass="errors"></f:errors></td>
</tr>

<tr>

    <td>Description</td>
    <td><f:textarea path="description"/></td>
    <td><f:errors path="description" cssClass="errors"></f:errors></td>
</tr>
```

Vue : produits.jsp

```
<tr>

    <td>Prix</td>
    <td><f:input path="prix"/></td>
    <td><f:errors path="prix" cssClass="errors"></f:errors></td>
</tr>

<tr>

    <td>Quantité</td>
    <td><f:input path="quantite"/></td>
    <td><f:errors path="quantite" cssClass="errors"></f:errors></td>
</tr>

<tr>

    <td>Sélectionner?</td>
    <td><f:checkbox path="selected"/></td>
    <td><f:errors path="selected" cssClass="errors"></f:errors></td>
</tr>
```

Vue : produits.jsp

```
<tr>

    <td>Photo</td>

    <td>
        <c:if test="${produit.idProduit!=null}">
            
        </c:if>
    </td>

    <td>
        <input type="file" name="file"/>
    </td>

</tr>
<tr>
    <td><input type="submit" value="Save"></td>
</tr>
</table>
</f:form>
</div>
```

Vue : produits.jsp

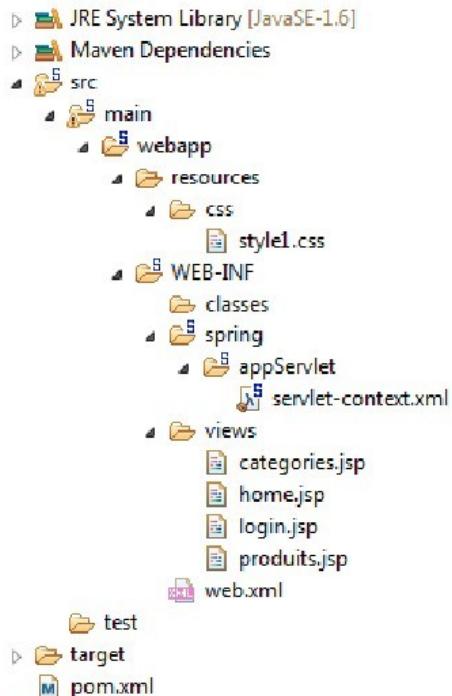
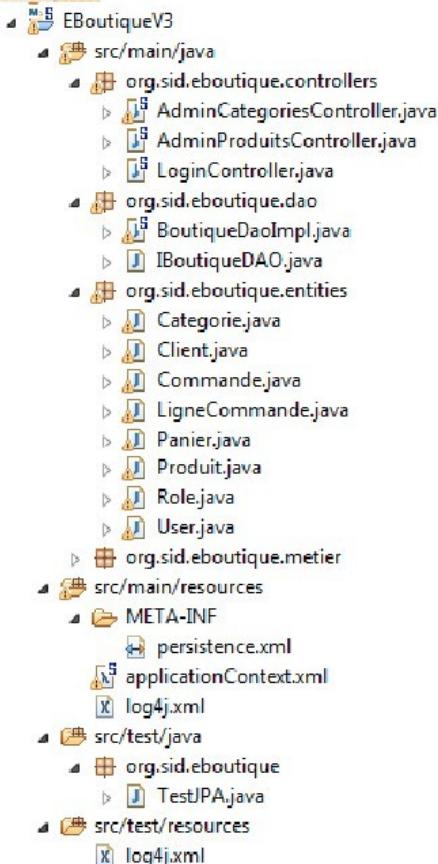
```
<div id="tabProduits" class="cadre">
    <table class="tab1">
        <tr>
            <th>ID</th><th>Désignation</th><th>Description</th>
            <th>catégorie</th><th>Prix</th><th>Quantité</th><th>Selected</th>
            <th>Photo</th>
        </tr>
        <c:forEach items="${produits}" var="p">
            <tr>
                <td>${p.idProduit }</td><td>${p.designation }</td>
                <td>${p.description }</td><td>${p.categorie.nomCategorie }</td>
                <td>${p.prix }</td><td>${p.quantite }</td> <td>${p.selected }</td>
                <td></td>
                <td><a href="suppProd?idProd=${p.idProduit }">Supp</a></td>
                <td><a href="editProd?idProd=${p.idProduit }">Edit</a></td>
            </tr>
        </c:forEach>
    </table>
</div>
```

Feuille de style CSS :

/ EBoutiqueV2/src/main/webapp/resources/css/style.css/EBoutiqueV2/src/main/webapp/resources/css/style.css

```
div.cadre{  
    border: 1px dotted gray;  
    border-radius:10px;  
    margin: 10px;  
    padding: 10px;  
}  
.tabStyle1 th{  
    border: 1px dotted gray;  
    background: pink;  
    padding: 5px;  
}  
.tabStyle1 td{  
    border: 1px dotted gray;  
    background: white;  
    padding: 5px;  
}
```

Structure du projet





- SPRING SECURITY

Maven Dependencies

```
<!-- Spring Security-->
<dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-core</artifactId>
    <version>3.2.0.RELEASE</version>
</dependency>
<dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-config</artifactId>
    <version>3.2.0.RELEASE</version>
</dependency>
<dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-web</artifactId>
    <version>3.2.0.RELEASE</version>
</dependency>
```

web.xml

Déclarer le filtre DelegatingFilterProxy dans le fichier web.xml

Toutes les requêtes HTTP passent par ce filtre.

Le nom du filtre est : **springSecurityFilterChain**

Ce nom devrait correspondre au nom d'un bean spring qui sera déployé par ContextLoaderListener et qui contient les règles de sécurité à exécuter.

```
<!-- Spring Security -->
<filter>
    <filter-name>springSecurityFilterChain</filter-name>
    <filter-
        class>org.springframework.web.filter.DelegatingFilterProxy</filter-class>
</filter>
<filter-mapping>
    <filter-name>springSecurityFilterChain</filter-name>
    <url-pattern>/*</url-pattern>
</filter-mapping>
```

Base de données

Créer deux tables :

- Users : qui contient les utilisateurs autorisés à accéder à l’application
- Role : qui contient les rôles de chaque utilisateur

user_id	actived	password	user_name
1	1	c00cf25ad42683b3df679c61f42c6bda	admin1
2	1	c84258e9c39059a89ab77d846ddab909	admin2

idRole	roleName	user_id
1	ROLE_ADMIN_CAT	1
2	ROLE_ADMIN_PROD	1
3	ROLE_ADMIN_PROD	2

applicationContext.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:tx="http://www.springframework.org/schema/tx"
       xmlns:context="http://www.springframework.org/schema/context"
       xmlns:s="http://www.springframework.org/schema/security"
       xsi:schemaLocation="http://www.springframework.org/schema/security
                           http://www.springframework.org/schema/security/spring-security-3.2.xsd
                           http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd
                           http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx-3.2.xsd
                           http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-3.2.xsd">

    <bean id="datasource" class="org.springframework.jdbc.datasource.DriverManagerDataSource">
        <property name="driverClassName" value="com.mysql.jdbc.Driver"></property>
        <property name="url" value="jdbc:mysql://localhost:3306/sid_eboutique"></property>
        <property name="username" value="root"></property>
        <property name="password" value=""></property>
    </bean>
```

applicationContext.xml

```
<s:http>
    <s:intercept-url pattern="/produits/**" access="ROLE_ADMIN_PROD"/>
    <s:intercept-url pattern="/categories/**" access="ROLE_ADMIN_CAT"/>
    <s:form-login login-page="/login" default-target-url="/produits/index"
        authentication-failure-url="/login" />
    <s:logout logout-success-url="/logout" />
</s:http>
<s:authentication-manager>
    <s:authentication-provider>
        <s:password-encoder hash="md5"></s:password-encoder>
        <s:jdbc-user-service data-source-ref="datasource"
            users-by-username-query="select user_name,password, activated
                from users where user_name=?"
            authorities-by-username-query="select u.user_name, r.roleName from users u, role r
                where u.user_id = r.user_id and u.user_name =? " />
        <!--
        <s:user-service>
            <s:user name="admin1" password="admin1" authorities="ROLE_ADMIN_PROD"/>
            <s:user name="admin2" authorities="ROLE_ADMIN_CAT,ROLE_ADMIN_PROD" password="admin2" />
        </s:user-service>
        -->
    </s:authentication-provider>
</s:authentication-manager>
</beans>
```

Contrôleur

```
package org.ensembl.sid.controllers;

import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;

@Controller
public class LoginController {
    @RequestMapping("/login")
    public String login(){
        return "login";
    }
    @RequestMapping("/logout")
    public String login(){
        return "login";
    }
}
```

login.jsp

```
<form action="j_spring_security_check" method="post">
  <table>
    <tr>
      <td>Login</td>
      <td><input type="text" name="j_username"></td>
    </tr>
    <tr>
      <td>Pass word</td><td>
      <input type="password" name="j_password"></td>
    </tr>
    <tr>
      <td><input type="submit" value="Login"></td>
    </tr>
  </table>
</form>
```

Lien Logout

```
<a href=<c:url value="/j_spring_security_logout" />" >  
Logout</a>
```