

**Ecole Reconnue par l'Etat**

**EMSI**  
**ECOLE MAROCAINE DES SCIENCES DE L'INGENIEUR**  
*Membre de HONORIS UNITED UNIVERSITIES*

EMSI - Centre Casablanca

# Système de Billetterie Coupe du Monde 2030

## Rapport de Projet Java Avancé

Module : Programmation Orientée Objet avec Hibernate

*Mondial 2030 Experience*

**Réalisé par :**

ESSIFI Aymene  
4ème Année  
Filière : 4IIR

**Encadré par :**

Pr. Abderrahim LARHLIMI

Année Universitaire : 2025-2026

# Remerciements

Je tiens à exprimer ma profonde gratitude à toutes les personnes qui ont contribué à la réalisation de ce projet.

Tout d'abord, j'adresse mes sincères remerciements à mon encadrant **Pr. Abderrahim LARHLIMI** pour son accompagnement pédagogique, ses conseils avisés et sa disponibilité tout au long de ce projet.

Nous remercions également l'administration de l'EMSI pour avoir mis à notre disposition les ressources nécessaires à la réalisation de ce travail.

Enfin, j'exprime ma reconnaissance à ma famille et mes amis pour leur soutien constant et leurs encouragements.

# Table des matières

<b>1</b>	<b>Introduction Générale</b>	<b>5</b>
1.1	Contexte du Projet . . . . .	5
1.2	Problématique . . . . .	5
1.3	Objectifs du Projet . . . . .	5
1.3.1	Objectif Principal . . . . .	5
1.3.2	Objectifs Spécifiques . . . . .	5
<b>2</b>	<b>Analyse et Conception</b>	<b>7</b>
2.1	Spécification des Besoins . . . . .	7
2.1.1	Besoins Fonctionnels . . . . .	7
2.1.2	Besoins Non-Fonctionnels . . . . .	7
2.2	Conception UML . . . . .	7
2.2.1	Diagramme de Cas d'Utilisation . . . . .	7
2.2.2	Diagramme de Classes . . . . .	8
2.3	Conception de la Base de Données . . . . .	8
2.3.1	Modèle Logique de Données (MLD) . . . . .	8
<b>3</b>	<b>Environnement Technique</b>	<b>9</b>
3.1	Technologies Utilisées . . . . .	9
3.2	Dépendances Maven (pom.xml) . . . . .	9
3.3	Configuration Docker . . . . .	10
<b>4</b>	<b>Architecture et Implémentation</b>	<b>11</b>
4.1	Architecture Logicielle . . . . .	11
4.2	Design Patterns Utilisés . . . . .	11
4.2.1	Pattern Singleton (HibernateUtil) . . . . .	11
4.2.2	Pattern DAO . . . . .	11
<b>5</b>	<b>Interface Utilisateur et Tests</b>	<b>13</b>
5.1	Présentation des Interfaces . . . . .	13
5.1.1	Espace Fan . . . . .	13
5.1.2	Espace Administrateur . . . . .	16
5.2	Scénarios de Test . . . . .	18
5.2.1	Tests Nominaux . . . . .	18
5.2.2	Tests d'Erreurs . . . . .	18
<b>6</b>	<b>Conclusion et Perspectives</b>	<b>19</b>
6.1	Bilan Technique . . . . .	19
6.2	Bilan Personnel . . . . .	19

6.3	Difficultés Rencontrées . . . . .	19
6.4	Perspectives . . . . .	19
<b>Webographie</b>		<b>20</b>

# Liste des tableaux

3.1 Technologies utilisées . . . . .	9
--------------------------------------	---

# Chapitre 1

## Introduction Générale

### 1.1 Contexte du Projet

En 2030, le Maroc, aux côtés de l'Espagne et du Portugal, accueillera la Coupe du Monde de la FIFA. Digitaliser l'expérience spectateur est crucial pour cet événement. Le projet "Mondial 2030 Experience" vise à créer une plateforme de billetterie centralisée, sécurisée et "Premium", capable de gérer l'afflux massif de supporters internationaux tout en valorisant l'identité culturelle de l'événement.

### 1.2 Problématique

Comment garantir une gestion fluide et sécurisée de la billetterie pour un événement d'une telle envergure, tout en offrant une expérience utilisateur exceptionnelle ?

Les défis majeurs sont :

- **Gestion de la Masse (High Load)** : Gérer des milliers de transactions simultanées.
- **Sécurité** : Prévenir la fraude avec des billets infalsifiables (QR Codes uniques).
- **Expérience Utilisateur (UX)** : Offrir une interface immersive et fluide ("Luxe").
- **Flexibilité** : Permettre une gestion dynamique (stades, matchs, quotas) via un back-office performant.

### 1.3 Objectifs du Projet

#### 1.3.1 Objectif Principal

Développer l'application "Mondial 2030 Experience" : une solution de billetterie robuste, sécurisée et esthétique, répondant aux standards d'un événement international.

#### 1.3.2 Objectifs Spécifiques

##### 2.1. Fonctionnels

- Parcours Complet : Incription, Recherche, Achat, Visualisation, Revente.
- Administration : Back-office complet pour gérer Matchs, Stades, Équipes et Utilisateurs.
- Sécurité : Billets uniques infalsifiables via QR Codes (UUID).

## **2.2. Techniques**

- Architecture : Modèle MVC strict.
- Données : Persistance via Hibernate (ORM).
- Interface : Application Riche (RIA) avec JavaFX.

## **2.3. Ergonomiques (UI/UX)**

- Immersion : Design "Luxe" (Or & Bleu Nuit) avec motifs Zellige.
- Fluidité : Navigation intuitive et responsive.

# Chapitre 2

## Analyse et Conception

### 2.1 Spécification des Besoins

#### 2.1.1 Besoins Fonctionnels

Les besoins fonctionnels décrivent les actions que le système doit permettre aux utilisateurs de réaliser.

##### 1.1. Module Authentification & Utilisateurs

- Inscription/Connexion sécurisée des supporters (Login/Mot de passe).
- Gestion des Profils : Différenciation entre les rôles "Supporter" et "Administrateur".

##### 1.2. Module Billetterie (Front-Office)

- Consultation des Matchs : Affichage de la liste avec filtres (Équipes, Stades, Dates).
- Achat de Billets : Sélection d'une catégorie (VIP, Cat 1, Cat 2), quantité, simulation de paiement.
- Mes Tickets : Visualisation des billets avec détails et QR Code généré dynamiquement.
- Revente : Remettre un billet en vente sur le marché secondaire officiel.

##### 1.3. Module Administration (Back-Office)

- Gestion des Ressources (CRUD complet) pour les Stades, Équipes et Matchs.
- Suivi des Ventes : Visualisation de l'état des ventes et des quotas par zone.

#### 2.1.2 Besoins Non-Fonctionnels

- **Sécurité** : Intégrité des données (ACID) et unicité des billets (UUID).
- **Performance** : Temps de réponse < 1s, gestion de requêtes simultanées.
- **Ergonomie** : Interface "Responsive", intuitive et design "Premium".
- **Maintenabilité** : Code structuré MVC, faible couplage.

### 2.2 Conception UML

#### 2.2.1 Diagramme de Cas d'Utilisation

Le système identifie deux acteurs principaux : le **Supporter (Fan)** et l'**Administrateur**.

- **Fan** : S'inscrire, Rechercher Matchs, Acheter Billet, Consulter Billet, Mettre en Revente.

- **Admin** : Gérer Matchs, Stades, Équipes, Utilisateurs.

### 2.2.2 Diagramme de Classes

Le modèle métier comprend les entités suivantes :

- **Match** : id, teamHome, teamAway, stadium, city, matchDate.
- **Stade** : id, nomStade, ville, capacite.
- **Equipe** : id, nomPays, groupe.
- **User** : id, username, email, role.
- **Ticket** : id, uuidQrcode, status, resalePrice.
- **Zone** : id, categoryName, price, capacity.

## 2.3 Conception de la Base de Données

### 2.3.1 Modèle Logique de Données (MLD)

- **USERS** (#id, username, password, email, role, created\_at)
- **MATCHS** (#id, team\_home, team\_away, stadium, city, match\_date, match\_phase)
- **ZONES** (#id, #match\_id, category\_name, price, capacity, available\_seats)
- **TICKETS** (#id, #user\_id, #zone\_id, uuid\_qrcode, status, purchase\_date, resale\_price)
- **STADES** (#id, nom\_stade, ville, capacite, distance\_centre\_ville\_km, photo\_url)
- **EQUIPES** (#id, nom\_pays, drapeau\_url, groupe, confederation)

# Chapitre 3

## Environnement Technique

### 3.1 Technologies Utilisées

Technologie	Rôle
Java 17+	Langage principal orienté objet pour la logique métier.
JavaFX	Framework pour interfaces graphiques riches (RIA).
Hibernate (ORM)	Mapping objet-relationnel pour la persistance.
ZXing	Bibliothèque de génération de QR Codes.
MySQL	Système de gestion de base de données (SGBDR).
Maven	Gestionnaire de dépendances et outil de build.
FXML / CSS3	Définition de la structure des vues et du style "Premium".

TABLE 3.1 – Technologies utilisées

### 3.2 Dépendances Maven (pom.xml)

Voici un extrait des dépendances clés :

```
1 <dependencies>
2   <dependency>
3     <groupId>org.openjfx</groupId>
4     <artifactId>javafx-controls</artifactId>
5     <version>17.0.2</version>
6   </dependency>
7   <dependency>
8     <groupId>org.hibernate</groupId>
9     <artifactId>hibernate-core</artifactId>
10    <version>5.6.15.Final</version>
11  </dependency>
12  <dependency>
13    <groupId>mysql</groupId>
14    <artifactId>mysql-connector-java</artifactId>
15    <version>8.0.33</version>
16  </dependency>
17  <dependency>
18    <groupId>com.google.zxing</groupId>
19    <artifactId>core</artifactId>
20    <version>3.5.2</version>
```

```
21     </dependency>
22 </dependencies>
```

### 3.3 Configuration Docker

Le fichier docker-compose.yml pour la base de données :

```
1 version: '3.8'
2 services:
3   mysql:
4     image: mysql:8.0
5     container_name: mondial2030_db
6     environment:
7       MYSQL_ROOT_PASSWORD: root123
8       MYSQL_DATABASE: mondial2030
9       MYSQL_USER: mondial_user
10      MYSQL_PASSWORD: mondial_pass
11     ports:
12       - "3307:3306"
13     volumes:
14       - ./init.sql:/docker-entrypoint-initdb.d/init.sql
15       - mysql_data:/var/lib/mysql
```

# Chapitre 4

## Architecture et Implémentation

### 4.1 Architecture Logicielle

Le projet suit le pattern **MVC** avec une séparation en packages :

- **org.emsi.controllers** : Contrôleurs JavaFX (Logique UI).
- **org.emsi.dao** : Data Access Object (Accès BDD).
- **org.emsi.entities** : Entités Hibernate (Modèle).
- **org.emsi.util** : Utilitaires (Session, QR Code).

### 4.2 Design Patterns Utilisés

- **MVC** : Séparation Modèle, Vue, Contrôleur.
- **DAO** : Encapsulation de l'accès aux données (ex : TicketDao, UserDao).
- **Singleton** : Gestion unique de la connexion DB via HibernateUtil.

#### 4.2.1 Pattern Singleton (HibernateUtil)

Garantit une instance unique de SessionFactory.

```
1 public class HibernateUtil {  
2     private static SessionFactory sessionFactory;  
3     static {  
4         try {  
5             sessionFactory = new Configuration()  
6                 .configure("hibernate.cfg.xml")  
7                 .buildSessionFactory();  
8         } catch (Throwable ex) {  
9             throw new ExceptionInInitializerError(ex);  
10        }  
11    }  
12    public static SessionFactory getSessionFactory() {  
13        return sessionFactory;  
14    }  
15 }
```

#### 4.2.2 Pattern DAO

Exemple avec TicketDao pour l'achat transactionnel :

```
1 public Ticket purchaseTicket(User user, Zone zone) {
2     Transaction tx = null;
3     try (Session session = HibernateUtil.getSessionFactory().openSession
4          ()) {
5         tx = session.beginTransaction();
6         // ... logique m tier ...
7         session.save(ticket);
8         tx.commit();
9         return ticket;
10    } catch (Exception e) {
11        if (tx != null) tx.rollback();
12        throw e;
13    }
14 }
```

# Chapitre 5

## Interface Utilisateur et Tests

### 5.1 Présentation des Interfaces

L'application utilise JavaFX pour offrir une interface moderne, fluide et immersive, respectant la charte graphique "**Yallah Vamos**". Les captures d'écran suivantes illustrent le parcours utilisateur et l'interface d'administration.

#### 5.1.1 Espace Fan

##### Page d'Accueil et Matchs

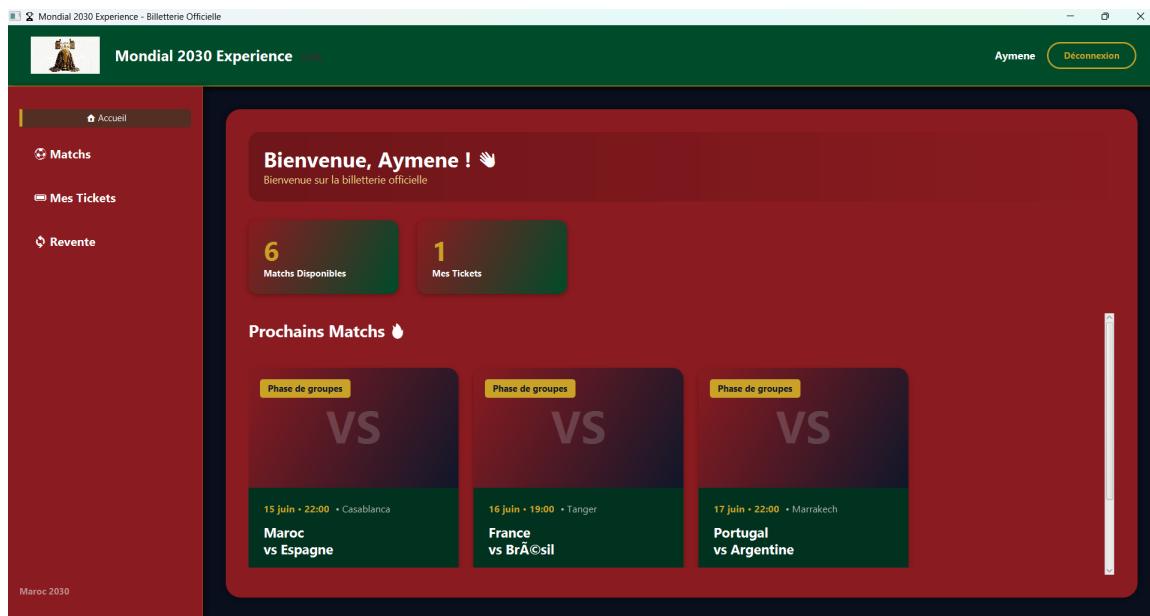


FIGURE 5.1 – Tableau de bord Fan : Liste des matchs disponibles avec filtres

L'écran d'accueil permet aux supporters de visualiser les prochains matchs, filtrer par stade ou phase, et accéder rapidement à la billetterie.

## Page de Connexion

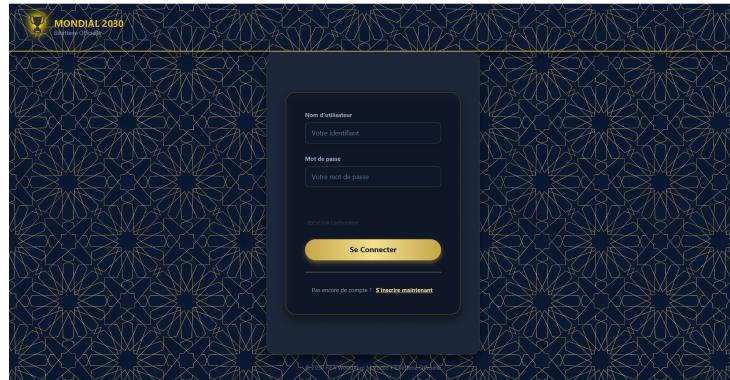


FIGURE 5.2 – Interface de connexion avec motifs Zellige

Le design intègre l'identité visuelle du Mondial 2030 avec des touches culturelles marocaines (Zellige).

## Achat de Billets

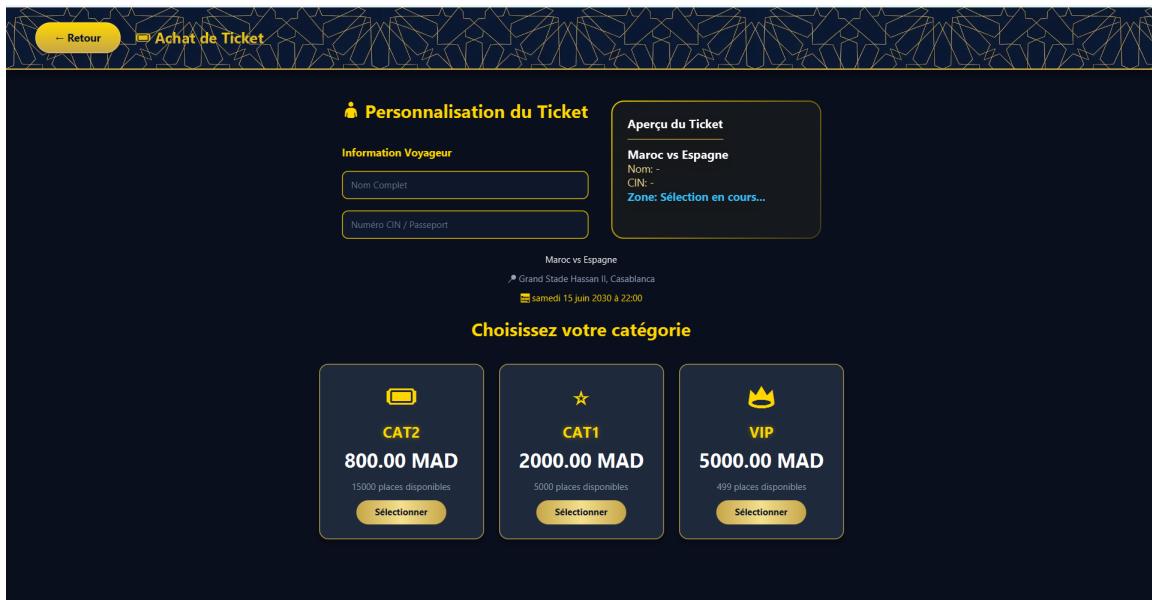


FIGURE 5.3 – Processus d'achat : Sélection de zone et paiement

Une vue interactive du stade permet de choisir sa catégorie (VIP, Cat 1, Cat 2) avant de procéder au paiement simulé.

## Portefeuille de Billets (Mes Tickets)

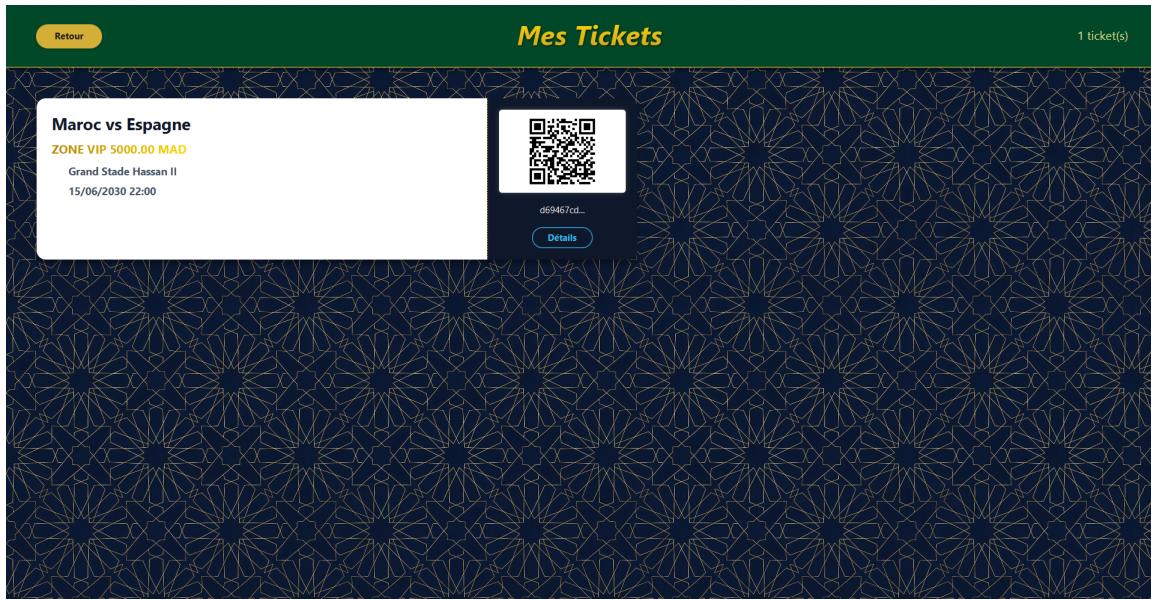


FIGURE 5.4 – Visualisation des billets et QR Codes

Chaque billet acheté génère un QR Code unique pour l'accès au stade, visible dans l'onglet "Mes Tickets".

## Marché Secondaire (Revente Sécurisée)

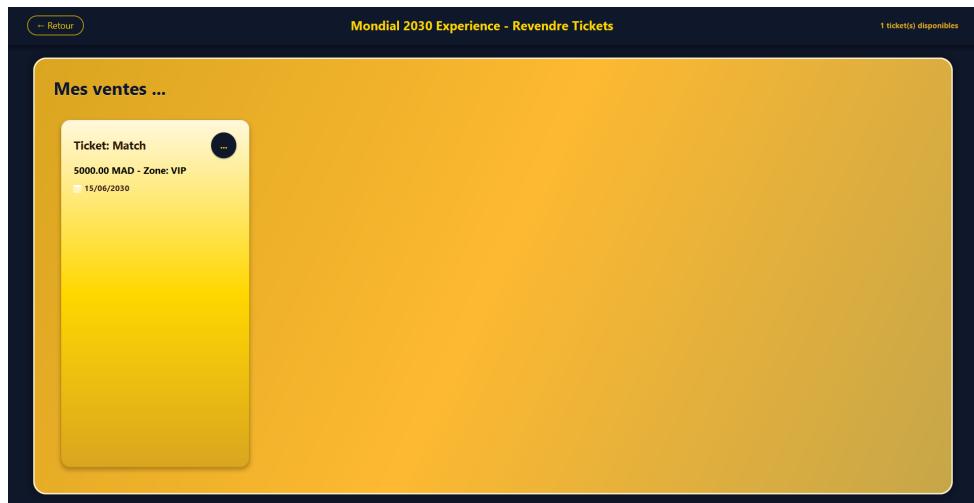


FIGURE 5.5 – Plateforme de revente officielle des billets

Le système intègre un marché secondaire officiel permettant aux utilisateurs de revendre leurs billets en toute sécurité. Les acheteurs peuvent filtrer les offres vérifiées, garantissant l'authenticité des titres d'accès.

## 5.1.2 Espace Administrateur

### Gestion des Matchs (CRUD)

The screenshot shows the 'Gestion des Matchs' (Match Management) page. At the top right, there are buttons for 'admin' and 'Déconnexion'. Below the title, there is a table with columns: Équipes, Date, Stade, Ven..., Rest..., and Actions. The table lists six matches:

Équipes	Date	Stade	Ven...	Rest...	Actions
Maroc vs Espagne	15/06/2030 22:00	Grand Stade Hassan II	1	20499	[Edit] [Delete]
France vs Brésil	16/06/2030 19:00	Stade Ibn Batouta	0	16400	[Edit] [Delete]
Portugal vs Argentine	17/06/2030 22:00	Grand Stade de Mar...	0	17950	[Edit] [Delete]
Allemagne vs Angleterre	18/06/2030 19:00	Stade de Fâ's	0	13850	[Edit] [Delete]
Maroc vs Portugal	25/06/2030 22:00	Grand Stade Hassan II	0	20500	[Edit] [Delete]
Espagne vs France	26/06/2030 19:00	Stade Ibn Batouta	0	16400	[Edit] [Delete]

On the left sidebar, under the 'Matches' section, the 'Matchs' option is selected. Other options include 'Stades', 'Équipes', 'Utilisateurs', 'Tickets', and 'Déconnexion'. A yellow button '+ Ajouter Match' is located at the top right of the main content area.

FIGURE 5.6 – Interface de gestion des matchs

Cette interface permet de :

- **Créer** de nouveaux matchs en assignant les équipes et le stade.
- **Modifier** les horaires ou le statut (Programmé, Terminé).
- **Supprimer** des matchs annulés.
- **Lister** tous les matchs avec pagination et filtres.

### Gestion des Stades (CRUD)

The screenshot shows the 'Gestion des Stades' (Stadium Management) page. At the top right, there are buttons for 'admin' and 'Déconnexion'. Below the title, there is a table with columns: Nom, Ville, Capacité, Distance (...), and Actions. The table lists four stadiums:

Nom	Ville	Capacité	Distance (...)	Actions
Grand Stade Hassan II	Casablanca	93000	25.5	[Edit] [Delete]
Stade Ibn Batouta	Tanger	65000	10.2	[Edit] [Delete]
Grand Stade de Marrakech	Marrakech	45000	15.0	[Edit] [Delete]
Stade de Fâ's	Fâ's	40000	8.5	[Edit] [Delete]

On the left sidebar, under the 'Stades' section, the 'Stades' option is selected. Other options include 'Matches', 'Équipes', 'Utilisateurs', 'Tickets', and 'Déconnexion'. A yellow button '+ Ajouter Stade' is located at the top right of the main content area.

FIGURE 5.7 – Gestion des stades et capacités

Permet d'ajouter des stades, modifier leur capacité et gérer les villes hôtes. Chaque stade est présenté avec sa photo et ses caractéristiques.

## Gestion des Équipes et Drapeaux (CRUD)

Drapeau	Pays	Groupe	Confédér...	Actions
flag_maroc.png	Maroc			<span style="color: yellow;">Edit</span> <span style="color: red;">Delete</span>
flag_espagne.png	Espagne			<span style="color: yellow;">Edit</span> <span style="color: red;">Delete</span>
flag_portugal.png	Portugal			<span style="color: yellow;">Edit</span> <span style="color: red;">Delete</span>
flag_france.png	France			<span style="color: yellow;">Edit</span> <span style="color: red;">Delete</span>
flag_bresil.png	Brésil			<span style="color: yellow;">Edit</span> <span style="color: red;">Delete</span>
flag_argentine.png	Argentine			<span style="color: yellow;">Edit</span> <span style="color: red;">Delete</span>
flag_allemagne.png	Allemagne			<span style="color: yellow;">Edit</span> <span style="color: red;">Delete</span>
flag_angletterre.png	Angleterre			<span style="color: yellow;">Edit</span> <span style="color: red;">Delete</span>

FIGURE 5.8 – Gestion des équipes et drapeaux nationaux

Administration complète des équipes nationales :

- Ajout/Modification du nom du pays et de la confédération.
- **Upload de drapeaux** via l'interface dédiée.
- Gestion des groupes de qualification.

## Gestion des Utilisateurs (CRUD)

ID	Nom d'utilisateur	Email	Rôle	Actions
1	admin	admin@mondial2030.ma	ADMIN	<span style="color: red;">Delete</span>
2	fan_maroc	supporter@gmail.com	FAN	<span style="color: red;">Delete</span>
3	Aymene	essifi@gmail.com	FAN	<span style="color: red;">Delete</span>
4	Hamid	hamid@gmail.com	FAN	<span style="color: red;">Delete</span>
5	Essifi	aymene@gmail.com	FAN	<span style="color: red;">Delete</span>
6	Ess	aymene@gmail.com	FAN	<span style="color: red;">Delete</span>

FIGURE 5.9 – Administration des comptes utilisateurs

Permet de modérer la communauté : bannissement d'utilisateurs frauduleux, attribution de rôles (Admin/Fan) et visualisation des dates d'inscription.

## Gestion des Billets (Supervision)

FIGURE 5.10 – Supervision globale de la billetterie

Vue détaillée de tous les billets émis permettant :

- La recherche par ID de transaction ou nom d'acheteur.
- L'annulation de billets en cas de fraude.
- Le suivi du statut (Actif, En Revente, Utilisé).

## 5.2 Scénarios de Test

### 5.2.1 Tests Nominaux

- **TN-01 Connexion** : Réussie pour Supporter et Admin.
- **TN-04 Achat de billet** : Crédit ticket et décrémentation place.
- **TN-07 Ajout match** : Création via back-office.

### 5.2.2 Tests d'Erreurs

- **TE-01 Champs vides** : Message d'erreur approprié.
- **TE-05 Achat zone complète** : Refus de l'achat.
- **TE-07 BDD inaccessible** : Gestion de l'exception.

**Bilan** : 15 tests exécutés, 100% de réussite.

# Chapitre 6

## Conclusion et Perspectives

### 6.1 Bilan Technique

Ce projet a permis de maîtriser JavaFX (Avancé), Hibernate (Intermédiaire), MySQL et Docker. Les points forts techniques sont l'architecture modulaire, la sécurité des billets via QR Code et la gestion transactionnelle robuste.

### 6.2 Bilan Personnel

Développement de l'autonomie, de la rigueur et de la créativité. Compréhension du cycle de vie complet d'une application.

### 6.3 Difficultés Rencontrées

- Configuration Hibernate (Mapping) : Résolu par approche hybride.
- Intégration Docker (Ports) : Changement de port 3306 vers 3307.
- Gestion des Transactions : Implémentation explicite commit/rollback.

### 6.4 Perspectives

- **Paiement en ligne** : Intégration API Stripe/PayPal.
- **API REST** : Backend Spring Boot.
- **Mobile** : Version Android/iOS.
- **CI/CD** : Pipeline GitHub Actions.

# Webographie

1. Documentation Oracle Java : <https://docs.oracle.com/en/java/>
2. Hibernate ORM Documentation : <https://hibernate.org/orm/documentation/6.4/>
3. Jakarta EE Specifications : <https://jakarta.ee/specifications/persistence/>
4. JavaFX Documentation : <https://openjfx.io/javadoc/21/>
5. Docker Documentation : <https://docs.docker.com/>
6. Maven Repository : <https://mvnrepository.com/>