**Communauté Française de Belgique**

**Institut des Carrières Commerciales**

Ville de Bruxelles

Rue de la Fontaine, 4

1000 Bruxelles

**Projet Réservations  
Documentation technique**

Projet d’intégration développement

**Abdelrahim Ali**

2024 – 2025

Table des matières

[Méthodologie & techniques de développement 3](#_Toc206424731)

[Langage et frameworks 3](#_Toc206424732)

[Bibliothèques et outils principaux 3](#_Toc206424733)

[Logiciels utilisés (développement & déploiement) 3](#_Toc206424734)

[Méthodes de gestion de projet 4](#_Toc206424735)

[Fonctionnalités implémentées (avec URLs) 4](#_Toc206424736)

[Principales routes (Web & API) 6](#_Toc206424737)

[Développements futurs 8](#_Toc206424738)

[Gestion de la sécurité 8](#_Toc206424739)

[Procédures d’identification (profils) 8](#_Toc206424740)

[API consommée 8](#_Toc206424741)

[API produites (JSON) 8](#_Toc206424742)

[Flux RSS / Atom / JSON 8](#_Toc206424743)

[Export / Import de données 9](#_Toc206424744)

[Procédures de déploiement 9](#_Toc206424745)

[URLs 9](#_Toc206424746)

## Méthodologie & techniques de développement

Développement itératif avec gestion de versions Git (branche main). Stack Spring Boot (MVC, Data JPA, Security), moteur de templates Thymeleaf, migrations Flyway, base MySQL, Hébergement de l’app et de la BD sur la plateforme cloud Railway.

# Langage et frameworks

**Langage:** Java 17

**Spring Boot (parent):** 3.3.5

## Bibliothèques et outils principaux

|  |  |  |
| --- | --- | --- |
| Dépendance | Artifact | Version |
| com.rometools | rome | 1.18.0 |
| org.thymeleaf.extras | thymeleaf-extras-springsecurity6 | (gérée par Spring Boot) |
| com.opencsv | opencsv | 5.9 |
| org.apache.poi | poi-ooxml | 5.2.5 |
| com.github.librepdf | openpdf | 1.3.39 |
| com.stripe | stripe-java | 26.0.0 |
| org.springframework.boot | spring-boot-starter-actuator | (gérée par Spring Boot) |
| org.springframework.boot | spring-boot-starter-data-jpa | (gérée par Spring Boot) |
| org.springframework.boot | spring-boot-starter-thymeleaf | (gérée par Spring Boot) |
| org.springframework.boot | spring-boot-starter-data-jpa | (gérée par Spring Boot) |
| org.springframework.boot | spring-boot-starter | (gérée par Spring Boot) |
| org.springframework.boot | spring-boot-starter-thymeleaf | (gérée par Spring Boot) |
| nz.net.ultraq.thymeleaf | thymeleaf-layout-dialect | (gérée par Spring Boot) |
| org.springframework.boot | spring-boot-starter-security | (gérée par Spring Boot) |
| org.springframework.boot | spring-boot-starter-web | (gérée par Spring Boot) |
| org.flywaydb | flyway-core | (gérée par Spring Boot) |
| org.flywaydb | flyway-mysql | (gérée par Spring Boot) |
| org.thymeleaf.extras | thymeleaf-extras-springsecurity6 | (gérée par Spring Boot) |
| org.springframework.boot | spring-boot-devtools | (gérée par Spring Boot) |
| com.mysql | mysql-connector-j | (gérée par Spring Boot) |
| org.springframework.boot | spring-boot-starter-test | (gérée par Spring Boot) |

# Logiciels utilisés (développement & déploiement)

- IDE : IntelliJ IDEA / VS Code (au choix).

- Base de données : MySQL 8 ; outil : MySQL Workbench (exports/dumps).

- Build : Maven 3.9.9

- Serveur d’application : Spring Boot (Tomcat embarqué).

- Navigateurs : Chrome / Firefox pour les tests UI.

# Méthodes de gestion de projet

Gestion via Git (commits, push). Organisation simple : branche principale, messages de commits explicites.

# Fonctionnalités implémentées (avec URLs)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Fonctionnalité | Description | Méthode | URL | Accès |
| Accueil | Page d’accueil et navigation | GET | <http://localhost:8080/> | Public |
| Liste des représentations | Catalogue des représentations | GET | <http://localhost:8080/representations> | Authentifié |
| Détail représentation | Fiche avec places restantes et formulaire de réservation | GET | [http://localhost:8080/representations/{id}](http://localhost:8080/representations/%7Bid%7D) | Authentifié |
| Mon profil | Edition du profil + liste de mes réservations | GET | <http://localhost:8080/profile> | Authentifié |
| Mettre à jour le profil | Soumission du formulaire profil | POST | <http://localhost:8080/profile> | Authentifié |
| Réserver | Réserver un nombre de place d’une représentation | POST | [http://localhost:8080/representations/{id}](http://localhost:8080/representations/%7bid%7d) | Authentifié |
| Annuler une réservation | Annule si statut ≠ CANCELED | POST | [http://localhost:8080/profile/reservations/{id}/cancel](http://localhost:8080/profile/reservations/%7Bid%7D/cancel) | Authentifié |
| Checkout Stripe | Démarre le paiement pour une réservation | GET | <https://checkout.stripe.com/c/pay/cs_test_a17wqadfxRunOveOZXVt4eTxI2F87GSNnB8KvBb5aQTWensPOkFtZPRJZK#fidkdWxOYHwnPyd1blpxYHZxWjA0V3JQdUhXdk5zZ2JEfzFEV3VBcGc8fGRTQFNyZndwSG1zcUxJf2kxR3N8SDZWVWA3NEtJTzVdXTBIPDZwbFJpY2pyT2x1QD1KajIwREJ1VVN2SFFMcTJgNTVIXElNT1d%2FSycpJ2N3amhWYHdzYHcnP3F3cGApJ2lkfGpwcVF8dWAnPyd2bGtiaWBabHFgaCcpJ2BrZGdpYFVpZGZgbWppYWB3dic%2FcXdwYHgl> | Authentifié |
| Retour paiement succès | Confirme le paiement et redirige profil | GET | <http://localhost:8080/payments/success?session_id=...&rid=>... | Authentifié |
| Retour paiement annulé | Annulé côté Stripe, message flash | GET | <http://localhost:8080/profile> | Authentifié |
| Export réservations CSV | Export des réservations de l’utilisateur | GET | <http://localhost:8080/profile/exports/reservations.csv> | Authentifié |
| Export réservations XLSX | Export Excel. | GET | <http://localhost:8080/profile/exports/reservations.xlsx> | Authentifié |
| Export réservations PDF | Export PDF. | GET | <http://localhost:8080/profile/exports/reservations.pdf> | Authentifié |
| Import (ADMIN) - page | Interface d’import de représen  -tations. | GET | <http://localhost:8080/admin/import> | ADMIN |
| Télécharger template CSV | Modèle pour l’import. | GET | <http://localhost:8080/admin/import/representations/template.csv> | ADMIN |
| Importer représentations | Envoi du fichier CSV/XLSX. Possibilités de Dry-run (valider sans écrire) | POST | <http://localhost:8080/admin/import/representations> | ADMIN |
| Flux RSS | 20 dernières représentations (RSS 2.0). | GET | <http://localhost:8080/feeds/representations.rss> | Authentifié |
| Flux Atom | 20 dernières représentations (Atom 1.0). | GET | <http://localhost:8080/feeds/representations.atom> | Authentifié |
| Flux JSON | 20 dernières représentations (JSON). | GET | <http://localhost:8080/feeds/representations.json> | Authentifié |
| Liste des artiste | Afficher la liste des artistes | GET | <http://localhost:8080/artists> | Authentifié |
| Détails d’un artiste | Voir les détails d’un artiste | GET | <http://localhost:8080/artists/1> | ADMIN |
| Crée un artiste | / | PUT | <http://localhost:8080/artists/create> | ADMIN |
| Supprimer un artist | / | Delete | <http://localhost:8080/artists> | ADMIN |
| Modifier l’affiliation à une troupe | / | Post | <http://localhost:8080/artists/1> | ADMIN |
| API JSON – artistes (liste) | Endpoint REST de lecture. | GET | <http://localhost:8080/api/artists> | Authentifié |
| API JSON – artiste (id) | Détail d’un artiste. | GET | [http://localhost:8080/api/artists/{id}](http://localhost:8080/api/artists/%7Bid%7D) | Authentifié |
| API JSON – artistes (CRUD ADMIN) | Création / mise à jour / suppression. | POST/PUT/DELETE | [http://localhost:8080/api/admin/artists[/{id}](http://localhost:8080/api/admin/artists%5B/%7Bid%7D)] | ADMIN |
| Liste des shows | Catalogue de spectacles avec filtres. | GET | <http://localhost:8080/shows> | Authentifié |
| Liste des shows paginé | Catalogue de spectacles paginé | GET | <http://localhost:8080/shows?page=0>  [http://localhost:8080/shows?page=1](http://localhost:8080/shows?page=10) | Authentifié |
| Détail d’un show | Fiche show. | GET | [http://localhost:8080/shows/{id}](http://localhost:8080/shows/%7Bid%7D) | Authentifié |
| Shows par titre | Filtrer par titre. | GET | [http://localhost:8080/shows/artist/{id}](http://localhost:8080/shows/artist/%7Bid%7D) | Authentifié |
| Shows par mot clé | Filtrer par mot clé. | GET | <http://localhost:8080/shows?search=société> | Authentifié |
| Shows triés par titre croissant/décroissant | Trier par tire croissant ou décroissant. | GET | <http://localhost:8080/shows?sort=title&dir=asc>  <http://localhost:8080/shows?sort=title&dir=desc> | Authentifié |
| Shows triés par prix croissant décroissant | Trier par prix croissant ou décroissant. | GET | <http://localhost:8080/shows?sort=price&dir=asc>  <http://localhost:8080/shows?sort=price&dir=desc> | Authentifié |
| Créer un tag | Créer et assigner un tag à un show | POST | http://localhost:8080/shows/{id} | Admin |
| Exclure tag (exemple) | Exemple de filtrage par tag. | GET | [http://localhost:8080/shows/exclude-tag/{tag}](http://localhost:8080/shows/exclude-tag/%7Btag%7D) | Authentifié |
| Artistes – liste | Catalogue des artistes. | GET | <http://localhost:8080/artists> | Authentifié |
| Artistes – détail | Fiche artiste. | GET | [http://localhost:8080/artists/{id}](http://localhost:8080/artists/%7Bid%7D) | Authentifié |
| Authentification – login | Formulaire de connexion. | GET | <http://localhost:8080/login> | Public |
| Authentification – logout | Déconnexion (CSRF). | POST | <http://localhost:8080/login?logout> | Authentifié |
| Inscription | Création de compte. | GET/POST | <http://localhost:8080/register> | Public |

## 

## Principales routes (Web & API)

|  |  |  |
| --- | --- | --- |
| Méthode | Chemin | Contrôleur |
| GET | / | HomeController.java |
| POST | /admin/import/representations | AdminImportController.java |
| GET | /admin/import/representations/template.csv | AdminImportController.java |
| POST | /api/admin/artists | ArtistApiController.java |
| DELETE | /api/admin/artists/{id} | ArtistApiController.java |
| PUT | /api/admin/artists/{id} | ArtistApiController.java |
| GET | /api/artists | ArtistApiController.java |
| GET | /api/artists/{id} | ArtistApiController.java |
| GET | /artists | ArtistController.java |
| GET | /artists/create | ArtistController.java |
| POST | /artists/create | ArtistController.java |
| DELETE | /artists/{id} | ArtistController.java |
| GET | /artists/{id} | ArtistController.java |
| GET | /artists/{id}/edit | ArtistController.java |
| PUT | /artists/{id}/edit | ArtistController.java |
| POST | /artists/{id}/troupe | ArtistController.java |
| GET | /feeds/representations.atom | FeedController.java |
| GET | /feeds/representations.json | FeedController.java |
| GET | /feeds/representations.rss | FeedController.java |
| GET | /localities | LocalityController.java |
| GET | /localities/{id} | LocalityController.java |
| GET | /login | AuthController.java |
| GET | /payments/cancel | PaymentController.java |
| GET | /payments/checkout | PaymentController.java |
| GET | /payments/success | PaymentController.java |
| POST | /payments/webhook | PaymentController.java |
| GET | /profile/exports/reservations.csv | ProfileExportController.java |
| GET | /profile/exports/reservations.pdf | ProfileExportController.java |
| GET | /profile/exports/reservations.xlsx | ProfileExportController.java |
| GET | /profile/reservations | ProfileReservationController.java |
| POST | /profile/reservations/{id}/cancel | ProfileReservationController.java |
| POST | /profile/reservations/{id}/paid | ProfileReservationController.java |
| GET | /register | AuthController.java |
| POST | /register | AuthController.java |
| GET | /representations | RepresentationController.java |
| GET | /representations/{id} | RepresentationController.java |
| POST | /representations/{id}/reserve | ReservationWebController.java |
| GET | /roles | RoleController.java |
| GET | /roles/{id} | RoleController.java |
| GET | /shows | ShowController.java |
| GET | /shows/exclude-tag/{tag} | ShowController.java |
| GET | /shows/{id} | ShowController.java |
| POST | /shows/{id}/add-tag | ShowController.java |
| GET | /types | TypeController.java |
| GET | /types/{id} | TypeController.java |

# Développements futurs

- Webhook Stripe robuste + idempotence + signature vérifiée.

- Slugs pour shows et pages détail SEO-friendly.

- Tests unitaires et tests MVC (profil, réservation).

- Notifications e-mail (confirmation de réservation/paiement).

# Gestion de la sécurité

Spring Security 6 : login formulaire `/login`, logout `/logout`, accès public aux pages statiques & `/`, protection CSRF activée.

Autorisations :

• Accès ADMIN requis pour `/admin/\*\*` (import et d’autres fonctionnalités).

• Paiement : `/payments/checkout` authentifié/

• Autres routes authentifiées par défaut (voir SecurityFilterChain).

# Procédures d’identification (profils)

Profils : ADMIN, MEMBER, AFFILIATE. Des utilisateurs de démonstration sont insérés via migrations (`users`, `roles`, `user\_role`).

Les mots de passe en base sont chiffrés (BCrypt) et non fournis en clair.

Procédure recommandée :

1) S'enregistrer via `/register`.

2) Promouvoir en ADMIN via SQL : INSERT INTO user\_role(user\_id, role\_id) VALUES(<id>,

# API consommée

Stripe Checkout (SDK `com.stripe:stripe-java:26.0.0`).

• Endpoint d’app : `/payments/checkout` → création d’une Session Checkout (mode=payment).

• Webhook : `/payments/webhook` → marque la réservation PAYÉE (événements checkout.session.completed).

Configuration : `stripe.secret` (clé secrète), `app.base-url` pour URLs de retour.

# API produites (JSON)

`GET /api/artists` → liste paginable d’artistes.

`GET /api/artists/{id}` → détail artiste.

`POST/PUT/DELETE /api/admin/artists` → CRUD (protégé ADMIN).

# Flux RSS / Atom / JSON

• RSS : `/feeds/representations.rss`

• Atom : `/feeds/representations.atom`

• JSON : `/feeds/representations.json`

# Export / Import de données

Export des réservations (profil) :

• CSV : `/profile/exports/reservations.csv`

• XLSX : `/profile/exports/reservations.xlsx`

• PDF : `/profile/exports/reservations.pdf`

Import admin des représentations :

• Page : `/admin/import`

• POST : `/admin/import/representations` ; template :

`/admin/import/representations/template.csv`

# Procédures de déploiement

Prérequis : JDK 17, Maven 3.9.9, MySQL 8 avec base `reservations`.

Configuration : `src/main/resources/application.properties` :

• `spring.datasource.url=jdbc:mysql://<host>:<port>/reservations`

• `spring.datasource.username`, `spring.datasource.password`

• `stripe.secret` (clé secrète Stripe).

Initialisation : Flyway migre le schéma au démarrage (`spring.flyway.baseline-on-migrate=true`).

Commande : `mvn -DskipTests package` puis `java -jar target/reservations-0.0.1-SNAPSHOT.jar`.

Serveur : Tomcat embarqué (port 8080).

# URLs

URL de développement (GitHub) : <https://github.com/aliabdelrahim/ReservationsProject.git>

URL de déploiement (prod) : https://reservationsproject-production.up.railway.app/

Instance locale par défaut : http://localhost:8080)