

Projet DevOps – Gestion d’Abonnements

1. Introduction

L’application **Gestion d’Abonnements** est une application permettant aux utilisateurs de centraliser, gérer et suivre leurs abonnements (streaming, sport, services numériques, etc.). Elle vise à améliorer la maîtrise des dépenses récurrentes et à éviter les oublis grâce à un suivi automatisé basé sur les dates.

La version **0.1.0** correspond à la première release fonctionnelle du projet. Elle met en œuvre les fonctionnalités essentielles attendues pour la Version 1 du projet.

2. Type d’application

Application Web avec un back-end Java. Une API REST est proposée de manière optionnelle pour l’accès aux données.

3. Fonctionnalités implémentées – Version 0.1.0

3.1. F1 : Gestion complète des abonnements (CRUD)

Cette fonctionnalité permet de gérer le cycle de vie complet d’un abonnement :

- Ajout d’un abonnement
- Modification d’un abonnement
- Suppression d’un abonnement
- Recherche d’abonnements

Les données sont sauvegardées automatiquement dans un fichier texte `abonnements.txt`, assurant la persistance entre deux exécutions de l’application.

3.1.1. API REST (optionnelle – Spark Java)

Une API REST minimale basée sur **Spark Java** permet d’interagir avec les abonnements :

```
GET    /api/abonnements
POST   /api/abonnements
PUT    /api/abonnements/:id
```

```
DELETE /api/abonnements/:id
```

Les données sont échangées au format JSON.

3.2. F2 : Système d'alertes

Un système d'alertes a été implémenté afin d'améliorer le suivi des abonnements :

- Détection automatique des abonnements expirés
- Alerte lorsque la date de fin approche (X jours avant expiration)
- Indication visuelle de l'état de l'abonnement (actif, expiré, proche de l'expiration)
- Calculs de dates réalisés à l'aide de l'API `java.time`

4. Interface utilisateur

La version 0.1.0 propose une interface simple et fonctionnelle :

- Affichage responsive
- Tableau de bord avec indicateurs principaux :
 - nombre total d'abonnements
 - abonnements actifs
 - abonnements expirés
- Import et export des données au format JSON
- Design léger inspiré du glassmorphism

5. Données manipulées

Les données gérées dans cette version incluent :

- Nom du service
- Dates de début et de fin
- Prix de l'abonnement
- Catégorie
- Date de dernière utilisation
- Statut de l'abonnement (actif / expiré)

Les données sont stockées localement dans un fichier texte et peuvent être importées ou exportées au format JSON.

6. Interconnexions techniques

- Front-end : HTML, CSS, JavaScript
- Back-end : Java
- API REST : Spark Java
- Persistance : fichier texte (`abonnements.txt`)

7. Structure du projet

Le projet suit une structure Maven standard :

```
.
├── pom.xml
├── abonnements.txt
├── docs/
│   └── rapport/
│       └── documentation_v0.1.adoc
├── src/
│   ├── main/
│   │   ├── java/
│   │   │   ├── com/
│   │   │   │   ├── example/
│   │   │   │   │   ├── abonnement/
│   │   │   │   │   │   ├── Abonnement.java
│   │   │   │   │   │   └── GestionAbonnements.java

```

L'architecture respecte les principes de la programmation orientée objet (POO).

8. Comment lancer le projet

8.1. Prérequis

- Java JDK 17
- Maven
- Un IDE (Visual Studio Code, IntelliJ IDEA, etc.)

8.2. Récupération du code

```
git clone <URL_DU_DEPOT>
cd Projet-Dev-Ops
```

8.3. Compilation

```
mvn clean package
```

8.4. Lancement de l'application

```
mvn exec:java -Dexec.mainClass=com.example.abonnement.GestionAbonnements
```

8.5. Lancement de l'API REST (optionnel)

```
mvn exec:java -Dexec.mainClass=com.projet.api.ApiServer
```

9. Limites de la version 0.1.0

- Fonctionnalités avancées (statistiques détaillées, chatbot, notifications, intégration bancaire) non incluses dans cette version
- Tests automatisés encore limités
- Persistance basée sur fichier texte (évolution possible vers base de données)

10. Perspectives (v0.2)

Les évolutions prévues pour la version suivante incluent :

- Amélioration de l'interface utilisateur
- Ajout de statistiques avancées
- Renforcement des tests (unitaires et intégration)
- Évolution de la persistance des données