Année universitaire : 2024 -2025
ING 1 Génie Logiciel
Fares Aloulou - Rayen Braiek

Application mobile de gestion des plannings d'examens et de surveillance : Rapport de solution

1. Introduction

Ce rapport présente la solution technique adoptée pour développer une application mobile permettant aux enseignants de consulter leur planning de surveillance et aux étudiants d'accéder à leur planning d'examens. Cette solution vise à améliorer la transparence, la communication et la gestion du temps durant la période des examens.

2. Objectifs de l'application mobile

- Offrir un accès rapide et sécurisé aux plannings.
- Faciliter la consultation depuis un smartphone.
- Permettre aux utilisateurs de recevoir des notifications en cas de changement.

3. Public cible

- **Enseignants** : pour consulter leurs créneaux de surveillance.
- **Étudiants** : pour visualiser leurs dates d'examens.

4. Fonctionnalités principales

Enseignant :

- Accès sécurisé par authentification.
- Affichage du planning de surveillance avec salle, heure, matière.
- Notification en cas de mise à jour du planning.

Étudiant :

- Accès sécurisé.
- Affichage du planning d'examen : matière, salle, date, surveillant.
- Notification en cas de modification.

5. Architecture de la solution

- **Frontend (Mobile App)** :
    - Framework : Flutter
    - Navigation fluide entre les pages
    - Interface responsive et simple
- **Backend (API REST - Spring Boot)** :
    - Gestion des utilisateurs, enseignants, étudiants, examens
    - Endpoints sécurisés (JWT Tokens)
    - Exposition des plannings à partir de la base

- **Base de données (PostgreSQL)** :
    - Tables `teacher`, `exam`, `student`, `exam_supervisors`, `niveau`, `notifications`

6. Détails de l'implémentation

. Authentification par JWT

L'application mobile utilise un système d'authentification sécurisé basé sur des **JSON Web Tokens (JWT)**. Ce mécanisme permet une authentification **stateless**, où aucun état n'est conservé côté serveur.

. Fonctionnement :

1. L'utilisateur (enseignant ou étudiant) saisit ses identifiants sur l'application mobile.
2. Ces identifiants sont envoyés via une requête `POST` vers l'endpoint `/login`.
3. En cas de succès, le serveur renvoie un **JWT signé**, contenant :
    - L'identifiant de l'utilisateur.
    - Son **rôle** (enseignant ou étudiant).
4. Ce **token est stocké localement** (dans le stockage sécurisé de l'application mobile).
5. Pour chaque appel API ultérieur, ce token est ajouté dans l'en-tête HTTP `Authorization: Bearer <token>`.

Vérification du rôle :

Le serveur extrait le rôle depuis le JWT et autorise ou bloque l'accès selon les endpoints :

- Accès **enseignant** : autorisé uniquement si `role === "teacher"`.
- Accès **étudiant** : autorisé uniquement si `role === "student"`.

. Communication front-end / back-end : `api_service`

Tous les échanges entre l'interface mobile et le back-end sont gérés par le module `api_service`. Celui-ci:

- Insère automatiquement le **token JWT** dans les en-têtes des requêtes.
- Redirige l'utilisateur vers la page de login en cas de token expiré ou invalide.
- Gère les appels aux endpoints suivants :

Planning de surveillance (enseignant)

- **Endpoint** : `/teacher/{id}/exams`
- **Méthode** : `GET`
- **Protection** : Token JWT avec rôle "teacher"
- **Retour** : Liste des créneaux de surveillance (heure, salle, matière)

Planning d'examens (étudiant)

- **Endpoint** : `/student/niveau`
- **Méthode** : `GET`
- **Protection** : Token JWT avec rôle "student"
- **Retour** : Planning d'examen de l'étudiant

. Notifications (Server-Sent Events - SSE) :

Technologie : Utilisation de Server-Sent Events (SSE) pour envoyer des notifications en temps réel.

Déclenchement : Une notification est générée automatiquement lors d'une modification d'examen côté serveur.

Contenu : Le message envoyé contient les détails de la mise à jour (ex. : nouvel horaire, changement de salle), au format JSON. Destinataires : Seuls les enseignants ou étudiants concernés, identifiés par leur email via le token JWT, reçoivent la notification.

Connexion SSE : Après le login, l'application Flutter ouvre une connexion SSE sécurisée vers /notifications/streamStudent ou /notifications/StreamTeacher en envoyant le JWT.

8. Technologies utilisées

- **Backend** : Java + Spring Boot
- **Mobile** : Flutter (Dart)
- **Base de données** : PostgreSQL
- **API de notifications** : Intégration Server-Sent Events (SSE)
- **Contrôle de version** : Git + GitHub

9. Conclusion

L'application mobile proposée est une solution moderne et efficace pour centraliser l'accès aux plannings d'examens et de surveillance. Grâce à son architecture modulaire et à l'utilisation des dernières technologies, elle garantit une bonne expérience utilisateur tout en assurant la sécurité et la fiabilité des données.