Développement d’une plateforme de gestion d’événements et de rendez-vous

# Contexte et objectif

L’établissement souhaite développer une application lourde en Java permettant de gérer efficacement deux aspects clés de la rentrée :

* La **gestion de stock des fournitures**, indispensable pour assurer une organisation fluide du matériel pédagogique.
* Les **dossiers d’inscription des nouveaux élèves**, avec un suivi personnalisé et des rendez-vous organisés entre étudiants et professeurs.

# Expression des besoins

## Fonctionnalités principales

**Gestion des rôles et utilisateurs**

1. Trois rôles principaux sont définis :
   * **Secrétaire** : Création des fiches étudiants et dossiers d’inscription.
   * **Professeur** : Prise en charge des dossiers d’inscription, organisation des rendez-vous, et demande de fournitures.
   * **Gestionnaire de stock** : Gestion des fournitures, fournisseurs, demandes de réapprovisionnement.
2. Chaque utilisateur est identifié par son : **nom**, **prénom**, **email**.

**Gestion des inscriptions des étudiants**

1. **Création de fiche étudiant** par une secrétaire (si l’étudiant est inconnu). Informations nécessaires :
   * Nom, prénom, dernier diplôme obtenu, email, téléphone, adresse.
2. **Dossier d’inscription** : Créé par une secrétaire pour un étudiant.
   * Informations requises :
     1. Date, heure, filière d’intérêt, motivations de l’étudiant.
3. **Rendez-vous avec un professeur** :
   * Un professeur peut organiser un rendez-vous avec un étudiant si le dossier est intéressant.
   * Gestion des **disponibilités des salles** (trois salles disponibles, une demi-journée par rendez-vous).
   * Le rendez-vous ne peut pas être créé si les salles sont déjà occupées.

**Gestion des fournitures**

1. **Demandes de fournitures par les professeurs** :
   * Le professeur précise les articles souhaités, les quantités et la raison de la demande.
2. **Validation par le gestionnaire de stock** :
   * Les demandes validées mettent à jour automatiquement les stocks.
   * Les demandes refusées restent enregistrées avec une justification.
3. **Gestion des fiches fournitures** :
   * Une fourniture est décrite par : libellé, description, et liste des fournisseurs possibles (avec prix pour chaque fournisseur).
4. **Réapprovisionnement** :
   * Le gestionnaire doit pouvoir passer commande auprès des fournisseurs lorsque les stocks sont faibles.

## Cybersécurité et RGPD

* Contrôle des accès :
  + Mots de passe forts et gestion sécurisée.
  + Journalisation des connexions : date et heure de la dernière connexion.
  + Gestion stricte des autorisations selon les rôles.
* Journalisation des actions des utilisateurs :
* Enregistrement des connexions (date et heure).
* Historique des créations/modifications.

# Organisation du projet

## Livrables attendus

* **Code source** : Stocké dans un dépôt Git pour permettre le suivi des contributions et de l’avancement.
* **Documentation technique et fonctionnelle** : Explications claires du code, de l’architecture et de l’utilisation.
* **Application fonctionnelle** : Respectant l’ensemble des besoins et prête à être déployée/testée.

## Contraintes de temps

* **Rendu final du projet** : TBD.
* **Mise en place de Git** pour le suivi du développement.
* Revues régulières avec l'informaticien pour valider l’avancement et répondre aux questions.

## Méthodologie et outils

* Méthodologie **Agile** :
  + Réunions hebdomadaires pour suivre l’avancement.
  + Livraison incrémentale des fonctionnalités.
* **Outils** recommandés :
  + IDE : IntelliJ IDEA, Eclipse ou NetBeans.
  + Gestion de version : GitHub, GitLab.
  + Base de données : MySQL.
  + JavaFX ou Swing pour l’interface utilisateur.

# BONUS et pistes de réflexion

Pour pousser le projet plus loin :

1. **Reporting et statistiques** :
   * Générer des rapports, par exemple :
     + Statistiques sur les dossiers d’inscription par filière.
     + Historique des demandes de fournitures par professeur.
     + Analyse des taux d’occupation des salles de rendez-vous.
2. **Alertes et notifications** :
   * Alertes automatiques pour les niveaux de stock critiques.
   * Notifications pour rappeler les rendez-vous aux professeurs.
3. **Exportation de documents** :
   * Export des dossiers d’inscription ou des rapports au format PDF.

## Pistes de réflexion technique

1. **Gestion avancée des salles de rendez-vous** :
   * Implémenter un système pour détecter automatiquement les créneaux disponibles et proposer des alternatives en cas de conflit.
2. **Interface graphique (GUI)** :
   * Développer une interface utilisateur en **JavaFX** pour une meilleure ergonomie. Par exemple :
     + Calendrier interactif pour les rendez-vous.
3. **Performance et recherche avancée** :
   * Optimiser la gestion des données en simulant un grand volume d’utilisateurs et de fournitures (plusieurs centaines).
   * Ajouter une fonctionnalité de recherche avancée pour trouver rapidement des fiches d’étudiants ou des fournitures spécifiques.
4. **Améliorations en cybersécurité** :
   * Ajouter une double authentification pour les utilisateurs.
   * Mettre en œuvre des tests de pénétration pour identifier des failles potentielles dans l’application.
5. **Recherche intelligente** : Trouver un dossier ou un produit via une barre de recherche.

# Contraintes techniques

1. **Technologies principales** :
   * Développement en JavaFX (application lourde).
   * Base de données relationnelle (MySQL).
2. **Architecture logicielle** :
   * Utilisation de l’architecture MVC ou MVVM pour organiser le code.