



Brief 6

Part 1:

Le laboratoire médical **TechLab** souhaite mettre en place un système de gestion complet pour optimiser ses opérations. L'objectif principal est d'améliorer l'efficacité et la précision dans le traitement des analyses médicales. Le système doit couvrir divers aspects de la gestion, du suivi des échantillons à la gestion des résultats. La transition vers ce système aidera TechLab à fournir un service plus rapide et plus précis à ses patients.

Fonctionnalités du Système de Gestion du Laboratoire "LabXpert"

- 1. Enregistrement des Échantillons :

Les techniciens peuvent enregistrer de nouveaux échantillons en spécifiant les informations pertinentes telles que le patient, le type d'analyse et la date de prélèvement.

- 2. Suivi des Analyses en Cours :

Une interface conviviale permet aux techniciens et aux responsables de laboratoire

de suivre en temps réel l'état d'avancement des analyses en cours, avec des détails spécifiques pour chaque échantillon.

- **3. Gestion des Résultats :**

Les résultats des analyses sont consignés de manière systématique, permettant un accès rapide aux informations et la possibilité de partager les résultats avec les professionnels de la santé concernés.

- **4. Gestion des Patients :**

Un module dédié offre la possibilité de gérer les informations relatives aux patients, assurant une centralisation des données et une navigation facilitée.

- **5. Inventaire des Réactifs :**

Intégration d'un suivi des stocks pour garantir la disponibilité des réactifs nécessaires aux différentes analyses.

- **6. Gestion des Utilisateurs :**

Une interface d'administration permet de gérer les droits d'accès et les informations des utilisateurs, assurant une sécurité accrue des données.

- **7. Planification des Analyses :**

Possibilité de planifier les analyses en fonction de la charge de travail, optimisant ainsi l'utilisation des ressources du laboratoire.

- **8. Rapports Statistiques :** Génération de rapports statistiques pour évaluer les performances du laboratoire, identifier les tendances et prendre des décisions basées sur les données.

Note : Votre Mission vous la trouverez dans le fichier ressource “Mission 1”

Stack Technique de l'Application "LabXpert"

:

- **Langage de Programmation : Java**
- **Backend : Spring boot API RESTful**
- **Gestion de Dépendances : Apache Maven**
- **Base de Données : PostgreSQL**

- **Serveur d'Application : Apache Tomcat**
- **Testing : JUnit & Mockito**
- **CICD : Jenkins**
- **Gestion des tâches : un outil de votre choix .**
- **Système de Gestion de Version : Git et Github**
- **Documentation d'API : SWAGGER**
- **Logging : Log4j pour la gestion des journaux**

Les CLasses :

1. Échantillon :

- `idEchantillon`
- `patient`
- `typeanalyse`
- `statusEchantillon`
- `datePrelevement`

2. Analyse :

- `idAnalyse`
- `patient`
- `echantillon`
- `user`
- `dateDebut`
- `dateFin`
- `tests`
- `commentaires`

3. Test :

- `idTest`
- `typeTest`
- `statusTest`
- `analyse`
- `reactive`
- `tools`

4. Patient:

- `idPatient`
- `nomComplet`
- `adresse`
- `telephone`
- `email`
- `sex`
- `historiqueAnalyse`

5. Réactif :

- `idReactif`
- `nomReactif`
- `descreption`
- `quantityReactif`
- `dateExperation`
- `test`
- `fournisseur`

5. Utilisateur (User):

- Identifiant (id)
- Nom d'utilisateur
- Mot de passe (crypté)
- Rôle (technicien, responsable de laboratoire, administrateur)
- Informations personnelles

6. User :

- `idUser`
- `username`
- `password`
- `adresse`
- `telephone`
- `email`
- `sex`
- `role`

7. Rapport :

- `idRapport`
- `rapporType`
- `periodeDebut`
- `periodeFin`
- `historiqueAnalyse`

8. Tools :

- `id`
- `nomTool`

- `utilisation`
- `quantity`

9. TypeTest :

- `maxType1`
- `minType1`
- `maxType2`
- `minType2`
- `maxType3`
- `minType3`

Enums :

- RapportType
- Role
- StatusEchantillon
- StatusTest
- TypeAnalyse
- UniteMesure

Ces entités devraient couvrir les principales fonctionnalités décrites dans le cahier des charges. Il est important de souligner que la conception précise des entités peut varier en fonction des détails spécifiques du système et des besoins réels de l'application. Vous pouvez ajuster ces modèles en fonction de votre contexte particulier.

Echantillon

Analyse

- Résultat
- Réactif
- Utilisateur
- Rapport Statistique

Diagram class & Use Case

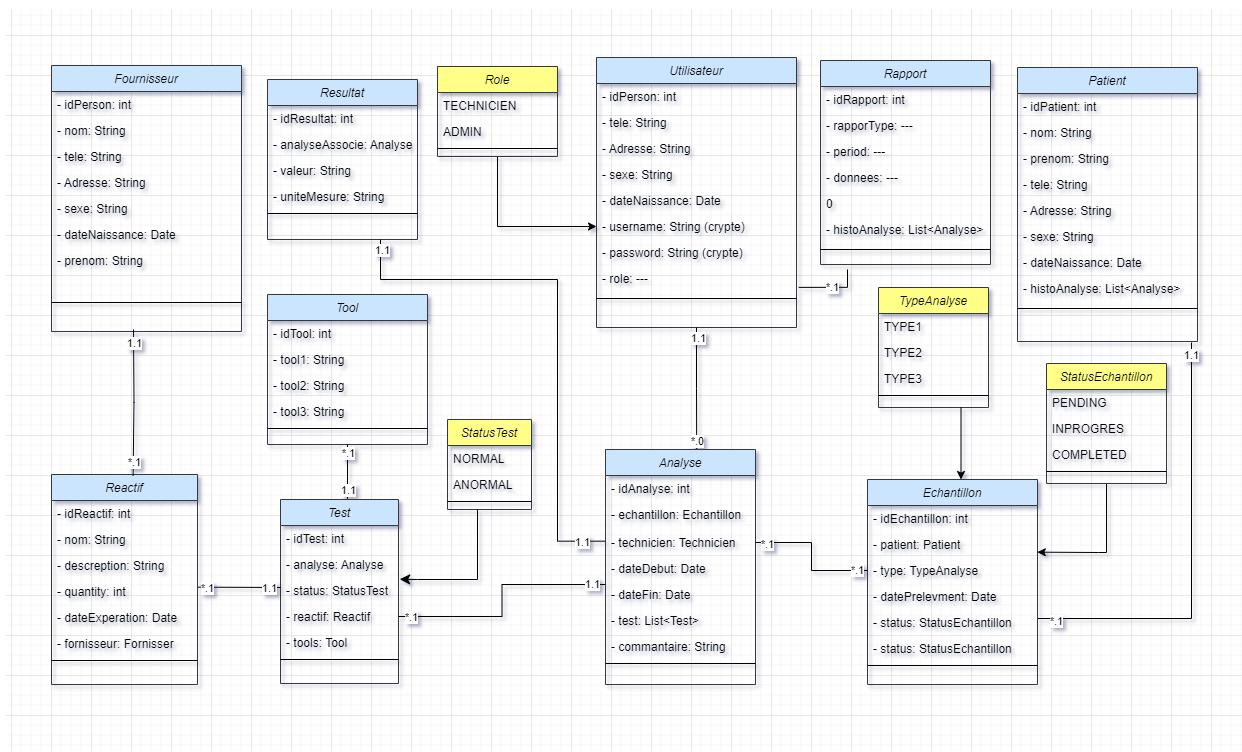
.IO

[Diagram Class.drawio](#)

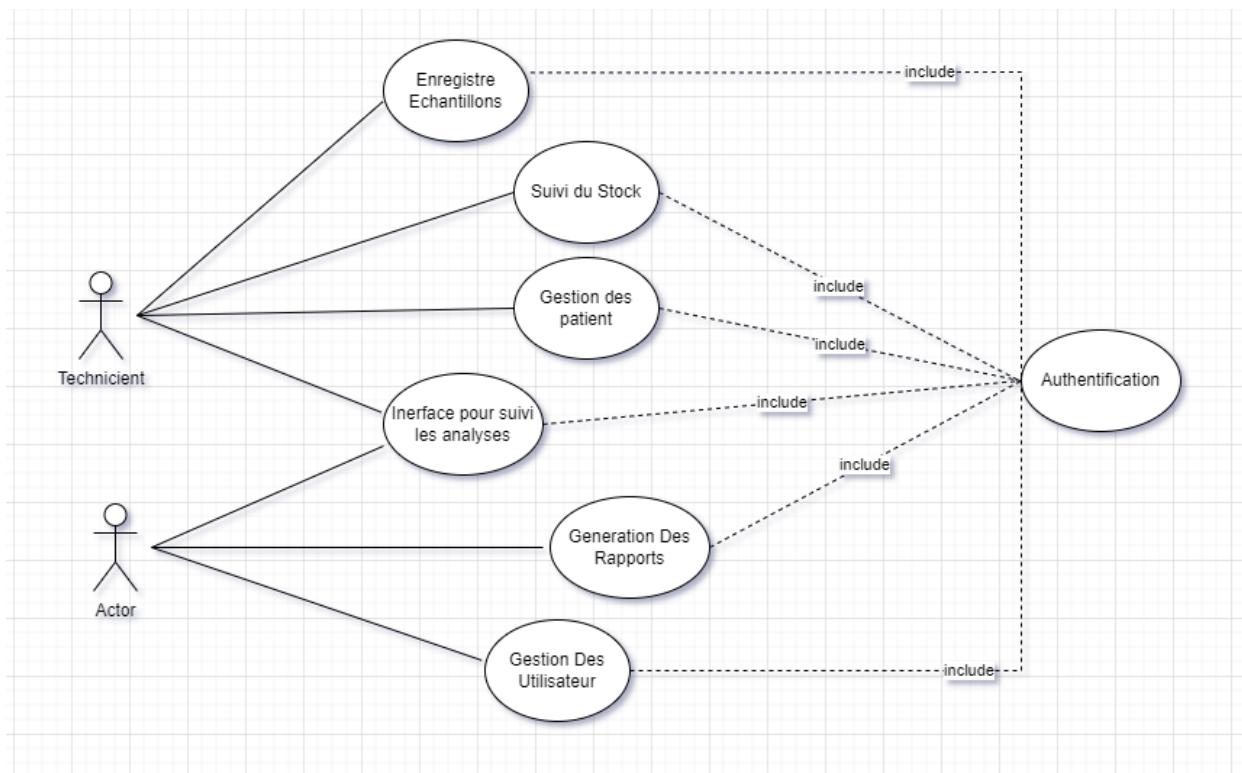
[use-case.drawio](#)

.PNG

Le Diagram de Class



Le Diagram de Cas d'utilisation



Classes de Test:

1. TestEchantillon :

Cette classe de test vérifiera toutes les méthodes et fonctionnalités associées à la classe Echantillon. Par exemple, elle peut comprendre des tests pour l'enregistrement d'un nouvel échantillon, l'obtention des détails d'un échantillon spécifique, la mise à jour des informations d'un échantillon, etc.

2. TestAnalyse :

Cette classe de test comprendra des tests pour toutes les méthodes et fonctionnalités liées à la classe Analyse. Elle peut comprendre des tests pour le démarrage d'une nouvelle analyse, l'obtention des détails d'une analyse spécifique, la mise à jour des informations d'une analyse, etc.

3. TestTest :

Cette classe de test comprendra des tests pour toutes les méthodes et fonctionnalités liées à la classe Test. Elle peut comprendre des tests pour l'ajout d'un nouveau test, l'obtention des détails d'un test spécifique, la mise à jour des informations d'un test, etc.

4. TestPatient :

Cette classe de test comprendra des tests pour toutes les méthodes et fonctionnalités liées à la classe Patient. Elle peut comprendre des tests pour l'ajout d'un nouveau patient, l'obtention des détails d'un patient spécifique, la mise à jour des informations d'un patient, etc.

5. TestReactif :

Cette classe de test comprendra des tests pour toutes les méthodes et fonctionnalités liées à la classe Réactif. Elle peut comprendre des tests pour l'ajout d'un nouveau réactif, l'obtention des détails d'un réactif spécifique, la mise à jour des informations d'un réactif, etc.

6. TestUtilisateur :

Cette classe de test comprendra des tests pour toutes les méthodes et fonctionnalités liées à la classe Utilisateur. Elle peut comprendre des tests pour l'ajout d'un nouvel utilisateur, l'obtention des détails d'un utilisateur spécifique, la mise à jour des informations d'un utilisateur, etc.

7. TestRapport :

Cette classe de test comprendra des tests pour toutes les méthodes et fonctionnalités liées à la classe Rapport. Elle peut comprendre des tests pour la création d'un nouveau rapport, l'obtention des détails d'un rapport spécifique, la mise à jour des informations d'un rapport, etc.

La Structure de project :

```
com.yassin.labxpert
|-- application
|   |-- LabXpertApplication.java
|
|-- entity
|   |-- Analyse.java
|   |-- Echantillon.java
|   |-- Fournisseur.java
|   |-- Patient.java
|   |-- Rapport.java
|   |-- Reagent.java
|   |-- Test.java
|   |-- Tool.java
|   |-- TypeTest.java
|   |-- User.java
|
|-- repository
|   |-- AnalyseRepository.java
|   |-- EchantillonRepository.java
|   |-- FournisseurRepository.java
|   |-- PatientRepository.java
|   |-- RapportRepository.java
|   |-- ReagentRepository.java
|   |-- TestRepository.java
|   |-- ToolRepository.java
|   |-- TypeTestRepository.java
|   |-- UserRepository.java
```

```
|  
|-- LabXpertApplicationTests.java
```

La Resultat de Client (test):

The screenshot shows a Java IDE interface with the following details:

- Project Structure:** The project is named "LabAnalyse". It contains a "UserRepository" package with "LabAnalyseApplication", "resources", and "test" sub-packages. The "test" package contains "java" and "org.yassin.labxpert" sub-packages, which further contain "entity", "AnalyseTest", and "EchantillonTest" classes.
- Code Editor:** The "PatientTest.java" file is open. It includes annotations like `@SpringBootTest` and `@Autowired`, and defines a class `PatientTest` with methods `insertPatients()`, `getPatients()`, and `deletePatients()`.
- Test Results:** The "PatientTest" class is listed in the test runner with a status of "Passed". Below it, the individual test methods are listed with their execution times:
 - `insertPatients()`: 897 ms
 - `getPatients()`: 510ms
 - `deletePatients()`: 323 ms
 - `getPatients()`: 64 msA tooltip indicates "Failed to start: 3, passed: 0".
- Console Output:** The terminal window shows the command "C:\Program Files\Java\jdk1.8.0_301\bin\java.exe" ... followed by Spring Boot test context logs.
- System Tray:** The taskbar shows icons for File Explorer, Google Chrome, Microsoft Edge, and Notepad.
- System Information:** The system tray displays the date (1/12/2024), time (4:06 PM), battery level (69%), and network connection.