

---

La réalisation d'une  
application système  
expert

# Plan d'Assurance Qualité Logicielle

---

**Date : 1/03/2023**

**Version : 1.0**

**Statut : Final**

**Auteurs :**

**Laouane Ilyass**

**Youssef El Housni**

**Ben Hamou Mohamed**

**Zakaria Hida**

**Aiman Mouhat**

## Table des matières

|   |           |
|---|-----------|
| <b>1. But, portée, responsabilité .....</b>           | <b>4</b>  |
| 1.1. Objectif du document .....                       | 4         |
| 1.2. Portée du document .....                         | 4         |
| 1.3. Responsabilités associées .....                  | 5         |
| 1.4. Procédure d'évolution du PAQL .....              | 5         |
| <b>2. Documentation utilisée.....</b>                 | <b>6</b>  |
| 2.1. Documents de référence .....                     | 6         |
| <b>3. Terminologie .....</b>                          | <b>6</b>  |
| <b>4. Organisation .....</b>                          | <b>7</b>  |
| 4.1. Maîtrise d'ouvrage .....                         | 7         |
| 4.2. Maîtrise d'œuvre .....                           | 10        |
| 4.3. Chef de projet .....                             | 11        |
| 4.4. Responsable qualité.....                         | 11        |
| <b>5. Cycle de vie .....</b>                          | <b>12</b> |
| 5.1. Motivation du choix .....                        | 12        |
| 5.2. Description des étapes du cycle de vie.....      | 13        |
| 5.3. Phase du cycle de vie : .....                    | 19        |
| <b>6. Documentation produite .....</b>                | <b>21</b> |
| <b>7. Gestion de la configuration .....</b>           | <b>21</b> |
| 7.1. Outils de travail collaboratif .....             | 21        |
| 7.2. Environnement technique .....                    | 22        |
| <b>8. Gestion des modifications.....</b>              | <b>22</b> |
| 8.1. Procédure de modification .....                  | 22        |
| 8.2. Règles d'évolution des numéros de version .....  | 23        |
| 8.3. Règles d'évolution du statut .....               | 23        |
| <b>10. Méthodes, outils, règles, normes.....</b>      | <b>24</b> |
| 10.1. Méthodes .....                                  | 24        |
| 10.2. Outils .....                                    | 24        |
| 10.3. Règles et normes à respecter.....               | 25        |
| <b>11. Exigences et évaluation de la qualité.....</b> | <b>30</b> |
| 11.1. Facteurs .....                                  | 30        |
| 11.2. Critères.....                                   | 30        |
| 11.3. Évaluation .....                                | 32        |
| <b>12. Reproduction, protection, livraison .....</b>  | <b>35</b> |
| 12.1. Procédure de reproduction .....                 | 35        |

|   |    |
|---|----|
| <b>14. Suivi de l'application du PAQL</b> ..... | 36 |
| <b>14.1. Validation des documents</b> .....     | 36 |
| <b>14.2. Relecture</b> .....                    | 36 |

# 1. But, portée, responsabilité

## 1.1. Objectif du document

Le plan d'assurance qualité (PAQL) a pour but de définir les méthodes et outils utilisés par le projet, ainsi que les mesures à prendre et les étapes pour contrôler et s'assurer de la qualité du projet.

Tout document produit sera soumis au contrôle de qualité. Il devra être conforme aux règles définies dans ce document. Tout document non conforme devra être corrigé.

- **Méthode de gestion de notre projet :**

Nous avons adopté la méthode agile Scrum pour la gestion de notre projet. En effet, l'agilité va nous apporter des bénéfices absolument indispensables à la réussite de ce projet

- **Les mesures à prendre et les étapes pour contrôler et s'assurer de la qualité du projet :**

- ✓ Évaluation Initiale du Projet :
- ✓ Évaluation du PAQL en Début de Projet
- ✓ Évaluations Continues Pendant le Développement .
- ✓ Évaluation de la Base de Connaissances du Système Expert .
- ✓ Évaluation de la Sécurité et de la Confidentialité .
- ✓ Évaluation de la Documentation .
- ✓ Évaluation Finale du Projet .

## 1.2. Portée du document

Ce document est destiné :

- **À l'encadrant:**

- ✓ **Professeur Benhadou**

- À l'équipe du projet:
  - ✓ Laouane Ilyass
  - ✓ Zakaria Hida
  - ✓ Ben Hamou Mo
  - ✓ Mouhat Aiman
  - ✓ El Housni Youssef

### 1.3. Responsabilités associées

Le responsable qualité est chargée de la rédaction du présent PAQL ainsi que de veiller à son application et son évolution, en collaboration avec le chef de projet. C'est à lui de décider des actions à entreprendre si le PAQL n'est pas appliqué.

- Le responsable qualité du projet : Youssef El Housni
- Ses compétences:
  - Rigueur et méthode dans le travail
  - Excellente capacité d'analyse et de synthèse
  - Maîtrise de Java
  - Maîtrise des Outils de Gestion de Projet
  - Compétences en Test Logiciel
  - Compréhension de la Sécurité Logicielle

### 1.4. Procédure d'évolution du PAQL

Le PAQL est élaboré au début du projet. A chaque étape, il est susceptible d'être modifié, auquel cas la modification sera indiquée dans la table de l'historique.

- ✓ recueillir / stocker les informations
- ✓ analyse des informations

- ✓ Utilisation des résultats (à transmettre à tous les partenaires) pour atteindre les objectifs du projet

## 2. Documentation utilisée

### 2.1. Documents de référence

Le tableau suivant récapitule les principales sources documentaires qui seront utilisées dans le cadre de ce projet.

| <i>But</i>              | <i>Source</i>        |
|-------------------------|----------------------|
| Rédaction des documents | Cours Pr.Benhadou    |
| Rédaction des documents | Cours Pr.Aoun        |
| Rédaction des documents | Cours OpenClassrooms |

| IDE      | Documentation IDE   |
|----------|---|
| NetBeans | C'est un environnement de développement qui permet de créer rapidement et facilement des applications de haute qualité. |

## 3. Terminologie

Les termes suivants sont les termes spécifiques utilisés dans le PAQL.

- › CdC : Cahier des Charges

- › CdB : Cahier de Bord
- › DCG : Dossier de conception globale
- › DCD : Document de Conception Détaillée
- › DSE : Dossier de Spécifications Externes
- › PAQL : Plan d'Assurance Qualité Logiciel
- › PDL : Plan de Développement Logiciel
- › MOA : Maîtrise d'ouvrage
- › **SQL** : Structured Query Language
- › **UML** : Unified Modeling Language
- › **SGBD** : système de gestion de base de données.
- › **PAQL** : Plan d'Assurance Qualité Logiciel
  
- › **POO** : programmation orienté objet

## 4. Organisation

### 4.1. Maîtrise d'ouvrage

#### 4.1.1. MOA

› [ENSEM](#)

#### 4.1.2. MOAd

› [Pr. Benhadou](#)

## 4.2. Maîtrise d'œuvre

### 4.2.1. MOE

- ✓ **Laouane Ilyass**
- ✓ **Zakaria Hida**
- ✓ **Ben Hamou Mo**
- ✓ **Mouhat Aiman**
- ✓ **El Housni Youssef**

La maîtrise d'oeuvre déléguée est l'équipe de développement composée de :

- ✓ **Laouane Ilyass**
- ✓ **Zakaria Hida**
- ✓ **Ben Hamou Mo**
- ✓ **Mouhat Aiman**
- ✓ **El Housni Youssef**

Leur rôle est de proposer une solution logicielle visant à assister les utilisateurs dans le diagnostic et la résolution de problèmes techniques liés à leurs machines. Elle offre une approche interactive et intelligente pour répondre aux besoins spécifiques des utilisateurs. Voici un aperçu des fonctionnalités et contraintes de l'application :

Pour cela, ils doivent faire en sorte que :

- › Le planning soit observé et réajusté d'un commun accord si besoin
- › Les normes de qualité soient définies et respectées
- › La validation du code produit soit conforme au PAQL



### 4.3. Chef de projet

Le chef de projet sera **Professeur Benhadou** pour la période **d'un mois**, Le chef de projet est responsable :

- ✓de la planification du projet
- ✓du contrôle de l'avancement du projet
- ✓de la mobilisation des moyens nécessaires de la coordination des travaux de chacun

### 4.4. Responsable qualité

Le responsable qualité sera **Youssef EL Housni** pour la période **d'un mois**  
Le responsable qualité a pour mission de :

- ✓définir les règles de retour arrière et les procédures de modification
- ✓veiller à la diffusion et au respect des règles particulières au projet
- ✓gérer l'évolution du PAQL
- ✓superviser les actions de vérification et de validation

## 5. Cycle de vie

### 5.1. Motivation du choix

Chaque exigence du client peut être satisfaite de manière indépendante des autres. Ainsi, l'utilisation d'un cycle de vie permettant de développer chaque module de manière autonome et complète est appropriée. Le produit final sera donc livré de manière incrémentielle par lots successifs. Le cycle de vie choisi suit la méthodologie Scrum, caractérisée par un ensemble de réunions clairement définies et strictement limitées dans le temps (time-boxing).



## 5.2. Description des étapes du cycle de vie

### ➤ **Analyse des besoins :**

L'application d'expert vise à répondre de manière efficace et sécurisée aux besoins des utilisateurs en matière de résolution de problèmes techniques. En permettant aux utilisateurs de s'authentifier de manière sécurisée, l'application offre un accès personnalisé à une base de connaissances complète, proposant des solutions pré-établies pour les problèmes courants. Lorsqu'un utilisateur ne trouve pas de solution dans la base de connaissances, il peut poser une question spécifique, déclenchant ainsi une transmission efficace à des experts pour une assistance personnalisée. Les réponses des experts sont ensuite affichées sur le compte utilisateur ayant posé la question, favorisant une communication transparente. L'ajout des nouveaux problèmes-réponses dans l'application par les experts enrichit constamment la base de connaissances, accessible à tous les utilisateurs. La confidentialité des échanges, la traçabilité des interactions, et un mécanisme de rétroaction utilisateur contribuent à assurer une expérience utilisateur conviviale, sécurisée et constamment améliorée.

### ➤ **Spécifications externes :**

## **1. Tâche complexe 1 : Authentification et Gestion des Utilisateurs :**

### **Voici les tâches élémentaires de cette tâche complexe :**

- ✓ Créer un formulaire d'inscription pour les utilisateurs.
- ✓ Mettre en place un mécanisme de vérification sécurisé des identités.
- ✓ Implémenter une fonctionnalité de connexion sécurisée avec gestion des sessions.

## **2. Tâche complexe : Mise à jour de la Base de Connaissances :**

**Voici les tâches élémentaires de cette tâche complexe :**

- ✓ Élaborer une base de connaissances avec une structure hiérarchique et des catégories de problèmes.
- ✓ Intégrer un algorithme de suggestion de solutions basé sur les diagnostics.

## **3. Tâche complexe 3 : « Publication des Réponses sur le Compte Utilisateur :»**

**Voici les tâches élémentaires de cette tâche complexe :**

- ✓ Élaborer une interface pour afficher les réponses sur le compte utilisateur.
- ✓ Mettre en œuvre un système de notification pour informer les utilisateurs des nouvelles réponses.
- ✓ Tester la fonctionnalité d'affichage des réponses dans différents scénarios.

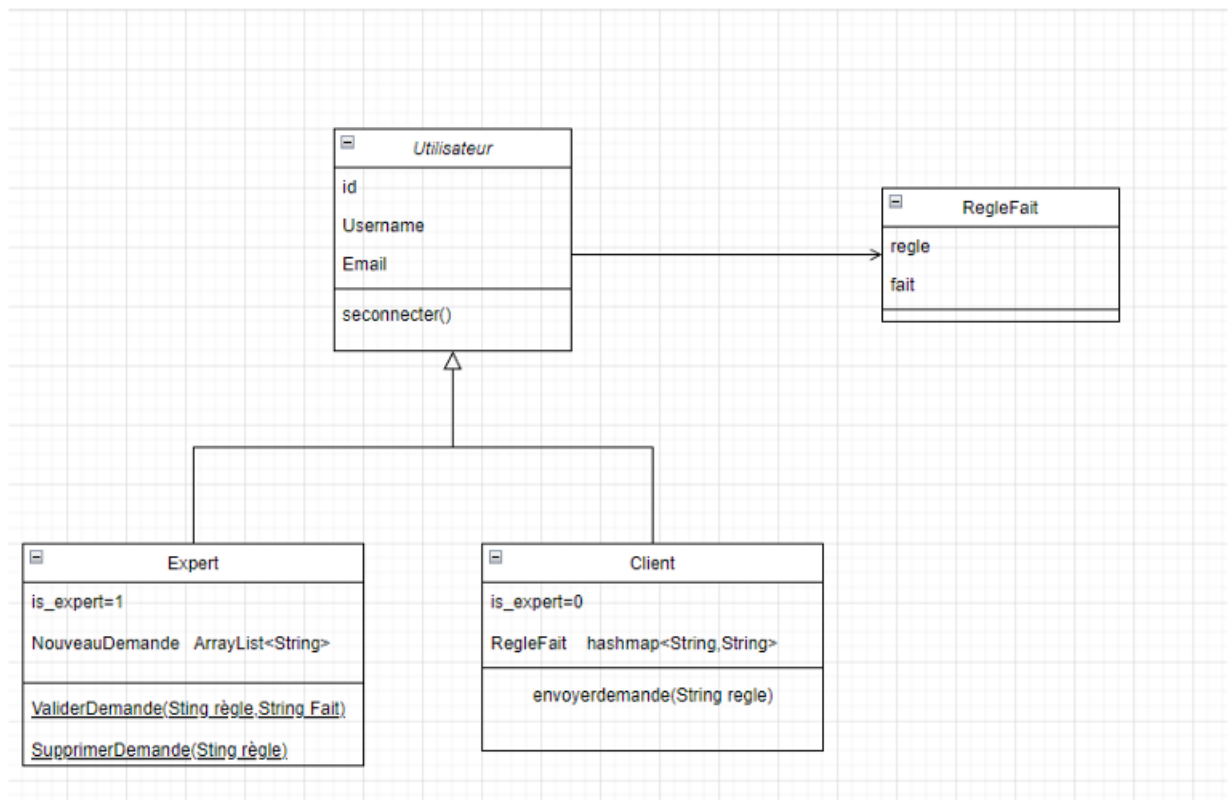
## **4. Tâche complexe 4 : « Interaction Utilisateur-Expert :»**

**Voici les tâches élémentaires de cette tâche complexe:**

- ✓ Développer un formulaire de saisie de questions pour les utilisateurs.
- ✓ Établir un mécanisme de transmission des questions aux experts.
- ✓ Mettre en place un tableau de bord pour les experts, affichant les questions non résolues.

► **Conception :**

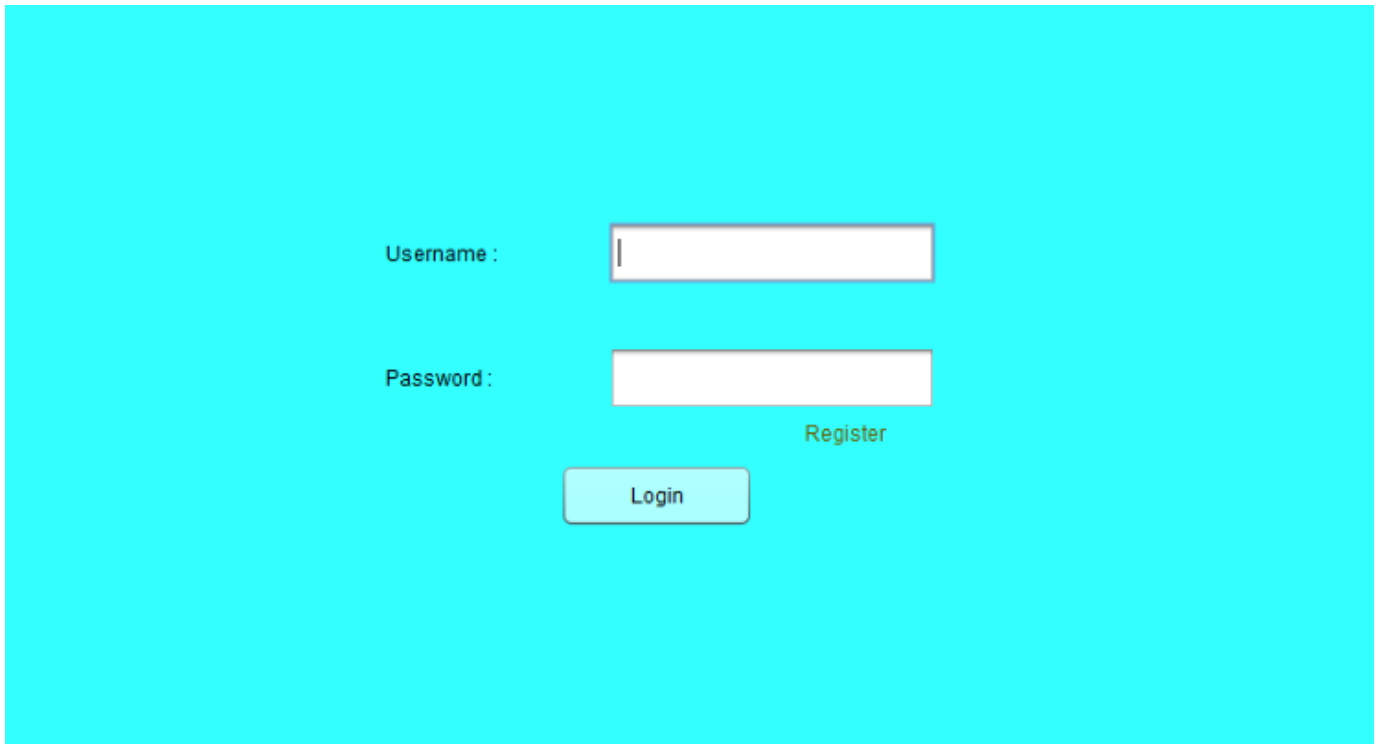
Voici le diagramme des classes de l'application:



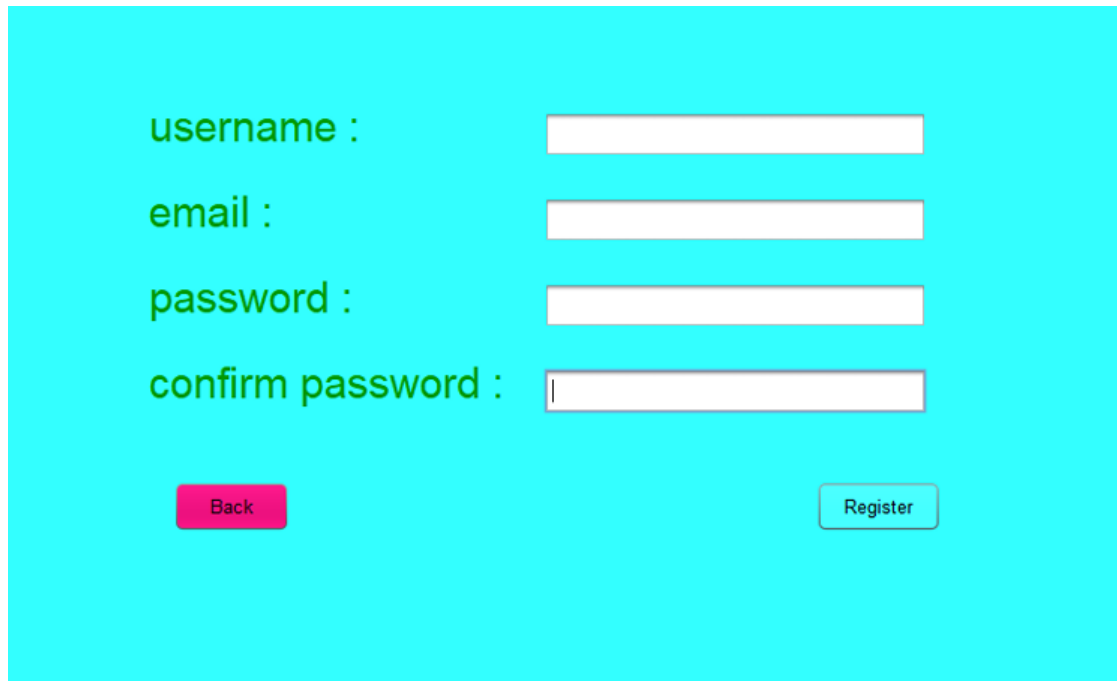
### › Tests/Livraison :

Voici l'application que nous avons réalisé :

Page d'authentification :

The image shows a screenshot of a web page for authentication. It has a light blue background. On the left, there is a vertical sidebar with a dark blue header containing the text "ENSEM" in white. Below the header, there are several menu items in white text: "Accueil", "A propos", "Contact", "FAQ", "Politique de confidentialité", "Conditions d'utilisation", "Mentions légales", "Liens utiles", "Partenaires", "Presse", "Recherche", "Services", "Statistiques", "Téléchargements", "Vie de l'école", "Vie de la recherche", "Vie de la société". The main content area on the right contains the authentication form. It has a title "Page d'authentification" in bold. Below the title, there are two input fields: "Username :" and "Password :". To the right of the "Password :" field, there is a link "Register" in blue text. Below the input fields, there is a blue button with the text "Login" in white.

Interface d'enregistrement :

A user registration form with a light blue background. It contains four text input fields for "username", "email", "password", and "confirm password". Below the fields are two buttons: a red "Back" button and a blue "Register" button.

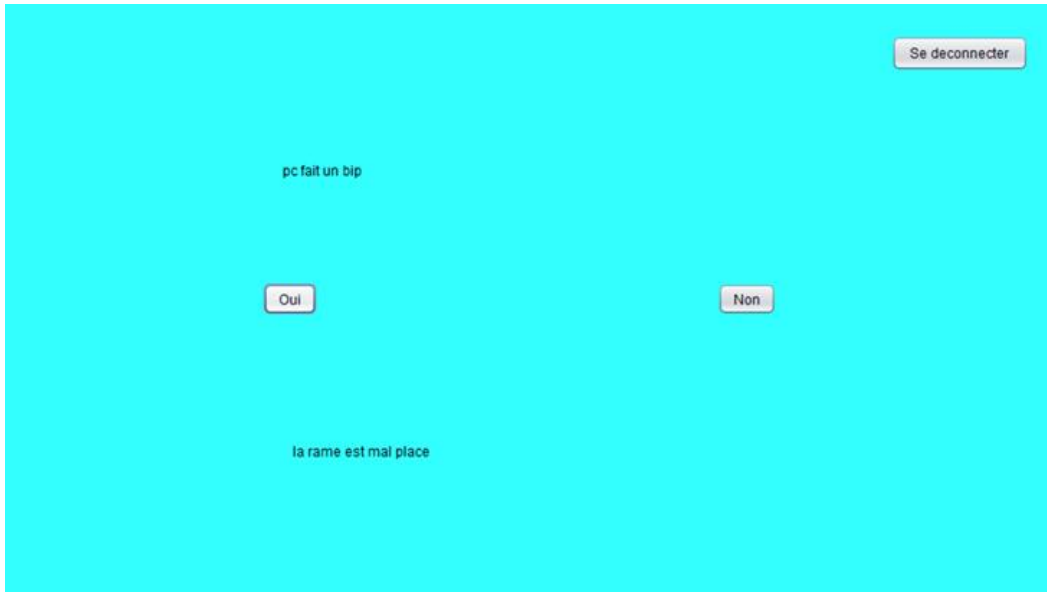
username :

email :

password :

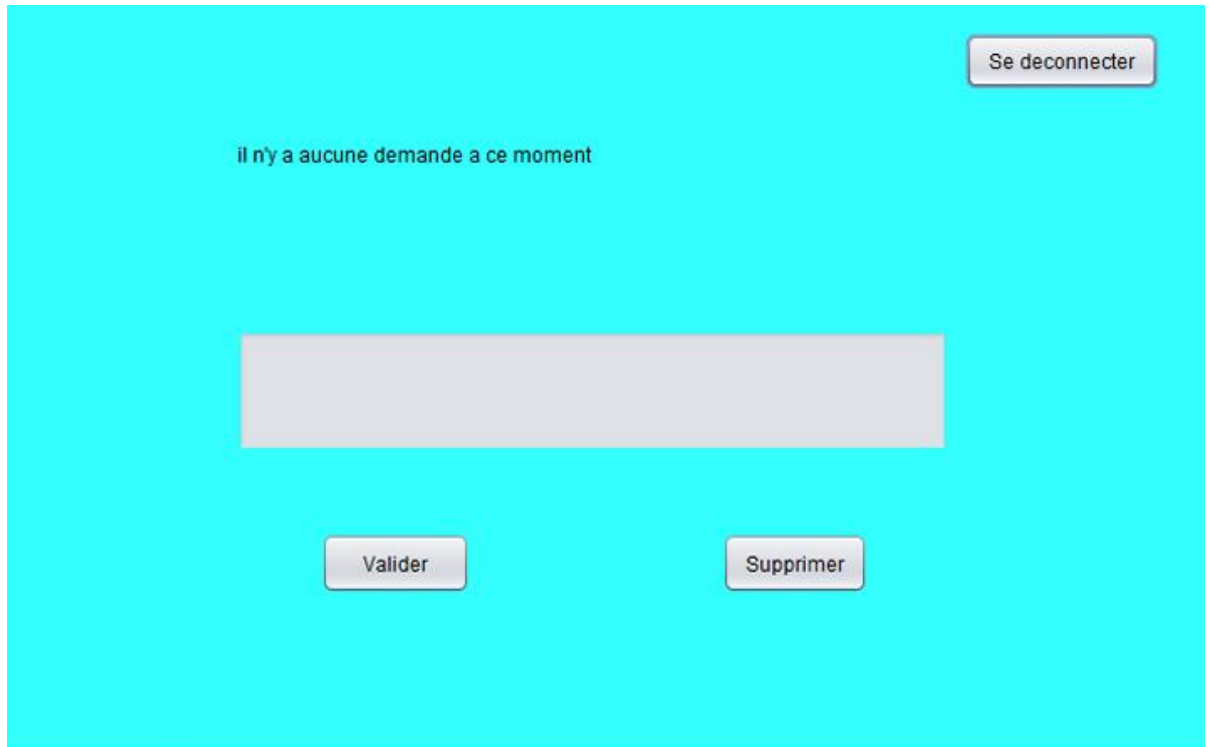
confirm password :

[Back](#) [Register](#)



**Interface d'expert :**





### 5.3. Phase du cycle de vie :

Voici pour chaque étape du cycle de vie « **Scrum** » les documents requis et produits, ainsi que les conditions de passage d'une étape à une autre. [\[Le tableau en bas est un exemple de listing des documents livrés dans un cycle de vie en V, utiliser le ainsi que le support de cours pour faire le bon tableau pour votre projet\]](#)

|                        | Activités  | Documents   |  | Condition  |
|------------------------|--|---|--|--|
|                        |  | Requis  | Produits   |  |
| Analyse des besoins    | L'analyse des besoins de l'application se focalise sur la définition précise des fonctionnalités et des caractéristiques nécessaires pour répondre aux attentes des utilisateurs et atteindre les objectifs du projet. Pour l'application d'expert proposée, plusieurs besoins clés ont été identifiés | CdC   | PAQL<br>PDL  | Validation de la méthode de gestion de projet et de la modélisation<br><br>Par Zakaria |
| Spécifications Externe | Étude détaillée du logiciel à réaliser, préparation des tests système.   | <ul style="list-style-type: none"> <li>CdC</li> </ul> | <ul style="list-style-type: none"> <li>Rapport de fonctionnalités détaillés</li> </ul> | Validation du rapport par Zakaria  |

## 6. Documentation produite

A l'issue de ce projet, plusieurs documents auront été produits. Certains devront être livrés avec le produit fini, certains seront disponibles si le client les souhaite et certains ne serviront qu'à l'équipe de développement. Le tableau suivant précise pour chaque document son statut.

| Document                            | Statut      |
|-------------------------------------|-------------|
| Cahier des charges                  | Livable     |
| Plan d'assurance qualité logicielle | Livable     |
| Plan de développement logiciel      | Privé       |
| Rapport de fonctionnalités          | Livable     |
| Diagramme de classes                | Consultable |
| Plans de tests                      | Livable     |
| Jeu de tests                        | Livable     |

- Description des différents statuts :
  - Livable : Doit être fourni au client
  - Consultable : Peut être consulté par le client
  - Privé : Destiné à l'équipe du projet uniquement

## 7. Gestion de la configuration

La configuration est l'ensemble des éléments suivants : code source exécutable, outils de développement et de test utilisés, documents et données.

### 7.1. Outils de travail collaboratif

Voici les outils que nous avons utilisé pour gérer notre travail collaboratif :

✓ **GitHub /Git :**

**GitHub** nous a facilité la collaboration en utilisant **git**. C'est une plateforme qui nous a permis de poser le code dans un stockage dans le cloud afin que notre équipe puisse travailler sur le même projet.

✓ **Google Meet:**

Google meet était le moyen efficace que nous avons utilisé pour se réunir à distance afin de discuter l'avancement du projet

## 7.2. Environnement technique

Le développement est effectué sous environnement :

- ✓ OS: Windows (version 10)
- ✓ IDE: NetBeans
- ✓ Langage de programmation : Java (version 17)
- ✓ Base de données: MySQL

## 8. Gestion des modifications

### 8.1. Procédure de modification

9. Les modifications peuvent avoir deux origines :

- › Détection d'une anomalie
- › Demande d'évolution

Dans le cas d'une détection d'anomalie, il faut en trouver la source puis la corriger dans les plus brefs délais.

Dans le cas d'une demande d'évolution du logiciel par le client, il faut dans un premier temps réaliser une étude de faisabilité, pour ensuite modifier le logiciel en conséquence si cela s'avère réalisable dans les délais impartis.

Les membres du MOEd sont responsables de la mise en œuvre de ces modifications.

## 8.2. Règles d'évolution des numéros de version

Documents et applications sont identifiés par un numéro de version, un numéro de révision et un numéro de correction comme suit : « NuméroVersion.NuméroRévision.NuméroCorrection ».

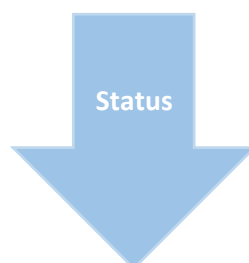
Trois cas peuvent nécessiter un changement de version.

- › **Une nouvelle livraison de l'application ou du document :**
  - ✓ On incrémente le numéro de version. Les numéros de révision et de correction reviennent à zéro.
- › **Une évolution :**
  - ✓ On incrémente le numéro de révision. Le numéro de correction revient à zéro.
- › **Une correction d'anomalie(s) :**
  - ✓ On incrémente le numéro de correction

## 8.3. Règles d'évolution du statut

Pour chaque document, un statut doit être mentionné. Celui-ci peut-être :

- › Documents : PAQL, rapport final, rapport de la modélisation, rapport de la conception



**Final : ces documents ont été validés. Ils peuvent être consultés et livrés**

## 10. Méthodes, outils, règles, normes

### 10.1. Méthodes

✓ **UML a été utilisée pour toutes les phases de la conception de ce projet.**

Les tests seront préparés pendant les premières étapes du projet, à savoir :

- › Analyse des besoins : Préparation des tests d'acceptation
- › Spécifications externes : Préparation des tests systèmes
- › Conception globale : Préparation des tests d'intégration
- › Conception détaillée : Préparation des tests unitaires

Ces tests devront être préparés au fur et à mesure de l'avancement du projet.

### 10.2. Outils

Le tableau suivant récapitule les outils qui seront utilisés dans le cadre de ce projet.

| Fonctions                      | Outils                         |
|--------------------------------|--------------------------------|
| Editeur de texte               | <b>NetBeans</b>                |
| Editeur UML                    | <b>Draw.io</b>                 |
| Environnement de développement | <b>NetBeans</b>                |
| Développement collaboratif     | <b>Git/GitHub, Google meet</b> |
| Gestion de projets             | <b>Click-up</b>                |

## 10.3. Règles et normes à respecter

### 9.3.1. Règles de présentation

Tous les documents liés à ce projet devront respecter un même modèle :

› page de garde

✓ Nom du projet : Développement d'une application système expert

✓ Nom du document : PAQL

✓ Date de dernière modification : 01/01/2024

✓ Numéro de version : 1.0

✓ Auteur(s) :

- ✓ Laouane Ilyass
- ✓ Mouhat Aiman
- ✓ Hida Zakaria
- ✓ El housni Youssef
- ✓ Ben hamou Mohamed

### 9.3.2. Règles de programmation

✓ **Le système sera développé en langage JAVA**

✓ Historique des modifications

› Pour chaque méthode

✓ Description succincte de la méthode

✓ Description des paramètres utilisés

✓ Description de la valeur de retour

› Convention de nommage

✓ Le nom d'une classe commence par une majuscule. Si ce nom est lui-même composé de plusieurs noms, chacun d'entre eux commence par une majuscule. Les autres lettres sont en minuscules.

→Exemple :

- `public class Composant{ }`
- `public class TypeComposant{ }`

✓ Le nom d'une méthode ou d'une variable commence par une minuscule. Si ce nom est lui-même composé de plusieurs noms, chacun d'entre eux commence par une majuscule. Les autres lettres sont en minuscules.



→Exemple :

- `public void init(){ }`
- `public void doPost{ }`

✓ Le nom des variables doit être significatif pour une meilleure compréhension du code.

Le respect de ces règles de programmation devra être vérifié à l'aide de CheckStyle (plugin Eclipse). Un minimum d'avertissements devra être présent. Aucun seuil n'est fixé car certaines erreurs de CheckStyle n'ont pas un grand impact sur le style de programmation mais nécessite par contre un grand investissement en temps.

Il s'agira donc de réduire au maximum le nombre d'avertissements en perdant un minimum de temps.

### 9.3.1. Normes sur les comptes-rendus de réunions

Chaque réunion / revues / meeting donnera lieu à la rédaction d'un compte-rendu. Celui-ci se compose de la manière suivante :

« nom\_date ».

✓ nom : étude du CdC

✓ jj : 20

✓ mois : 04

✓ aa : 20

Par exemple, le compte-rendu de l'étude du CdC du 20 avril 2023 serait nommé : « etude\_CdC\_200423 ».

### 9.3.2. Normes sur les documents livrables

#### **notre document possède le numéro de version 1.0**

Tout incrément du premier indice (x) implique un changement notoire du document. Tout incrément du second indice (y) implique une modification minimale du document.

### 9.3.3. Normes sur les fiches de relecture

Chaque diffusion d'un document donne lieu à une relecture et au remplissage d'une fiche de relecture. Celle-ci se compose de la manière suivante : «nomDocAssocié\_Relecteur\_Date».

- ✓ nomDocAssocié : Le nom du document abrégé associé à la fiche de relecture
- ✓ Relecteur : Le nom du relecteur
- ✓ Date : La date de création de la fiche

Par exemple, la fiche de relecture associée au Plan d'Assurance Qualité (PAQL) créé par Zakaria Hida le 7 février 2023 sera nommée : « PAQL\_Camilleri\_06022023 ». Un modèle de fiche est fourni : « Modele\_fiche\_relecture » ainsi qu'une fiche explicative annexe :

« PAQL\_Fiche\_relecture ».

## 11. Exigences et évaluation de la qualité

### 11.1. Facteurs


Les facteurs de qualité suivant ont été identifiés comme importants pour ce projet et devront être validés :

- › **Portabilité** : Aptitude du logiciel à être transféré d'un matériel et/ou d'un environnement logiciel à un autre
- › **Maintenabilité** : Aptitude du logiciel à pouvoir être corrigé facilement.

### 11.2. Critères

Chaque facteur est lié à des critères précis. Parmi ceux-ci, un minimum de 4 critères par facteur devra être assuré.

| Maintenabilité   | Portabilité  | Sécurité   |
|--|--|--|
| <ul style="list-style-type: none"> <li>Instrumentation</li> <li>Consistance</li> <li>Simplicité</li> <li>Modularité</li> <li>Auto-description</li> <li>Concision</li> <li>Communicabilité</li> </ul> | <ul style="list-style-type: none"> <li>Simplicité</li> <li>Modularité</li> <li>Autodescription</li> <li>Indépendance logicielle</li> <li>Indépendance machine</li> </ul> | <ul style="list-style-type: none"> <li>Confidentialité</li> <li>Intégrité</li> </ul> |

 Principaux critères retenus

 Autres critères

#### 10.2.1. Motivation des choix

**Les critères de confidentialité, simplicité, consistance et intégrité ont été écartés.**

## 11.3. Évaluation

La partie suivante présente les mesures qui seront effectuées pour quantifier le respect de chaque critère sélectionné. Les conditions de validation de chaque critère seront également précisées.

### 10.3.1. Modularité

La modularité est l'aptitude d'un logiciel à être composé de modules indépendants.

le langage de programmation utilisé dans ce projet est **JAVA**, la modularité sera calculée en fonction du nombre de lignes de code composant chaque classe.

La taille maximale recommandée d'une classe **JAVA** est de 500 lignes de code. Au-delà, le code devient trop complexe

Une classe sera considérée « valide » si elle respecte l'expression suivante :

$$\text{ClasseValide} = \text{NombreDeLignesDeCode} < 500$$

Le critère « modularité » sera donc mesuré par l'expression suivante :

$$\text{Modularité} = 10 * (\text{NombreDeClassesValides} / \text{NombreTotalDeClasses})$$

Condition de validation de ce critère :

La note sur 10 obtenue devra être supérieure à 8.5.

- **Nombre de classes : 26, nos classes sont validées**
- **Modularité : (26/26) \*10=10**

### 10.3.2. Auto-description

L'auto-description est l'aptitude d'un logiciel à fournir la description de chacune de ses fonctions.

Le langage de programmation utilisé dans ce projet est **JAVA**

Une classe sera considérée « valide » si elle respecte l'expression suivante :

$$\text{ClasseValide} = \text{NombreDeLignesDeCommentaires} / \text{NombreDeLignesDeCode} > 30\%$$

Le critère « auto-description » sera donc mesuré par l'expression suivante :

$$\text{Auto-Description} = 10 * (\text{NombreDeClassesValides} / \text{NombreTotalDeClasses})$$

Condition de validation de ce critère :

La note sur 10 obtenue devra être supérieure à 8.5.

- **Les classes ne sont pas valides**
- **Auto-description: 0.5**

### 10.3.3. Indépendance logicielle

L'indépendance logicielle est l'aptitude d'un logiciel à ne pas être lié, de par son fonctionnement, à un environnement logiciel particulier. Pour respecter ce critère, le logiciel issu de ce projet devra fonctionner pareillement sous Windows 10 et sous Linux (Fedora).

Le critère « indépendance logicielle » sera donc mesuré par l'expression suivante :

$$\text{Indépendance logicielle} = 10 * (\text{FonctionneIndependammentDeL'OS})$$

Condition de validation de ce critère :

La note sur 10 obtenue devra être égale à 10

#### 10.3.5. Simplicité

La simplicité est l'aptitude d'un logiciel à avoir un fonctionnement interne compréhensible facilement.

Pour cela, des règles de programmation ont été prises (voir chapitre IX.3.B).

Une fonction sera considérée « valide » si elle respecte l'expression suivante :

FonctionValide = NormesDeProgrammationRespectées (vrai ou faux)

Le critère « simplicité » sera donc mesuré par l'expression suivante :

$$\text{Simplicité} = 10 * (\text{NombreDeFonctionsValides} / \text{NombreTotalDeFonctions})$$

Condition de validation de ce critère :

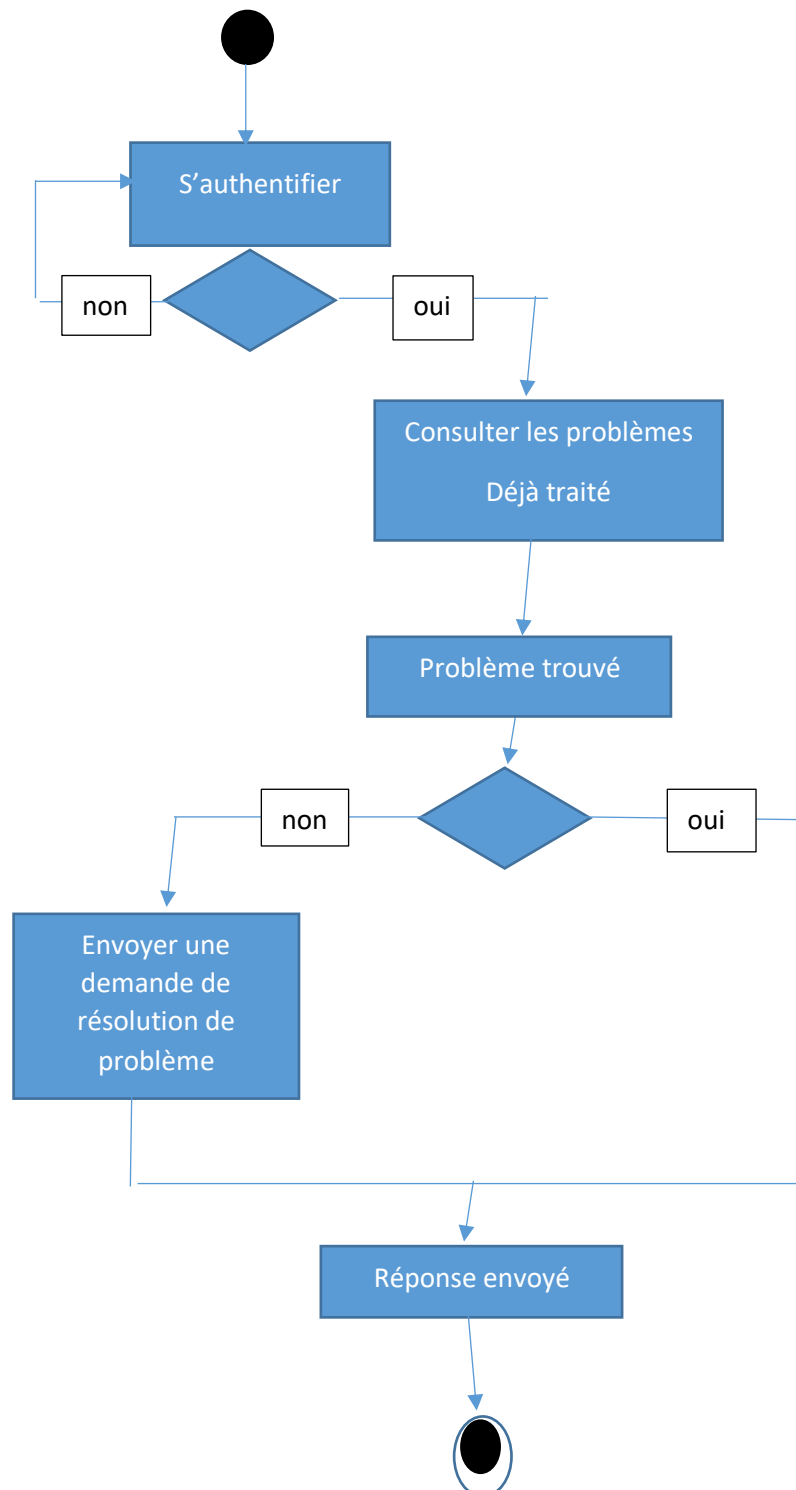
La note sur 10 obtenue devra être supérieure à 8.5.

- **FonctionValide: Vrai**
- **Simplicité:  $10 * (34/34) = 10$**



## 12. Reproduction, protection, livraison

### 12.1. Procédure de reproduction



### 13.1. Livraison et installation

Le logiciel final sera remis à notre professeur Bnhadou. Il contiendra **le rapport, le code source, et la présentation**

Les documents livrables seront fournis dans un répertoire **Système expert**

Lors de la livraison du produit au client, une procédure complète d'installation sera fournie (dans le manuel utilisateur) afin de permettre à des personnes étrangères à l'équipe de développement d'installer et d'utiliser le produit sans difficultés.

## 14. Suivi de l'application du PAQL

### 14.1. Validation des documents

Tous les documents rédigés seront relus par le responsable qualité et modifiés en cas de non- conformité avec le présent Plan d'Assurance Qualité Logicielle.

De la même façon, les documents de type « livrables » seront relus et devront être validés par le MOEd.

Les audits prévus permettront également de vérifier le bon respect du PAQL. En effet, pendant ces réunions, la situation du produit et l'avancement du projet seront examinés méthodiquement.

➤ **Tous les documents du livrable sont validés par MOED Benhadou**

### 14.2. Relecture

La lecture croisée sera effectuée au minimum par les 2 membres de l'équipe de développement (l'un rédige, l'autre relit) et par le MOEd.

L'auteur d'un document assure la réalisation des corrections proposés par les relecteurs et gère le changement des numéros de versions.

L'enchaînement des tâches est le suivant :

- › **Création du document**
- › **Diffusion aux relecteurs potentiels avec la fiche de relecture associée**
- › Validation finale, après plusieurs cycles de diffusion-correction, effectuée par le MOEd Benhadou