Cahier des charges

Version 1.0

Les demandes client :

* Développer une application pour faire le suivi des plans de tests. L’application doit communiquer avec Jenkins.
* Le pipeline ne doit pas passer si un test utilisateur est périmé par rapport à la périodicité demandée.
* Avoir un historique des TU échoués.
* Je dois avoir un tableau de bord avec plusieurs indicateurs.
* Je dois avoir une gestion des droits de mes utilisateurs.
* Quand un test utilisateur ne passe pas, je dois pouvoir répertorier un nouveau bug.
* Je dois avoir un tableau des bugs ouverts, et les catégoriser par niveau de gravité. Un bug répertorié avec un niveau de gravité élevé va bloquer le pipeline.
* Faire une reprise de données du tableau Excel pour l'intégrer dans la base de données.

Spécifications Techniques :

* Système d’exploitation : Windows / Linux
* Langages : Java 11 - JavaScript
* Technologies : Spring 2.6.4 – Angular 2.0
* Base de données : MySQL Server (en local)
* Serveur : X

Spécifications Fonctionnelles :

* Authentification à l’application
  + Connexion
  + Déconnexion
  + Mot de passe oublié
  + Contacter un administrateur
* Tableau de bord (Sélection d’un projet / Création d’un projet)
  + Création d’un projet déjà existant => Import d’un fichier CSV
  + Selon un template ?
  + Suppression d’un projet
  + Modification d’un projet
* Menu d’accès aux différents sous-menu / fonctionnalités
  + Menu d’intégration des plans de tests
  + Menu d’un test utilisateur
* Tableau de bord (2 versions Utilisateurs / Développeur)
  + Utilisateurs :
    - Exécution d’un test
    - Ajouter des bugs
  + Développeur :
    - Gestion des tests, création suppression
    - Archivage des tests
* Menu d’intégration des plans de tests
  + Création d’un plan de tests
  + Suppression d’un plan de tests
  + Modification d’un plan de tests
  + Accès l’historique d’un plan de tests
* Menu d’un test utilisateur
  + Création d’un test utilisateur
    - Plan de tests
    - Test :
      * Thématique
      * Groupe
      * Fonctionnalité
      * Détails du test
      * Périodicité
      * Date de test
      * Version testée
      * Critique
      * Bug
      * Fixé
      * Bug relevé par l’utilisateur
  + Suppression d’un test utilisateur
  + Modification d’un test utilisateur
  + Historique d’un test utilisateur
* Historique des tests utilisateurs échoués
  + Accès aux détails du test
* Intégration au pipeline
  + Git
  + Configuration
    - Token d’accès
    - API address
  + Monitoring des tests
    - Condition de succès du pipeline
    - Gestion de la criticité des tests
    - Arrêt du pipeline selon critère
* Gestion des comptes (mode admin seulement)
  + Création d’un compte développeur / utilisateurs
  + Gestion des droits de compte
  + Supprimer un compte
  + (Historique des actions d’un compte) ?

Rôles :

* Admin :
  + Gestion des plans tests
  + Gestion des utilisateurs
  + Intégration au pipeline
* Développeur :
  + Accès au suivi
  + Gestion des plans de tests
  + Gestion des bugs
* Utilisateur :
  + Création de bug
  + Ajouter un bug utilisateur

Mineur : Bug non gênant

Majeur : Bug gênant mais non bloquant

Critique : Bug empêchant un usage correct de l’application Comment catégoriser ces « bugs » en fonction des données collectées ?

Maquette :

|  |  |  |  |
| --- | --- | --- | --- |
| **Date :** | **08/03/2021** | **Version 1.0** | **Remarques** |
| **Participants** | **Visa** |  |  |
| Chad |  |  |  |
| Tito |  |  |  |
| Mickael |  |  |  |
| Thomas | **Ok** |  |  |
| Client |  |  |  |

Signatures :