Cahier des Charges

Résumé

Ce projet vise à créer une plateforme de réservation de services en ligne en utilisant Spring Boot pour le back-end et React pour le front-end. Il fournira aux utilisateurs une expérience conviviale pour rechercher, réserver et gérer divers services.

Les fonctionnalités incluent un système d'authentification sécurisé, une interface de recherche et de réservation intuitive, la gestion des profils utilisateurs et des historiques de réservation, ainsi qu'un système de paiement sécurisé.

L'architecture reposera sur une API RESTful développée avec Spring Boot pour le back-end, garantissant une communication fluide avec le front-end React. La sécurité sera assurée par JWT.

La conception de l'interface utilisateur sera attrayante et réactive, développée en utilisant React et des composants de bibliothèques telles que Material-UI.

La phase de test sera approfondie avant le déploiement sur un serveur en ligne, avec une gestion de projet agile pour une adaptation aux besoins changeants.

# 1 Présentation du projet

## Contexte

Pour assurer le suivi et l'amélioration de mon projet académique, centré sur la réservation de services en ligne, j'envisage de mettre en place un système de suivi des utilisateurs inscrits. Ce système sera essentiel pour optimiser l'expérience des utilisateurs, présenter efficacement le projet et établir un réseau de contacts pertinent.

Mon projet académique, axé sur la réservation de services en ligne, repose sur une architecture comprenant Spring Boot pour le backend, React pour l'interface web et Java avec Android Studio pour l'application mobile. Afin de garantir la sécurité des données, j'utiliserai JWT (JSON Web Tokens).

Tout comme le master en Bio-informatique Moléculaire Méthodes et Analyses (BMMA) de l'université Lyon 1, la réalisation de ce projet nécessite un suivi attentif des utilisateurs afin de pouvoir améliorer continuellement le service offert, adapter l'interface utilisateur et développer un réseau d'utilisateurs fidèles.

Pour ce faire, une application web sera développée, offrant une interface simple et conviviale pour les utilisateurs inscrits. Bien que le projet soit encore au stade initial, une version partielle a déjà été réalisée, servant de point de départ pour le développement d'une application robuste et accessible.

Parallèlement, je m'inspire des études régulières réalisées par le ministère de l'enseignement supérieur, où des données statistiques sont collectées sur le devenir des étudiants. De manière similaire, je collecterai des données pertinentes sur l'utilisation de mon application, permettant ainsi d'évaluer son impact et d'identifier les axes d'amélioration.

## Objectifs

1. Augmentation du nombre d'utilisateurs actifs : Le projet doit viser à attirer un nombre croissant d'utilisateurs actifs sur la plateforme, mesuré par le nombre d'inscriptions et de connexions régulières.
2. Taux de conversion amélioré : En mettant en place une interface conviviale et des fonctionnalités attrayantes, le projet devrait augmenter le taux de conversion des visiteurs en utilisateurs actifs et des utilisateurs actifs en réservations confirmées.
3. Réduction du taux d'abandon du processus de réservation : En optimisant le flux de réservation et en offrant une expérience utilisateur fluide, le projet devrait réduire le taux d'abandon du processus de réservation avant son achèvement.
4. Amélioration de la satisfaction des utilisateurs : À travers des fonctionnalités intuitives, une navigation aisée et un service client réactif, le projet devrait améliorer la satisfaction globale des utilisateurs, mesurée par des enquêtes de satisfaction et des retours d'expérience positifs.
5. Accroissement du chiffre d'affaires : En augmentant le volume des réservations et en favorisant des transactions réussies, le projet devrait contribuer à une augmentation significative du chiffre d'affaires généré par la plateforme.

## Description de l'existant

Pas de’ existant

## Critères d'acceptabilité du produit :

1. Fonctionnalité :

* Les utilisateurs doivent pouvoir rechercher, réserver et annuler des services.
* Le système doit envoyer des notifications aux utilisateurs pour confirmer leurs réservations.
* Les utilisateurs doivent pouvoir payer pour les services réservés en ligne.

1. Performance :

* Le temps de réponse du système doit être rapide, même avec un grand nombre d'utilisateurs simultanés.
* Les transactions financières doivent être sécurisées et effectuées sans délai excessif.

1. Compatibilité :

* Le système doit être compatible avec différents navigateurs web (Chrome, Firefox, Safari, etc.).
* Le système doit être compatible avec les appareils mobiles courants (iOS, Android).

1. Sécurité :

* Le système doit protéger les données personnelles des utilisateurs conformément aux réglementations en vigueur (GDPR, CCPA, etc.).
* Les transactions financières doivent être sécurisées en utilisant des protocoles de cryptage appropriés.

1. Convivialité :

* L'interface utilisateur doit être intuitive et facile à naviguer.
* Les messages d'erreur doivent être clairs et informatifs pour guider les utilisateurs en cas de problème.

1. Évolutivité :

* Le système doit être conçu de manière à pouvoir ajouter facilement de nouvelles fonctionnalités à l'avenir.
* Il doit être capable de gérer une augmentation du nombre d'utilisateurs sans

### Procédure de validation :

1. Tests unitaires et d'intégration :

* Effectuer des tests unitaires pour chaque composant logiciel.
* Tester l'intégration entre les différentes parties du système (back-end, front-end, base de données, etc.).

1. Tests de régression :

* Réexécuter les tests existants pour s'assurer qu'aucune nouvelle fonctionnalité n'a introduit de régression.

1. Tests de performance :

* Effectuer des tests de charge pour évaluer la performance du système sous différentes conditions de charge.
* Identifier et résoudre les goulets d'étranglement de performance.

1. Tests de sécurité :

* Effectuer des tests de sécurité pour identifier et corriger les vulnérabilités potentielles.
* Utiliser des outils d'analyse statique et dynamique pour détecter les failles de sécurité.

1. Tests de convivialité :

* Effectuer des tests utilisateurs pour évaluer la convivialité de l'interface utilisateur.
* Recueillir les commentaires des utilisateurs pour identifier les points d'amélioration.

1. Tests de compatibilité :

* Tester le système sur différents navigateurs web et appareils mobiles pour s'assurer de sa compatibilité.
* Corriger les problèmes d'affichage ou de fonctionnement rencontrés sur des plateformes spécifiques.

## 2. Expression des besoins : 2.1 Besoins fonctionnels :

Les besoins fonctionnels décrivent les opérations et les transformations que le logiciel doit effectuer. Voici quelques exemples de spécifications fonctionnelles pour un système de réservation de services en ligne utilisant Spring Boot et React pour le front-end, et Java pour le mobile :

1. **Création de compte :**
   * Les utilisateurs doivent pouvoir s'inscrire en fournissant des informations telles que leur nom, leur adresse e-mail et leur mot de passe.
2. **Recherche de services :**
   * Les utilisateurs doivent pouvoir rechercher des services disponibles en fonction de différents critères tels que la date, l'heure, la localisation, etc.
3. **Affichage des détails du service :**
   * Les utilisateurs doivent pouvoir consulter les détails d'un service, y compris sa description, son prix, sa disponibilité, etc.
4. **Réservation de services :**
   * Les utilisateurs doivent pouvoir réserver un service en sélectionnant la date, l'heure et toute autre option pertinente.
5. **Gestion des réservations :**
   * Les utilisateurs doivent pouvoir consulter, modifier ou annuler leurs réservations existantes.
6. **Paiement en ligne :**
   * Les utilisateurs doivent pouvoir effectuer le paiement en ligne pour les services réservés en utilisant des méthodes de paiement sécurisées.
7. **Confirmation de réservation :**
   * Les utilisateurs doivent recevoir une confirmation de leur réservation par e-mail ou par notification dans l'application.

### 2.2 Besoins non fonctionnels :

Les besoins non fonctionnels spécifient les contraintes et les caractéristiques du système qui ne sont pas liées directement aux fonctionnalités. Voici quelques exemples de spécifications non fonctionnelles pour le même système de réservation de services en ligne :

1. **Performances :**
   * Le système doit être capable de gérer un grand nombre d'utilisateurs simultanés sans compromettre les performances.
   * Le temps de réponse du système ne doit pas dépasser une certaine limite, par exemple, 2 secondes pour charger une page ou effectuer une action.
2. **Sécurité :**
   * Les données des utilisateurs doivent être cryptées lors de leur transmission sur le réseau.
   * Le système doit être protégé contre les attaques telles que les injections SQL, les attaques par force brute, etc.
3. **Fiabilité :**
   * Le système doit être disponible et opérationnel la plupart du temps, avec un temps d'arrêt minimal pour la maintenance prévue.
   * Les erreurs doivent être gérées de manière appropriée pour éviter les pannes catastrophiques.
4. **Compatibilité :**
   * Le système doit être compatible avec les navigateurs web courants tels que Chrome, Firefox, Safari, etc.
   * L'application mobile doit être compatible avec les principales versions d'iOS et d'Android.
5. **Évolutivité :**
   * Le système doit être conçu de manière à pouvoir être étendu facilement pour prendre en charge de nouvelles fonctionnalités à l'avenir.
   * Il doit être capable de s'adapter à une augmentation du nombre d'utilisateurs et de services proposés.
6. **Convivialité :**
   * L'interface utilisateur doit être conviviale et intuitive, avec des instructions claires pour guider les utilisateurs tout au long du processus de réservation.
   * L'application mobile doit être réactive et offrir une expérience utilisateur fluide sur différents appareils et tailles d'écran.

## 3. Expression des besoins :

### 1. Coûts :

Il n’y a pas de budget alloué pour ce projet. Le travail se fera donc avec les machines personnelles, ou bien celles mises à disposition par l’université. Par ailleurs l’application web préexistante et les données à stocker sont disponibles pour les membres du projet.

Les bibliothèques et programmes utilisés pour le développement ou le rendu sont disponibles gratuitement.

### 2. Délais :

Le cahier des charges, définissant les besoins et les objectifs du projet, est à rendre pour le 20/04/2024.

Les délivrables du projet sera quant à lui rendu pour le 12/05/2024, et la soutenance orale se déroulera le 12/05/2024.

### 3. Contraintes techniques :

Les contraintes techniques concernent principalement les langages utilisés :

Cette bibliothèque JavaScript moderne permet de créer une interface utilisateur interactive et réactive, offrant ainsi une expérience fluide aux utilisateurs lors de la navigation sur le site web. Grâce à React, je peux développer des composants réutilisables et structurer efficacement l'interface pour une maintenance simplifiée.

Pour le backend, j'ai choisi d'utiliser Spring Boot, un framework Java qui simplifie le développement d'applications web. Avec Spring Boot, je peux rapidement mettre en place des fonctionnalités telles que la gestion des utilisateurs, la sécurité et l'accès à la base de données. Cela me permet de concentrer mes efforts sur le développement des fonctionnalités principales de l'application, tout en bénéficiant d'une architecture robuste et évolutive.

En ce qui concerne l'application mobile, je développe en Java avec Android Studio. Cette combinaison me permet de créer une expérience utilisateur native sur les appareils Android, en exploitant pleinement les fonctionnalités du système d'exploitation. En utilisant Java avec Android Studio, je peux garantir une performance optimale et une intégration fluide avec les appareils Android, offrant ainsi une expérience utilisateur cohérente sur toutes les plateformes.

1. React : React est une bibliothèque JavaScript pour la création d'interfaces utilisateur interactives et réactives. Elle permet de diviser l'interface en composants réutilisables, facilitant ainsi le développement d'applications web dynamiques.
2. Spring Boot : Spring Boot est un framework Java qui simplifie le développement d'applications web en fournissant une configuration par défaut et des fonctionnalités prêtes à l'emploi. Il offre une architecture robuste pour la création d'applications backend en Java, facilitant ainsi le développement et la maintenance des applications.
3. Java avec Android Studio : Java est un langage de programmation utilisé pour le développement d'applications mobiles, web et d'entreprise. En combinaison avec Android Studio, l'IDE officiel pour Android, Java permet de créer des applications mobiles natives pour la plateforme Android. Android Studio fournit des outils avancés pour concevoir, déboguer et déployer des applications Android.

## Déroulement du projet :

### Planification :

La planification originale suit le diagramme de Gantt suivant :

### Plan d’assurance qualité :

Les différents responsables de projet pourront évaluer la qualité des livrables lors de réunions régulières. De plus, une opportunité de tester l'application pourrait se présenter lors de la cérémonie de remise des diplômes, si le développement avance comme prévu. Ce test serait une occasion d'évaluer la performance de l'application et son intégration de nouvelles données.

### Documentation :

Les livrables client et serveur seront accompagnés d'un document "README.md" qui fournira des détails essentiels sur l'application, son utilisation, ainsi que des exemples et des instructions pour les tests nécessaires à sa validation et à sa maintenance. De plus, le code sera auto-documenté afin de faciliter sa réutilisation ultérieure dans d'autres projets.

### Responsabilités :

#### a. Maîtrise d’ouvrage :

#### b. Maîtrise d’œuvre :

L’équipe de réalisation du projet est composée de :

• Badr Eddine Slioui

• Abderrafie Elguessab