Cahier des charges

1. Introduction

a. Contexte du projet

L'entreprise souhaite moderniser la gestion de la réservation de ses salles de réunions en remplaçant l'ancienne méthode papier par une application Web. Cette initiative vise à résoudre plusieurs problématiques rencontrées par les collaborateurs, telles que la difficulté à trouver des salles disponibles, l'optimisation de l'utilisation des salles, et la gestion à distance des réservations.

b. Objectif du projet

L'objectif principal est de développer une application Web qui permettra de gérer efficacement la réservation des salles de réunions. Les besoins spécifiques incluent :

- Faciliter la recherche de salles disponibles.
- Optimiser l'utilisation des salles.
- Permettre la gestion à distance des réservations.

2. Description des fonctionnalités

a. Fonctionnalités pour les collaborateurs

- Lister les salles disponibles.
- Voir le statut de réservation des salles.
- Faire une réservation.
- Annuler une réservation.

b. Fonctionnalités pour les Office Managers

- Toutes les fonctionnalités des collaborateurs.
- Ajouter ou supprimer des salles.
- Empêcher temporairement la réservation de certaines salles.
- Annuler des réservations existantes.
- Changer la capacité maximale des salles.

3. Contraintes techniques

a. Langage et Framework

L'application doit être développée en Python en utilisant le framework Django.

b. Déploiement

- L'application doit pouvoir être déployée dans un container Docker.
- Une instance de l'application doit pouvoir être configurée pour chaque immeuble.

c. Intégration et Maintenance

- L'application doit s'intégrer dans le SI actuel de l'entreprise.
- La maintenance sera assurée par les équipes de développement de l'entreprise.
- Chaque fonctionnalité doit être couverte par des tests unitaires.

4. Livrables

a. Documentation

- Un cahier des charges.
- Un planning prévisionnel estimant le nombre d'itérations nécessaires.

b. Code Source

- Un repository Git avec le code source de l'application.
- Un Dockerfile permettant de containeriser l'application.
- Un fichier d'intégration continue Gitlab CI pour exécuter les tests et déployer le container applicatif.

5. Gestion des risques

a. Risques identifiés

i. Risques liés à la Conception et au Développement

- **Complexité Technique** : Effectuer des revues de code régulières, utiliser des tests unitaires et d'intégration, et suivre les meilleures pratiques de développement.
- **Intégration avec le SI Existant** : Collaborer étroitement avec l'équipe infrastructure, effectuer des tests d'intégration réguliers.
- **Sécurité** : Effectuer des audits de sécurité réguliers, suivre les meilleures pratiques de développement sécurisé.

ii. Risques de Planification

- **Estimation Incorrecte des Efforts :** Utiliser des techniques d'estimation comme le Planning Poker, ajuster les estimations en fonction des retours d'expérience.
- **Changements de Scope :** Gérer les changements de scope avec un processus de gestion des changements, prioriser les fonctionnalités.
- **Dépendances Externes :** Identifier les dépendances dès le début, planifier des marges de temps, avoir des plans de contingence.

iii. Risques liés à la Maintenance

- **Documentation Insuffisante :** Assurer une documentation complète et à jour, former les équipes de maintenance.
- **Turnover des Équipes :** Documenter les connaissances, former plusieurs membres de l'équipe sur les aspects critiques du projet.
- **Évolution Technologique :** Suivre les évolutions technologiques, planifier des mises à jour régulières.

b. Plan de mitigation

i. Mitigation des Risques liés à la Conception et au Développement

- **Complexité Technique :**
  - Effectuer des revues de code régulières.
  - Utiliser des tests unitaires et d'intégration.
  - Suivre les meilleures pratiques de développement.
- **Intégration avec le SI Existant :**
  - Collaborer étroitement avec l'équipe infrastructure.
  - Effectuer des tests d'intégration réguliers.
- **Sécurité :**
  - Effectuer des audits de sécurité réguliers.
  - Suivre les meilleures pratiques de développement sécurisé.

ii. Mitigation des Risques de Planification

- **Estimation Incorrecte des Efforts :**
  - Utiliser des techniques d'estimation comme le Planning Poker.
  - Ajuster les estimations en fonction des retours d'expérience.

- **Changements de Scope :**
    - Gérer les changements de scope avec un processus de gestion des changements.
    - Prioriser les fonctionnalités.
- **Dépendances Externes :**
    - Identifier les dépendances dès le début.
    - Planifier des marges de temps.
    - Avoir des plans de contingence.

iii. Mitigation des Risques liés à la Maintenance

- **Documentation Insuffisante :**
    - Assurer une documentation complète et à jour.
    - Former les équipes de maintenance.
- **Turnover des Équipes :**
    - Documenter les connaissances.
    - Former plusieurs membres de l'équipe sur les aspects critiques du projet.
- **Évolution Technologique :**
    - Suivre les évolutions technologiques.
    - Planifier des mises à jour régulières.

6. Use Cases

a. Use Case : Réserver une Salle

Acteur : Collaborateur

Préconditions : Le collaborateur est authentifié.

Scénario Nominal :

- Le collaborateur sélectionne une salle disponible.
- Le collaborateur choisit une plage horaire pour la réservation.
- Le système vérifie la disponibilité de la salle pour la plage horaire choisie.
- Le système confirme la réservation et met à jour le statut de la salle.

Postconditions : La salle est réservée pour la plage horaire choisie.

b. Use Case : Annuler une Réservation

Acteur : Collaborateur

Préconditions : Le collaborateur est authentifié et a une réservation active.

Scénario Nominal :

- Le collaborateur sélectionne la réservation à annuler.
- Le système vérifie que la réservation appartient bien au collaborateur.
- Le système annule la réservation et met à jour le statut de la salle.
- Postconditions : La réservation est annulée et la salle est de nouveau disponible.

c. Use Case : Ajouter une Salle

Acteur : Office Manager

Préconditions : L'Office Manager est authentifié.

Scénario Nominal :

- L'Office Manager entre les détails de la nouvelle salle (nom, capacité, etc.).
- Le système vérifie les informations entrées.
- Le système ajoute la nouvelle salle à la base de données.
- Postconditions : La nouvelle salle est ajoutée et disponible pour réservation.

7. Modélisation de la base de données

a. Entités et Attributs

i. Salle

- **ID** : Identifiant unique de la salle (Primary Key)
- **Nom** : Nom de la salle
- **Capacité** : Capacité maximale de la salle
- **Statut** : Statut de la salle (disponible, réservée, bloquée)

ii. Réservation

- **ID** : Identifiant unique de la réservation (Primary Key)
- **ID_Salle** : Identifiant de la salle réservée (Foreign Key)
- **ID_Collaborateur** : Identifiant du collaborateur ayant fait la réservation (Foreign Key)
- **Date_Début** : Date et heure de début de la réservation

- **Date_Fin** : Date et heure de fin de la réservation

iii. Collaborateur

- **ID** : Identifiant unique du collaborateur (Primary Key)
- **Nom** : Nom du collaborateur
- **Email** : Adresse email du collaborateur
- **Rôle** : Rôle du collaborateur (collaborateur, office manager)

b. Relations

- Une **Salle** peut avoir plusieurs **Réservations**.
- Un **Collaborateur** peut avoir plusieurs **Réservations**.
- Une **Réservation** est liée à une **Salle** et à un **Collaborateur**.

c. Diagramme ER (Entité-Relations)