Conception de Logiciels

Joe Haifa

Lea Bchouty

[Cahier des Charges : 3](#_Toc178502560)

[1. Introduction 3](#_Toc178502561)

[2. Objectifs 3](#_Toc178502562)

[3. Fonctionnalités Détailées 3](#_Toc178502563)

[a. Authentification 3](#_Toc178502564)

[b. Gestion des clients 4](#_Toc178502565)

[c. Gestion des employés 4](#_Toc178502566)

[d. Gestion des rendez-vous 4](#_Toc178502567)

[e. Gestion des paramètres 4](#_Toc178502568)

[f. Page d'accueil 5](#_Toc178502569)

[g. Perspectives 5](#_Toc178502570)

[4. Contraintes techniques 5](#_Toc178502571)

[5. Interfaces utilisateurs (UI) 6](#_Toc178502572)

[6. Tests et Validation 6](#_Toc178502573)

[7. Livrables 6](#_Toc178502574)

[8. Planning (Sprints) 6](#_Toc178502575)

[Diagramme de Gantt : 7](#_Toc178502576)

[Diagramme de Classe: 8](#_Toc178502577)

Cahier des Charges :

1. Introduction

Le Gym Management System est une application développée en JavaFX pour faciliter la gestion des activités d'une salle de sport. Elle inclut la gestion des clients, des employés, des rendez-vous(séances privées entre client et employé), ainsi que les paramètres de l'application. Ce système est destiné à offrir une solution efficace et intuitive pour les administrateurs de salle de gym.

## 2. Objectifs

- Gestion centralisée des données : Centraliser toutes les informations des clients, employés, rendez-vous et paramètres dans une base de données.

- Interface utilisateur intuitive : Créer une interface utilisateur simple et efficace, permettant à l'administrateur de naviguer facilement entre les différentes pages.

- Gestion des utilisateurs : Authentification sécurisée avec la possibilité de modifier le mot de passe administrateur.

- Gestion des employés et clients : Ajouter, modifier ou supprimer des informations sur les clients et les employés.

- Gestion des rendez-vous : Organiser des rendez-vous entre clients et employés, et permettre leur gestion (ajout, suppression).

- Paramétrage : Permettre la modification de certains paramètres de l'application, comme le changement de base de données.

3. Fonctionnalités Détailées

### a. Authentification

- Login : Le système doit offrir une page de connexion pour s'assurer que seuls les administrateurs ou les utilisateurs autorisés puissent accéder à l'application.

- Mot de passe sécurisé : Les informations de connexion (nom d'utilisateur et mot de passe) doivent être stockées de manière sécurisée, avec des options de récupération ou de modification de mot de passe via la page PasswordPage.

### b. Gestion des clients

- CustomerPage: Cette page permettra de :

- Ajouter un nouveau client avec des informations personnelles comme le nom, prénom, numéro de téléphone, adresse email, date du début de l’abonnement etc.

- Modifier les informations existantes des clients en fonction des besoins.

- Supprimer un client de la base de données si nécessaire.

### c. Gestion des employés

- EmployeePage : Cette page offrira des fonctionnalités similaires à celles de CustomerPage, mais pour les employés.

- Ajouter un employé avec des informations comme le nom, rôle, salaire ,ect.

- Modifier les informations existantes.

- Supprimer un employé.

### d. Gestion des rendez-vous

- AppointmentPage : Cette page permettra de gérer les rendez-vous entre clients et employés. Les fonctionnalités incluent :

- Ajouter un rendez-vous avec un client et un employé spécifique, incluant la date, l’heure et d’autres détails pertinents.

- Supprimer un rendez-vous existant.

e. Gestion des paramètres

- SettingPage : Cette page permet de modifier les paramètres liés à la base de données. Cela inclut :

- La modification des informations de connexion à la base de données.

- La sélection ou le changement de la base de données active.

### f. Page d'accueil

- HomePage : La page d'accueil regroupera toutes les fonctionnalités principales avec des raccourcis vers les pages CustomerPage, EmployeePage, AppointmentPage, LoginPage\*\*, et SettingPage. Cette page servira de tableau de bord pour l’administrateur.

### g. Perspectives

Si j'ai suffisamment de temps, j'envisage une page qui permettrait de gérer l'inventaire des équipements, tels que les bancs de musculation, les haltères, et d'autres matériels de gym.

Les fonctionnalités incluraient :

* Ajouter de nouveaux articles dans le stock, en spécifiant des détails comme le nom de l'article, le type, et la quantité.
* Modifier des articles existants pour mettre à jour leurs informations ou ajuster les quantités.
* Supprimer des articles du stock lorsque ceux-ci ne sont plus disponibles.

## 4. Contraintes techniques

- Base de données : Le système doit se connecter à une base de données relationnelle (comme MySQL, PostgreSQL ou SQLite). La connexion doit être gérée via la classe GymData, qui assure l'exécution des opérations CRUD (Create, Read, Update, Delete) pour les différentes entités.

- Sécurité : Les données sensibles, comme les mots de passe des administrateurs, doivent être stockées de façon sécurisée .

- Extensibilité : Le système doit être suffisamment flexible pour permettre des ajouts futurs, comme la gestion d’autres types d’entités .

### 5. Interfaces utilisateurs (UI)

- Convivialité : L'application doit offrir une interface utilisateur ergonomique, facile à naviguer pour les administrateurs non techniques.

## 6. Tests et Validation

- Test unitaire : Chaque classe et fonctionnalité doit être testée individuellement (Login, Customer, Employee, Appointment).

- Test d’intégration : Vérifier que toutes les pages communiquent bien entre elles et que les données de la base sont correctement manipulées.

- Test utilisateur: Simuler des interactions réelles avec des données pour vérifier l’ergonomie et la bonne exécution de l'application.

## 7. Livrables

- Code source complet et documenté.

- Documentation utilisateur : Un guide pour l'administrateur expliquant comment utiliser l'application.

- Manuel de maintenance : Un manuel pour les développeurs expliquant comment faire évoluer l'application (ajout de nouvelles fonctionnalités, correction de bugs, etc.).

## 8. Planning (Sprints)

Le développement du projet est divisé en plusieurs \*\*sprints\*\* :

- Sprint 1 (01/10/2024 - 18/10/2024) : Développement des classes de base (Person, Customer, Employee, Appointment,Login).

- Sprint 2 (19/10/2024 - 04/11/2024) : Développement des classes CustomerPage,LoginPage et GymData .

- Sprint 3 (05/11/2024 - 22/11/2024) : Développement des classes EmployeePage, PasswordPage et poursuite du développement de GymData.

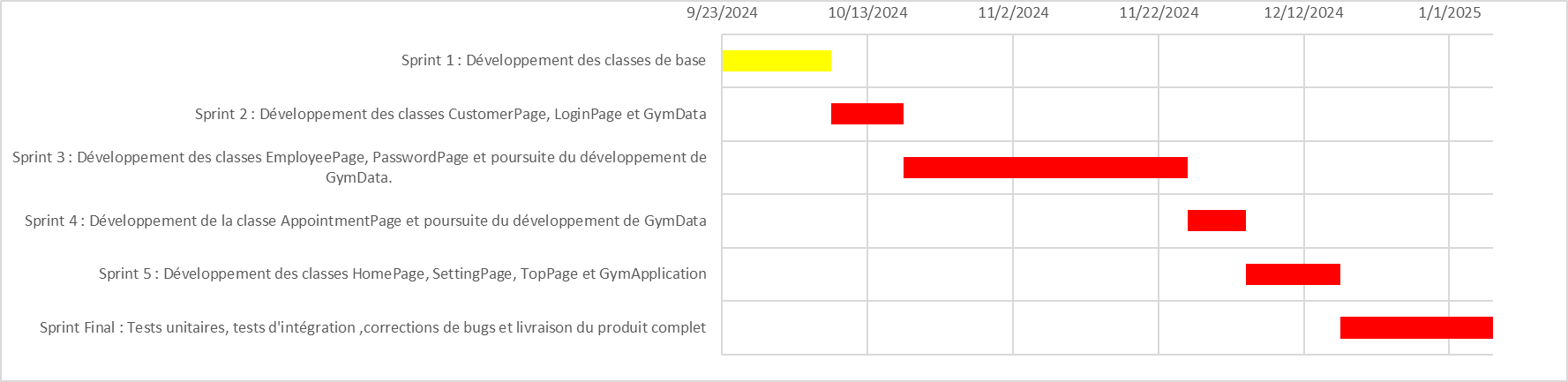
- Sprint 4 (23/11/2024 - 04/12/2024) : Développement de la classe AppointmentPage et poursuite du développement de GymData.

- Sprint 5 (05/12/2024 - 17/12/2024) : Développement des classes HomePage, SettingPage, TopPage et GymApplication

- Sprint final (18/12/2024 - 07/01/2025) : Tests unitaires, tests d'intégration ,corrections de bugs et livraison du produit complet.

# Diagramme de Gantt :

|  |  |  |  |
| --- | --- | --- | --- |
| **Tâches** | **Début** | **Fin** | **Durée** |
| Sprint 1 : Développement des classes de base | 23/09/2024 | 08/10/2024 | 15 |
| Sprint 2 : Développement des classes CustomerPage,LoginPage et GymData . | 08/10/2024 | 18/10/2024 | 10 |
| Sprint 3 : Développement des classes EmployeePage, PasswordPage et poursuite du développement de GymData. | 18/11/2024 | 26/11/2024 | 8 |
| Sprint 4 : Développement de la classe AppointmentPage et poursuite du développement de GymData. | 26/11/2024 | 04/12/2024 | 8 |
| Sprint 5 : Développement des classes HomePage, SettingPage, TopPage et GymApplication | 04/12/2024 | 17/12/2024 | 13 |
| Sprint Final : Tests unitaires, tests d'intégration ,corrections de bugs et livraison du produit complet. | 17/12/2024 | 07/01/2025 | 21 |



# Diagramme de Classe:

