

RAPPORT DE PROJET DE FIN D'ANNÉE

Spécialité : Génie Logiciel

Conception et développement d'une
application pour l'université

Par

YASSINE BOUZEKRI
CHERIF OMAR
KELMY HYBA
GUEDOIR KHALIL
ABID SIRINE
GDOURA AHMED

Année Universitaire : 2023-2024

Table des matières

Introduction Générale	1
1 Cadre de projet	2
1.1 Introduction	2
1.2 Présentation de l'organisme d'accueil	2
1.2.1 Description de l'existant	2
1.3 Critique de l'existant	3
1.4 Solution proposée	3
1.4.1 Identification des besoins fonctionnels	3
1.4.2 Identification des besoins non fonctionnels	4
1.5 Méthodologie de travail	4
1.5.1 Méthodologie Agile	4
1.5.2 Scrum	4
1.6 Conclusion	5
2 Sprint 0	6
2.1 Introduction	6
2.2 Product Backlog	6
2.3 Diagramme de cas d'utilisation	8
2.4 Diagramme de classe d'analyse	8
2.5 Architecture logique globale	9
2.5.1 Couche de Présentation (Front-end)	9
2.5.2 Couche de Services (Back-end)	10
2.5.3 Couche de Persistance (Base de données)	10
2.5.4 Couche d'Intégration (APIs et Services Externes)	10
2.6 Maquettes	11
2.7 Conclusion	12
Conclusion Générale	13

Table des figures

1.1	Logo de l'université	2
1.2	Une itération selon la méthode Scrum[5]	5
2.1	Diagramme de cas d'utilisation	8
2.2	Diagramme de classe d'analyse	9
2.3	Diagramme de cas d'utilisation	11
2.4	Design Interface Login	11
2.5	Design Interface Signup	12
2.6	Design Interface Home	12

Liste des tableaux

2.1	Description détaillée des acteurs	7
-----	---	---

Introduction Générale

Dans un monde en constante évolution, les établissements universitaires se doivent de suivre le rythme en adoptant les nouvelles technologies et en proposant des solutions innovantes. L'Université du Futur se positionne comme un acteur majeur dans cette transformation, en mettant en place une plateforme de services associatifs destinée à enrichir la vie étudiante et à favoriser l'épanouissement des associations universitaires.

L'objectif principal de ce projet est de développer une application web intuitive et multifonctionnelle pour l'Université de ESPRIT, accessible via le site "www.universite.tn". Cette plateforme sera conçue pour répondre aux besoins spécifiques des associations étudiantes, en leur offrant une multitude de services visant à simplifier la gestion et l'organisation de leurs activités.

Les associations étudiantes jouent un rôle crucial dans la vie universitaire. Elles permettent aux étudiants de se regrouper autour de centres d'intérêts communs, de développer des compétences organisationnelles et de leadership, et de créer des liens sociaux forts. Cependant, la gestion de ces associations peut s'avérer complexe et chronophage, nécessitant des outils efficaces pour faciliter la coordination et la communication.

Dans ce cadre, s'inscrit notre projet de fin d'année dont l'objectif est de concevoir et développer une application pour l'université, doté de la plateforme web, une solution informatique assurant la facilité de .

Pour la modélisation de notre projet, nous avons suivi la méthodologie agile "Scrum"

Le présent document définit le travail que nous avons effectué, organisé en quatre chapitres afin d'aboutir à une application fiable et satisfaisante :

- Chapitre 01 intitulé "Cadre de Projet" dont on va présenter une étude de l'existant en proposant une solution qui satisfait les besoins de l'utilisateur.
- Chapitre 02 intitulé "Sprint 0" dont va spécifier le backlog de produit, les diagrammes de cas d'utilisation et de classe d'analyse ainsi l'architecture logique globale.
- Chapitre 03 intitulé "release 1" :
- Chapitre 04 intitulé "release 2" :

Enfin, nous synthétisons notre rapport par une conclusion générale et perspectives.

Chapitre 1

Cadre de projet

1.1 Introduction

Dans notre premier chapitre, nous allons commencer par la présentation de l'organisme d'accueil dans lequel se déroule notre projet de fin d'études.

Par la suite, nous allons aborder l'étude de l'existant en présentant la solution actuelle et ses critiques pour extraire nos objectifs et finalement il est dédié à la présentation de la méthode de travail ainsi les outils utilisés pour la réalisation de notre projet.

1.2 Présentation de l'organisme d'accueil

L'Université, ...



FIGURE 1.1 – Logo de l'université

1.2.1 Description de l'existant

La solution web que nous allons développer est dédiée à un grand nombre de services destinés à améliorer et à simplifier la gestion des associations étudiantes. De ce fait, l'Université a pris l'initiative de proposer des solutions web et mobiles utilisant les nouvelles techniques informatiques pour répondre aux besoins croissants de ses utilisateurs.

Actuellement, les associations et clubs étudiants doivent souvent se tourner vers des outils disparates et peu intégrés, tels que des feuilles de calcul, des groupes de messagerie, et des applications de réservation indépendantes, pour gérer leurs activités quotidiennes. Cette fragmentation des outils rend la coordination et la communication moins efficaces et plus chronophages.

1.3 Critique de l'existant

Actuellement, les étudiants utilisent une combinaison d'e-mails, de réseaux sociaux, et de plateformes de messagerie pour communiquer, ce qui peut entraîner des pertes d'informations et des malentendus. Les réservations d'espaces se font souvent par des demandes manuelles ou via des systèmes non intégrés, nécessitant une intervention administrative lourde et ralentissant les processus. De plus, l'organisation d'événements et d'activités de team-building est souvent entravée par un manque de visibilité et de coordination, ce qui réduit la participation et l'engagement des étudiants.

1.4 Solution proposée

Pour répondre aux défis et aux limitations des solutions existantes, l'Université de [nom de l'université] propose de développer une plateforme web et mobile intégrée, dédiée à la gestion et à l'optimisation des services associatifs. Cette solution innovante utilisera les nouvelles technologies informatiques pour offrir une expérience utilisateur fluide et répondre aux besoins spécifiques des associations étudiantes.

1.4.1 Identification des besoins fonctionnels

Les besoins fonctionnels représentent les principales fonctionnalités du système. Vers la recherche d'une perfection, notre application devra répondre aux besoins suivants :

- **La gestion des clubs** : Permet aux responsables la création, la consultation, la modification et la suppression d'un club, ainsi que l'acceptation des demandes de participation des utilisateurs. Permet aux utilisateurs de participer aux clubs et les consulter.
- **La gestion des evenements** : Permet aux responsables la création, la modification et la suppression des evenements, ainsi que l'acceptation des demandes de participation des utilisateurs. Permet aux simples utilisateurs de participer et consulter les evenements.
- **La gestion des reservations** : Permet aux responsables des reservations de consulter, ajouter, modifier ou supprimer une réservation. Permet aux responsables de evenements de demander la reservation d'une salle.
- **La gestion de Team Buildings** : Permet aux responsables de consulter, ajouter, supprimer ou modifier des teams buildings. Permet aux utilisateurs de participer et consulter les team buildings.
- **La gestion des covoiturages** : Permet aux chauffeurs de publier des covoiturages, aux utilisateurs de participer, et aux responsables de gérer les chauffeurs.
- **La gestion des utilisateurs** : Permet à l'administrateur de gérer utilisateurs, modifier le statut d'un utilisateur entre un simple utilisateur et un responsable.
- **La gestion des status** : L'administrateur peut modifier en activé et désactivé le statut d'un club, un evenement, une reservation, ou un team building.

1.4.2 Identification des besoins non fonctionnels

Ces besoins sont les contraintes techniques exigées et les fonctionnalités nécessaires pour rendre l'application plus performante. Pour bien répondre aux besoins des utilisateurs, notre futur système doit répondre aux critères suivants :

- **Fiabilité et efficacité** : L'application devra être disponible et opérationnelle à tout moment et doit assurer un fonctionnement fiable ainsi qu'un temps de réponse de l'ordre de temps réel.
- **Evolutivité** : L'application devra être susceptible à des éventuelles améliorations.
- **Ergonomie** : Les interfaces ainsi que les messages d'aide doivent être claires et faciles à comprendre par l'utilisateur.
- **Sécurité** : L'accès aux informations doit être possible seulement après la vérification des droits d'accès. Ainsi, tout utilisateur doit passer par une phase d'authentification à fin de pouvoir consulter les services offerts par l'application.

1.5 Méthodologie de travail

1.5.1 Méthodologie Agile

La méthodologie Agile s'oppose généralement aux méthodologies traditionnelles de gestion de projet. Cette méthode place les besoins du client au centre des priorités et privilégie le dialogue entre toutes les parties prenantes du projet.

A l'origine, cette approche a été créée pour les projets de développement web et informatique. Aujourd'hui, la méthode Agile est de plus en plus répandue car elle est adaptable à de nombreux types de projets, tous secteurs confondus.[5]

1.5.2 Scrum

Scrum est la méthodologie de travail pour le développement et la maintenance de produits complexes permettant de répondre à des problèmes complexes et changeants, tout en livrant de manière créative et productive des produits de la plus grande valeur possible. Scrum utilise une approche incrémentale à fin optimiser la prédictibilité et pour contrôler le risque[6].

Sprint

La planification du sprint comprend l'identification des points prioritaires que l'équipe pense pouvoir réaliser au cours du sprint.

La revue du sprint a lieu à la fin de chaque sprint où l'équipe de développement présente les fonctionnalités terminées. L'incrément produit est éventuellement livrable et la mise en place du prochain sprint peut être anticipée.

La rétrospective du sprint permet d'améliorer le processus de développement des sprints suivants en faisant un point sur le sprint réalisé.

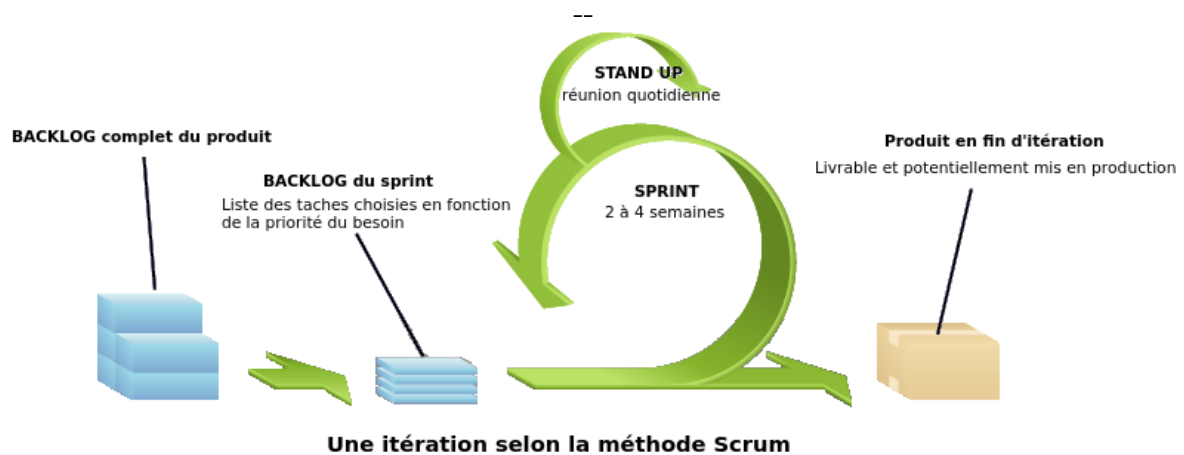


FIGURE 1.2 – Une itération selon la méthode Scrum[5]

1.6 Conclusion

Ce chapitre présente un aperçu sur le contexte de notre projet de fin d'études. Nous avons présenté l'organisme d'accueil, puis nous avons fait une définition générale à propos le marketplace dont nous avons fait sorti une problématique et nous avons critiqué les solutions existant pour pouvoir atteindre une solution plus pratique, conviviale et innovante.

Enfin, nous avons présenté la méthode de travail et les outils de développement qui nous ont permis de réaliser ce projet.

Chapitre 2

Sprint 0

2.1 Introduction

Dans ce chapitre nous allons décrire le sprint 0, commençant par le « Product Backlog » ainsi que la planification des sprints. Aussi nous allons présenter les besoins de notre système de manière formelle par le diagramme de cas d'utilisation et le diagramme de classe analyse

2.2 Product Backlog

Le Product Backlog est l'artefact le plus important de Scrum, c'est l'ensemble des caractéristiques fonctionnelles ou techniques qui constituent le produit souhaité. Les caractéristiques fonctionnelles sont appelées des histoires utilisateurs (user story) et les caractéristiques techniques sont appelées des histoires techniques (Technical story). Le Tableau ci-dessous résume le product backlog de notre application. Il est à noter que nous n'avons pas cité les histoires techniques comme la préparation de la maquette graphique, les travaux de conception et les jeux de tests, etc...

Backlog de produit	Priorité	Estimation	Planification
En tant qu'utilisateur, je peux m'inscrire	1	Moyen	
En tant qu'utilisateur, je peux m'authentifier	1	Moyen	
En tant qu'administrateur, je peux changer le role d'un utilisateur	1	Moyen	
En tant qu'administrateur, je peux changer le statut d'un club, evenement ou team building	1	Moyen	
En tant que responsable club, je peux créer, modifier ou mettre en pause un club	1	Moyen	
En tant que responsable evenement, je peux creer, modifier ou annuler un evenement	1	Moyen	
En tant que responsable covoiturage, je peux donner la main a des chauffeurs de publier des covoiturations	1	Moyen	
En tant que responsable Team-Building, je peux creer, modifier ou annuler un Team-Building	1	Moyen	
En tant que responsable Team-Building, je peux accepter des membres pour le Team-Building	1	Moyen	
En tant que responsable club, je peux accepter des membres pour le club	2	Moyen	
En tant que responsable evenement, je peux accepter des membres pour l'evenement	2	Moyen	
En tant que responsable covoiturage, je peux donner la main a des chauffeurs de publier des covoiturations	2	Moyen	
En tant qu'utilisateur, je peux publier, chercher, annuler, et consulter les covoiturations	2	Moyen	
En tant qu'utilisateur, je peux consulter ou demander de participer à un evenement	3	Moyen	
En tant qu'utilisateur, je peux consulter ou demander de participer à un Team-Building	3	Moyen	
En tant que responsable reservations, je peux gerer les reservations pour les evenements ainsi que les team buildings	3	Moyen	

--

TABLE 2.1 – Description détaillée des acteurs

2.3 Diagramme de cas d'utilisation

Dans cette section nous présentons les besoins de notre système de manière formelle. C'est à dire en utilisant le diagramme des cas d'utilisation global. Les fonctionnalités globales offertes par l'application sont représentées dans le diagramme ci- dessous. Tous les cas d'utilisation nécessitent une authentification comme précondition.

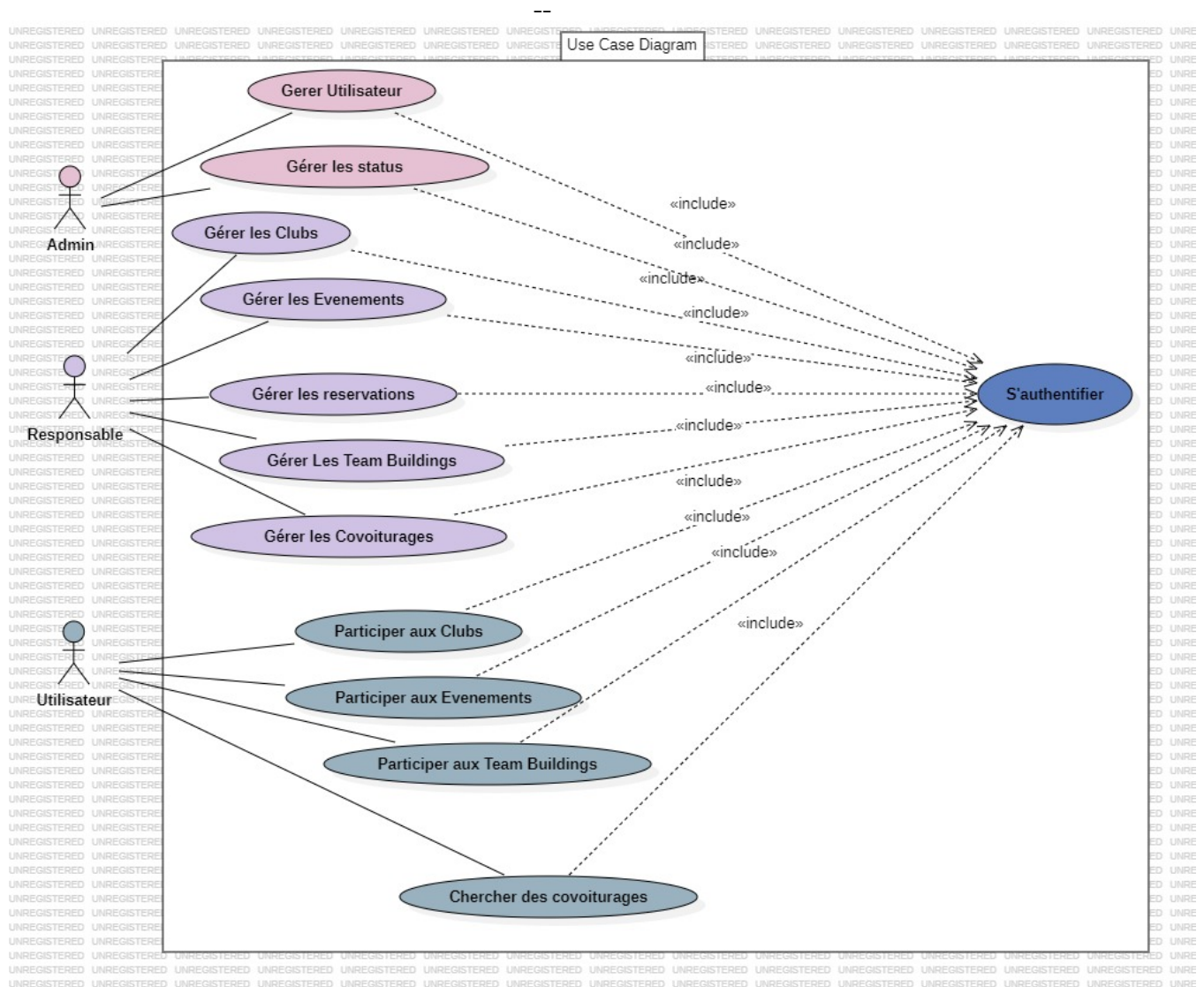


FIGURE 2.1 – Diagramme de cas d'utilisation

2.4 Diagramme de classe d'analyse

Le diagramme de classe est la représentation de la structure statique en terme de classes et de relations. Après avoir décrit l'ensemble des cas d'utilisation, nous aboutissons au diagramme de classe d'analyse suivant :

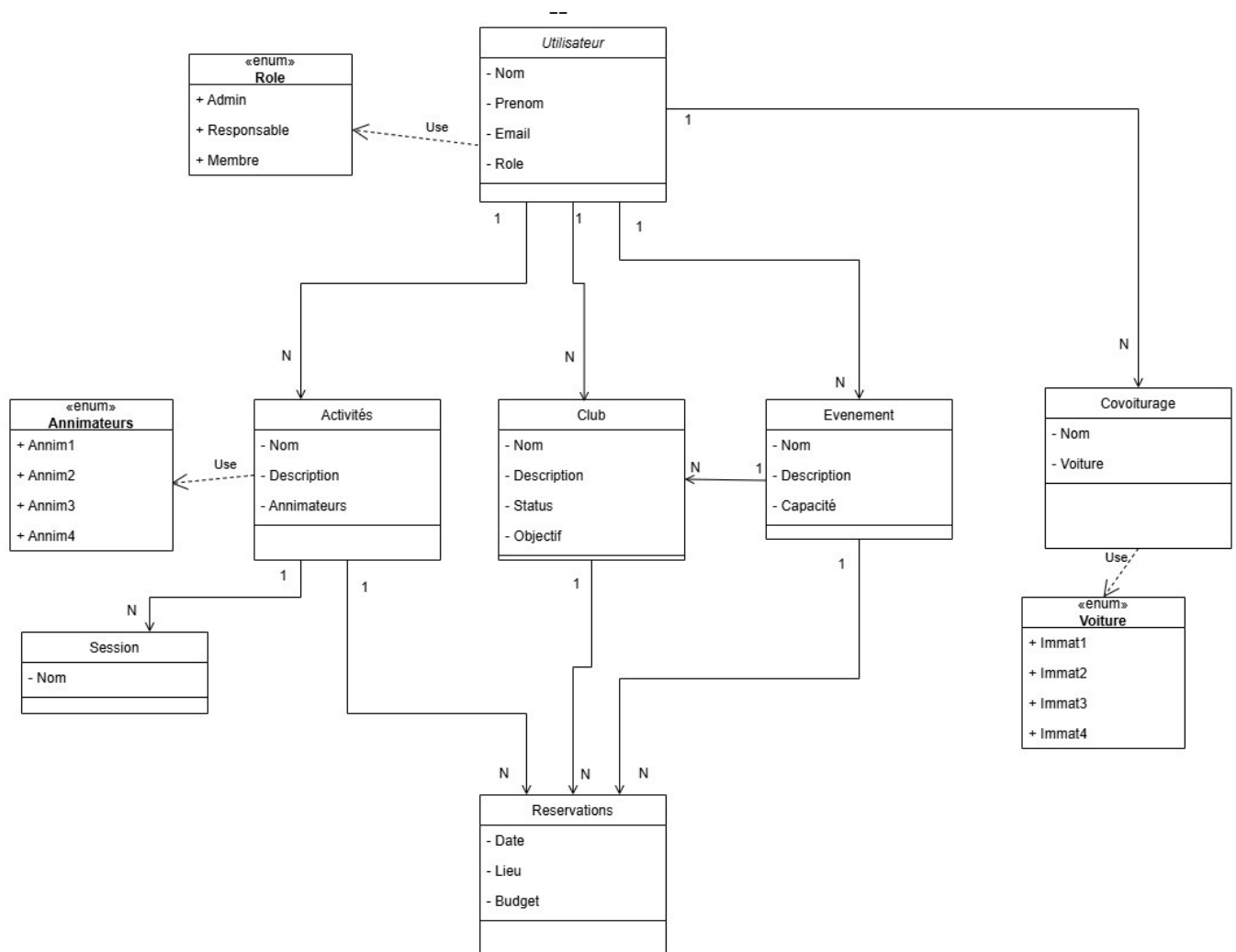


FIGURE 2.2 – Diagramme de classe d'analyse

2.5 Architecture logique globale

L'architecture logique représente la décomposition logique du projet en couches. Pour développer la solution, nous utiliserons Angular pour la partie front-end et Spring Boot pour la partie back-end. Cette architecture en couches permettra de structurer le projet de manière modulaire et de faciliter la maintenance, l'évolutivité et la réutilisabilité des composants. Voici une vue d'ensemble de l'architecture logique :

2.5.1 Couche de Présentation (Front-end)

Technologie : Angular

Cette couche est responsable de l'interface utilisateur (UI) et de l'expérience utilisateur (UX). Elle comprend les composants Angular, les services, et les modules nécessaires pour afficher les données et interagir avec les utilisateurs. Les principales responsabilités de cette couche incluent :

- Affichage des pages web et des interfaces utilisateur.

- Gestion de la navigation et des routes.
- Interactions avec les utilisateurs via des formulaires et des événements.
- Communication avec la couche de services via des appels HTTP.

2.5.2 Couche de Services (Back-end)

Technologie : Spring Boot

Cette couche gère la logique métier et les règles d'application. Elle comprend les contrôleurs, les services, et les gestionnaires d'entités qui orchestrent les opérations et les transactions. Les principales responsabilités de cette couche incluent :

- Traitement des requêtes HTTP reçues du front-end.
- Application des règles métiers et de la logique d'application.
- Gestion des transactions et des opérations de données.
- Communication avec la couche de persistance pour accéder aux données.

2.5.3 Couche de Persistance (Base de données)

Cette couche est responsable de l'accès aux données et de leur gestion. Elle comprend les entités, les repositories, et les configurations de la base de données. Les principales responsabilités de cette couche incluent :

- Modélisation des entités et des relations de la base de données.
- Accès aux données et exécution des requêtes SQL.
- Gestion des connexions et des transactions de la base de données.
- Migrations de schéma de base de données et initialisation des données.

2.5.4 Couche d'Intégration (APIs et Services Externes)

Cette couche gère les interactions avec les services externes et les APIs. Elle comprend les intégrations avec les services tiers et les échanges de données entre le système et les systèmes externes. Les principales responsabilités de cette couche incluent :

- Communication avec les services externes via des APIs RESTful.
- Gestion des protocoles de communication et des formats de données.
- Sécurité et authentification pour les appels de services externes.
- Transformation et validation des données échangées.

La (figure) illustre une représentation de l'architecture globale de l'application :

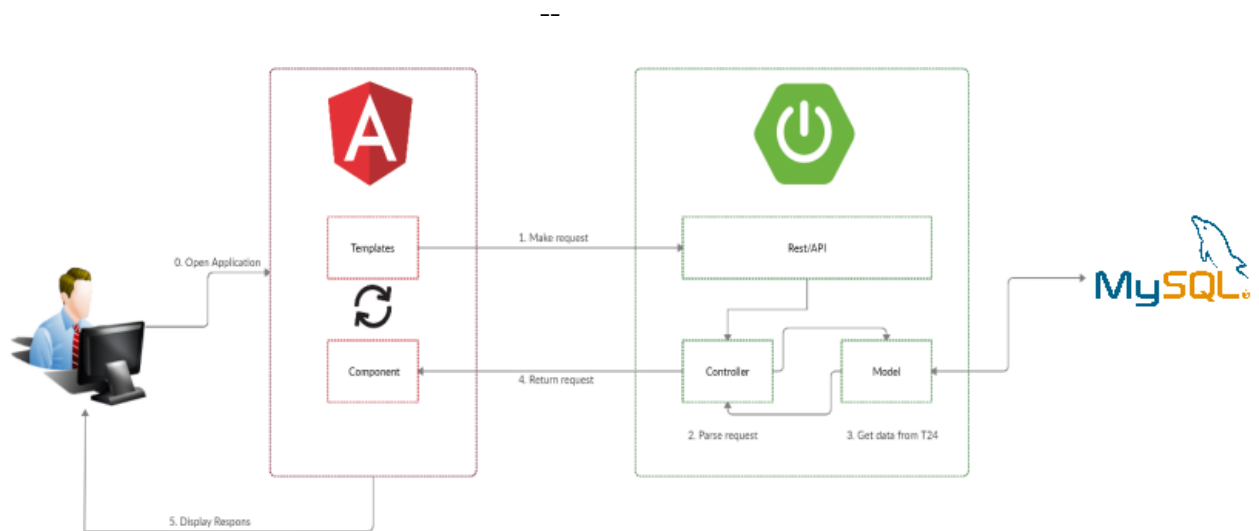


FIGURE 2.3 – Diagramme de cas d'utilisation

2.6 Maquettes

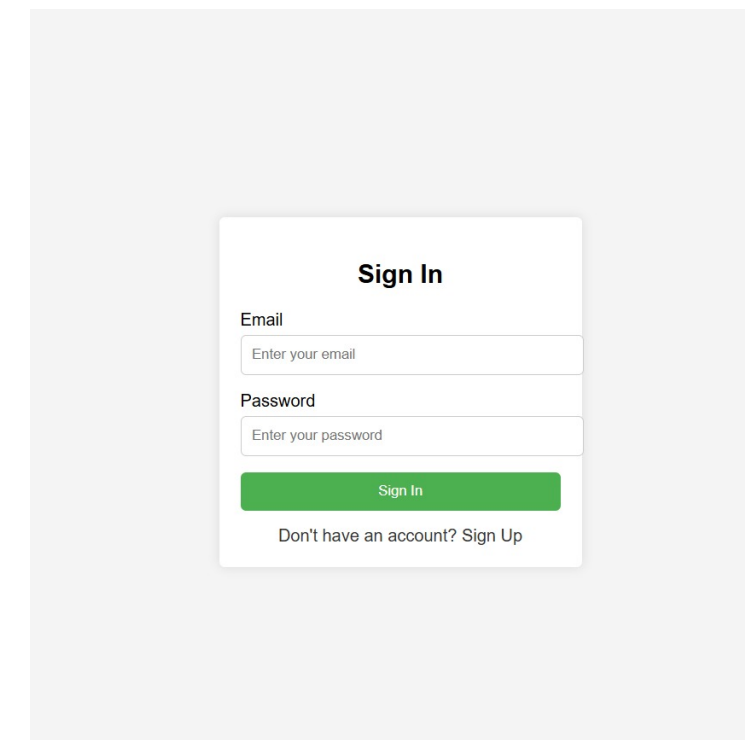
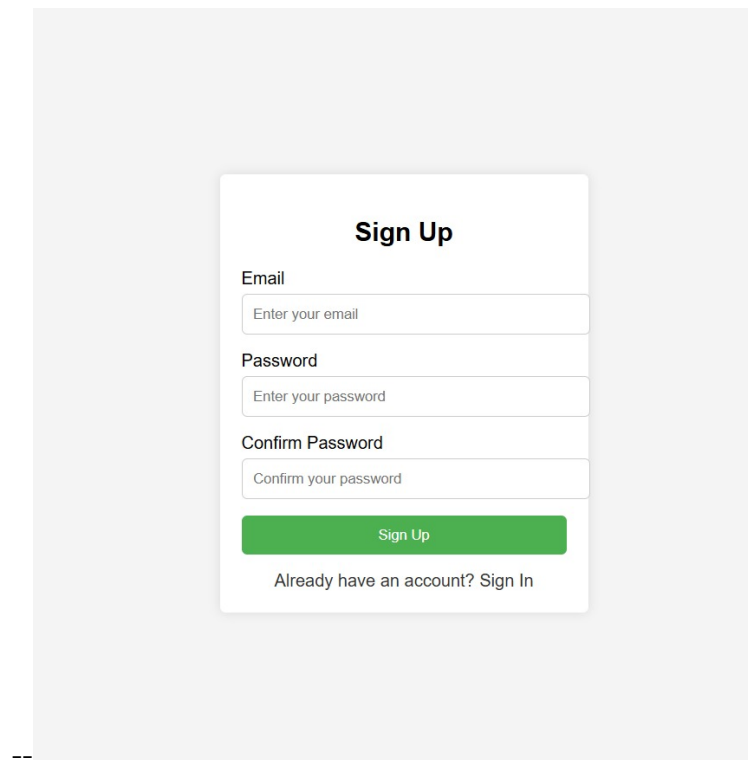


FIGURE 2.4 – Design Interface Login



The image shows a 'Sign Up' form centered on a light gray background. The form is a white card with a green 'Sign Up' button. It includes fields for Email, Password, and Confirm Password, each with a placeholder text. Below the button is a link for users who already have an account.

Sign Up

Email
Enter your email

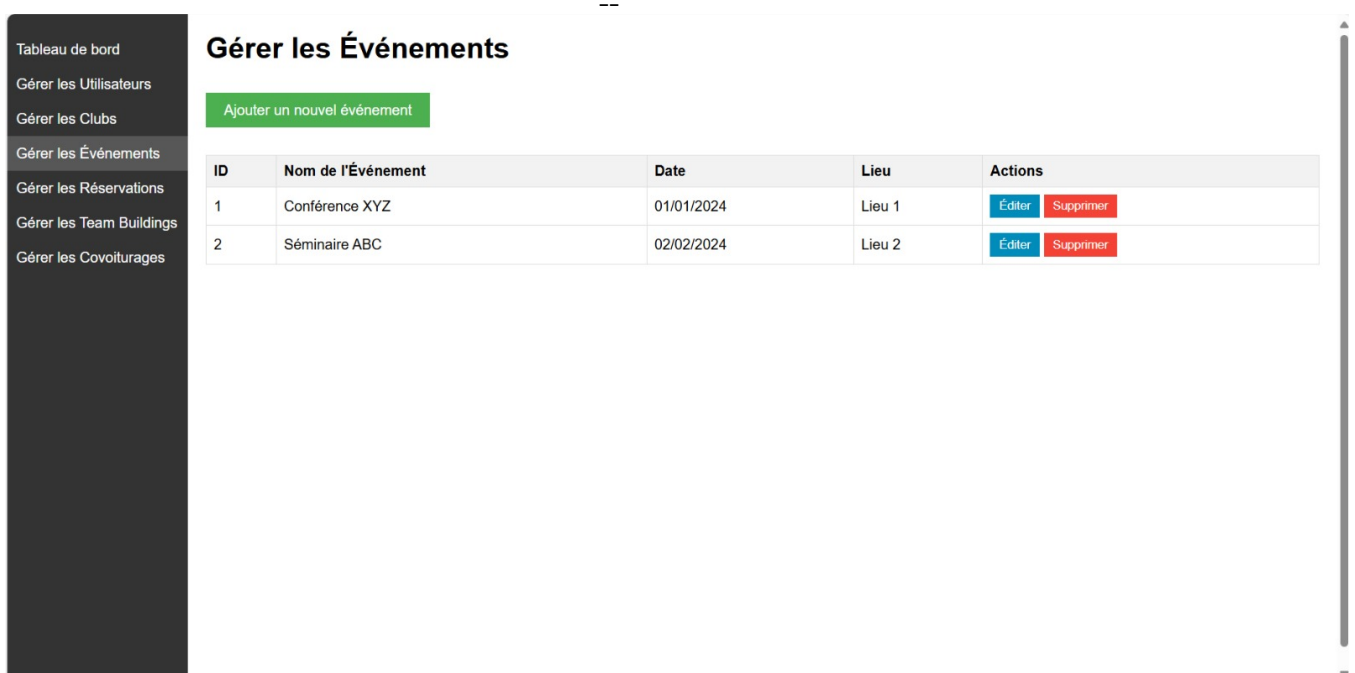
Password
Enter your password

Confirm Password
Confirm your password

Sign Up

Already have an account? Sign In

FIGURE 2.5 – Design Interface Signup



The image shows a dashboard interface for managing events. On the left is a dark sidebar with navigation links. The main content area has a title 'Gérer les Événements' and a green button to add a new event. Below this is a table with two rows of event data, each with 'Éditer' and 'Supprimer' action buttons.

Gérer les Événements

Ajouter un nouvel événement

ID	Nom de l'Événement	Date	Lieu	Actions
1	Conférence XYZ	01/01/2024	Lieu 1	Éditer Supprimer
2	Séminaire ABC	02/02/2024	Lieu 2	Éditer Supprimer

FIGURE 2.6 – Design Interface Home

2.7 Conclusion

Dans ce chapitre nous avons présenté le Product Backlog, le diagramme de cas d'utilisation global et le diagramme de classes d'analyse ainsi que la planification des sprints. Nous allons détailler dans le prochain chapitre le sprint 1.

Conclusion Générale

Au terme de ce rapport, nous pouvons conclure que ce stage de fin d'année nous a donné une occasion opportune qui permet de confronter l'acquis théorique à l'environnement pratique.

Bibliographie

- [1] Cours de Dr TEBOURSKI.Wafa intitulé
"Méthodologie de conception orienté objet"
Consulté le 22/05/2024.

Netographie

- [1] Méthode Agile Sprint
<https://fr.mailjet.com/blog/news/methode-agile-scrum/>
- [2] Scrum
<https://www.planzone.fr/blog/quest-ce-que-la-methodologie-scrum>
- [3] besoin non fonctionnel,
<https://savoir.plus/besoins-fonctionnels-non-fonctionnels/>
- [4] Compatibilité et portabilité
<https://fr.wikipedia.org/wiki/Qualit>

Consulté le 23/05/2024.