Table des matières

Table des figures

Liste des tableaux

Chapitre 1

Étude et analyse des besoins

Introduction

Ce premier chapitre est dédié à l'étude préalable de projet. nous commence par présenter l'organisme d'accueil ainsi que le service auquel on était affectées. Ensuite, nous expose la problématique et les objectifs, suivis de l'analyse et de la critique de l'existant. Enfin, nous identifie les besoins et on définit la méthodologie de développement qui sera adoptée.

1.1 Présentation de l'organisme d'accueil

Nous présente dans cette section l'organisme d'accueil en exposant ses services.

1.1.1 Présentation de l'entreprise d'accueil : Tradrly

Située au cœur de Mahdia, en Tunisie, cette startup se distingue par son engagement dans la conception de solutions technologiques innovantes, intégrant les technologies Web, mobiles et l'intelligence artificielle.

Sa mission est d'accompagner les entreprises dans la concrétisation de leurs idées en développant des solutions sur mesure adaptées à leurs besoins spécifiques. Grâce à une équipe d'experts passionnés par la transformation numérique, elle propose des services de développement avancés destinés aussi bien aux startups qu'aux entreprises établies.

Son approche vise à garantir des parcours numériques optimisés, en mettant l'accent sur la qualité et la performance des solutions proposées. Chaque projet est mené avec l'objectif de dépasser les attentes des clients, en apportant une valeur ajoutée à leurs activités. À travers cette démarche, elle contribue activement à l'évolution des technologies et à l'innovation dans le domaine du numérique

FIGURE 1.1 – Logo Tradrly

1.1.2 Services de Tradrly

Parmi les services fournis par Tradrly, on cite :

- Technologie .
- information et Internet.

1.2 Cadre de projet

Dans cette section, nous analyserons la problématique afin de définir clairement les objectifs à atteindre.

1.2.1 Problématique

Dans un écosystème entrepreneurial en constante évolution, les startups rencontrent des défis majeurs pour accroître leur visibilité, structurer leurs informations et attirer les talents adéquats. D'un côté, les entrepreneurs et investisseurs recherchent un espace centralisé pour découvrir des startups, leurs domaines d'activité, leurs projets et leurs offres d'emploi et de stage. De l'autre, les chercheurs d'emploi et les talents souhaitent accéder facilement aux offres, qu'elles soient d'emploi ou de stage, et postuler efficacement.

Comment concevoir une plateforme web intuitive et dynamique qui centralise les informations des startups, facilite la diffusion des offres d'emploi et de stage, et optimise le processus de candidature pour les talents, tout en favorisant l'engagement et les interactions entre les parties prenantes ?

1.2.2 Objectifs du projet

Dans le cadre de ce projet de fin d'études et afin d'apporter une solution à la problématique décrite ci-dessous, nous proposons le développement d'une application web permettant de :

- **Centralisation des informations sur les startups :** Fournir une plateforme unique regroupant les informations essentielles sur les startups (domaines d'activité, solutions, équipe, etc.).
- **Facilitation du recrutement et des candidatures :** Publier et gérer les offres d'emploi ou stage avec des descriptions détaillées pour faciliter la candidature directement via la plateforme.

- **Filtrage des CV avec un modèle IA :** Implémenter un système de tri automatique des CV basé sur l'intelligence artificielle, permettant d'analyser et de classer les candidatures en fonction de leur pertinence par rapport aux offres d'emploi.
- **Évaluation des candidats via un système de test QCM :** Intégrer un test de présélection sous forme de QCM avec une évaluation automatisée des résultats pour faciliter la sélection des candidats qualifiés.

1.3 Étude de l'existant

Cette étape est considérée comme l'une des principales étapes lors de la réalisation d'un nouveau projet.Durant cette phase, nous avons mené une démarche de collecte d'informations sur les solutions existantes répondant à la problématique, dans le but d'identifier leurs points faibles. Cette analyse nous permettra d'éviter ces écueils tout en offrant un ensemble de fonctionnalités répondant aux exigences des utilisateurs.

1.3.1 Analyse de l'existant

En s'appuyant sur les objectifs définis, nous avons identifié plusieurs outils existants susceptibles de répondre aux besoins exprimés. Ces outils seront présentés dans la suite de cette section.

1.3.1.1 wind-consulting

Fondée en 2016, Wind Consulting accompagne les entreprises dans leur transformation digitale en mettant à leur disposition son expertise dans deux domaines clés : le conseil et le digital. En tant que partenaire stratégique, l'entreprise propose des solutions innovantes adaptées aux enjeux d'agilité, de performance et de développement de ses clients.
Ce site propose les fonctionnalités suivantes :

- La publication d'actualités.
- La présentation des services proposés.
- La diffusion d'offres d'emploi.
- Soumettre une candidature à une offre d'emploi.
- La mise en ligne d'offres de stage.
- Le partage d'informations sur les technologies émergentes.

1.3.1.2 satoripop

Satoripop conçoit des écosystèmes logiciels de pointe qui évoluent avec votre entreprise et maximisent votre retour sur investissement.

FIGURE 1.2 – Logo "wind-consulting"

L'entreprise transforme les défis complexes en opportunités en créant des solutions sur mesure, de bout en bout, qui donnent vie à votre vision—que vous soyez en phase de croissance avec une startup ou en quête d'optimisation pour une grande entreprise.

Du retail à la banque, et bien plus encore, Satoripop combine créativité, expertise technologique et design centré sur l'utilisateur pour offrir des solutions prêtes pour l'avenir, propulsant ainsi la croissance et redéfinissant les secteurs d'activité. Ce site propose les fonctionnalités suivantes :

- Présentation des services proposés.
- Présentation des solutions.
- Diffusion des offres d'emploi.
- Soumettre une candidature à une offre d'emploi.
- Postuler spontanément, même sans offre d'emploi disponible.

FIGURE 1.3 – Logo "satoripop"

1.3.1.3 Anypli

Anypli est une agence de communication digitale et de développement web et mobile composée de plus de 70 collaborateurs.

Grâce à une collaboration étroite avec ses partenaires, Anypli accompagne les entreprises dans leur développement en ligne, en élaborant des stratégies webmarketing sur-mesure et en proposant des solutions adaptées à leurs objectifs. L'agence se base sur les dernières tendances digitales et les comportements des clients pour maximiser l'impact des actions menées.

- Présentation des services proposés.
- Présentation des solutions adaptées.

FIGURE 1.4 – Logo "Anypli"

1.3.2 Critique de l'existant

Les applications Wind Consulting, Satoripop et Anypli offrent des services utiles, mais manquent de fonctionnalités interactives et communautaires. Wind Consulting se concentre sur les offres d'emploi et de stage sans favoriser l'engagement des utilisateurs. Satoripop ne permet pas une personnalisation avancée des offres, et l'absence de postulation spontanée limite son attrait. Anypli, bien qu'axée sur les services proposés, ne facilite pas les interactions entre recruteurs et candidats et manque de suivi des candidatures, rendant l'expérience utilisateur moins fluide et dynamique.
Le tableau 1.1 présente une étude comparative de ces plateformes en analysant leurs principaux avantages et limitations.

TABLE 1.1 – tableau comparatif

Fonctionnalités	**wind-consulting**	**satoripop**	**Anypli**
La publication d'actualités.	✓	✗	✗
La présentation des services proposés	✓	✓	✓
La diffusion d'offres d'emploi	✓	✓	✗
Soumettre une candidature à une offre d'emploi	✓	✓	✗
La mise en ligne d'offres de stage	✓	✗	✗
Le partage d'informations sur les technologies émergentes	✓	✗	✗
Présentation des solutions	✗	✓	✓
Postuler spontanément, même sans offre d'emploi disponible	✗	✓	✗

1.3.3 Solution envisagée

Pour pallier les limitations des applications existantes et répondre aux besoins spécifiques de notre startup, nous proposons de développer une application web dédiée. Cette plateforme centralisera des informations clés sur notre entreprise, notamment nos domaines d'activité, les solutions offertes et les membres de notre équipe. Elle permettra également la publication et la gestion détaillée des offres d'emploi et de stage, facilitant ainsi les candidatures directes des utilisateurs. De plus, un système de filtrage des CV basé sur l'intelligence artificielle analysera et classera automatiquement les candidatures en fonction de leur adéquation avec les postes proposés. Enfin, l'intégration de tests de présélection sous forme de questionnaires à choix multiples, avec une évaluation automatisée des résultats, aidera à identifier rapidement les candidats les plus qualifiés.

1.3.4 Méthodologie agile

Pour répondre aux exigences de notre projet, nous avons opté pour un processus de développement agile qui est plus efficace et moins rigide que les méthodes classiques. Les méthodes agiles [9] offrent une plus grande flexibilité et une meilleure visibilité dans la gestion du projet, ce qui permet de générer un produit de haute qualité. De plus, elles permettent à l'équipe de parcourir les risques beaucoup plus rapidement qu'avec les outils de gestion de projet traditionnels.

1.3.5 Méthode Scrum

SCRUM illustrée par la figure 1.5, est une méthode agile qui est définie par un ensemble de pratiques qui doivent être inclues dans la phase de développement. SCRUM est subdivisée en des blocs temporaires appelés Sprint, qui durent généralement de deux à quatre semaines. Chaque Sprint est une entité à part et est planifié collectivement par tous les membres de l'équipe. Durant les Sprints, une réunion quotidienne se déroule pendant quinze minutes où chaque membre discute les tâches qu'il a réalisé, les différents obstacles qu'il a rencontrés et son futur plan pour les vingt-quatre heures à venir. [10].

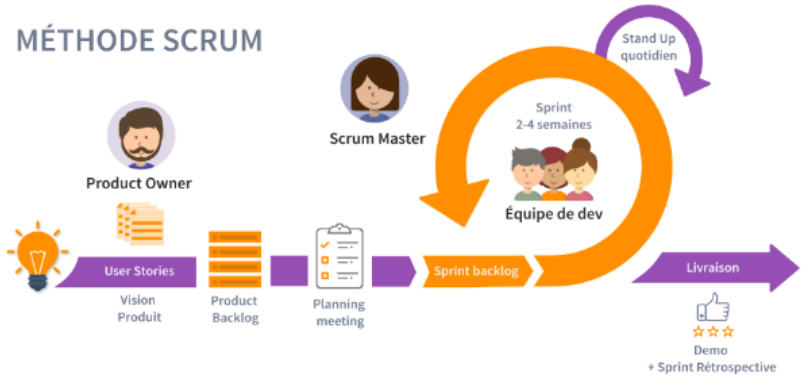

FIGURE 1.5 – Le processus Scrum

Scrum définit 3 rôles qui sont :

- **Product Owner :** Le Product Owner est le représentant du client au sein de l'équipe projet. Il a pour responsabilité de définir et de hiérarchiser les fonctionnalités du produit, ainsi que de décider de la date et du contenu de chaque sprint, en se basant sur les estimations de charge fournies par l'équipe.
- **Scrum Master :** Son rôle est de veiller à ce que l'équipe Scrum suive correctement la méthodologie Scrum, d'éliminer les obstacles qui entravent le travail de l'équipe, et de faciliter la communication et la collaboration au sein de l'équipe pour atteindre les 11 objectifs du projet. Le Scrum Master agit comme un mentor et un défenseur de la méthodologie Scrum pour assurer son efficacité.
- **Équipe de développement :** L'équipe Scrum englobe tous les rôles traditionnellement requis dans un projet, y compris les concepteurs, les développeurs, les testeurs, et autres. Cette équipe est autoorganisée et demeure stable tout au long de la durée d'un sprint .

1.4 Planification

Le diagramme de Gantt [11] est un outil couramment employé en ordonnancement et gestion de projet pour afficher chronologiquement les différentes phases par lesquelles passent un projet. Il offre une représentation visuelle de sa progression.Pour notre cas, le diagramme de GANTT illustré par la figure 1.6 est composé de 7 phases incluant la phase préliminaire du projet, 4 sprints ainsi que la rédaction du rapport.

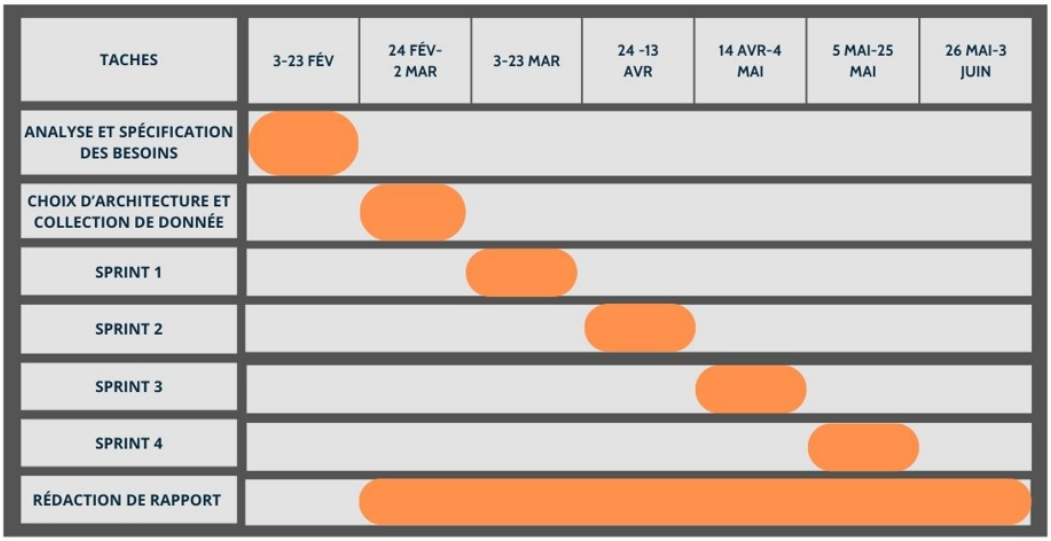

FIGURE 1.6 – Diagramme de GANTT

1.5 Conclusion

Dans ce premier chapitre, nous avons commencé par la présentation de l'organisme d'accueil pour notre projet de fin d'études. Par la suite, nous avons exposé en détail la problématique qui a constitué le point de départ de notre mission, tout en dévoilant la solution que nous avons élaborée pour y répondre, ultérieurement les objectifs que nous nous sommes fixés. Finalement nous avons clôturer ce chapitre en exposant la méthodologie que nous avons consciencieusement adoptée tout au long de notre stage

Chapitre 2

Spécification des besoins

Introduction

Dans ce chapitre nous intéressons de l'étape de spécifications de besoins en effet nous allons Identifier les différents acteurs de notre application et analyser les besoins fonctionnels et non fonctionnels. Ces besoins seront par la suite exprimés sous forme des diagrammes de cas d'utilisation.

2.1 Analyse des besoins

Cette étape vise à définir précisément l'objectif de notre travail en identifiant clairement les besoins à satisfaire. Nous commencerons par présenter les besoins fonctionnels, puis nous aborderons les besoins non fonctionnels.

2.1.1 Identification des acteurs

Un acteur représente un rôle joué par une entité externe interagissant directement avec le système étudié. Cette entité peut être un être humain, un matériel ou tout autre système interagissant avec le système modélisé. L'activité du système vise à satisfaire les besoins de ses acteurs. Dans notre solution, nous identifions deux acteurs principaux

- **Candidat :** une personne en recherche d'emploi ou de stage, pouvant postuler pour des offres spécifiques, soumettre une candidature spontanée ou être invitée à passer des tests ou évaluations dans le cadre du processus de recrutement ou de développement professionnel. Après avoir postulé à une offre, il peut consulter et suivre l'état d'avancement de ses candidatures via la plateforme dédiée, pour chaque offre à laquelle il a postulé.
- **L'administrateur :** ce l'administrateur de l'application web incluent la gestion complète du site officiel de la société, permettant la mise à jour des informations (à propos, services, equipe), la

publication et la modification d'offres d'emploi ou de stage, ainsi que la gestion des candidatures (consultation, filtrage, mise à jour des statuts). L'admin peut inviter les candidats à passer un test d'évaluation pour évaluer leurs compétences .

2.1.2 Besoins fonctionnels

Les besoins fonctionnels de notre application web englobent diverses fonctionnalités visant à garantir une expérience utilisateur complète et efficiente. Parmi ces éléments, nous identifions :

- **Besoins fonctionnels pour Candidat :**

Pour cet acteur, notre solution doit offrir les fonctionnalités suivantes :

Parcourir tout le contenu public Le candidat peut accéder librement à l'ensemble des informations publiques disponibles sur le site officiel de la société, telles que les sections « Équipe», « Services » ou « Contact », afin de mieux comprendre l'entreprise et ses activités.

Parcourir les offres d'emploi et de stage Le candidat peut consulter la liste des offres d'emploi et de stage publiées sur la plateforme

Postuler aux offres d'emploi ou de stage Le candidat peut soumettre sa candidature à une offre spécifique en fournissant son CV et ses informations personnelles directement via la plateforme, ou il peut envoyer une candidature spontanée sans se référer à une offre particulière.

Participer aux tests QCM Le candidat peut être invité à passer des tests sous forme de questionnaires à choix multiples (QCM) pour évaluer ses compétences techniques ou professionnelles, dans le cadre du processus de recrutement ou d'évaluation.

Suivre leurs candidatures Le candidat peut consulter et suivre l'état d'avancement de chacune de ses candidatures pour chaque offre à laquelle il a postulé, via un espace personnel sur la plateforme.

- **Besoins fonctionnels pour l'administrateur :**

Pour cet acteur, notre solution doit offrir les fonctionnalités suivantes :

Gérer et mettre à jour tout le contenu du site Web L'administrateur peut modifier, ajouter ou supprimer le contenu public du site officiel de la société, incluant les sections telles que « Équipe », « Services », ou « Contact », ainsi que les fichiers multimédias (images, vidéos, documents), pour garantir des informations à jour et cohérentes.

Gérer les offres d'emploi L'administrateur peut créer, modifier, supprimer ou archiver des offres d'emploi, en précisant les détails (titre, description, compétences, localisation, type de contrat) et gérer leur visibilité (publication, brouillon, désactivation) sur la plateforme.

Gérer les offres de stage L'administrateur peut publier, éditer, supprimer ou archiver des offres de stage, en définissant les critères spécifiques (durée, domaine, prérequis) et contrôler leur accessibilité sur le site, de manière similaire à la gestion des offres d'emploi.

Configurer et surveiller les tests QCM L'administrateur peut concevoir, paramétrer et attribuer des tests sous forme de questionnaires à choix multiples (QCM) aux candidats, suivre leur progression et analyser les résultats pour évaluer leurs compétences dans le cadre du processus de recrutement.

Gérer les candidats L'administrateur peut consulter les candidatures soumises, filtrer et trier les profils selon des critères (offre, statut, date), mettre à jour l'état des candidatures (passe un test, refusée), inviter les candidats à passer des tests .

Voir le tableau de bord de son application L'administrateur peut accéder à un tableau de bord centralisé affichant des informations clés, telles que les statistiques sur les candidatures, l'état des offres, les performances du site et les activités récentes, pour superviser efficacement l'ensemble de l'application.

2.1.3 Besoins non fonctionnels

Les besoins non fonctionnels complètent les besoins fonctionnels. C'est un ensemble de propriétés qui permettent d'améliorer la qualité du produit. Parmi ces critères, nous nous concentrons dans notre projet sur les besoins suivants :

- **Sécurité** : Confidentialité des données :Garantir la protection des informations personnelles de l'utilisateur et des données sensibles.
- **L'ergonomie** : L'application propose une interface conviviale, caractérisée par une disposition claire et des couleurs sobres, ce qui facilite la navigation et rend l'expérience utilisateur agréable et intuitive.
- **La disponibilité** : Cette plateforme est disponible de manière fiable, minimisant les temps d'indisponibilité pour les utilisateurs.
- **Performance** : la plateforme fonctionne de manière efficace et réactive, offrant des temps de réponse rapides et une expérience utilisateur fluide.

2.2 Modélisation des besoins fonctionnels

Dans cette section, nous allons exposer les divers diagrammes de cas d'utilisation qui nous permettant de structurer les besoins des utilisateurs et les objectifs du système visant à atteindre.

2.2.1 Vue globale de système

La figure 2.1 présente le diagramme de cas d'utilisation général de l'application, mettant en avant les principales fonctionnalités du système ainsi que ses interactions avec les différents acteurs.

FIGURE 2.1 – Diagramme de cas d'utilisation général

2.2.2 Raffinement de cas d'utilisation

Afin de mieux mettre en évidence les besoins de notre système, nous allons élaborer une décomposition détaillée du diagramme de cas d'utilisation général en créant des sous Diagramme de cas d'utilisation pour chaque composant du système sur lequel nous avons travaillé.

- **Raffinement du cas d'utilisation "Consulter les offres disponibles" :**

La figure 2.2 illustre le raffinement du cas d'utilisation "Consulter les offres disponibles".

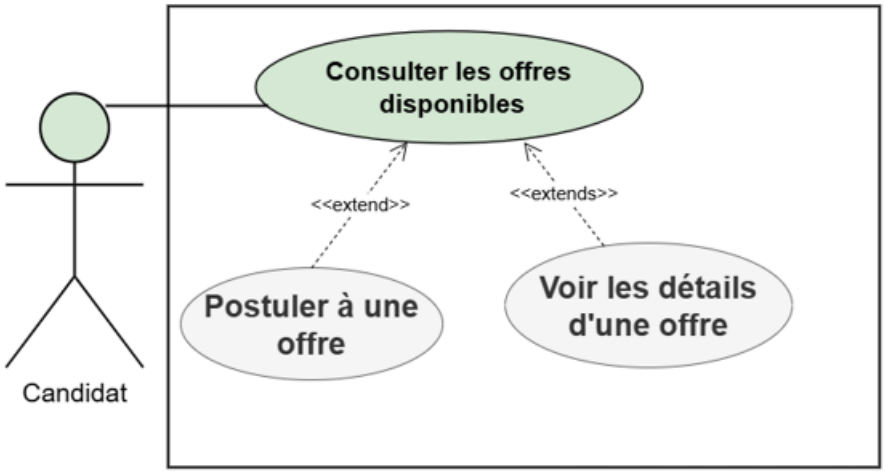

FIGURE 2.2 – Diagramme de cas d'utilisation raffiné «Consulter les offres disponibles»

Le tableau 2.1 présente la description textuelle du cas d'utilisation "Postuler à une offre ".

TABLE 2.1 – Description du cas d'utilisation : Consulter les offres disponibles - Postuler à une offre

Cas d'utilisation	Postuler à une offre
Acteur principal	Candidat
Objectif	Permettre à un candidat de postuler à une offre disponible dans le système.
Pré-conditions	- Le candidat doit avoir un profil complet (CV, informations personnelles, etc.). - Une offre doit être disponible et active.
Post-conditions	- La candidature est soumise et enregistrée dans le système. - Le candidat reçoit une confirmation de soumission. - L'administrateur reçoit une notification en temps réel concernant la nouvelle candidature.
Scénario principal	1. Le candidat consulte la liste des offres disponibles. 2. Il sélectionne une offre qui l'intéresse. 3. Le système affiche les détails de l'offre. 4. Le candidat clique sur le bouton pour postuler. 5. Il saisit les informations nécessaires pour postuler (CV, informations personnelles). 6. Le système enregistre la candidature et envoie une confirmation. 7. L'administrateur reçoit une notification en temps réel indiquant une nouvelle candidature.
Scénarios alternatifs	- Si le profil est incomplet, le système affiche un message d'erreur et invite le candidat à le compléter.

- **Raffinement du cas d'utilisation "Consulter leur postulation" :**

La figure 2.3 illustre le raffinement du cas d'utilisation "Consulter leur postulation". Le tableau 2.2 présente la description textuelle du cas d'utilisation ”Modifer un cv ”.

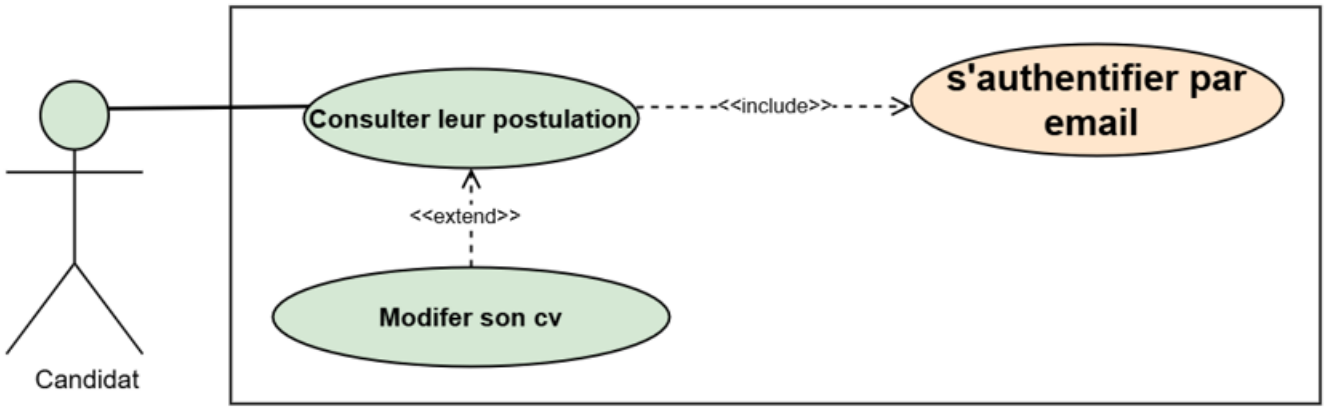

FIGURE 2.3 – Diagramme de cas d'utilisation raffiné «Consulter leur postulation»

TABLE 2.2 – Description du cas d'utilisation : Consulter leur postulation-Modifier son cv

Cas d'utilisation	Modifier un CV
Acteur principal	Candidat
Objectif	Permettre à un candidat de mettre à jour son CV déjà soumis dans une candidature.
Pré-conditions	- Le candidat doit être connecté. - Une candidature existante doit être présente dans le système.
Post-conditions	- Le nouveau CV est enregistré dans le système. - Le candidat reçoit une confirmation de la mise à jour. - L'administrateur peut consulter le CV mis à jour.
Scénario principal	1. Le candidat accède à la section "Mes candidatures". 2. Il sélectionne la candidature à modifier. 3. Le système affiche les détails, y compris le bouton "Modifier le CV". 4. Le candidat clique sur le bouton et sélectionne un nouveau fichier CV. 5. Le système remplace l'ancien CV par le nouveau. 6. Une confirmation de mise à jour est affichée.
Scénarios alternatifs	- Si aucun fichier n'est sélectionné, le système affiche un message d'erreur. - Si le format du fichier est invalide, le système refuse la mise à jour.

- **Raffinement du cas d'utilisation "Gérer les offres " :**

La figure 2.4 illustre le raffinement du cas d'utilisation "Gérer les offres".

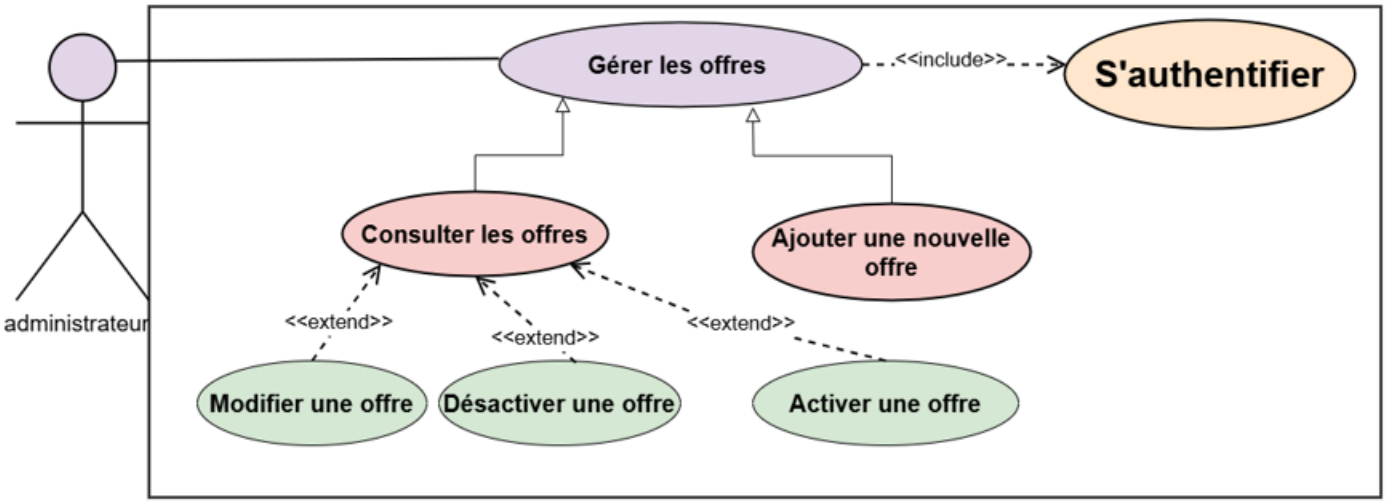

FIGURE 2.4 – Diagramme de cas d'utilisation raffiné «Gérer les offres »

Le tableau 2.3 présente la description textuelle du cas d'utilisation "Gérer les offres-Ajouter une nouvelle offre".

TABLE 2.3 – Description du cas d'utilisation : Ajouter une nouvelle offre

Cas d'utilisation	Ajouter une offre
Acteur principal	Administrateur
Objectif	Permettre à l'administrateur de publier une nouvelle offre d'emploi ou de stage sur la plateforme.
Pré-conditions	- L'administrateur doit être connecté au système.
Post-conditions	- L'offre est enregistrée dans la base de données. - L'offre est visible par les candidats sur la plateforme.
Scénario principal	1. L'administrateur accède à la section de gestion des offres. 2. Il clique sur le bouton " Nouvelle Offre". 3. Le système affiche un formulaire de création d'offre. 4. L'administrateur saisit les informations de l'offre (titre, description,date de clôture, etc.). 5. Il valide le formulaire. 6. Le système enregistre l'offre .
Scénarios alternatifs	- Si des champs obligatoires sont manquants, le système affiche un message d'erreur.

Le tableau 2.4 présente la description textuelle du cas d'utilisation "Gérer les offres-Activer une offre.

TABLE 2.4 – Description du cas d'utilisation : Activer une offre

Cas d'utilisation	Activer une offre
Acteur principal	Administrateur
Objectif	Permettre à l'administrateur de rendre une offre visible et disponible pour les candidats.
Pré-conditions	- L'administrateur doit être connecté. - L'offre à activer doit exister dans le système.
Post-conditions	- L'offre est marquée comme "active". - L'offre devient visible sur la plateforme pour les candidats.
Scénario principal	1. L'administrateur accède à la liste des offres. 2. Il sélectionne une offre actuellement désactivée. 3. Il clique sur le bouton "Activer". 4. Le système met à jour l'état de l'offre à "active". 5. la nouvel liste apres update affiche

- **Raffinement du cas d'utilisation "Gérer les candidats" :**

La figure 2.5 illustre le raffinement du cas d'utilisation "Gérer les candidats".

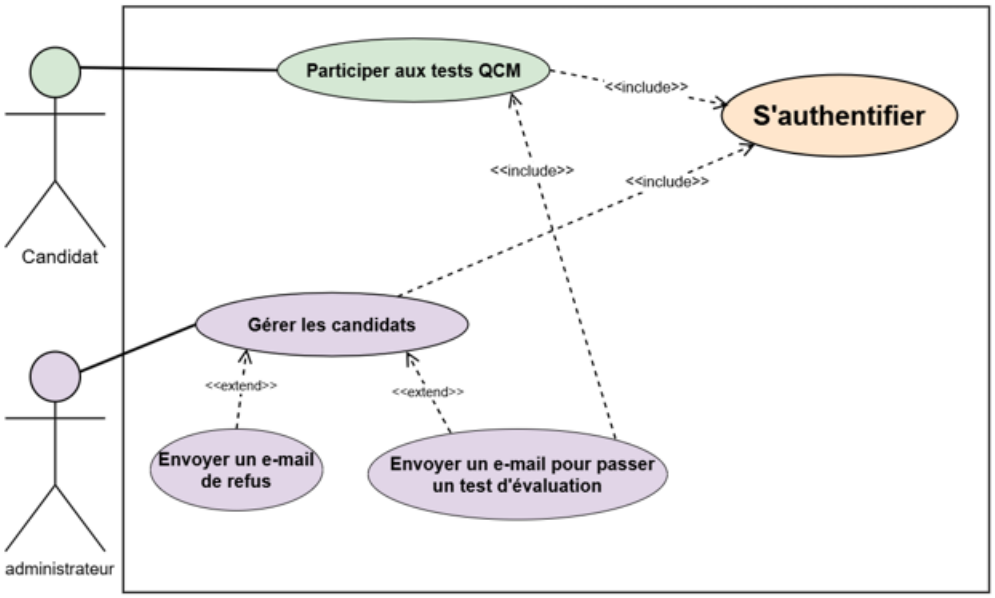

FIGURE 2.5 – Diagramme de cas d'utilisation raffiné «Gérer les candidats »

Le tableau 2.5 présente la description textuelle du cas d'utilisation "Gérer les candidats- Envoyer un e-mail pour passer un test d'évaluation".

TABLE 2.5 – Description du cas d'utilisation Gestion des candidats - Envoi d'un e-mail pour un test d'évaluation

Cas d'utilisation	Envoyer un e-mail pour passer un test d'évaluation
Acteur principal	Administrateur
Objectif	Permettre à l'administrateur d'envoyer un e-mail à un candidat pour l'inviter à passer un test d'évaluation.
Pré-conditions	- L'administrateur doit être connecté au système. - Une offre doit exister et avoir des candidats associés. - Les informations du candidat (nom, e-mail) doivent être disponibles.
Post-conditions	- L'e-mail est envoyé au candidat avec les détails du test d'évaluation. - L'état du candidat est mis à jour dans le système «testPassed ».
Scénario principal	1. L'administrateur accède à la section de gestion des candidats. 2. Le système affiche la liste de candidats 3. Il sélectionne un candidat dans la liste des candidats pour une offre. 4. Il clique sur l'option « Envoyer un test ». 5. Le système envoie l'e-mail au candidat et met à jour l'état du candidat.
Scénarios alternatifs	- Si l'e-mail du candidat est invalide, le système affiche un message d'erreur.

Le tableau 2.6 présente la description textuelle du cas d'utilisation "Gérer les candidats- Passer un test d'évaluation ".

TABLE 2.6 – Description du cas d'utilisation : Passer un test d'évaluation

Cas d'utilisation	Passer un test d'évaluation
Acteur principal	Candidat
Objectif	Permettre au candidat de passer un test d'évaluation associé à une offre spécifique.
Préconditions	- Le candidat doit avoir reçu un courriel contenant le lien vers le test. - Le candidat doit être authentifié sur la plateforme. - Le test ne doit pas avoir déjà été passé par le candidat
Postconditions	- Le statut du candidat est mis à jour (« Test complété »). - Le score du test est calculé et sauvegardé.
Scénario principal	1. Le candidat clique sur le lien du test reçu par courriel. 2. Le système affiche la page de connexion. 3. Le candidat s'authentifie à l'aide des identifiants fournis. 4. Le système redirige vers une page présentant les conditions du test. 5. Le candidat clique sur le bouton « Oui, je suis prêt ». 6. Le système affiche le test sous forme de questionnaire à choix multiples (QCM). 7. Le candidat répond aux questions. 8. Le candidat soumet ses réponses. 9. Le système calcule le score, l'enregistre et met à jour le statut du candidat.
Scénarios alternatifs	- Si le candidat a déjà passé le test, un message l'en informe et l'accès au test est bloqué. - Si le temps imparti est écoulé, le système affiche un message d'expiration et rend le test inaccessible.

Conclusion

Ce chapitre a été consacré à l'analyse et la spécification des besoins du projet. En premier lieu, nous avons identifié les différents acteurs du système. Ensuite, nous avons analysé les besoins fonctionnels et non fonctionnels de chaque partie sur laquelle nous avons travaillé. Finalement, nous sommes arrivés à tracer et à raffiner les diagrammes des cas d'utilisation correspondants au système global et les parties sur lesquelles nous avons travaillé.

Chapitre 3

Étude Conceptuelle

Introduction

Après avoir décortiqué les besoins des utilisateurs, nous abordons dans ce chapitre l'étude conceptuelle. En effet dans une première partie, nous allons présenter la conception générale qui exposera l'architecture logique ainsi que physique de notre plateforme. Ensuite nous allons détailles la vue statique de notre projet en se basant sur le diagramme de classes. Enfin, la vue dynamique sera présentée et ceci en élaborant les diagrammes de séquences.

3.1 Conception générale

En reconnaissant son importance fondamentale, cette section se concentre sur la conception architecturale de notre solution. Nous débutons en clarifiant les deux aspects essentiels de cette conception : l'architecture logique et l'architecture physique.

3.1.1 Architecture logique

L'architecture logique définit l'organisation interne de l'application en termes de couches et de responsabilités. Elle vise à séparer les différentes préoccupations fonctionnelles (interface utilisateur, logique métier, accès aux données) afin de faciliter la maintenance, l'évolution du code et la réutilisation des composants. Dans notre projet, nous avons adopté une approche modulaire combinant le modèle MVC pour le back-end avec Node.js, et une architecture adaptée à React avec Redux pour le front-end.

3.1.1.1 Architecture back-end

Le modèle de conception **MVC (Model - View - Controller)**, illustré dans la figure 3.1, est une architecture logicielle qui divise une application en trois composants principaux : **le Modèle**, **la Vue**, et

le Contrôleur.

- **Modèle (Model)** : Le modèle représente les *données* de l'application ainsi que la *logique métier* qui les régit. Il est responsable de la gestion des règles métier, de la structure des données, et de leur persistance.
- **Vue (View)** : La vue représente l'*interface utilisateur*. Elle affiche les données obtenues du modèle ou transmises par le contrôleur, et se met à jour en cas de changement du modèle, souvent via un mécanisme d'observation.
- **Contrôleur (Controller)** : Le contrôleur traite les *interactions de l'utilisateur*, met à jour le modèle en conséquence, puis sélectionne la vue appropriée à afficher. Il agit comme intermédiaire entre la vue et le modèle.

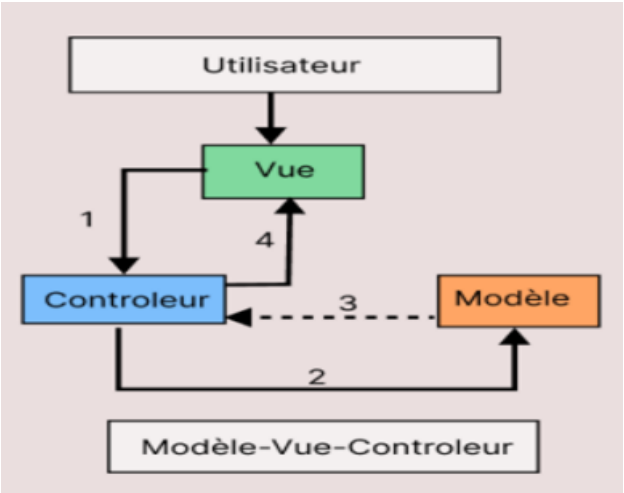

FIGURE 3.1 – Architecture MVC : Modèle - Vue - Contrôleur

L'architecture **MVC** est largement adoptée pour sa capacité à découpler les différentes préoccupations d'une application. Parmi ses principaux avantages, on peut citer :

- **Séparation des responsabilités** : Chaque composant (modèle, vue, contrôleur) est responsable d'une tâche spécifique, ce qui permet une gestion plus claire et plus modulable du code.
- **Facilité de maintenance et évolutivité** : Les modifications apportées à un composant, qu'il s'agisse de la logique métier ou de l'interface utilisateur, n'affectent pas les autres, ce qui facilite l'évolution de l'application sans risque de régression.
- **Facilité de tests unitaires** : La séparation nette permet de tester chaque composant indépendamment, ce qui améliore la couverture des tests et leur efficacité.

3.1.1.2 Architecture front-end

L'architecture **React Redux** est un modèle de conception populaire pour gérer l'état des applications React de manière prévisible et efficace. Elle repose sur un flux de données unidirectionnel et sur plusieurs

composants clés qui interagissent pour gérer l'état global de l'application. Voici une explication détaillée de cette architecture :

- **Store** : C'est l'objet central qui détient l'état global de l'application. Il agit comme un conteneur unique pour toutes les données de l'application, permettant un accès centralisé.
- **Actions** : Ce sont des objets JavaScript simples qui décrivent les changements à apporter à l'état. Elles sont la seule source d'information pour le store, envoyant des instructions sur les modifications à effectuer.
- **Reducers** : Ce sont des fonctions pures qui spécifient comment l'état de l'application change en réponse aux actions. Elles prennent l'état actuel et une action, et retournent un nouvel état.
- **Dispatch** : C'est la méthode utilisée pour envoyer des actions au store, déclenchant ainsi les changements d'état.
- **Selectors** : Ce sont des fonctions qui extraient des parties spécifiques de l'état du store, permettant aux composants d'accéder uniquement aux données dont ils ont besoin.
- **Middleware** : Il s'agit de fonctions optionnelles qui s'exécutent entre l'envoi d'une action et le moment où elle atteint le reducer. Elles sont utiles pour la gestion des effets secondaires, comme les appels API asynchrones.
- **Provider** : C'est un composant React qui enveloppe l'application et rend le store accessible à tous les composants enfants.

Le flux de données dans cette architecture suit un cycle unidirectionnel :

1. L'utilisateur interagit avec l'interface utilisateur.
2. Cette interaction déclenche une action.
3. L'action est envoyée (dispatched) au store.
4. Les reducers traitent l'action et mettent à jour l'état.
5. Le store notifie les composants abonnés du changement d'état.
6. Les composants se re-renderisent avec le nouvel état.

3.2 Architecture physique

L'architecture physique de notre système est détaillée à travers un diagramme de déploiement. Ce diagramme illustre la disposition physique des différents modules constituant le système ainsi que les connexions qui les relient. La figure 3.2 présente les ressources matérielles et logicielles, ainsi que les interactions entre les composants de notre application web *Tradrly*.

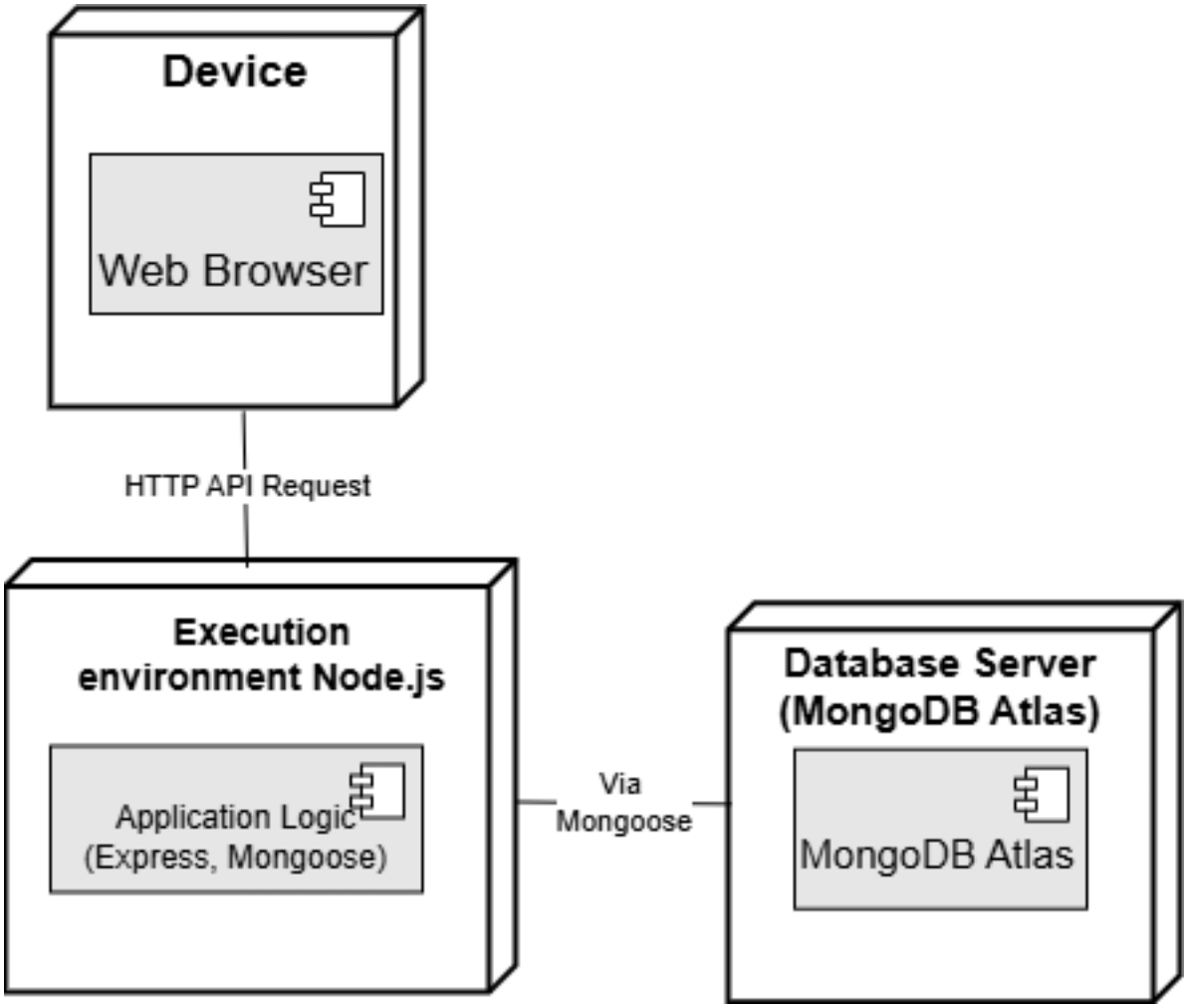

FIGURE 3.2 – Diagramme de déploiement de l'application *Tradrly*

3.3 Conception détaillé

Dans cette section, nous abordons la conception détaillée de notre projet, en mettant en évidence les aspects statiques et dynamiques à travers la modélisation des diagrammes de classes et des diagrammes de séquences.

3.3.1 Vue statique

La figure 3.3 illustre le diagramme de classes composé de différentes interfaces et classes liées entre elles, intervenant dans la récupération des informations de l'utilisateur connecté au système.

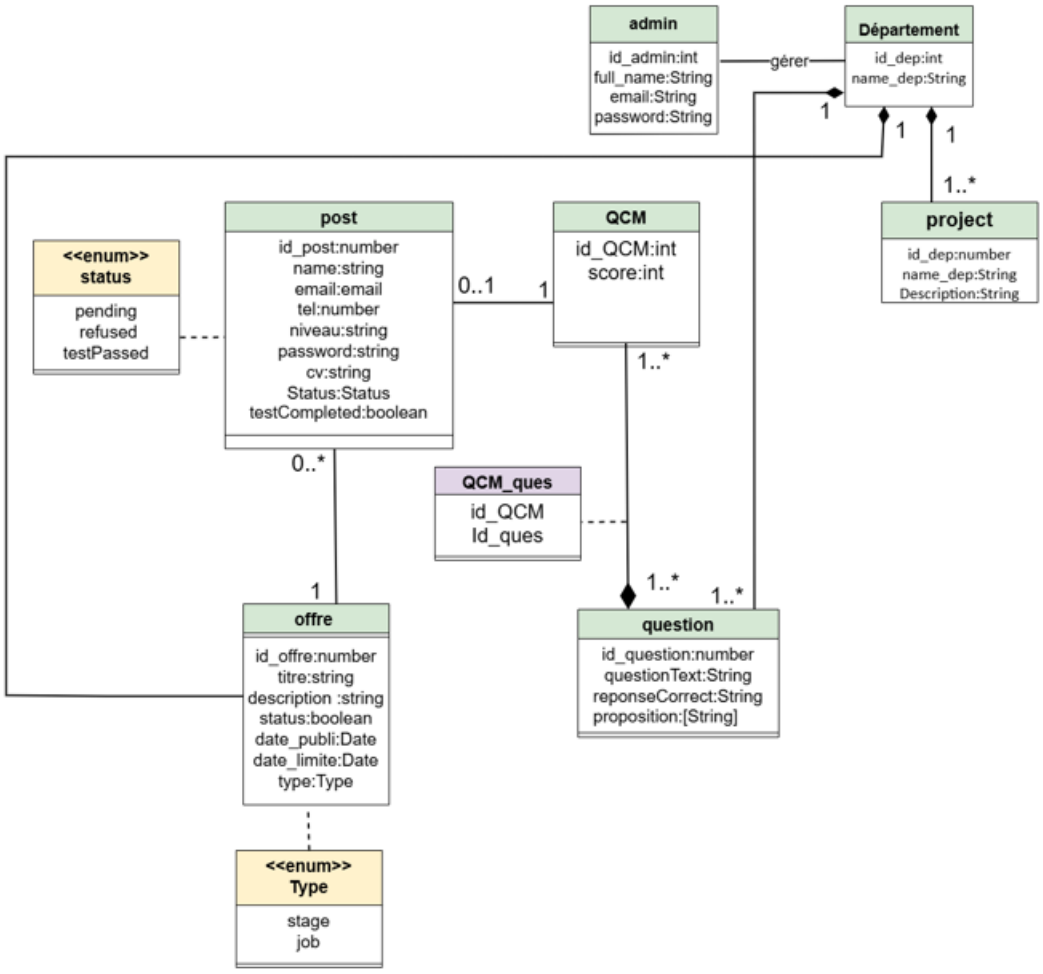

FIGURE 3.3 – Diagramme de classes

Parmi les classes principales du diagramme, nous distinguons :

Admin : Un administrateur est une personne qui gère la plateforme. Il possède un identifiant, un nom complet, une adresse e-mail et un mot de passe pour se connecter à l'interface d'administration.

Département : Un département représente une unité organisationnelle pouvant gérer plusieurs projets. Il est défini par un identifiant unique et un nom.

Project : Un projet est une entité liée à un département. Il comporte un identifiant, un nom et une description pour expliquer ses objectifs ou son contenu.

Offre : Une offre représente une opportunité de stage ou d'emploi. Elle contient un identifiant, un titre, une description, un statut indiquant si l'offre est active, une date de publication, une date limite, et un type (stage ou job).

Post : Un post désigne une candidature soumise par un utilisateur en réponse à une offre. Il comprend les informations personnelles du candidat (nom, email, téléphone, niveau, mot de passe), un lien vers son CV, le statut de la candidature (en attente, refusée ou test réussi), et un indicateur si le test a été complété.

QCM : Un QCM (questionnaire à choix multiples) est un test associé à une candidature. Il contient un identifiant et un score obtenu après évaluation.

Question : Une question est un élément de test utilisé dans les QCM. Elle est définie par un identifiant, le texte de la question, la bonne réponse, ainsi qu'une liste de propositions possibles.

QCM_ques : Il s'agit d'une table d'association entre les QCM et les questions, permettant d'indiquer quelles questions appartiennent à quels tests.

Status (enum) : Représente l'état d'une candidature. Il peut prendre les valeurs suivantes : *pending* (en attente), *refused* (refusée), *testPassed* (test réussi).

Type (enum) : Représente le type d'une offre. Il peut s'agir soit d'un *stage*, soit d'un *job*.

3.4 Vue dynamique

Pour montrer la vue dynamique de notre système, nous optons pour les diagrammes de séquence objet. Dans la suite de cette section, nous avons choisir d'exposer les diagrammes de séquences relatifs à l'authentification, l'inscription, la création de covoiturage et l'ajout de véhicule.

3.4.1 Diagramme de séquence relatif à l'authentification

Pour mettre en évidence l'interaction entre les différents composants de notre système, nous exposons le diagramme de séquence présenté dans la figure 3.4, décrivant le processus d'authentification de l'administrateur

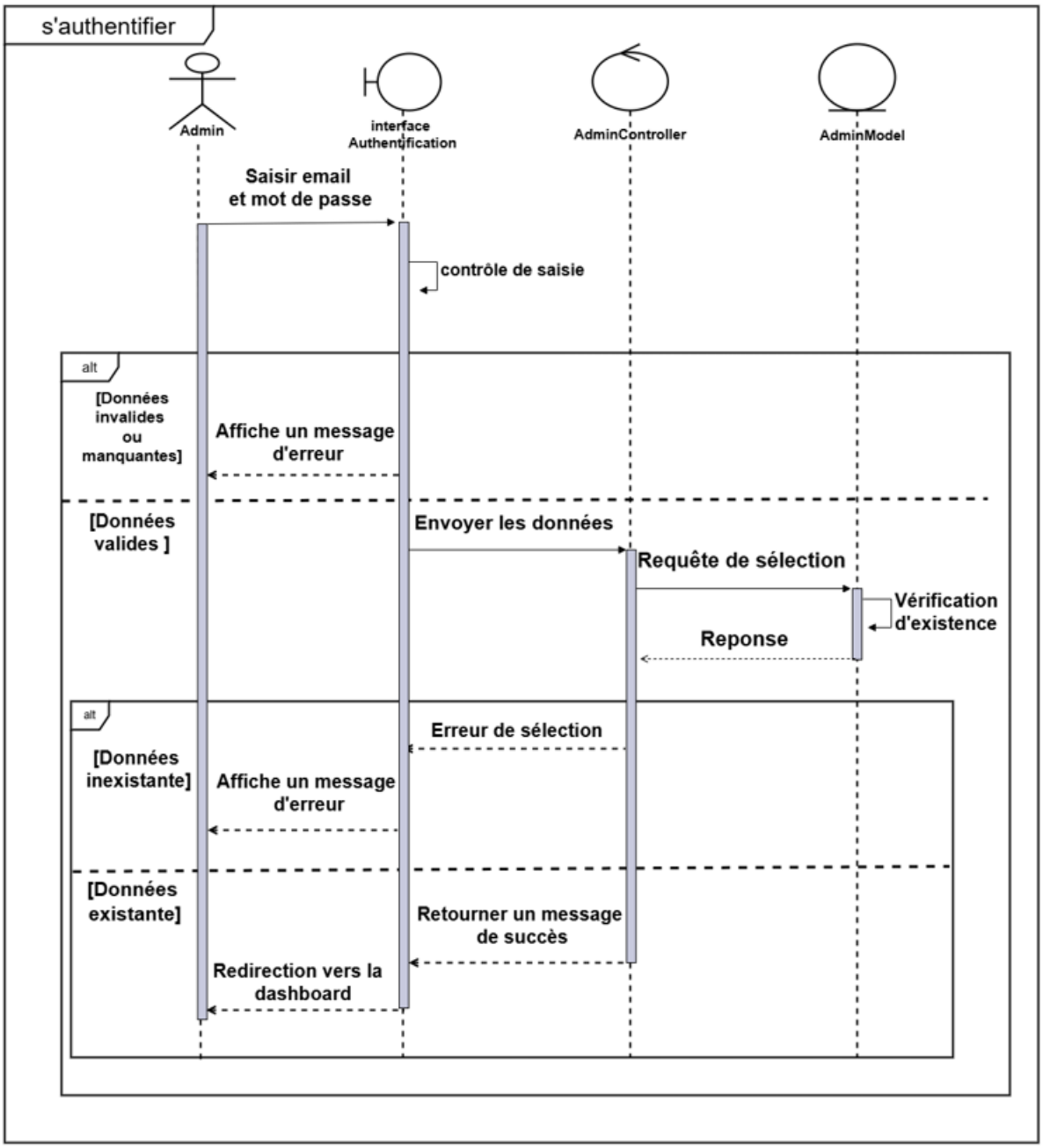

FIGURE 3.4 – Diagramme de séquence relatif à l'authentification

3.4.2 Diagramme de séquence relatif à l'ajout d'une offre

Pour illustrer l'interaction entre les différents composants de notre système, nous présentons le diagramme de séquence de la figure 3.5, décrivant le processus d'ajout d'une offre par un administrateur.

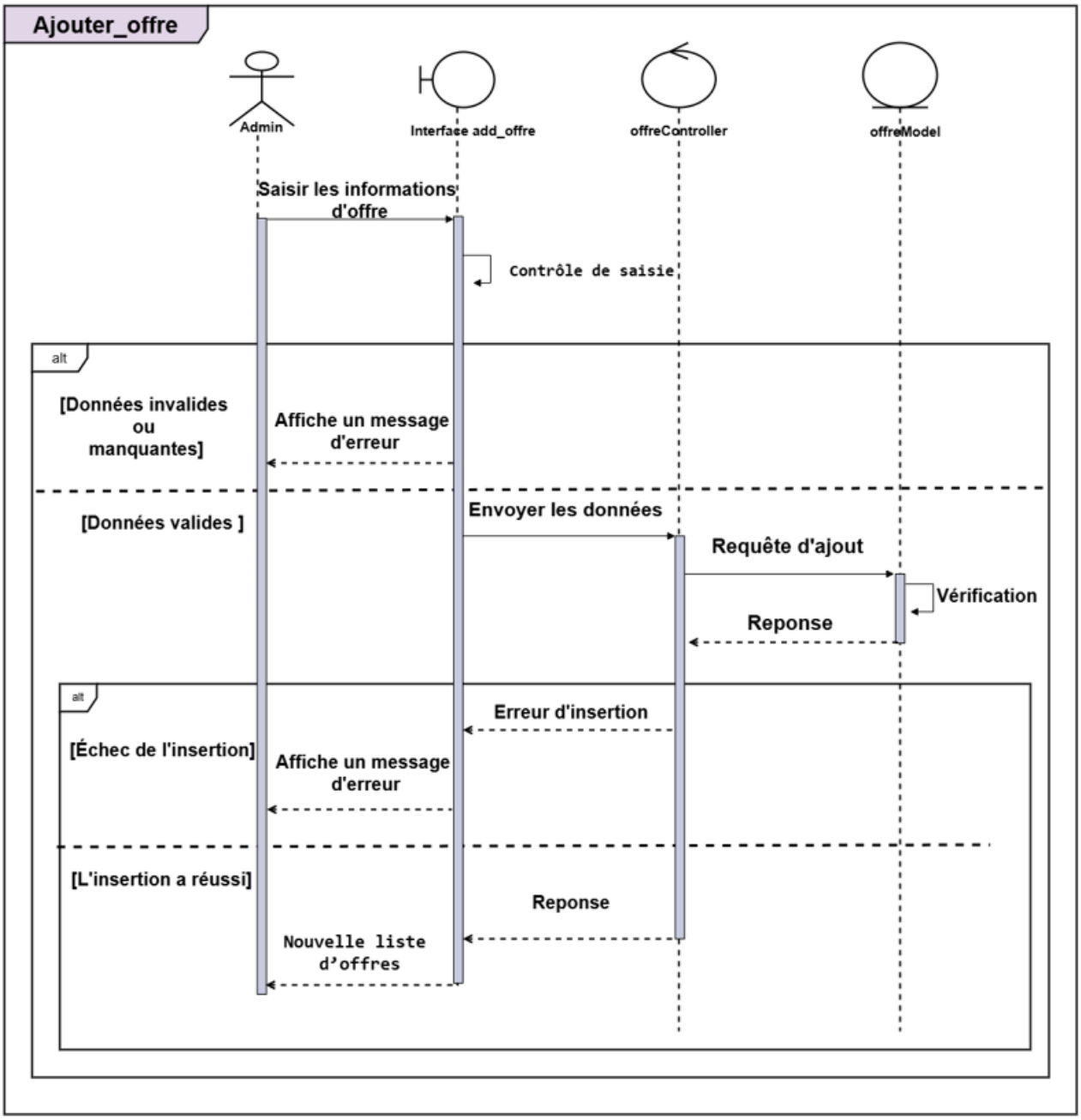

FIGURE 3.5 – Diagramme de séquence relatif à l'ajout d'une offre

3.4.3 Diagramme de séquence relatif à la suppression d'un département

Pour illustrer l'interaction entre les différents composants de notre système, nous présentons le diagramme de séquence de la figure 3.6, décrivant le processus de suppression d'un département par un administrateur.

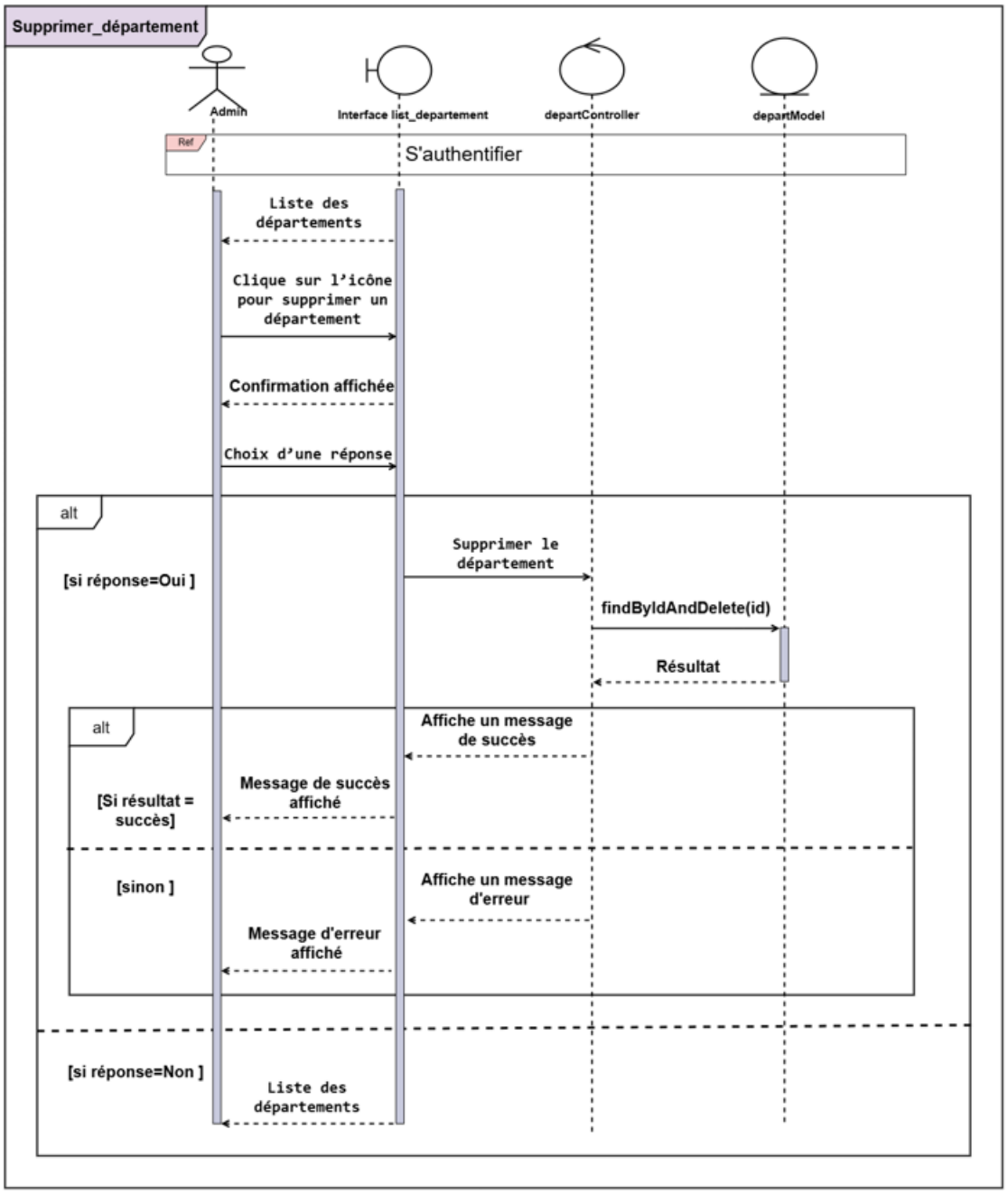

FIGURE 3.6 – Diagramme de séquence relatif à la suppression d'un département

Conclusion

Ce chapitre a été consacré à la présentation de l'étude conceptuelle de la solution à mettre en place. Nous avons détaillé tout d'abord les architectures logique et physique. Ensuite nous avons exposé les vue statiques et dynamique en se basant respectivement sur le diagramme de classes et de séquences.

Chapitre 4

Réalisation

Introduction

Une fois la phase de conception de l'application est terminée, nous pouvons maintenant passer à la réalisation qui fait l'objet de ce chapitre et élaborée minutieusement les diagrammes associés. En effet, nous décrirons l'environnement de travail, en mettant en lumière les technologies et les outils que nous avons sélectionnés. Ensuite, nous présenterons un aperçu des interfaces réalisées tout en expliquant les fonctionnalités développées

4.1 Environnement de développement

Dans cette section, nous dévoilons les fondements de notre solution en exposant les frameworks et les logiciels qui ont été mobilisés pour sa réalisation.

4.1.1 Technologies utilisées

Afin de réaliser notre application, nous avons choisi les technologies suivantes :

1. **Node.jS** Node.js est un environnement de serveur open source multiplateforme qui peut s'exécuter sur Windows, Linux, Unix, macOS, etc. C'est un environnement d'exécution JavaScript back-end, s'exécute sur le moteur JavaScript V8 et se basant sur du code JavaScript en dehors d'un navigateur Web

FIGURE 4.1 – Logo Node.js

2. **Express.js** ExpressJS est un framework qui se veut minimaliste. Très léger, il apporte peu de surcouches pour garder des performances optimales et une exécution rapide

FIGURE 4.2 – Logo Express.js

3. **ReactJS** React (aussi appelé React.js ou ReactJS) est une bibliothèque (library) JavaScript, permettant de développer des applications web qui peuvent être rechargées sans recompiler

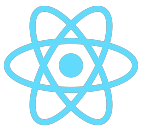

FIGURE 4.3 – Logo ReactJS

4. **MongoDB** MongoDB est une base de données NoSQL orientée documents, qui stocke les données dans des documents BSON (Binary JSON). Elle permet une grande flexibilité et évolutivité, adaptée aux applications nécessitant une gestion efficace de grandes quantités de données variées.

FIGURE 4.4 – Logo MongoDB

5. **Socket.io** est un module de Node.js qui permet de créer des Web Sockets, c'est-dire des connexions bidirectionnelles entre clients et serveur qui permettent une communication en temps réel sur un autre protocole que le protocole http normalement utilisé dans les pages web.

FIGURE 4.5 – Logo Socket.io

6. **SurveyJS** SurveyJS est une bibliothèque JavaScript open source conçue pour la création dynamique de formulaires, de questionnaires et de sondages web. Elle permet de générer des interfaces utilisateur interactives avec divers types de questions (choix multiples, texte, notation, etc.), d'appliquer des logiques conditionnelles, de valider les réponses, et de collecter les résultats sous différents formats.

FIGURE 4.6 – Logo SurveyJS

4.1.2 Logiciels utilisés

1. **Visual Studio Code** Visual Studio Code est un éditeur de code open-source développé par Microsoft supportant un très grand nombre de langages grâce à des extensions. Il supporte l'auto-complétions, la coloration syntaxique, le débogage, et les commandes git

FIGURE 4.7 – Logo Visual Studio Code

2. **Git** Git est un système de contrôle de version distribué largement utilisé dans le développement de logiciels

FIGURE 4.8 – Logo Git

3. **draw.io** C'est un outil qui permet aux utilisateurs de créer différents types de diagrammes tels que des organigrammes, des diagrammes de flux, des diagrammes de réseau, des schémas de base de données, etc. de manière collaborative et intuitive

FIGURE 4.9 – Logo draw.io

4. **Overleaf** Overleaf est une plateforme de rédaction collaborative en ligne qui utilise le système LaTeX pour la création de documents scientifiques, techniques et académiques. Elle permet aux

utilisateurs de travailler sur un même document en temps réel, de manière synchronisée et d'éditer le contenu sans nécessiter d'installation de logiciel sur leur ordinateur.

FIGURE 4.10 – Logo Overleaf

Webographie