



**REPUBLIQUE TUNISIENNE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR
ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE TUNIS EL MANAR**

**FACULTE DES SCIENCES DE TUNIS
DEPARTEMENT DES SCIENCES DE L'INFORMATIQUE**

**RAPPORT DE
PROJET FEDERE LCS2**

EtudJob

Réalisé par :

Abdellatif Ranim

Amri Rihem

Hasnaoui Nour

Satouri Ranim

Encadré par : Faouzi MOUSSA

Organisme d'accueil: FST – DSI

Chapitre 1: État de l'Art et Fondements Métiers

Plan Du Chapitre

Introduction.

1- Contexte Métier du Projet

2- Problématique Métier

3- Proposition de Valeur

4- Étude de Marché

5- Backlog Métier

Conclusion

Introduction

Le présent chapitre offre une plongée approfondie dans le domaine spécifique du projet, en mettant en lumière les éléments clés du contexte métier, les défis rencontrés, ainsi que la proposition de valeur du projet. À travers une exploration méthodique, ce chapitre vise à éclairer les fondements sur lesquels repose le développement de l'application dédiée aux emplois à temps partiel pour les étudiants.

1.1. Contexte Métier du Projet

Le projet Etudjob s'inscrit dans le dynamique et prometteur secteur d'emploi des étudiants à temps partiel visant à répondre de manière innovante aux besoins spécifiques des étudiants en quête d'opportunités de travail à temps partiel adaptées à leurs horaires et engagements académiques. Ainsi, La concrétisation de ce projet découle d'une prise de conscience approfondie des défis auxquels les étudiants sont confrontés et qui vont bien au-delà du simple besoin d'opportunités de travail à temps partiel.

En effet, en observant la réalité des étudiants, nous avons amèrement constaté que certains sont contraints d'abandonner leurs parcours académiques en raison de difficultés financières, accentuées par la pression exercée sur leurs familles pour répondre à des besoins fondamentaux tels que l'achat d'un ordinateur portable ou de matériel de recherche.

En se positionnant comme une plateforme en ligne dédiée, Etudjob aspire à devenir le pilier incontournable pour les étudiants cherchant à générer un revenu d'appoint tout en poursuivant leurs études.

1.2. Problématique Métier

1) Défis :

- **Manque de Plateformes Centralisées:** Les étudiants peinent à trouver des emplois temporaires adaptés à leurs horaires académiques en raison de l'absence d'une plateforme centralisée qui regroupe ces offres d'emploi, ce qui rend la recherche fastidieuse et inefficace.

- **Inadéquation des Offres d'Emploi avec les Horaires Académiques:** Beaucoup d'emplois disponibles ne prennent pas en compte les contraintes horaires des étudiants, rendant difficile la conciliation entre études et travail.
- **Pressions Financières Accrues sur les Étudiants:** L'inflation et la hausse des coûts de vie exacerbent la pression financière sur les étudiants, les poussant parfois à abandonner leurs études faute de moyens.
- **Inflation constante :** L'inflation économique aggrave la difficulté pour les étudiants de subvenir à leurs besoins financiers, augmentant ainsi la nécessité d'emplois temporaires.
- **Risques d'Arnaques et d'Exploitation:** Le recours à des offres d'emploi éparpillées sur les réseaux sociaux expose les étudiants à des risques d'arnaques et d'exploitation, en raison du manque de vérification et d'organisation des offres.

2) Objectifs :

- **Créer une plateforme conviviale :** Développer une plateforme conviviale et accessible pour les étudiants afin de faciliter leur accès à des emplois temporaires adaptés à leurs horaires académiques.
- **Vérification et Sécurisation des Offres d'Emploi:** Assurer une vérification rigoureuse des offres publiées sur la plateforme pour protéger les étudiants contre les arnaques et les exploitations.
- **Adaptation des Emplois aux Horaires Académiques:** Offrir une gamme d'emplois flexibles, adaptés aux horaires académiques des étudiants, permettant une meilleure gestion du temps entre études et travail.
- **Support Financier aux Étudiants:** À travers l'accès facilité à des emplois temporaires, contribuer à alléger la pression financière sur les étudiants, leur permettant de poursuivre leurs études sans le fardeau d'un abandon dû à des contraintes financières.
- **Faciliter la connexion employeur-étudiant :** Offrir aux employeurs une plateforme efficace pour trouver des candidats adaptés à leurs besoins, favorisant ainsi une correspondance optimale entre l'offre et la demande.

1.3. Proposition de Valeur

EtudJob se distingue par son engagement à fournir une plateforme inclusive, créant une connexion harmonieuse entre les étudiants et une variété d'opportunités de travail à temps partiel qui répondent à leurs besoins spécifiques. La grande valeur pour les étudiants réside dans la flexibilité qu'elle propose, permettant d'ajuster les engagements en fonction des contraintes académiques et des préférences personnelles. Quant aux employeurs, la proposition de valeur se cristallise dans la facilité d'accès à une main-d'œuvre qualifiée et flexible. Cette plateforme novatrice leur permet d'identifier rapidement des étudiants compétents, prêts à prendre en charge une gamme variée de tâches, allant du babysitting aux missions domestiques plus complexes, à l'écriture d'un cv ... Ainsi, EtudJob offre une solution agile, répondant de manière proactive aux besoins spécifiques de chaque partie prenante, créant un écosystème où l'ambition et la flexibilité se rencontrent pour façonner des opportunités et stimuler la réussite.

1.4. Étude de Marché

1.4.1. Analyse de l'environnement concurrentiel et des tendances du marché liées au projet :

L'étude de marché constitue une étape cruciale pour situer le projet dans son environnement concurrentiel et identifier les tendances du marché. Dans le contexte tunisien, l'analyse de l'environnement concurrentiel révèle un paysage marqué par une absence de plateformes spécialisées dans la recherche d'emplois à temps partiel pour les étudiants. En effet, les solutions actuelles se concentrent principalement sur les opportunités de stage ou d'emploi à temps plein, ne répondant pas adéquatement aux besoins spécifiques des étudiants à la recherche de revenus complémentaires tout en poursuivant leurs études. Les offres d'emploi publiées sur les réseaux sociaux, tels que Facebook, sont souvent non fiables et ne proposent pas une expérience de recherche d'emploi structurée et sécurisée. Face à cette lacune sur le marché tunisien, notre application se distingue en proposant une plateforme dédiée exclusivement aux emplois à temps partiel pour les étudiants. Contrairement aux autres solutions, notre application offre une interface conviviale et intuitive, facilitant la recherche et la candidature pour les

étudiants tout en offrant une variété d'opportunités adaptées à leurs horaires et à leurs compétences.

1.4.2. Identification des opportunités et des menaces potentielles :

En intégrant des fonctionnalités innovantes telles que des algorithmes de recommandation personnalisés notre application vise à fournir une expérience optimale et sécurisée pour les deux parties. Notre engagement envers la fiabilité et la pertinence des offres d'emploi garantit aux étudiants un accès à des opportunités vérifiées et appropriées, contribuant ainsi à renforcer leur indépendance financière et leur développement professionnel.

Cependant, des menaces subsistent, notamment la nécessité de communiquer efficacement sur la valeur ajoutée d'EtudJob par rapport aux réseaux sociaux. Il est impératif de sensibiliser les utilisateurs potentiels aux avantages en termes de sécurité, de diversité des opportunités et d'efficacité que propose la plateforme. Une stratégie de communication solide contribuera à éduquer le marché et à positionner EtudJob comme la solution préférée pour les emplois à temps partiel étudiants

1.5. Backlog Métier

ID	Thème	User Story	Priorité	Sprint
1	Enregistrement et Profils	En tant qu'employeur et étudiant, je peux créer et gérer un profil.	1	1
2	Publication d'Offres d'Emploi	En tant qu'employeur et étudiant, je peux publier des offres d'emploi pour des emplois à temps partiel.	1	

		En tant qu'employeur et étudiant, je peux spécifier les détails de l'emploi.	1	
		En tant qu'employeur et étudiant, je peux gérer et mettre à jour mes offres d'emploi.	1	
3	Consulter les offres d'Emplois	En tant qu'étudiant, je peux consulter les offres des emplois à temps partiel en fonction de différents critères (lieu, domaine, etc.)	1	2
		En tant qu'étudiant, je peux ajouter un intérêt.	1	
		En tant qu'employeur, je peux consulter les autres offres des emplois	1	
4	Gestion de Profil Étudiant	En tant qu'employeur et étudiant, je peux mettre à jour mes informations de profil.	1	

5	Consulter application	En tant qu'administrateur , je peux consulter les utilisateurs et les offres.	1	3
6	Tests et Validation	En tant qu'administrateur, je peux Créer des scénarios de test pour chaque fonctionnalité.	1	3
		En tant qu'administrateur, je peux Effectuer des tests unitaires, d'intégration et des tests d'acceptation utilisateur.	1	

Tableau1 : backlog du produit

Chapitre 2 : Fondements Technologiques et Cadre Conceptuel du Projet

Plan Du Chapitre

Introduction

1- Revue de la Littérature

2- Contexte Informatique

3- État de l'Art en Architecture Logicielle

4- Outils et Méthodologies

5- Cadre Conceptuel

Conclusion

Introduction

Le deuxième chapitre présente le contexte informatique en couvrant les technologies, les architectures logicielles et les concepts essentiels, tels que les frameworks, les design patterns, les EDI, les langages de programmation et le backlog. Ces éléments forment la base du cadre conceptuel, soulignant les choix et les approches qui guideront le développement du projet.

2.1. Revue de la Littérature

Revue de la littérature pour EtudJob : Nous avons précisément identifié les progrès majeurs, les méthodes efficaces et les résultats concrets applicables à notre projet. Les avancées notables comprennent l'intégration réussie de technologies de filtrage visant à aligner les offres d'emploi avec les disponibilités des étudiants. Les méthodologies mettent en avant une approche centrée sur l'expérience utilisateur, garantissant ainsi une correspondance efficace entre les compétences des étudiants et les besoins des employeurs. Ainsi, cette revue de la littérature guide directement la conception d'EtudJob en se basant sur des fonctionnalités pratiques pour maximiser l'impact de notre plateforme dans le domaine spécifique des emplois étudiants.

2.2. Contexte Informatique

Les concepts fondamentaux qui sous-tendent le projet etudJob ont pour dessein de familiariser l'équipe de développement avec des principes essentiels tels que l'architecture logicielle, la gestion de bases de données, la sécurité informatique, et les concepts liés au développement web. Mettant l'accent sur ces fondements, l'objectif est de créer une base de connaissances commune, favorisant ainsi une compréhension partagée des exigences techniques du projet.

Parallèlement, la revue des technologies et des langages de programmation pertinents pour le projet révèle des choix spécifiques, avec l'adoption de Spring Boot et React. Spring Boot, en tant que framework Java, offre une architecture robuste pour le développement côté serveur, facilitant la création de services web performants. Du côté client, React est privilégié pour son approche réactive dans la construction d'interfaces utilisateur interactives offrant ainsi une expérience utilisateur (UX) et une interface utilisateur (UI) de haute qualité.

2.3. État de l'Art en Architecture Logicielle

2.3.1. Les Architectures :

Face à une diversité d'architectures dont chacune ayant ses atouts séduisants pour les développeurs, le choix de la plus adaptée pour notre plateforme a devenu une décision cruciale.

En effet nous nous sommes confrontés à la question passionnante de déterminer la voie qui assurera la flexibilité, la robustesse et la convivialité que notre plateforme exige. Le problème était que chacune de ces architectures, promet une expérience de développement unique, mais il est impératif de choisir celle qui sera le pilier solide sur lequel reposera notre vision.

Chaque choix architectural offre un éclat différent.

2.3.1.1. Architecture en couches

L'architecture en couches structure les applications en niveaux hiérarchiques pour séparer les fonctions et améliorer la modularité et l'entretien. Chaque couche a des responsabilités définies, facilitant la réduction de duplication de code et l'adaptabilité. Les couches supérieures s'appuient sur les inférieures pour des services, encapsulant les implémentations pour simplifier la maintenance et augmenter la réutilisabilité.

Dans une structure en couches typique, quatre couches primaires sont observées :

1-La couche de présentation

2-La couche application

3-La couche d'accès aux données

4-La couche de données

Avantages :

1- Modularité: La séparation en couches aide les développeurs à se concentrer sur des segments spécifiques, simplifiant et organisant le code pour une meilleure gestion et réutilisation dans divers projets.

2- Évolutivité: Les couches indépendantes permettent une optimisation et une mise à l'échelle séparées, facilitant l'adaptation aux besoins changeants et améliorant les performances, essentiel pour les applications d'envergure.

3- Maintenabilité: L'isolement des fonctions au sein de couches distinctes rend les mises à jour et modifications plus aisées, sans impacts majeurs sur l'architecture globale.

4- Testabilité: La possibilité de tester chaque couche séparément assure la fiabilité et la robustesse de chaque composant et de l'application dans son ensemble, aligné sur des pratiques comme celles d'AppMaster pour éliminer la dette technique.

5- Interopérabilité: La structure facilitant l'intégration avec d'autres systèmes, grâce à la création d'interfaces claires comme les API RESTful et la documentation Swagger, promeut une communication efficace entre différents services.

2.3.1.2. EDA - Architecture pilotée par les événements

L'architecture orientée événements (de l'anglais eventdriven architecture, ou EDA) est une forme d'architecture de médiation qui est un modèle d'interaction applicative mettant en œuvre des services (composants logiciels) répondant à des sollicitations externes avec une forte cohérence interne (par l'utilisation d'un format d'échange pivot, le plus souvent JSON ou XML), et des couplages externes lâches (par l'utilisation d'événements)

Event-Driven Architecture (EDA)

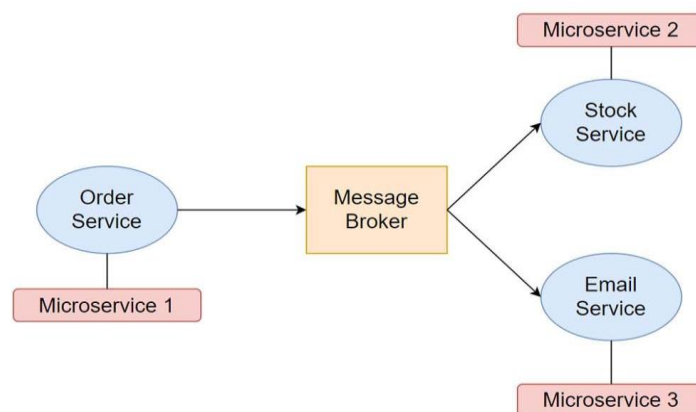


Figure 1 : Architecture EDA

Avantages :

1. **Conception** : Réactivité et adaptabilité en temps réel. Ajout et modification de composants sans perturber le système.
2. **Évolutivité** : Mise à l'échelle indépendante des composants pour une utilisation efficace des ressources et une réactivité optimale.

3. **Modularité** : Conception modulaire favorisée par le couplage lâche des composants et la communication par événements. Amélioration de la maintenabilité et de la réutilisabilité du code.
4. **Résilience** : Les défaillances d'un composant n'affectent pas nécessairement le système entier. Réduction du risque de pannes en cascade.
5. **Réactivité en temps réel** : Excellente pour les situations nécessitant des actions immédiates basées sur des événements. Réponse rapide aux changements de conditions ou aux interactions des utilisateurs.

Inconvénients :

1. **Complexité** : Conception et implémentation complexes dues à la gestion de la propagation des événements, des sources d'événements et des styles de traitement.
2. **Débogage** : Débogage et résolution des problèmes difficiles en raison de la nature décentralisée des interactions distribuées.
3. **Ordonnancement des événements** : Défi de garantir l'ordre correct des événements, en particulier dans les systèmes distribués. Risque de comportement inattendu en cas de séquençement incorrect.
4. **Cohérence des données** : Difficulté de maintenir la cohérence des données entre plusieurs composants. Les événements peuvent déclencher des changements dans différentes parties du système.
5. **Courbe d'apprentissage** : Adaptation nécessaire pour les développeurs habitués aux architectures traditionnelles. Compréhension du paradigme piloté par les événements et de ses concepts.

2.3.1.3. L'Architecture Orientée Services (SOA)

L'architecture orientée services (SOA) est un modèle de conception qui vise à développer des applications modulaires composées de services indépendants, chacun remplissant une tâche spécifique et communiquant entre eux.

La SOA définit un moyen de rendre les composants logiciels **réutilisables et interopérables** grâce à des interfaces de services. Ces services utilisent des normes d'interface communes et un modèle architectural, ce qui permet de les incorporer rapidement dans de nouvelles applications. Voici quelques points clés :

- **Services indépendants** : Chaque service d'une architecture SOA comprend le code et les données nécessaires à l'exécution d'une fonction métier distincte (par exemple, vérification de la solvabilité d'un client, calcul d'une mensualité de prêt, traitement d'une demande de prêt immobilier).
- **Couplage lâche** : Les interfaces de service fournissent un couplage lâche, ce qui signifie qu'elles peuvent être appelées sans savoir comment le service est mis en œuvre en dessous, réduisant ainsi les dépendances entre les applications.
- **Protocoles standard** : Les services sont exposés via des protocoles réseau standard tels que SOAP/HTTP ou Restful HTTP (JSON/HTTP) pour envoyer des demandes de lecture ou de modification de données.
- **Gouvernance des services** : Contrôle du cycle de vie du développement et publication des services dans un registre pour une réutilisation efficace.

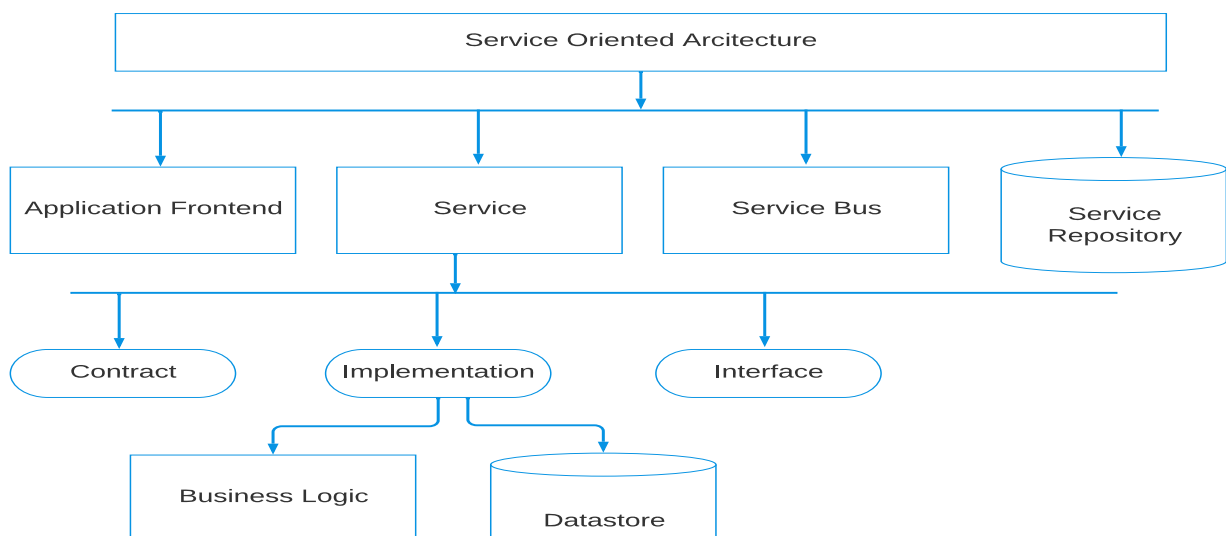


Figure 2 : architecture SOA

Avantages

1. **Agilité et flexibilité** : La SOA permet de décomposer les systèmes complexes en services modulaires plus petits, permettant ainsi de répondre rapidement aux évolutions des besoins.
2. **Réutilisabilité** : Les services peuvent être réutilisés dans différentes applications, évitant ainsi la duplication de fonctionnalités.

3. **Mise sur le marché plus rapide** : La réutilisabilité facilite le développement et la mise en production plus rapides.

Inconvénients

1. **Complexité initiale** : La mise en place d'une SOA peut être complexe, nécessitant une planification minutieuse.
2. **Coût** : La SOA peut être coûteuse en termes de ressources humaines, de développement et de technologie.
3. **Maintenance** : La gestion de multiples services nécessite une surveillance continue.

2.3.1.4. Architecture Microservices

Les microservices sont un système de développement logiciel où une application est construite sous forme d'un ensemble de petits services indépendants. Contrairement à l'architecture monolithique, où l'application est développée comme une seule unité, les microservices fonctionnent de manière autonome. Chaque microservice peut même être écrit dans un langage de programmation différent. Ils communiquent entre eux via des requêtes HTTP à leurs APIs.

Avantages

1. **Évolutivité** : Les microservices peuvent être déployés et mis à l'échelle indépendamment les uns des autres. Cela permet d'ajuster la capacité en fonction des besoins spécifiques de chaque service.
2. **Flexibilité** : Chaque service est développé indépendamment, ce qui facilite les mises à jour et les modifications. Vous pouvez apporter des changements à un microservice sans affecter l'ensemble de l'application.
3. **Fonctionnalité modulaire** : Les modules sont indépendants, ce qui simplifie la maintenance et la gestion des fonctionnalités.

Inconvénients

1. **Complexité accrue** : La gestion de multiples services nécessite une coordination minutieuse. La communication entre les microservices doit être soigneusement gérée.
2. **Sécurité des réseaux** : Assurer la sécurité des communications entre les microservices peut être un défi.

2.3.1.5. Architecture MVC :

L'architecture MVC est l'une des architectures logicielles les plus utilisées pour les applications Web, elle se compose de 3 modules :

- **Modèle** : noyau de l'application qui gère les données, permet de récupérer les informations dans la base de données, de les organiser pour qu'elles puissent ensuite être traitées par le contrôleur.
- **Vue** : composant graphique de l'interface qui permet de présenter les données du modèle à l'utilisateur.
- **Contrôleur** : composant responsable des prises de décision, gère la logique du code qui prend des décisions, il est l'intermédiaire entre le modèle et la vue.

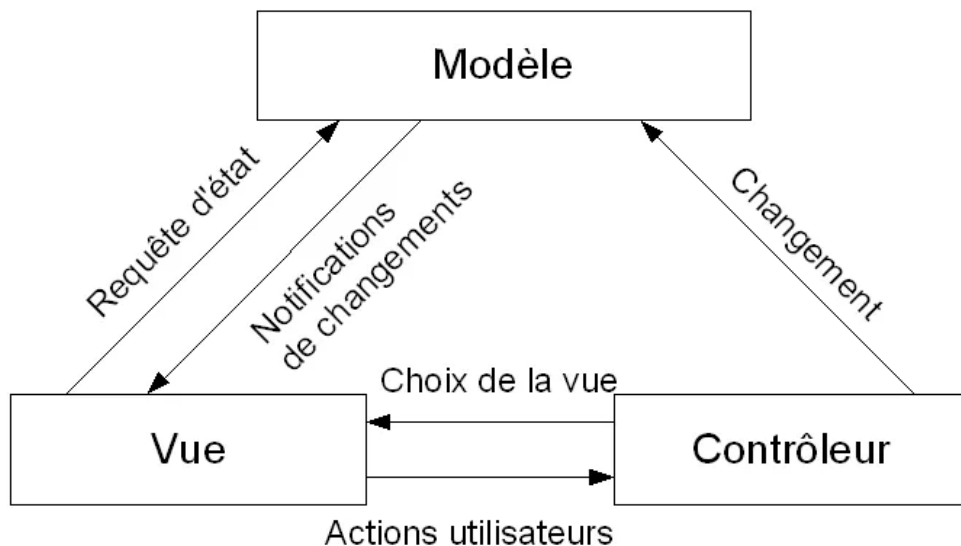


Figure 3: Architecture MVC

Fonctionnement

Le contrôleur va demander au modèle les données, les analyser, prendre des décisions et renvoyer le texte à afficher à la vue.

avantages

- Meilleure organisation du code;
- Diminution de la complexité lors de la conception;
- Conception claire et efficace grâce à la séparation des données de la vue et du contrôleur;

- Possibilité de réutilisation de code dans d'autres applications;
- Un gain de temps de maintenance et d'évolution du site;
- Une plus grande souplesse pour organiser le développement du site entre différents développeurs;
- Plus de facilité pour les tests unitaires.

Inconvénients

- Augmentation de la complexité lors de l'implantation
- Éventuel cloisonnement des développeurs
- Architecture complexe pour des petits projets
- Le nombre important de fichiers représente une charge non négligeable dans un projet.
- **Monitoring et gouvernance** : Suivre et gérer l'ensemble des microservices peut être complexes .

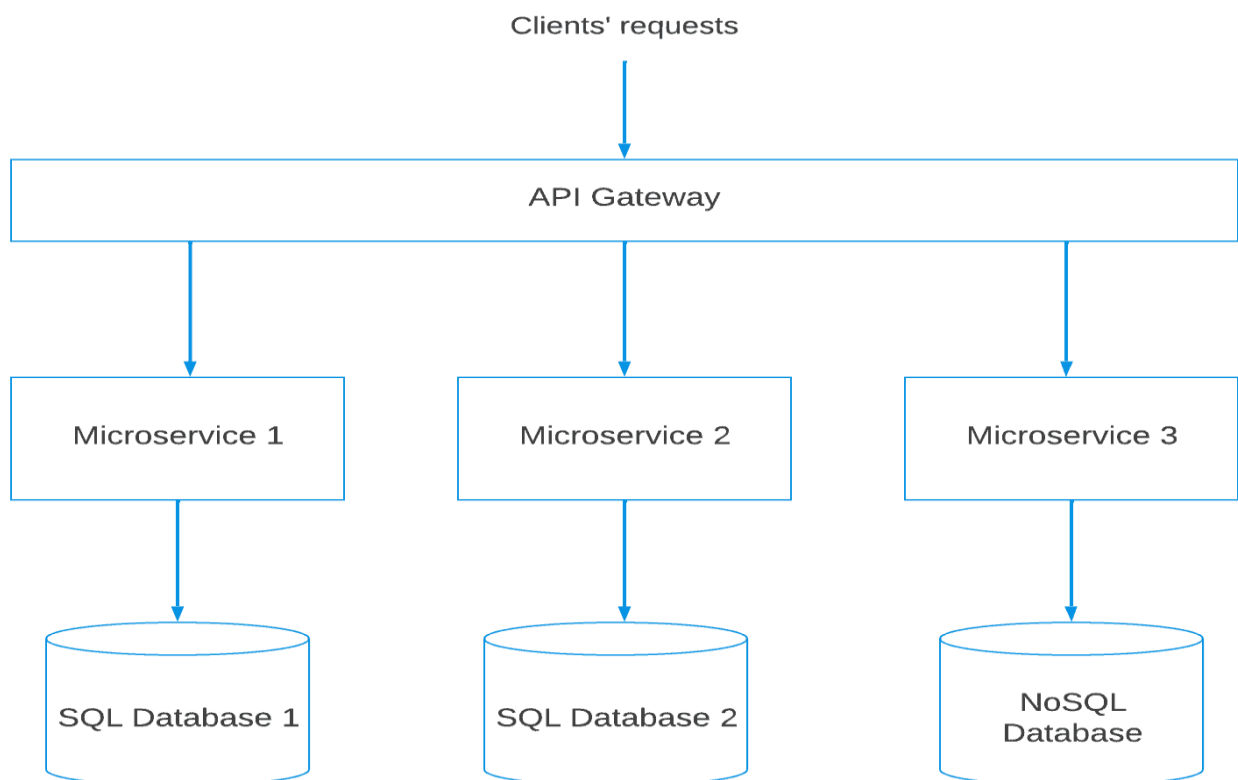


Figure 4 : Architecture Microservices

2.4. Outils et Méthodologie

2.4.1. Framework

SpringBoot :

Java Spring Framework (Spring Framework) est une infrastructure open source d'entreprise couramment utilisée qui permet de créer des applications autonomes de production qui fonctionnent sur la machine virtuelle Java (JVM).

Java Spring Boot (Spring Boot) est un outil qui accélère et simplifie le développement d'applications Web et de microservices avec Spring Framework grâce à trois fonctionnalités principales :

- *Configuration automatique*
- *Approche directive de la configuration*
- *Possibilité de créer des applications autonomes*

Ces fonctionnalités fonctionnent ensemble pour fournir un outil qui permet de configurer une application Spring avec une configuration et une installation minimale.

Pourquoi SpringBoot ?

Spring Boot est choisi comme framework principal pour le développement backend de l'application en raison de sa robustesse, de sa flexibilité et de sa facilité de configuration. En utilisant Spring Boot, nous bénéficions d'une architecture solide basée sur des conventions bien établies, ce qui accélère le processus de développement et permet une gestion efficace des dépendances. De plus, la vaste communauté de développeurs et la documentation exhaustive de Spring Boot offrent un support précieux pour résoudre rapidement les problèmes et tirer pleinement parti des fonctionnalités avancées du framework.

Reactjs

React (également connu sous les noms React.js ou ReactJS) est une bibliothèque JavaScript open-source et gratuite pour la construction d'interfaces utilisateur basées sur des composants, destinée au développement frontal. Elle est maintenue par Meta (anciennement Facebook) ainsi que par une communauté de développeurs individuels et d'entreprises.

React peut être utilisé pour développer des applications monopages, mobiles, ou rendues côté serveur avec des cadres de travail tels que Next.js. Étant donné que React se concentre uniquement sur l'interface utilisateur et le rendu des composants dans le DOM, les applications React s'appuient souvent sur des bibliothèques pour le routage et autres fonctionnalités côté client. Un avantage clé de React est qu'il ne re-rend que les parties de la page qui ont changé, évitant ainsi le re-rendu inutile d'éléments du DOM inchangés.

Pourquoi ReactJS ?

React.js est sélectionné comme bibliothèque frontale pour l'interface utilisateur de l'application en raison de sa modularité, de sa réactivité et de sa facilité d'utilisation. Avec React.js, nous pouvons créer des composants réutilisables et structurer efficacement l'interface utilisateur de manière déclarative. De plus, la virtual DOM de React.js permet des mises à jour efficaces et optimisées de l'interface utilisateur, améliorant ainsi les performances globales de l'application. La richesse de la communauté et l'abondance des ressources disponibles font de React.js un choix solide pour le développement d'applications web interactives et dynamiques.

2.4.2. Design patterns

Design Pattern Observer (Observateur) :

Description : Le pattern Observer est idéal pour gérer les notifications et les mises à jour en temps réel sur la plateforme.

application :

Personnalisation des Notifications : Les utilisateurs peuvent définir leurs préférences et critères pour recevoir des notifications ciblées sur les opportunités qui les intéressent.

Gestion des Mises à Jour en Temps Réel : Lorsqu'une nouvelle opportunité est ajoutée ou modifiée sur la plateforme, les utilisateurs inscrits recevront automatiquement des notifications pour rester informés.

Design Pattern Strategy :

Description :

Approche de conception logicielle qui permet de définir une famille d'algorithmes, de les encapsuler individuellement et de les rendre interchangeables.

Cette flexibilité permet aux clients de déléguer le choix de l'algorithme approprié à l'exécution, offrant ainsi une modularité et une extensibilité accrues au système.

Application :

En utilisant ce design pattern, on va gérer efficacement les différents types d'opportunités de travail en fonction des besoins spécifiques des étudiants.

Par exemple, avoir des stratégies distinctes pour les offres de travail basées sur la localisation, le domaine d'étude, l'historique.

Les utilisateurs pourraient alors sélectionner dynamiquement la stratégie correspondant le mieux à leurs critères actuels.

Renforçant la Stratégie dans notre architecture logicielle, on va offrir une flexibilité essentielle dans la sélection et la gestion des offres d'emploi à temps partiel sur la plateforme, renforçant ainsi son attrait et son utilité pour les étudiants

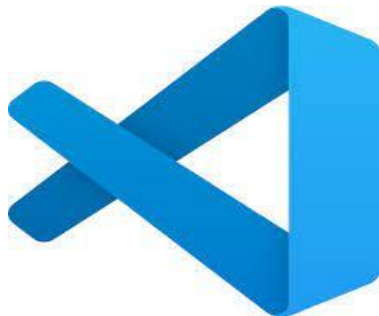
Design Pattern Singleton (Singleton) :

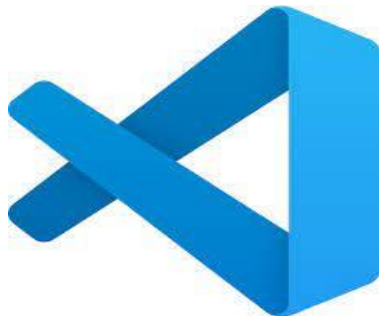
Description : c'est une approche de conception qui garantit qu'une classe n'a qu'une seule instance et fournit un point d'accès global à cette instance.


Cela permet de contrôler strictement l'accès à une ressource partagée et d'assurer qu'une seule instance de cette classe est créée.

Application : Pour votre plateforme, le pattern Singleton est utilisé pour des composants critiques et uniques tels que la gestion des utilisateurs, la connexion à la base de données. Ainsi, on va assurer une cohérence et une gestion efficace des ressources essentielles pour le bon fonctionnement de la plateforme

2.4.3. Environnements de Développement Intégrés

<p>Visual Studio Code (VSCode) est un éditeur de code polyvalent et intuitif, apprécié pour sa légèreté, sa rapidité et ses nombreuses fonctionnalités qui améliorent l'efficacité des développeurs.</p>	
---	--



<p>STS (Spring Tool Suite) est un environnement de développement intégré (EDI) basé sur Eclipse, spécialement conçu pour le développement d'applications Spring, offrant une gamme complète d'outils pour simplifier et accélérer le processus de développement logiciel.</p>	
--	--

2.4.4. Langages de programmation :

Pour notre projet on a utilisé Java pour le back-end et JavaScript pour le front-end

Java (Back-end) :

Description : Java est un langage de programmation polyvalent et robuste largement utilisé pour le développement back-end.

Il offre une performance élevée, une grande portabilité et une vaste communauté de développeurs.

Java est particulièrement adapté pour la gestion des données, la logique métier, les opérations complexes et la communication avec les bases de données.

Avantages : Typage statique fort, gestion automatique de la mémoire, multithreading efficace, large écosystème de bibliothèques et frameworks (Spring, Hibernate), sécurité renforcée.

Application : Java est idéal pour la création d'API , la manipulation de données, l'implémentation de logique métier complexe et la gestion des interactions avec les bases de données dans votre plateforme.

JavaScript (Front-end) :

Description : JavaScript est le langage de programmation incontournable pour le développement front-end web.

Il permet d'ajouter des fonctionnalités interactives aux sites web, d'animer les éléments de l'interface utilisateur et d'interagir avec les utilisateurs en temps réel. IL est essentiel pour la création d'applications web réactives et dynamiques.

Avantages : Facilité d'intégration avec HTML et CSS, exécution côté client, large compatibilité avec les navigateurs, riche ensemble de bibliothèques (React, Angular, Vue.js).

Application : JavaScript sera utilisé pour concevoir l'interface utilisateur interactive de votre plateforme, gérer les interactions utilisateur en temps réel, effectuer des requêtes asynchrones vers le back-end et rendre l'expérience utilisateur fluide et attrayante.

2.4.5 Méthodologies :

Méthode Agile :

La méthode agile est une approche du développement logiciel dont l'objectif est de distribuer en continu des logiciels opérationnels créés sur la base d'itérations rapides.

Quand vous travaillez en mode Agile, vous travaillez en de petits cycles courts que l'on appelle sprints ou itérations qui durent généralement entre 1 semaine et 1 mois.

les phases d'un projet agile :

- | | |
|---|--|
| <i>1-Le cadrage du projet.</i> | <i>5-La reprise du processus.</i> |
| <i>2-La préparation du backlog.</i> | <i>6-L'ouverture au changement. ...</i> |
| <i>3-L'exécution des tâches.</i> | <i>7-Une accélération du Time-to-Market...</i> |
| <i>4-La prise en compte des retours client.</i> | <i>8-Une transparence totale.</i> |

les avantages de la méthode agile :

- 1-L'importance du retour d'information*
- 2-Une très forte flexibilité*
- 3-L'implication de l'équipe*

Quelles sont les méthodes agiles :

1-La méthode agile scrum :

Cette première méthode est aussi la plus populaire. Son principe de base est le sprint. Ce terme désigne des cycles de projets très courts. Leur durée peut varier de quelques heures à quelques semaines. On décompose ainsi le projet initial en différents sprints. Cette vision itérative se déroule de la manière suivante :

Cadrage initial du projet et définition des objectifs ;

Liste de l'ensemble des demandes du client ;
Réalisation des tâches du sprint ;
Prise en compte des retours du client ;
Répétition du processus.

2-La méthode agile kanban :

Kanban, signifiant « étiquette » en japonais, utilise un système visuel de cartes pour représenter les tâches, organisées sur un tableau en catégories "À faire", "En cours", et "Réalisé", favorisant la visualisation du travail et l'autonomie des équipes.

3-La méthode agile extreprogramming (XP) :

L'ExtremeProgramming (XP) est spécifiquement conçu pour le développement logiciel, avec des cycles de travail très courts pour l'ajout et la validation rapide de nouvelles fonctionnalités. Il se distingue par un travail en binôme pour la révision continue du code, visant à produire un code de haute qualité et facile à maintenir.

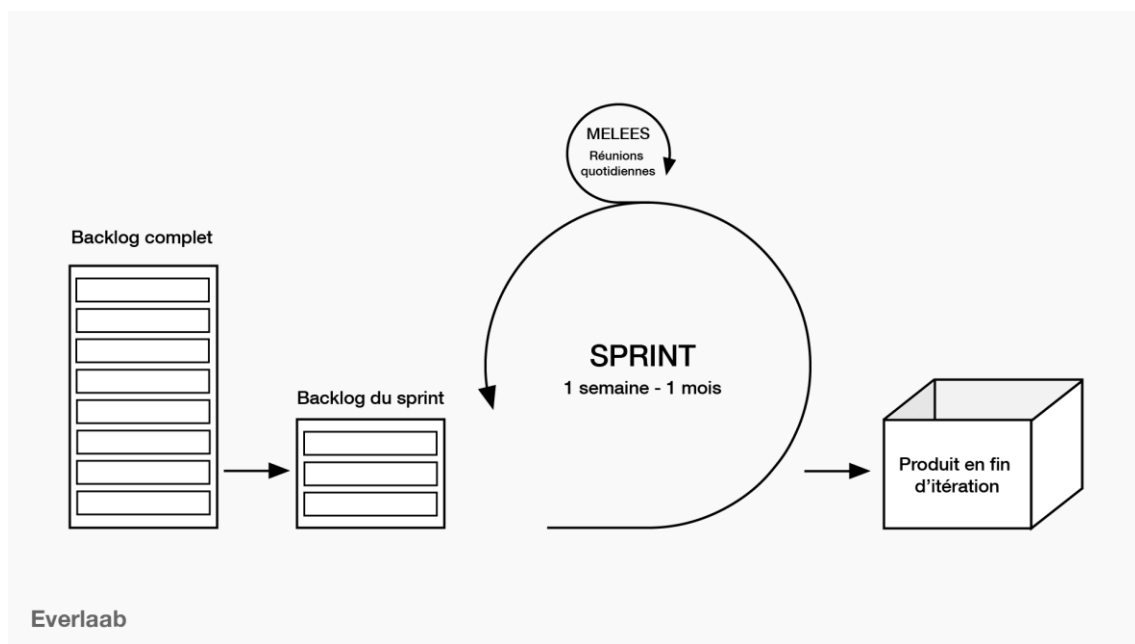


Figure 5 : La méthode agile extrême programming (XP)

Backlog :

Le backlog, en tant que liste ordonnée de tâches et de fonctionnalités à réaliser, nous permet de visualiser clairement les éléments à traiter, de définir les priorités en fonction des besoins des utilisateurs et du business, et de suivre l'avancement du projet de manière structurée.

En intégrant le backlog dans le processus de gestion de projet, on garantit une meilleure coordination entre les membres de l'équipe, une communication fluide et une livraison efficace des fonctionnalités attendues par les utilisateurs finaux.

2.5. Cadre Conceptuel

2.5.1. Architecture Logicielle

Frontend (React)

Utilisation d'une architecture basée sur composants pour une gestion modulaire.

Intégration de la gestion d'état avec Redux pour une cohérence des données.

Mise en œuvre d'un routage efficace pour une navigation fluide.

Backend (Spring Boot)

Utilisation de l'architecture MVC pour séparer les préoccupations.

Mise en place de services RESTful pour permettre la communication avec le frontend.

Intégration de Spring Security pour assurer la sécurité des données étudiantes.

2.5.2. Frameworks

React:

Exploitation des fonctionnalités de React pour créer une interface utilisateur réactive.

Utilisation de Redux pour la gestion de l'état global.

Spring Boot:

Exploitation des fonctionnalités de Spring Boot pour simplifier le développement.

Utilisation de Spring Data JPA pour faciliter la manipulation de la base de données.

2.5.3. Design Patterns

Frontend:

Utilisation du modèle de conception Composite pour la construction de composants.

Mise en œuvre du modèle Observer pour gérer les mises à jour d'état.

Backend:

Utilisation du modèle MVC pour séparer les responsabilités.

Application des modèles Factory et Singleton selon les besoins.

2.5.4. EDI

-vsCode

-Spring tools suite

2.5.5. Langages de Programmation

Frontend:

JavaScript pour la logique côté client.

JSX pour la création de composants React.

Backend:

Java pour le développement côté serveur avec Spring Boot.

2.5.6. Backlog

1. Enregistrement et Profils
2. Publication d'Offres d'Emploi
3. consulter les offres d'Emplois
4. Gestion de Profil Étudiant
5. consulter application
6. Tests et Validation

Biographie

Méthode Agile : <https://www.slack.com>

- Fig 1 : <https://www.slack.com>

Architecture MVC

:<https://medium.com/@belcaid.mehdi/larchitecture-logicielle-mvc-1a8bbb5cf6dc>

- Fig2:

<https://medium.com/@belcaid.mehdi/larchitecture-logicielle-mvc-1a8bbb5cf6dc>

Architecture en couches :<https://appmaster.io/>

Architecture Microservices :<https://www.redhat.com/fr/topics/microservices>

- Fig : <https://www.baeldung.com/cs/microservices-soa-differences>

Architecture SOA :<https://www.redhat.com/fr/topics/cloud-native-apps/what-is-service-oriented-architecture>

- Fig : <https://www.baeldung.com/cs/microservices-soa-differences>

Spring Boot : <https://www.ibm.com/fr-fr/topics/java-spring-boot>

React Js : [https://en.wikipedia.org/wiki/React_\(software\)](https://en.wikipedia.org/wiki/React_(software))

ArchitectureEDA :

https://fr.wikipedia.org/wiki/Architecture_orient%C3%A9e_%C3%A9v%C3%A9nements

https://fr.wikipedia.org/wiki/Architecture_orient%C3%A9e_%C3%A9v%C3%A9nements

[https://www.baeldung.com/cs/eda-software-](https://www.baeldung.com/cs/eda-software-design?fbclid=IwAR1q841s83a_MFbGdejwWTVd1MqAQV54YFdFrxcLciWsnEvnbbhQJ0zVUQ)

[design?fbclid=IwAR1q841s83a_MFbGdejwWTVd1MqAQV54YFdFrxcLciWsnEvnbbhQJ0zVUQ](https://www.baeldung.com/cs/eda-software-design?fbclid=IwAR1q841s83a_MFbGdejwWTVd1MqAQV54YFdFrxcLciWsnEvnbbhQJ0zVUQ)