Université de Bourgogne, UFR Sciences et Techniques, Département I.E.M.

RAPPORT PROJET TUTORÉ

parcoursup
Entrez dans l'enseignement supérieur

Nom et Prénom
Ouédraogo Constant
Aydin Kieran
Ber Lucas
Echaroux Inès
Diallo Aissatou
Maouassi Nouhad
Kut Kemal
Mandione Dia Seydina
Keita Sidy
Nabé Mamoudou

Responsables : Annabelle Gillet, Marinette Savonnet, Nadine Cullot, Claire Bourgeois-République.

2024-2025

Table des matières

1. Introduction . . . 3
2. Objectifs du Projet . . . 3
3. Organisation du Groupe et Répartition des Rôles . . . 3
4. Description des Fonctionnalités . . . 5
   - 4. 1 Étude des Données Open Data . . . 5
   - 4. 2 Conception du Schéma de Stockage des Données . . . 5
   - 4. 3 Étude de l'Algorithme d'Appariement . . . 5
   - 4. 4 Restitution des Données sur un Site Web . . . 5
   - 4. 5 Présentation des Données . . . 5
5. Dépendances entre les Fonctionnalités . . . 5
6. Choix Techniques . . . 6
   - 6. 1 Technologies pour la Gestion des Données . . . 6
   - 6. 2 Langages et Outils de Développement . . . 7
   - 6. 3 Environnement de Travail . . . 9
7. Répartition des Tâches . . . 11
   - 7. 1 Tâches du Premier Semestre . . . 11
8. Conclusion . . . 12

1. Introduction

Depuis son lancement, beaucoup d'étudiants rapportent que la plateforme Parcoursup n'est pas en mesure de répondre à leurs attentes, en effet, nous constatons beaucoup d'insatisfactions dûs à l'incompréhension des méthodes de sélection de l'algorithme. Le projet consiste à réaliser une **analyse complète de la plateforme Parcoursup**, utilisée depuis 2018 par les lycéens pour formuler leurs vœux d'orientation dans l'enseignement supérieur. Cette étude porte sur la période de 2020 à 2023 et a pour objectif d'améliorer la **compréhension des mécanismes de sélection** qui déterminent l'acceptation ou le refus des candidatures.

Le projet s'articule en deux phases : une première partie dédiée à la conception du socle technique et à l'analyse de l'algorithme d'appariement utilisé par Parcoursup, et une seconde partie orientée sur l'analyse des données et la proposition d'améliorations de l'algorithme. Ce travail permettra de mieux comprendre les processus de décision qui influencent le parcours des étudiants et d'identifier d'éventuelles pistes d'optimisation du système.

2. Objectifs du Projet

Le projet tutoré est structuré en deux semestres :

1. Premier semestre :
   L'objectif principal est la conception de la fondation technique de l'application. Cela implique la création d'un schéma de stockage pour les données, l'étude des données disponibles en open data, et une première analyse de l'algorithme d'appariement utilisé par Parcoursup, ainsi que la restitution des données collectées sous forme interactive.

2. Deuxième semestre :
   L'objectif sera d'approfondir l'analyse des données afin d'établir des comparaisons entre les années et les formations. Nous proposerons également des pistes d'amélioration de l'algorithme d'appariement, notamment en étudiant les biais potentiels.

3. Organisation du Groupe et Répartition des Rôles

Nom	Prénom	Alternant	Rôles
NABE	Mamoudou	Non	Responsable du groupe
DIALLO	Aissatou	Non	Responsable adjoint
OUEDRAOGO	Constant	Non	Référent développeur
DIA	Seydina Mandione	Non	Devops
MAOUASSI	Nouhad	Non	Référent qualité
AYDIN	Kieran	Non	Référent données
BER	Lucas	Oui	Développeur
KUT	Kemal	Oui	Développeur
KEITA	Sidy Mahmoud	Oui	Développeur
ECHAROUX	Inès	Oui	Développeur

Pour ce projet, notre groupe s'est réparti les rôles de manière à assurer une cohérence dans l'organisation des tâches et des responsabilités techniques :

— **Responsable de groupe : NABE Mamoudou**
Le responsable de groupe est chargé de coordonner les efforts de l'équipe et de faire le lien avec les encadrants. Il s'assure également du respect des délais et du suivi du projet.

— **DevOps : DIA Seydina Mandione**
Le DevOps est en charge de l'installation et de la configuration des environnements de travail, y compris les machines virtuelles (VM). Il veille à la cohérence des outils et à la gestion des versions.

— **Référent Données : AYDIN Kieran**
Le référent données garantit la qualité et la cohérence des données collectées et utilisées tout au long du projet. Il est aussi responsable de l'analyse des données en open data.

— **Référent Développeur : OUEDRAOGO Constant**
Le référent développeur s'assure de la cohérence des développements réalisés par l'équipe. Il valide l'architecture technique et supervise l'intégration des différentes fonctionnalités.

— **Référent Qualité : MAOUASSI Nouhad**
Le référent qualité est responsable de la mise en place des tests et de la validation des fonctionnalités selon les critères définis en début de projet.

— **Développeurs : BER Lucas, KUT Kemal, KEITA Sidy Mahmoud, ECHAROUX Inès**
Les développeurs participent activement à la réalisation des différentes fonctionnalités définies dans le projet. Ils collaborent sous la supervision du référent développeur pour assurer l'implémentation des fonctionnalités, tout en respectant les bonnes pratiques de développement.

4. Description des Fonctionnalités

4. 1 Étude des Données Open Data

Dans un premier temps, nous procéderons à l'identification et à l'analyse des données publiques disponibles concernant Parcoursup pour les années 2020 à 2023. Ces données incluent les affectations des étudiants, les formations proposées, et les statistiques d'admission. Elles seront utilisées pour construire le modèle de données de notre application et pour effectuer des comparaisons et des analyses au second semestre.

4. 2 Conception du Schéma de Stockage des Données

L'une des tâches principales sera la conception d'un schéma de base de données qui permettra de stocker les informations collectées. Ce schéma devra prendre en compte les relations entre les étudiants, leurs vœux d'admission, les formations, ainsi que les décisions d'affectation. L'objectif est de structurer les données de manière à faciliter leur analyse future.

4. 3 Étude de l'Algorithme d'Appariement

Nous analyserons le fonctionnement de l'algorithme d'appariement de Parcoursup, en nous concentrant sur son évolution au cours des années 2020 à 2023. Nous évaluerons la manière dont il sélectionne les étudiants, les critères utilisés, et les éventuels biais introduits dans le processus de sélection. Cette analyse sera la base de notre étude critique et des propositions d'améliorations à formuler lors du second semestre.

4. 4 Restitution des Données sur un Site Web

Une restitution des données sera effectuée via un site web, afin de présenter les résultats des analyses de manière claire et accessible. Ce site permettra de visualiser les données clés sous forme de tableaux et de graphiques interactifs. Il sera construit en utilisant un framework léger tel que Flask pour le backend et des bibliothèques de visualisation comme Chart.js ou D3.js pour la représentation des données.

4. 5 Présentation des Données

Enfin, nous concevrons une interface simple permettant de visualiser les données collectées. Cette interface sera pensée pour être facilement utilisable, avec des graphiques et des tableaux permettant de comprendre rapidement les principales tendances des données.

5. Dépendances entre les Fonctionnalités

Les différentes fonctionnalités sont interdépendantes :

— La restitution des données sur le site web et la présentation des données reposent sur la conception du schéma de stockage, qui est elle-même dépendante de l'étude des données open data.

— L'analyse de l'algorithme d'appariement nécessite les données stockées pour réaliser des simulations et tester des scénarios.

6. Choix Techniques

6. 1 Technologies pour la Gestion des Données

Nous avons choisi Neo4j comme système de gestion de base de données, en raison de ses avantages spécifiques dans le contexte de notre projet. Neo4j est une base de données orientée graphes, conçue pour modéliser et gérer des données interconnectées. Voici les raisons principales qui justifient ce choix par rapport à d'autres systèmes comme des bases de données relationnelles :

— **Modélisation naturelle des relations** : Dans le cadre de Parcoursup, nous devons gérer des relations complexes entre plusieurs entités, telles que les étudiants, les vœux d'orientation, les formations et les décisions d'affectation. Neo4j est particulièrement bien adapté pour ce type de données, car il modélise ces relations sous forme de nœuds (représentant les entités) et d'arêtes (représentant les connexions entre elles). Contrairement à une base relationnelle où les relations sont définies à travers des jointures coûteuses, Neo4j peut interroger directement les connexions entre les nœuds, ce qui améliore la performance dans ce type de cas.

— **Efficacité des requêtes complexes** : La plateforme Parcoursup repose sur des relations multiples et complexes : un étudiant peut avoir plusieurs vœux, chacun de ces vœux peut concerner plusieurs formations, et ces formations peuvent être liées à plusieurs décisions d'affectation. Avec Neo4j, les requêtes qui nécessitent de traverser plusieurs niveaux de relations (comme la recherche des affectations pour un étudiant ou l'analyse des connexions entre les différentes formations) sont optimisées. En comparaison, dans une base relationnelle, ces types de requêtes nécessiteraient de nombreuses jointures qui peuvent entraîner une dégradation des performances avec des ensembles de données volumineux.

— **Adaptation aux analyses de graphes** : Neo4j fournit des algorithmes d'analyse de graphes prêts à l'emploi (comme les recherches de chemin, les calculs de centralité ou de similarité) qui sont particulièrement utiles pour notre projet. Par exemple, l'algorithme d'appariement de Parcoursup pourrait bénéficier d'une analyse plus fine des connexions entre étudiants et formations, en mesurant la proximité entre certains vœux et les affectations passées. Cette capacité à effectuer des analyses de graphe permet de mieux comprendre et optimiser l'algorithme d'appariement, ce qui serait plus complexe à réaliser avec une base de données relationnelle.

— **Flexibilité et évolutivité** : Neo4j offre une grande flexibilité dans la manière dont les données sont structurées. Contrairement aux bases relationnelles, où le schéma des données doit être bien défini à l'avance, Neo4j permet d'ajouter de nouvelles relations ou entités sans avoir à restructurer entièrement la base de données. Cette flexibilité est un

atout important dans notre projet, car il est possible que les besoins d'analyse évoluent au fil du temps, en particulier lorsque nous étudierons l'algorithme d'appariement et ses biais au second semestre.

— **Cas d'utilisation pertinent pour Parcoursup** : Parcoursup est une plateforme qui gère des données dynamiques et hautement connectées. Neo4j permet de traiter efficacement ces relations à grande échelle, tout en facilitant la visualisation et l'analyse des interconnexions. Les algorithmes d'appariement et les données de décisions d'affectation sont plus naturellement modélisés avec un graphe, où chaque étudiant, chaque formation, et chaque vœu est un nœud interconnecté. Cela permet de retracer facilement les parcours et de proposer des analyses plus détaillées.

— **Conclusion sur le choix de Neo4j** : En résumé, Neo4j représente une solution parfaitement adaptée aux besoins de notre projet en raison de sa capacité à modéliser des relations complexes, à fournir des performances optimisées pour les requêtes traversant plusieurs niveaux de relations, et à offrir une grande flexibilité dans la gestion des données. En comparaison avec une base de données relationnelle, Neo4j simplifie considérablement l'analyse des données et l'optimisation de l'algorithme d'appariement de Parcoursup.

6. 2 Langages et Outils de Développement

Pour le développement de notre projet, nous avons choisi une pile technologique qui s'intègre bien avec Neo4j, tout en permettant une manipulation efficace des données et une restitution claire et interactive sur un site web. Voici les détails des principaux outils et langages que nous allons utiliser :

1. **Python : Langage principal pour l'analyse des données**
   Nous avons opté pour Python pour plusieurs raisons :
   — **Compatibilité avec Neo4j** : Python offre un support natif pour interagir avec Neo4j via des bibliothèques comme `Py2neo` ou `Neo4j Python Driver`, facilitant l'exécution des requêtes Cypher et la gestion des graphes.
   — **Écosystème de bibliothèques pour l'analyse** : Python dispose d'outils comme `Pandas` et `NumPy` qui sont essentiels pour la manipulation et l'analyse de données. Pandas permet de structurer les données sous forme de tableaux (DataFrames), facilitant ainsi leur transformation et analyse, tandis que NumPy fournit des capacités de calcul rapide et efficace, particulièrement utile pour des opérations mathématiques et statistiques.
   — **Flexibilité et évolutivité** : Python s'intègre facilement avec d'autres technologies et frameworks, offrant ainsi la flexibilité de faire évoluer le projet au fur et à mesure des besoins. De plus, sa syntaxe simple et sa grande communauté le rendent facile à maintenir et à étendre.

2. **Flask : Framework pour le Backend du site web**
   Pour le développement du backend de notre site web, nous avons choisi Flask, un fra-

mework minimaliste. Flask présente plusieurs avantages dans le cadre de ce projet :
- **Légèreté** : Flask est un micro-framework, ce qui signifie qu'il impose peu de contraintes architecturales tout en offrant la possibilité d'ajouter des fonctionnalités à mesure que les besoins évoluent. Cela en fait un bon choix pour une application web simple où la performance et la rapidité de développement sont importantes.
- **Intégration facile avec Neo4j** : Grâce à l'écosystème Python, Flask peut facilement interagir avec la base de données Neo4j. Il nous permet de gérer les requêtes utilisateurs pour l'affichage des données stockées dans Neo4j, tout en garantissant une séparation claire entre le traitement des données et leur restitution.
- **Extensibilité** : Flask peut être enrichi avec de nombreuses extensions pour ajouter des fonctionnalités comme l'authentification, la gestion des sessions, ou encore des API REST, ce qui pourrait s'avérer utile dans les phases ultérieures du projet.

3. **Bibliothèques de visualisation : Chart.js et D3.js**
   Une des fonctionnalités clés de notre site web est la restitution visuelle des données, permettant aux utilisateurs de comprendre et d'analyser facilement les résultats du projet. Nous avons choisi deux bibliothèques pour cette tâche :
   - **Chart.js** : C'est une bibliothèque JavaScript pour la création de graphiques interactifs. Elle est idéale pour représenter des données sous forme de barres, de lignes, de pie charts, etc. Chart.js est facile à intégrer dans un site Flask, permettant de rendre les analyses de données accessibles à travers des représentations visuelles claires.
   - **D3.js** : Pour des visualisations plus complexes et personnalisées, nous utiliserons `D3.js` (Data-Driven Documents). Cette bibliothèque permet de créer des visualisations interactives et dynamiques basées sur les données fournies. D3.js sera utile pour visualiser les relations complexes issues des graphes Neo4j, comme les connexions entre étudiants, formations, et affectations, ce qui est plus difficile à représenter avec des graphiques traditionnels.

   Ces deux bibliothèques permettront de fournir aux utilisateurs du site une interface interactive et intuitive pour naviguer dans les données, facilitant la compréhension des tendances et des relations au sein des données.

4. **Intégration de Neo4j dans l'application**
   La combinaison de Flask et Python permet une intégration fluide avec Neo4j, en utilisant le driver `Neo4j Python Driver` pour exécuter des requêtes Cypher et récupérer les données directement depuis la base de données orientée graphe. Grâce à l'API que nous allons mettre en place via Flask, les données stockées dans Neo4j pourront être manipulées et renvoyées en temps réel au site web, où elles seront visualisées dynamiquement avec Chart.js et D3.js.

Conclusion

En choisissant cette pile technologique, nous garantissons une gestion efficace des données complexes avec Neo4j, tout en offrant une flexibilité et une rapidité de développement grâce à Python et Flask. Les outils de visualisation comme Chart.js et D3.js fourniront aux utilisateurs du site web une compréhension des résultats de nos analyses facile.

6. 3 Environnement de Travail

1. **GitLab : Gestion du Code Source et Collaboration**

Bien que l'utilisation de GitLab soit imposée dans le cadre de ce projet, nous allons exploiter ses nombreuses fonctionnalités pour améliorer la collaboration et la gestion du code de manière optimale :

— **Suivi des versions et des modifications du code** : GitLab repose sur le système de versioning Git, permettant à chaque membre du groupe de travailler sur des branches individuelles. Cela garantit que chaque développeur peut tester et expérimenter des fonctionnalités sans interférer avec les autres. Les `merge requests` facilitent la révision et l'intégration du code, tout en assurant une traçabilité et une transparence totale. Ainsi, l'équipe pourra collaborer efficacement tout en réduisant les risques de conflits de code.

— **Automatisation via CI/CD** : GitLab offre des pipelines d'intégration continue (CI) et de déploiement continu (CD), ce qui nous permet d'automatiser le processus de test et de déploiement. Cela garantit que le code passe par des tests automatisés avant d'être fusionné, limitant ainsi les risques d'introduction de bugs ou de régressions dans l'application.

— **Collaboration facilitée** : Grâce aux `issues` et aux `wikis` intégrés, GitLab centralise la gestion des tâches et la documentation. Chaque membre de l'équipe pourra signaler des problèmes, assigner des tâches, et collaborer de manière structurée autour de chaque fonctionnalité à développer. Cette gestion centralisée est un atout majeur pour assurer que tous les membres sont alignés sur les objectifs et les avancements du projet.

— **Sécurité et gestion des permissions** : GitLab permet une gestion des permissions pour chaque membre du groupe. Cela garantit que les accès sont bien contrôlés, et que seuls les membres autorisés peuvent réaliser des actions spécifiques (comme fusionner du code ou effectuer des déploiements). En combinant cela avec les pipelines CI/CD automatisés, nous nous assurons que les processus critiques, comme la mise en production, sont sécurisés et audités.

2. **Docker : Uniformisation des Environnements et Déploiement**

Docker sera utilisé pour standardiser les environnements de développement, faciliter le déploiement de l'application et assurer une continuité entre les environnements de développement, de test, et de production :

— **Isolation des environnements** : Docker nous permet de créer des conteneurs légers

qui isolent chaque composant de l'application (backend, base de données, etc.) dans un environnement indépendant. Chaque développeur peut ainsi exécuter l'application dans un conteneur identique, garantissant que le comportement de l'application est le même sur toutes les machines, quel que soit le système d'exploitation ou la configuration locale. Cela élimine les classiques problèmes de compatibilité d'environnement.

— **Facilité de déploiement** : Grâce à Docker, nous pourrons facilement déployer l'application dans des environnements de test et de production en utilisant les mêmes conteneurs. Cela signifie que nous pouvons tester localement le même code que celui qui sera déployé en production, réduisant ainsi les erreurs dues à des différences d'infrastructure. De plus, Docker facilite le déploiement rapide de l'application à grande échelle grâce à des outils tels que `Docker Compose` ou `Kubernetes`, si nécessaire.

— **Automatisation et intégration avec GitLab** : En combinant Docker avec les pipelines CI/CD de GitLab, nous pourrons automatiser l'exécution des tests et la validation du code dans des environnements Docker. Cela permet de vérifier automatiquement que l'application fonctionne correctement avant que les modifications ne soient fusionnées et déployées. L'automatisation des tests dans des conteneurs Docker garantit que les tests s'exécutent de manière cohérente, quel que soit l'environnement d'exécution.

— **Simplification du développement avec Docker Compose** : Pour le développement et les tests locaux, nous utiliserons `Docker Compose`, un outil qui permet de définir et de gérer plusieurs conteneurs simultanément. Cela nous permettra de lancer facilement tous les services dont l'application a besoin (par exemple, le backend, la base de données Neo4j, le serveur web) en une seule commande, rendant le processus de développement beaucoup plus fluide et évitant les erreurs de configuration.

7. Répartition des Tâches

Dans cette section, nous détaillons l'affectation des tâches au sein du groupe pour assurer une bonne organisation et une progression cohérente tout au long du projet. Chaque membre de l'équipe est responsable de différentes étapes du projet, en fonction de ses compétences et de son rôle.

7. 1 Tâches du Premier Semestre

1. **Étude des données Open Data** :
   — Responsable principal : **AYDIN Kieran** (Référent Données)
   — Collaborateurs : **Toute l'équipe**
   Cette tâche implique l'identification et l'analyse des données open data liées à Parcoursup, afin de déterminer leur structure et leur pertinence pour le projet.
2. **Conception du schéma de stockage des données** :
   — Responsable principal : **AYDIN Kieran** (Référent Données)
   — Collaborateurs : **OUEDRAOGO Constant**, **ECHAROUX Inès**, **BER Lucas**
   Cette tâche consiste à concevoir un schéma de base de données en utilisant Neo4j, en tenant compte des relations complexes entre les étudiants, les formations, et les décisions d'affectation.
3. **Développement du backend de l'application** :
   — Responsable principal : **NABE Mamoudou (Responsable d'équipe)**
   — Collaborateurs : **ECHAROUX Inès**, **AYDIN Kieran**, **BER Lucas**, **KUT Kemal**, **DIA Seydina**
   Le développement du backend se fait en utilisant Python et Flask, permettant de manipuler les données stockées et de les exposer via une API.
4. **Restitution des données sur un site web** :
   — Responsable principal : **OUEDRAOGO Constant (Référent développeur)**
   — Collaborateurs : **MAOUASSI Nouhad**, **NABE Mamoudou**, **DIALLO Aissatou**, **KEITA Sidy Mahmoud**
   La création de l'interface web pour la restitution des données sera réalisée en utilisant des bibliothèques de visualisation comme Chart.js et D3.js.
5. **Tests et validation** :
   — Responsable principal : **MAOUASSI Nouhad** (Référent Qualité)
   — Collaborateurs : **DIALLO Aissatou**, **KEITA Sidy Mahmoud**
   Les tests automatisés et manuels des fonctionnalités développées seront réalisés en s'assurant de la qualité du code à travers l'intégration continue avec GitLab.
6. **Mise en place des environnements de développement et de production** :
   — Responsable principal : **DIA Seydina Mandione** (DevOps)
   — Collaborateurs : **KUT Kemal**
   Cette tâche consiste à configurer Docker pour standardiser les environnements de développement, de test et de production.

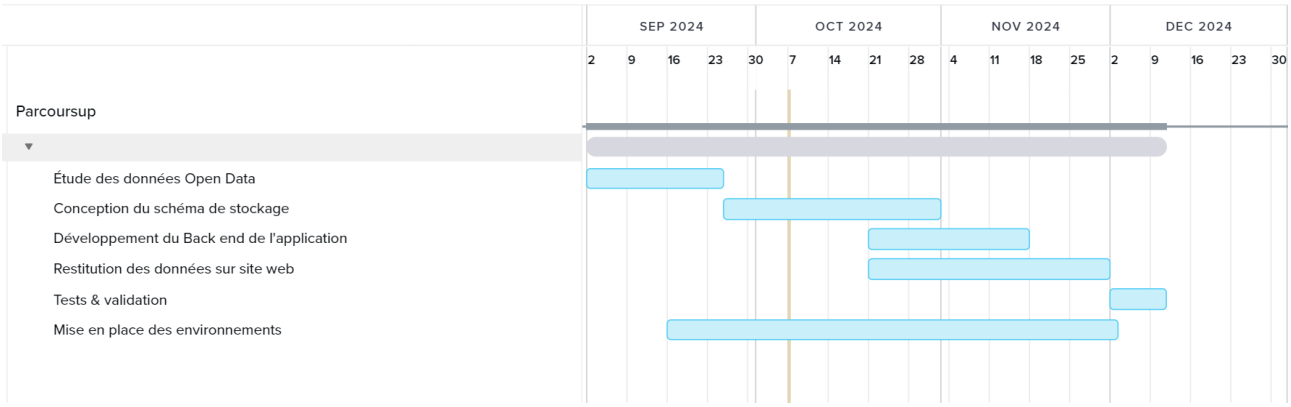

FIGURE 1.1 – Répartition des tâches

8. Conclusion

Ce projet tutoré nous offre l'opportunité de développer des compétences clés en gestion de projet, analyse de données, et développement d'applications web. En nous concentrant sur l'analyse de la plateforme Parcoursup pour la période 2020-2023, nous abordons un sujet pertinent et complexe qui soulève des enjeux importants en termes d'algorithmes d'appariement et de traitement de données interconnectées.

Le premier semestre sera principalement dédié à la mise en place du socle technique de l'application, comprenant la conception d'un schéma de stockage, l'étude de l'algorithme d'appariement, ainsi que la restitution des données sur un site web. Nous avons choisi d'utiliser Neo4j, une base de données orientée graphes, pour sa capacité à modéliser efficacement les relations complexes inhérentes à Parcoursup. Couplé à des outils de développement modernes comme Python et Flask, ce choix nous permet de créer une architecture flexible et scalable pour répondre aux exigences du projet.

La répartition des rôles au sein de notre groupe nous permet d'assurer une organisation cohérente et un bon suivi des tâches. Chaque membre contribue activement à la réalisation des différentes fonctionnalités, tout en respectant les bonnes pratiques de développement collaboratif grâce à GitLab. Bien que GitLab soit imposé, nous avons pleinement exploité ses fonctionnalités avancées pour la gestion du code source et l'automatisation des tests. De plus, l'utilisation de Docker pour l'uniformisation des environnements et le déploiement nous permettra d'assurer une stabilité tout au long du cycle de développement, en limitant les erreurs dues aux différences d'infrastructure.

Le deuxième semestre sera quant à lui centré sur l'approfondissement de l'analyse des données, avec pour objectif de proposer des améliorations à l'algorithme d'appariement en identifiant les biais potentiels et en étudiant des solutions innovantes. Nous mettrons également en avant des comparaisons territoriales et une étude spécifique sur l'Université de Bourgogne.

En conclusion, ce projet tutoré constitue un cadre idéal pour appliquer et approfondir nos compétences en ingénierie logicielle, gestion des données et développement collaboratif. À travers ce projet, nous visons à produire des résultats concrets et exploitables pour améliorer la performance de Parcoursup, tout en garantissant une expérience utilisateur fluide et une

restitution des données pertinente et accessible via une interface web.