

# GuideMetier

## Système de recommandation de carrière

The screenshot shows the GuideMetier website. On the left, there's a sidebar with a dark background featuring a smiling robot icon, the text "GuideMetier" and "Rencontrez votre futur métier", and three buttons: "Chatbot" (highlighted), "Liste des métiers", and "Centres d'intérêt". The main content area has a light gray background. At the top, it says "Chatbot". Below that is a message box containing a welcome message and instructions. A search bar below the message box contains the text "Je cherche un métier permettant le télétravail". To the left of the search bar, a list of potential careers is displayed in orange-outlined boxes: "Expert-comptable / Experte-comptable", "Développeur / Développeuse", "Chargé / Chargée de recrutement", "Ingénieur / Ingénieure système informatique", "Conseiller / Conseillère en gestion de patrimoine", "Contrôleur / Contrôleuse des impôts", and "Administrateur / Administratrice réseau, sécurité". At the bottom of the main area, there's an input field "Envoyer un message..." and a red "Envoyer" button.

**Aram BABAYAN**  
**L2 - Réseaux**

<b>I. Introduction.....</b>	<b>3</b>
<b>II. Contexte.....</b>	<b>3</b>
<b>III. Manuel de l'utilisateur.....</b>	<b>5</b>
1. Installation du projet .....	5
2. Lancement du projet.....	6
En ligne.....	6
En local via web.....	6
En local via CLI.....	6
3. Fonctionnalités du projet.....	7
1. Chatbot.....	7
2. Exemples de saisie .....	8
2. Liste des métiers .....	11
3. Fiche détaillée du métier .....	12
4. Menu contextuel.....	14
5. Offres d'emploi .....	15
6. Marché du travail.....	16
7. Métiers possibles .....	16
8. Métiers proches .....	18
9. Ce qu'il faut savoir .....	19
<b>IV. Choix des langages.....</b>	<b>20</b>
<b>V. Problèmes rencontrés.....</b>	<b>21</b>
1. Trouver des datasets pertinents en français.....	21
2. Récolté des données – Tentative avec Indeed .....	21
1. Scraping Simple.....	21
2. Utilisation d'un Navigateur Simulé .....	21
3. Échec dû au Captcha .....	21
3. Migration vers Pôle emploi .....	21
<b>VI. Rapport technique.....</b>	<b>22</b>
1. Récupération de la liste des métiers.....	22
2. Récupération des fiches (détail) des métiers .....	24
3. Réalisation d'un scraper de l'API Pôle emploi .....	26
3. Analyse et exploitation du fichier JSON.....	28
4. Réalisation d'un chatbot .....	33
5. Implémentation d'une nouvelle interface graphique .....	37
<b>VII. Structure et code source documenté .....</b>	<b>41</b>
Code source documenté .....	42
README.md.....	42
chatbot.py .....	43
serveur.py .....	46
requirements.txt.....	48
templates/index.html .....	48
static/chatbot.js .....	49
static/style.css.....	58
models/metier.py .....	61
models/metiers.py.....	65
api/job_offers_api.py .....	68
<b>VIII. Budget, temps alloué et coûts .....</b>	<b>69</b>
<b>IX. Pistes d'amélioration .....</b>	<b>69</b>
<b>X. Conclusion .....</b>	<b>70</b>
<b>XI. Sources.....</b>	<b>70</b>

## I. Introduction

Dans notre ère numérique, où les choix professionnels abondent et où le monde du travail évolue rapidement, trouver sa voie peut s'avérer complexe. Face à cette profusion d'opportunités, la nécessité d'un outil qui facilite la navigation dans le paysage professionnel devient impérative. C'est dans ce contexte qu'émerge "GuideMetier".

GuideMetier est une combinaison de pattern matching, de traitement automatique des langues et de scraping de données en temps réel.

Il est capable de suggérer des carrières basées sur les compétences et intérêts de l'utilisateur, mais également de récupérer et présenter des informations actualisées relatives aux offres d'emploi récemment publiées, fourni ainsi une perspective holistique et en temps réel du marché du travail. Il également offre une analyse du marché du travail en mettant en lumière des données cruciales comme le nombre d'offres d'emploi par rapport aux demandeurs, offrant ainsi une vision claire et en temps réel des dynamiques du marché.

## II. Contexte

Le programme a été développé en :

- **HTML5 et CSS3 avec Bootstrap 5.3** ont permis de créer une interface utilisateur intuitive et responsive. L'usage de **Bootstrap 5.3** a facilité l'implémentation de composants UI complexes, tout en garantissant une compatibilité cross-browser
- **JavaScript avec jQuery et moment.js** ont contribué à gérer la dynamique côté client de l'application. jQuery a simplifié la manipulation du DOM et la gestion des événements, tandis que Moment.js a été indispensable pour manipuler et formater les dates et les heures des offres d'emploi, le tout en respectant la localisation en français à partir du format YYYY-MM-DD.
- **Python 3** (avec les librairies Spacy, Flask et Requests)
  - **Flask** a permis de lancer le projet sous forme d'interface web (avec une interface d'un chatbot) et d'implémenter des API backend utilisables par le chatbot.
  - **Flask-CORS** a permis de résoudre les problèmes liés au partage de ressources entre origines, facilitant ainsi les requêtes intersites.
  - **Spacy** a été utilisé pour le traitement automatique du langage naturel, notamment pour l'analyse de texte et le pattern matching, essentiels pour générer des suggestions de métiers.
  - **Requests** a permis de communiquer efficacement avec des services externes via des appels API, notamment pour récupérer les offres d'emploi en lien avec les suggestions de métiers.

Le développement a été réalisé sur **IntelliJ IDEA**, un environnement de développement intégré (IDE) très complet de la gamme JetBrains (Android Studio, CLion, PhpStorm...).

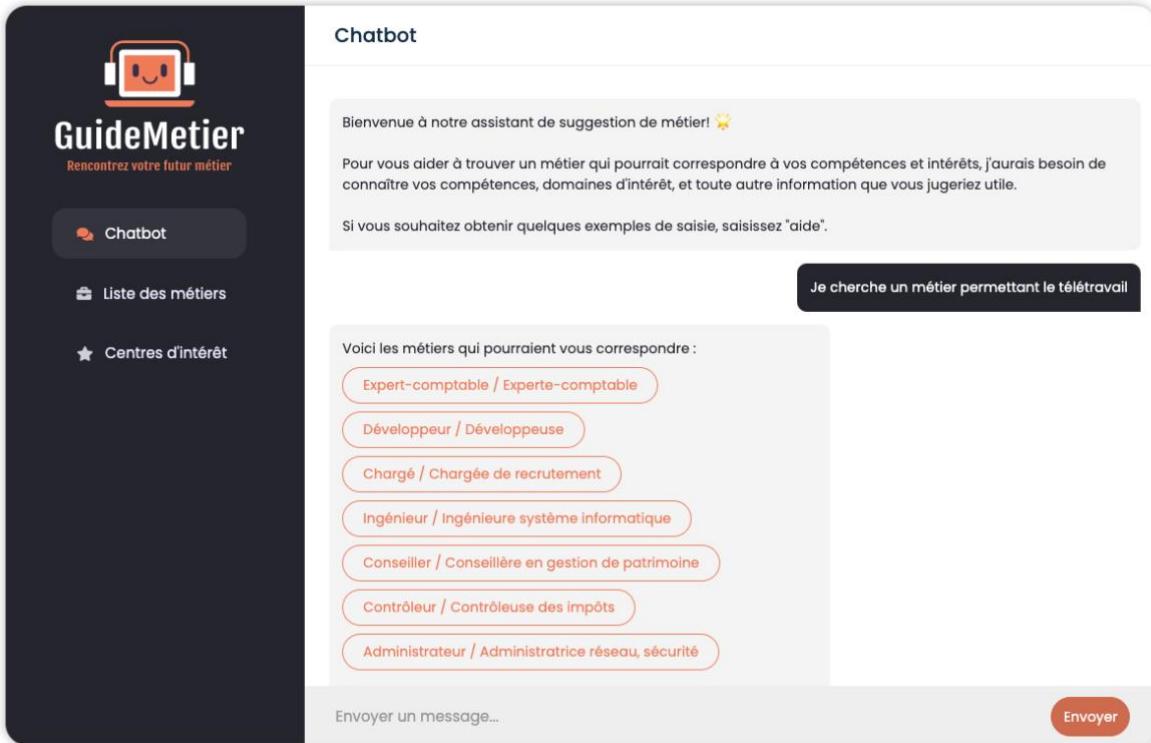
Plusieurs outils/plugins de ce IDE ont été utiles dans la réalisation du programme :

- Git (versioning, ajout de nouvelles fonctionnalités et structures petit à petit)
- Github Copilot (<https://github.com/features/copilot> - amélioration de la productivité pendant la réalisation en apportant des suggestions des solutions)

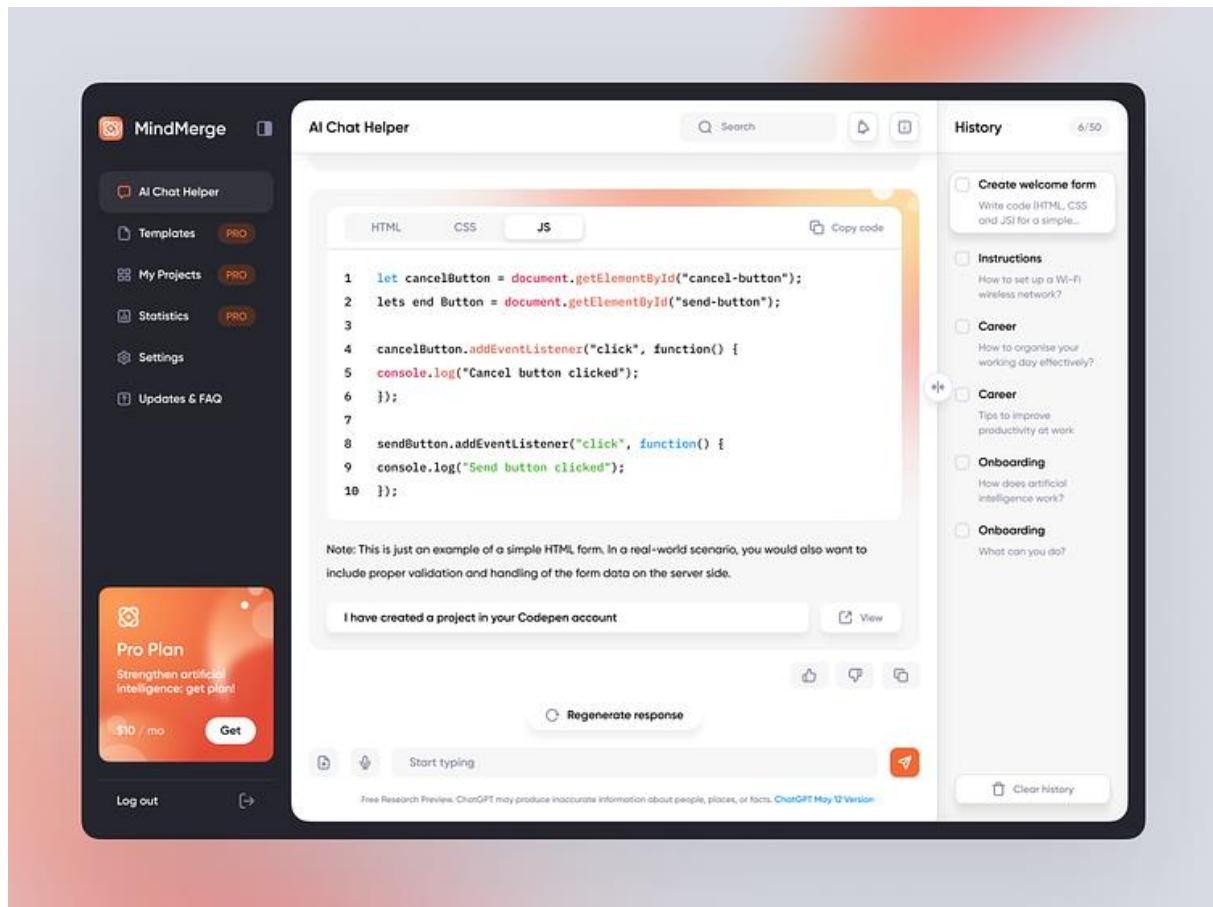
## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

Le programme dispose d'une interface graphique.



Celle-ci a été inspirée d'une maquette se trouvant sur le site Dribbble :  
<https://dribbble.com/shots/22026872-Chatbot-Web-UI> :



Source : <https://dribbble.com/shots/22026872-Chatbot-Web-UI>

## **III. Manuel de l'utilisateur**

### **1. Installation du projet**

Avant de lancer le projet, il est nécessaire d'installer certaines dépendances. Voici les étapes pour installer le projet :

#### **1. Accéder au dossier du projet :**

Naviguez jusqu'au dossier du projet en utilisant la commande

```
cd GuideMetier
```

#### **2. Installer Python :**

Assurez-vous que Python est installé sur votre machine. Si ce n'est pas le cas, vous pouvez le télécharger et l'installer à partir du site officiel de Python.

#### **3. Créer un environnement virtuel (optionnel) :**

Il est recommandé de créer un environnement virtuel pour isoler les dépendances du projet.

Pour cela exécutez la commande suivante :

```
python -m venv venv
```

Activez l'environnement virtuel :

- **Sur Windows :**  
venv\Scripts\activate
- **Sur macOS et Linux :**  
source venv/bin/activate

#### **4. Installer les dépendances :**

Installez les dépendances listées dans le fichier requirements.txt. Voici les dépendances du projet :

- Flask 2.2.0 : Python >=3.5
- Flask-CORS : Python >=2.7
- Spacy 3.6.1 : Python >=3.6,<3.10
- Requests 2.31.0 : Python >=3.6

Utilisez la commande suivante pour installer toutes les dépendances en une seule fois :

```
pip install -r requirements.txt
```

#### **5. Téléchargez le modèle fr\_core\_news\_sm pour Spacy**

Vous devez ensuite télécharger le modèle fr\_core\_news\_sm pour Spacy avec la commande :

```
python -m spacy download fr_core_news_sm
```

En fonction de votre version de Python par défaut vous pouvez être obligé d'utiliser la commande python3 à la place de python si celle-ci n'est pas en version 3.

Une fois ces étapes terminées, vous pouvez passer à l'étape de lancement du projet.

## 2. Lancement du projet

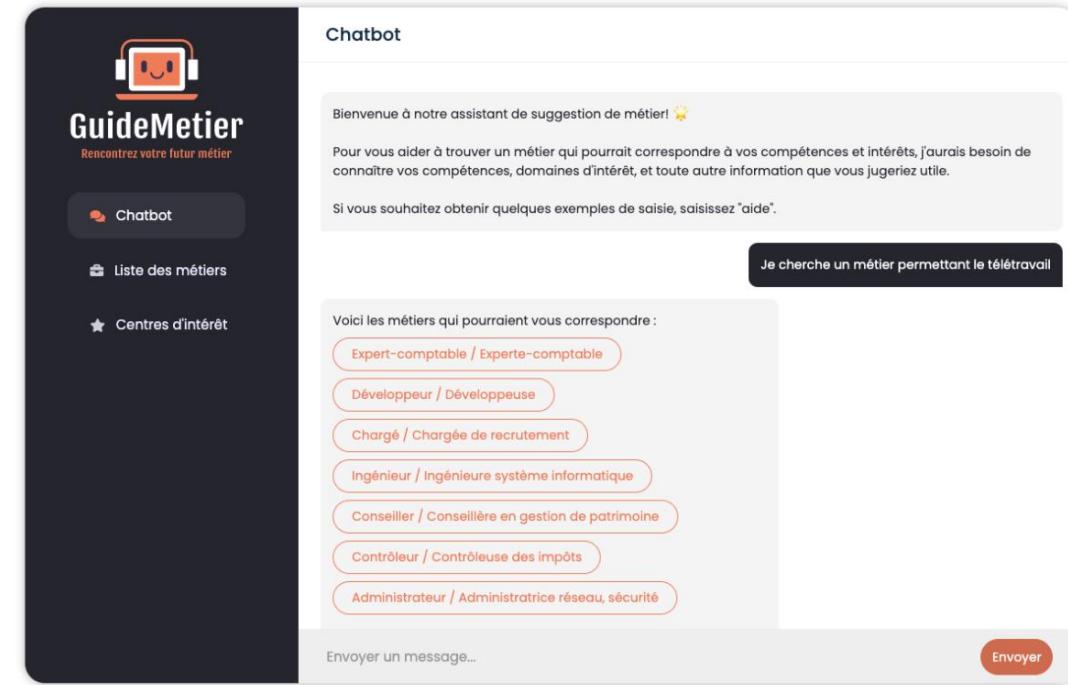
Le projet peut être lancé de trois manières. Chacune de ces méthodes vous donne accès à différentes facettes du projet.

**Note Importante :** Dans les trois cas de lancement, le projet nécessite d'initier l'ensemble des patterns à la bibliothèque Spacy. Ce processus peut entraîner un délai d'attente lors du démarrage (variant de 30 secondes à 2 minutes).

Une fois l'initialisation terminée, le chatbot répond immédiatement aux requêtes sans attente nécessaire.

### En ligne

Pour accéder à la version en production (publiée) du site, ouvrez votre navigateur web et accédez à l'adresse <https://guidemetier.lxc.fr/>



### En local via web

Vous avez également la possibilité de lancer le projet en local avec le serveur web local avec la commande suivante :

```
python serveur.py
```

Une fois le serveur lancé, Flask rendra le site accessible via votre navigateur web à l'adresse locale <http://localhost:5000/>.

### En local via CLI

Si vous préférez une version plus simple et textuelle, vous pouvez lancer le chatbot en mode terminal en exécutant la commande suivante :

```
python chatbot.py
```

Attention, la version CLI est plus limitée et ne fournira que les noms et les codes des métiers trouvant par rapport à votre saisie sans retourner des réponses détaillées.

```
raspberrypi:~ sh-3.2# python chatbot.py
Démarrage du chatbot...
```

Le projet nécessite d'initier l'ensemble des patterns à la bibliothèque Spacy, ce qui peut nécessiter un délai d'attente relativement long au démarrage.

### 3. Fonctionnalités du projet

Le projet est composé des fonctionnalités suivantes :

#### 1. Chatbot

Le chatbot est composé d'une partie front (site web) et d'une partie backend (API Flask qui renvoie la réponse). Le projet dispose d'une interface permettant de dialoguer avec le chatbot pour obtenir des suggestions de métiers basées sur vos compétences et intérêts.

Voici un exemple :

The screenshot shows the GuideMetier website interface. On the left, there is a dark sidebar with the logo 'GuideMetier' and the tagline 'Rencontrez votre futur métier'. It features three buttons: 'Chatbot' (highlighted in orange), 'Liste des métiers', and 'Centres d'intérêt'. The main content area has a light gray background. At the top right, there is a button labeled 'Je cherche un travail acceptant le télétravail'. Below it, a section titled 'Voici les métiers qui pourraient vous correspondre:' lists various job titles in orange-outlined boxes. At the bottom of the page, there is a text input field 'Envoyer un message...' and a red 'Envoyer' button.

- Planneur / Planteuse stratégique
- Secrétaire
- Responsable relation client (CRM)
- Employé / Employée de laboratoire cinématographique
- Conseiller / Conseillère en gestion de patrimoine
- Enquêteur / Enquêteuse sondage
- Chargé / Chargée de recrutement
- Analyste financier / financière
- Formateur / Formatrice
- Analyste de crédits et risques bancaires
- Chargé / Chargée de communication
- Assistant / Assistante ressources humaines
- Comptable
- Consultant / Consultante informatique, Product Owner
- Responsable marketing
- Directeur / Directrice d'agence bancaire
- Ingénieur / Ingénierie système informatique
- Technicien / Technicienne informatique

## 2. Exemples de saisie

Voici quelques exemples de saisies qui sont détectés facilement :

### Requête :

Bonjour! Je suis passionné par la technologie, surtout le développement web. Je connais le HTML, CSS et JavaScript, et j'ai déjà créé quelques sites web pour des amis.

### Réponse (interne) :

Compétence détectée: HTML, Label: M1805

Compétence détectée: JavaScript, Label: E1205

Vous pourriez envisager les métiers suivants : Développeur / Développeuse, Graphiste, Graphiste (code: M1805, E1205, E1205)

The screenshot shows a mobile application interface for 'GuideMetier'. On the left, there's a dark sidebar with the 'GuideMetier' logo and the tagline 'Rencontrez votre futur métier'. It has three buttons: 'Chatbot' (selected), 'Liste des métiers', and 'Centres d'intérêt'. The main area is titled 'Chatbot' and contains a message from the bot: 'Bienvenue à notre assistant de suggestion de métier! ☀️'. Below it, a message from the user reads: 'Bonjour! Je suis passionné par la technologie, surtout le développement web. Je connais le HTML, CSS et JavaScript, et j'ai déjà créé quelques sites web pour des amis.' The bot responds with: 'Voici les métiers qui pourraient vous correspondre :'. Two suggestions are listed in orange rounded rectangles: 'Développeur / Développeuse' and 'Graphiste'. At the bottom, there's an input field 'Envoyer un message...' and a red 'Envoyer' button.

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

Voici quelques autres exemples de saisies :

### Requête :

Je suis curieux au sujet du Designer UI/UX. Pouvez-vous m'en dire plus?

### Réponse (interne) :

Compétence détectée: Designer, Label: H1204

Vous pourriez envisager les métiers suivants : Designer industriel / Designeuse industrielle (code: H1204)

Je suis curieux au sujet du Designer UI/UX. Pouvez-vous m'en dire plus?

Voici les métiers qui pourraient vous correspondre :

Designer industriel / Designeuse industrielle

### Requête :

Ma passion pour le théâtre m'a amené à jouer dans plusieurs productions locales. Par ailleurs, je joue du piano depuis mon enfance.

### Réponse :

Ma passion pour le théâtre m'a amené à jouer dans plusieurs productions locales. Par ailleurs, je joue du piano depuis mon enfance.

Voici les métiers qui pourraient vous correspondre :

Comédien / Comédienne

Musicien / Musicienne

Professeur / Professeure d'art, de musique

### Requête :

Je joue du piano depuis mon enfance

### Réponse :

je joue du piano depuis mon enfance

Voici les métiers qui pourraient vous correspondre :

Professeur / Professeure d'art, de musique

Musicien / Musicienne

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

### Requête :

Je suis chef de projet web avec une expertise en marketing digital. J'ai également écrit plusieurs articles en tant que journaliste.

### Réponse :

Je suis chef de projet web avec une expertise en marketing digital. J'ai également écrit plusieurs articles en tant que journaliste.

Voici les métiers qui pourraient vous correspondre :

Média planneur / planneuse

Responsable marketing

### Requête :

J'ai une maîtrise en droit et j'ai principalement travaillé dans le domaine du droit commercial. J'ai également aidé des entreprises dans leur développement économique

### Réponse :

J'ai une maîtrise en droit et j'ai principalement travaillé dans le domaine du droit commercial.  
J'ai également aidé des entreprises dans leur développement économique

Voici les métiers qui pourraient vous correspondre :

Courtier / Courtière en assurances

Cadre dirigeant / dirigeante d'administration centrale

Chef / Cheffe de département en grande distribution

Responsable commercial / commerciale

Consultant / Consultante en organisation et management

Responsable de magasin

Président Directeur Général / Présidente Directrice Générale (PDG)

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

## 2. Liste des métiers

Le site dispose d'un bouton "Liste des métiers" qui vous permet de consulter une liste exhaustive de 550 métiers. Chaque métier est cliquable et vous redirige vers une fiche détaillée.

The screenshot shows the GuideMetier website interface. On the left, there's a sidebar with a logo of a smiling robot head wearing headphones, the text "GuideMetier" and "Rencontrez votre futur métier", and three buttons: "Chatbot", "Liste des métiers" (which is highlighted), and "Centres d'intérêt". The main content area has a header "Chatbot" and a message from the chatbot: "Bienvenue à notre assistant de suggestion de métier! ☀️ Pour vous aider à trouver un métier qui pourrait correspondre à vos compétences et intérêts, j'aurais besoin de connaître vos compétences, domaines d'intérêt, et toute autre information que vous jugeriez utile. Si vous souhaitez obtenir quelques exemples de saisie, saisissez "aide".". A button "Je souhaite voir la liste des métiers" is visible. Below this, a heading "Voici la liste de tous les métiers :" is followed by a vertical list of 15 clickable cards, each containing a pair of professions separated by a slash: "Conducteur / Conductrice de tracteur", "Bûcheron / Bûcheronne, Élagueur / Élagueuse", "Agent / Agente d'entretien des espaces naturels", "Jardinier / Jardinière", "Garde forestier / forestière", "Forestier / Forestière", "Conseiller / Conseillère agricole", "Technicien / Technicienne agricole", "Ingénieur / Ingénieure agronome", "Vendangeur / Vendangeuse", "Aide agricole", and "Aide d'élevage". At the bottom of the list is a text input field "Envoyer un message..." and a red "Envoyer" button.

La liste est composée de 550 métiers, donc reste relativement longue à parcourir, cependant cela permet d'avoir un aperçu global sur l'ensemble des métiers possibles lorsque l'interaction avec le chatbot ne suffit pas pour trouver un métier intéressant.

Il suffit ensuite de cliquer sur un résultat pour accéder à la fiche détaillé du métier qui permettra de consulter les informations détaillées relatives au métier sélectionné

This screenshot shows a detailed view of the "Liste des métiers" section. It displays a vertical list of 10 clickable cards, each containing a pair of professions separated by a slash: "Responsable d'exploitation transport routier de personnes", "Exploitant / Exploitante transport routier de marchandises", "Agent / Agente d'exploitation transport routier de personnes", "Conducteur / Conductrice de train", "Agent / Agente de contrôle", "Aiguilleur / Aiguilleuse du rail", "Agent / Agente d'exploitation des remontées mécaniques", and "Cheminot / Cheminote". At the bottom of the list is a text input field "Envoyer un message..." and a red "Envoyer" button.

### 3. Fiche détaillée du métier

La fiche détaillée du métier fournit des informations complètes, y compris les missions principales, le domaine d'activité, les conditions d'accès, les conditions de travail, les horaires, les compétences requises et même une vidéo de présentation du métier.

The screenshot shows the GuideMetier platform interface. On the left sidebar, there's a logo of a smiling robot head with headphones, followed by the text "GuideMetier" and "Rencontrez votre futur métier". Below this are three buttons: "Chatbot" (with a microphone icon), "Liste des métiers" (with a list icon), and "Centres d'intérêt" (with a star icon). The main content area has a title "Chatbot" and a list of related professions: Développeur / Développeuse, Technicien / Technicienne support technique, Chef / Cheffe comptable, Responsable commercial / commerciale, and Technicien / Technicienne de maintenance en informatique. A button at the bottom says "Je souhaite en savoir plus sur le métier de Ingénieur / Ingénierie système informatique". The central panel details the "Ingénieur / Ingénierie système informatique" role, mentioning expertise in system information and the domain of informatics, internet, and telecommunications. It lists primary missions such as advising management on IT evolution, supporting teams, and ensuring quality and security. It also specifies requirements like a Master's degree in informatics and English proficiency. A section titled "Comment y accéder" discusses entry requirements, and another titled "Ce qu'il faut savoir faire:" lists skills in communication, innovation, data processing, research, and economics. At the bottom, there's a text input field "Envoyer un message..." and a red "Envoyer" button.

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

The screenshot shows a detailed job profile for a Network Engineer. On the left, there's a sidebar with a logo of a smiling robot head, the text "GuideMetier", and "Rencontrez votre futur métier". Below it are three buttons: "Chatbot" (with a speech bubble icon), "Liste des métiers" (with a document icon), and "Centres d'intérêt" (with a star icon). The main content area has a title "Chatbot" and several sections:

- Développement économique (22%)**
  - Relation client
  - Développement commercial
  - Stratégie de développement
- Collaboration, Management, Social, Soin (14%)**
  - Apprentissage et professionnalisation
  - Management
  - Conseil, Transmission

**Conditions de travail:**

- Déplacements professionnels
- Possibilité de télétravail

**Horaires et durée de travail:**

- Travail en astreinte

**Savoir-être professionnels**

- Travailler en équipe
- Prendre des initiatives et être force de proposition
- Faire preuve d'autonomie

**Vidéo de présentation:**

A video thumbnail showing a man in a server room. The thumbnail includes the text "Ingénieur système-réseau / Ingénierie système..." and "Regarder sur YouTube". There are also "À regarder" and "Partager" buttons.

At the bottom, there's a text input field "Envoyer un message..." and a red "Envoyer" button.

A la fin de la fiche détaillée, le site affiche un menu contextuel proposant d'accéder et d'afficher plus d'informations relatives au métier sélectionné :

The screenshot shows the same detailed job profile for a Network Engineer, but with a different layout. The video thumbnail is larger and centered. Below it is a light gray box containing a question and five buttons:

Souhaitez en savoir plus sur ce métier ?

- Ce qu'il faut savoir
- Métiers possibles
- Offres d'emploi
- Métiers proches
- Marché du travail

At the bottom, there's a text input field "Envoyer un message..." and a red "Envoyer" button.

Le menu contextuel sert à optimiser l'affichage des informations sur chaque fiche métier en ne montrant les différentes fonctionnalités supplémentaires que sur demande de l'utilisateur. Cette approche est particulièrement bénéfique pour des options comme "Offres d'emploi", qui requièrent un appel API en temps réel à service de

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

scraping. En limitant ces appels aux moments où l'utilisateur en fait explicitement la demande, nous minimisons le risque de bannissement par le site externe ciblé.

### 4. Menu contextuel

Après avoir consulté une fiche métier, un menu contextuel apparaît avec plusieurs options pour en savoir plus sur le métier sélectionné.

Les options incluent

- Ce qu'il faut savoir
- Métiers possibles
- Offres d'emploi
- Métiers proches
- Marché du travail

The screenshot shows a dark-themed web page for 'GuideMetier'. On the left, there's a sidebar with a robot icon, the 'GuideMetier' logo, and three buttons: 'Chatbot', 'Liste des métiers', and 'Centres d'intérêt'. The main content area has a light background. At the top right, there's a section titled 'Chatbot' with a list of 'Savoir-être professionnels' and a video thumbnail for 'Développeur Full Stack'. Below this, a large button asks if you want to know more about the job, with five options: 'Ce qu'il faut savoir', 'Métiers possibles', 'Offres d'emploi', 'Métiers proches', and 'Marché du travail'. At the bottom, there's a text input field 'Envoyer un message...' and a red 'Envoyer' button.

Après avoir cliqué sur une de ces options, la fonctionnalité en question s'affichera :

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

### 5. Offres d'emploi

Au clic sur l'action contextuelle « Offres d'emploi », le site présentera un tableau des dernières offres d'emploi postées sur Pôle emploi pour le métier en question :

Je souhaite voir les offres d'emploi récentes pour ce métier				
Date	Titre	Localisation	Contrat	Description
il y a un jour	Développeur Front-end React.JS (H/F)	LYON 01	CDI	<p>Notre client est une agence de e-commerce dans le tourisme. La société recherche un développeur Front-end React.JS afin de soutenir le développem...</p> <p>Voir l'offre</p>
il y a un jour	Développeur / Développeuse full-stack	LYON 01	CDI	<p>Notre client est une agence de e-commerce dans le tourisme. La société recherche un développeur Full-Stack Node.JS React.JS confirmé afin de sout...</p> <p>Entreprise AK Recrutement est un cabinet de recrutement spécialisé dans le secteur de l'informatique en région Auvergne-Rhône-Alpes. Nous couv...</p> <p>-Membre actif.ve d'une des équipes Agile, vous intervenez sur toutes les étapes du cycle de développement : -De la compréhension du besoin au déplo...</p> <p>PB Solutions recherche pour son Client équipementier aéronautique au sein d'un groupe international un Ingénieur Certification Logiciel H/F sur Châ...</p> <p>Voir l'offre</p>
il y a un jour	Lead Developer Logiciel C++ QT (H/F)	GRENOBLE	CDI	
il y a un jour	Développeur / Développeuse full-stack (H/F)	GRENOBLE	CDI	
il y a un jour	Ingénieur Certification Logiciel H/F (H/F)	CHATEAUROUX	CDI	

Le tableau est trié par date et comprend des détails tels que le titre, la description, le type de contrat, et un bouton "Voir l'offre" qui redirige vers l'offre d'emploi complète sur le site de Pôle emploi avec la possibilité de postuler :

The screenshot shows the Pôle emploi website interface. At the top, there are logos for the French Republic and Pôle emploi, followed by a menu icon and 'MENU'. Below the header, there are links for 'ESPACE PERSONNEL' and 'MA RECHERCHE D'OFFRES'. The main content area displays a job offer for a 'Développeur / Développeuse full-stack (H/F)' in Grenoble. The offer is dated August 26, 2023. It includes a brief description of the role, mentioning Agile development, deployment to production, and experience with UX/UI. To the right of the description, there are three circular icons for favoriting, sharing, and more options. A sidebar on the right provides details about the contract (indeterminate duration), working hours (35h), and salary (annual 35000€, monthly 55000€ over 12 months).

Offre n° 160KVMD

## Développeur / Développeuse full-stack (H/F)

38 - GRENOBLE - [Localiser avec Mappy](#)

Actualisé le 26 août 2023

-Membre actif.ve d'une des équipes Agile, vous intervenez sur toutes les étapes du cycle de développement :

-De la compréhension du besoin au déploiement en production en passant par les phases d'analyse, d'études, et de développement.

-En tant que développeur(euse) Web Full-Stack, vous avez à la fois une appétence pour les défis de robustesse et d'extensibilité d'un système distribué ainsi que pour les problématiques UX et UI d'applications web responsives.

-Vous savez que les meilleures solutions techniques sont le résultat d'une

Contrat à durée indéterminée  
 Contrat travail

35H Travail en journée

Salaire brut : Annuel de 35000,00 Euros à 55000,00 Euros sur 12 mois

## 6. Marché du travail

Cette section présente des statistiques clés sur le marché du travail pour le métier spécifique que vous consultez. Elle affiche le nombre total d'offres d'emploi déposées sur Pôle emploi au cours des 12 derniers mois ainsi que le nombre actuel de demandeurs d'emploi pour ce métier.

**Chatbot**

**Je souhaite en savoir plus sur le marché du travail pour ce métier**

**Le marché du travail**

✉ Offres d'emploi: **224810** déposées sur Pôle emploi au cours des 12 derniers mois.  
👤 Demandeurs d'emploi: **48070**

Bonne nouvelle ! Ce métier dispose de 5x plus d'offres d'emploi que de demandeurs d'emploi.

Envoyer un message... **Envoyer**

Sur la base de ces données, la section fournit une analyse comparative pour indiquer si le métier est en forte demande ou non. Par exemple, si le nombre d'offres d'emploi est significativement plus élevé que le nombre de demandeurs d'emploi, un message positif s'affichera pour indiquer que le métier dispose de plus d'opportunités que de candidats disponibles.

## 7. Métiers possibles

Cette section liste tous les métiers possibles que vous pouvez exercer en ayant un certain diplôme ou titre professionnel. Par exemple, un "Ingénieur informatique" peut exercer plusieurs métiers différents, et cette section vous donne une liste complète de ces options.

**Métiers possibles:**

- Analyste cogniticien / cogniticienne informatique
- Analyste d'application
- Analyste développeur / développeuse
- Analyste fonctionnel / fonctionnelle informatique
- Analyste organique informatique
- Analyste réseau informatique
- Analyste responsable d'application informatique
- Analyste télématicien
- Analyste-programmeur / Analyste-programmeuse de micro-informatique
- Analyste-programmeur / Analyste-programmeuse d'étude informatique
- Analyste-programmeur / Analyste-programmeuse en informatique industrielle
- Analyste-programmeur / Analyste-programmeuse gestion informatique
- Analyste-programmeur / Analyste-programmeuse informatique
- Chef de groupe logiciel informaticien
- Chef de projet étude et développement informatique
- Concepteur / Conceptrice d'application informatique
- Concepteur / Conceptrice informatique
- Concepteur / Conceptrice logiciel informatique
- Développeur / Développeuse d'application
- Développeur / Développeuse informatique
- Développeur / Développeuse multimédia
- Développeur / Développeuse web

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

The screenshot shows the GuideMetier website's "Chatbot" section. On the left, there's a sidebar with links: "Chatbot" (highlighted in orange), "Liste des métiers", and "Centres d'intérêt". The main content area has a header "Chatbot" and a question "Souhaitez en savoir plus sur ce métier ?". Below it are five buttons: "Ce qu'il faut savoir", "Métiers possibles" (highlighted in orange), "Offres d'emploi", "Métiers proches", and "Marché du travail". A large button at the bottom right says "Je souhaite en savoir plus sur les métiers possibles". The central column contains a heading "Métiers possibles:" followed by a long list of IT-related professions. At the bottom, there's a text input field "Envoyer un message..." and an "Envoyer" button.

Souhaitez en savoir plus sur ce métier ?

Chatbot

GuideMetier

Rencontrez votre futur métier

Chatbot

Liste des métiers

Centres d'intérêt

Métiers possibles

Offres d'emploi

Métiers proches

Marché du travail

Je souhaite en savoir plus sur les métiers possibles

**Métiers possibles:**

- Analyste cognitien / cognitienne informatique
- Analyste d'application
- Analyste développeur / développeuse
- Analyste fonctionnel / fonctionnelle informatique
- Analyste organique informatique
- Analyste réseau informatique
- Analyste responsable d'application informatique
- Analyste télématicien
- Analyste-programmeur / Analyste-programmeuse de micro-informatique
- Analyste-programmeur / Analyste-programmeuse d'étude informatique
- Analyste-programmeur / Analyste-programmeuse en informatique industrielle
- Analyste-programmeur / Analyste-programmeuse gestion informatique
- Analyste-programmeur / Analyste-programmeuse informatique
- Chef de groupe logicien informatien
- Chef de projet étude et développement informatique
- Concepteur / Conceptrice d'application informatique
- Concepteur / Conceptrice informatique
- Concepteur / Conceptrice logiciel informatique
- Développeur / Développeuse d'application
- Développeur / Développeuse informatique
- Responsable de gestion de configuration
- Testeur / Testeuse informatique
- Développeur / Développeuse logiciel de réalité augmentée
- Scrum Master
- Ingénieur informaticien / Ingénierie informatienne
- Développeur / Développeuse décisionnel - Business Intelligence
- Lead programmeur / programmeuse - jeux vidéo
- Programmeur / Programmeuse - jeux vidéo
- Développeur / Développeuse front-end
- Développeur / Développeuse de sécurité des systèmes d'information
- informaticien / Informaticienne analyste
- Informaticien chargé / Informaticienne chargée d'étude
- Ingénieur analyse-programmeur / Ingénierie analyse-programmeuse
- Ingénieur concepteur / Ingénierie conceptrice informatique
- Développeur / Développeuse full-stack
- Ingénieur / Ingénierie d'analyse et de programmation en informatique de gestion
- Ingénieur / Ingénierie de développement informatique
- Ingénieur / Ingénierie de réalisation informatique
- Ingénieur / Ingénierie d'étude en informatique de gestion
- Ingénieur / Ingénierie d'étude informatique
- Ingénieur / Ingénierie d'intégration applicative
- Développeur / Développeuse - jeux vidéo
- Développeur / Développeuse web mobile
- Analyste en cybersécurité
- Intégrateur / Intégratrice d'application informatique
- Programmeur / Programmeuse de maintenance informatique
- Développeur / Développeuse Big Data
- Développeur / Développeuse de logiciel de simulation optique et réalité virtuelle
- Webmaster développeur / développeuse
- Développeur / Développeuse back-end
- Chef de projet TMA - Tiers Maintenance Applicative
- Chef de projet maîtrise d'œuvre informatique
- Homologateur / Homologatrice logiciel
- Analyste concepteur / conceptrice informatique
- Analyste de gestion informatique
- Analyste d'étude informatique
- Analyste décisionnel - Business Intelligence
- Technicien / Technicienne programmation
- Analyste-programmeur / Analyste-programmeuse scientifique informatique
- Responsable de projet informatique
- Assistant / Assistante chef de projet informatique

Envoyer un message...

Envoyer

Cette liste est très exhaustive, voire parfois trop longue (à parcourir) en fonction du métier.

## 8. Métiers proches

Cette section affiche une liste de métiers qui sont similaires ou liés au métier que vous consultez, vous donnant ainsi plus d'options à considérer.

Je souhaite en savoir plus sur les métiers proches

**Métiers proches:**

- Conseil et maîtrise d'ouvrage en systèmes d'information
- Expertise et support en systèmes d'information
- Études et développement de réseaux de télécoms
- Direction des systèmes d'information
- Conception de produits touristiques

Chatbot

Regarder sur YouTube

Souhaitez en savoir plus sur ce métier ?

- Ce qu'il faut savoir
- Métiers possibles
- Offres d'emploi
- Métiers proches
- Marché du travail

Envoyer un message...

Envoyer

Je souhaite en savoir plus sur les métiers proches

**Métiers proches:**

- Conseil et maîtrise d'ouvrage en systèmes d'information
- Expertise et support en systèmes d'information
- Études et développement de réseaux de télécoms
- Direction des systèmes d'information
- Conception de produits touristiques

## 9. Ce qu'il faut savoir

Cette section fournit des informations détaillées sur le métier, y compris les domaines d'expertise, les normes professionnelles, les techniques utilisées et d'autres informations pertinentes pour le métier sélectionné.

Voici un exemple pour le métier de développeur :

### Chatbot

Je souhaite en savoir plus sur ce qu'il faut savoir dans ce métier

**Domaines d'expertise:**

- Ada
- Algorithmique
- Apache
- ASP.NET
- AutoitScript
- Basic
- C#
- C/C++
- Cobol
- CSS
- Delphi
- Fortran
- HTML
- Java
- J2EE
- J2ME
- Langages de programmation informatique
- Lisp
- Pascal
- Perl
- PHP
- Prolog
- Python
- Ruby
- Script Shell
- SQL
- VB.NET
- Visual Basic
- WLanguage
- XML
- Application web

**Normes et procédés:**

- Animation d'instances agiles (mêlée, démonstration, rétrospective, ...)
- Formalisation de processus de gestion
- Marketing / Mercatique
- Méthode AGILE
- Méthodologie SAFe
- Méthodologie Scrum
- Modélisation informatique
- Principes d'intégration de matériels et de logiciels
- Procédures d'appels d'offres
- Protocoles et normes télécoms
- Urbanisation des systèmes d'information
- Règles de sécurité Informatique et Télécoms
- Système KANBAN
- Système LEAN

**Produits, outils et matières:**

- Matériel de contrôle embarqué
- Réseaux informatiques et télécoms
- Produits multimédias

## IV. Choix des langages

Pour la réalisation de ce projet, plusieurs langages et frameworks ont été utilisés, chacun pour des raisons spécifiques :

**Python:** Ce langage a été choisi pour son excellente prise en charge des bibliothèques de traitement du langage naturel comme Spacy. De plus, Python est largement utilisé dans le domaine de l'analyse de données et du machine learning, ce qui le rend idéal pour le pattern matching et l'analyse des compétences et des intérêts. Sa syntaxe claire et sa communauté active en font un choix naturel pour le développement rapide et efficace.

**Spacy:** Spécifiquement choisi pour le traitement du langage naturel. Il offre des fonctionnalités avancées comme la reconnaissance d'entités nommées (l'entraînement d'un modèle ou usage d'un modèle existant), la lemmatisation, la segmentation de texte, et la gestion des formes plurielles et singulières des mots. Ces capacités sont indispensables pour le pattern matching évolué, l'analyse sémantique et la personnalisation des recommandations de carrière.

**Flask:** Ce micro-framework Python a été choisi pour le développement du backend surtout car il est léger, facile à utiliser et parfaitement adapté pour créer des API REST.

**Requests:** Cette bibliothèque Python a été utilisée pour le scraping de données et les appels API. Elle est très simple à utiliser et efficace pour récupérer des données depuis le web.

**HTML/CSS:** Utilisés pour le développement de l'interface utilisateur. Ils offrent une grande flexibilité dans la conception de l'UI et sont compatibles avec tous les navigateurs modernes.

**JavaScript (Frontend):** Utilisé pour créer une interface utilisateur interactive. Il fonctionne bien avec divers frameworks frontend et est essentiel pour communiquer avec le backend via des API REST.

**jQuery:** Cette bibliothèque JavaScript a été intégrée pour simplifier la manipulation du DOM, la gestion des événements et les appels AJAX. Elle est largement utilisée et bien documentée, ce qui accélère le développement et facilite la maintenance.

**Bootstrap:** Utilisé pour le développement frontend, ce framework CSS offre une variété de composants UI pré-conçus qui sont à la fois esthétiques et fonctionnels. Il permet une conception réactive sans avoir à écrire beaucoup de CSS personnalisé.

## V. Problèmes rencontrés

Au cours du développement de "GuideMetier", j'ai fait face à plusieurs défis :

### 1. Trouver des datasets pertinents en français.

L'un des obstacles majeurs lors de la création de "GuideMetier" a été le manque de datasets en français, riches en détails sur diverses carrières. Bien que de nombreuses bases de données soient accessibles, notamment sur Kaggle, elles sont majoritairement en anglais et souvent inadaptées au contexte professionnel français, ou incomplètes pour aborder notre problématique de façon exhaustive.

Face à cette insuffisance, j'ai dû redoubler d'efforts pour recueillir et traiter manuellement les données à partir de sites d'offres d'emploi en utilisant la technique de web scraping.

### 2. Récolté des données – Tentative avec Indeed

#### 1. Scraping Simple

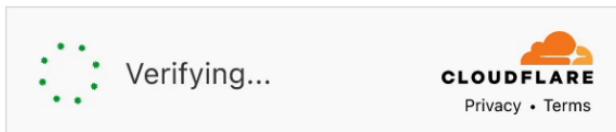
Ma première approche avec Indeed s'est basée sur un scraping simple, c'est-à-dire sans simuler un navigateur. Cette méthode, souvent efficace sur d'autres sites, s'est heurtée rapidement aux protections renforcées de CloudFlare présentes sur Indeed.

#### 2. Utilisation d'un Navigateur Simulé

Face aux complications initiales, j'ai tenté d'utiliser un robot qui simule un navigateur dans l'espoir de contourner les protections. Cela semblait être une solution prometteuse car elle permet d'imiter les actions d'un utilisateur humain. Cependant, même cette approche s'est avérée être un défi en soi.

#### 3. Échec dû au Captcha

Malgré mes efforts pour simuler un navigateur, les protections de CloudFlare ont fini par détecter l'activité automatisée. La demande récurrente de validation par captcha est devenue un obstacle demandant un effort trop important, rendant cette méthode d'extraction de données depuis Indeed pratiquement non faisable dans les délais.



### 3. Migration vers Pôle emploi

Après les obstacles rencontrés avec Indeed, j'ai exploré plusieurs sources à la recherche de données de qualité requérant un effort modéré de collecte. Mon exploration m'a finalement conduit au site de Pôle emploi.

Après une analyse détaillée de la structure de site j'ai redirigé mes efforts vers ce site qui s'est avéré être une mine d'or d'informations pertinentes.

Non seulement il offre une base de données complète sur les métiers, mais il détaille aussi les offres d'emploi, le marché du travail, et bien plus. Sa structure m'a permis de collecter les données nécessaires sans énormément de difficultés.

## VI. Rapport technique

### 1. Récupération de la liste des métiers

Le site de Pôle emploi dispose d'une liste exhaustive de 550 métiers qui mènent aux fiches métier détaillées : <https://candidat.pole-emploi.fr/metierscope/metiers>

The screenshot shows the 'Liste des métiers' (List of Professions) page on the Pole emploi website. The page has a clean, modern design with a light gray background. On the left, there's a sidebar with an alphabetical navigation grid from A to Z. The main content area features a pink decorative banner with icons of gears and documents. Below the banner, a message encourages users to discover professions alphabetically. A horizontal menu bar at the top includes the French flag, the Pôle emploi logo, a 'MENU' icon, and a 'CONNEXION' button. Several profession names are listed in boxes with arrows pointing right, indicating they lead to detailed job descriptions.

Après une analyse de la construction du site (dans de la partie « Network » des outils de développeur du navigateur) cela devient facile de constater que le site récupère et affiche toutes ces informations suite à une requête API REST faite à l'« endpoint » suivant :  
<https://candidat.pole-emploi.fr/gw-metierscope/jobs/groupByFirstLetter>

Ce endpoint REST API ne semble pas être protégé par des clés dynamiques API, protections du type Basic Auth dans les entêtes, ou d'autre mécanismes. Il est possible d'accéder directement aux données sans contraintes.

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

The screenshot shows a search interface for jobs on the Pôle emploi.fr website. The interface includes a sidebar with letters A-Z, a main search area with three columns of job titles, and a detailed preview section below. Below the preview is a developer tools Network tab showing JSON data for the search results.

**Main Search Area:**

- Accessoiriste → Accompagnant / Accompagnante d'élèves en situation de handicap (AESH) → Accompagnateur médico-social / Accompagnatrice médicosociale**
- Acheteur / Acheteuse → Actuaire → Administrateur / Administratrice réseau, sécurité**
- Afficheur / Afficheuse → Affréteur / Affréteuse → Agent administratif / Agente administrative banque ou assurance**

**Developer Tools Network Tab (Preview Section):**

```

{
  "A": [
    {
      "romeCode": "L1503", "label": "Décor et accessoires spectacle", "mainName": "Accessoiriste", "romeCode": "K2104", "label": "Éducation et surveillance d'établissements d'enseignement", "mainName": "Accompagnant / Accompagnante d'élèves en situation de handicap (AESH)", "romeCode": "K1301", "label": "Accompagnateur médico-social / Accompagnatrice médicosociale"
    },
    {
      "romeCode": "L1503", "label": "Décor et accessoires spectacle", "mainName": "Accessoiriste", "romeCode": "M1101", "label": "Achats", "mainName": "Acheteur / Acheteuse"
    },
    {
      "romeCode": "L1503", "label": "Décor et accessoires spectacle", "mainName": "Accessoiriste", "romeCode": "M1801", "label": "Administration de systèmes d'information", "mainName": "Actuaire"
    },
    {
      "romeCode": "C1105", "label": "Etudes actuarielles en assurances", "mainName": "Administrateur / Administratrice réseau, sécurité", "romeCode": "I1201", "label": "Affréteur / Affréteuse"
    },
    {
      "romeCode": "C1401", "label": "Gestion et banque et assurance", "mainName": "Agent administratif / Agente administrative banque ou assurance", "romeCode": "N1201", "label": "Afficheur / Afficheuse"
    },
    {
      "romeCode": "M1601", "label": "Accueil et renseignements", "mainName": "Agent / Agente d'accueil", "romeCode": "K1205", "label": "Information sociale", "mainName": "Agente d'accueil social"
    },
    {
      "romeCode": "K2301", "label": "Distribution et assainissement d'eau", "mainName": "Agent / Agente d'assainissement", "romeCode": "H3302", "label": "Opérateur d'assemblage, tri ou emballage", "mainName": "Agent / Agente de conditionnement"
    },
    {
      "romeCode": "K1501", "label": "Application des règles financières publiques", "mainName": "Agent / Agente de constatation des douanes", "romeCode": "N4302", "label": "Contrôle des transports en commun", "mainName": "Agent contrôleur"
    },
    {
      "romeCode": "K2305", "label": "Salubrité et traitement de nuisibles", "mainName": "Agent / Agente de désinsectisation", "romeCode": "H2801", "label": "D'équipement de transformation du verre", "mainName": "Agent / Agente de fabrication du verre"
    },
    {
      "romeCode": "E1304", "label": "Façonnage et routage", "mainName": "Agent / Agente de façonnage et routage", "romeCode": "I1309", "label": "Maintenance électrique", "mainName": "Agent / Agente de maintenance électrique"
    }
  ]
}
  
```

**Developer Tools Network Tab (Detailed Preview Section):**

```

{
  "groupByFirstLetter": [
    {
      "romeCode": "G1801", "label": "Café, bar brasserie", "mainName": "Barman / Barmaid", "romeCode": "G1801", "label": "Café, bar brasserie", "mainName": "Barman / Barmaid"
    },
    {
      "romeCode": "B1603", "label": "Réalisation d'ouvrages en bijouterie, joaillerie et orfèvrerie", "mainName": "Bijoutier / Bijoutière", "romeCode": "B1603", "label": "Réalisation d'ouvrages en bijouterie, joaillerie et orfèvrerie", "mainName": "Bijoutier / Bijoutière"
    },
    {
      "romeCode": "I1201", "label": "Biologie médicale", "mainName": "Biologiste médical / médicale", "romeCode": "I1201", "label": "Biologie médicale", "mainName": "Biologiste médical / médicale"
    },
    {
      "romeCode": "H2201", "label": "Blanchisserie industrielle", "mainName": "Conduite d'équipement de fabrication de papier ou de carton", "romeCode": "H2201", "label": "Blanchisserie industrielle", "mainName": "Conduite d'équipement de fabrication de papier ou de carton"
    },
    {
      "romeCode": "H3101", "label": "Conduite d'équipement de fabrication de papier ou de carton", "mainName": "Bobinier / Bobinière", "romeCode": "H3101", "label": "Conduite d'équipement de fabrication de papier ou de carton", "mainName": "Bobinier / Bobinière"
    },
    {
      "romeCode": "H2601", "label": "Bobinage électrique", "mainName": "Boucherie", "romeCode": "H2601", "label": "Bobinage électrique", "mainName": "Boucherie"
    },
    {
      "romeCode": "D1101", "label": "Boucherie", "mainName": "Boucher / Bouchère", "romeCode": "D1101", "label": "Boucherie", "mainName": "Boucher / Bouchère"
    },
    {
      "romeCode": "H2101", "label": "Abattage et découpe des viandes", "mainName": "Boulangerie - viennoiserie", "romeCode": "H2101", "label": "Abattage et découpe des viandes", "mainName": "Boulangerie - viennoiserie"
    },
    {
      "romeCode": "D1102", "label": "Achat vente d'objets d'art, anciens ou d'occasion", "mainName": "Boulanger / Boulangère", "romeCode": "D1102", "label": "Achat vente d'objets d'art, anciens ou d'occasion", "mainName": "Boulanger / Boulangère"
    },
    {
      "romeCode": "H1804", "label": "Réalisation d'ouvrages d'art en fil", "mainName": "Brodeur / Brodeuse", "romeCode": "H1804", "label": "Réalisation d'ouvrages d'art en fil", "mainName": "Brodeur / Brodeuse"
    },
    {
      "romeCode": "A1201", "label": "Bûcheronnage et élagage", "mainName": "Bûcheronnage et élagage", "romeCode": "A1201", "label": "Bûcheronnage et élagage", "mainName": "Bûcheronnage et élagage"
    },
    {
      "romeCode": "H2602", "label": "Câblage électrique et électromécanique", "mainName": "Danseur / Danseuse", "romeCode": "H2602", "label": "Câblage électrique et électromécanique", "mainName": "Danseur / Danseuse"
    },
    {
      "romeCode": "L1201", "label": "Danse", "mainName": "Danseur / Danseuse", "romeCode": "L1201", "label": "Danse", "mainName": "Danseur / Danseuse"
    },
    {
      "romeCode": "H3401", "label": "Conduite de traitement d'abrasion de surface", "mainName": "Illustrateur / Illustratrice", "romeCode": "H3401", "label": "Conduite de traitement d'abrasion de surface", "mainName": "Illustrateur / Illustratrice"
    },
    {
      "romeCode": "A1204", "label": "Protection du patrimoine naturel", "mainName": "Kinésithérapeute", "romeCode": "A1204", "label": "Protection du patrimoine naturel", "mainName": "Kinésithérapeute"
    },
    {
      "romeCode": "B1604", "label": "Réparation - montage en systèmes horlogers", "mainName": "Kinésithérapeute", "romeCode": "B1604", "label": "Réparation - montage en systèmes horlogers", "mainName": "Kinésithérapeute"
    },
    {
      "romeCode": "B1101", "label": "Création en arts plastiques", "mainName": "Illustrateur / Illustratrice", "romeCode": "B1101", "label": "Création en arts plastiques", "mainName": "Illustrateur / Illustratrice"
    },
    {
      "romeCode": "A1203", "label": "Aménagement et entretien des espaces verts", "mainName": "Kinésithérapeute", "romeCode": "A1203", "label": "Aménagement et entretien des espaces verts", "mainName": "Kinésithérapeute"
    },
    {
      "romeCode": "J1404", "label": "Kinésithérapie", "mainName": "Kinésithérapeute", "romeCode": "J1404", "label": "Kinésithérapie", "mainName": "Kinésithérapeute"
    },
    {
      "romeCode": "K2202", "label": "Lavage de vitres", "mainName": "Laveur / Laveuse de vitres", "romeCode": "K2202", "label": "Lavage de vitres", "mainName": "Laveur / Laveuse de vitres"
    },
    {
      "romeCode": "H2203", "label": "Conduite d'installation de production de panneaux bois", "mainName": "Laveur / Laveuse de vitres", "romeCode": "H2203", "label": "Conduite d'installation de production de panneaux bois", "mainName": "Laveur / Laveuse de vitres"
    }
  ]
}
  
```

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

Nous pouvons voir qu'il s'agit des objets JSON qui peuvent alors être facilement récupérées, transformées (si besoin) et réutilisées pour notre projet.

## 2. Récupération des fiches (détail) des métiers

Une fois cette étape confirmée il faut analyser le comportement du site encore plus loin, par exemple lorsque l'on clique sur une fiche métier pour vérifier si le site utilise toujours la REST API pour obtenir les informations détaillées du métier sélectionné.

Pour tester il suffit d'avoir les outils de développeur ouverts depuis la liste puis de cliquer sur une fiche métier et analyser les requêtes faites par le navigateur dans la partie « Network ».

The screenshot shows the Pôle emploi website for the 'Accessoiriste' profession. The page layout includes a sidebar with links like 'Tout savoir sur ce métier', 'Compétences', 'Contextes de travail', 'Métiers proches', and 'Pour aller plus loin'. The main content area displays the title 'Décor et accessoires spectacle' and a summary of the profession's main tasks. At the bottom, the browser's developer tools Network tab is active, showing a list of network requests. The 'L1503' request is highlighted, and its response is displayed as a JSON object:

```
▼ {romeCode: "L1503", label: "Décor et accessoires spectacle", mainName: "Accessoiriste", accessText: "Ce métier est accessible avec un diplôme de niveau CAP à Bac+2 (BTS, ...)", appellations: [{code: "10203", label: "Accessoiriste"}, {code: "10204", label: "Accessoiriste"}], description: "Conçoit et effectue la mise en place des décors et des accessoires pour les spectacles, des tournages, selon les intentions artistiques du metteur en scène, du réalisateur, du chorégraphe, ..., la réglementation sécurité des Etablissements Recevant du Public (ERP) et les impératifs de la production.", domain: {label: "Audiovisuel et spectacle", labelUrl: "audiovisuel-et-spectacle", ...}, label: "Décor et accessoires spectacle", mainName: "Accessoiriste", romeCode: "L1503", videoUrl: "https://www.youtube.com/embed/se9QDPR2PWhw"}
```

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

Nous avons accédé au la page suivante :

<https://candidat.pole-emploi.fr/metierscope/fiche-metier/L1503/accessoiriste>

On peut voir que l'ensemble des informations présentes sur les fiches sont également récupérées depuis une API REST.

Pour notre exemple l'appel fait par le navigateur est une requête GET n'ayant aucune authentification (Basic Auth ou autre) ni vérification (CORS...) qui peut donc être librement utilisée :

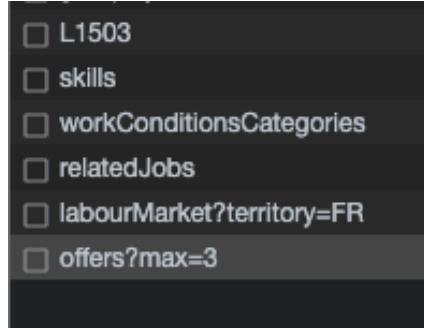
<https://candidat.pole-emploi.fr/gw-metierscope/job/L1503>

Nous pouvons voir ainsi que le code « L1503 » est tout simplement l'identifiant du métier (qu'on a vu dans la liste des métiers dans la requête précédente)

Ainsi, pour pouvoir accéder à la fiche (détail) d'un métier dans le but de récupérer l'ensemble des informations utiles de chaque métier sous format JSON, il faut utiliser l'identifiant du métier nommé « romeCode » récupéré à partir JSON comportant la liste des métiers et faire un autre appel GET en respectant ce format :

<https://candidat.pole-emploi.fr/gw-metierscope/job/:romeCode>

Nous pouvons voir en revanche que ce n'est pas tout. Le site fait des appels similaires pour accéder à plusieurs sections du site qui respectent le même format :



<https://candidat.pole-emploi.fr/gw-metierscope/job/L1503/skills>

<https://candidat.pole-emploi.fr/gw-metierscope/job/L1503/workConditionsCategories>

<https://candidat.pole-emploi.fr/gw-metierscope/job/L1503/relatedJobs>

<https://candidat.pole-emploi.fr/gw-metierscope/job/L1503/labourMarket?territory=FR>

<https://candidat.pole-emploi.fr/gw-metierscope/job/L1503/offers?max=3>

Ces endpoints enrichissent la fiche métier des informations complémentaires comme :

- Les compétences recherchées
- Les conditions de travail
- Les emplois similaires
- Le marché du travail pour ce métier
- Offres d'emploi
- Vidéo de présentation du métier
- ...

Ainsi, il s'est avéré très facile de récupérer l'ensemble des informations en réalisant un script en Python qui récupère la liste JSON, parcours cette liste JSON et accède à chaque fiche métier (les 6 endpoints GET de l'API REST) pour obtenir l'ensemble des informations de tous les métiers.

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

### 3. Réalisation d'un scraper de l'API Pôle emploi

Ayant toutes ces informations j'ai pu réaliser de zero un API scraper du site de Pôle emploi :

```
# API Scraping de Pôle emploi
import requests
import json
import time

# Récupération de la liste des métiers au format JSON depuis l'API de Pôle Emploi
response = requests.get('https://candidat.pole-emploi.fr/gw-metierscope/jobs/groupByFirstLetter')
metiers = {} # Dictionnaire pour stocker les métiers
i = 0 # Compteur pour suivre le nombre de métiers traités

# Conversion de la réponse en JSON
response_json = response.json()

# Calcul du nombre total de métiers
total_métiers = 0
for key, items in response_json.items():
    total_métiers += len(items)

# Affichage du nombre total de métiers
print("Nombre de métiers: " + str(total_métiers))

# Parcours et traitement des métiers
for key, items in response_json.items():
    for métier in items:
        # Récupération des détails du métier
        métier['métier'] = requests.get('https://candidat.pole-emploi.fr/gw-metierscope/job/' + métier['romeCode']).json()
        # Récupération des compétences associées au métier
        métier['skills'] = requests.get('https://candidat.pole-emploi.fr/gw-metierscope/job/' + métier['romeCode'] + '/skills').json()
        # Récupération des conditions de travail associées au métier
        métier['workConditionsCategories'] = requests.get('https://candidat.pole-emploi.fr/gw-metierscope/job/' + métier['romeCode'] + '/workConditionsCategories').json()

        # Stockage du métier dans le dictionnaire
        metiers[métier['romeCode']] = métier

        # Mise à jour et affichage du compteur
        i += 1
        print(str(i) + "/" + str(total_métiers))
        print(str(round(i/total_métiers*100, 2)) + '%')
        print(métier['label'])
        print('-----')

        # Pause pour éviter de surcharger l'API
        time.sleep(0.3)

# Sauvegarde des métiers dans un fichier JSON
with open('dataset/metiers-offres.json', 'w') as outfile:
    json.dump(metiers, outfile)
print('THE END')
```

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

```
# API Scraping de Pôle emploi
import requests
import json
import time

# Pôle emploi - API Scraping

# Récupération de la liste des métiers au format JSON depuis l'API de Pôle Emploi
response = requests.get('https://candidat.pole-emploi.fr/gw-metierscope/jobs/groupByFirstLetter')
metiers = {} # Dictionnaire pour stocker les métiers
i = 0 # Compteur pour suivre le nombre de métiers traités

# Conversion de la réponse en JSON
response_json = response.json()

# Calcul du nombre total de métiers
total_métiers = 0
for key, items in response_json.items():
    total_métiers += len(items)

# Affichage du nombre total de métiers
print("Nombre de métiers: " + str(total_métiers))

# Parcours et traitement des métiers
for key, items in response_json.items():
    for métier in items:
        # Récupération des détails du métier
        métier['metier'] = requests.get('https://candidat.pole-emploi.fr/gw-metierscope/job/' + métier['romeCode'])
        # Récupération des compétences associées au métier
        métier['skills'] = requests.get('https://candidat.pole-emploi.fr/gw-metierscope/job/' + métier['romeCode'])
        # Récupération des conditions de travail associées au métier
        métier['workConditionsCategories'] = requests.get('https://candidat.pole-emploi.fr/gw-metierscope/job/' + métier['romeCode'])

        # Stockage du métier dans le dictionnaire
        metiers[métier['romeCode']] = métier

        # Mise à jour et affichage du compteur
        i += 1
        print(str(i) + "/" + str(total_métiers))
        print(str(round(i/total_métiers*100, 2)) + '%')
        print(métier['label'])
        print('-----')

        # Pause pour éviter de surcharger l'API
        time.sleep(0.3)

# Sauvegarde des métiers dans un fichier JSON
with open('dataset/metiers-offres.json', 'w') as outfile:
    json.dump(metiers, outfile)

print('THE END')
```

Terminal	Local
sh-3.2# python scraper-dataset.py	
Nombre de métiers: 532	
1/532	
0.19%	
Décor et accessoires spéciaux	
2/532	
0.38%	
Éducation et surveillance	
3/532	
0.56%	
Accompagnement médico-social	
4/532	
0.75%	
Achats	
5/532	
0.94%	
Études actuarielles en assurance	
6/532	
1.13%	
Administration de système	
7/532	
1.32%	
Entretien d'affichage et d'éclairage	
8/532	
1.5%	
Affrètement transport	
9/532	
1.69%	
Gestion en banque et assurance	
10/532	
1.88%	
Accueil et renseignement	
11/532	
2.07%	
Information sociale	
12/532	
2.26%	
Promotion d'artistes et culture	
13/532	
2.44%	

J'ai réalisé de la même manière un autre scraper API pour obtenir dans un autre fichier la liste des 150 offres d'emploi les plus récentes pour chaque métier dans le cas où les informations des métiers récupérées initialement ne seraient pas suffisantes et qu'il faudrait extraire en plus des données à partir d'une grande base des offres d'emploi.

Le scraper récupère l'ensemble des informations JSON et les stocke dans une collection JSON comportant l'ensemble des métiers. Chaque objet « métier » contient toutes les informations détaillées récupérées des différents appels API. Toutes ces données sont ensuite sauvegardées dans le fichier `metiers.json`.

Le fichier final JSON est considéré comme un dataset (et en quelque sorte une base de données) qui sera ensuite utilisé par le projet final.

### 3. Analyse et exploitation du fichier JSON

Une fois le dataset sauvegardé il était nécessaire d'analyser en détail les données recoltées afin de trouver la manière de les exploiter.

Voici l'aperçu du fichier JSON comportant la liste des metiers ainsi que le détail de chaque métier dans l'IDE IntelliJ IDEA :

```
metiers-simple.json ×
1  {
2    "M1806": {
3      "label": "Conseil et maîtrise d'ouvrage en systèmes d'information",
4      "labourMarket": {
5        "jobOffers": 64060,
6        "jobSeekers": 10040
7      },
8      "mainName": "Consultant / Consultante informatique, Product Owner",
9      "metier": {"accessText": "Le métier est accessible avec un Master (M1, Master professionnel, diplôme d'ingénieur, 141 > relatedJobs": [...], 163 romeCode": "M1806", 537 skills": [...]}, 538 > "workConditionsCategories": [ 549 >   {"label": "CONDITIONS_TRAVAIL"}, 549 >   {"label": "HORAIRE_ET_DUREE_TRAVAIL"} 560 ]
561 },
562 "M1805": {
563   "label": "Études et développement informatique",
564   "labourMarket": {
565     "jobOffers": 224810,
566     "jobSeekers": 48070
567   },
568   "mainName": "Développeur / Développeuse", 569 >   "metier": {"accessText": "Ce métier est accessible avec un diplôme de niveau Bac+2 (BTS, DUT) à Master (MIAGE, dip 937 > relatedJobs": [...], 959 romeCode": "M1805", 960 skills": {
961 >   "domainsSkills": [...],
961 >   "knowledgeCategories": [...],
961 >   "professionalSoftSkills": [...]
961 > },
962   "workConditionsCategories": [...]
1531 > },
1532 },
1533 "M1807": {
1534   "label": "Exploitation de systèmes de communication et de commandement",
1535   "labourMarket": {
1536     "jobOffers": 2310,
1537     "jobSeekers": 250
1538 },
1539   "mainName": "Opérateur / Opératrice télécom aux armées",
1540   "metier": {"accessText": "Ce métier est accessible avec un CAP/BEP en électronique.\nUne formation militaire et de 1541 > relatedJobs": [...], 1542 romeCode": "M1807", 1543 skills": [...]}, 1544 > "workConditionsCategories": [...]
1545 },
1546 "M1808": {
1547   "label": "Information géographique",
1548   "labourMarket": {
1549     "jobOffers": 3110,
1550     "jobSeekers": 1130
1551 },
1552   "mainName": "Cartographe",
1553   "metier": {"accessText": "Ce métier est accessible avec un diplôme de niveau Bac+2 (BTS, DUT) ou Bac+3 (Licence, L 2187 > relatedJobs": [...], 2209 romeCode": "M1808", 2210 skills": [...]},
2210 }
```

A partir de ces données il serait ensuite nécessaire de qualifier les données potentiellement pertinentes pour l'usage du chatbot qu'on pourrait utiliser sous forme de patterns.

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

```
562 "M1805": {
563   "label": "Etudes et développement informatique",
564   "labourMarket": {
565     "jobOffers": 224810,
566     "jobSeekers": 48070
567   },
568   "mainName": "Développeur / Développeuse",
569   "metier": {
570     "accessText": "Ce métier est accessible avec un diplôme de niveau Bac+2 (BTS, DUT) à Master (MIAGE, diplôme d'ingénieur, Master professionn",
571     "appellations": [
572       {"code": "10933", "label": "Analyste cogniticien / cogniticienne informatique"},
573       {"code": "10935", "label": "Analyste d'application"},
574       {"code": "10944", "label": "Analyste développeur / développeuse"},
575       {"code": "10950", "label": "Analyste fonctionnel / fonctionnelle informatique"},
576       {"code": "10952", "label": "Analyste organique informatique"},
577       {"code": "10956", "label": "Analyste réseau informatique"},
578       {"code": "10957", "label": "Analyste responsable d'application informatique"},
579       {"code": "10959", "label": "Analyste télématicaie"},
580       {"code": "10960", "label": "Analyste programmeur / Analyste-programmeuse de micro-informatique"},
581       {"code": "10961", "label": "Analyste-programmeur / Analyste-programmeuse d'étude informatique"},
582       {"code": "10962", "label": "Analyste-programmeur / Analyste-programmeuse en informatique industrielle"},
583       {"code": "10963", "label": "Analyste-programmeur / Analyste-programmeuse gestion informatique"},
584       {"code": "10964", "label": "Analyste-programmeur / Analyste-programmeuse informatique"},
585       {"code": "12168", "label": "Chef de groupe logiciel informaticien"},
586       {"code": "12243", "label": "Chef de projet étude et développement informatique"},
587       {"code": "12814", "label": "Concepteur / Conceptrice d'application informatique"},
588       {"code": "12828", "label": "Concepteur / Conceptrice informatique"},
589       {"code": "12829", "label": "Concepteur / Conceptrice logiciel informatique"},
590       {"code": "14148", "label": "Développeur / Développeuse d'application"},
591       {"code": "14153", "label": "Développeur / Développeuse informatique"},
592       {"code": "14155", "label": "Développeur / Développeuse multimédia"},
593       {"code": "14156", "label": "Développeur / Développeuse web"},
594       {"code": "14167", "label": "Didacticien / Didacticienne informatique"},
595       {"code": "15753", "label": "Ingénieur / Ingénierie en développement d'applications"},
596       {"code": "15627", "label": "Ingénieur / Ingénierie d'application informatique"},
597       {"code": "15631", "label": "Ingénieur / Ingénierie de conception informatique"},
598       {"code": "15696", "label": "Ingénieur / Ingénierie d'étude en applications scientifiques informatiques"},
599       {"code": "15698", "label": "Ingénieur / Ingénierie d'étude et développement informatique"},
600       {"code": "15700", "label": "Ingénieur / Ingénierie d'étude logiciel informatique"},
601       {"code": "15714", "label": "Ingénieur / Ingénierie développement logiciel informatique"},
602       {"code": "15824", "label": "Ingénieur / Ingénierie informatique développement en temps réel"},
603       {"code": "15568", "label": "Informaticien / Informaticienne d'application"},
604       {"code": "15569", "label": "Informaticien / Informaticienne de développement"},
605       {"code": "15580", "label": "Ingénieur / Ingénierie analyste en système d'information"},
606       {"code": "15581", "label": "Ingénieur / Ingénierie analyse informatique"},
607       {"code": "15826", "label": "Ingénieur / Ingénierie logiciel informatique"},
608       {"code": "15867", "label": "Ingénieur / Ingénierie programme informatique"},
609       {"code": "17558", "label": "Paramétrateur / Paramétreuse logiciel ERP"},
610       {"code": "18192", "label": "Programmeur / Programmeuse informatique"},
611       {"code": "18193", "label": "Programmeur / Programmeuse logiciel de base informatique"},
612       {"code": "18927", "label": "Responsable des développements informatiques"},
613       {"code": "18961", "label": "Responsable d'étude informatique"},
614       {"code": "18187", "label": "Programmeur / Programmeuse d'applications"},
615       {"code": "18189", "label": "Programmeur / Programmeuse d'études"},
616       {"code": "18191", "label": "Programmeur industriel / Programmeuse industrielle"},
617       {"code": "18625", "label": "Responsable d'application informatique"},
618       {"code": "18637", "label": "Responsable d'atelier de génie logiciel"},
619       {"code": "18719", "label": "Responsable de gestion de configuration"},
```

Cela pourrait être la liste des connaissances ou les domaines d'expertise du métier concerné, par exemple :

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

```
1047  [
1048    "knowledgeCategories": [
1049      {
1050        "knowledges": [
1051          {"label": "Ada"}, {"label": "Algorithmique"}, {"label": "Apache"}, {"label": "ASP.NET"}, {"label": "AutoItScript"}, {"label": "Basic"}, {"label": "C#"}, {"label": "C/C++"}, {"label": "Cobol"}, {"label": "CSS"}, {"label": "Delphi"}, {"label": "Fortran"}, {"label": "HTML"}, {"label": "Java"}, {"label": "J2EE"}, {"label": "J2ME"}, {"label": "Langages de programmation informatique"}, {"label": "Lisp"}, {"label": "Pascal"}, {"label": "Perl"}, {"label": "PHP"}, {"label": "Prolog"}, {"label": "Python"}, {"label": "Ruby"}, {"label": "Script Shell"}, {"label": "SQL"}, {"label": "VB.NET"}, {"label": "Visual Basic"}, {"label": "WLanguage"}, {"label": "XML"}, {"label": "Application web"}, {"label": "Architecture propriétaire"}, {"label": "Architecture web"}, {"label": "Business Intelligence (BI) / Informatique décisionnelle"}, {"label": "Informatique de gestion"}, {"label": "Informatique scientifique et technique"}, {"label": "Progiciels de gestion intégrée d'entreprise (ERP)"}, {"label": "Programmation logicielle"}, {"label": "Publication Assistée par Ordinateur (PAO)"}, {"label": "Système embarqué"}, {"label": "Systèmes d'exploitation informatique"}, {"label": "Comptabilité générale"}, {"label": "Finance"}, {"label": "Gestion commerciale, relation clients"}, {"label": "Gestion de configuration"}, {"label": "Informatique industrielle"}, {"label": "Technologies de l'accessibilité numérique"}, {"label": "Anglais technique"}
1098  ],
1099  "label": "Domaines d'expertise"
1100 }]
```

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

```
1081      "label": "Architecture propriétaire",
1082      {"label": "Architecture web"},  
1083      {"label": "Business Intelligence (BI) / Informatique décisionnelle"},  
1084      {"label": "Informatique de gestion"},  
1085      {"label": "Informatique scientifique et technique"},  
1086      {"label": "Progiciels de gestion intégrée d'entreprise (ERP)"},  
1087      {"label": "Programmation logicielle"},  
1088      {"label": "Publication Assistée par Ordinateur (PAO)"},  
1089      {"label": "Système embarqué"},  
1090      {"label": "Systèmes d'exploitation informatique"},  
1091      {"label": "Comptabilité générale"},  
1092      {"label": "Finance"},  
1093      {"label": "Gestion commerciale, relation clients"},  
1094      {"label": "Gestion de configuration"},  
1095      {"label": "Informatique industrielle"},  
1096      {"label": "Technologies de l'accessibilité numérique"},  
1097      {"label": "Anglais technique"}  
1098    ],  
1099    "label": "Domaines d'expertise"  
1100  },  
1101  {  
1102    "knowledges": [  
1103      {"label": "Animation d'instances agiles (mélée, démonstration, rétrospective, ...)"},  
1104      {"label": "Formalisation de processus de gestion"},  
1105      {"label": "Marketing / Mercatique"},  
1106      {"label": "Méthode AGILE"},  
1107      {"label": "Méthodologie SAFe"},  
1108      {"label": "Méthodologie Scrum"},  
1109      {"label": "Modélisation informatique"},  
1110      {"label": "Principes d'intégration de matériels et de logiciels"},  
1111      {"label": "Procédures d'appels d'offres"},  
1112      {"label": "Protocoles et normes télécoms"},  
1113      {"label": "Urbanisation des systèmes d'information"},  
1114      {"label": "Règles de sécurité Informatique et Télécoms"},  
1115      {"label": "Système KANBAN"},  
1116      {"label": "Système LEAN"}  
1117    ],  
1118    "label": "Normes et procédés"  
1119  },  
1120  {  
1121    "knowledges": [  
1122      {"label": "Matériel de contrôle embarqué"},  
1123      {"label": "Réseaux informatiques et télécoms"},  
1124      {"label": "Produits multimédias"}  
1125    ],  
1126    "label": "Produits, outils et matières"  
1127  }  
1128 ],  
1129  "professionalSoftSkills": [  
1130    {"label": "Travailler en équipe"},  
1131    {"label": "Faire preuve d'autonomie"},  
1132    {"label": "Faire preuve de rigueur et de précision"},  
1133    {"label": "Faire preuve de créativité, d'inventivité"}  
1134  ]  
1135 },  
1136  "workConditionsCategories": [  
1137    {  
1138      "label": "CONDITIONS_TRAVAIL",  
1139      "label": "Sous-conditions": [
```

En regardant le fichier JSON nous pouvons constater qu'il y a de très nombreuses informations extrêmement utiles qu'on pourrait exploiter dans notre chatbot que ce soit pour tenter d'appliquer du pattern matching, tout comme pour afficher des informations détaillées sur chaque métier.

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

```
695 |     "romeCode": "M1805",
696 |     "skills": [
697 |       "domainsSkills": [
698 |         {
699 |           "label": "Communication, Création, Innovation, Nouvelles technologies",
700 |           "score": 49,
701 |           "stakes": [
702 |             {
703 |               "code": "8",
704 |               "hardSkills": [
705 |                 {
706 |                   "label": "Communiquer à l'oral en langue étrangère"
707 |                 },
708 |                 {
709 |                   "label": "Communiquer à l'écrit en langue étrangère"
710 |                 },
711 |                 {
712 |                   "label": "Communiquer auprès de ses clients internes et externes"
713 |                 }
714 |               ],
715 |               "label": "Communication, Multimédia"
716 |             },
717 |             {
718 |               "code": "10",
719 |               "hardSkills": [
720 |                 {
721 |                   "label": "Structurer, synthétiser des informations"
722 |                 },
723 |                 {
724 |                   "label": "Analyser, exploiter, structurer des données"
725 |                 },
726 |                 {
727 |                   "label": "Créer une documentation technique"
728 |                 },
729 |                 {
730 |                   "label": "Rédiger un cahier des charges, des spécifications techniques"
731 |                 }
732 |               ],
733 |               "label": "Traitement de l'information et des données"
734 |             },
735 |             {
736 |               "code": "11",
737 |               "hardSkills": [
```

Les conditions de travail (télétravail...) pourraient être également utiles pour le chatbot (par exemple pour répondre à la saisie “je cherche un travail acceptant le télétravail” avec la liste des métiers concernés) :

```
666 |     "title": "Métiers de l'informatique, internet et téléco
667 |   },
668 |   "label": "Études et développement informatique",
669 |   "mainName": "Développeur / Développeuse",
670 |   "romeCode": "M1805",
671 |   "videoUrl": "https://www.youtube.com/embed/RRDSL80dVrU"
672 | },
673 | >   "relatedJobs": [...],
695 |   "romeCode": "M1805",
696 |   "skills": [...],
1274 |   "workConditionsCategories": [
1275 |     {
1276 |       "label": "CONDITIONS_TRAVAIL",
1277 |       "workConditions": [
1278 |         {
1279 |           "label": "Déplacements professionnels"
1280 |         },
1281 |         {
1282 |           "label": "Possibilité de télétravail"
1283 |         }
1284 |       ]
1285 |     },
1286 |     {
1287 |       "label": "HORAIRE_ET_DUREE_TRAVAIL",
1288 |       "workConditions": [
1289 |         {
1290 |           "label": "Travail en astreinte"
1291 |         },
1292 |         {
1293 |           "label": "Travail le week-end"
1294 |         }
1295 |       ]
1296 |     }
1297 |   ],
1298 | }
```

## 4. Réalisation d'un chatbot

En utilisant ces données JSON j'ai d'abord réalisé un chatbot simple en python en utilisant Spacy qui détectait les mots et tentait de correspondre aux métiers :

```
import spacy
from spacy.matcher import Matcher
from spacy import displacy
import json

from spacy.tokens.span import Span


class SuggestionMetier:
    def __init__(self):
        self.metiers = self.load_metiers()
        self.nlp = spacy.load("fr_core_news_sm")

    # Récupérer la liste de compétences
    pattern_list = self.get_pattern_list()

    # Créer le Matcher enrichi avec la liste de compétences
    self.matcher = self.create_skill_matcher(pattern_list)

    def load_metiers(self):
        with open('dataset/metiers.json', 'r') as json_file:
            return json.load(json_file)

    def get_pattern_list(self):
        pattern_list = []

        for code, data in self.metiers.items():
            # Ensure data exists and is a dictionary
            if not data or not isinstance(data, dict):
                continue

            skills = data.get("skills", {})

            # mainName
            mainName = data.get("mainName")
            if mainName:
                pattern_list.append((mainName, code))

            # appellations
            appellations = data.get("metier", {}).get("appellations", [])
            for appellation in appellations:
                label = appellation.get("label")
                if label:
                    pattern_list.append((label, code))

            # workConditionsCategories
            workConditionsCategories = data.get("workConditionsCategories", [])
            for work_condition_category in workConditionsCategories:
                for work_condition in work_condition_category.get("workConditions", []):
                    label = work_condition.get("label")
                    if label:
                        pattern_list.append((label, code))

            # Skills from knowledgeCategories
            knowledgeCategories = skills.get("knowledgeCategories", [])
            for category in knowledgeCategories:
                for k in category.get('knowledges', []):
                    label = k.get('label')
                    if label:
                        pattern_list.append((label, code))

            # Skills from professionalSoftSkills
            professionalSoftSkills = skills.get("professionalSoftSkills", [])
            for skill in professionalSoftSkills:
                label = skill.get('label')
                if label and label != "":
                    pattern_list.append((label, code))

    # Convert the list to a set to remove duplicates, then back to a list
    pattern_list = list(set(pattern_list))
```

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

```
return pattern_list

def create_skill_matcher(self, pattern_list):
    matcher = Matcher(self.nlp.vocab)

    for skill, metier_code in pattern_list:
        # Diviser la compétence en mots individuels
        doc = self.nlp(skill.lower())
        pattern = [{"LEMMA": token.lemma_} for token in doc]
        # Ici, nous utilisons le metier_code comme identifiant pour chaque motif
        matcher.add(metier_code, [pattern])

    return matcher

def getSuggestions(self, texte, visualiser=False):
    doc = self.nlp(texte)
    matches = self.matcher(doc)
    print(f"\nTexte: {texte}")

    for match_id, start, end in matches:
        rule_id = self.nlp.vocab.strings[match_id] # Convertir match_id en chaîne
        span = doc[start:end]
        doc.ents = list(doc.ents) + [span] # add span to doc.ents
        print(f"Compétence détectée: {span.text}, Label: {rule_id}")

    if visualiser:
        displacy.serve(doc, style='ent')

    return [
        {
            "code": self.nlp.vocab.strings[match_id],
            "label": self.metiers[self.nlp.vocab.strings[match_id]]["mainName"],
            "detail": self.metiers[self.nlp.vocab.strings[match_id]]
        } for match_id, _, _ in matches
    ]

def sugg[red]er_message(self, suggestions):
    if not suggestions:
        return "Aucune suggestion de métier pour l'instant."
    else:
        metiers_suggested = ", ".join([s["label"] for s in suggestions])
        return "Vous pourriez envisager les métiers suivants : " + metiers_suggested + " (code: " + ", ".join([s["code"] for s in suggestions]) + ")"

if __name__ == '__main__':
    chat = SuggestionMetier()

    while True:
        texte = input("Entrez votre texte: ")
        suggestions = chat.getSuggestions(texte)
        print(chat.sugg[red]er_message(suggestions))
```

```
2023-08-19 13:48:55+02:00 [TensorFlowCompetenceExtractor/0] [INFO] WARNING: Could not find TensorRT
Entrez votre texte: informatique

Texte: informatique
Compétence détectée: Informatique, Label: H1209
Compétence détectée: Informatique, Label: K2402
Compétence détectée: Informatique, Label: I1401
Compétence détectée: Informatique, Label: E1108
Compétence détectée: Informatique, Label: I1305
Compétence détectée: Informatique, Label: H1208
Compétence détectée: Informatique, Label: K2108
Compétence détectée: Informatique, Label: I1307
Compétence détectée: Informatique, Label: K2107
Compétence détectée: Informatique, Label: F1107
Vous pourriez envisager les métiers suivants : Technicien / Technicienne études électroniques, Ingénieur / Ingénierie de recherche scientifique, Technicien / Technicienne de maintenance en informatique, Traducteur / Traductrice, Électronicien / Électronicienne de maintenance, Automaticien / Automaticienne en bureau d'études, Professeur / Professeure des universités, Installateur / Installatrice fibre optique, Professeur / Professeure de collège et de lycée, Géomètre (code: H1209, K2402, I1401, E1108, I1305, H1208, K2108, I1307, K2107, F1107)
Entrez votre texte: vendeur

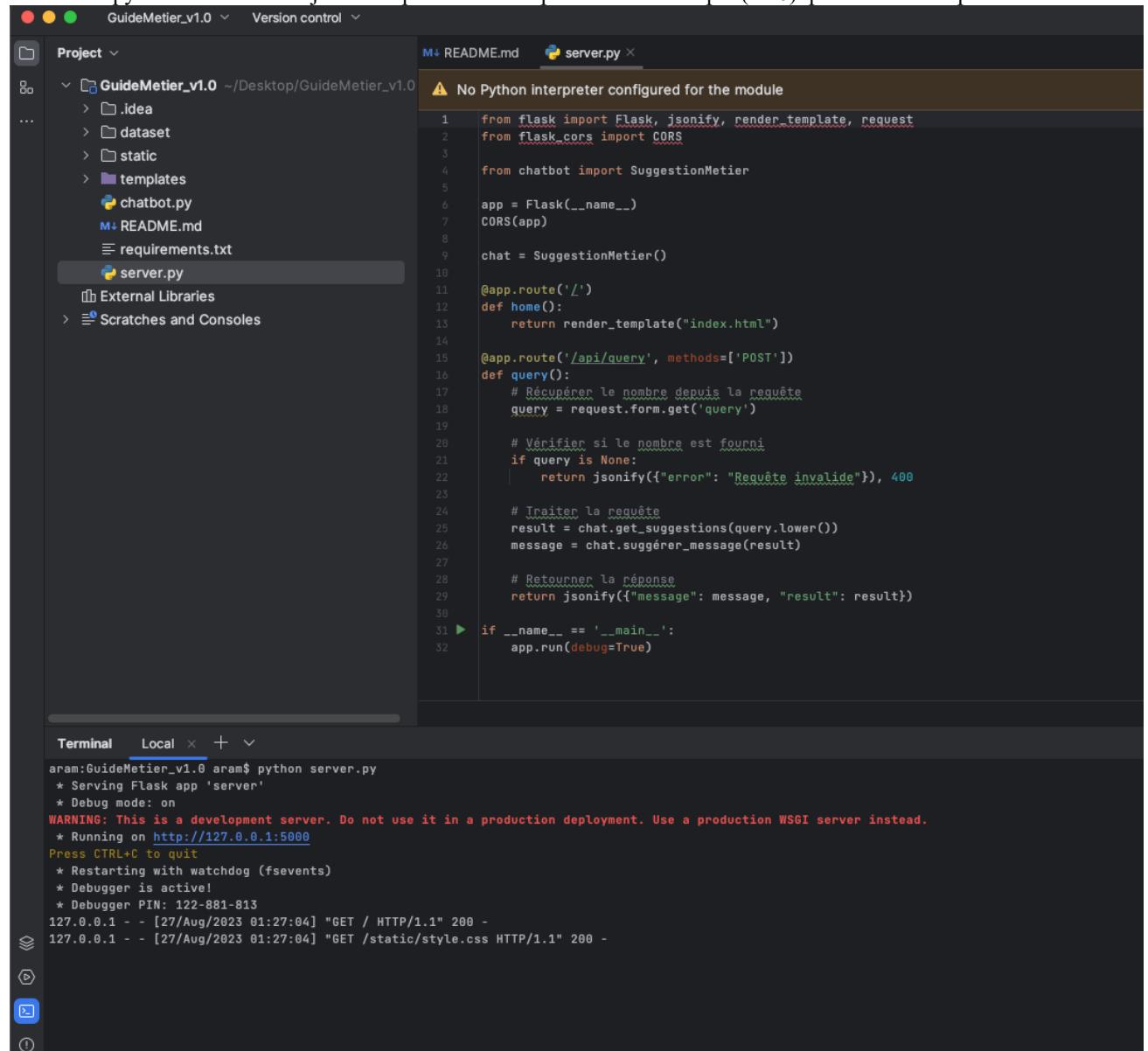
Texte: vendeur
Aucune suggestion de métier pour l'instant.
Entrez votre texte: acheteur

Texte: acheteur
Aucune suggestion de métier pour l'instant.
Entrez votre texte:
```

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

J'ai ensuite réalisé une interface web v1.0 lié au chatbot grâce au framework Flask. Pour communiquer entre le chatbot en python et le site web j'ai dû implémenter une première API simple (v1.0) qui était utilisée par le site :



The screenshot shows the PyCharm IDE interface. The project tree on the left lists files and folders: .idea, dataset, static, templates, chatbot.py, README.md, requirements.txt, and server.py (which is selected). The main editor window shows the 'server.py' code:

```
from flask import Flask, jsonify, render_template, request
from flask_cors import CORS
from chatbot import SuggestionMetier

app = Flask(__name__)
CORS(app)

chat = SuggestionMetier()

@app.route('/')
def home():
    return render_template("index.html")

@app.route('/api/query', methods=['POST'])
def query():
    # Récupérer le nom depuis la requête
    query = request.form.get('query')

    # Vérifier si le nom est fourni
    if query is None:
        return jsonify({"error": "Requête invalide"}), 400

    # Traiter la requête
    result = chat.get_suggestions(query.lower())
    message = chat.suggérer_message(result)

    # Retourner la réponse
    return jsonify({"message": message, "result": result})

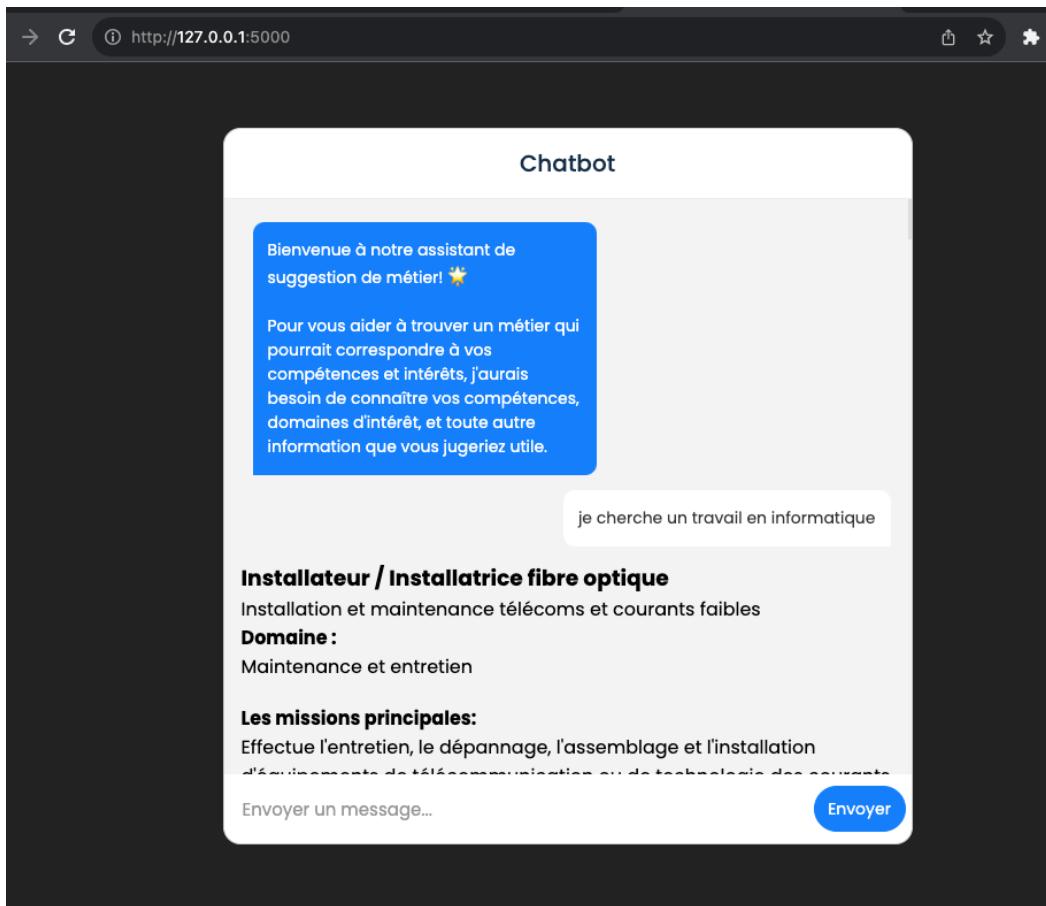
if __name__ == '__main__':
    app.run(debug=True)
```

The terminal at the bottom shows the command `python server.py` being run and the Flask development server starting:

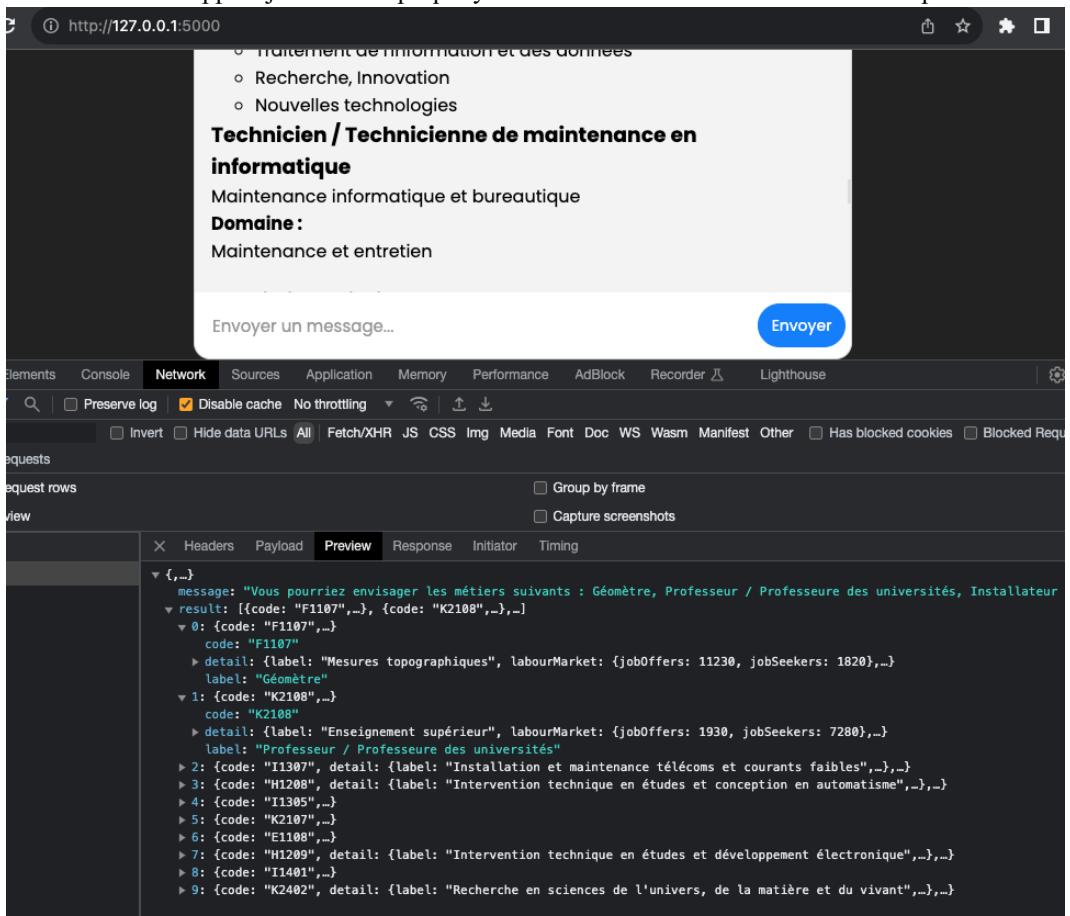
```
aram:GuideMetier_v1.0 aram$ python server.py
* Serving Flask app 'server'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with watchdog (fsevents)
* Debugger is active!
* Debugger PIN: 122-881-813
127.0.0.1 - - [27/Aug/2023 01:27:04] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [27/Aug/2023 01:27:04] "GET /static/style.css HTTP/1.1" 200 -
```

L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4



Le site faisait un appel ajax POST /api/query avec la saisie utilisateur et affichait la réponse du serveur :



## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

Tandis que côté serveur cela donnait ceci :

```
aram:GuideMetier_v1.0 aram$ python server.py
* Serving Flask app 'server'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with watchdog (fsevents)
* Debugger is active!
* Debugger PIN: 122-881-813
127.0.0.1 - - [27/Aug/2023 01:27:04] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [27/Aug/2023 01:27:04] "GET /static/style.css HTTP/1.1" 200 -

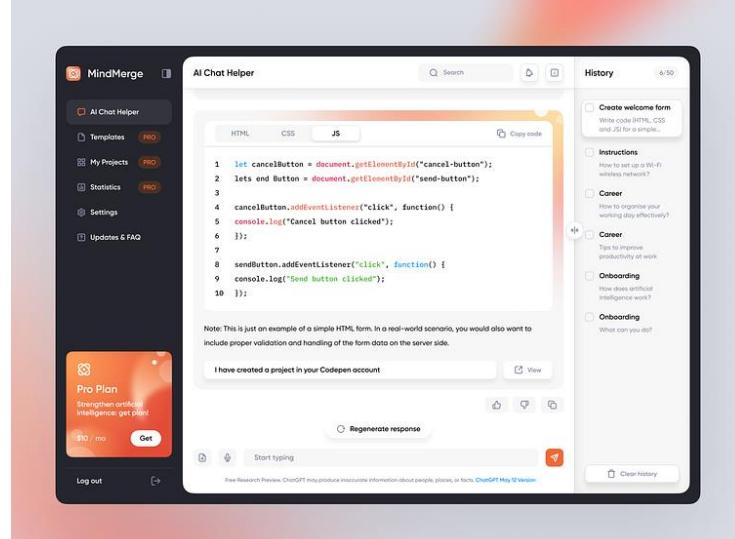
Texte: informatique
Compétence détectée: informatique, Label: F1107
Compétence détectée: informatique, Label: K2108
Compétence détectée: informatique, Label: I1307
Compétence détectée: informatique, Label: H1208
Compétence détectée: informatique, Label: I1305
Compétence détectée: informatique, Label: K2107
Compétence détectée: informatique, Label: E1108
Compétence détectée: informatique, Label: H1209
Compétence détectée: informatique, Label: I1401
Compétence détectée: informatique, Label: K2402
127.0.0.1 - - [27/Aug/2023 01:27:25] "POST /api/query HTTP/1.1" 200 -
127.0.0.1 - - [27/Aug/2023 01:27:34] "GET /static/style.css HTTP/1.1" 304 -

Texte: informatique
```

```
Texte: je cherche un travail en informatique
Compétence détectée: informatique, Label: F1107
Compétence détectée: informatique, Label: K2108
Compétence détectée: informatique, Label: I1307
Compétence détectée: informatique, Label: H1208
Compétence détectée: informatique, Label: I1305
Compétence détectée: informatique, Label: K2107
Compétence détectée: informatique, Label: E1108
Compétence détectée: informatique, Label: H1209
Compétence détectée: informatique, Label: I1401
Compétence détectée: informatique, Label: K2402
127.0.0.1 - - [27/Aug/2023 01:33:23] "POST /api/query HTTP/1.1" 200 -
```

## 5. Implémentation d'une nouvelle interface graphique

Après plusieurs tests je me suis rendu compte que l'interface n'était pas suffisamment grande, et donc non optimale pour un affichage d'une grande quantité de données, alors que le nombre de données possibles à exploiter était majeur. Il fallait remédier à cela, j'ai ainsi décidé d'implémenter une interface web en m'inspirant d'une maquette trouvée sur le site de Dribbble : <https://dribbble.com/shots/22026872-Chatbot-Web-UI> :



## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

The screenshot shows the GuideMetier website's chatbot interface. On the left, there's a sidebar with a logo of a smiling robot head, the text "GuideMetier" and "Rencontrez votre futur métier", and three buttons: "Chatbot", "Liste des métiers", and "Centres d'intérêt". The main area is titled "Chatbot" and contains a message from the bot: "Bienvenue à notre assistant de suggestion de métier! ☀️". It asks for user input: "Pour vous aider à trouver un métier qui pourra correspondre à vos compétences et intérêts, j'aurais besoin de connaître vos compétences, domaines d'intérêt, et toute autre information que vous jugeriez utile." Below this, it says "Si vous souhaitez obtenir quelques exemples de saisie, saisissez 'aide'." A "télétravail" button is visible in the top right corner. A large central box lists "Voici les métiers qui pourraient vous correspondre:" followed by a vertical stack of 13 job titles in orange-outlined boxes: "Planneur / Planteuse stratégique", "Secrétaire", "Responsable relation client (CRM)", "Employé / Employée de laboratoire cinématographique", "Conseiller / Conseillère en gestion de patrimoine", "Enquêteur / Enquêteuse sondage", "Chargé / Chargée de recrutement", "Analyste financier / financière", "Formateur / Formatrice", "Analyste de crédits et risques bancaires", "Chargé / Chargée de communication", "Assistant / Assistante ressources humaines". At the bottom of this box are "Envoyer un message..." and "Envoyer" buttons.

Dans le même sens de nouvelles fonctionnalités ont été ajoutés comme par exemple la possibilité de sélectionner un métier parmi une liste des correspondances trouvées plutôt que d'afficher directement la fiche de chaque métier un par un (et générer une liste extrêmement longue à parcourir).

Dans le même contexte des boutons contextuels ont été ajoutés pour permettre d'en savoir plus sur le métier, en évitant de faire tous les appels d'un coup.

The screenshot shows a list of buttons under the heading "Souhaitez en savoir plus sur ce métier ?". The buttons are: "Ce qu'il faut savoir", "Métiers possibles", "Offres d'emploi", "Métiers proches", and "Marché du travail". Each button is enclosed in an orange-outlined rounded rectangle.

Ces boutons contextuels réalisent des appels AJAX vers de nouveaux points de terminaison de l'API côté serveur, développée avec Flask :

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

```
58
59     # Retour de la réponse en JSON
60     return jsonify({"message": message, "result": result})
61
62 @app.route('/api/metiers/<id>', methods=['GET', 'POST'])
63 def metier_detail(id):
64     """
65     Route pour obtenir les détails d'un métier spécifique.
66
67     Cette route prend un identifiant de métier en paramètre et renvoie les détails
68     de ce métier en format JSON.
69
70     :param id: Identifiant du métier.
71     :return: Réponse JSON contenant les détails du métier.
72     """
73     # Validation de l'identifiant
74     if id is None:
75         return jsonify({"error": "Requête invalide"}), 400
76
77     # Traitement de la requête pour obtenir les détails du métier
78     result = chat.get_metiers().get(id)
79     if result is None:
80         return jsonify({"error": "Aucun métier trouvé"}), 404
81
82     # Retour de la réponse en JSON
83     return jsonify({"result": result.to_dict()})
84
85 @app.route('/api/metiers', methods=['GET', 'POST'])
86 def metiers():
87     """
88     Route pour obtenir la liste complète des métiers.
89
90     Cette route renvoie une liste de tous les métiers disponibles dans le système,
91     ainsi que leur nombre total.
92
93     :return: Réponse JSON contenant la liste des métiers et leur nombre.
94     """
95     # Traitement de la requête pour obtenir tous les métiers
96     metiers = chat.get_metiers().get_all()
97     result = {key: metier.to_dict() for key, metier in metiers.items()}
98
99     # Retour de la réponse en JSON
100    return jsonify({"result": result, "count": len(result)})
101
102 @app.route('/api/emplois/<id>', methods=['GET', 'POST'])
103 def metier_offres_emploi(id):
104     """
105     Route pour obtenir les offres d'emploi associées à un métier spécifique.
106
107     Cette route prend un identifiant de métier en paramètre et renvoie une liste
108     d'offres d'emploi associées à ce métier.
109
110     :param id: Identifiant du métier.
111     :return: Réponse JSON contenant la liste des offres d'emploi et leur nombre.
112     """
113     try:
114         # Traitement de la requête pour obtenir les offres d'emploi
115         result = job_offers_api.get_job_offers(job_id=id, max_offers=10)
116     except:
117         metier_offres_emploi() > try > if len(result) == 0
```

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

```
72
73     return jsonify({"result": result, "count": len(result)})
74
75 @app.route('/api/emplois/', methods=['GET', 'POST'])
76 def metier_offres_emploi(id):
77     """
78     Route pour obtenir les offres d'emploi associées à un métier spécifique.
79
80     Cette route prend un identifiant de métier en paramètre et renvoie une liste
81     d'offres d'emploi associées à ce métier.
82
83     :param id: Identifiant du métier.
84     :return: Réponse JSON contenant la liste des offres d'emploi et leur nombre.
85     """
86     try:
87         # Traitement de la requête pour obtenir les offres d'emploi
88         result = job_offers_api.get_job_offers(job_id=id, max_offers=10)
89
90         if len(result) == 0:
91             return jsonify({"error": "Aucun métier trouvé"}), 404
92
93         # Retour de la réponse en JSON
94         return jsonify({"result": result, "count": len(result)})
95
96     except requests.RequestException:
97         return jsonify({"error": "Erreur lors de la communication avec l'API d'emploi."}), 500
98
99 # Point d'entrée principal de l'application
100 if __name__ == '__main__':
101     app.run(debug=True)
```

Cette route permet d'obtenir une liste des offres d'emploi en temps réel pour un métier sélectionné :

```
3
4 class JobOffersAPI:
5     """
6     Client API pour interroger les offres d'emploi.
7
8     Cette classe permet de récupérer les offres d'emploi à partir d'une API.
9     Elle peut être étendue pour intégrer d'autres endpoints liés aux emplois.
10
11     Dépendances:
12     | - requests: nécessaire pour effectuer des requêtes HTTP.
13
14     Exemple d'utilisation:
15
16     >>> api = JobOffersAPI()
17     >>> offers = api.get_job_offers('M1805', 10)
18     >>> print(offers)
19
20     :param base_url: L'URL de base de l'API. Par défaut, il s'agit de 'https://localhost:5000'.
21     :type base_url: str
22     """
23
24     def __init__(self, base_url='https://candidat.pole-emploi.fr/gw-metierscope'):
25         self.base_url = base_url
26
27     def get_job_offers(self, job_id, max_offers):
28         """
29             Récupère les offres d'emploi pour un identifiant de job spécifié.
30
31             :param job_id: L'identifiant du job pour lequel récupérer les offres.
32             :type job_id: str
33
34             :param max_offers: Le nombre maximal d'offres à récupérer.
35             :type max_offers: int
36
37             :return: Une liste des offres d'emploi.
38             :rtype: list[dict]
39
40             :raises requests.RequestException: Si la requête échoue.
41
42             Exemple d'utilisation:
43
44             >>> api = JobOffersAPI()
45             >>> offers = api.get_job_offers('M1805', 10)
46             >>> print(offers[0]['title']) # Affiche le titre de la première offre
47
48
49             url = f'{self.base_url}/job/{job_id}/offers?max={max_offers}'
50             response = requests.get(url)
51
52             # Gérer les éventuelles erreurs
53             response.raise_for_status()
54
55             return response.json()
```

## VII. Structure et code source documenté

Le projet est organisé en plusieurs fichiers et dossiers qui ont chacun un rôle spécifique dans le fonctionnement de l'application. Voici une description de chaque élément :

- **README.md**  
Ce fichier contient toutes les informations nécessaires pour comprendre le projet, comment l'installer, le configurer et l'utiliser.
- **chatbot.py**  
C'est le cœur du projet. Ce fichier Python contient la logique métier du chatbot, y compris l'interaction avec l'utilisateur et la génération des suggestions de métiers.
- **serveur.py**  
Ce fichier Python utilise Flask pour créer un serveur web et une API backend. Il gère les requêtes HTTP et communique avec chatbot.py pour obtenir les réponses à envoyer à l'utilisateur.
- **requirements.txt**  
Ce fichier liste toutes les dépendances Python nécessaires pour exécuter le projet. Il est utilisé pour installer ces dépendances en une seule commande (pip install -r requirements.txt).
- **templates/index.html**  
Ce fichier HTML constitue la structure de base du site web. Il est utilisé par le serveur Flask pour générer la page web que voit l'utilisateur.
- **static/chatbot.js**  
Ce fichier JavaScript gère la partie client du chatbot. Il s'occupe de l'interaction avec l'utilisateur sur la page web, envoie les requêtes au serveur via AJAX et affiche les réponses.
- **static/style.css**  
Ce fichier contient tous les styles CSS qui sont appliqués à la page web, autrement dit l'interface utilisateur (UI)
- **static/img/logo.png**  
Ce fichier contient le logo du projet, qui est affiché sur la page web.
- **models/metier.py**  
Ce fichier définit la classe Metier, qui est responsable de la gestion des données relatives à un métier spécifique. La classe contient des méthodes et des propriétés pour accéder aux différentes informations d'un métier, comme son ID, son nom, ses appellations, ses conditions de travail, etc.
- **models/metiers.py**  
Ce fichier contient la classe Metiers, qui gère l'ensemble des métiers disponibles. Elle charge les données des métiers à partir d'une source de données et les stocke dans un dictionnaire pour un accès rapide. La classe fournit également des méthodes pour récupérer des métiers spécifiques par leur ID ou leur label.
- **api/job\_offers\_api.py**  
Ce fichier contient la classe responsable de la récupération des offres d'emploi à partir d'une API externe. Elle gère les appels API pour obtenir les dernières offres d'emploi en relation avec un métier spécifique et les retourne sous un format exploitable par le reste de l'application.
- **dataset/metiers.json**  
Ce fichier JSON contient le dataset complet de tous les métiers. Il est utilisé comme source de données principale pour le projet et est chargé par la classe Metiers pour initialiser l'ensemble des objets Metier.
- **dataset/metiers-simple.json**  
Ce fichier JSON est un échantillon réduit contenant seulement 4 métiers. Il est principalement utilisé pour les tests et le développement, permettant d'éviter des temps de chargement longs lors de l'exécution de tests ou de l'initialisation du projet.

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

## Code source documenté

Voici l'ensemble des fichiers constituant le code source documenté du projet

### README.md

Voici le fichier README.md :

```
# Système de Suggestion de métiers
```

Ce projet sert à suggérer des métiers en fonction des compétences et intérêts des utilisateurs.

```
## Configuration initiale
```

```
### Prerequisites
```

- `Flask 2.2.0` : Python >=3.5
- `Flask-CORS` : Python >=2.7
- `Spacy 3.6.1` : Python >=3.6,<3.10
- `Requests 2.31.0` : Python >=3.6

```
---
```

```
### Installation
```

1. Installez les dépendances à l'aide de pip:

```
```bash
pip install -r requirements.txt
````
```

2. Téléchargez le modèle `fr\_core\_news\_sm` pour Spacy :

```
```bash
python -m spacy download fr_core_news_sm
````
```

```
## Utilisation
```

```
### Lancement du serveur
```

Pour démarrer le serveur, exécutez :

```
```bash
python server.py
````
```

Le serveur démarrera en mode debug et le site web sera accessible via l'URL <http://127.0.0.1:5000/>.

```
### Test du Chatbot
```

Pour tester les fonctionnalités du Chatbot sans passer par le site web, exécutez :

```
```bash
python chatbot.py
````
```

Suivez ensuite les instructions à l'écran.

```
## Contribution
```

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

Les contributions sont les bienvenues! Veuillez créer une "pull request" avec vos modifications.

---

### chatbot.py

Voici le fichier chatbot.py :

```
"""
Auteur: Aram BABAYAN
Date: 01/08/2023
Version: 1.0
"""

import spacy
from spacy.matcher import Matcher
from spacy import displacy

from models.metiers import Metiers
import time

class SuggestionMetier:
    """
    Classe pour suggérer des métiers basés sur le texte donné.
    """

    def __init__(self):
        """
        Constructeur de la classe SuggestionMetier.
        """
        self.metiers = Metiers('dataset/metiers.json')
        self.nlp = spacy.load("fr_core_news_sm")

        pattern_list = self.get_pattern_list()
        self.matcher = self.create_skill_matcher(pattern_list)

    def get_metiers(self) -> Metiers:
        """
        Retourne l'objet Metiers.

        Returns:
            Metiers: L'objet Metiers.
        """
        return self.metiers

    def extract_data(self, data_list, metier_id):
        """
        Extrait les données pertinentes de la liste donnée.

        Args:
            data_list (list): Liste des données à extraire.
            metier_id (str): ID du métier associé.

        Returns:
            list: Liste des données extraites.
        """
        extracted = []
        for item in data_list:
            label = item.get('label', None) if isinstance(item, dict) else item
            if label:
                # label = self.clean(label)

```

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

```
        extracted.append((label, metier_id))
    return extracted

def get_pattern_list(self):
    """
    Obtient la liste des motifs pour le Matcher.

    Returns:
        list: Liste des motifs.
    """

    pattern_list = []
    for id, metier in self.metiers.get_all().items():
        pattern_list.extend(self.extract_data([metier.name], id))
        pattern_list.extend(self.extract_data(metier.appellations, id))
        pattern_list.extend(self.extract_data(metier.work_conditions, id))
        pattern_list.extend(self.extract_data([k for category in
metier.knowledge_categories for k in category.get('knowledges', [])], id))
        pattern_list.extend(self.extract_data(metier.professional_soft_skills, id))

    pattern_list = list(set(pattern_list))
    return pattern_list

def create_skill_matcher(self, pattern_list):
    """
    Crée un Matcher basé sur la liste des motifs.

    Args:
        pattern_list (list): Liste des motifs.

    Returns:
        Matcher: L'objet Matcher.
    """

    matcher = Matcher(self.nlp.vocab)

    # Pré-traiter les compétences en bloc
    skills_texts = [skill.lower() for skill, _ in pattern_list]
    docs = list(self.nlp.pipe(skills_texts))

    for (skill, metier_code), doc in zip(pattern_list, docs):
        pattern = [{"LEMMA": token.lemma_} for token in doc]
        matcher.add(metier_code, [pattern])

    return matcher

def clean(self, skill):
    """
    Nettoie et lemmatise le texte donné.

    Args:
        skill (str): Texte à nettoyer.

    Returns:
        str: Texte nettoyé.
    """

    custom_stoplist = ["possibilité", "faire"]
    tokens = [token for token in self.nlp(skill.lower())]
    tokens = [token for token in tokens if not token.is_stop and not token.is_punct and
token.text not in custom_stoplist]
    lemmatized_tokens = [token.lemma_ for token in tokens]
    return " ".join(lemmatized_tokens)
```

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

```
def get_suggestions(self, texte, visualiser=False):
    """
    Obtient des suggestions de métiers basées sur le texte donné.

    Args:
        texte (str): Texte à analyser.
        visualiser (bool, optional): Si True, visualise les entités. Defaults to False.

    Returns:
        list: Liste des suggestions de métiers.
    """
    print("Texte nettoyé: " + self.clean(texte))
    doc = self.nlp(self.clean(texte))
    matches = self.matcher(doc)
    result_list = set()
    for match_id, start, end in matches:
        rule_id = self.nlp.vocab.strings[match_id]
        span = doc[start:end]
        print('Compétence trouvée: ', span.text, rule_id)
        if rule_id not in result_list:
            doc.ents = list(doc.ents) + [span]
            result_list.add(rule_id)

    if visualiser:
        displacy.serve(doc, style='ent')

    results = []
    for metier_id in result_list:
        metier = self.metiers.get_metier_by_id(metier_id)
        results.append({
            "code": metier_id,
            "label": metier.name,
        })

    return results

def suggerer_message(self, suggestions):
    """
    Génère un message basé sur les suggestions de métiers.

    Args:
        suggestions (list): Liste des suggestions de métiers.

    Returns:
        str: Message suggéré.
    """
    if not suggestions:
        return "Aucune suggestion de métier pour l'instant."
    else:
        metiers_suggested = ", ".join([s["label"] for s in suggestions])
        return "Vous pourriez envisager les métiers suivants : " + metiers_suggested + \
               "(code: " + ", ".join([s["code"] for s in suggestions]) + ")"

if __name__ == '__main__':
    print('Démarrage du chatbot...')
    chat = SuggestionMetier()

    while True:
        texte = input("Entrez votre texte: ")
        suggestions = chat.get_suggestions(texte)
        print(chat.suggerer_message(suggestions))
```

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

### serveur.py

Voici le fichier serveur.py:

```
"""
Auteur: Aram BABAYAN
Date: 01/08/2023
Version: 1.0
"""

# Importation des bibliothèques nécessaires
import time
from flask import Flask, jsonify, render_template, request
from flask_cors import CORS
import requests

# Importation des modules personnalisés
from api.job_offers_api import JobOffersAPI
from chatbot import SuggestionMetier

# Initialisation de l'application Flask et configuration CORS
app = Flask(__name__)
CORS(app)

# Initialisation des objets pour le chatbot et l'API des offres d'emploi
chat = SuggestionMetier()
job_offers_api = JobOffersAPI()

@app.route('/')
def home():
    """
    Route pour la page d'accueil.

    Cette route sert à afficher la page d'accueil de l'application.

    :return: Le template HTML de la page d'accueil.
    """
    return render_template("index.html")

@app.route('/api/query', methods=['POST'])
def query():
    """
    Route pour traiter les requêtes de chat.

    Cette route prend en charge les requêtes POST contenant une requête de chat
    et renvoie une réponse formatée en JSON.

    :return: Réponse JSON contenant le message et le résultat.
    """
    # Récupération de la requête depuis le formulaire
    query = request.form.get('query')

    # Validation de la requête
    if query is None:
        return jsonify({"error": "Requête invalide"}), 400

    # Traitement de la requête via le chatbot
    result = chat.get_suggestions(query)
    message = chat.sugḡer_message(result)

    # Retour de la réponse en JSON
    return jsonify({"message": message, "result": result})
```

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

```
@app.route('/api/metiers/<id>', methods=['GET', 'POST'])
def metier_detail(id):
    """
    Route pour obtenir les détails d'un métier spécifique.

    Cette route prend un identifiant de métier en paramètre et renvoie les détails
    de ce métier en format JSON.

    :param id: Identifiant du métier.
    :return: Réponse JSON contenant les détails du métier.
    """

    # Validation de l'identifiant
    if id is None:
        return jsonify({"error": "Requête invalide"}), 400

    # Traitement de la requête pour obtenir les détails du métier
    result = chat.get_metiers().get(id)
    if result is None:
        return jsonify({"error": "Aucun métier trouvé"}), 404

    # Retour de la réponse en JSON
    return jsonify({"result": result.to_dict()})

@app.route('/api/metiers', methods=['GET', 'POST'])
def metiers():
    """
    Route pour obtenir la liste complète des métiers.

    Cette route renvoie une liste de tous les métiers disponibles dans le système,
    ainsi que leur nombre total.

    :return: Réponse JSON contenant la liste des métiers et leur nombre.
    """

    # Traitement de la requête pour obtenir tous les métiers
    metiers = chat.get_metiers().get_all()
    result = {key: metier.to_dict() for key, metier in metiers.items()}

    # Retour de la réponse en JSON
    return jsonify({"result": result, "count": len(result)})

@app.route('/api/emplois/<id>', methods=['GET', 'POST'])
def metier_offres_emploi(id):
    """
    Route pour obtenir les offres d'emploi associées à un métier spécifique.

    Cette route prend un identifiant de métier en paramètre et renvoie une liste
    d'offres d'emploi associées à ce métier.

    :param id: Identifiant du métier.
    :return: Réponse JSON contenant la liste des offres d'emploi et leur nombre.
    """

    try:
        # Traitement de la requête pour obtenir les offres d'emploi
        result = job_offers_api.get_job_offers(job_id=id, max_offers=10)

        if len(result) == 0:
            return jsonify({"error": "Aucun métier trouvé"}), 404

        # Retour de la réponse en JSON
        return jsonify({"result": result, "count": len(result)})

    except requests.RequestException:
```

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

```
    return jsonify({"error": "Erreur lors de la communication avec l'API d'emploi."}), 500

# Point d'entrée principal de l'application
if __name__ == '__main__':
    app.run(debug=True)
```

### requirements.txt

Voici le fichier requirements.txt :

```
flask~=2.2.0
flask_cors
spacy~=3.6.1
requests~=2.31.0
```

### templates/index.html

Voici le fichier templates/index.html :

```
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Chatbot</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-KK94CHFLLe+nY2dmCWGMq91rCGa5gtU4mk92HdvYe+M/SXH301p5ILy+dN9+nJOZ" crossorigin="anonymous">
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha3/dist/js/bootstrap.bundle.min.js" integrity="sha384-ENjdO4Dr2bkBFxQpeoTz1HICje39Wm4jDKdf19U8gI4ddQ3GYNs7NTKfAdVQSze" crossorigin="anonymous"></script>
    <script src="https://code.jquery.com/jquery-latest.min.js"></script>
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.10.5/font/bootstrap-icons.css">
    <script src="https://cdnjs.cloudflare.com/ajax/libs/lodash.js/4.17.21/lodash.min.js"></script>
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.1/css/all.min.css">

    <script src="https://cdnjs.cloudflare.com/ajax/libs/moment.js/2.29.1/moment.min.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/moment.js/2.29.1/locale/fr.min.js"></script>

    <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}>
    <script src="{{ url_for('static', filename='chatbot.js') }}"></script>
</head>
<body>

    <div class="chat-wrapper h-100 w-100">
        <div class="row h-100">
            <div class="col-lg-3 col-md-4">
                <div class="logo-wrapper align-items-center text-center">
                    
                </div>
                <div class="d-flex justify-content-center">
                    <div class="menu">
                        <div class="col-md-12">
                            <ul class="list-group">
                                <li class="list-group-item d-flex align-items-center active">
                                    <i class="fas fa-comments mr-2"></i>
                                    <a href="/">Chatbot</a>
                                </li>
```

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

```
<li class="list-group-item d-flex align-items-center" data-role="job-list">
    <i class="fas fa-briefcase mr-2"></i>
    <a href="#">Liste des métiers</a>
</li>
<li class="list-group-item d-flex align-items-center">
    <i class="fas fa-star mr-2"></i>
    <a href="#">Centres d'intérêt</a>
</li>
</ul>
</div>
</div>
</div>

<div class="col-lg-9 col-md-8">
    <div class="wrapper">
        <div class="title">Chatbot</div>
        <div class="form">
            <div class="bot-inbox inbox">
                <div class="msg-header">
                    <p>
                        Bienvenue à notre assistant de suggestion de métier! ⭐
                <br><br>
                    Pour vous aider à trouver un métier qui pourrait
                    correspondre à vos compétences et intérêts, j'aurais besoin de connaître vos compétences,
                    domaines d'intérêt, et toute autre information que vous jugeriez utile.<br><br>
                    Si vous souhaitez obtenir quelques exemples de saisie,
                    saisissez "aide".
                </p>
                </div>
            </div>
            <div class="typing-field">
                <div class="input-query">
                    <input id="query" type="text" placeholder="Envoyer un message..." required value="télétravail">
                    <button id="send-btn">Envoyer</button>
                </div>
            </div>
        </div>
    </div>
</div>

</body>
</html>
```

### static/chatbot.js

Voici le fichier static/chatbot.js :

```
class GuideMetierAPI {
    // Récupérer des suggestions basées sur un message
    query(message) {
        return $.post('/api/query', {query: message});
    }

    // Récupérer des détails sur un métier spécifique par ID
    getMetierDetail(id) {
        return $.ajax({
            url: '/api/metiers/${id}',
            method: 'GET'
```

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

```
        });
    }

    // Récupérer une liste de tous les métiers
    getAllMetiers() {
        return $.ajax({
            url: '/api/metiers',
            method: 'GET'
        });
    }

    // Récupérer des offres d'emploi pour un métier spécifique par ID
    getJobOffersForMetier(id) {
        return $.ajax({
            url: '/api/emplois/${id}',
            method: 'GET'
        });
    }
}

function nl2br (str, is_xhtml) {
    if (typeof str === 'undefined' || str === null) {
        return '';
    }
    var breakTag = (is_xhtml || typeof is_xhtml === 'undefined') ? '<br />' : '<br>';
    return (str + '').replace(/([>]\r\n)?(\r\n|\n\r|\r|\n)/g, '$1' + breakTag + '$2');
}

$(document).on('click', '#send-btn', async function () {
    var message = $('#query').val();
    if (!message.length) return false;

    // Ajouter le message à la liste - fonction
    renderUserMessage(message);

    // Scroll en bas du chat
    $('.form').scrollTop($('.form')[0].scrollHeight);

    // On envoie la requête à l'GuideMetierAPI
    let api = new GuideMetierAPI();
    let response = await api.query(message)
    $('.loading-icon').remove();

    if (response.result.length === 0) return notFound();

    renderJobList(response.result)

    scrollDown()
});

$(document).on('click', '.list-group li', async function() {
    $(this).parent().find('.active').removeClass('active')
    $(this).addClass('active')
})

$(document).on('click', '[data-role="job-list"]', async function() {
    renderUserMessage('Je souhaite voir la liste des métiers');

    // On envoie la requête à l'GuideMetierAPI
    let api = new GuideMetierAPI();
    let response = await api.getAllMetiers()
```

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

```
// On affiche côté client le message transmis
$('.loading-icon').remove();

if (response.result.length === 0) return notFound();

renderAllJobList(response.result, false)
})

$(document).on('click', '[data-role="job-detail"]', async function() {
    let id = $(this).data('rome');
    renderUserMessage('Je souhaite en savoir plus sur le métier de ' + $(this).text());

    let api = new GuideMetierAPI();
    let data = await api.getMetierDetail(id);
    let item = data.result;

    let container = '<div class="item-container">';

    container += '<h6>' + item.metier.mainName + '</h6>';
    container += '<p>' + item.label + '</p>';
    if (item.metier.domain) {
        container += '<h7>Domaine :</h7><p>' + item.metier.domain.label + '</p>';
    }

    container += '<h7>Les missions principales:</h7>';
    container += '<p class="desc">' + nl2br(item.metier.description).split("\n").join('<br>')
+ '</p>';

    if (item.metier.accessText && item.metier.accessText.length > 0) {
        container += '<h7>Comment y accéder</h7> ' +
item.metier.accessText.split("\n").join('<br>') + '</p>';
    }

    let skillsList = '<ul>';
    $.each(item.skills.domainsSkills, function (i, domainSkill) {
        if (domainSkill.score > 13) {
            skillsList += '<li>' + domainSkill.label + ' (' + domainSkill.score + '%)</li>';
            let stakesList = '<ul>';
            $.each(domainSkill.stakes, function (j, stake) {
                stakesList += '<li>' + stake.label + '</li>';
            });
            skillsList += stakesList + '</ul>';
        }
    });
    skillsList += '</ul>';
    container += '<h7>Ce qu'il faut savoir faire:</h7>' + skillsList;
    container += '</div>';

$.each(item.workConditionsCategories, function(i, category) {
    let workConditionsList = $('<ul></ul>');
    $.each(category.workConditions, function(j, condition) {
        workConditionsList.append('<li>' + condition.label + '</li>');
    });
    container += '<h7>' + translateLabel(category.label) + ':</h7>' +
workConditionsList.prop('outerHTML');
});

console.log(item)
if (item.skills.professionalSoftSkills && item.skills.professionalSoftSkills.length > 0) {
    let professionalSoftSkills = $('<ul></ul>');
    $.each(item.skills.professionalSoftSkills, function(j, condition) {
```

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

```
        professionalSoftSkills.append('<li>' + condition.label + '</li>');
    });
    container += '<h7>Savoir-être professionnels</h7>' +
professionalSoftSkills.prop('outerHTML');
}

container +=
```
<h7>Vidéo de présentation :</h7>
<div class="text-center">
<iframe class="embed-responsive-item" id="ytplayer" type="text/html" width="640"
height="360" src="${item.metier.videoUrl}" allowfullscreen=""></iframe>
</div>

await renderMessage(container);
await renderJobDetailActions(id);
scrollDown();
};

function translateLabel(label) {
switch(label) {
    case "CONDITIONS_TRAVAIL":
        return "Conditions de travail";
    case "HORAIRE_ET_DUREE_TRAVAIL":
        return "Horaires et durée de travail";
    default:
        return label;
}
}

function renderJobDetailActions(id) {
let html =
'Souhaitez en savoir plus sur ce métier ?' +
'<button class="btn btn-primary" data-role="job-detail-knowledgeCategories" data-
rome="' + id + '">Ce qu'il faut savoir</button>' +
'<button class="btn btn-primary" data-role="job-detail-appellations" data-rome="' + id +
'">Métiers possibles</button>' +
'<button class="btn btn-primary" data-role="job-detail-jobOffers" data-rome="' + id +
'">Offres d'emploi</button>' +
'<button class="btn btn-primary" data-role="job-detail-relatedJobs" data-rome="' + id +
'">Métiers proches</button>' +
'<button class="btn btn-primary" data-role="job-detail-labourMarket" data-rome="' + id +
'">Marché du travail</button>'
return renderMessage(html);
}

$(document).off('click', '[data-role="job-detail-labourMarket"]');
$(document).on('click', '[data-role="job-detail-labourMarket"]', async function() {
let id = $(this).data('rome');
renderUserMessage('Je souhaite en savoir plus sur le marché du travail pour ce métier');

let api = new GuideMetierAPI();
let data = await api.getMetierDetail(id);
let item = data.result;

let container = '<div class="item-container card">';
// Ajout du titre
container += '<div class="card-header">Le marché du travail</div>; // Utiliser le
composant "card-header" de Bootstrap pour le titre

container += '<div class="card-body">; // Contenu principal de la carte
```

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

```
if (!item.labourMarket) {
    return await renderMessage('Aucune information sur le marché du travail disponible');
}

// Affiche les offres d'emploi avec une icône (exemple)
if (item.labourMarket.jobOffers) {
    container += `<p class="mb-2"><i class="bi bi-briefcase-fill me-2"></i>Offres
d'emploi: <span class="badge bg-primary">' + item.labourMarket.jobOffers +
    '</span> déposées sur Pôle emploi au cours des 12 derniers mois.</p>`;
}

// Affiche les demandeurs d'emploi avec une icône (exemple)
if (item.labourMarket.jobSeekers) {
    container += `<p class="mb-2"><i class="bi bi-person-fill me-2"></i>Demandeurs
d'emploi: <span class="badge bg-warning">' + item.labourMarket.jobSeekers + '</span></p>`;
}

container += `</div>`; // Fin du contenu principal de la carte
container += `</div>`; // Fin de la carte

if (item.labourMarket.jobOffers && item.labourMarket.jobSeekers) {
    let alertClass;
    let ratioMessage;
    let ratio = item.labourMarket.jobOffers / item.labourMarket.jobSeekers;

    if (ratio > 1) {
        alertClass = "alert-success";
        ratioMessage = `Bonne nouvelle ! Ce métier dispose de ${Math.round(ratio)}x plus
d'offres d'emploi que de demandeurs d'emploi.`;
    } else if (ratio < 1) {
        alertClass = "alert-danger";
        ratioMessage = `Ce métier dispose de ${Math.round(1/ratio)}x plus de demandeurs
d'emploi que d'offres d'emploi.`;
    } else {
        alertClass = "alert-info";
        ratioMessage = `Le nombre d'offres d'emploi est égal au nombre de demandeurs
d'emploi pour ce métier.`;
    }

    container += `
        <div class="alert ${alertClass} mt-3">
            ${ratioMessage}
        </div>
    `;
}

await renderMessage(container);
scrollDown();
});

$(document).off('click', '[data-role="job-detail-knowledgeCategories"]');
$(document).on('click', '[data-role="job-detail-knowledgeCategories"]', async function() {
    let id = $(this).data('rome');
    renderUserMessage('Je souhaite en savoir plus sur ce qu'il faut savoir dans ce métier');

    let api = new GuideMetierAPI();
    let data = await api.getMetierDetail(id);
    let item = data.result;
```

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

```
let container = '<div class="item-container">';

if (!item.skills || !item.skills.knowledgeCategories ||
item.skills.knowledgeCategories.length == 0) {
    return await renderMessage('Aucune information disponible');
}

$.each(item.skills.knowledgeCategories, function(i, category) {
    let knowledgeList = $('<ul></ul>');
    $.each(category.knowledges, function(j, knowledge) {
        knowledgeList.append('<li>' + knowledge.label + '</li>');
    });
    container += '<h7>' + category.label + ':</h7>' + knowledgeList.prop('outerHTML');
});

container += '</div>';

await renderMessage(container);
scrollDown();
});

;

$(document).on('click', '[data-role="job-detail-jobOffers"]', async function() {
    let id = $(this).data('rome');
    renderUserMessage('Je souhaite voir les offres d\'emploi récentes pour ce métier');

    await renderJobOffers(id);
});

$(document).on('click', '[data-role="job-detail-appellations"]', async function() {
    let id = $(this).data('rome');
    renderUserMessage('Je souhaite en savoir plus sur les métiers possibles');

    let api = new GuideMetierAPI();
    let data = await api.getMetierDetail(id);
    let item = data.result;

    let container = '<div class="item-container">';

    // Code pour afficher les appellations
    if (item.metier.appellations && item.metier.appellations.length > 0) {
        let appellationList = $('<ul></ul>');
        $.each(item.metier.appellations, function(i, appellation) {
            appellationList.append('<li>' + appellation.label + '</li>');
        });
        container += '<h7>Métiers possibles:</h7>' + appellationList.prop('outerHTML');
    }

    container += '</div>';

    await renderMessage(container);
    scrollDown();
});

$(document).on('click', '[data-role="job-detail-relatedJobs"]', async function() {
    let id = $(this).data('rome');
    renderUserMessage('Je souhaite en savoir plus sur les métiers proches');

    let api = new GuideMetierAPI();
    let data = await api.getMetierDetail(id);
```

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

```
let item = data.result;

let container = '<div class="item-container">';

if (item.relatedJobs && item.relatedJobs.length > 0) {
    let relatedJobsList = $('<ul></ul>');
    $.each(item.relatedJobs, function(i, relatedJob) {
        relatedJobsList.append('<li>' + relatedJob.label + '</li>');
    });
    container += '<h7>Métiers proches:</h7>' + relatedJobsList.prop('outerHTML');
}

container += '</div>';

await renderMessage(container);
scrollDown();
});

function scrollDown() {
    setTimeout(function() {
        // scroll en bas du chat une fois l'animation CSS terminée
        $('.form').scrollTop($(".form")[0].scrollHeight);
    }, 200);
}

function notFound(item) {
    return $('.form').append(
        '<div class="bot-inbox inbox">' +
        '<div class="msg-header"><p>Désolé, je n\'ai pas compris votre
question.</p></div>' +
        '</div>';
    );
}

function renderMessage(message, speed = 3) {
    return new Promise((resolve, reject) => {
        $('.loading-icon').remove();
        let containerHtml =
            '<div class="bot-inbox inbox">' +
            '<div class="msg-header"></div>' +
            '</div>';

        $('.form').append(containerHtml);

        let $messageBox = $('.form .msg-header').last();

        if (speed === false) {
            $messageBox.html(message);
            scrollDown()
            resolve();
            return;
        }

        let currentContent = "";
        function typeMessage(index, speed) {
            if (index < message.length) {
                currentContent += message[index];
                $messageBox.html(currentContent);
            }
        }

        for (let i = 0; i < message.length; i++) {
            setTimeout(typeMessage, speed * i, i);
        }
    });
}
```

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

```
        setTimeout(() => {
            typeMessage(index + 1);
            scrollDown()
        }, speed);
    } else {
        resolve();
    }
}

// Start typing the message
typeMessage(0, speed);
});;
}

function renderUserMessage(message) {
    // On rajoute le message saisi
    $('.form').append(
        '<div class="user-inbox inbox">' +
        '<div class="msg-header"><p>' + message + '</p></div>' +
        '</div>'
    );
    $('#query').val(' ');

    // Petit loader en attente de la réponse AJAX
    $('.form').append(
        '<div class="loading-icon lds-ellipsis"><div></div><div></div><div></div><div></div><div></div></div>'
    );
}

function renderJobList(jobs) {
    let message = "Voici les métiers qui pourraient vous correspondre :";

    $.each(jobs, function(index, item) {
        message += "<button class='btn btn-primary' data-role='job-detail' data-rome='${item.code || item.romeCode}'>${item.mainName || item.label}</button> ";
    });

    renderMessage(message);
}

function renderAllJobList(jobs) {
    let message = "Voici la liste de tous les métiers :";

    $.each(jobs, function(index, item) {
        message += "<button class='btn btn-primary' data-role='job-detail' data-rome='${item.code || item.romeCode}'>${item.mainName || item.label}</button> ";
    });

    renderMessage(message, false);
}

async function renderAllMetiers() {
    let api = new GuideMetierAPI();
    let data = await api.getAllMetiers();

    let message = '<ul class="metiers-list">';

    $.each(data.result, function(index, item) {
```

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

```
        message += <button class="btn btn-primary" data-role="job-detail" data-
rome="${item.romeCode}">${item.mainName || item.label}</button>;
    });

    message += '</ul>';
    renderMessage(message);
}

async function renderJobOffers(id) {
    let api = new GuideMetierAPI();
    let offersResponse = await api.getJobOffersForMetier(id);
    let offers = offersResponse.result;

    if(offersResponse.count === 0) {
        renderMessage('<div class="guide-inbox inbox">Aucune offre d\'emploi disponible pour
ce métier.</div>');
        return;
    }

    let message = 'Voici les récentes offres d\'emploi disponibles pour ce métier :';

    let table = '<table class="offers-table">';
    table += '<thead><tr>' +
        '<th width="130px">Date</th>' +
        '<th>Titre</th>' +
        '<th width="130px">Localisation</th>' +
        '<th width="180px">Contrat</th>' +
        '<th>Description</th>' +
        '<th width="100px"></th>' +
        '</tr></thead><tbody>';

    $.each(offers, function(index, offer) {
        let detailUrl = 'https://candidat.pole-emploi.fr/offres/recherche/detail/';
        let shortDescription = offer.description.length > 150 ? offer.description.substr(0,
147) + "..." : offer.description;

        var date = offer.creationDate
        var m = moment(date);
        table += <tr>
            <td>${m.fromNow()}</td>
            <td>${offer.title}</td>
            <td>${offer.location}</td>
            <td>${offer.contractType}</td>
            <td>${shortDescription}</td>
            <td>
                <a target="_blank" href="${detailUrl + offer.referenceOffre}" class="btn btn-
primary btn-sm">
                    Voir l'offre
                </a>
            </td>
        </tr>;
    });

    table += '</tbody></table>';

    await renderMessage(message);
    await renderMessage(table);
}
$(document).on('keyup', '#query', function(e) {
```

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

```
// Envoi de la pquête lorsqu'on appuie sur Entrée dans le contexte du chat
if ((e.keyCode || e.which) === 13) {
    $('#send-btn').trigger('click')
}
});
```

### static/style.css

Voici le fichier static/style.css :

```
@import
url('https://fonts.googleapis.com/css2?family=Poppins:wght@200;300;400;500;600;700&display=swa
p');
* {margin: 0; padding: 0; box-sizing: border-box; font-family: 'Poppins', sans-serif}
html,body {display: grid; height: 100%; width: 100%; place-items: center; background: #24252d; }
::selection {color: #fff; background: #007bff}
::-webkit-scrollbar {width: 3px; border-radius: 25px}
::-webkit-scrollbar-track {background: #f1f1f1}
::-webkit-scrollbar-thumb {background: #ddd}
::-webkit-scrollbar-thumb:hover {background: #ccc}
.wrapper {width: 100%; background: #fff; /* max-width: 600px; */ border-radius: 0; }
.wrapper .title {color: #142e49; font-size: 20px; font-weight: 500; line-height: 60px; border-
bottom: 1px solid #eee; padding-left: 30px; box-shadow: 0 0px 2px #3333330f; position: relative; }
.wrapper .form {padding: 20px 15px; overflow-y: auto; background: #f3f3f3; height:
100%; background: #fff; overflow-y: scroll; max-height: calc(100vh - 120px); }
.wrapper .form .inbox {width: 100%; display: flex; align-items: baseline; animation: slideTop
0.3s ease; margin: 13px 0; }
.wrapper .form .user-inbox {justify-content: flex-end; }
.wrapper .form .inbox .msg-header {/* max-width: 53%; */ margin-left: 10px; height: auto; }
.form .inbox .msg-header {color: #24252d; background: #f17b54; padding: 14px 13px; font-size:
14px; border-radius: 10px 10px 10px 0; background: #f4f4f4; }
.form .user-inbox.inbox .msg-header {border-radius: 10px 10px 0px; background: #24252d; color:
#fff; }
.form .user-inbox .msg-header {color: #333; background: #555; }
.wrapper .typing-field {display: flex; height: 60px; width: 100%; align-items: center; justify-
content: space-evenly; border-radius: 0; background: #eee; bottom: 0px; }
.wrapper .typing-field .input-query {height: 40px; width: 100%; position: relative; }
.wrapper .typing-field .input-query input {height: 100%; outline: none; border: 1px solid
transparent; padding: 0 80px 0 30px; border-radius: 3px; font-size: 15px; background:
transparent; transition: all 0.3s ease; width: calc(100% - 110px); }
.typing-field .input-query input:focus {/* border-color: rgba(0,123,255,0.8); */ }
.input-query input::placeholder {color: #999999; transition: all 0.3s ease}
.input-query input:focus::placeholder {color: #bfbfbf}
.wrapper .typing-field .input-query button {height: 40px; color: #fff; cursor: pointer; outline:
none; background: #d06a49 !important; border: 1px solid #d06a49 !important; transition: all 0.3s
ease; width: 80px; border-radius: 30px; font-size: 14px; z-index: 9999; margin: 0; }
.wrapper .typing-field .input-query input:valid ~ button {opacity: 1; pointer-events: auto}
.typing-field .input-query button:hover {background: #006fef}

/* animations css */
.lds-ellipsis {display: inline-block; position: relative; width: 90px; height: 60px;
background: #eee; border-radius: 10px; text-align: center; transform: scale(0.7) }
.lds-ellipsis div {position: absolute; top: 23px; width: 13px; height: 13px; border-radius:
50%; background: #ccc; animation-timing-function: cubic-bezier(0, 1, 1, 0); margin-left: 8px}
.lds-ellipsis div:nth-child(1) {left: 8px; animation: lds-ellipsis1 0.6s infinite}
.lds-ellipsis div:nth-child(2) {left: 8px; animation: lds-ellipsis2 0.6s infinite}
.lds-ellipsis div:nth-child(3) {left: 32px; animation: lds-ellipsis2 0.6s infinite}
.lds-ellipsis div:nth-child(4) {left: 56px; animation: lds-ellipsis3 0.6s infinite}

@keyframes lds-ellipsis1 {
    0% {transform: scale(0)}
    100% {transform: scale(1)}
```

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

```
}

@keyframes lds-ellipsis3 {
    0% {transform: scale(1)}
    100% {transform: scale(0)}
}
@keyframes lds-ellipsis2 {
    0% {transform: translate(0, 0)}
    100% {transform: translate(24px, 0)}
}

@keyframes slideTop {
    0% {margin-bottom: -25px}
    100% {margin-bottom: 0}
}

ul {
    margin-left: 20px;
}

h4 {
    margin-top: 20px;
}

.list-group li {
    background: transparent;
    border: 0;
    padding: 5px 14px 3px 19px;
    margin-left: 0;
    margin: 5px 0px;
    font-size: 15px;
    width: 100%;
}

ul.list-group {
    margin: 0;
    color: #fff;
    max-width: 200px;
    width: 100%
}

ul.list-group li i, ul.list-group li a {
    color: #eaecff;
    text-decoration: none;
    padding: 10px 6px;
}

ul.list-group li i {
    color: #ced0d8;
}

.list-group-item.active {
    border-radius: 15px;
    background: #33343e;
    margin: 5px 0px;
}

ul.list-group li.active a {
    color: #fff;
}
```

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

```
ul.list-group li.active i {
    color: #f17b54;
}

.wrapper {
    height: 100%;
    max-height: 100%;
    width: 100%;
    overflow: hidden;
}

.form .user-inbox.inbox .msg-header p {
    padding: 0;
    margin: 0;
}

.inbox button {
    border: 1px solid #f17b54;
    color: #f17b54;
    margin: 8px 0 0;
    padding: 7px 20px;
    font-size: 14px;
    background: transparent;
    border-radius: 100px;
    display: block;
    text-align: left;
}

.inbox button:hover {
    border-color: #f17b54;
    background: #f17b54;
    color: #fff;
}

.chat-wrapper, .chat-wrapper > .row, .chat-wrapper .col-md-9 {
    margin: 0;
    padding: 0;
}

h7 {
    display: block;
    font-weight: bold;
}

.wrapper .form .inbox .msg-header p {margin: 0}

.chat-wrapper table td, .chat-wrapper table th {
    padding: 2px 3px;
}

h5, h6, h7 {
    margin-top: 10px;
}
```

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

### models/metier.py

Voici le fichier models/metier.py :

```
"""
Auteur: Aram BABAYAN
Date: 01/08/2023
Version: 1.0
"""

class Metier:
    """
    Classe Metier pour gérer les informations sur un métier.

    Attributs:
        data (dict): Dictionnaire contenant les données du métier.
    """

    def __init__(self, data):
        """
        Constructeur de la classe.

        Args:
            data (dict): Dictionnaire contenant les données du métier.
        """
        self.data = data

    def to_dict(self):
        """
        Convertit les données de l'objet en dictionnaire.

        Returns:
            dict: Dictionnaire contenant les données du métier.
        """
        return self.data

    @property
    def id(self):
        """
        Obtient l'ID du métier.

        Returns:
            str: ID du métier.
        """
        return self.data.get('romeCode', '')

    @property
    def label(self):
        """
        Obtient le label du métier.

        Returns:
            str: Label du métier.
        """
        return self.data.get('label', '')

    @property
    def name(self):
        """
        Obtient le nom principal du métier.

        Returns:
            str: Nom principal du métier.
        """

```

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

```
        return self.data.get('mainName', '')  
    @property  
    def job_offers(self):  
        """  
        Obtient le nombre d'offres d'emploi pour ce métier.  
  
        Returns:  
            int: Nombre d'offres d'emploi.  
        """  
        return self.data.get('labourMarket', {}).get('jobOffers', 0)  
  
    @property  
    def job_seekers(self):  
        """  
        Obtient le nombre de demandeurs d'emploi pour ce métier.  
  
        Returns:  
            int: Nombre de demandeurs d'emploi.  
        """  
        return self.data.get('labourMarket', {}).get('jobSeekers', 0)  
  
    @property  
    def metier(self):  
        """  
        Obtient les données spécifiques au métier.  
  
        Returns:  
            dict: Dictionnaire contenant les données spécifiques au métier.  
        """  
        return self.data.get('metier', {})  
  
    @property  
    def description(self):  
        """  
        Obtient la description du métier.  
  
        Returns:  
            str: Description du métier.  
        """  
        return self.metier.get('description', '')  
  
    @property  
    def access_text(self):  
        """  
        Obtient le texte d'accès au métier.  
  
        Returns:  
            str: Texte d'accès au métier.  
        """  
        return self.metier.get('accessText', '')  
  
    @property  
    def appellations(self):  
        """  
        Obtient la liste des appellations du métier.  
  
        Returns:  
            list: Liste des appellations du métier.  
        """  
        return [appellation.get('label', '') for appellation in self.appellations_list]  
  
    @property
```

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

```
def appellations_list(self):
    """
    Obtient la liste brute des appellations du métier.

    Returns:
        list: Liste brute des appellations.
    """
    return self.metier.get('appellations', [])

@property
def video_url(self):
    """
    Obtient l'URL de la vidéo associée au métier.

    Returns:
        str: URL de la vidéo.
    """
    return self.metier.get('videoUrl', '')

@property
def domain(self):
    """
    Obtient le domaine associé au métier.

    Returns:
        dict: Dictionnaire contenant les données du domaine.
    """
    return self.metier.get('domain', {})

@property
def domain_label(self):
    """
    Obtient le label du domaine associé au métier.

    Returns:
        str: Label du domaine.
    """
    return self.domain.get('label', '')

@property
def domain_url(self):
    """
    Obtient l'URL du label du domaine associé au métier.

    Returns:
        str: URL du label du domaine.
    """
    return self.domain.get('labelUrl', '')

@property
def domain_title(self):
    """
    Obtient le titre du domaine associé au métier.

    Returns:
        str: Titre du domaine.
    """
    return self.domain.get('title', '')

@property
def domain_description(self):
    """
```

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

```
    Obtient la description du domaine associé au métier.

    Returns:
        str: Description du domaine.
    """
    return self.domain.get('meta', '')

@property
def related_jobs(self):
    """
    Obtient la liste des métiers liés.

    Returns:
        list: Liste des métiers liés.
    """
    return [job.get('label', '') for job in self.related_jobs_list]

@property
def related_jobs_list(self):
    """
    Obtient la liste des métiers liés.

    Returns:
        list: Liste des métiers liés.
    """
    return self.data.get('relatedJobs', [])

@property
def skills(self):
    """
    Obtient les compétences associées au métier.

    Returns:
        dict: Dictionnaire des compétences.
    """
    return self.data.get('skills', {})

@property
def domains_skills(self):
    """
    Obtient les compétences par domaine.

    Returns:
        list: Liste des compétences par domaine.
    """
    return self.skills.get('domainsSkills', [])

@property
def knowledge_categories(self):
    """
    Obtient les catégories de connaissances.

    Returns:
        list: Liste des catégories de connaissances.
    """
    return self.skills.get('knowledgeCategories', [])

@property
def professional_soft_skills(self):
    """
    Obtient les compétences douces professionnelles.

    Returns:

```

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

```
        list: Liste des compétences douces professionnelles.  
    """  
    return self.skills.get('professionalSoftSkills', [])  
  
    @property  
    def work_conditions(self):  
        """  
        Obtient les conditions de travail.  
  
        Returns:  
            list: Liste des conditions de travail.  
        """  
        return [cond.get('label', '') for category in self.work_conditions_categories for cond in category.get('workConditions', [])]  
  
    @property  
    def work_conditions_categories_label(self):  
        """  
        Obtient les labels des catégories de conditions de travail.  
  
        Returns:  
            list: Liste des labels des catégories de conditions de travail.  
        """  
        return [category.get('label', '') for category in self.work_conditions_categories]  
  
    @property  
    def work_conditions_categories(self):  
        """  
        Obtient les catégories de conditions de travail.  
  
        Returns:  
            list: Liste des catégories de conditions de travail.  
        """  
        return self.data.get('workConditionsCategories', [])
```

## models/metiers.py

Voici le fichier models/metiers.py :

```
"""  
Auteur: Aram BABAYAN  
Date: 01/08/2023  
Version: 1.0  
"""  
  
import json  
import os  
from models.metier import Metier  
  
class Metiers:  
    """  
    Classe Metiers pour gérer une collection de métiers.  
  
    Attributs:  
        data_source (str): Chemin du fichier source contenant les données des métiers.  
        metiers_data (dict): Dictionnaire contenant les données brutes des métiers.  
        metiers (dict): Dictionnaire contenant les objets Metier.  
    """  
  
    def __init__(self, data_source):  
        """  
        Constructeur de la classe.  
    """
```

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

```
Args:  
    data_source (str): Chemin du fichier source contenant les données des métiers.  
"""  
    self.data_source = data_source  
    self.metiers_data = self.load_metiers()  
    self.metiers = {}  
    for key, data in self.metiers_data.items():  
        self.metiers[key] = Metier(data)  
  
def load_metiers(self):  
    """  
    Charge les données des métiers depuis le fichier source.  
  
    Returns:  
        dict: Dictionnaire contenant les données brutes des métiers.  
    """  
    file_path = os.path.abspath(__file__)  
    dir_path = os.path.dirname(file_path)  
    with open(dir_path + '/../' + self.data_source, 'r') as json_file:  
        return json.load(json_file)  
  
def get_all(self):  
    """  
    Obtient tous les métiers.  
  
    Returns:  
        dict: Dictionnaire contenant tous les objets Metier.  
    """  
    return self.metiers  
  
def get(self, id):  
    """  
    Obtient un métier par son ID.  
  
    Args:  
        id (str): ID du métier.  
  
    Returns:  
        Metier: Objet Metier correspondant à l'ID, ou None si non trouvé.  
    """  
    return self.metiers[id] if id in self.metiers else None  
  
def get_metier_by_label(self, label):  
    """  
    Obtient un métier par son label.  
  
    Args:  
        label (str): Label du métier.  
  
    Returns:  
        Metier: Objet Metier correspondant au label, ou None si non trouvé.  
    """  
    for metier in self.metiers:  
        if metier.label == label:  
            return metier  
    return None  
  
def get_metier_by_id(self, id):  
    """  
    Obtient un métier par son ID.
```

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

```
Args:  
    id (str): ID du métier.  
  
Returns:  
    Metier: Objet Metier correspondant à l'ID, ou None si non trouvé.  
"""  
    return self.metiers.get(id, None)  
  
def get_metiers_with_high_demand(self):  
    """  
    Obtient les métiers avec une forte demande.  
  
    Returns:  
        list: Liste des objets Metier avec une forte demande.  
    """  
    return [metier for metier in self.metiers if metier.job_offers > metier.job_seekers]  
  
def get_metiers_by_domain(self, domain_label):  
    """  
    Obtient les métiers par domaine.  
  
    Args:  
        domain_label (str): Label du domaine.  
  
    Returns:  
        list: Liste des objets Metier dans le domaine spécifié.  
    """  
    return [metier for metier in self.metiers if metier.domain_label == domain_label]  
  
def get_most_common_work_conditions(self, top_n=5):  
    """  
    Obtient les conditions de travail les plus courantes.  
  
    Args:  
        top_n (int): Nombre de conditions de travail les plus courantes à retourner.  
  
    Returns:  
        list: Liste des conditions de travail les plus courantes.  
    """  
    conditions_count = {}  
    for metier in self.metiers:  
        for condition in metier.work_conditions:  
            conditions_count[condition] = conditions_count.get(condition, 0) + 1  
  
    return sorted(conditions_count.items(), key=lambda x: x[1], reverse=True)[:top_n]  
  
def get_metiers_with_videos(self):  
    """  
    Obtient les métiers ayant une URL vidéo associée.  
  
    Returns:  
        list: Liste des objets Metier ayant une URL vidéo.  
    """  
    return [metier for metier in self.metiers if metier.video_url]
```

## L2 – Réseaux

Aram BABAYAN - ID : 20004019 - NNE : 0JYJQT02ZA4

### api/job\_offers\_api.py

Voici le fichier api/job\_offers\_api.py :

```
"""
Auteur: Aram BABAYAN
Date: 01/08/2023
Version: 1.0
"""

import requests

class JobOffersAPI:
    """
    Client API pour interroger le sites des offres d'emploi.

    Cette classe permet de récupérer les offres d'emploi à partir d'une API.
    Elle peut être étendue pour intégrer d'autres endpoints liés aux emplois.

    Dépendances:
        - requests: nécessaire pour effectuer des requêtes HTTP.

    Exemple d'utilisation:
        >>> api = JobOffersAPI()
        >>> offers = api.get_job_offers('M1805', 10)
        >>> print(offers)

    :param base_url: L'URL de base de l'API. Par défaut, il s'agit de
        'https://localhost:5000'.
    :type base_url: str
    """

    def __init__(self, base_url='https://candidat.pole-emploi.fr/gw-metierscope'):
        self.base_url = base_url

    def get_job_offers(self, job_id, max_offers):
        """
        Récupère les offres d'emploi pour un identifiant de job spécifié.

        :param job_id: L'identifiant du job pour lequel récupérer les offres.
        :type job_id: str

        :param max_offers: Le nombre maximal d'offres à récupérer.
        :type max_offers: int

        :return: Une liste des offres d'emploi.
        :rtype: list[dict]

        :raises requests.RequestException: Si la requête échoue.

    Exemple d'utilisation:
        >>> api = JobOffersAPI()
        >>> offers = api.get_job_offers('M1805', 10)
        >>> print(offers[0]['title'])  # Affiche le titre de la première offre
        """

        url = f'{self.base_url}/job/{job_id}/offers?max={max_offers}'
        response = requests.get(url)

        # Gérer les éventuelles erreurs
        response.raise_for_status()

        return response.json()
```

## VIII. Budget, temps alloué et coûts

Le projet a été réalisé en tant que projet personnel et éducatif, donc aucun budget n'a été alloué. Il a permis en revanche d'acquérir des connaissances et des compétences dans plusieurs domaines. Le temps alloué pour la recherche, le développement, l'évolution et la rédaction du rapport était d'environ 1 mois et demi.

## IX. Pistes d'amélioration

Voici quelques perspectives d'évolution du projet que je me suis noté pour ce projet :

- **Optimisation des Performances**

Un point essentiel à aborder est le temps de démarrage de l'application. Actuellement, selon la configuration de la machine, ce démarrage peut prendre même plusieurs minutes... C'est causé par le grand nombre de données à traiter sous forme de patterns « lemmatisés » par la librairie Spacy.

Il serait donc bénéfique d'investir du temps et des ressources pour peaufiner et optimiser le code, réduire la charge lors du démarrage, et éventuellement envisager de revoir certaines architectures ou mécanismes sous-jacents pour accélérer ce processus.

- **Métadonnées plus détaillées sur les métiers**

Intégrer des données plus détaillées comme le salaire moyen pour chaque poste. Cela pourrait permettre de cibler les offres d'emploi par ville, par salaire, etc...

- **Top des métiers / métiers populaires**

Afficher les métiers qui sont les plus en demande actuellement (à partir des données du marché de travail)

- **Améliorer la reconnaissance des entités nommées**

Actuellement, les seules entités nommées actuellement sont les codes métiers utilisés pour identifier le métier en fonction du pattern et de la saisie nettoyée et lemmatisée. Il faudrait entraîner son propre modèle avec des données spécifiques au domaine dans des contextes différents pour améliorer la précision de la reconnaissance des entités.

- **Améliorer le traitement automatique des langues**

L'objectif serait d'adopter des techniques plus sophistiquées pour améliorer la précision des suggestions de métiers. Cela pourrait inclure :

- **Annotation manuelle des données**

Pour améliorer la qualité des données d'entraînement du modèle.

- **Nettoyage avancé des données**

Optimiser les techniques de nettoyage de données (de saisie et de patterns) plus avancées pour éliminer les bruits et les incohérences.

- **Algorithmes de Similarité**

Au lieu d'utiliser un pattern matching exact, un algorithme pourrait être employé pour comparer la similarité entre la saisie de l'utilisateur et les métiers disponibles, permettant ainsi des correspondances plus pertinentes.

## X. Conclusion

La réalisation du projet GuideMetier m'a permis d'approfondir mes connaissances dans divers domaines tels que le traitement du langage naturel, le développement d'API REST, et le scraping de données. Chaque étape du projet a été une opportunité d'apprentissage, me permettant de mieux comprendre comment intégrer différentes technologies pour créer une solution complète.

Il m'a également offert l'opportunité d'explorer des problématiques liées au développement d'applications interactives, un sujet qui me passionne. Le défi de créer une interface utilisateur intuitive et réactive, tout en assurant une communication fluide avec le backend, a été particulièrement enrichissant.

Enfin, il m'a donné l'occasion de travailler sur un projet complet et intéressant de A à Z. Ce n'était pas simplement une série de tâches isolées, mais un processus intégré nécessitant une planification, une exécution et une résolution de problèmes efficaces. Les défis rencontrés en cours de route m'ont appris à évaluer les différentes solutions possibles, tenter une approche, échouer, puis réessayer avec une autre méthode jusqu'à trouver celle qui s'aligne le mieux avec les objectifs du projet.

Dans l'ensemble, GuideMetier représente une étape significative dans mon parcours professionnel, consolidant mes compétences techniques tout en élargissant ma compréhension des complexités inhérentes au développement de systèmes interactifs et intelligents.

## XI. Sources

Données issues du site de Pôle emploi

<https://candidat.pole-emploi.fr/metierscope/>

Inspiration UI a partir de la maquette

<https://dribbble.com/shots/22026872-Chatbot-Web-UI>

API backend (Flask)

<https://openclassrooms.com/fr/courses/4525361-realisez-un-dashboard-avec-tableau/5774811-creez-une-api-avec-flask>

Nettoyage de la saisie / patterns

<https://blog.octo.com/nettoyage-du-texte-en-nlp-moins-de-vocabulaire-moins-de-bruit/>

Pattern-matching et entraînement de modèle NER (reconnaissance des entités nommées)

<https://fr.shapip.com/blog/named-entity-recognition-and-its-types/>

<https://spacy.io/usage/rule-based-matching#entityruler>

[https://github.com/explosion/spaCy/blob/v2.3.x/examples/training/train\\_ner.py](https://github.com/explosion/spaCy/blob/v2.3.x/examples/training/train_ner.py)

<https://www.kaggle.com/datasets/dataturks/resume-entities-for-ner>

Appels requests pour l'API scrawler

<https://pypi.org/project/requests/>