Projet chef d’œuvre

Détection des troubles de la santé mentale et du besoin de traitement avec l'IA chez les travailleurs du domaine de la technologie.

Geoffroy Daumer

Table des matières

[1. Introduction du projet 3](#_Toc161958602)

[2. Analyse de la demande 4](#_Toc161958603)

[2.1 Enjeux réglementaires pour le traitement des données de l'application 4](#_Toc161958604)

[2.2 Les utilisateurs du projet 4](#_Toc161958605)

[2.4 Gestion du projet 4](#_Toc161958606)

[3. Gestion des données analytiques du projet 6](#_Toc161958607)

[3.1 Analyse exploratoire des données 6](#_Toc161958608)

[3.2 Mise en place d'une base de donnée analytique 15](#_Toc161958609)

[4. Etat de l'art 16](#_Toc161958610)

[5. Choix techniques liés au projet 17](#_Toc161958611)

[5.1 Partie IA 17](#_Toc161958612)

[5.2 Partie application 18](#_Toc161958613)

[6. L'organisation technique et l'environnement de développement tout au long de la production 22](#_Toc161958614)

[7. Bilan du projet et améliorations envisageables 23](#_Toc161958615)

[8. Conclusion 24](#_Toc161958616)

[Références 25](#_Toc161958617)

[Annexes 26](#_Toc161958618)

# 1. Introduction du projet

Nous vivons dans une société où les cas de problèmes de santé mentale sont en forte croissance, et ils restent à ce jour mal soignés.

Dans le cas des travailleurs du domaine de la technologie, on remarque qu'ils sont souvent assis, derrière des écrans. Ce manque d'activité physique, voir sociale, peut être un facteur d'un bien être diminuant.

Avec l'IA, on pourrait détecter plus facilement les troubles de la santé mentale, et ainsi les prendre en charge plus rapidement.

Un client m'a contacté pour développer une application qui intégrerait l'IA, cette application permettrait de détecter le besoin de traitement pour des problèmes de santé mentale chez les travailleurs du domaine de la technologie.

Ce client a mené une enquête auprès des travailleurs, leur posant un certain nombre de questions, à choix multiples, et recueillant les résultats sous forme d'un jeu de données.

Parmi les questions posées, une question "Avez-vous recherché un traitement pour des problèmes de santé mentale ?", c'est cette question qui servira de cible à l'algorithme d'IA, pour l'entraînement du modèle de prédiction.

La solution finale apportée est une application sur laquelle on peut se connecter et répondre à un questionnaire, un algorithme d'IA prédira si nous devrions considérer le fait de prendre un traitement pour la santé mentale.

# 2. Analyse de la demande

### 2.1 Enjeux réglementaires pour le traitement des données de l'application

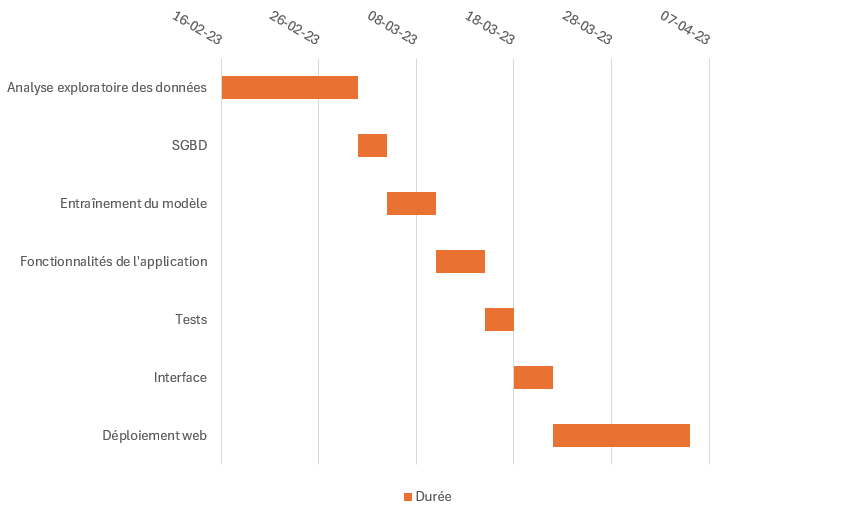
Il n'y a pas de règlementations particulières concernant l'utilisation de ces données, elles sont en libre accès sur Kaggle (1).

### 2.2 Les utilisateurs du projet

L'application est destinée à être utilisée par les travailleurs du domaine de la technologie, afin qu'ils aient un aperçu de leur état de santé mentale.

### 2.4 Gestion du projet

Voici le diagramme de Gantt de suivi de réalisation du projet :



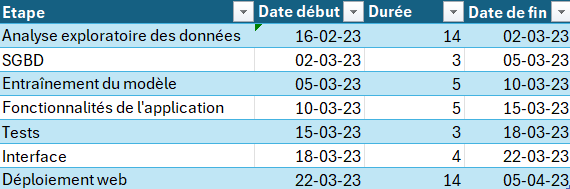


Figure 1 : Diagramme de Gantt, avec le tableau des périodes

Pour faire un suivi de l'avancement du projet, nous avons fait trois réunion.

- Une réunion pour présenter l'analyse exploratoire des données, et les méthodes de nettoyage choisies

- Une réunion pour présenter l'entraînement du modèle, et les résultats obtenus

- Une réunion pour présenter l'application fonctionnelle

Outre les réunions, nous communiquions par mails et appels quand c'était nécessaire.

Voir Annexe 1 : Rapports d'avancement pour plus de détails

# 3. Gestion des données analytiques du projet

### 3.1 Analyse exploratoire des données

#### 3.1.1 Présentation générale

Une image contenant capture d’écran, texte, noir

Description générée automatiquementVoici un échantillon du jeu de données :

Figure 2 : Echantillon du jeu de données

Le jeu de données comprend 1259 lignes et 27 colonnes, il y a donc 1259 questionnaires remplis.

Dans le jeu, il y a une colonne "Timestamp" (utilité inconnue), "Age" (la seule caractéristique continue), "Genre", "Country", "state" (pour les États américains), "comments" (pour des commentaires optionnels ).

Le reste des colonnes estdes questions à choix multiples, allant de 2 à 5 choix.

La colonne qui demande si le patient a déjà demandé un traitement pour la santé mentale s'appelle "treatment". Comme dit dans l'introduction, cette colonne représentera la sortie à prédire par le modèle d'IA.

Voici la liste des questions en français, par rapport aux colonnes du jeu de données :

self\_employed : Êtes-vous travailleur indépendant ?

family\_history : Avez-vous des antécédents de problèmes de santé mentale dans la famille ?

work\_interfere : Si vous avez un problème de santé mentale, sentez-vous que cela interfère avec votre travail ?

no\_employees : Combien d'employés votre entreprise ou organisation possède-t-elle ?

remote\_work : Travaillez-vous à distance (en dehors du bureau) au moins 50% du temps ?

tech\_company : Votre employeur est-il principalement une entreprise ou organisation technologique ?

benefits : Votre employeur offre-t-il des prestations en matière de santé mentale aux employés ?

care\_options : Connaissez-vous ces prestations ?

wellness\_program : Votre employeur a-t-il déjà abordé la question de la santé mentale dans le cadre d'un programme de bien-être des employés ?

seek\_help : Votre employeur met-il à votre disposition des ressources permettant d'en savoir plus sur les problèmes de santé mentale et sur la manière de demander de l'aide ?

anonymity : Votre anonymat est-il protégé si vous choisissez de profiter des ressources de traitement de la santé mentale ou de la toxicomanie ?

leave : Est-il facile pour vous de prendre un congé médical pour un problème de santé mentale ?

mental\_health\_consequence : Pensez-vous que le fait de parler d'un problème de santé mentale à votre employeur aurait des conséquences négatives ?

phys\_health\_consequence : Pensez-vous que le fait de parler d'un problème de santé physique à votre employeur aurait des conséquences négatives ?

coworkers : Seriez-vous prêt à discuter d'un problème de santé mentale avec vos collègues de travail ?

supervisor : Seriez-vous prêt à discuter d'un problème de santé mentale avec votre (vos) supérieur(s) hiérarchique(s) ?

mental\_health\_interview : Parleriez-vous d'un problème de santé mentale à un employeur potentiel lors d'un entretien ?

phys\_health\_interview : Parleriez-vous d'un problème de santé physique à un employeur potentiel lors d'un entretien ?

mental\_vs\_physical : Pensez-vous que votre employeur prend autant au sérieux la santé mentale que la santé physique ?

obs\_consequence : Avez-vous entendu parler ou observé des conséquences négatives pour les collègues souffrant de troubles mentaux sur votre lieu de travail ?

#### 3.1.2 Gestion des valeurs manquantes



Figure 3 : Valeurs manquantes du jeu de données

Les colonnes "state" et "comments" seront supprimées.

"self\_employed" signifie "êtes-vous travailleur indépendant ?", les valeurs manquantes ont été remplacées par une nouvelle catégorie : "Other", il y avait "Yes" et "No".

"work\_interfere" signifie "si vous avez déjà eu des problèmes de santé mentale, avez-vous senti que cela interférait avec votre travail ?", les valeurs manquantes ont été remplacées par "Not concerned".

#### 3.1.3 Colonnes supprimées

Liste des colonnes supprimées :

- "Country" et "state" : les pays étaient trop nombreux et trop disproportionnés.

- "Timestamp" : pas de sens connu à la colonne, les dates sont presque toutes les mêmes.

- "comments" : intégrer une section de commentaire pour la prédiction de l'algorithme serait trop difficile à gérer, et en plus il y a une grande majorité de valeurs manquantes.

#### 3.1.4 Ingénierie de colonnes

La colonne Age possédait des valeurs aberrantes (ex : 99999999), tout questionnaires avec des âges n'étant pas entre 0 et 100 ont été supprimés.

La colonne Genre possédait des dizaines de genres différents, Les dénominations définissant les hommes et les femmes ont été regroupées dans leur genre respectif, le reste a été regroupé sous une étiquette "Other".

#### 3.1.5 Visualisations

Répartition de la colonne "treatment" (cible du modèle d'IA) :

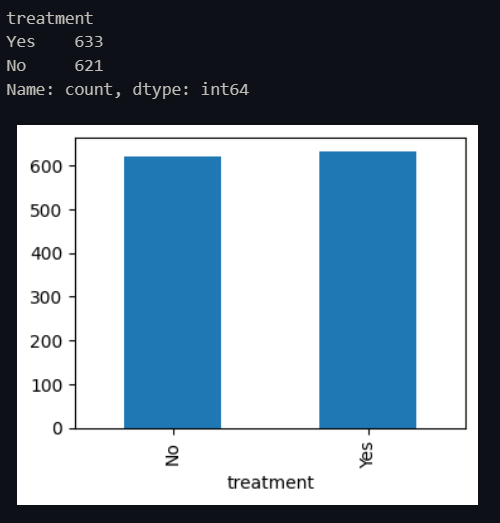


Figure 4 : Répartition des valeurs de la colonne "treatment"

La colonne "treatment" est répartie de manière équilibrée, c'est parfait pour entraîner un modèle.

Le prochain graphique représente la corrélation de la colonne de sortie (treatment) avec les autres colonnes :

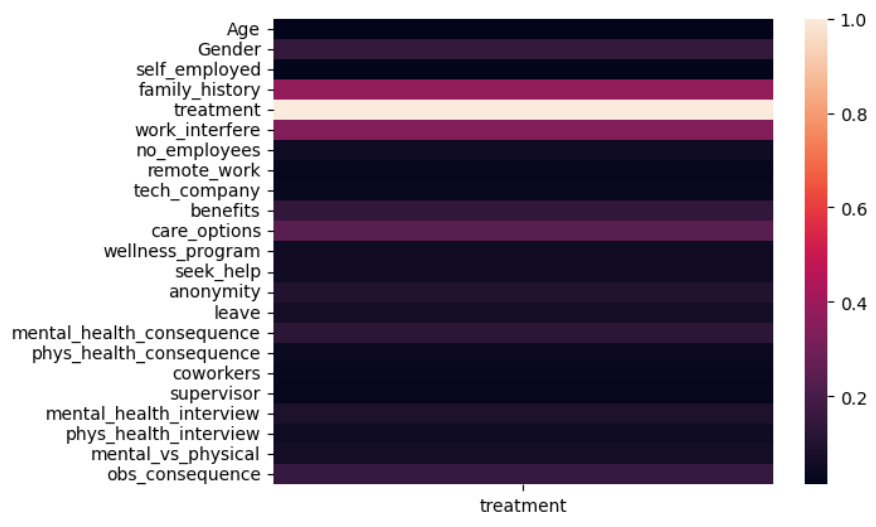


Figure 5 : Corrélation entre la colonne traitement et les autres colonnes

On voit sur la figure 6 les colonnes qui sont corrélées le plus par rapport au fait d'avoir fait la demande d'un traitement. Nous analyserons dans la section suivante, pour chaque colonne qui obtient une forte corrélation, un diagramme de la repartition des valeurs de cette colonne par rapport au fait d'avoir fait une demande de traitement.

family\_history :

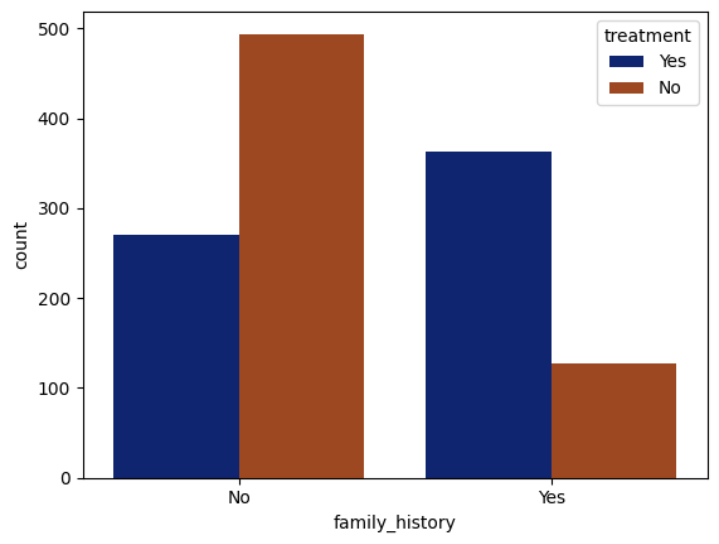


Figure 6 : Corrélation entre antécédents familiaux et demande de traitement

On constate que les personnes ayant des antécédents familiaux ont plus de chance d'avoir déjà fait une demande de traitement.

work\_interfere :

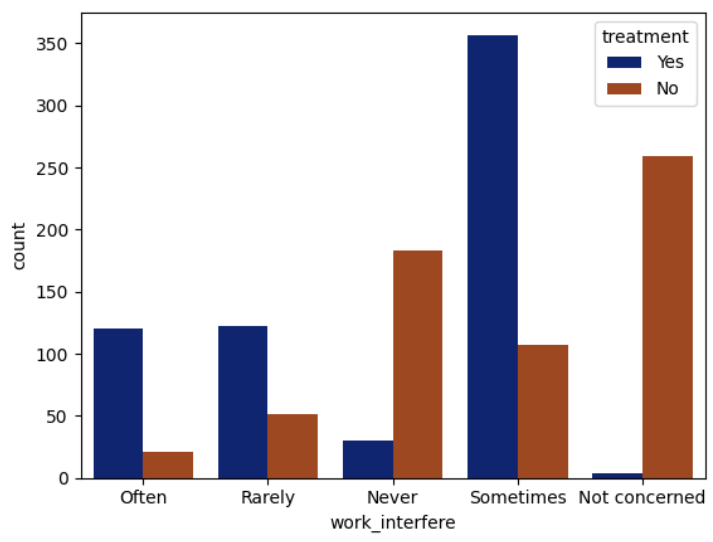


Figure 7 : Corrélation entre la colonne "work\_inference" et "traitement"

Les personnes qui disent qu'elles ont un problème de santé mentale et qu'il interfère avec le travail ("work\_interfere") sont corrélées avec la colonne traitement, c'est logique puisqu'elles affirment avoir un problème de santé mentale.

Plus la fréquence d'interfération du problème mental avec le travail est grande, plus il y a de chance pour que la personne ait déjà demandé un traitement.

care\_options :

"care\_options", c'est le fait que les employés connaissent ou non les prestations de leur entreprise concernant la santé mentale des employés.

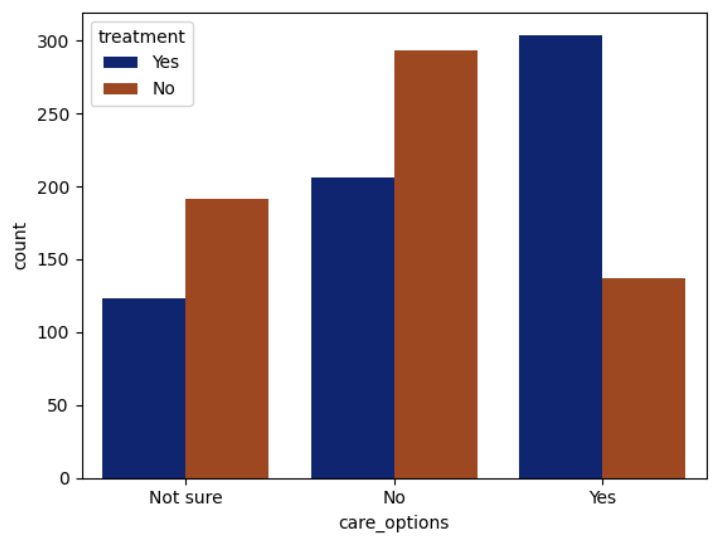


Figure 8 : Corrélation entre care\_options et treatment

On remarque sur la figure 9 que les personnes qui connaissent les prestations de l'entreprise en matière de santé mentale pour les salariés ont plus souvent fait une demande de traitement.

obs\_consequence :

" Avez-vous entendu parler ou observé des conséquences négatives pour les collègues souffrant de troubles mentaux sur votre lieu de travail ?"

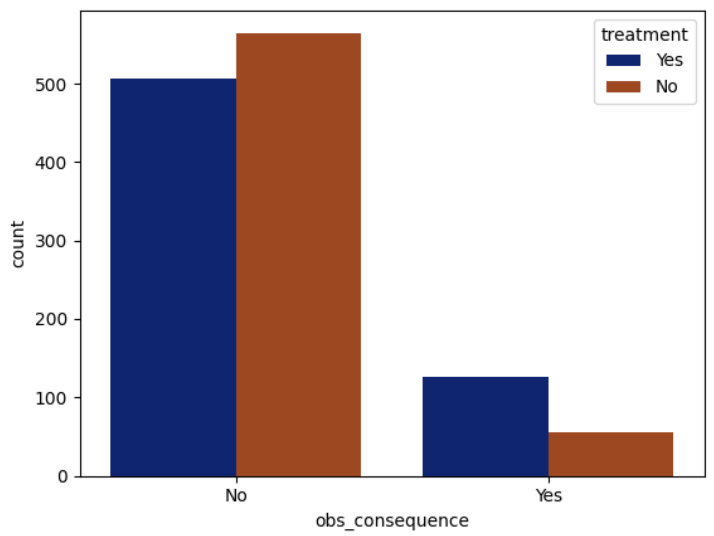


Figure 9 : Corrélation entre obs\_consequence et treatment

Les gens qui répondent oui ont plus souvent fait une demande de traitement.

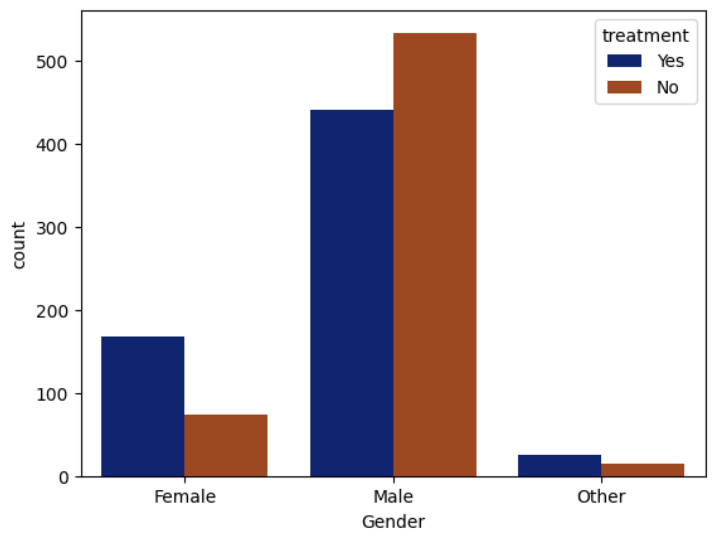
Genre :

Figure 10 : Corrélation entre le genre et traitement

Les femmes sont plus sujettes à la demande de traitement.

benefits :

Votre employeur offre-t-il des prestations en matière de santé mentale aux employés ?

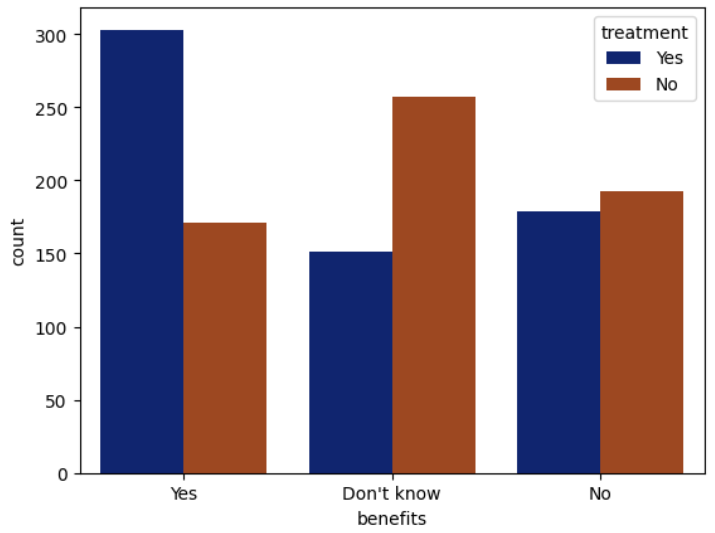


Figure 11 : Corrélation entre benefits et traitement

Les gens qui répondent oui ont plus souvent fait la demande d'un traitement, ceux qui ne savent pas n'en ont plus souvent pas fait.

mental\_health\_consequence :

Pensez-vous que le fait de parler d'un problème de santé mentale à votre employeur aurait des conséquences négatives ?

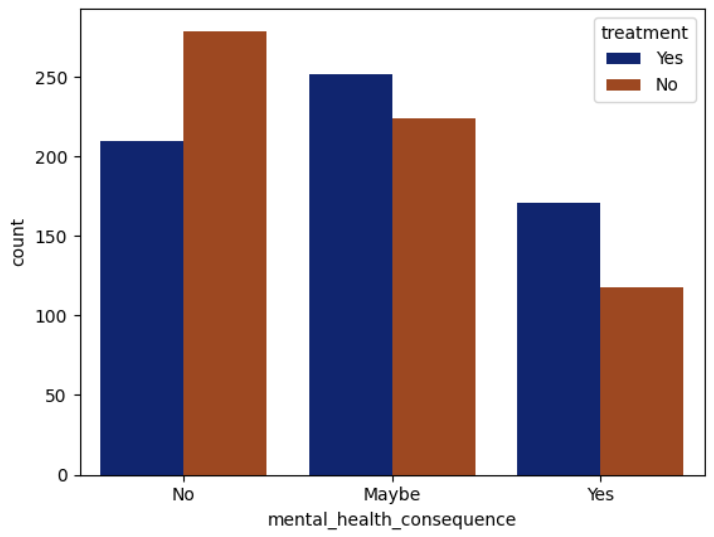


Figure 12 : Corrélation entre mental\_health\_consequence et traitement

Les gens qui pensent que le fait de discuter d'un problème de santé mentale avec leur employeur aurait des conséquences négatives ont plus souvent demandé un traitement. Pour ceux qui répondent non, ils ont moins souvent demandé un traitement.

### 3.2 Mise en place d'une base de donnée analytique

La stratégie de nettoyage des données décrite plus haut est rassemblée dans un script python.

Une fois les données nettoyées, elles sont enregistrées dans une base de données SQLite, dans une table "questionnaires", cela permet de réutiliser les données sans avoir à les nettoyer à chaque fois.

Voir les détails de la mise en place de la base de donnée dans Annexe 1

# 4. Etat de l'art

Pour réaliser la veille sur les algorithmes à utiliser, il faut chercher du côté des modèles de classification, car la cible est une catégorie : besoin d'un traitement ou non.

De plus, on sait que notre jeu de données se compose essentiellement de variables catégoriques.

Voici une liste des modèles de classification (2) (3) (4):

**La machine à vecteurs de support (SVM)** : Classificateur linéaire, sépare les données à travers des lignes (hyperplans). Fonctionne bien pour les identifier des classes simples (exemple deux sorties).

**Les arbres de décision** : Algorithme classifiant les données sous forme de branches. On part d'une racine, et chaque échantillon prend une direction en fonction de ses caractéristiques. Ce qui permet de prédire une variable de réponse.

**La répartition en K-moyennes (K-means)** : Trie les données en différents groupes en fonction de leurs caractéristiques. Etablit une moyenne de référence dans chaque groupe pour définir un profil type. Bonne précision

**Le KNN (K-nearest neighbors)** : Classifie les variables d'un jeu de données en analysant les similitudes entre elles. Utilise un graphique et calcule la différence entre les différents points, l'échantillon est enregistré à la même catégorie que ses n plus proches voisins.

**La régression logistique** : Effectue des corrélations simples entre les entrées et les sorties, pour un nombre fini de résultats.

**Les réseaux de neurones** (5): Modèle de classification très puissant inspiré par le fonctionnement du cerveau humain. Composés de plusieurs couches de neurones interconnectés.

Un réseau est composé d'une couche d'entrée, de couches cachées et d'une couche de sortie.

# 5. Choix techniques liés au projet

Voici le schéma technique général de l'application :

Une image contenant texte, capture d’écran, diagramme, Police

Description générée automatiquement

Figure 13 : Schéma fonctionnel utilisé pour le projet

### 5.1 Partie IA

J'ai entrainé le modèle d'IA à partir de la librairie python "Scikit-learn", elle regroupe l'essentiel des modèles de machine learning. Les modèles ont été entraînés sur une interface jupyter notebook.

Pour le prétraitement des données, j'ai standardisé les colonnes continues (l'âge), et j'ai encodé en "One Hot" les colonnes discrètes (les autres questions).

Plusieurs modèles ont été comparés :

- LogisticRegression

- KNeighborsClassifier

- DecisionTreeClassifier

- AdaBoostClassifier

- GradientBoostingClassifier

- RandomForestClassifier

Pour déterminer le meilleur modèle, la précision, le recall et le f1 score des modèles ont été calculés.



Figure 14 : Résultats

Le modèle qui a obtenu les meilleures performances est la régression logistiques, avec un f1 score de 0.83. Le modèle final est un pipeline qui combine la transformation des colonnes et le modèle.

### 

### 

Précision = vrais positifs /

(vrais positifs + faux positifs)

Recall = vrais positifs /

(vrais positifs + faux négatifs)

*Figure 25 : Matrice de confusion*

Pour augmenter la certitude lorsqu’on prédit quelqu’un malade, il faut favoriser la précision.

Si on veut détecter plus facilement les positifs, il faut favoriser le rappel.

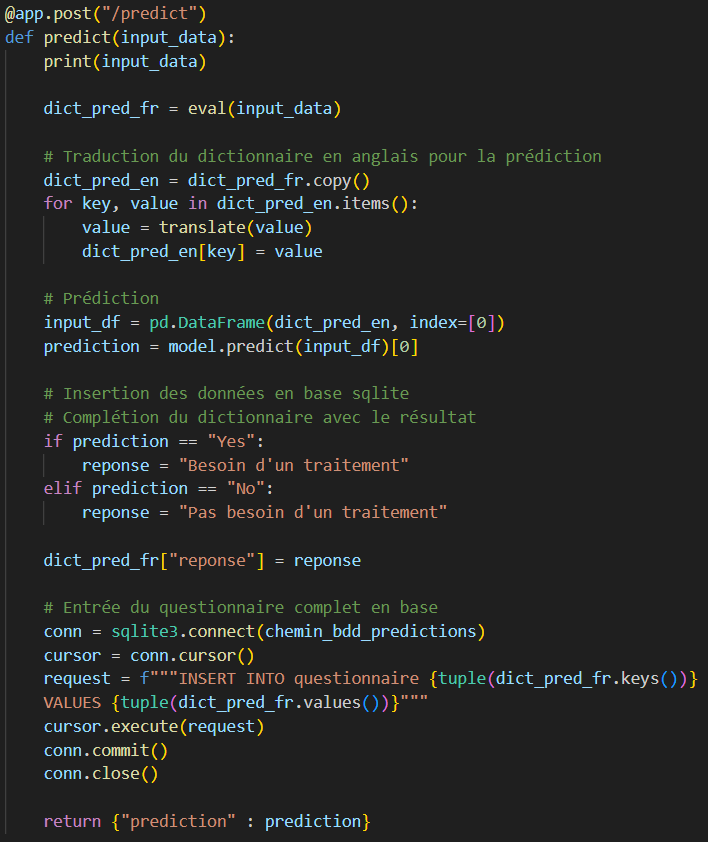
On veut favoriser la précision.

### 5.2 Partie application

L’analyse des besoins back, front et base de données de l’application est détaillé dans l’annexe 3.

#### 5.2.1 Back-end

J'ai utilisé une API FastAPI pour le développement des fonctionnalités de l'application.

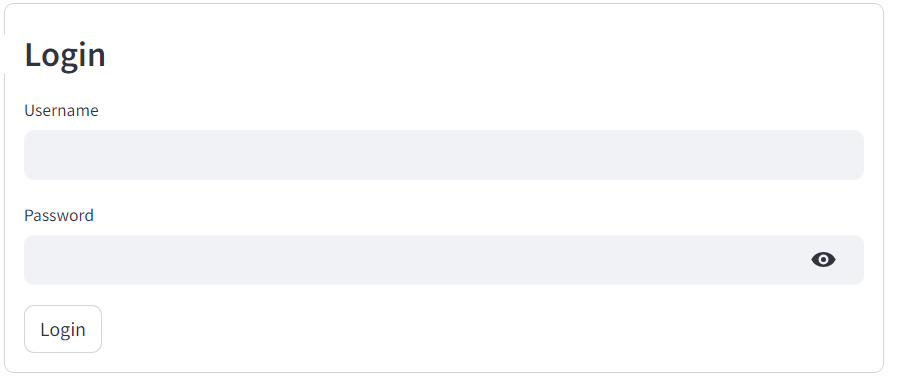


*Figure 15 : Fonctionnalité de prédiction de l’API*

#### 5.2.2 Front-end

Pour réaliser ce projet, j'ai choisi de réaliser la partie front de l'application avec la librairie streamlit.

Le front contient une page de connexion, et l’application.



*Figure 16 : Page de connexion*

Une image contenant texte, capture d’écran, Police

Description générée automatiquement

*Figure 17 : Haut de page application*

*Figure 18 : Bas de page application*

#### 5.2.3 Base de données de l’application

J’ai utilisé une base de données Sqlite car très simple d’utilisation, et j’avais des relations entre les tables donc choix d’une base de données relationnelle.

Les questionnaires et leur réponses sont enregistrés dans une table questionnaires.

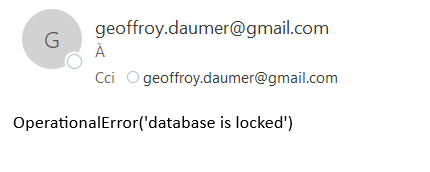
Dans cette table, il y a une clé étrangère qui fait une relation avec une table Utilisateur (annexe 3), ainsi, on sait qui a rempli chaque questionnaire.

Chaque fois qu’une prédiction est effectuée sur un questionnaire, le questionnaire est enregistré en base. Il y a donc une base de donnée pour l'entraînement du modèle (Partie 3.2), et une autre pour l’application.

La mise en place de la base de données est détaillées dans l’annexe 4

#### 5.2.4 Tracking

Pour le tracking des performances, étant en local, j'ai fait un script python qui se lance en même temps que l'application (annexe 5). Le script envoie des requêtes toutes les 4 secondes à l'API et écrit sur le terminal la bonne éxécution des requêtes, s'il y a une erreur, on voit l'erreur en direct et je recevrais un mail.

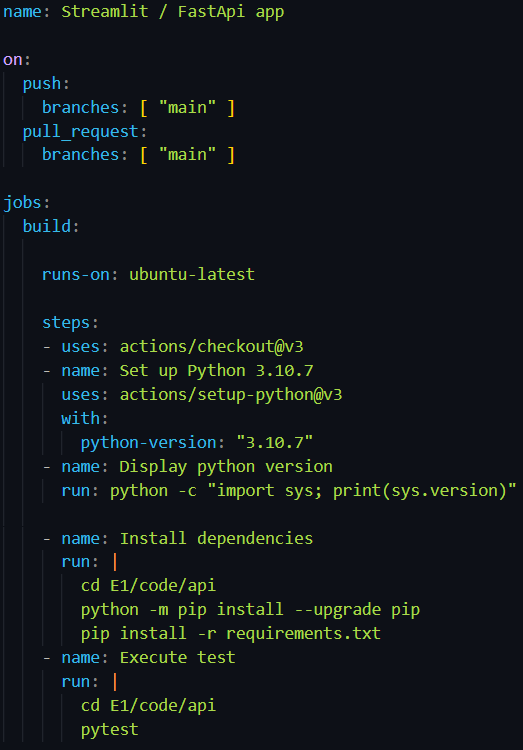


*Figure 19 : Exemple de mails reçus avec une erreur*

#### 5.2.5 Automatisation des tests

A la racine de mon répertoire github, j'ai un dossier .github/workflow avec un fichier .yml, qui éxécute des instructions avec github action, à chaque push et pull.

Les instructions comprennent les tests, c'est-à-dire la commande pytest, qui éxécute automatiquement les scripts de test de l'API.



*Figure 20 : Fichier .yml pour Github actions*

#### 5.2.6 Lancement de l'application

Mon application se lance avec l'éxécution d'un script .bat, qui lance trois terminaux, un pour l'API, un pour l'application streamlit, un pour le tracking.

Une image contenant texte, capture d’écran, Police

Description générée automatiquementJe n'ai pas déployé mon appli sur internet parce que j'ai des problèmes de connexion avec Heroku, pour les autres plateformes de déploiement, je n'ai rien trouvé de simple et gratuit d'utilisation.

Figure 21 : Terminaux ouverts au lancement de l'application

Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

Description générée automatiquement

Tracker de performances

Back

Front

Figure 22 : Script .bat de lancement

Le script .bat active un environnement virtuel, pour lancer l'API et le tracking. Ensuite, il active un autre environnement virtuel pour lancer le front.

# 6. L'organisation technique et l'environnement de développement tout au long de la production

Après avoir développé mon API, il fallait que j'y intègre une base de données pour les compétences de la certification. J'ai donc regardé ce qui se faisait en ligne, je me suis rapidement aperçu que j'étais bloqué parce que c'était payant (Google Cloud Plateform notamment). Nous avions des accès gratuits à Heroku, mais dans mon cas il y avait un problème avec la double authentification.

Je me suis finalement résigné à déployer en local.

J'ai eu des problèmes de compatibilité avec Docker et Windows, des packages d'environnements virtuels spécifiques à Windows (Jupyter notebook) faisaient buguer Docker. Je me suis dit que j'allais avoir pleins de problèmes si je continuais avec Windows, j'ai donc décidé de muter mon environnement de développement sur WSL.

J'ai en même temps laissé tomber Docker, ayant abandonné l'idée d'un déploiement web.

J'ai installé la version GNU/Linux Debian, et j'ai eu des problèmes ingérables de python, sql, il y avait des bugs très farfelus, la seule solution fut de quitter WSL.

# 7. Bilan du projet et améliorations envisageables

Pour un cas d'utilisation concret, nous avons vu que toutes les questions ne sont pas importantes pour la détermination du besoin de traitement. Un questionnaire plus pertinent pourrait être utilisé pour encore améliorer l'application. Cela impliquerait une vraie investigation, aussi bien dans les connaissances en santé mentale et en psychologie, que pour faire remplir un nombre suffisant de questionnaires. Le champ des possibles est illimité.

Une amélioration envisageable plus accessible pourrait être l'implémentation du niveau de certitude dans la prédiction de l'état de santé mentale. Ainsi l'utilisateur saurait si l'algorithme est sûr ou pas sûr

Je pense aussi que cette application est une bonne base pour développer de nouveaux outils pour prendre en charge la santé mentale des gens, comme avec des chatbots.

# Références

1. **Dataset.** [En ligne] https://www.kaggle.com/datasets/osmi/mental-health-in-tech-survey/data.

2. ***https://datascientest.com/algorithme-de-classification-definition-et-principaux-modeles.***

*3.* ***Machine\_learnia.*** *[En ligne] https://www.youtube.com/watch?v=EUD07IiviJg&list=PLO\_fdPEVlfKqUF5BPKjGSh7aV9aBshrpY.*

*4.* ***Statquest.*** *[En ligne] https://www.youtube.com/watch?v=\_L39rN6gz7Y&t=813s.*

*5.* ***ChatGPT.*** *[En ligne] https://chat.openai.com/.*

# Annexes

Tout le code du projet se trouve dans un répertoire Github : https://github.com/gdaume24/Projet-chef-oeuvre

[Annexe 1 : Détails mise en place base de données analytique 28](#_Toc164038323)

[Annexe 2 : Rapports d'avancement 30](#_Toc164038324)

[Annexe 3 : Analyse du besoin back, front et base de donnée de l’application 31](#_Toc164038325)

[Annexe 4 : Mise en place de la base de données de l'application 33](#_Toc164038326)

[Annexe 5 : Script de tracking des performances de l'application 34](#_Toc164038327)

Annexe 1 : Détails mise en place base de données analytique

Pour mettre en place la base de données, j'ai commencé par faire un modèle conceptuel des données (MCD) / modèle relationnel des données (MLD) sur Jmerise. Voici le MLD, c'est sensiblement la même chose que le MCD mais écrit légèrement différemment.

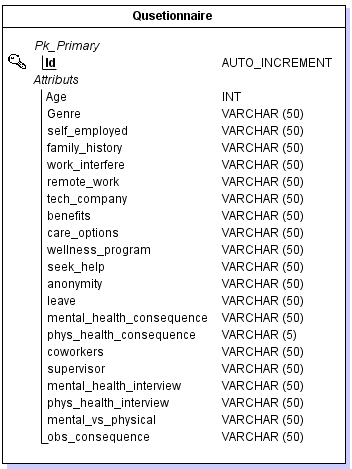


Figure 23 : Modèle relationnel des données de la base de données analytique

J'ai opté pour une base de données relationnelle, SQLite. Nous ne sommes pas dans un cas où nous avons d'énorme volume de données, ni de documents complexes, type vidéos, à gérer, auquel cas nous aurions dû choisir une base de données non relationnelle.

Procédure de mise en place de la BDD analytique :

- Plan MCD et MLD.

- Téléchargement de SQLite

- Installation

- Ajout du chemin du dossier dans ma variable d'environnement PATH

- Test de lancement de l'application dans le terminal

- Création d'une database SQLite en python :

con = sqlite3.connect("db.db")

- Import du jeu de données :

* Chargement du fichier csv en jeu de données

df = pd.read\_csv("..\data\survey.csv")

* Nettoyage du jeu de données avec une fonction personnalisée

df = nettoyage\_df(df)

* Import du jeu de données dans la base :

df.to\_sql('questionnaires', con=engine, if\_exists='fail')

Annexe 2 : Rapports d'avancement

Les réunions se sont déroulées les jeudis, à 14 heures, avec le Pr René Gallimard et Pr Hervé Ducroc.

Rapport 1, 29 février

Présentation du fichier d'analyse exploratoire des données :

- Stratégie adoptée pour les valeurs manquantes, colonnes inutiles, lignes supprimées, simplification de la colonne "Genre".

- Graphiques de représentation des données et interprétation

- Résumé des stratégie adaptées (script python)

Nous nous sommes mis d'accord sur les stratégies adoptées de nettoyage de données, nous avons exploré ensemble les résultats de l'analyse exploratoire des données.

Rapport 2, 7 mars

Présentation de l'entraînement du modèle et des résultats :

- Vulgarisation du machine learning

- Présentation des résultats

- Présentation d'un test

Lors de l'analyse des résultats, les professeurs ont cherché à savoir le fonctionnement interne du meilleur modèle, à savoir régression logistique, je me suis donc informé et leur ai présenté son fonctionnement.

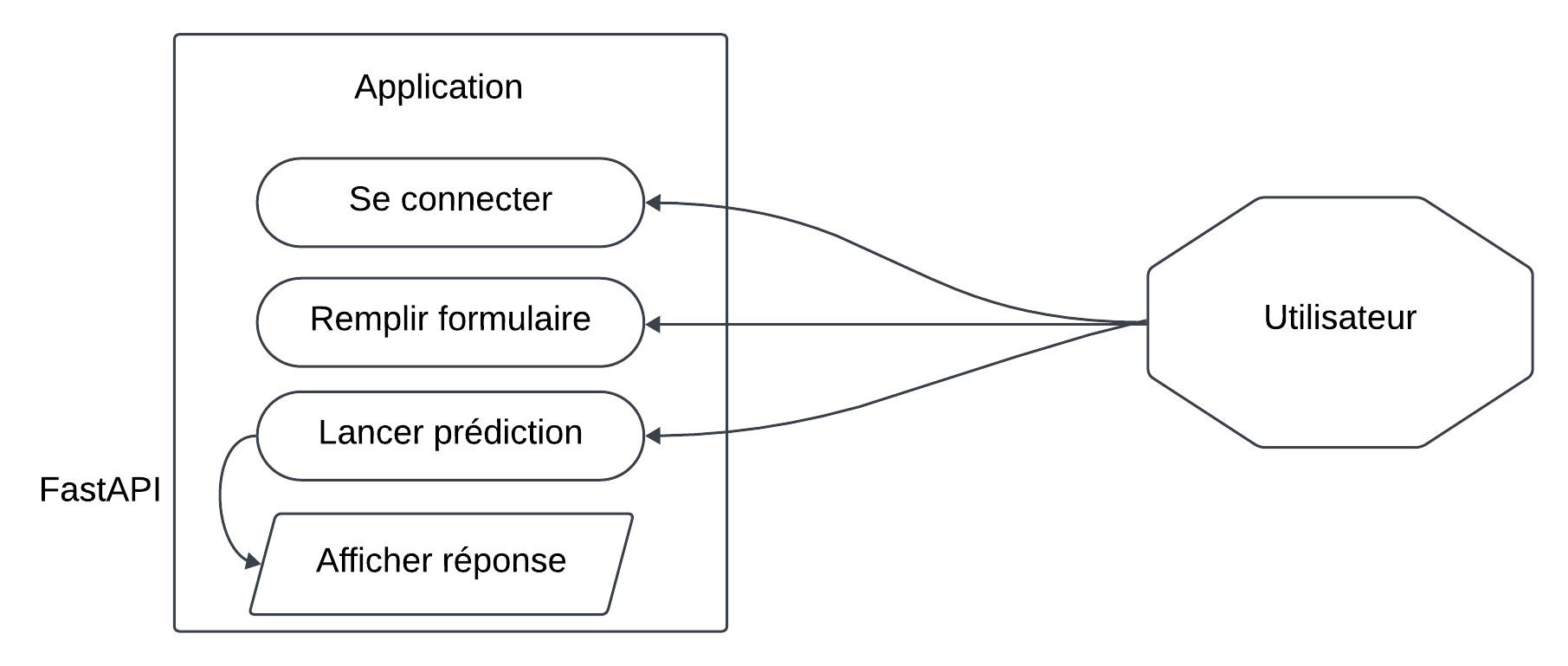
Rapport 3, 28 mars

Présentation de l'application fonctionnelle :

Nous avons vu ensemble si le produit fini était conforme aux attentes, nous avons discuté du fait d'intégrer un niveau de certitude dans les prédictions.

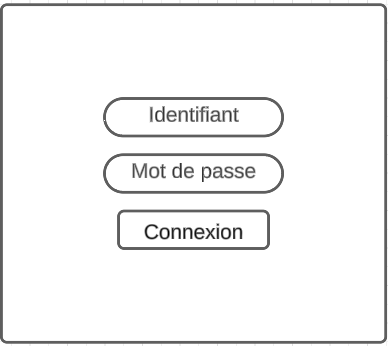
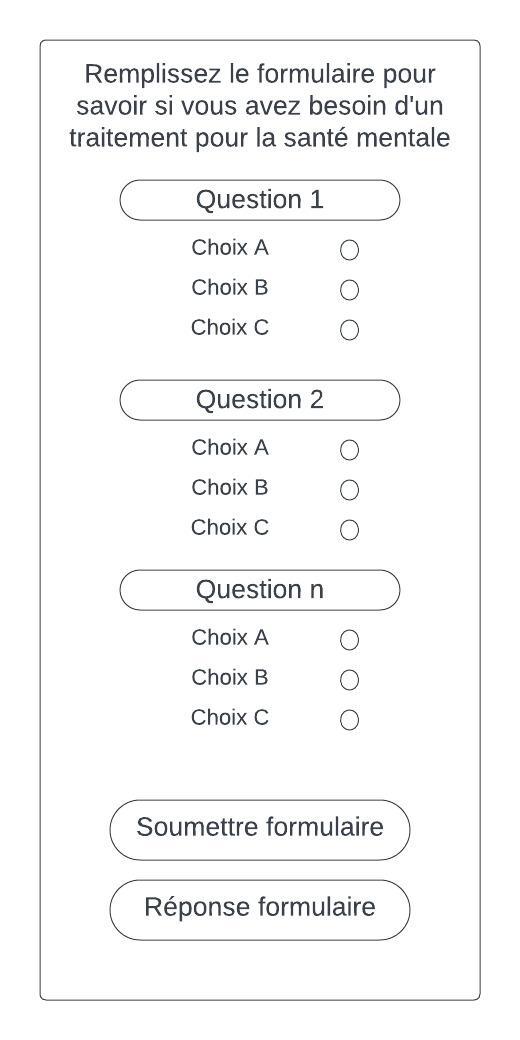
Annexe 3 : Analyse du besoin back, front et base de donnée de l’application

Analyse du besoin des fonctionnalités de l’application



*Figure 24 : Formalisation des fonctionnalités de l'application*

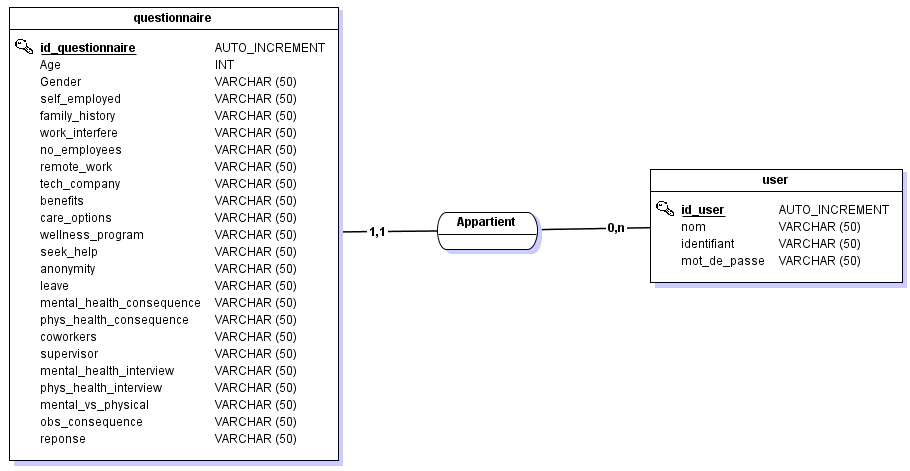
Maquettes des parties front-end



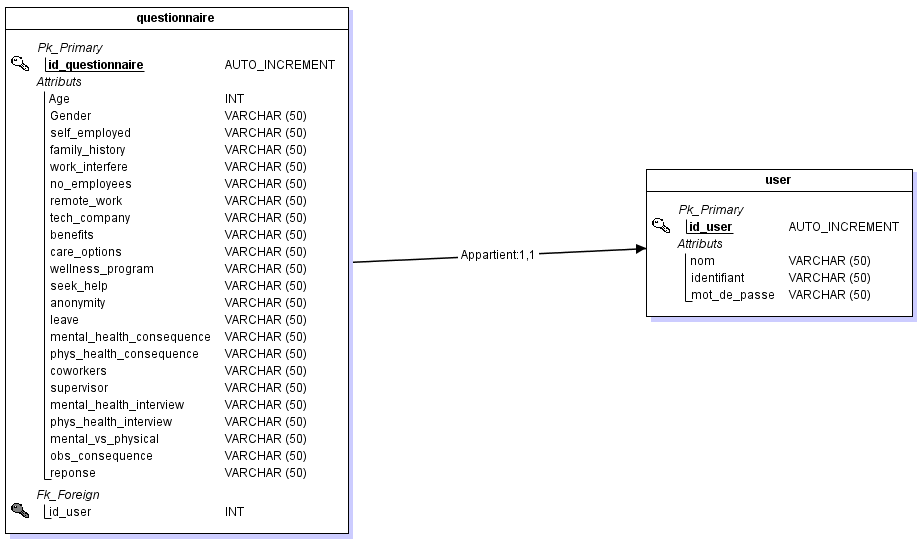
*Figure 25 : Maquette page de connexion*

*Figure 26 : Maquette visuelle de l’application*

Modèle conceptuel et relationnel de la base de données de l’application

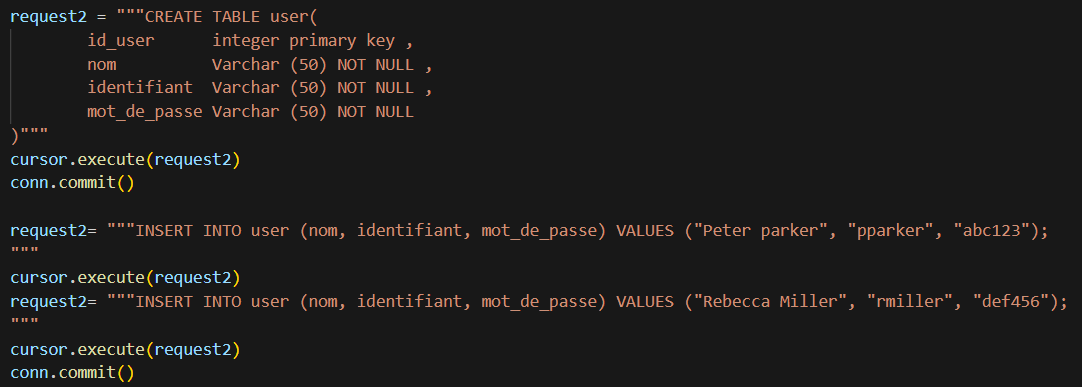


*Figure 27 : Modèle conceptuel des données*

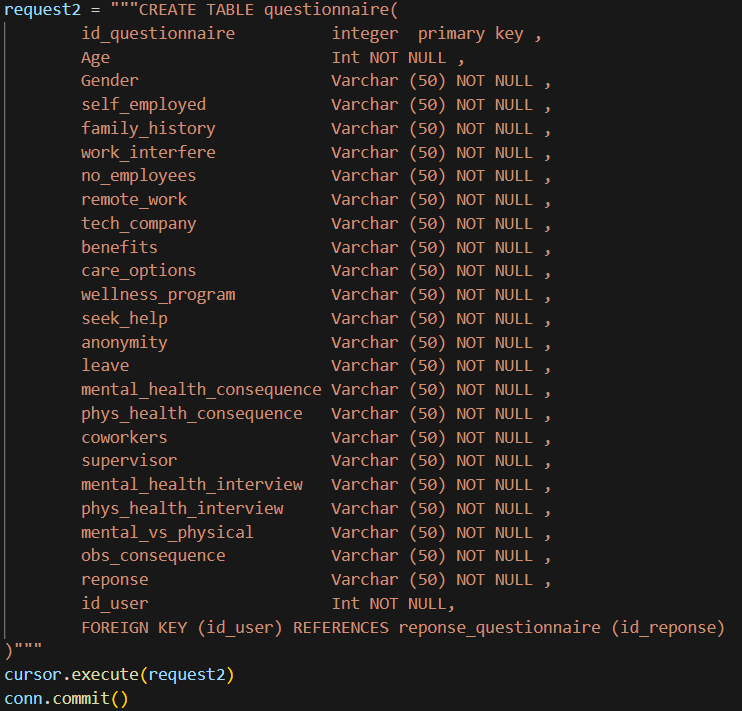


*Figure 28 : Modèle relationnel des données*

Annexe 4 : Mise en place de la base de données de l'application

- Création des tables inspirées du script SQL obtenu sur Jmerise à partir du MCD

*Figure 29 : Création de la table user et insertion des données*

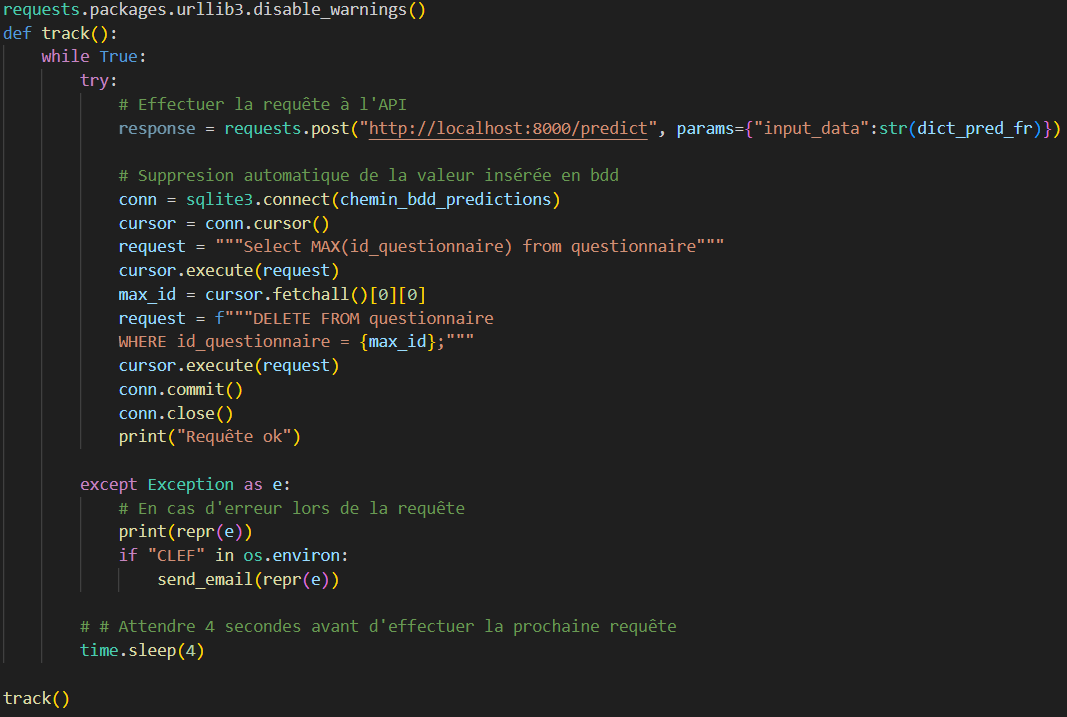


*Figure 30 : Création de la table questionnaires*

- Test d'une insertion dans la base d’un questionnaire en python

- Test d’insertion via FastAPI

- Tester d’insertion via l'application

Annexe 5 : Script de tracking des performances de l'application

*Figure 31 : Script qui traque les performances*