

# 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.

## Table des matières

1. Introduction du projet.....	3
2. Analyse de la demande .....	4
2.1 Enjeux réglementaires pour le traitement des données de l'application .....	4
2.2 Les utilisateurs du projet .....	4
2.4 Gestion du projet.....	4
3. Gestion des données analytiques du projet.....	6
3.1 Analyse exploratoire des données .....	6
3.2 Mise en place d'une base de donnée analytique .....	15
4. Etat de l'art .....	16
5. Choix techniques liés au projet .....	17
5.1 Partie IA .....	17
5.2 Partie application.....	19
6. L'organisation technique et l'environnement de développement .....	24
7. Bilan du projet et améliorations envisageables .....	25
Références.....	26
Annexes .....	27

## 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 déjà 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 de besoin de traitement.

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

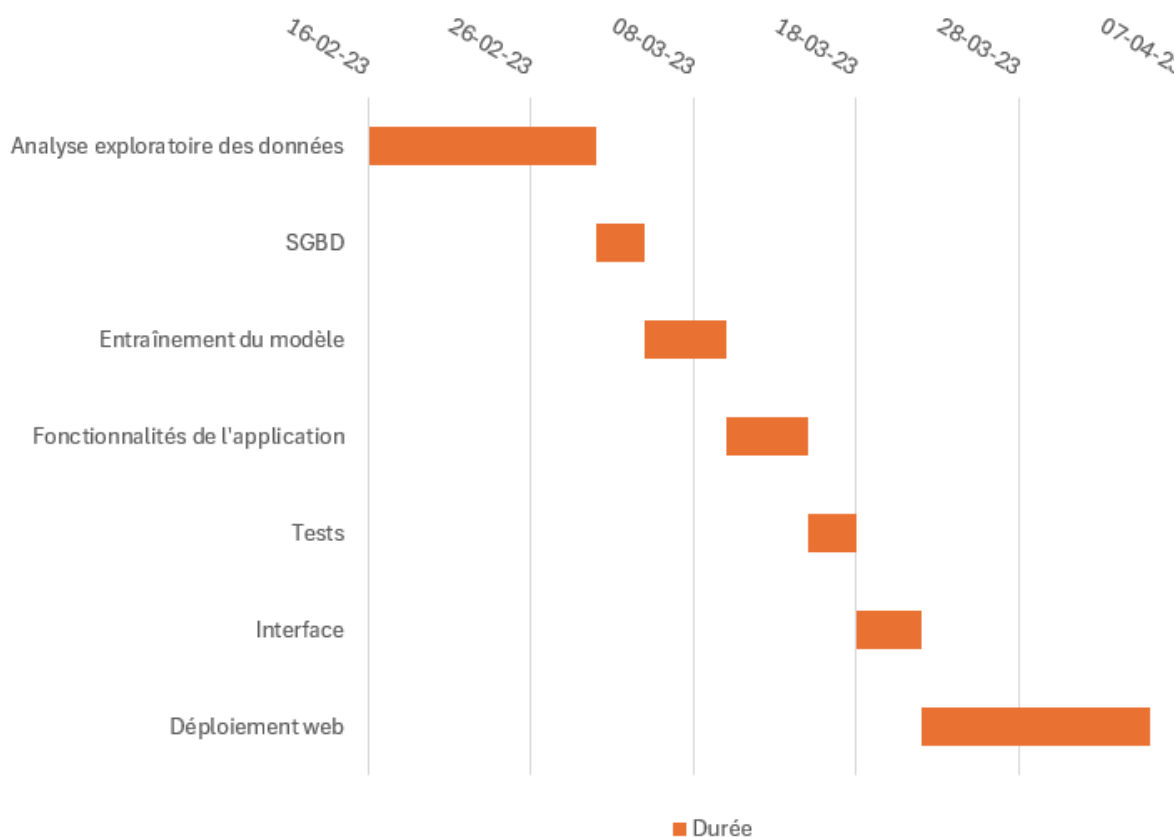
Il n'y a pas de réglementations particulières concernant l'utilisation de ces données, elles sont en libre accès sur Kaggle (1).

### 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 :



Etape	Date début	Durée	Date de fin
Analyse exploratoire des données	16-02-23	14	02-03-23
SGBD	02-03-23	3	05-03-23
Entraînement du modèle	05-03-23	5	10-03-23
Fonctionnalités de l'application	10-03-23	5	15-03-23
Tests	15-03-23	3	18-03-23
Interface	18-03-23	4	22-03-23
Déploiement web	22-03-23	14	05-04-23

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

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

- 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 communiquons par mails et appels quand c'était nécessaire.

Voir annexe 2 : Rapports d'avancements pour plus de détails.

### 3. Gestion des données analytiques du projet

#### 3.1 Analyse exploratoire des données

##### 3.1.1 Présentation générale

Voici un échantillon du jeu de données :

	Timestamp"	Age	Gender	Country	state	self_employed	family_history	treatment	work_interfere	no_employees	remote_work	tech_company	benefits	care_o
0	2014-08-27 11:29:31	37	Female	United States	IL	NaN	No	Yes	Often	6-25	No	Yes	Yes	N
1	2014-08-27 11:29:37	44	M	United States	IN	NaN	No	No	Rarely	More than 1000	No	No	Don't know	
2	2014-08-27 11:29:44	32	Male	Canada	NaN	NaN	No	No	Rarely	6-25	No	Yes	No	
3	2014-08-27 11:29:46	31	Male	United Kingdom	NaN	NaN	Yes	Yes	Often	26-100	No	Yes	No	
4	2014-08-27 11:30:22	31	Male	United States	TX	NaN	No	No	Never	100-500	Yes	Yes	Yes	

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 est des 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, qui correspondent 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

```
df.isnull().sum()
```

Timestamp"	0
Age	0
Gender	0
Country	0
state	515
self_employed	18
family_history	0
treatment	0
work_interfere	264
no_employees	0
remote_work	0
tech_company	0
benefits	0
care_options	0
wellness_program	0
seek_help	0
anonymity	0
leave	0
mental_health_consequence	0
phys_health_consequence	0
coworkers	0
supervisor	0
mental_health_interview	0
phys_health_interview	0
mental_vs_physical	0
obs_consequence	0
comments	1095
dtype: int64	

Figure 3 : Valeurs manquantes du jeu de données

Quatre colonnes possèdent des valeurs manquantes, il faut trouver une solution pour gérer chacune de ces valeurs manquantes.

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 interférerait 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) :

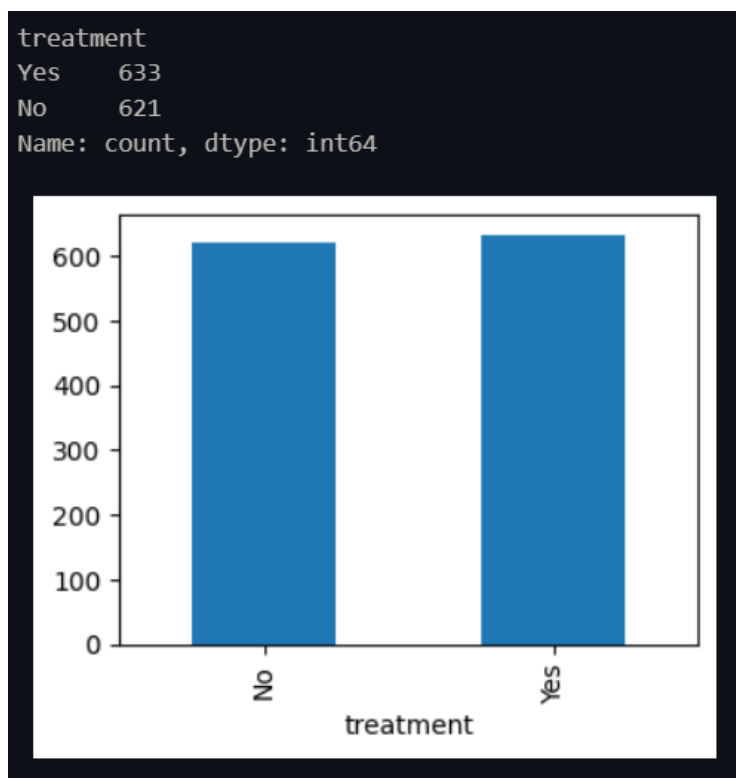


Figure 4 : Répartition des valeurs de la colonne "treatment"

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 :

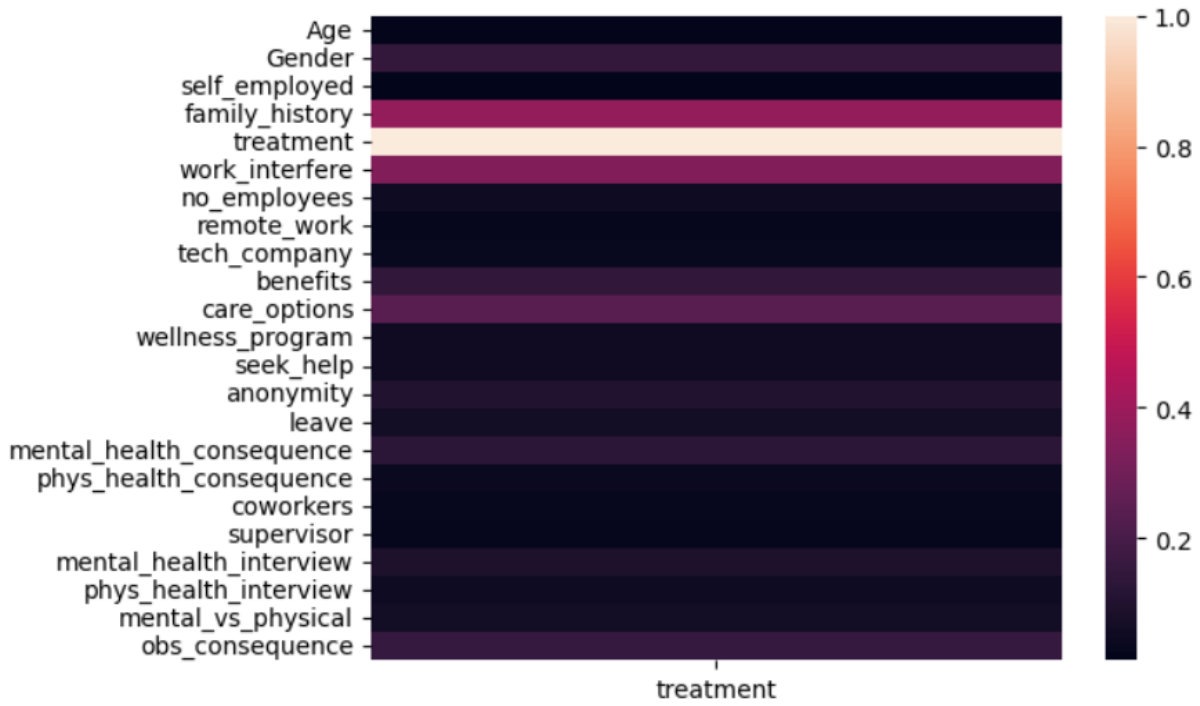


Figure 5 : Corrélation entre la colonne traitement et les autres colonnes

On voit sur la figure 5 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 répartition des valeurs de cette colonne par rapport au fait d'avoir fait une demande de traitement.

family\_history :

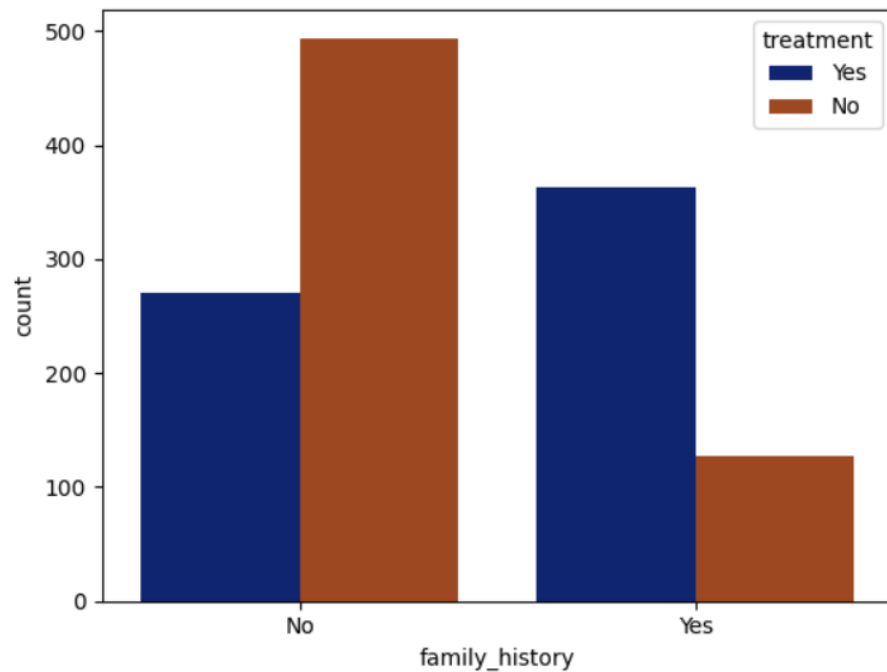


Figure 6 : Corrélation entre antécédents familiaux et demande de traitement

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

work\_interfere :

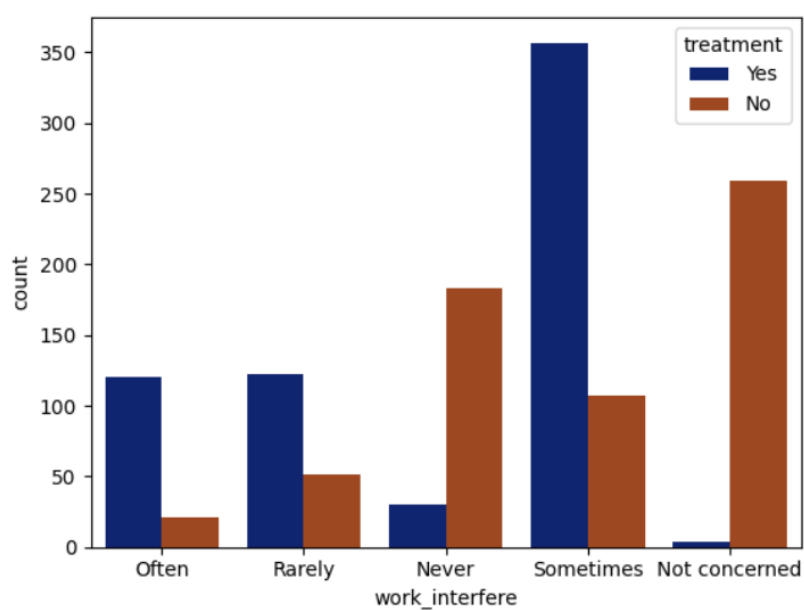


Figure 7 : Corrélation entre la colonne "work\_inference" et "traitement"

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'interférence 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.

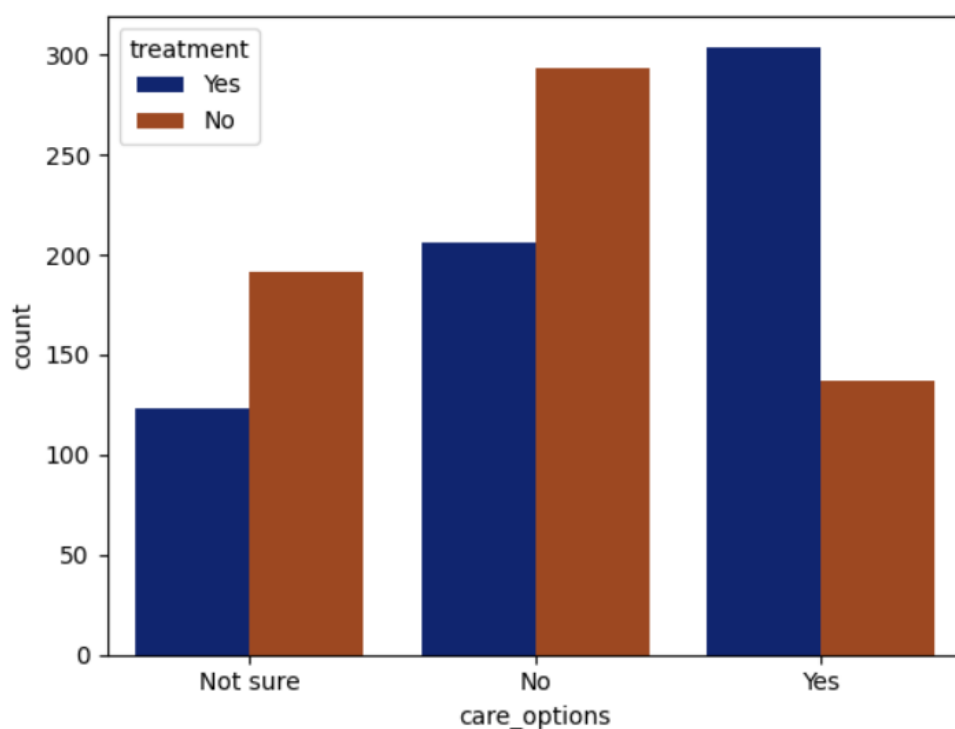


Figure 8 : Corrélation entre care\_options et treatment

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 ?"

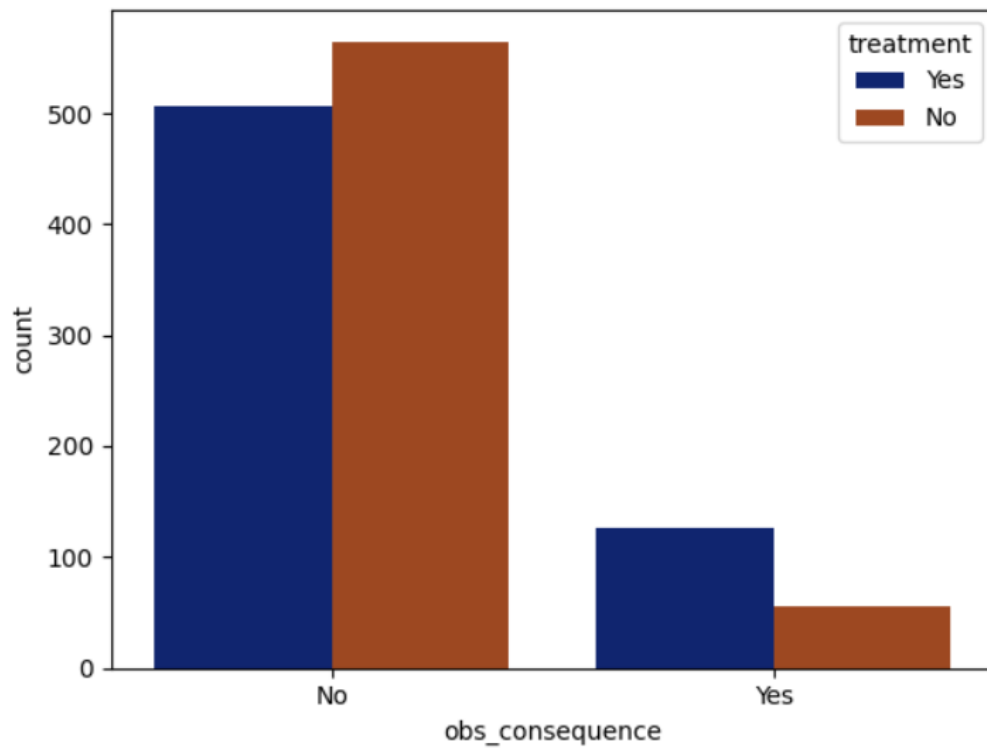


Figure 9 : Corrélation entre obs\_consequence et treatment

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

Genre :

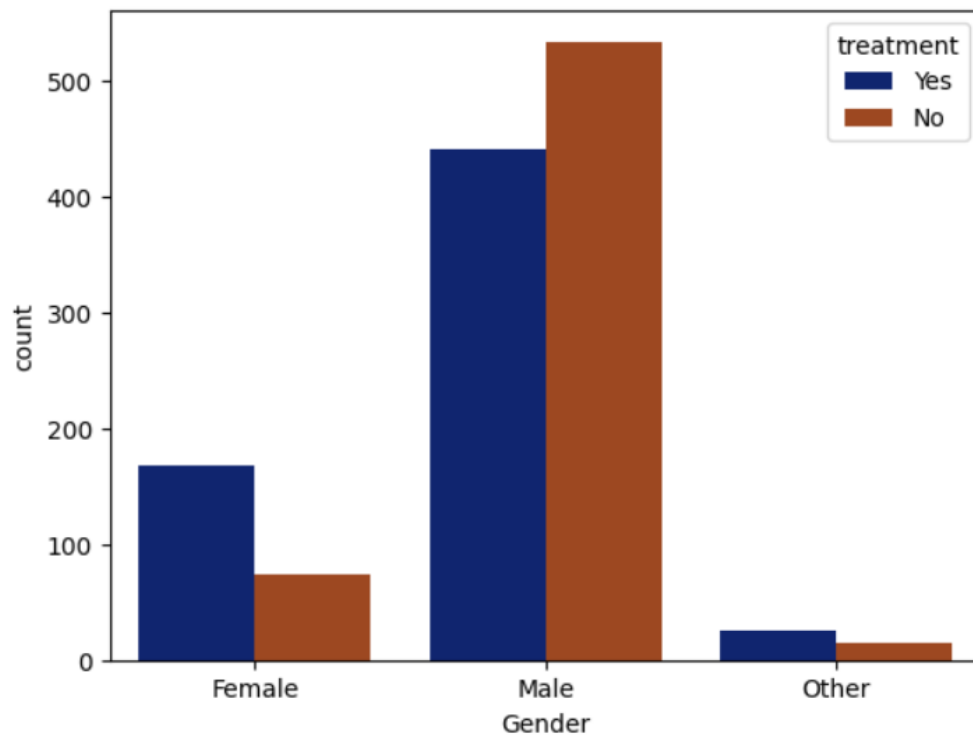


Figure 10 : Corrélation entre le genre et traitement

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 ?

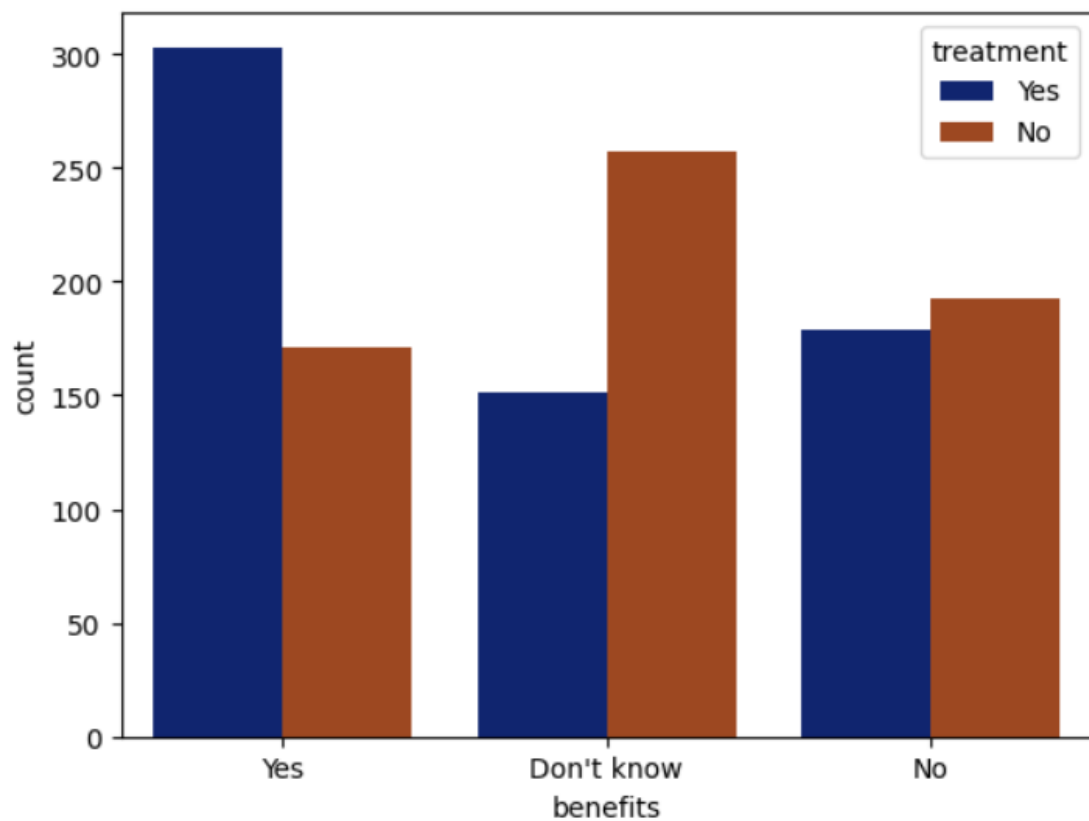


Figure 11 : Corrélation entre benefits et traitement

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 ?

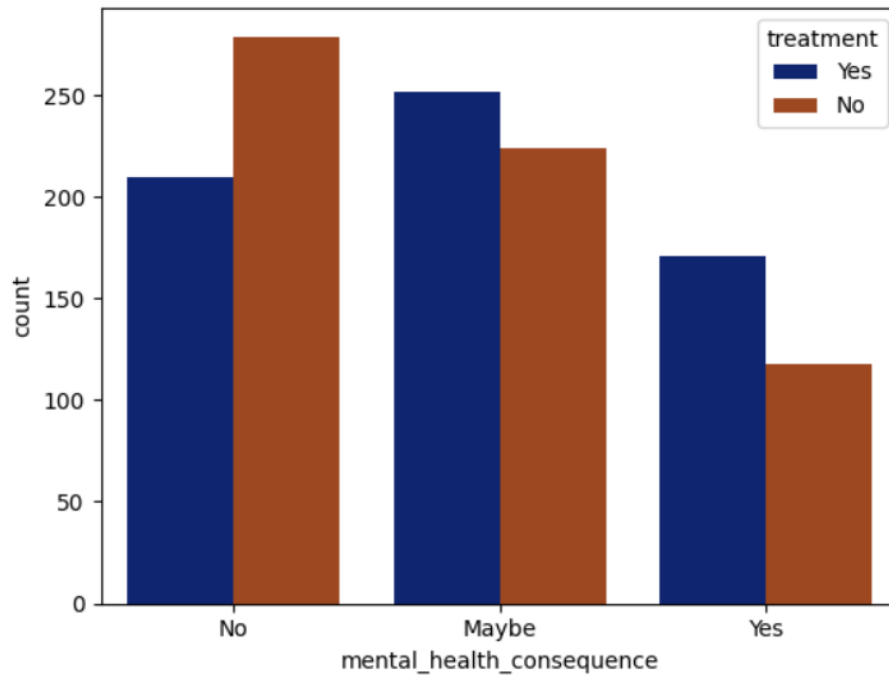


Figure 12 : Corrélation entre mental\_health\_consequence et traitement

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. Ceux qui répondent non ont moins souvent demandé un traitement.

### 3.2 Mise en place d'une base de données 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 données dans l'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 :

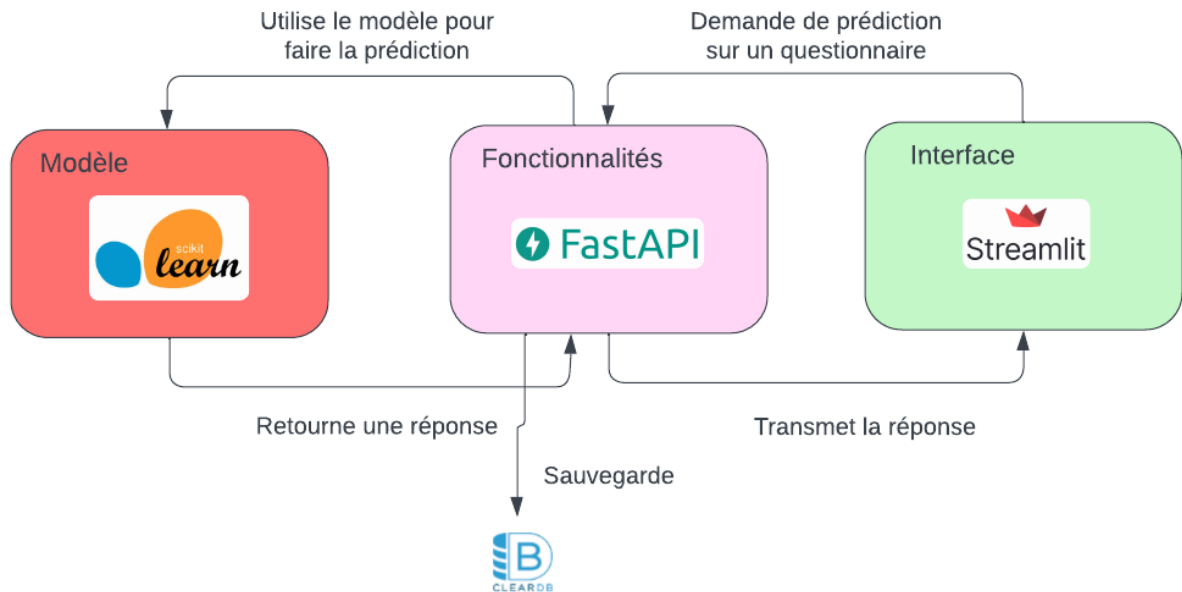


Figure 13 : Schéma fonctionnel utilisé pour le projet

### 5.1 Partie IA

J'ai entraîné 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.

	LogisticRegression	KNeighborsClassifier	DecisionTreeClassifier	AdaBoostClassifier	GradientBoostingClassifier	RandomForestClassifier
precision	0.84	0.796	0.769	0.811	0.808	0.802
recall	0.82	0.639	0.680	0.811	0.828	0.828
f1	0.83	0.709	0.722	0.811	0.818	0.815

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.

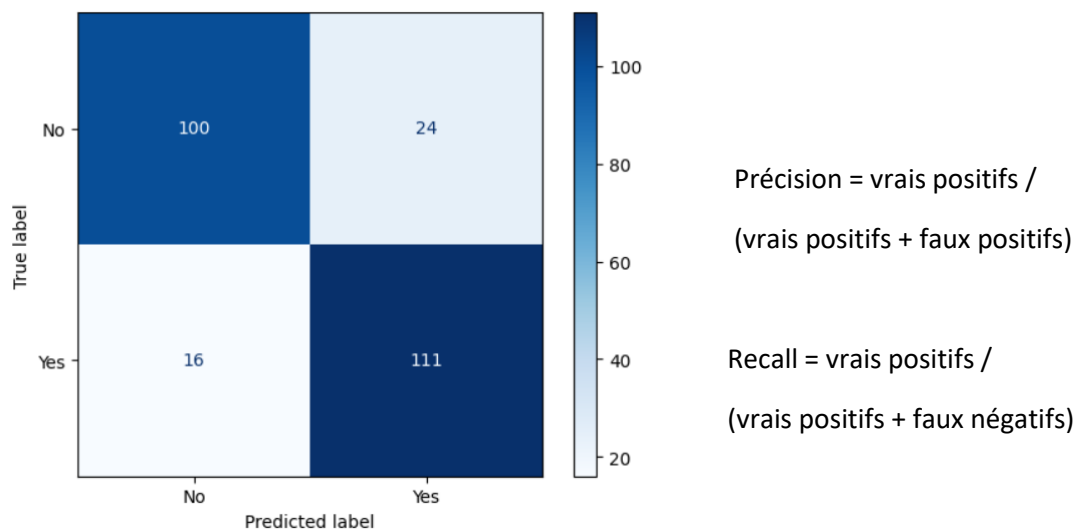


Figure 25 : Matrice de confusion

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.

```
@app.post("/predict")
def predict(input_data):
    print(input_data)

    dict_pred_fr = eval(input_data)

    # Traduction du dictionnaire en anglais pour la prédiction
    dict_pred_en = dict_pred_fr.copy()
    for key, value in dict_pred_en.items():
        value = translate(value)
        dict_pred_en[key] = value

    # Prédiction
    input_df = pd.DataFrame(dict_pred_en, index=[0])
    prediction = model.predict(input_df)[0]

    # Insertion des données en base sqlite
    # Complétion du dictionnaire avec le résultat
    if prediction == "Yes":
        reponse = "Besoin d'un traitement"
    elif prediction == "No":
        reponse = "Pas besoin d'un traitement"

    dict_pred_fr["reponse"] = reponse

    # Entrée du questionnaire complet en base
    conn = sqlite3.connect(chemin_bdd_predictions)
    cursor = conn.cursor()
    request = f"INSERT INTO questionnaire {tuple(dict_pred_fr.keys())}
VALUES {tuple(dict_pred_fr.values())}"
    cursor.execute(request)
    conn.commit()
    conn.close()

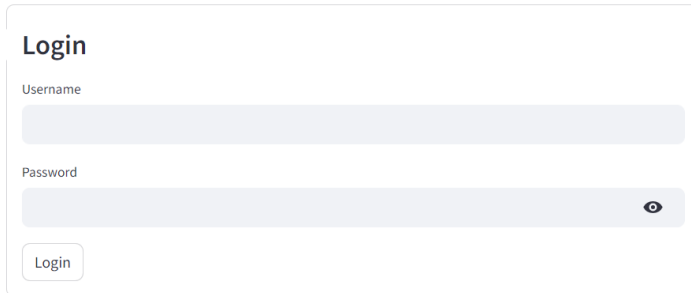
    return {"prediction" : prediction}
```

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.



The image shows a login form titled "Login". It contains two input fields: "Username" and "Password". The "Password" field has a toggle icon (an eye) to the right of it. Below the fields is a "Login" button.

Figure 16 : Page de connexion

## Détection de besoin de traitement pour la santé mentale avec l'IA

Quel est votre âge ?

Genre

- ☒ Homme  
☐ Femme  
☐ Autre

Êtes-vous travailleur indépendant ?

- ☒ Oui  
☐ Non  
☐ Autre

Avez-vous des antécédents de problèmes de santé mentale dans la famille ?

- ☒ Oui

- ☒ Oui  
☐ Non  
☐ Peut-être

Pensez-vous que votre employeur prend autant au sérieux la santé mentale que la santé physique ?

- ☒ Oui  
☐ Non  
☐ Je ne sais pas

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

- ☒ Oui  
☐ Non

Lancer la prédiction

Vous devriez considérer le fait de prendre un traitement ou de chercher de l'aide.

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 leurs réponses sont enregistrés dans une table questionnaires.

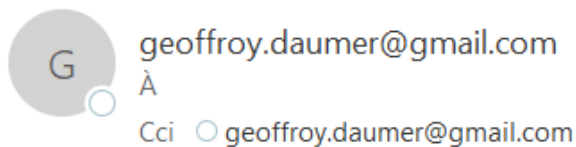
Dans cette table, il y a une clé étrangère qui fait une relation avec une table Utilisateur (annexe 3), ainsi, on sait qui a rempli chaque questionnaire.

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

La mise en place de la base de données est détaillée 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 exécution des requêtes, s'il y a une erreur, on voit l'erreur en direct et je recevrais un mail.



`OperationalError('database is locked')`

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 exécute des instructions avec github action, à chaque push et pull.

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

```
name: Streamlit / FastApi app

on:
  push:
    branches: [ "main" ]
  pull_request:
    branches: [ "main" ]

jobs:
  build:

    runs-on: ubuntu-latest

    steps:
      - uses: actions/checkout@v3
      - name: Set up Python 3.10.7
        uses: actions/setup-python@v3
        with:
          python-version: "3.10.7"
      - name: Display python version
        run: python -c "import sys; print(sys.version)"

      - name: Install dependencies
        run: |
          cd E1/code/api
          python -m pip install --upgrade pip
          pip install -r requirements.txt
      - name: Execute test
        run: |
          cd E1/code/api
          pytest
```

Figure 20 : Fichier `.yml` pour Github actions

## 5.2.6 Lancement de l'application

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

Je 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.

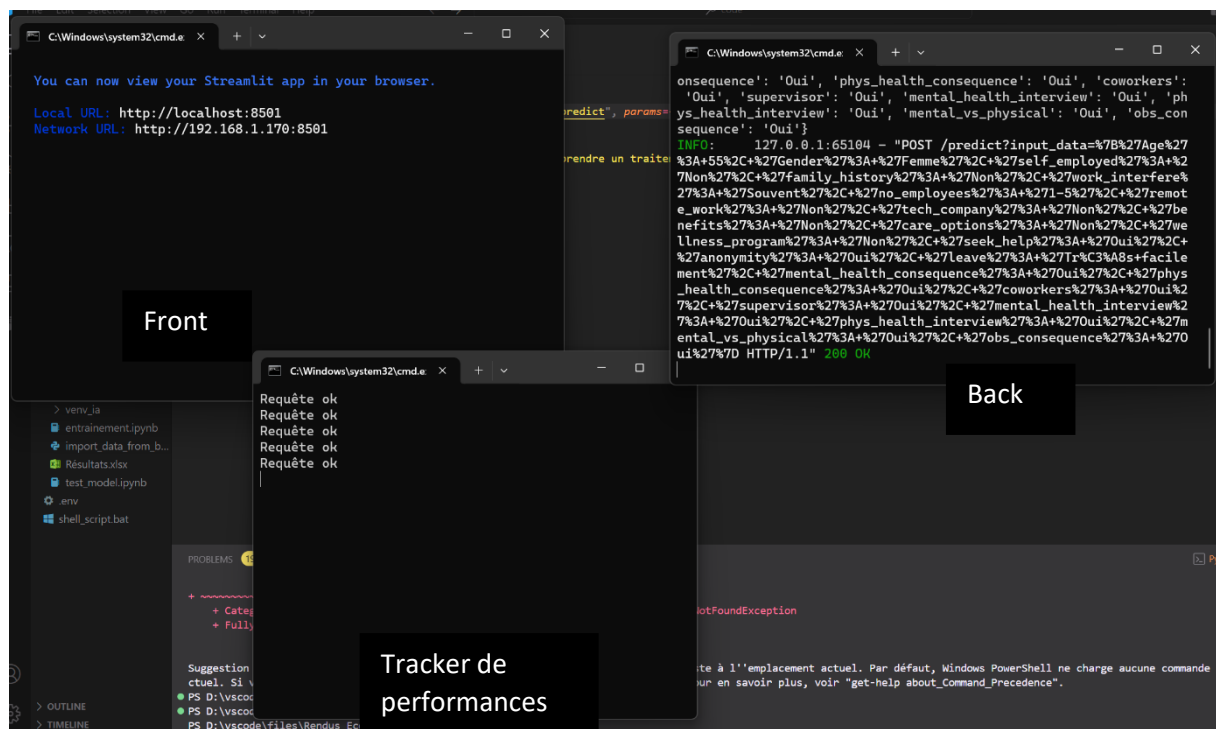


Figure 21 : Terminals ouverts au lancement de l'application

```
@echo off

rem Activer api
cd api
call venv_api\Scripts\activate.bat
start cmd /k "python api.py"

rem Lancer tracker
start cmd /k "python tracking.py"

rem Lancer appli
cd ../app
call venv_app\Scripts\activate.bat
start cmd /k "streamlit run app.py"
```

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

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 Platform notamment). Nous avons 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=EUD07liviJg&list=PLO\\_fdPEVIfKqUF5BPKjGSh7aV9aBshrpY](https://www.youtube.com/watch?v=EUD07liviJg&list=PLO_fdPEVIfKqUF5BPKjGSh7aV9aBshrpY).
4. **Statquest.** [En ligne] [https://www.youtube.com/watch?v=\\_L39rN6gz7Y&t=813s](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
Annexe 2 : Rapports d'avancement .....	30
Annexe 3 : Analyse du besoin back, front et base de donnée de l'application.....	31
Annexe 4 : Mise en place de la base de données de l'application .....	33
Annexe 5 : Script de tracking des performances de l'application .....	34

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.


Qusetionnaire	
<i>Pk_Primary</i>	
 <u>Id</u>	AUTO_INCREMENT
<i>Attributs</i>	
Age	INT
Genre	VARCHAR (50)
self_employed	VARCHAR (50)
family_history	VARCHAR (50)
work_interfere	VARCHAR (50)
remote_work	VARCHAR (50)
tech_company	VARCHAR (50)
benefits	VARCHAR (50)
care_options	VARCHAR (50)
wellness_program	VARCHAR (50)
seek_help	VARCHAR (50)
anonymity	VARCHAR (50)
leave	VARCHAR (50)
mental_health_consequence	VARCHAR (50)
phys_health_consequence	VARCHAR (5)
coworkers	VARCHAR (50)
supervisor	VARCHAR (50)
mental_health_interview	VARCHAR (50)
phys_health_interview	VARCHAR (50)
mental_vs_physical	VARCHAR (50)
obs_consequence	VARCHAR (50)

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')
```

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 stratégies 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.

## Analyse du besoin des fonctionnalités de l'application

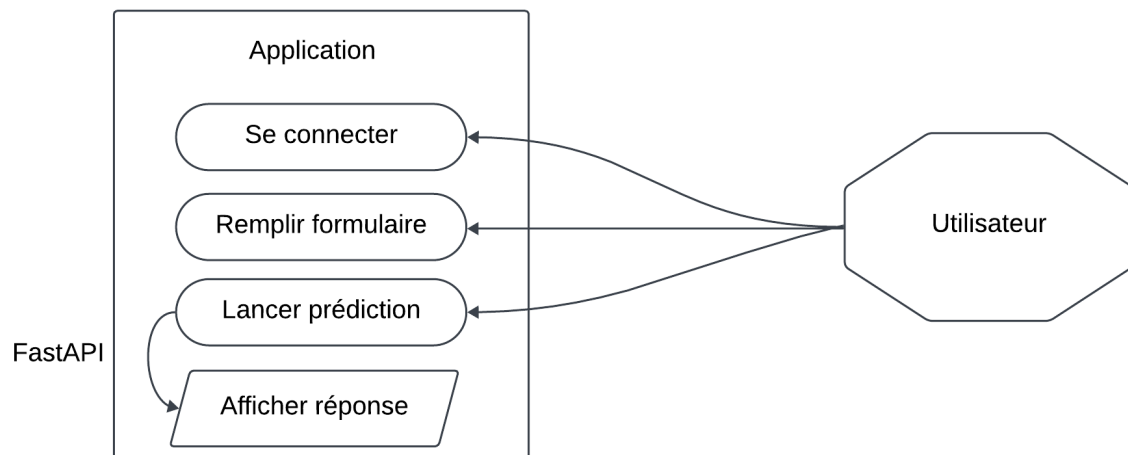


Figure 24 : Formalisation des fonctionnalités de l'application

## Maquettes des parties front-end

Maquette de la page de connexion. Elle est présentée dans un rectangle et contient trois éléments alignés verticalement : un bouton "Identifiant", un bouton "Mot de passe" et un bouton "Connexion".

Figure 25 : Maquette page de connexion

Maquette visuelle de l'application. Elle est présentée dans un rectangle et contient le texte "Remplissez le formulaire pour savoir si vous avez besoin d'un traitement pour la santé mentale". En dessous, il y a trois sections de questions : "Question 1", "Question 2" et "Question n". Chaque section contient trois options : "Choix A", "Choix B" et "Choix C", chacune suivie d'un bouton radio. En bas de la maquette, il y a deux boutons : "Soumettre formulaire" et "Réponse formulaire".

Figure 26 : Maquette visuelle de l'application

## Modèle conceptuel et relationnel de la base de données de l'application

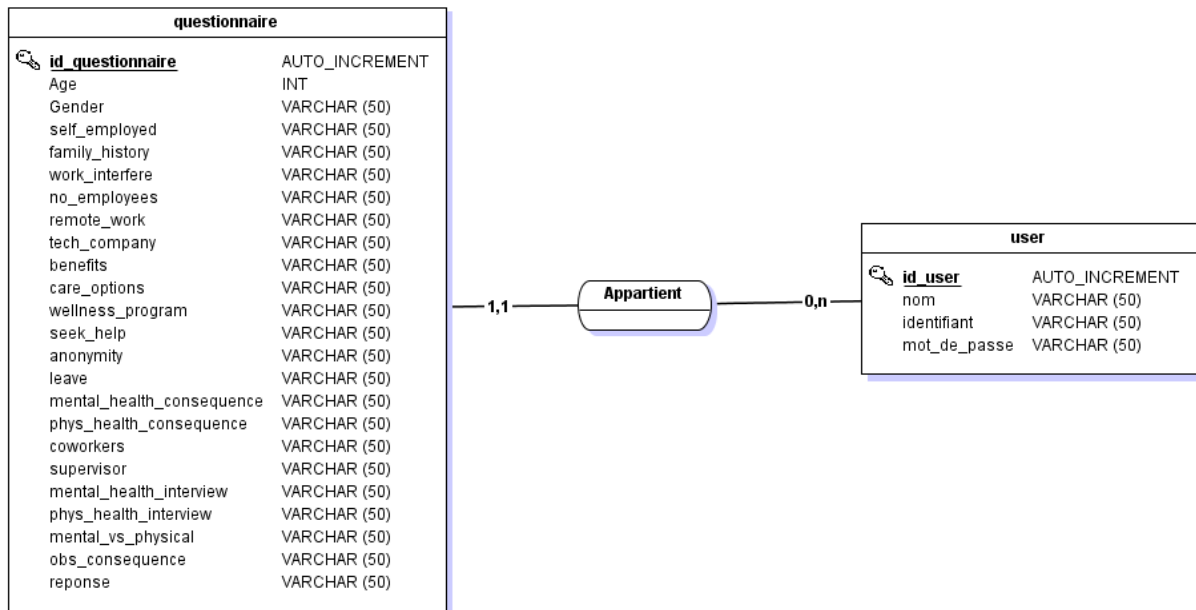


Figure 27 : Modèle conceptuel des données

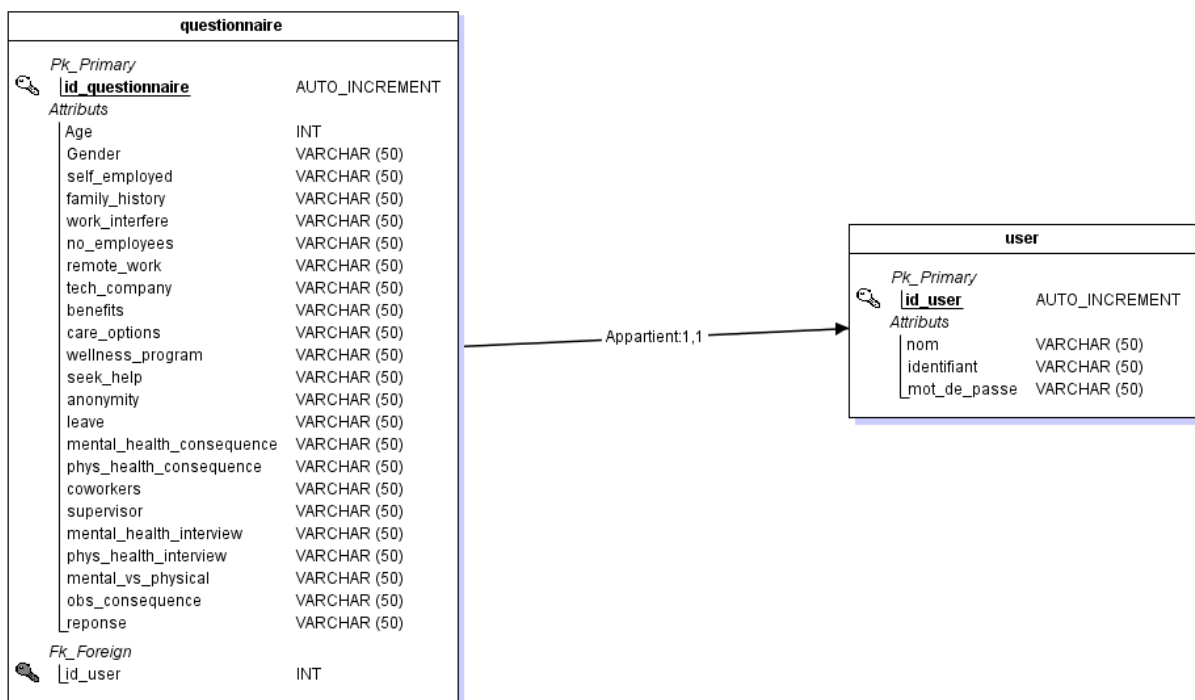


Figure 28 : Modèle relationnel des données



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

```
request2 = """CREATE TABLE user(  
    id_user      integer primary key ,  
    nom          Varchar (50) NOT NULL ,  
    identifiant  Varchar (50) NOT NULL ,  
    mot_de_passe Varchar (50) NOT NULL  
)"""  
cursor.execute(request2)  
conn.commit()  
  
request2= """INSERT INTO user (nom, identifiant, mot_de_passe) VALUES ("Peter parker", "pparker", "abc123");  
"""  
cursor.execute(request2)  
request2= """INSERT INTO user (nom, identifiant, mot_de_passe) VALUES ("Rebecca Miller", "rmiller", "def456");  
"""  
cursor.execute(request2)  
conn.commit()
```

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

```
request2 = """CREATE TABLE questionnaire(  
    id_questionnaire      integer primary key ,  
    Age                   Int NOT NULL ,  
    Gender                 Varchar (50) NOT NULL ,  
    self_employed          Varchar (50) NOT NULL ,  
    family_history          Varchar (50) NOT NULL ,  
    work_interfere          Varchar (50) NOT NULL ,  
    no_employees            Varchar (50) NOT NULL ,  
    remote_work             Varchar (50) NOT NULL ,  
    tech_company            Varchar (50) NOT NULL ,  
    benefits                Varchar (50) NOT NULL ,  
    care_options            Varchar (50) NOT NULL ,  
    wellness_program        Varchar (50) NOT NULL ,  
    seek_help               Varchar (50) NOT NULL ,  
    anonymity               Varchar (50) NOT NULL ,  
    leave                   Varchar (50) NOT NULL ,  
    mental_health_consequence Varchar (50) NOT NULL ,  
    phys_health_consequence Varchar (50) NOT NULL ,  
    coworkers               Varchar (50) NOT NULL ,  
    supervisor              Varchar (50) NOT NULL ,  
    mental_health_interview Varchar (50) NOT NULL ,  
    phys_health_interview   Varchar (50) NOT NULL ,  
    mental_vs_physical       Varchar (50) NOT NULL ,  
    obs_consequence         Varchar (50) NOT NULL ,  
    reponse                  Varchar (50) NOT NULL ,  
    id_user                  Int NOT NULL,  
    FOREIGN KEY (id_user) REFERENCES reponse_questionnaire (id_reponse)  
)"""  
cursor.execute(request2)  
conn.commit()
```

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*

```
requests.packages.urllib3.disable_warnings()
def track():
    while True:
        try:
            # Effectuer la requête à l'API
            response = requests.post("http://localhost:8000/predict", params={"input_data":str(dict_pred_fr)})

            # Suppression automatique de la valeur insérée en bdd
            conn = sqlite3.connect(chemin_bdd_predictions)
            cursor = conn.cursor()
            request = """Select MAX(id_questionnaire) from questionnaire"""
            cursor.execute(request)
            max_id = cursor.fetchall()[0][0]
            request = f"""DELETE FROM questionnaire
            WHERE id_questionnaire = {max_id};"""
            cursor.execute(request)
            conn.commit()
            conn.close()
            print("Requête ok")

        except Exception as e:
            # En cas d'erreur lors de la requête
            print(repr(e))
            if "CLEF" in os.environ:
                send_email(repr(e))

        # # Attendre 4 secondes avant d'effectuer la prochaine requête
        time.sleep(4)

track()
```

Figure 31 : Script qui traque les performances