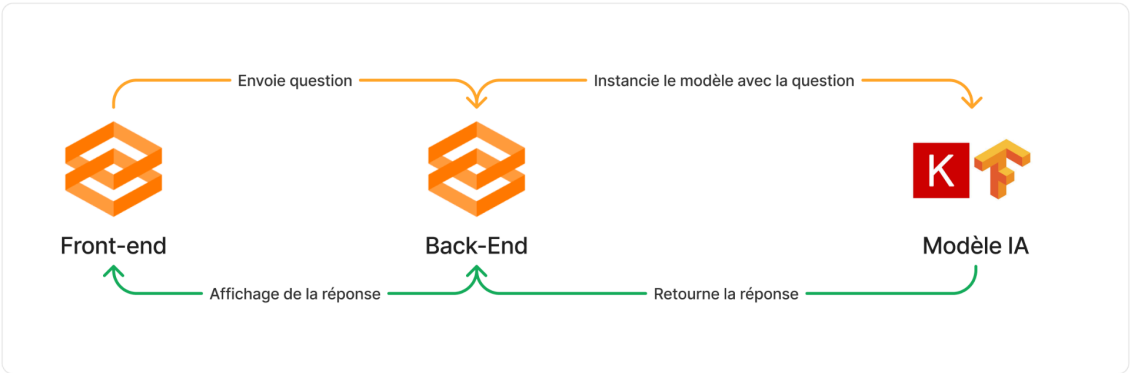
Equipe cancer - LIVRABLE 4

Description fonctionnel

Description fonctionnel



La communication se fait avec Gradio et la partie applicative avec TensorFlow + Keras.

Front-end Gradio

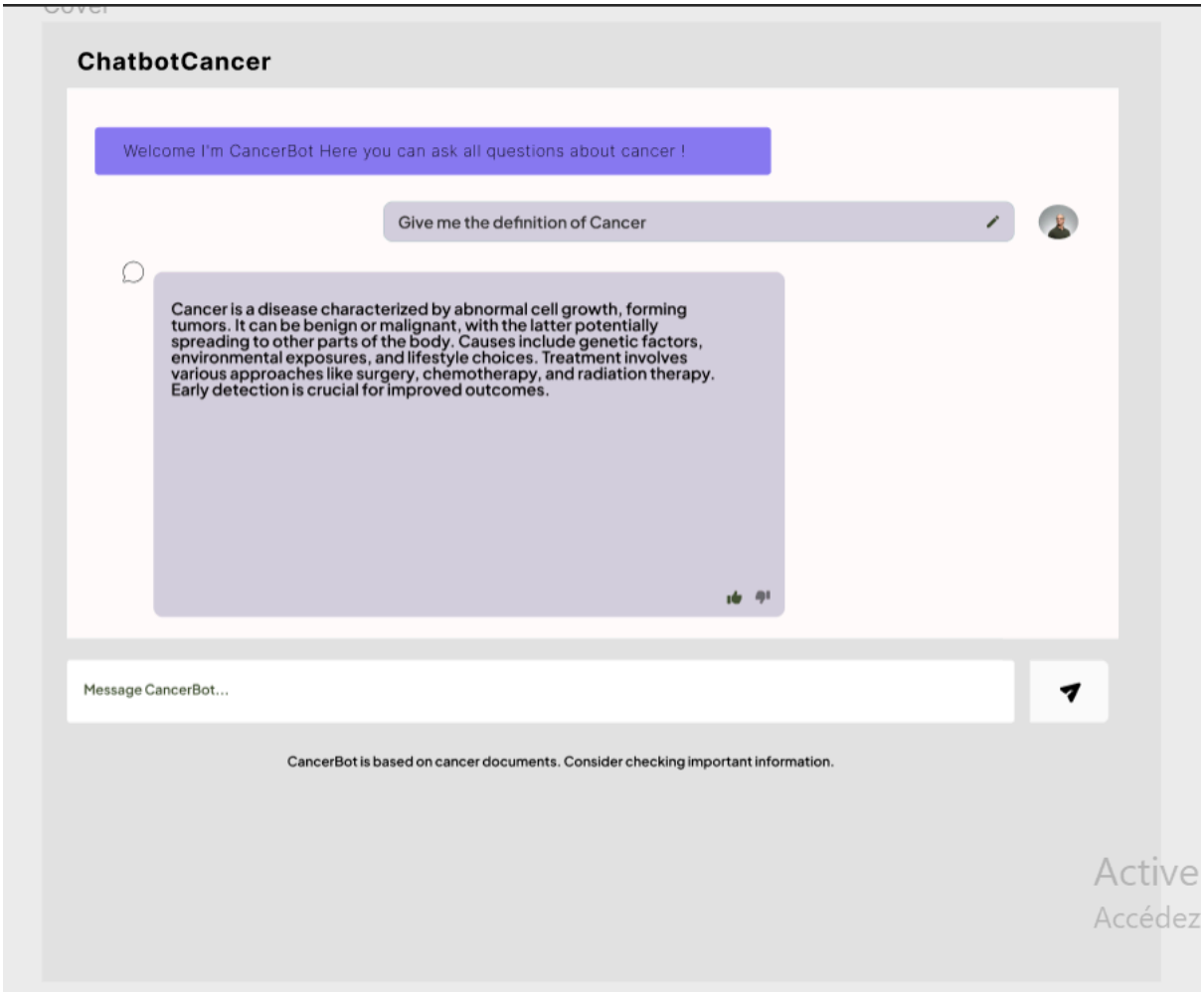
- Salma et Selma

Lien du Google Colab :
GradioApp.ipynb

Au cours du processus de développement du front-end du chatbot, nous avons suivi une méthodologie structurée pour atteindre les objectifs fixés. Voici un résumé des principales étapes et réalisations.

Conception de la Maquette du Chatbot

Avant de commencer le développement, nous avons créé une représentation du chatbot que nous voulions. Cette étape préliminaire nous a donné un aperçu des fonctionnalités et de l'interface utilisateur que nous voulions.



Maquette Chatbot

https://www.figma.com/file/v6S6V5DNyEtjvoKLq0UmTY/Maquette-Chatbot?type=design&node-id=0%3A1&mode=design&t=15xBYdEkipMA3QIQ-1

Analyse de la Documentation de Gradio

Pour implémenter le front-end, nous avons choisi d'utiliser Gradio et nous avons donc étudié sa documentation. Nous avons noté que Gradio présente certaines limites en termes de personnalisation CSS, cependant, cela ne nous a pas découragés, et nous avons décidé de trouver une solution.

Personnalisation CSS

Malgré les limitations CSS de Gradio, nous avons réussi à trouver des solutions créatives pour intégrer les éléments de la maquette du chatbot.Nous avons aussi rajouté des éléments qui n'étaient pas prévue dans la maquette tels qu'une image d'arrière plan.Cela à conduit à une version finale différente de la conception initiale mais qui a gagné en originalité grâce à ces ajouts.

Malgré ces petits problèmes, nous avons réussi à adapter le chatbot que nous avons créé à la conception initiale . L'attention portée aux détails et la persévérance ont été nécessaires pour s'assurer que l'interface utilisateur répondait à nos attentes.

Back-end Gradio

- Maxime, Walid

Le chatbot n'a pas nécessité la mise en place d'un back-end complexe. Nous avons opté pour une approche minimale en chargeant directement les poids du modèle GPT-2 dans le front-end, éliminant ainsi la nécessité d'une architecture back-end compliquée. Cette méthode simplifiée a non seulement allégé le processus de développement, mais a également préservé la puissance du modèle tout en limitant l'usage du back-end.

Mise en place des méthodes de communication

Lorsqu'une question est posée via l'interface utilisateur, la requête est transmise à la fonction `return_message` (question) (à modifier)côté back-end. Elle évalue la longueur du message et avertit l'utilisateur si elle ne répond pas aux critères. .Ensuite cette fonction utilise la méthode generate_response du chatbot pour générer une réponse en se basant sur le modèle GPT-2 préalablement chargé. La réponse est intégrée à l'historique de conversation entre l'utilisateur et le chatbot. Cette approche, orientée vers le front-end, maintient la simplicité du système.

```python
def return_message(message, history, model=gpt2_lm, max_length=128):
    if len(message) <= 1:
        gr.Warning('Please enter a message with more than one character.')
    elif len(message) > max_length:
        gr.Warning(f"Input should not exceed {max_length} characters.")
    else:
        cancer_answer = generate_text(model, message)
        message = "**You**\n" + message
        history.append([message, f"**CancerBot**\n{cancer_answer}"])
    return "", history
```

Fonction de gestion de message

Notre approche de développement du chatbot avec Gradio et GPT-2 a été caractérisée par une adaptabilité judicieuse. Les contraintes CSS de Gradio ont été surmontées côté front-end, permettant une concrétisation de notre maquette avec quelques modifications en plus . Côté back-end, l'utilisation

minimaliste de GPT-2, intégré via Gradio, a simplifié le processus tout en garantissant une expérience utilisateur fluide. La fonction `return_message` limite les réponses à 128 caractères, favorisant des interactions concises. En combinant ces éléments, nous avons réussi à créer un chatbot fonctionnel et réactif.

Fine-tuning

- Chrinovic, Amadou, Yassine

Du côté du fine-tuning de notre modèle, nous avons eu plusieurs problèmes que nous avons réussi à surmonter.

Ressource

Dû à l'immense quantité de données, nous avons eu un problème de ressources (GPU) lors de nos premières exécution de la méthode `fit` sur notre dataset.
Sur Google Colab mais également en local avec une carte graphique AMD qui ne pouvait pas être exploitée dans notre cas en raison de nombreux problèmes d'incompatibilité avec TensorFlow.
Par conséquent, nous avons utilisé des méthodes d'optimisation du GPU et de la RAM sur Colab, qui nous a permis d'exécuter notre `fit`. Suite à une suggestion de M. FAYE, nous avons utilisé LoRa, une méthode qui permet de geler certains poids du modèle GPT-2.
Cependant, un autre problème est survenu par la suite.

Les Données

Lors de nos premiers tests après le fine tuning nous avons remarqué qu'il y avait un problème, lorsque nous avons essayé de générer un texte à partir du nouveau modèle, cependant nous avons remarqué que le texte généré pour la question « what is a thyroid cancer ? » était « thyroid cancer is a » avec énormément d'espaces et de tabulations.
En cherchant le problème nous nous sommes rendu compte que le dataset comportait énormément d'erreurs que nous n'avions pas vu et par conséquent que nous n'avions pas réglé, en effet le dataset comportait énormément de doublons et avait des problèmes d'encodages. Nous sommes passés de 7569 lignes à 996 après doublons ce qui n'est pas du tout négligeable.

Pour remédier à ce problème nous avons nettoyé les données (data cleaning) pour régler les problèmes d'encodage (liens HTML, caractères accentués, ...).

Après avoir réglé tous ces problèmes nous avons enfin réussi à faire nos premiers tests sans toucher aux paramètres de la méthode `generate` et de la méthode `compile`.

Les prochains livrables :
Par la suite, nous avons prévu d'ajouter à notre application Gradio plusieurs sliders afin de jouer avec les paramètres plus facilement via la méthode `compile` (Top K, Top P, Beams, ...) et avoir les meilleures réponses possibles.