## Quelle(s) technique(s) avez-vous utilisé? / What technic(s) did you use?

* Complétion de texte / Text completion prompts
* Basé sur des instructions / Instruction-based prompts
* Choix multiples / Multiple-choice prompts
* Contextuel / Contextual prompts
* Atténuation des biais / Bias mitigation prompts
* Mise au point / Fine-tuning and interactive prompts
* Autre / Other \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## Quel(s) paramètre(s) avez-vous configuré(s)? / What parameters did you use?

* Top K – number :
* Top P – number :
* Temperature – number :
* Max New Tokens – number :
* Do Sample – Default | True | False
* Return Text - Default | True | False
* Return Full Text – Default | True | False
* Return Tensors - Default | True | False
* Clean Up Tokenization Spaces - Default | True | False
* Prefix – string :
* Handle Long Generation - Default | None| Hole
* Autre / Other \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## Décrire l’incidence du paramètrage si utilisé / Describe the impact of the parameters if used

TOP K pour avoir plusieurs token qui sont pris en charge du au fait qu’on a beaucoup de tokens avec l’historique.

Temperature est assé haut puisque nous voulons que le robot « feel » comme un robot et donne les mêmes réponses.

Max New Tokens puisque l’on veut pas créer des réponses trop longues

## Décrire les expérimentations faites ayant mené à la logique derrière votre bot / Describe the experiments carried out that led to the logic behind your bot

Nous avons d’abord expérimenté avec la version .js des transformers de hugginface en react, sans succès.

Nous avons donc ensuite opté pour un pipeline de requête simple et un historique de messages que l’on garde en mémoire pour ensuite que le robot complète la discussion et cela a bien fonctionné.

Ensuite on a testé plusieurs messages de système potentiels, certain pour la complétude, d’autres pour le mettre en personnage.

On a ensuite placé l’historique des message et demandé au robot de compléter la conversation.

On a également spécifié que le robot doit donner un diagnostic après une série de message pour qu’il dise d’aller à l’hôpital ou non.

Voici notre méthodologie :

On envoie 3 prompts, d’abord le system message pour mettre le bot en contexte, ensuite le message des historiques puis finalement on prépare le bot pour compléter la prochaine phrase. Voici un exemple :

“  
Here is a conversation history. Just write a single line of text to continue the conversation as medibot. Do not write more than one line of text.

system: Hello, I'm MediBot, your virtual assistant for today. Before we start, do you consent to share health-related information with me for the purpose of medical triage? Please respond with 'Yes' to consent or 'No' to decline.  
user: hello!

medibot:  
”

Ensuite on concatène le prochain message à l’historique et on continue comme ça jusqu’au diagnostic.

## Autre information pertinente / Other relevant information

On utilise react et d’autres libraires npm alors pour rouler le projet il faut :  
npm install

npm run dev

Aussi pour le UX on a rajouté une petite animation en attendant la requête.