

GNG2501

Mise à jour du progrès du projet de conception

Groupe FA1-5: Outil LMS



uOttawa

Soumis par:

Rodric Dib | 300340357

Anthony Alam | 300347280

KiroYoussef | 300349861

Ryan Appolon | 300373200

Othmane El Belghiti |
300280304

Le 22 septembre 2024

Université d'Ottawa

Table of Contents

Liste des acronymes.....	4
Introduction	4
Rapport de développement durable et CPX (livrable B).....	4
B.1 - Développement durable.....	4
B.1 (1) - Conséquences économiques, environnementales et sociales	4
B.1 (2) - Analyse du cycle de vie	6
B.2 - Conception pour X	7
B.2 (1 et 2) - 5 facteurs importants pour la conception	7
3. Priorisation des tâches par le logiciel	8
4. Exportation et lecture de fichiers CSV	8
5. Le logiciel comporte un "ChatBot"	8
Conclusion	9
Annexe	9
Livrable de projet C: Définition du problème, développement de concepts et plan de projet.....	11
C.1 - Définition du problème	11
C.1 (1) - Liste de besoins/problèmes	11
C.1 (2) - Énoncé de problème	12
Établir les métriques et unités (liées aux besoins).....	12
Étalonnage sur les métriques : produits compétitifs.....	13
C.1 (4) - Spécifications cibles	14
C.2 - Développement de concepts	16
(1) Création d'un algorithme qui analysera les fichiers CSV à l'aide de Python	16
(2) Création d'un algorithme qui analysera les fichiers CSV à l'aide de JavaScript	16
(3) Création d'un algorithme qui analysera les fichiers CSV à l'aide de Java	17
(1) Création d'un "ChatBot" statique et invariable.....	17
(2) Création d'un "ChatBot" basé sur le modèle "Ollama"	18
(3) Création d'un "ChatBot" basé sur le modèle "GPT-4"	18
(4) Création d'un "ChatBot" basé sur le modèle de "groqcloud"	18
(5) Création d'un "ChatBot" basé sur le modèle de "Gemini 1.5"	19
C.3 - Plan de projet	24
Livrable D : Conception détaillée et BOM.....	35
1. Résumez la rétroaction des clients reçue lors de votre deuxième rencontre client et énoncez clairement ce qui doit être changé ou amélioré par rapport à votre concept.	35
2. Basé sur votre rencontre de clients, développez un concept détaillé et mis à jour de votre concept qui inclut :	36
3. Expliquez les éléments à prendre en compte pour concevoir ou fabriquer votre concept, en fonction de votre concept détaillée mise à jour, afin de respecter vos facteurs CPX. Certains facteurs sont-ils plus importants que d'autres? Pourquoi?	40
4. Fournissez une liste détaillée des compétences et des ressources dont vous disposez pour vous permettre de créer votre concept. S'il manque des compétences ou des ressources pour compléter votre concept, décrivez comment vous allez les obtenir.	41

5. Fournissez une évaluation réaliste du temps requis pour mettre en œuvre votre concept et du temps réel dont votre groupe et ses membres individuels disposent.	43
6. Définissez toute autre hypothèse de produit critique qui pourrait affecter votre capacité à mettre en œuvre votre concept. Par exemple : les valeurs acceptables pour une spécification, la disponibilité d'un matériel/une composante, ou une fonctionnalité critique.	44
7. Fournissez une nomenclature des matériaux et des composantes (NDM ou BOM) détaillée pour votre prototype final, qui sera présenté à votre gestionnaire de projet pour approbation et achat. Assurez-vous d'inclure des liens électroniques pour chaque produit dans votre BOM (incluant les items à \$0). On vous donnera jusqu'à un maximum de \$50 ou \$100 (dépendant de votre projet) pour le développement de votre prototype final seulement. Avant de faire des achats, vous devez consulter le guide suivant:	45
Conclusion:	46
E.1 - Prototype 1	47
E.2 - Présentation sur le progrès du projet	52
(a) Composantes clés	52
(b) Résumé et présentation de la rétroaction du client	53
(c) Plan à venir et analyse de notre capacité à suivre le plan jusqu'à date.....	54
(a) Présentation des éléments de E.1	55
(b) Démonstration en temps réel de(s) prototype(s)	55
E.3 - Rétroaction des pairs et dynamiques d'équipes.....	56
Conclusion	56
Livrable F : Contraintes de conception et prototype 2	57
Introduction :	57
F.1 : Contraintes de conception.....	57
1.3. Fournissez des preuves (p.ex. analyse, calculs et/ou simulations simples, recherche) pour démontrer l'habileté de votre conception à satisfaire les contraintes. Justifiez le processus et les méthodes que vous avez utilisées.	58
F.2 : Prototype 2 :	61
2.1. Résumez toute nouvelle rétroaction des clients que vous avez reçus ou tous nouveaux résultats d'essais et énoncez clairement ce qui doit être changé ou amélioré par rapport à votre concept. Mettez à jour votre concept détaillé en conséquence.	61
2.2. Définissez vos hypothèses de produit les plus critiques que vous n'avez pas encore confirmées avec des essais. Expliquez quels facteurs CPX du Livrable du projet B cet(s) hypothèse(s) se rapportent.	63
2.3. Développez un deuxième ensemble de prototypes qui vous aideront dans votre cheminement vers la création de votre prototype final et faites l'essai des hypothèses de produit critique au fur et à mesure.	63
2.4. Documentez votre dernier prototype(s) en utilisant autant d'esquisses/diagrammes/ photos que nécessaire et expliquez le but et le fonctionnement du(es) prototype(s).	64
2.5. Faites l'essai de votre prototype, analysez et évaluez sa performance par rapport aux spécifications cibles mises à jour développées en premier pour le Livrable de projet C et documentez tous les résultats de vos essais, ainsi que les spécifications de votre prototype. Présentez vos essais en forme de tableau bien organisé qui démontre les résultats attendus comparés aux résultats réels (c.-à-d. faire la comparaison des spécifications mesurées de votre prototype à vos spécifications cibles en incluant les deux dans un tableau similaire à celui que vous avez développé pour le Livrable de projet C).	65
2.6. Exposez les grandes lignes sur ce que votre équipe a l'intention de présenter à vos clients, ainsi que l'information que vous voulez recueillir lors de votre prochaine rencontre client.	66

Liste des acronymes

Acronyme	Définition
LMS	Learning Management Tool
CPX	Conception pour X
RSI	Rendement sur Investissement
API	“Application Programming Interface”
IA	Intelligence Artificielle
CSV	“Comma-Separated Values”

Introduction

Notre projet porte sur la création d’un outil LMS qui permettra de rendre l’éducation à l’Université Rutgers. Présentement, les cours offerts à Rutgers sont tous enregistrés sur une plateforme en ligne nommée Canvas. Il existe une extension à Canvas, nommé Ally qui fournit des recommandations aux professeurs pour améliorer l’accessibilité de leurs cours. Ally n’est pas aimé par les professeurs et le personnel de l’Université qui n’ont pas des connaissances techniques car cette extension donne souvent des recommandations très longues et vagues. Nous avons été demandés de trouver une nouvelle façon de donner ses recommandations aux professeurs — par l’entremise d’un “ChatBot”. La création et l’entretien de cet outil aura de nombreuses conséquences économiques, environnementales et sociales qu’il faudra considérer. Dans ce rapport, nous allons discuter des conséquences liées au “ChatBot”, analyser son cycle de vie et lui définir 5 CPX.

Rapport de développement durable et CPX (livrable B)

B.1 - Développement durable

B.1 (1) - Conséquences économiques, environnementales et sociales

Il y a de nombreux facteurs à considérer concernant l’utilisation d’un “ChatBot” pour renforcer l’accessibilité des cours à Rutgers.

D’abord, il y a les impacts économiques entraînés par la création et l’entretien de cet outil. D’une part, les investissements dans l’éducation pour les personnes ayant des handicaps (comme le “ChatBot”) a un RSI 2 à 3 fois plus élevé en comparaison avec les gens qui ont des compétences régulières¹. Ce RSI est encore plus prononcé dans les pays

¹ Patrinos, A. - World Bank Blogs: <https://blogs.worldbank.org/en/education/disability-and-educationcharity-investment>

moins développés comme le Népal. Dans ce pays, le Dr. Kamal Lamichhane, un spécialiste dans l'étude des handicaps, estime que les investissements éducatifs résultent à un retour de 19 à 26%². D'autre part, suite à la création du "ChatBot" beaucoup d'argent devra être consacré pour l'entretien³. Il faudra employer au moins 1 ingénieur professionnel pour continuer à assurer la vitesse et l'efficacité de l'outil. Il est aussi important de considérer le fait que cet outil utilisera presque certainement le modèle d'intelligence artificielle de ChatGPT pour offrir de l'aide aux professeurs. À cause qu'il faut payer pour ce service, des frais variables⁴ (qui dépendent sur le nombre et la complexité des questions posées à l'outil) seront liés à l'utilisation du "ChatBot".

Ensuite, il ne faut pas oublier les conséquences nocives de l'utilisation de cet outil sur l'environnement. En premier lieu, en comparant l'empreinte électrique de l'utilisation d'un "ChatBot" (avec le ChatGPT API comme "backend") et recherche sur Google, il est clair que notre produit potentiel consommera beaucoup d'électricité. Effectivement, en posant la même question, le modèle d'IA derrière ChatGPT consomme 15 fois plus d'énergie qu'une recherche google⁵. De plus, il ne faut pas oublier le fait que des centaines des millions de puces électroniques sont utilisées pour assurer le bon fonctionnement de ChatGPT. La création de ces puces a un grand impact écologique. Par exemple, l'usine d'Intel, qui manufacture ceux-ci produit 15 000 tonnes de déchets dans 3 mois d'opération⁶. Les puces utilisées pour l'IA se retrouveront dans des processeurs graphiques. Malheureusement, avec l'avancement rapide de la technologie ceux-ci doivent être remplacées tous les 5 ans⁷; cela produit alors beaucoup de déchets électroniques.

Dernièrement, la création de notre produit aura des impacts sociaux importants, notamment dans la communauté des personnes handicapées. 80 % d'études réalisées dans les pays à faible revenu remarquent une corrélation entre la pauvreté et les personnes handicapées⁸. L'exclusion de l'éducation pour les personnes handicapées peut mener à leur capacité affaiblie de contribuer à la société qu'ils s'y retrouvent. Elle rend eux et leurs familles plus susceptibles à la pauvreté, et elle diminue aussi considérablement leur potentiel de revenus. En excès, cette diminution pourrait limiter la croissance économique du pays. L'accessibilité à des ressources pour aider les personnes handicapées à accéder l'éducation, comme notre produit par exemple, améliorera leur capacité de contribuer à la

² Ibid

³ Tiffany – ChatInsight: <https://www.chatinsight.ai/chatbots/chatbot-maintenance-checklist/#:~:text=However%2C%20chatbots%20need%20training%20and,will%20also%20fix%20technical%20issues>

⁴ OpenAI: <https://openai.com/api/pricing/>

⁵ Sreedhar, N – mintlounge: <https://lifestyle.livemint.com/news/big-story/ai-carbon-footprint-openaichatgpt-water-google-microsoft-111697802189371.html>

⁶ Belton, P – The Guardian: <https://www.theguardian.com/environment/2021/sep/18/semiconductor-siliconchips-carbon-footprint-climate>

⁷ Unicorn Platform Blog: <https://unicornplatform.com/blog/how-long-should-a-gpu-actually-last-expect-3-5years/>

⁸ Summary Report: The Economic Costs of Exclusion and Gains of Inclusion of People with Disabilities: <https://www.lshtm.ac.uk/sites/default/files/2020->

⁹ [/Summary%20Report_Costs%20of%20Exclusion_print.pdf](#)

société et ainsi promouvra la croissance économique nationale. Par exemple, au Népal, c'est estimé que l'intégration des personnes souffrant de déficiences physiques ou des sens dans les écoles générerait un rendement de salaire de 20% de plus¹⁰. Enfin, notre produit, un outil auquel les personnes handicapées peuvent en s'outiller, pourra rendre l'éducation plus accessible: il y aura moins de chance qu'ils entraînent un impact économique négatif pour leurs communautés. Aussi, il y aura plus d'emplois pour les travailleurs dans le secteur de l'éducation et il y aura plus d'emplois à salaire plus élevé pour les personnes handicapées. Tous ces stimulations économiques au sein de ces communautés amélioreront la qualité de vie pour les personnes ayant des handicaps et leur entourage.

B.1 (2) - Analyse du cycle de vie

Pour comparer notre produit à un produit similaire, on peut baser sur le cycle de vie d'un modèle d'intelligence artificielle développé par OpenAI, ChatGPT-4. Le cycle de vie de ChatGPT-4 inclut des phases similaires à celles que nous allons développer pour notre projet. Voici comment les étapes de l'ACV pourraient s'appliquer à ChatGPT-4:

Objectif et champ de l'étude : Pour ChatGPT-4, on évalue l'impact du développement du modèle, de son déploiement sur des serveurs, de son utilisation quotidienne par les utilisateurs et la maintenance régulière sans inclure les appareils des utilisateurs. L'objectif est de mesurer l'impact environnemental; la consommation d'énergie et les émissions de CO₂ tout en évaluant la satisfaction de l'utilisateur.

Analyse de l'inventaire: L'analyse de l'inventaire de ChatGPT-4 se concentre sur les flux et les matériaux et d'énergie au sein du système, ainsi que son interaction avec l'environnement. ChatGPT-4 fonctionne principalement dans des centres de données qui consomment d'importantes quantités d'énergie. L'énergie utilisée provient généralement de sources variées, allant des combustibles fossiles aux énergies renouvelables, ce qui impacte directement l'empreinte carbone globale. De plus, ChatGPT-4 se repose sur des matériels physiques. Les serveurs qui traitent les données sont fabriqués à partir de métaux et de divers matériaux divers, comme l'aluminium, le cuivre et le silicium¹¹. La production de ces serveurs nécessite des ressources naturelles, ce qui implique une consommation des matières premières lors de la fabrication et du transport. Finalement, l'énergie consommée par les centres de données entraîne des émissions de CO₂ et d'autres gaz à effet de serre. Lors de l'entraînement du modèle, la demande énergétique atteint des niveaux élevés qui contribue à une empreinte carbone significative. L'utilisation régulière du modèle pour répondre aux demandes des utilisateurs produit aussi des émissions, selon l'efficacité énergétique des serveurs.

¹⁰ Ibid

¹¹ Parton, H – RICS: <https://ww3.rics.org/uk/en/modus/natural-environment/renewables/data-centres-rawmaterials.html#:~:text=Raw%20materials%20involved%20range%20from,concrete%20of%20the%20actual%20buildings>.

Évaluation de l'impact: L'évaluation de l'impact de ChatGPT-4 repose sur les résultats de l'analyse de l'inventaire. À cette étape, on examine les différents indicateurs d'impacts et on détaille les résultats dans plusieurs catégories.

- (1) **Empreinte carbone:** Chaque requête traitée par ChatGPT-4 émet environ 0.0003 kg de CO₂. Selon les estimations, ChatGPT émettrait 8,4 tonnes de dioxyde de carbone par an, alors que les émissions de Google s'élèvent à plusieurs millions de tonnes par an. De plus, les émissions de CO₂ liées à l'entraînement des modèles sont déjà élevées, avec l'entraînement de GPT-3 émettant 552 tonnes de CO₂¹².
- (2) **Empreinte énergétique:** L'impact environnemental de ChatGPT inclut une empreinte énergétique significative, avec une consommation estimée à 121,517 MWh d'électricité par an si un travailleur américain sur dix utilise le service hebdomadairement. Cela équivaut à l'électricité nécessaire pour éclairer tous les foyers de Washington DC pendant 20 jours¹³.
- (3) **Empreinte aquatique:** L'eau est également un facteur important, utilisée pour refroidir les serveurs des centres de données. Par exemple, 50 requêtes à ChatGPT équivalent à la consommation d'eau d'une bouteille d'un demi-litre¹⁴.

Interprétation: L'interprétation du cycle de vie pour notre "ChatBot" LMS Tool doit se concentrer sur les impacts environnementaux similaires à ceux de ChatGPT, comme l'énergie nécessaire pour faire fonctionner les serveurs et l'utilisation d'eau pour les refroidir. Nous devons évaluer ces éléments pour minimiser la consommation d'énergie et les émissions de CO₂ tout au long du développement et de l'utilisation de notre produit. Cela signifie que dès la conception de notre "ChatBot", nous devons penser à des solutions pour limiter son empreinte écologique, par exemple en optimisant le code pour réduire les ressources nécessaires ou en hébergeant notre service sur des serveurs utilisant des énergies renouvelables. L'objectif est de rendre notre outil d'analyse d'accessibilité sur la plateforme LMS plus durable, tout en garantissant son efficacité et son accessibilité pour les utilisateurs.

B.2 - Conception pour X

B.2 (1 et 2) - 5 facteurs importants pour la conception

1. Le logiciel est simple d'utilisation.

¹² Richard, R – 24pm academy: <https://24pm.com/gpt/921-chatgpt-et-les-chatbots-sont-ils>

¹³ Le Gall, B - Slate: <https://www.slate.fr/tech-internet/chercheur-calcul-empreinte-carbon-chat-gpt-ia-eau>

¹⁴ Raphael, B et Mahier, T - Génération IA: <https://generationia.flint.media/p/comment-calculer-vraiment-impact-carbone-de-chatgpt-climat-ia-eau>

- a. Le client veut que le logiciel guide de manière interactive les professeurs, n'étant pas tous formés en matière d'accessibilité.
- b. Accompagne l'utilisateur dans les modifications qu'il doit apporter, avec des instructions claires, un fonctionnement étape par étape et avec la possibilité de poser des questions. (ex: TurboTax)
- c. Aucune qualification nécessaire pour l'utilisation du logiciel.

2. Le logiciel est conforme aux normes d'accessibilité.

- a. Les normes d'accessibilité WCAG 2.1 AA définissent les différents critères que le logiciel prend en compte pour guider les modifications proposées aux professeurs.
- b. Par la vérification de la conformité aux normes d'accessibilité, le logiciel permet l'inclusion des étudiants en situation de handicap.
- c. Le logiciel doit être mis à jour et respecter les dernières normes.

3. Priorisation des tâches par le logiciel

- a. Les professeurs ont du mal à travailler avec la solution actuelle qui propose de longues listes de modifications à faire. Le logiciel doit pouvoir mettre en avant les corrections à appliquer en se basant sur l'imminence d'utilisation des documents concernés. (Modification au "cours 2" avant "cours 3").
- b. Algorithme qui renvoie les modifications prioritaires, en prenant en compte, la date, l'importance du document à corriger pour les élèves et la difficulté de la modification.
- c. Métrique: nombre de corrections faites par jour/semaine/mois
- d. Encourager l'utilisateur avec des modifications réalisables et pertinentes à court terme.

4. Exportation et lecture de fichiers CSV

- a. Le client demande que le logiciel puisse lire des fichiers CSV.
- b. Importation de fichiers CSV et analyse de ces derniers, pointant les problèmes identifiés et donnant ainsi des recommandations adaptées.
- c. Le logiciel permet également l'exportation des résultats d'analyse sous forme de fichiers CSV pour une manipulation ou un partage ultérieur.

5. Le logiciel comporte un "ChatBot"

- a. Le "ChatBot" doit pouvoir interagir de manière fluide avec l'utilisateur, répondre à ses questions concernant l'amélioration de documents, notamment en ce qui concerne les aspects liés à l'accessibilité.
- b. Le "ChatBot" fournit des conseils personnalisés à l'utilisateur

- c. Le "ChatBot" est basé sur une API d'intelligence artificielle afin d'améliorer la pertinence et l'efficacité des réponses, garantissant une assistance en temps réel et adaptée aux besoins de l'utilisateur.

Conclusion

Il est évident qu'un "ChatBot", conçu pour améliorer l'accessibilité des cours en ligne offerts à l'Université Rutgers, aura des impacts positifs et négatifs.

D'abord, malgré que le développement et l'entretien du "ChatBot" sera coûteuse, il est quasiment certain que cet investissement aura un RSI significatif. Malheureusement, en ce qui concerne l'environnement, cet outil aura une grande empreinte énergétique et écologique. Toutefois, si on considère son effet social, c'est clair que le "ChatBot" aura que des impacts positifs sur la qualité de vie pour les personnes handicapées.

En terme du cycle de vie du "ChatBot", il sera important de se concentrer sur son impact environnemental. Cet impact est majoritairement nocif, particulièrement en ce qui concerne les empreintes carbone, énergétiques et aquatiques liés à l'utilisation de l'IA pour fournir du support aux utilisateurs de l'outil.

Finalement, nous avons décidé sur 5 CPX qui seront essentiels fin que le "ChatBot" puisse résoudre le problème d'accessibilité sur Canvas. Il est en premier lieu évident que cet outil soit facile à utiliser et qu'il soit conforme aux normes d'accessibilité. Le "ChatBot" aura aussi la capacité à identifier les tâches les plus importantes et faciles à effectuer afin d'améliorer l'accessibilité des cours en ligne. Cela doit se faire par l'entremise de fichiers CSV qui seront analysés par l'outil. Dernièrement, le "ChatBot" doit être, comme son nom l'indique, un logiciel interactif avec qui peut répondre aux questions des professeurs.

Annexe

Tableau à résultats nets triples

Résultat	Impact Positif	Impact Négatif
Économique	L'accès à l'éducation facilement accessible est gratuit	Nécessite de l'entretien et des frais à chaque fois que le produit est utilisé.

	Ça ne coute pas beaucoup à concevoir ce projet.	Nécessite une connexion internet, qui peut être hors de portée pour certains.
	Investir dans l'éducation des personnes ayant des handicaps vaut la peine.	Héberger l'application sur des serveurs peut générer des coûts.
Environnemental	Reduction d'utilisation de papier (impact écologique modeste).	Consommation significative d'énergie en comparaison à une recherche google.
	Réduction dans le besoin de construire de l'infrastructure (comme les centres d'appels) afin de répondre aux questions.	Tout le cycle de vie du prototype a un grand impact carbone, plastique et aquatique.
	Prototypage virtuel au lieu de prototypage physique. Moins d'émissions liés au transport.	Risque de surconsommation de ressources premières.
Social	Augmente l'accessibilité aux outils éducationnel déjà disponibles.	Risque de violation de la cybersécurité.
	Ça rend les applications compatibles plus ergonomique pour les utilisateurs.	Risque de dépendance pour ceux qui ne l'ont pas besoin.
	Le fait de rendre l'éducation plus accessible, améliore la qualité de vie pour les personnes ayant des handicaps.	Il y a un risque d'une stigmatisation pour son utilisation, associé à la perception de la marque.

Livrable de projet C: Définition du problème, développement de concepts et plan de projet

C.1 - Définition du problème

C.1 (1) - Liste de besoins/problèmes

À la suite de la première rencontre avec notre client, nous avons pu mieux comprendre les problèmes d'accessibilité reliés à *Canvas*. Nous avons aussi appris davantage sur les limites de l'extension *Ally*. Cela nous a permis de créer la liste de besoins et problèmes suivante:

Besoins:

Liste des besoins/problèmes des clients	1-5, où 5 est le plus important, et 1, le moins important
Doit être compatible avec les technologies d'assistance spécifiques sur lesquelles les utilisateurs s'appuient et à la manière dont ces technologies s'intègrent au logiciel LMS existant	4
La plateforme doit au moins adhérer au <i>WCAG 2.1</i> pour garantir l'accessibilité aux personnes handicapées	4
Le ChatBot doit être rapide	5
Pour les contenus visuels tels que les images ou les vidéos, des descriptions audios doivent être disponibles	1
Les utilisateurs doivent pouvoir personnaliser l'interface en fonction de leurs besoins, par exemple en ce qui concerne les couleurs ou la taille des icônes	4
Compatibilité avec les lecteurs d'écran pour les mal-voyants	3
Option de taille du texte réglable et contraste élevé	4
Le ChatBot doit être adapté aux besoins spécifiques du client, donc ça ne donne pas exactement la même la sortie pour chaque personne	5
Toutes les fonctionnalités doivent être disponibles via des raccourcis clavier, minimisant ainsi la dépendance à la souris	1
Doit être compatible avec des appareils personnels du clients, tels que des lecteurs d'écran, afficheurs braille, appareils auditifs, etc...	3

Des lois tels que la loi sur l'accessibilité pour les personnes handicapées de l'Ontario (LAPHO) imposent l'accessibilité des services en ligne, y compris des plateformes éducatives	1
Le ChatBot doit recueillir les commentaires des utilisateurs pour comprendre quelles sont les caractéristiques d'accessibilité qui fonctionnent le mieux et où des améliorations sont nécessaires dans le monde réel	5

Problèmes:


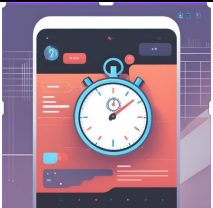
C.1 (2) - Énoncé de problème

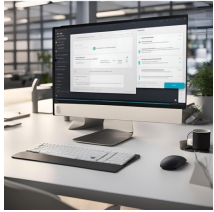
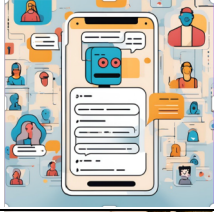

Énoncé Problème : Développer un chatbot interactif et facile à utiliser qui analyse le contenu des cours sur Canvas et priorise les recommandations pour améliorer l'accessibilité, en se basant sur les dernières règles d'accessibilité américaines.

Personnes affectées : Les professeurs de Rutgers, qui doivent se conformer aux nouvelles règles d'accessibilité, et les étudiants qui bénéficieraient d'un contenu éducatif plus accessible.

C.1 (3) - Liste de métriques

Établir les métriques et unités (liées aux besoins)

No. métrique	No. Besoins	Métrique	Importance	Unité	Images
1	1,10	Le nombre de technologies et appareils compatibles	3	-	
2	2,11	Pourcentage de conformité aux WCAG 2.1 et aux lois pertinentes	4	Pourcentage (%)	
3	3	Temps moyen pour obtenir une réponse du ChatBot	5	Secondes (ms)	

4	5	Nombre d'éléments de l'interface personnalisables	4	Nombre d'options de personnalisation	
5	9	Pourcentage de fonctionnalité	1	Pourcentage (%)	
6	12	Nombre de feedbacks collectés concernant les fonctionnalités d'accessibilité.	5	Nombre de retours	
7	5,7	Compatibilité avec les lecteurs d'écran et options de réglage du texte	4	-	
8	8	Taux de variabilité des réponses en fonction des utilisateurs	5	Variabilité des réponses (%)	
9	4,6	Nombre d'éléments visuels avec des descriptions audios disponibles	1	Nombre de descriptions disponibles	

Étant donné que le produit que nous prévoyons de concevoir pour le client, afin de satisfaire ses besoins particuliers, est une nouveauté sur le marché, nous effectueront une comparaison de produits existants qui répondent à des critères de conception similaires.

Pour effectuer cet étalonnage, nous comparerons les valeurs des différentes métriques des produits suivants:

- Chat GPT
- Sizzle AI
- Claude AI -
- Gemini AI
- Anthropic AI

Étalonnage sur les métriques : produits compétitifs

No	Métrique (Mesurable)	Unité	Importance	Chat GPT	Sizzle AI	Claude AI	Gemini AI	Anthropic AI
----	----------------------	-------	------------	----------	-----------	-----------	-----------	--------------

1	Taux de compatibilité des technologies d'assistance	% de compatibilité	4	90%	85%	88%	92%	90%
2	Respect des critères du WCAG 2.1	Nombre de critères	4	30/35	28/35	32/35	31/35	30/35
3	Temps de réponse du ChatBot	Millisecondes (ms)	5	200ms	250ms	190ms	180ms	210ms
4	% de contenu avec description audio	%	1	80%	75%	85%	90%	82%
5	Nombre d'options de personnalisation	Options disponibles	4	15	12	16	18	14
6	Compatibilité avec les lecteurs d'écran	% de compatibilité	3	95%	92%	96%	97%	93%
7	Gamme de réglage de la taille du texte et du contraste	% de flexibilité	4	85%	80%	90%	92%	88%
8	Taux de personnalisation de la sortie	% de réponses adaptées	5	85%	80%	87%	89%	86%
9	Nombre de fonctionnalités accessibles	Nombre de fonctionnalités	1	10	9	11	12	10
10	Compatibilité avec les appareils personnels	% de compatibilité	3	90%	85%	92%	93%	91%
11	Respect des exigences légales (LAPHO)	Oui/Non	1	Oui	Oui	Oui	Oui	Oui
12	Taux de feedback utilisateur intégré	% de feedback recueilli	5	85%	80%	88%	90%	85%

C.1 (4) - Spécifications cibles

Assigner des valeurs marginales et idéales:

No	Métrique (Mesurable)	Unité	Valeur cible idéale	Valeur marginaleme nt acceptable	Justification
1	Taux de compatibilité des technologies d'assistance	% de compatibilité	95%	90%	Doit être compatible avec la majorité des technologies d'assistance pour garantir une expérience utilisateur satisfaisante.

2	Respect des critères du WCAG 2.1	Nombre de critères	35/35	32/35	Le respect complet des standards WCAG est crucial pour l'accessibilité universelle.
3	Temps de réponse du ChatBot	Millisecondes (ms)	150ms	200ms	Un chatbot rapide améliore l'expérience utilisateur, surtout pour les personnes ayant des handicaps cognitifs ou moteurs.
4	% de contenu avec description audio	%	90%	85%	Des descriptions audios sont essentielles pour les malvoyants, ce qui augmente l'accessibilité des contenus visuels.
5	Nombre d'options de personnalisation	Options disponibles	18	15	Plus d'options de personnalisation permettent aux utilisateurs d'adapter l'interface à leurs besoins spécifiques.
6	Compatibilité avec les lecteurs d'écran	% de compatibilité	98%	95%	Les lecteurs d'écran sont l'une des technologies d'assistance les plus utilisées par les malvoyants, ce qui nécessite une compatibilité élevée.
7	Gamme de réglage de la taille du texte et du contraste	% de flexibilité	90%	85%	Le réglage de la taille du texte et du contraste est essentiel pour les personnes ayant des difficultés visuelles.
8	Taux de personnalisation de la sortie	% de réponses adaptées	90%	85%	Le chatbot doit fournir des réponses personnalisées pour répondre aux besoins variés des utilisateurs.
9	Nombre de fonctionnalités accessibles via raccourcis clavier	Nombre de fonctionnalités	12	10	Les raccourcis clavier sont cruciaux pour les utilisateurs qui ne peuvent pas utiliser la souris efficacement.
10	Compatibilité avec les appareils personnels	% de compatibilité	95%	90%	Une large compatibilité avec les appareils personnels est nécessaire pour offrir une accessibilité complète.

11	Respect des exigences légales (LAPHO)	Oui/Non	Oui	Oui	Adhérer aux lois sur l'accessibilité est obligatoire pour assurer la conformité légale.
12	Taux de feedback utilisateur intégré	% de feedback recueilli	90%	85%	Recueillir des feedbacks réguliers permet d'améliorer continuellement l'accessibilité et de répondre aux besoins réels des utilisateurs.

C.2 - Développement de concepts

C.2 (1) - Concepts du prototype final

Sous-système 1 : Composante de l'algorithme (analyse des fichiers CSV):

(1) Création d'un algorithme qui analysera les fichiers CSV à l'aide de Python

Le langage de programmation Python offre beaucoup de support pour l'automatisation de tâches répétitives, particulièrement en ce qui concerne l'analyse des fichiers CSV. Cela se fait très facilement avec la librairie csv qui peut simplement être importée dans un fichier Python (.py). Des opérations comme "read" et "write" ont déjà été conçues par d'autres programmeurs, alors nous pouvons tout simplement faire un appel à "open" et commencer notre analyse avec les fonctions faisant parti de la librairie csv.

À ce point, il suffit d'écrire un algorithme qui permettra d'accéder à une rangée du document CSV à la fois et extraire les informations importantes. Cela concerne beaucoup d'opérations qui concernent les chaînes de caractères alors, nous profiterons potentiellement des fonctions comme ".join()" ou des concepts comme le "string slicing" qui sont "propriétaires" à Python.

Le fait de choisir Python comme le langage de programmation utilisé pour le concept final améliorera potentiellement la priorisation des tâches (CPX) qui sont associés au ChatBot. Cela s'explique par le fait que Python offre beaucoup de support pour manipuler les chaînes de caractères. Conséquemment, il sera très facile de fournir des résultats facilement lisibles pour (potentiellement) un modèle d'intelligence artificielle qui fera le choix de prioriser les problèmes concernant l'accessibilité qui sont les plus importants.

(2) Création d'un algorithme qui analysera les fichiers CSV à l'aide de JavaScript

D'une manière similaire à Python, JavaScript est un autre langage de programmation qui offre du support pour faciliter l'analyse des fichiers CSV. JavaScript se distingue dans le fait que ce langage peut s'intégrer directement sur un site web (possible à cause d'un API nommé "FileReader"). Cela serait possiblement avantageux si nous décidions de laisser les

utilisateurs du prototype télécharger le fichier CSV sur la page internet qui sera dédié au “ChatBot”; ça sera alors une solution “all in one”.

JavaScript offre les mêmes fonctionnalités de “read” et “write” que Python. Toutefois, la manipulation des chaînes de caractères est plus intuitive avec Python.

Le fait de choisir JavaScript comme le langage de programmation primaire aura l’impact de simplifier l’utilisation (CPX) du prototype final. Cela s’explique par le fait qu’un outil unique sera offert aux utilisateurs. Cet outil prendra la forme d’un site web sur lequel des fichiers CSV peuvent être téléchargés. Par la suite, les utilisateurs pourront interagir avec le “ChatBot” sans avoir à aller avec une autre application.

(3) Création d’un algorithme qui analysera les fichiers CSV à l’aide de Java

Java à une approche légèrement différente en ce qui concerne la manière d’analyser les fichiers CSV. Au lieu d’importer une librairie ou utiliser un API, il suffit d’importer un “Class” intitulé “CSVReader” et “FileReader”. À partir de ce point, il faut simplement créer deux objets associés à ces “Classes” et commencer à analyser toutes les lignes d’un fichier CSV.

En comparaison avec Python et JavaScript, il semble que Java ne se distingue pas dans le traitement de fichier CSV. Il est plus facile de manipuler les chaînes de caractères en Python, qu’en Java et cette dernière ne peut pas s’intégrer directement dans les sites internet comme JavaScript.

Malgré le fait que Java est pire que Python en ce qui concerne le support pour manipuler les chaînes de caractères facilement, Java est quand même très bon. Alors, pour les mêmes raisons que pour Python, l’utilisation de Java permettra potentiellement d’améliorer le choix des composantes non-accessibles à prioriser.

Sous-système 2: Composante du “ChatBot”:

L’implémentation du “ChatBot” varie légèrement d’une langue de programmation à l’autre. Toutes les langues à haut niveau supportent la création d’un tel outil avec des cadres de programmation. Toutefois, l’utilisation de l’IA s’intègre mieux avec Python. Alors, si la solution concerne l’utilisation de modèles d’IA, la langue de programmation idéale sera cette dernière. Il est aussi important de noter que tous les concepts suivants répondent directement à notre dernier CPX, soit le fait que le logiciel comporte un “ChatBot”.

(1) Création d’un “ChatBot” statique et invariable

Du fait que cette solution n’implique pas l’IA, elle peut s’intégrer facilement dans un site internet en utilisant JavaScript. Cela vient directement améliorer la simplicité d’utilisation (CPX) du “ChatBot” car l’analyse du fichier CSV et la communication avec le “ChatBot” pourrait se faire sur la même page internet.

La limite associée à cette solution est le fait que le “ChatBot” ne pourra pas offrir des réponses spécifiques aux questions des utilisateurs. Toutefois, cette approche est très

économique parce que notre solution ne dépend pas d'un service offert par une autre entreprise (alors, aucun frais associé à l'utilisation du prototype). L'entretien de la solution est aussi très facile car l'outil consiste simplement en une série de questions associées à des réponses.

(2) Création d'un "ChatBot" basé sur le modèle "Ollama"

L'avantage du modèle "Ollama" pour la solution finale est le fait que ce dernier peut s'exécuter directement sur l'ordinateur de l'utilisateur. Cela voudrait dire que le "ChatBot" fonctionnera sans l'intervention d'un service extérieur et que l'utilisateur pourrait entraîner le modèle "Ollama" pour être apte dans le contexte de l'accessibilité. Cela rend cette solution plus difficile à utiliser, mais avec l'avantage que le prototype conformera aux normes d'accessibilité et pourrait prioriser les tâches correctement avec un peu d'entraînement.

Le problème associé à l'exécution de "Ollama" sur chaque ordinateur personnel est le fait que ce modèle requiert des cartes graphiques assez puissantes. Le "ChatBot" pourrait alors ne pas fonctionner sans avoir un appareil moderne, conçu pour des calculs intenses. Toutefois, parce que le modèle "Ollama" est "open source", l'utilisation du "ChatBot" sera gratuit.

(3) Création d'un "ChatBot" basé sur le modèle "GPT-4"

Le modèle "GPT-4" offre certainement les meilleurs résultats en termes de la qualité des réponses pour un "ChatBot". Cela est accentué par le fait que nous avons l'habileté d'entraîner un "GPT" pour accomplir une tâche précise. Conséquemment, nous pourrions donner ce "GPT" beaucoup d'information concernant les règles d'accessibilité à prioriser, le format des fichiers CSV à analyser et des méthodes typiques à résoudre des problèmes d'accessibilité pour différentes composantes d'un site internet. Tout cela vient directement supporter la facilité d'utilisation, le fait que le logiciel est conforme aux normes d'accessibilité, la priorisation des tâches ainsi que l'exportation et lecture des fichiers CSV (tous les CPX).

Malheureusement, il existe un problème avec cette solution; il faut payer un abonnement à "GPT-4" (20\$ US par mois) et des frais à chaque fois que le "ChatBot" est utilisé (frais pour l'utilisation de l'API).

(4) Création d'un "ChatBot" basé sur le modèle de "groqcloud"

Le modèle offert par "groqcloud" est un "bon à tout faire". Ce dernier est "open source", plus sécuritaire que la plateforme "Ollama" et offre de très bons résultats. Pour cette solution, le "ChatBot" fera appel à l'API mis en place par "groqcloud" qui a des restrictions très faibles pour le contexte de cette solution.

La limite associée à ce choix est le fait qu'en comparaison avec l'option "GPT", il n'y a aucun moyen d'entraîner un modèle de "groqcloud". Alors, les résultats laisseront un peu à désirer en comparaison avec l'option précédente. Malgré cela, cette solution réussie à répondre à tous les CPX identifiés, juste à un niveau inférieur au "ChatBot" de type "GPT". En comparaison, avec le modèle "Ollama", cette solution gagne en termes de sa facilité d'utilisation, mais perd en ce qui concerne la conformation aux normes d'accessibilité et priorisation des tâches (CPX).

(5) Création d'un "ChatBot" basé sur le modèle de "Gemini 1.5"

Le modèle "Gemini 1.5" est axé sur la flexibilité. Les solutions précédentes avaient des restrictions similaires concernant les limites imposées sur leur API, alors il n'y avait aucun point de les comparer à ce niveau. Ces limites concernent le nombre de "jetons informatiques" que l'API utilise afin de répondre à une question. Groq par exemple, impose une limite de 500 000 "jetons informatiques" par jour; encore pire il faut payer pour utiliser les "jetons informatiques" de ChatGPT-4.

"Gemini 1.5" n'a pas des restrictions strictes comme sa compétition. Effectivement, pour le plan gratuit, il n'y a aucune limite pour les "jetons informatiques" utilisés par jour. Au lieu, les utilisateurs n'ont pas le droit d'utiliser plus que 1 million par minute. Cela donne énormément de flexibilité aux utilisateurs du "ChatBot" car ils auront plus de liberté à demander des questions complexes (qui peuvent aussi concerner des images fournies à l'outil). Cela vient directement répondre aux CPX de la facilité d'utilisation et la priorisation des tâches par le logiciel car les utilisateurs pourraient aider au "ChatBot" à mieux comprendre leur situation particulière (dû au nombre de "jetons informatiques" à leur disposition).

C.2 (2) - Analyse et évaluation des spécifications cibles, avec justification du choix des processus et méthodes utilisées

L'analyse et l'évaluation des spécifications des cibles consistent à examiner la faisabilité des objectifs fixés pour un projet, en fonction des besoins du client et des contraintes techniques. Cela implique de s'assurer que chaque fonctionnalité répond aux attentes des utilisateurs, tout en respectant les normes d'accessibilité dans notre projet.

On a pu définir les spécifications cibles à partir des besoins exprimés et fixer des valeurs idéales et marginales pour chaque métrique. Par exemple, la rapidité du ChatBot est une métrique critique, avec une valeur idéale de réponse instantanée et une valeur marginale un peu plus longue. Ce choix est basé sur l'importance de l'efficacité dans l'usage quotidien des utilisateurs. De même, nous avons choisi des spécifications strictes pour l'intégration avec les lecteurs d'écran, afin de garantir que l'outil soit utilisable par des personnes malvoyantes, répondant ainsi aux exigences des normes de WCAG.

Parmi les autres cibles, la compatibilité avec les technologies d'assistance couvre une assistance particulière pour garantir une utilisation fluide de la plateforme. La personnalisation de l'interface, notamment ce qui concerne les options de réglage du contraste ou de la taille du texte, est une cible essentielle car elle permet d'adapter l'expérience utilisateur en fonction des besoins spécifiques. Enfin, la collecte des commentaires via ChatBot améliorera les fonctionnalités d'accessibilité et représentera un point crucial avec une valeur idéale d'interactivité personnalisée en temps réel.

Les méthodes utilisées pour établir ces spécifications incluent des recherches sur les standards actuels d'accessibilités, une comparaison avec des produits concurrents (ChatGPT, Gemini, Claude Ai, etc....). Ce processus d'analyse nous a permis d'identifier les écarts entre les produits existants et notre solution et de fixer des objectifs réalistes tout en maintenant un haut niveau d'exigence.

Option de concept du sous-système 1							
Critères de sélection	Facteur	Python		Javascript		Java	
Vitesse d'exécution	0.05	3	0.15	3	0.15	4	0.2
Fiabilité	0.35	3	1.05	2	0.7	5	1.75
Support	0.2	5	1	2	0.4	2	0.4
Compatibilité	0.4	5	2	1	0.4	4	1.6
Pointage total		4.2		1.65		3.95	
Classement		1		3		2	

Matrice décisionnelle du sous concept-1

Option de concept du sous-système 2											
Critères de sélection	Facteur	Modèle statique et invariable		Ollama		GPT-4		Groqcloud		Gemini 1.5	
Vitesse de réponse	0.3	5	1.5	5	1.5	4	1.2	5	1.5	5	1.5
Adaptabilité	0.35	1	0.35	5	1.75	5	1.75	1	0.35	5	1.75
Implémentation	0.25	3	0.75	2	0.5	4	1	4	1	4	1
Prix	0.1	5	0.5	5	0.5	1	0.1	3	0.3	5	0.5
Pointage total		3.1		4.25		4.05		3.15		4.75	
Classement		5		2		3		4		1	

Matrice décisionnelle du sous concept-2

C.2 (3) - Choix des solutions prometteuses

En s'appuyant sur les résultats des matrices décisionnelles, nous avons décidé avec notre client que nous allons poursuivre le projet en développant l'algorithme pour analyser les fichiers CSV avec Python et utiliser le modèle "Gemini 1.5" pour la création du "ChatBot".

D'abord, le choix d'utiliser Python est basé sur le fait que la manipulation des chaînes de caractères dans ce langage de programmation est très facile. Il y a aussi beaucoup de support pour la lecture des fichiers CSV déjà inclus dans Python. La majorité des membres de notre équipe sont aussi très aptes à programmer avec Python (plus que les autres langages de programmation identifiés pour ce but), alors ce choix semble être logique.

Ensuite, nous avons décidé de construire notre "ChatBot" avec le "Gemini 1.5" API. Ce choix est justifié par le fait que ce dernier offre beaucoup de flexibilité. Effectivement, "Gemini 1.5" a la capacité d'accepter 15 questions par minutes, qui peuvent tous faire recours à 1 millions de "jetons informatiques". Comme il a été déterminé dans l'échallonnage des produits métriques "Gemini 1.5" gagne dans quasiment toutes les catégories. Nous sommes alors d'avis que ce choix est correct.

C.2 (4) - Développement d'un concept global et justification de l'approche

Le concept global de notre projet repose sur deux éléments clés : l'analyse des fichiers CSV à l'aide de Python et la création d'un chatbot intelligent via l'API "Gemini 1.5". Ces deux aspects sont intégrés de manière à fournir une solution complète et efficace pour répondre aux besoins identifiés par le client.

Nous avons choisi Python pour sa simplicité dans la manipulation des fichiers CSV, notre familiarité avec le langage, et la disponibilité de bibliothèques puissantes pour gérer des données structurées. Le cœur de notre approche repose sur la capacité de Python à traiter de gros volumes de données rapidement et avec une syntaxe claire. L'analyse des fichiers CSV permettra d'extraire des informations pertinentes qui seront ensuite intégrées dans le processus de communication du chatbot.

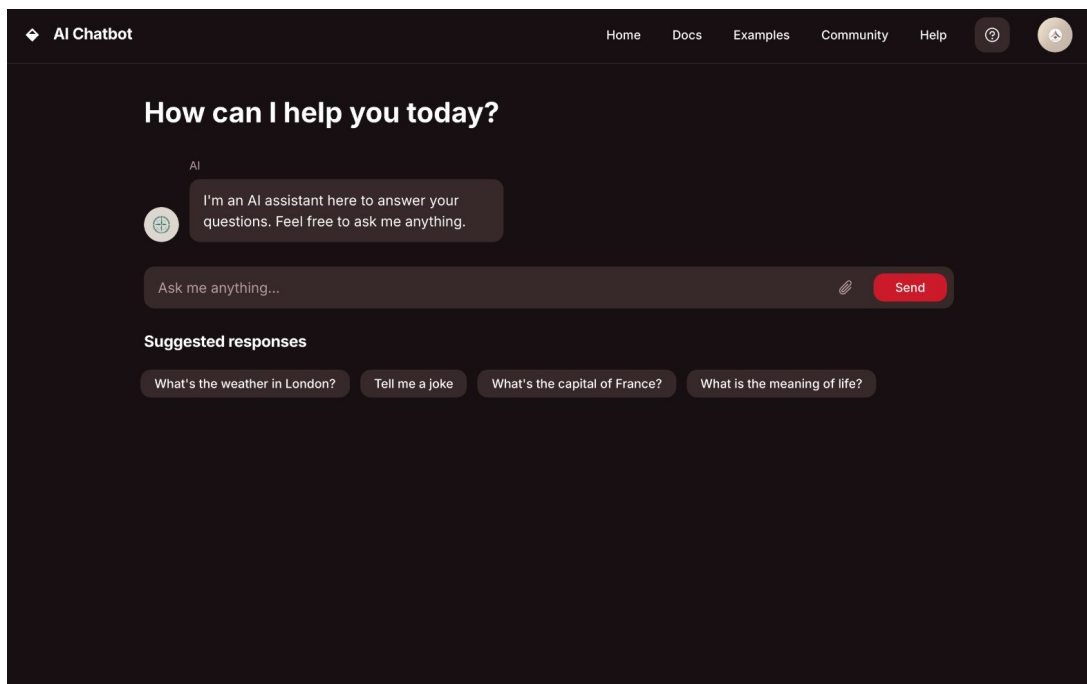
L'utilisation de l'API "Gemini 1.5" pour le développement du chatbot se justifie par sa capacité à gérer efficacement les requêtes complexes, avec un haut débit de traitement des questions. Le chatbot sera capable de répondre rapidement et précisément aux questions basées sur les données extraites des fichiers CSV. Grâce à l'intégration de l'API, le chatbot pourra offrir une expérience utilisateur fluide et interactive, avec des réponses

contextuelles et adaptées aux requêtes. En général, Gemini surpasse les autres modèles d'indilgence artificielle dans plusieurs critères.

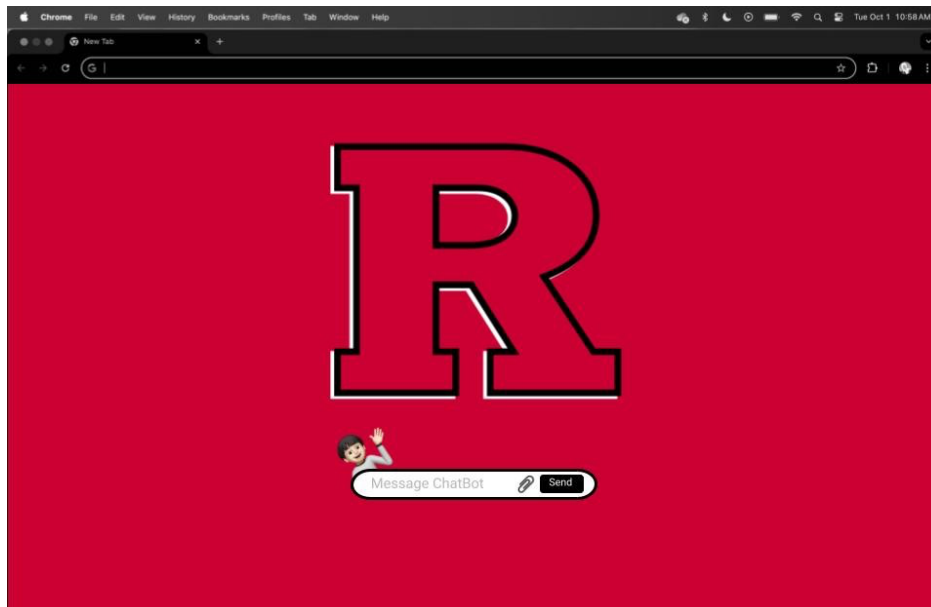
Nous croyons fermement que cette approche (Python + Gemini) maximisera l'efficacité tout en répondant parfaitement aux attentes du client.

C.2 (5) - Représentation visuelle du concept global

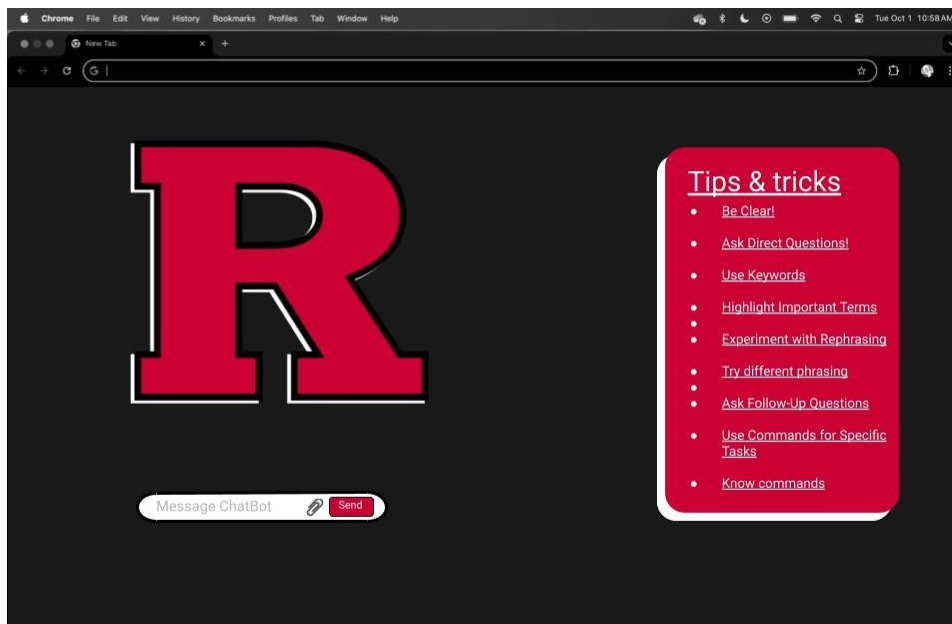
1.



2.



3.



C.2 (6) - Rapport entre le concept final et les spécifications cibles ainsi que les avantages et désavantages liés à cette solution

Le concept final que nous avons choisi consiste de la création d'un algorithme pour analyser les fichiers CSV écrit en Python et l'implémentation d'un "ChatBot" qui utilise le "Gemini 1.5" API. Le concept final, basé sur l'utilisation de l'API Gemini et de Python pour analyser les fichiers CSV, se rapporte aux spécifications cibles de plusieurs manières. Parmi les avantages, on note que Gemini offre des performances élevées grâce à sa capacité à traiter un grand volume de données rapidement, avec une limite de 1 million de jetons par minute, ce qui garantit une analyse efficace des fichiers CSV. De plus, cette

solution offre une grande flexibilité avec une limite de 15 questions par minute dans sa version gratuite, permettant de poser de nombreuses questions ou d'effectuer plusieurs analyses sans contraintes majeures. Python est particulièrement bien adapté à l'analyse des fichiers CSV grâce à ses nombreuses bibliothèques, ce qui facilite le développement des scripts pour une utilisation pratique. Enfin, l'API Gemini est une solution économique, car sa version gratuite répond à la majorité des besoins sans frais supplémentaires, et elle repose sur l'infrastructure fiable de Google, garantissant des résultats rapides et constants.

Cependant, cette solution comporte également des inconvénients. Le principal problème est lié à la confidentialité des données, car la version gratuite de Gemini collecte et stocke les informations pour améliorer les produits Google, ce qui peut poser des soucis en matière de protection des données, notamment vis-à-vis du RGPD (Règle général sur la protection des données). En outre, la dépendance à un service externe comme celui de Google implique que toute interruption ou modification du service pourrait impacter le projet. De plus, bien que Gemini soit puissant, sa gestion peut demander une certaine expertise technique pour adapter l'API aux besoins spécifiques et optimiser son utilisation, en particulier pour des tâches personnalisées. Enfin, il serait nécessaire de passer à la version payante pour éviter la collecte de données ce qui peut engendrer des coûts supplémentaires sur le long terme.

C.2 (7) - Rapport entre le concept final et les facteurs CPX choisis

Notre concept final suit parfaitement les CPX que nous avons choisis:

- L'exportation de fichiers .csv sera assuré par notre concept d'algorithme fonctionnant sur le langage python.
- La conformité aux normes d'accessibilité, la priorisation des tâches importantes et l'utilisation d'un "ChatBot" seront mises à l'œuvre par notre concept de "ChatBot" basé sur l'intelligence artificielle Gemini 1.5 que nous pourrions former et entraîner à ces CPX.
- La facilité d'utilisation quant à elle, découlera de la symbiose entre nos sousconcepts et de la simplicité de notre interface graphique.

C.3 - Plan de projet

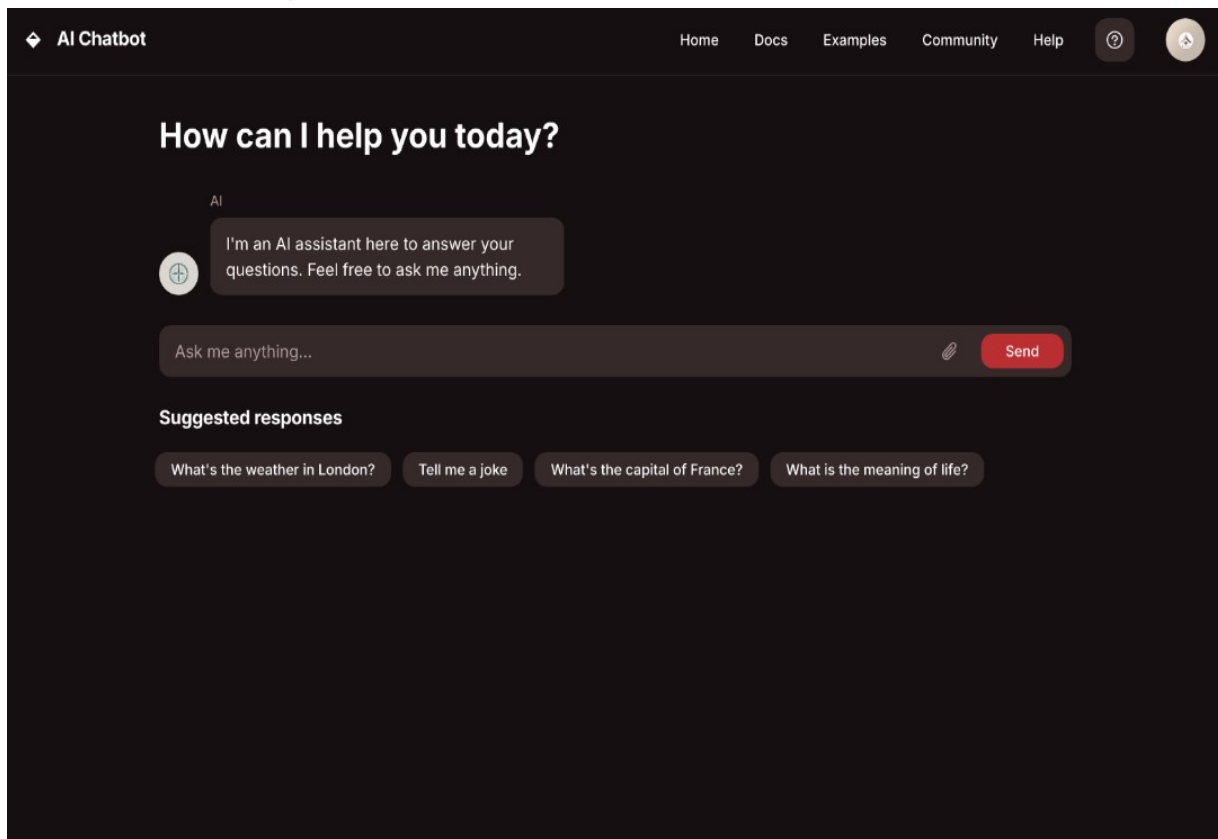
Le plan de projet a été mis à jour. Une image PNG du diagramme de Gantt représentant le plan est attaché à la soumission sur Brightspace.

1. Résumez la rétroaction des clients reçue lors de votre deuxième rencontre client et énoncé clairement ce qui doit être changé ou amélioré par rapport à votre concept.

Lors de la dernière rencontre avec le client, nous avons présenté trois concepts d'interface pour les utilisateurs du "ChatBot", parmi lesquels le client a choisi celle qu'il aimait le plus (c'était l'option #1 que nous avons inclus dans le livrable C). De plus, nous avons discuté des différentes options technologiques pour permettre l'interaction entre un utilisateur et le "ChatBot" qui assistera au processus de rendre les cours sur Canvas plus accessible. Après avoir évalué les forces et les désavantages des options possible avec notre client, nous avons décidé de poursuivre avec l'utilisation de l'API Gemini 1.5 qui a été développé par Google. Le client a confirmé que la faculté à l'Université Rutgers désire avoir une plateforme sécuritaire pour le "ChatBot" sur laquelle les informations importantes sur leurs cours ne peuvent pas être enregistrés par des entreprises extérieures. C'est alors une des raisons principales pour laquelle, nous avons décidé sur Gemini 1.5 car il est très facile à pivoter vers un modèle d'IA similaire offert par Google qui n'utilise pas les questions qu'il reçoit pour apprendre.

Par ailleurs, nous avons obtenu accès à un "dummy course" sur Canvas, ce qui nous permettra d'explorer cette plateforme en profondeur et mieux analyser comment les cours sont exportés en fichiers CSV; cela sera nécessaires pour mieux analyser l'accessibilité des cours.

L'interface utilisateur que le client a choisi:



2. Basé sur votre rencontre de clients, développez un concept détaillé et mis à jour de votre concept qui inclut :

- a. **Pour les prototypes physiques : Une représentation visuelle du concept globale, ainsi que chaque sous-système. Définissez clairement comment chaque sous-système est lié aux autres sous-systèmes (en incluant la quincaillerie et les fils électriques).**

Notre prototype sera entièrement créé à l'aide de programmation. Alors, il n'y a rien à indiquer ici.

b. Pour les prototypes logiciels : Des diagrammes d'interface et des organigrammes du concept global, ainsi que chaque sous-fonction. Définissez clairement comment chaque sous-fonction est liée aux autres sous-fonctions.

Le concept final de notre projet consiste à développer un "ChatBot" utilisant l'API google Gemini 1.5 et analyser des fichiers CSV extraits de Canvas, dans le but d'améliorer l'accessibilité des cours en ligne. Le "ChatBot" sera intégré à une interface utilisateur intuitive permettant aux utilisateurs de poser des questions sur les priorités d'accessibilité et de recevoir des recommandations basées sur des critères d'accessibilité unifiés.

L'API google Gemini 1.5 sera utilisée pour ces capacités d'analyse de texte et d'intelligence conversationnelle. Elle offre des réponses rapides et la gestion de larges volumes de données, jusqu'à 1 million de jetons par minute).

Python sera le langage principal pour l'intégration de l'API ainsi que pour l'analyse des fichiers CSV. Cela permettra d'identifier les points à améliorer en termes d'accessibilité.

Concernant l'interface utilisateur du "ChatBot": Les utilisateurs vont interagir avec le "ChatBot" via une interface graphique simplifiée. Cette interface inclura une zone de texte pour poser des questions et un affichage des réponses fournies par le "ChatBot".

En ce qui concerne l'analyse des fichiers CSV, nous ferons recours à Python pour identifier les éléments qui ne répondent pas aux critères d'accessibilité à partir de la forme exportée des cours. Ce script identifiera les domaines où l'accessibilité peut être améliorée. Le "ChatBot" utilisera les capacités de traitement du langage naturel (NLP) de Google Gemini pour interpréter les questions des utilisateurs et générer des réponses. Aussi, les utilisateurs pourront fournir le "ChatBot" avec des pistes d'améliorations afin d'améliorer les recommandations en matière d'accessibilité. Il proposera des solutions pour affiner les éléments identifiés.

Relations entre sous-fonctions: L'interface utilisateur communiquera avec un API que nous allons créer afin d'analyser un fichier CSV fourni par un utilisateur à l'aide Python ainsi que l'API Gemini 1.5 afin que le "ChatBot" fonctionne. Après que l'analyse des fichiers CSV sera complétée, les résultats sont transmis à Gemini 1.5 qui les utilisera pour répondre aux questions des utilisateurs. Les retours et suggestions générés par l'API sont affichés en temps tout de suite dans l'interface utilisateur qui facilitera ainsi une interaction fluide et intuitive entre l'utilisateur et le "ChatBot".

Liste des matériaux: Pour le développement de l'interface, nous utiliserons une combinaison de HTML, CSS et JavaScript. Ces technologies sont des éléments nécessaires de tous les sites internet. Afin de développer notre API (pour permettre de communiquer entre Python et JavaScript), nous allons utiliser le "Flask Framework", soit une extension de Python. Comme il a été mentionné précédemment, nous allons utiliser Python tout simplement pour l'analyse des fichiers CSV et l'API Gemini 1.5 pour le "ChatBot".

Les dimensions:

1) Interface utilisateur ("ChatBot") :

- **Taille de la boîte de communication :** environ 75% x 10% pixels (de la longueur et hauteur d'une fenêtre sur un ordinateur).

- **Taille du “toolbar” du site internet:** environ 95% x 20% pixels (de la longueur et hauteur d’une fenêtre sur un ordinateur).
- **Taille de la boîte des questions et réponses suggérés.:** environ 60% x 30% pixels (de la longueur et hauteur d’une fenêtre sur un ordinateur).
- **Tous les autres éléments du site internet ont moins d’importance et varieront selon les préférences du client lors des courriels hebdomadaires en notre groupe et lui.**

2) **Analyse CSV (Python) :**

- **Volume des données :** les fichiers CSV extraits de Canvas pourraient contenir entre 100 à 5000 lignes (selon la taille des cours et le nombre d’étudiants).
- **Taille du script d’analyse :** environ 300-500 lignes de code pour l’analyse et le traitement des données CSV.

3) **API Google Gemini :**

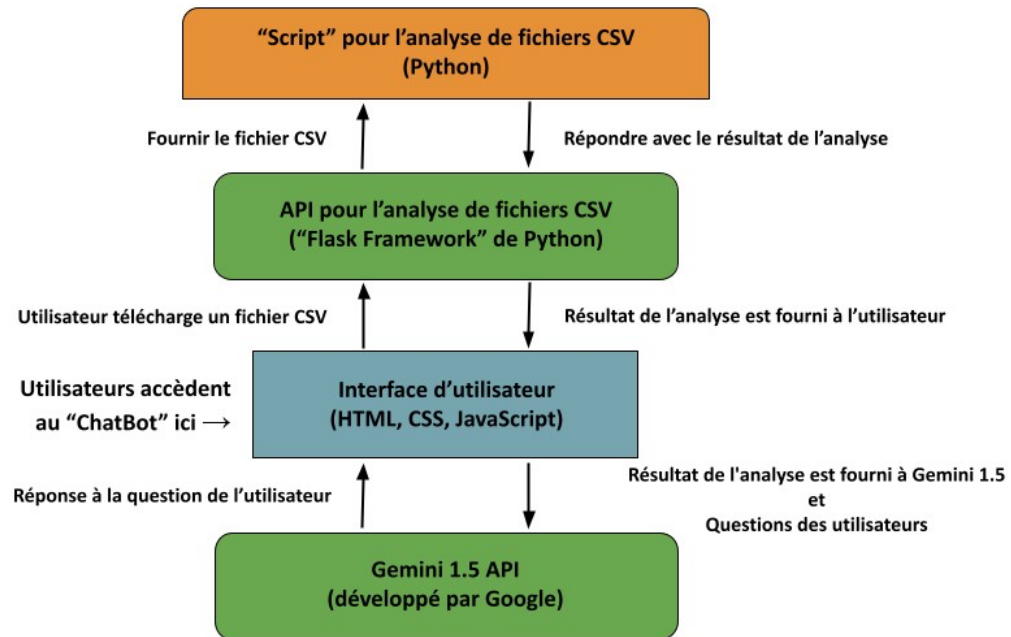
- **Capacité d’analyse :** traitement jusqu’à 1 million de jetons par minute.
- **Complexité d’intégration :** environ 100-200 lignes de code pour intégrer les appels API et gérer les réponses.

4) **API “Flask” :**

- **Fonctions :** Cet API aura qu’une fonction qui est de recevoir un fichier CSV exporté de Canvas et de fournir une analyse de cours en retour.
- **Intégration avec le projet :** Cet API sera exécuté sur nos ordinateurs. Cela nous permettra d’éviter le processus difficile de l’intégrer dans un serveur.

En combinant ces éléments, on peut estimer un total de 500 à 1000 lignes **de** code pour le projet complet, selon la complexité des interactions et le traitement des fichiers CSV.

Diagramme conceptuel des composantes du "LMS Tool"



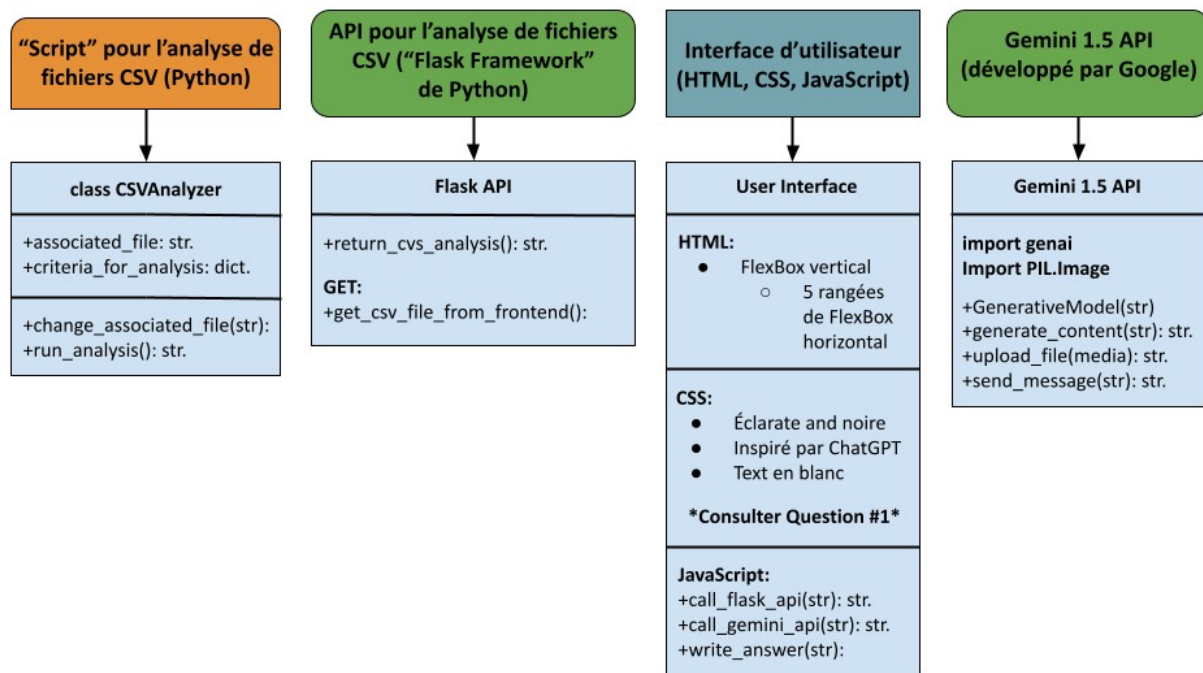
c. Voir :

https://fr.wiki.makerepo.com/wiki/D%C3%A9veloppement_professionnel/Pens%C3%A9e_conceptuelle/Conceptions_d%C3%A9tail%C3%A9es

Nous avons consulté le site internet du makerepo pour compléter ce livrable.

d. Assurez-vous que le niveau de détail de votre concept est suffisamment élevé pour que vous puissiez confier vos documents de conception à une personne externe afin qu'elle puisse fabriquer / assembler / programmer votre concept avec un minimum d'information supplémentaire de votre groupe ! Utilisez au maximum votre AE et GP pour des conseils à ce sujet

Diagramme conceptuel de l'implémentation technique des composantes du "LMS Tool"



3. Expliquez les éléments à prendre en compte pour concevoir ou fabriquer votre concept, en fonction de votre concept détaillée mise à jour, afin de respecter vos facteurs CPX. Certains facteurs sont-ils plus importants que d'autres? Pourquoi?

Pour concevoir et fabriquer le concept de "ChatBot" basé sur l'API Google Gemini et l'analyse de fichiers CSV de Canvas, nous avons pris en compte plusieurs facteurs importants pour garantir le succès du projet. Cependant, après une réflexion approfondie, nous avons décidé de remplacer deux des CPX originaux mentionnés dans le livrable B par des CPX plus adaptés à notre projet : Conception pour la fabrication (CPF) et Conception pour la maintenance (CPM). Ces deux nouveaux facteurs permettent une meilleure adaptation aux besoins spécifiques du développement du logiciel.

1. Conception pour la fabrication (CPF):

Le concept doit être conçu pour une intégration rapide et simple des différents composants logiciels comme l'API Google Gemini, analyse des CSV avec Python et l'interface utilisateur. De plus, le code doit être clair, bien structuré et facile à ajouter de nouvelles fonctionnalités. D'où, les matériaux sont principalement logiciels: Python, API, Canvas, fichiers CSV et JavaScript.

2. Conception pour la Maintenance (CPM):

Le code doit être conçu de manière à pouvoir être maintenu et mis à jour facilement. Par exemple, il est important de s'assurer que l'intégration de nouvelles fonctionnalités, comme l'ajout de critères d'accessibilités, puisse se faire sans avoir à réécrire le code entièrement. De plus, le projet devra inclure des documents de conception clairs et détaillés pour faciliter la maintenance future par d'autres développeurs.

3. Conception pour l'Accessibilité (CPA):

L'objectif du "ChatBot" est de rendre les cours plus accessibles, pour cela l'interface utilisateur doit être implémenter en tenant compte des besoins d'utilisateurs en matière d'accessibilité qui inclura l'utilisation de textes lisibles, couleurs contrastées et fonctions comme la compatibilité avec les lecteurs. En addition, le "ChatBot" doit également adhérer aux critères d'accessibilités du WCAG 2.1 pour que les recommandations soient efficaces.

On peut assumer que le facteur d'accessibilité est plus important que les autres car le but principal est de faciliter l'accessibilité des cours en ligne. L'intégration de critères d'accessibilité unifiés du "ChatBot" est essentielle pour garantir que les cours répondent aux normes requises qui pourra faciliter l'apprentissage des étudiants.

4. Fournissez une liste détaillée des compétences et des ressources dont vous disposez pour vous permettre de créer votre concept. S'il manque des compétences ou des ressources pour compléter votre concept, décrivez comment vous allez les obtenir.

Nous avons établi dans la partie (b) de la question #2 de ce livrable que notre prototype aura 4 composantes technologiques. Nous n'avons pas présentement les connaissances pour créer ces composantes, alors voici notre plan pour apprendre comment concevoir ces dernières:

(1) Création du "script" d'analyse des fichiers CSV Connaissances antérieures:

- (a) Utilisation de Python pour ouvrir et lire des fichiers CSV Concepts à apprendre:
- (b) Apprendre comment l'accessibilité des différentes composantes des cours sur Canvas est évalué → Analyser les critères du WCAG 2.1 et identifier les facteurs les plus importants à considérer pour chaque élément du cours.
- (c) Apprendre comment fournir les résultats de l'analyse dans un format acceptable pour l'API que nous allons construire → Apprendre sur le fonctionnement des API avec le "Flask Framework", particulièrement en ce qui concerne le format du "output" de l'API que nous allons construire.

(2) Création de l'API pour la communication entre le "script" et l'interface d'utilisateur

Connaissances antérieures:

- (d) Les API servent comme une interface entre deux programmes. Ils peuvent recevoir des demandes et fournissent un "output" précis pour chacune des celles-ci.

Concepts à apprendre:

- (e) Apprendre comment construire un API avec le "Flask Framework" qui est essentiellement une extension à Python → Visionner de nombreuses vidéos sur YouTube concernant l'implémentation des API avec "Flask" et comment ces dernières formattent leurs réponses.

- (f) Apprendre comment un “Flask” API peut interagir avec JavaScript (qui sera utilisé pour construire notre interface d'utilisateur) → Faire de la recherche concernant l'utilisation de l'interface “front-end” (JavaScript) comme moyen de communication avec le “back-end” (le “script” pour l'analyse des fichiers CSV par l'entremise du “Flask API”).

(3) Création de l'interface d'utilisateur Connaissances antérieures:

- (g) La majorité des membres de notre équipe ont déjà utilisé HTML, CSS et JavaScript afin de créer une interface d'utilisateur pour un site internet.

Concepts à apprendre:

- (h) Apprendre comment nous pouvons faire interagir l'interface d'utilisateur directement avec les deux API que nous allons utiliser → Il existe de la documentation pour ce but exact sur les sites internet du “Flask Framework” et le “Gemini 1.5 API”. Il faut tout simplement apprendre les étapes à effectuer sur les sites respectifs.
- (i) Apprendre comment représenter dynamiquement les réponses du “ChatBot” lorsque le “output” de l'API “Gemini 1.5” est reçu par notre interface d'utilisateur → Il existe des tutoriels pour la représentation dynamique avec JavaScript sur YouTube; nous aurons besoin de les visionner et apprendre la base de ce concept complexe.

(4) Utilisation de l'API Gemini 1.5

Connaissances antérieures:

- (j) Nous savons que l'API “Gemini 1.5” accepte des questions par l'entremise de demandes qui peuvent se faire directement à partir de quelques lignes de code.

Concepts à apprendre:

- (k) Apprendre comment faire passer des questions à cet API après que notre interface d'utilisateur puisse extraire cette dernière à partir de l'utilisateur → Il existe de nombreux tutoriels sur YouTube qui vont en détail sur comment utiliser l'API à partir d'une application; il faut tenter d'appliquer les stratégies mentionnées dans ces vidéos à notre propre prototype.
- (l) Apprendre comment utiliser l'API “Gemini 1.5” afin d'analyser des images qui sont téléchargées par les utilisateurs dans l'interface → Il existe de la documentation sur le site internet de Google qui explique les capacités en tant que vision pour le modèle de “Gemini 1.5”; nous allons analyser les informations présentées par l'entreprise et appliquer ces dernières à notre prototype.

5. Fournissez une évaluation réaliste du temps requis pour mettre en œuvre votre concept et du temps réel dont votre groupe et ses membres individuels disposent.

Tâche	Sous-tâche	Temps requis	Variation totale de temps requis	Temps réel disponible
Script d'analyse des fichiers .csv	Créer un algorithme qui analyse les fichiers	2-4 jours	3-6 jours	6 jours
	Mettre en place une méthode qui retourne l'analyse dans un format compatible	1-2 jours		
Mise en place de l'API Gemini	Faire le pont entre les questions posées sur l'interface et l'API Gemini	3-5 jours	4-7 jours	9 jours
	Mettre en place la possibilité de fournir une image à l'API Gemini	1-2 jours		
API pour la communication script/interface	Construire l'API	4-6 jours	6-9 jours	9 jours
	Mettre en place la compatibilité de l'API avec javascript	2-3 jours		
Création de l'interface utilisateur	Créer l'interface	2-3 jours	5-9 jours	10 jours
	Faire interagir l'interface avec les deux API	2-4 jours		
	Représenter dynamiquement les réponses du chatbot	1-2 jours		

Notre échéance pour réaliser notre premier concept est le 20 octobre, il est donc essentiel que la charge de travail soit distribuée équitablement entre tous les membres du groupe et que nous débutions les travaux au plus vite.

Il sera nécessaire de commencer simultanément toutes les tâches indépendantes, et que les membres minimisent leurs retards.

De plus si un membre estime avoir besoin d'aide en raison de sa charge personnelle, il doit en informer les autres membres pour trouver une solution permettant de respecter les échéances de l'équipe.

6. Définissez toute autre hypothèse de produit critique qui pourrait affecter votre capacité à mettre en œuvre votre concept. Par exemple : les valeurs acceptables pour une spécification, la disponibilité d'un matériel/une composante, ou une fonctionnalité critique.

Une hypothèse critique pour la mise en œuvre de notre ChatBot est que le système doit être capable de gérer un maximum de $\approx 67\,556$ utilisateurs simultanés sans ralentissement car ceci tiendrait compte de l'ultime cas limite, étant donné qu'il y a à peu près cet exact nombre de personnes à Rutgers actuellement.

De plus, il est essentiel que les serveurs et l'infrastructure technique nécessaires soient disponibles et performants (assez de *bandwidth / bande passante*) pour assurer un accès fluide et rapide d'information et etc. Les caractéristiques principales, tels que sa facilité à l'apprentissage plus accessible et son utilité quand ça viendra aux réponses des questions posées doivent être fonctionnels dès le lancement pour permettre le suivi des performances des utilisateurs.

Programmer en base de données n'est pas simple et pourra se prouver d'être une très grande étape d'apprentissage potentiel que l'on devra éventuellement surmonter.

Enfin, l'interface (front-end) doit aussi être complètement codée avec les langages HTML et JavaScript, et ceci pourrait causer problèmes pour ceux d'entre nous qui ne connaissons que le langage de programmation C.

7. Fournissez une nomenclature des matériaux et des composantes (NDM ou BOM) détaillée pour votre prototype final, qui sera présenté à votre gestionnaire de projet pour approbation et achat. Assurez-vous d'inclure des liens électroniques pour chaque produit dans votre BOM (incluant les items à \$0). On vous donnera jusqu'à un maximum de \$50 ou \$100 (dépendant de votre projet) pour le développement de votre prototype final seulement. Avant de faire des achats, vous devez consulter le guide suivant:

Nomenclature des matériaux

Item	Description	Lien électronique	Prix
Python	Langage de programmation pour développer le backend du chatbot.	https://www.python.org/downloads/	0\$
Flask	Framework web léger pour développer l'interface backend avec Python.	N/A	0\$

Gemini 1.5 API	API permettant d'intégrer les données d'accessibilité et analyses.	https://ai.google.dev/	0\$
HTML/CSS	Technologies pour structurer et styliser l'interface utilisateur du chatbot.	https://developer.mozilla.org/en-US/docs/Learn/HTML	0\$
JavaScript	Langage de programmation pour l'interactivité et le dynamisme de l'interface.	https://developer.mozilla.org/en-US/docs/Web/JavaScript	0\$
GitHub/Git	Plateforme pour gérer le code source et la collaboration.	https://github.com/	0\$
VS Code (IDE)	Environnement de développement intégré pour le codage.	https://code.visualstudio.com/	0\$
COÛT TOTAL			0\$

Conclusion:

Dans ce livrable, nous avons réussi à traduire la rétroaction de notre client dans un concept détaillé pour notre premier prototype. Dans ce concept détaillé, nous avons identifiés les sous-systèmes informatiques et plus tard, nous avons spécifié les exigences pour chacun des ces derniers. En outre, nous avons déterminé toutes les compétences importantes que nous avons besoin de développer afin de concevoir le prototype.

Afin de concevoir le prototype, nous avons créé un plan d'action qui est basé sur une évaluation réaliste du temps de complétion de chacun des sous-systèmes. Cela nous a permis d'identifier les hypothèses critiques qui empêcheront potentiellement la complétion du projet. Dernièrement, nous avons identifié une nomenclature des matériaux et des composantes pour notre premier prototype.

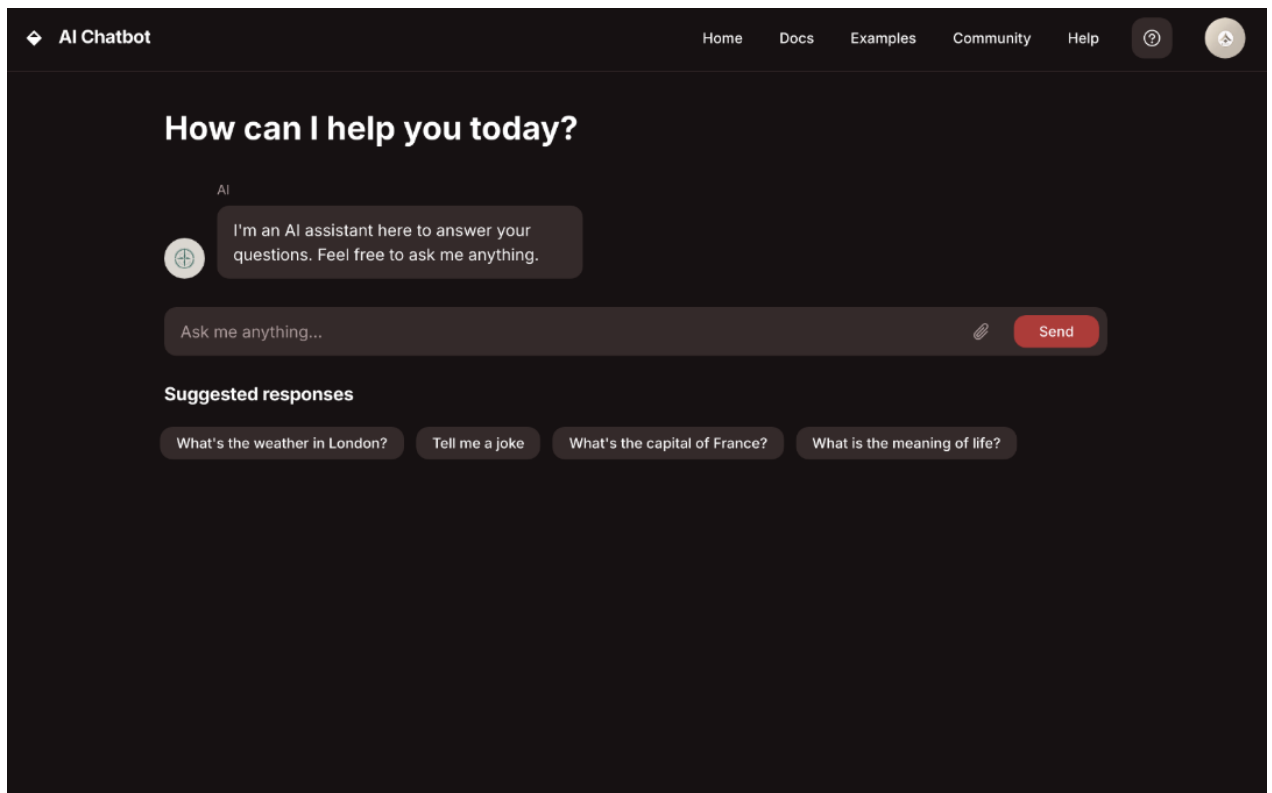
Livrable D : Conception détaillée et BOM

1. Résumez la rétroaction des clients reçue lors de votre deuxième rencontre client et énoncez clairement ce qui doit être changé ou amélioré par rapport à votre concept.

Lors de la dernière rencontre avec le client, nous avons présenté trois concepts d'interface pour les utilisateurs du "ChatBot", parmi lesquels le client a choisi celle qu'il aimait le plus (c'était l'option #1 que nous avons inclus dans le livrable C). De plus, nous avons discuté des différentes options technologiques pour permettre l'interaction entre un utilisateur et le "ChatBot" qui assistera au processus de rendre les cours sur Canvas plus accessible. Après avoir évalué les forces et les désavantages des options possible avec notre client, nous avons décidé de poursuivre avec l'utilisation de l'API Gemini 1.5 qui a été développé par Google. Le client a confirmé que la faculté à l'Université Rutgers désire avoir une plateforme sécuritaire pour le "ChatBot" sur laquelle les informations importantes sur leurs cours ne peuvent pas être enregistrés par des entreprises extérieures. C'est alors une des raisons principales pour laquelle, nous avons décidé sur Gemini 1.5 car il est très facile à pivoter vers un modèle d'IA similaire offert par Google qui n'utilise pas les questions qu'il reçoit pour apprendre.

Par ailleurs, nous avons obtenu accès à un "dummy course" sur Canvas, ce qui nous permettra d'explorer cette plateforme en profondeur et mieux analyser comment les cours sont exportés en fichiers CSV; cela sera nécessaires pour mieux analyser l'accessibilité des cours.

L'interface utilisateur que le client a choisi:



2. Basé sur votre rencontre de clients, développez un concept détaillé et mis à jour de votre concept qui inclut :

a. Pour les prototypes physiques : Une représentation visuelle du concept globale, ainsi que chaque sous-système. Définissez clairement comment chaque sous-système est lié aux autres sous-systèmes (en incluant la quincaillerie et les fils électriques).

Notre prototype sera entièrement créé à l'aide de programmation. Alors, il n'y a rien à indiquer ici.

b. Pour les prototypes logiciels : Des diagrammes d'interface et des organigrammes du concept global, ainsi que chaque sous-fonction. Définissez clairement comment chaque sous-fonction est liée aux autres sous-fonctions.

Le concept final de notre projet consiste à développer un "ChatBot" utilisant l'API google Gemini 1.5 et analyser des fichier CSV extraits de Canvas, dans le but d'améliorer l'accessibilité des cours en ligne. Le "ChatBot" sera intégré à une interface utilisateur intuitive permettant aux utilisateurs de poser des questions sur les priorités d'accessibilité et de recevoir des recommandations basées sur des critères d'accessibilité unifiés.

L'API google Gemini 1.5 sera utilisée pour ces capacités d'analyse de texte et d'intelligence conversationnelle. Elle offre des réponses rapides et la gestion de larges volumes de données, jusqu'à 1 million de jetons par minute).

Python sera le langage principal pour l'intégration de l'API ainsi que pour l'analyse des fichiers CSV. Cela permettra d'identifier les points à améliorer en termes d'accessibilité.

Concernant l'interface utilisateur du "ChatBot": Les utilisateurs vont interagir avec le "ChatBot" via une interface graphique simplifiée. Cette interface inclura une zone de texte pour poser des questions et un affichage des réponses fournies par le "ChatBot".

En ce qui concerne l'analyse des fichiers CSV, nous ferons recours à Python pour identifier les éléments qui ne répondent pas aux critères d'accessibilité à partir de la forme exportée des cours. Ce script identifiera les domaines où l'accessibilité peut être améliorée. Le "ChatBot" utilisera les capacités de traitement du langage naturel (NLP) de Google Gemini pour interpréter les questions des utilisateurs et générer des réponses. Aussi, les utilisateurs pourront fournir le "ChatBot" avec des pistes d'améliorations afin d'améliorer les recommandations en matière d'accessibilité. Il proposera des solutions pour affiner les éléments identifiés.

Relations entre sous-fonctions: L'interface utilisateur communiquera avec un API que nous allons créer afin d'analyser un fichier CSV fourni par un utilisateur à l'aide Python ainsi que l'API Gemini 1.5 afin que le "ChatBot" fonctionne. Après que l'analyse des fichiers CSV sera complété, les résultats sont transmis à Gemini 1.5 qui les utilisera pour répondre aux questions des utilisateurs. Les retours et suggestions générés par l'API sont affichés en temps tout de suite dans l'interface utilisateur qui facilitera ainsi une interaction fluide et intuitive entre l'utilisateur et le "ChatBot".

Liste des matériaux: Pour le développement de l'interface, nous utiliserons une combinaison de HTML, CSS et JavaScript. Ces technologies sont des éléments nécessaires de tous les sites internet. Afin de développer notre API (pour permettre de communiquer entre Python et JavaScript), nous allons utiliser le "Flask Framework", soit une extension de Python. Comme il a été mentionné précédemment, nous allons utiliser Python tout simplement pour l'analyse des fichiers CSV et l'API Gemini 1.5 pour le "ChatBot".

Les dimensions:

1. Interface utilisateur ("ChatBot") :

- **Taille de la boîte de communication :** environ 75% x 10% pixels (de la longueur et hauteur d'une fenêtre sur un ordinateur).
- **Taille du "toolbar" du site internet:** environ 95% x 20% pixels (de la longueur et hauteur d'une fenêtre sur un ordinateur).
- **Taille de la boîte des questions et réponses suggérés.:** environ 60% x 30% pixels (de la longueur et hauteur d'une fenêtre sur un ordinateur).

- **Tous les autres éléments du site internet ont moins d'importance et varieront selon les préférences du client lors des courriels hebdomadaires en notre groupe et lui.**

2. **Analyse CSV (Python) :**

- **Volume des données :** les fichiers CSV extraits de Canvas pourraient contenir entre 100 à 5000 lignes (selon la taille des cours et le nombre d'étudiants).
- **Taille du script d'analyse :** environ 300-500 lignes de code pour l'analyse et le traitement des données CSV.

3. **API Google Gemini :**

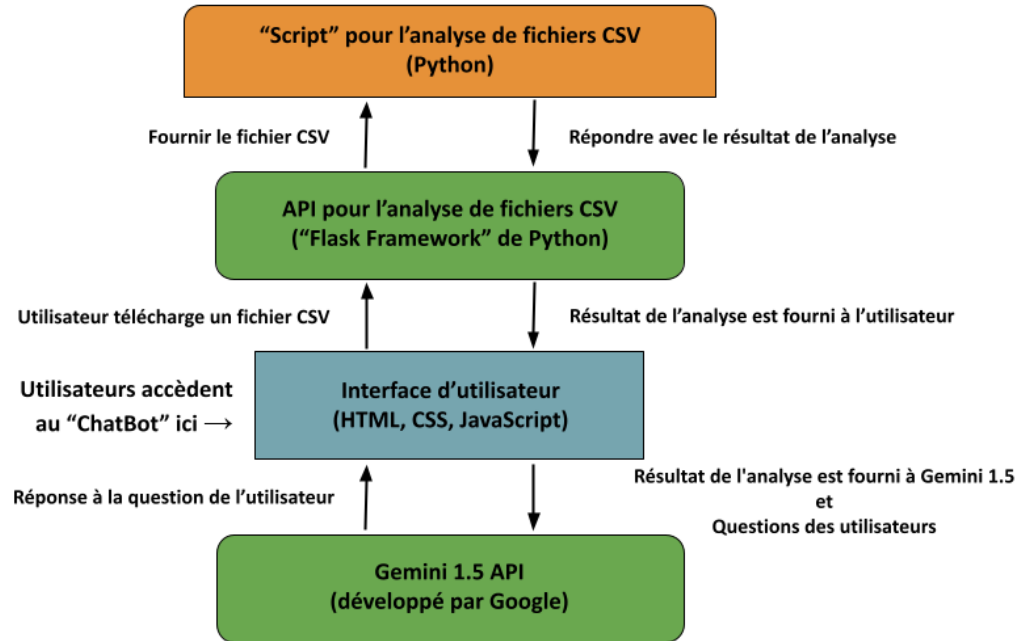
- **Capacité d'analyse :** traitement jusqu'à 1 million de jetons par minute.
- **Complexité d'intégration :** environ 100-200 lignes de code pour intégrer les appels API et gérer les réponses.

4. **API "Flask" :**

- **Fonctions :** Cet API aura qu'une fonction qui est de recevoir un fichier CSV exporté de Canvas et de fournir une analyse de cours en retour.
- **Intégration avec le projet :** Cet API sera exécuté sur nos ordinateurs. Cela nous permettra d'éviter le processus difficile de l'intégrer dans un serveur.

En combinant ces éléments, on peut estimer un total de 500 à 1000 lignes **de** code pour le projet complet, selon la complexité des interactions et le traitement des fichiers CSV.

Diagramme conceptuel des composantes du "LMS Tool"



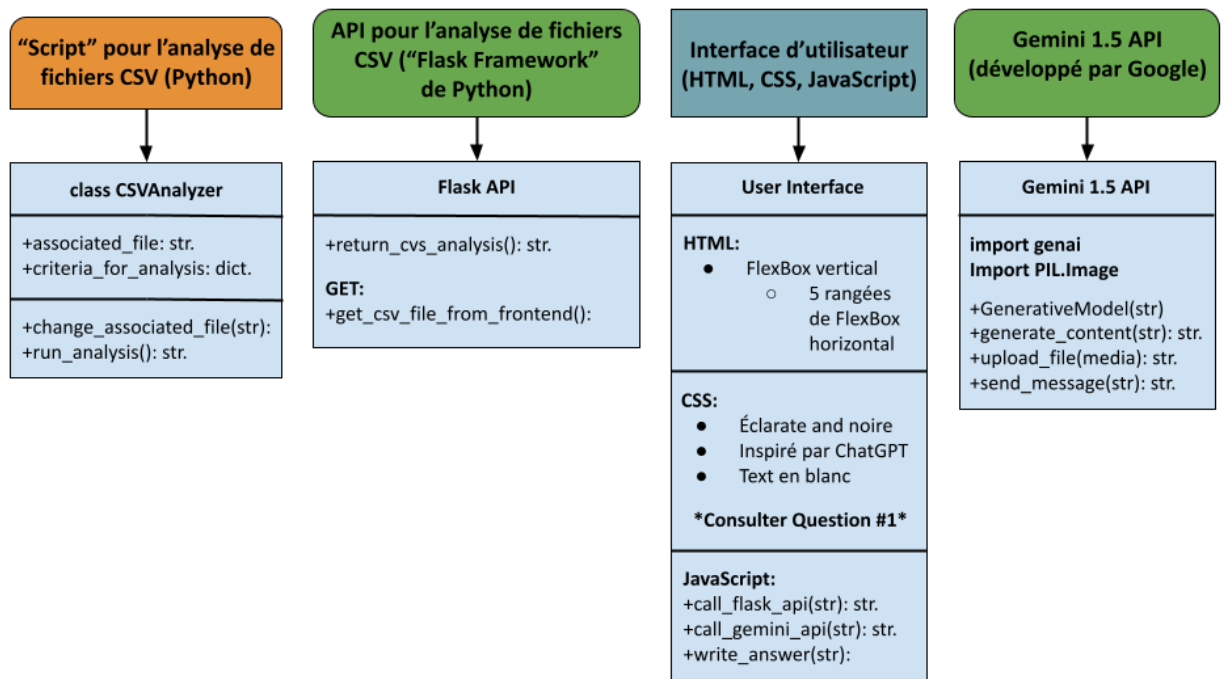
c. Voir :

https://fr.wiki.makerepo.com/wiki/D%C3%A9veloppement_professionnel/Pens%C3%A9e_conceptuelle/Conceptions_d%C3%A9taill%C3%A9es

Nous avons consulté le site internet du makerepo pour compléter ce livrable.

d. Assurez-vous que le niveau de détail de votre concept est suffisamment élevé pour que vous puissiez confier vos documents de conception à une personne externe afin qu'elle puisse fabriquer / assembler / programmer votre concept avec un minimum d'information supplémentaire de votre groupe ! Utilisez au maximum votre AE et GP pour des conseils à ce sujet

Diagramme conceptuel de l'implémentation technique des composantes du "LMS Tool"



3. Expliquez les éléments à prendre en compte pour concevoir ou fabriquer votre concept, en fonction de votre concept détaillée mise à jour, afin de respecter vos facteurs CPX. Certains facteurs sont-ils plus importants que d'autres? Pourquoi?

Pour concevoir et fabriquer le concept de "ChatBot" basé sur l'API Google Gemini et l'analyse de fichiers CSV de Canvas, nous avons pris en compte plusieurs facteurs importants pour garantir le succès du projet. Cependant, après une réflexion approfondie, nous avons décidé de remplacer deux des CPX originaux mentionnés dans le livrable B par des CPX plus adaptés à notre projet : Conception pour la fabrication (CPF) et Conception pour la maintenance (CPM). Ces deux nouveaux facteurs permettent une meilleure adaptation aux besoins spécifiques du développement du logiciel.

1. Conception pour la fabrication (CPF):

Le concept doit être conçu pour une intégration rapide et simple des différents composants logiciels comme l'API Google Gemini, analyse des CSV avec Python et l'interface utilisateur. De plus, le code doit être clair, bien structuré et facile à ajouter de nouvelles fonctionnalités. D'où, les matériaux sont principalement logiciels: Python, API, Canvas, fichiers CSV et JavaScript.

2. Conception pour la Maintenance (CPM):

Le code doit être conçu de manière à pouvoir être maintenu et mis à jour facilement. Par exemple, il est important de s'assurer que l'intégration de nouvelles fonctionnalités, comme l'ajout de critères d'accessibilités, puisse se faire sans avoir à réécrire le code entièrement. De plus, le projet devra inclure des documents de conception clairs et détaillés pour faciliter la maintenance future par d'autres développeurs.

3. Conception pour l'Accessibilité (CPA):

L'objectif du "ChatBot" est de rendre les cours plus accessibles, pour cela l'interface utilisateur doit être implémenter en tenant compte des besoins d'utilisateurs en matière d'accessibilité qui inclura l'utilisation de textes lisibles, couleurs contrastées et fonctions comme la compatibilité avec les lecteurs. En addition, le "ChatBot" doit également adhérer aux critères d'accessibilités du WCAG 2.1 pour que les recommandations soient efficaces.

On peut assumer que le facteur d'accessibilité est plus important que les autres car le but principal est de faciliter l'accessibilité des cours en ligne. L'intégration de critères d'accessibilité unifiés du "ChatBot" est essentielle pour garantir que les cours répondent aux normes requises qui pourra faciliter l'apprentissage des étudiants.

4. Fournissez une liste détaillée des compétences et des ressources dont vous disposez pour vous permettre de créer votre concept. S'il manque des compétences ou des ressources pour compléter votre concept, décrivez comment vous allez les obtenir.

Nous avons établi dans la partie (b) de la question #2 de ce livrable que notre prototype aura 4 composantes technologiques. Nous n'avons pas présentement les connaissances pour créer ces composantes, alors voici notre plan pour apprendre comment concevoir ces dernières:

1. Création du "script" d'analyse des fichiers CSV

Connaissances antérieures:

- a. Utilisation de Python pour ouvrir et lire des fichiers CSV

Concepts à apprendre:

- b. Apprendre comment l'accessibilité des différentes composantes des cours sur Canvas est évalué → Analyser les critères du WCAG 2.1 et identifier les facteurs les plus importants à considérer pour chaque élément du cours.
- c. Apprendre comment fournir les résultats de l'analyse dans un format acceptable pour l'API que nous allons construire → Apprendre sur le fonctionnement des API avec le "Flask Framework", particulièrement en ce qui concerne le format du "output" de l'API que nous allons construire.

2. **Création de l'API pour la communication entre le "script" et l'interface d'utilisateur**

Connaissances antérieures:

- d. Les API servent comme une interface entre deux programmes. Ils peuvent recevoir des demandes et fournissent un "output" précis pour chacune des celles-ci.

Concepts à apprendre:

- e. Apprendre comment construire un API avec le "Flask Framework" qui est essentiellement une extension à Python → Visionner de nombreuses vidéos sur YouTube concernant l'implémentation des API avec "Flask" et comment ces dernières formattent leurs réponses.
- f. Apprendre comment un "Flask" API peut interagir avec JavaScript (qui sera utilisé pour construire notre interface d'utilisateur) → Faire de la recherche concernant l'utilisation de l'interface "front-end" (JavaScript) comme moyen de communication avec le "back-end" (le "script" pour l'analyse des fichiers CSV par l'entremise du "Flask API").

3. **Création de l'interface d'utilisateur**

Connaissances antérieures:

- g. La majorité des membres de notre équipe ont déjà utilisé HTML, CSS et JavaScript afin de créer une interface d'utilisateur pour un site internet.

Concepts à apprendre:

- h. Apprendre comment nous pouvons faire interagir l'interface d'utilisateur directement avec les deux API que nous allons utiliser → Il existe de la documentation pour ce but exact sur les sites internet du "Flask Framework" et le "Gemini 1.5 API". Il faut tout simplement apprendre les étapes à effectuer sur les sites respectifs.
- i. Apprendre comment représenter dynamiquement les réponses du "ChatBot" lorsque le "output" de l'API "Gemini 1.5" est reçu par notre interface d'utilisateur → Il existe des tutoriels pour la représentation dynamique avec JavaScript sur YouTube; nous aurons besoins de les visionner et apprendre la base de ce concept complexe.

4. **Utilisation de l'API Gemini 1.5**

Connaissances antérieures:

- j. Nous savons que l'API "Gemini 1.5" accepte des questions par l'entremise de demandes qui peuvent se faire directement à partir de quelques lignes de code.

Concepts à apprendre:

- k. Apprendre comment faire passer des questions à cet API après que notre interface d'utilisateur puisse extraire cette dernière à partir de l'utilisateur → Il existe de nombreux tutoriels sur YouTube qui vont en détail sur comment utiliser l'API à partir d'une application; il faut tenter d'appliquer les stratégies mentionnées dans ces vidéos à notre propre prototype.
- l. Apprendre comment utiliser l'API "Gemini 1.5" afin d'analyser des images qui sont téléchargés par les utilisateurs dans l'interface → Il existe de la documentation sur le site internet de Google qui explique les capacités en tant que vision pour le modèle de "Gemini 1.5"; nous allons analyser les informations présentées par l'entreprise et appliquer ces dernières à notre prototype.

5. Fournissez une évaluation réaliste du temps requis pour mettre en œuvre votre concept et du temps réel dont votre groupe et ses membres individuels disposent.

Tâche	Sous-tâche	Temps requis	Variation totale de temps requis	Temps réel disponible
Script d'analyse des fichiers .csv	Créer un algorithme qui analyse les fichiers	2-4 jours	3-6 jours	6 jours
	Mettre en place une méthode qui retourne l'analyse dans un format compatible	1-2 jours		
Mise en place de l'API Gemini	Faire le pont entre les questions posées sur l'interface et l'API Gemini	3-5 jours	4-7 jours	9 jours
	Mettre en place la possibilité de	1-2 jours		

	fournir une image à l'API Gemini			
API pour la communication script/interface	Construire l'API	4-6 jours	6-9 jours	9 jours
	Mettre en place la compatibilité de l'API avec javascript	2-3 jours		
Création de l'interface utilisateur	Créer l'interface	2-3 jours	5-9 jours	10 jours
	Faire interagir l'interface avec les deux API	2-4 jours		
	Représenter dynamiquement les réponses du chatbot	1-2 jours		

Notre échéance pour réaliser notre premier concept est le 20 octobre, il est donc essentiel que la charge de travail soit distribuée équitablement entre tous les membres du groupe et que nous débutions les travaux au plus vite.

Il sera nécessaire de commencer simultanément toutes les tâches indépendantes, et que les membres minimisent leurs retards.

De plus si un membre estime avoir besoin d'aide en raison de sa charge personnelle, il doit en informer les autres membres pour trouver une solution permettant de respecter les échéances de l'équipe.

6. Définissez toute autre hypothèse de produit critique qui pourrait affecter votre capacité à mettre en œuvre votre concept. Par exemple : les valeurs acceptables pour une spécification, la disponibilité d'un matériel/une composante, ou une fonctionnalité critique.

Une hypothèse critique pour la mise en œuvre de notre ChatBot est que le système doit être capable de gérer un maximum de $\approx 67\,556$ utilisateurs simultanés sans ralentissement car ceci tiendrait compte de l'ultime cas limite, étant donné qu'il y a à peu près cet exact nombre de personnes à Rutgers actuellement.

De plus, il est essentiel que les serveurs et l'infrastructure technique nécessaires soient disponibles et performants (assez de *bandwidth / bande passante*) pour assurer un accès fluide et rapide d'information et etc. Les caractéristiques principales, tels que sa facilité à l'apprentissage plus accessible et son utilité quand ça viendra aux réponses des questions posées doivent être fonctionnels dès le lancement pour permettre le suivi des performances des utilisateurs.

Programmer en base de données n'est pas simple et pourra se prouver d'être une très grande étape d'apprentissage potentiel que l'on devra éventuellement surmonter.

Enfin, l'interface (front-end) doit aussi être complètement codée avec les langages HTML et JavaScript, et ceci pourrait causer problèmes pour ceux d'entre nous qui ne connaissons que le langage de programmation C.

7. Fournissez une nomenclature des matériaux et des composantes (NDM ou BOM) détaillée pour votre prototype final, qui sera présenté à votre gestionnaire de projet pour approbation et achat. Assurez-vous d'inclure des liens électroniques pour chaque produit dans votre BOM (incluant les items à \$0). On vous donnera jusqu'à un maximum de \$50 ou \$100 (dépendant de votre projet) pour le développement de votre prototype final seulement. Avant de faire des achats, vous devez consulter le guide suivant:

Nomenclature des matériaux

Item	Description	Lien électronique	Prix
Python	Langage de programmation pour développer le backend du chatbot.	https://www.python.org/downloads/	0\$
Flask	Framework web léger pour développer l'interface backend avec Python.	N/A	0\$
Gemini 1.5 API	API permettant d'intégrer les données	https://ai.google.dev/	0\$

		d'accessibilité et analyses.		
S	HTML/CS	Technologie s pour structurer et styliser l'interface utilisateur du chatbot.	https://developer.mozilla.org/en-US/docs/Learn/HTML	0\$
t	JavaScrip	Langage de programmation pour l'interactivité et le dynamisme de l'interface.	https://developer.mozilla.org/en-US/docs/Web/JavaScript	0\$
Git	GitHub/	Plateforme pour gérer le code source et la collaboration.	https://github.com/	0\$
(IDE)	VS Code	Environnem ent de développement intégré pour le codage.	https://code.visualstudio.com/	0\$
COÛT TOTAL				0\$

Conclusion:

Dans ce livrable, nous avons réussi à traduire la rétroaction de notre client dans un concept détaillé pour notre premier prototype. Dans ce concept détaillé, nous avons identifiés les sous-systèmes informatiques et plus tard, nous avons spécifié les exigences pour chacun des ces derniers. En outre, nous avons déterminé toutes les compétences importantes que nous avons besoin de développer afin de concevoir le prototype.

Afin de concevoir le prototype, nous avons créé un plan d'action qui est basé sur une évaluation réaliste du temps de complétion de chacun des sous-systèmes. Cela nous a permis d'identifier les hypothèses critiques qui empêcheront potentiellement la complétion du projet. Dernièrement, nous avons identifié une nomenclature des matériaux et des composantes pour notre premier prototypes.

Livrable E – Prototype 1, Présentation et Rétroaction

Introduction :

Ce livrable concerne le développement de notre premier prototype ainsi que la création d'une présentation formelle qui porte sur tout le progrès que nous avons fait jusqu'à ce point.

D'abord, nous allons expliquer nos hypothèses critiques et la façon que nous allons tester notre prototype. Par la suite, nous décrivons notre premier prototype de façon exhaustive, en précisant nos choix pour l'interface d'utilisateur et expliquant l'implémentation des composantes informatiques. Dans la section E.1.3, nous présentons les résultats des tests que nous avons effectués sur le prototype. Dernièrement, dans la section E.2, nous résumons le contenu de notre présentation.

E.1 - Prototype 1

E.1 (1) - Hypothèses critiques

Après avoir débuté notre conception, nous avons vite réalisé que les hypothèses critiques définies dans le livrable D devaient être réévaluées. Effectivement, parce que notre budget est limité, il est impossible de procurer un serveur dédié à notre client (ce dernier coûte plus que 100\$ par mois). Au lieu, nous planifions pivoter notre approche et rendre les processus informatiques, nécessaires pour fonctionnement de notre prototype complètement, associés à l'ordinateur de l'utilisateur. Cela nous permettra d'éviter les frais de serveur et d'offrir notre prototype à un nombre illimité d'utilisateurs. Toutefois, jusqu'à présent notre hypothèse critique concernant l'utilisation des technologies "Frontend" (comme HTML, CSS et JavaScript) demeure appropriée.

Validation de l'hypothèse sur l'exécution locale (sur l'ordinateur des utilisateurs) du prototype :

Afin de valider cette hypothèse, nous allons d'abord besoin de déterminer les spécifications minimums d'un ordinateur qui sera capable d'exécuter notre prototype. Il faudrait aussi considérer le temps nécessaire pour un utilisateur à partir de son ordinateur de :

(1) Faire recours au script d'analyse CSV et recevoir sa réponse

Afin d'évaluer la vitesse du script d'analyse CSV, il faudra utiliser la librairie "time" offerte par Python. Celle-ci permettra de chronométrer le temps entre le téléchargement du fichier CSV et la réponse qui contient le résultat de l'analyse. Afin d'avoir un bon aperçu du temps d'exécution du script d'analyse CSV, 10 tests seront effectués pour chacune des 3 variétés de fichiers CSV suivants : petit (moins que 500 rangées), moyen (moins que 1000 rangées) et grand (moins que 5000 rangées). Par la suite, nous pourrons établir un rapport général représentant : Nombre de rangées : Nombre de secondes.

(2) Communiquer avec le Gemini 1.5 API et recevoir la réponse.

À cause que la communication avec le Gemini 1.5 API sera effectué par l'entremise de code écrit en JavaScript, nous ferons recours à la librairie "Date". Comme la librairie "time" de Python, nous pourrons chronométrer le temps de réponse de l'API. Pour analyser la vitesse de ce dernier il faudra considérer séparer les questions des utilisateurs en deux groupes : questions textuelles et questions textuelles + image de support. Ces dernières sont beaucoup plus complexes et alors, prennent plus de temps à analyser par le modèle Gemini. Pour chacun des groupes, nous allons identifier 3 types de questions : petites questions (moins que 25 mots), questions de taille moyenne (moins que 50 mots) et grandes questions (moins que 100 mots). 10 essais pour chacune des trois catégories qui appartiennent à chacune des deux groupes seront effectués. Dans le cas du groupe qui contient des images, les questions seront accompagnées par des captures d'écran qui supporteront les questions.

Dernièrement, il faudrait élaborer une liste de toutes les technologies à installer sur un ordinateur afin qu'il puisse exécuter le prototype. Chacune de ces technologies devrait aussi être associée une valeur représentant sa difficulté d'installation.

Les tests mentionnés ci-dessus auront pour but d'évaluer les **CPX suivants : facilité d'utilisation** ainsi que **l'exportation et lecture des fichiers CSV**. D'abord, selon nous, si la vitesse de l'analyse des fichiers CSV et des réponses du Gemini 1.5 API est rapide, notre prototype sera facile à utiliser. Cela devient plus évident si on considère que notre prototype fonctionnera quand même sur les ordinateurs plus anciens. Effectivement, les limitations associées à leur puissance seront minimisées avec notre prototype et alors, les utilisateurs pourront recevoir le résultat de l'analyse et communiquer avec le "ChatBot" sans des délais prolongés. Aussi, si nous réussissons à analyser les fichiers CSV des cours à partir d'un site internet, nous avons nécessairement réussi à exporter ceux-ci à notre script d'analyse. Dans ce cas, nous avons clairement répondu à notre CPX qui concerne l'exportation et la lecture de ces fichiers.

Validation de l'hypothèse sur l'utilisation de HTML, CSS et JavaScript pour l'élaboration du prototype :

Afin de valider cette hypothèse, il suffit uniquement de considérer si, par la fin de la conception des prototypes, nous avons réussi à utiliser HTML, CSS et JavaScript pour la

conception. Cela sera un défi car la majorité des membres de notre équipe n'a pas déjà utilisé ces technologies.

Les facteurs CPX suivants se rapportent à cette hypothèse critiques : **la facilité d'utilisation** et le fait que notre produit **comporte un "ChatBot"**. Tout d'abord, les technologies de HTML, CSS et JavaScript ont été conçues afin de donner aux programmeurs la capacité de créer des interfaces d'utilisateurs interactives, attrayantes et faciles dans leur utilisation. Alors, en utilisant ces technologies nous aurons tous les outils à notre disposition afin de présenter un prototype qui sera très intuitif à utiliser. En plus de cela, parce que JavaScript a la capacité de manipuler les éléments HTML et les styles CSS, c'est une plateforme parfaite afin de créer une interface d'un ChatBot. Par exemple, à chaque fois qu'un utilisateur reçoit une réponse du Gemini API, nous pouvons avec JavaScript changer dynamiquement l'état du site internet et y insérer un élément textuel qui contient la réponse.

E.1 (2) - Premier(s) prototype(s)

Notre premier prototype utilise Flask pour créer une application web qui permet aux utilisateurs de télécharger des fichiers CSV extraits de cours sur Canvas, afin de les analyser en termes d'accessibilité. L'objectif principal est de détecter des problèmes d'accessibilité en fonction des recommandations d'Ally.

L'architecture repose sur quelques composants clés. D'abord, Flask, un framework léger pour Python, gère la logique backend de l'application. Il permet de définir plusieurs routes, comme la page d'accueil, des sections comme "community", "home", "docs" et "Exemples", et surtout une route pour le téléchargement de fichiers, qui est essentielle à l'analyse.

Lorsque l'utilisateur télécharge un fichier CSV via l'interface web, Flask vérifie si le fichier est bien du type CSV. Si c'est le cas, il est stocké dans un répertoire temporaire sur le serveur. Si le fichier n'est pas un CSV, un message d'erreur est renvoyé, et l'utilisateur est invité à essayer de nouveau.

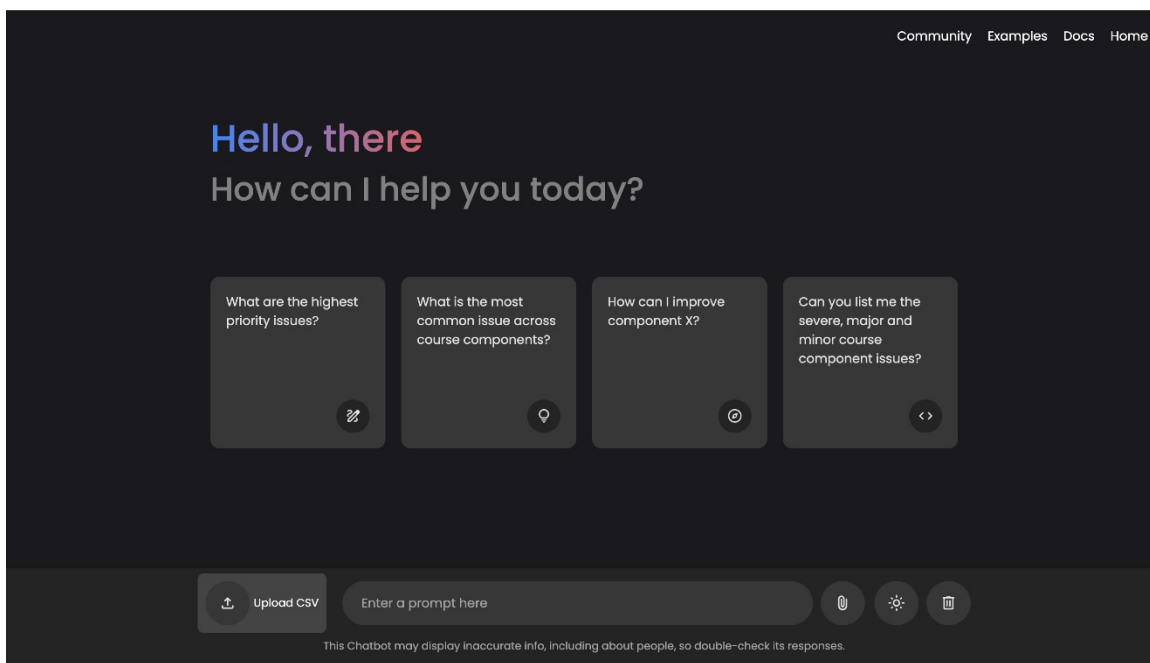
Ensuite, le fichier CSV est pris en charge par une classe que tu as nommée CSVAnalyser. Cette classe lit le contenu du fichier et analyse chaque ligne à la recherche de problèmes d'accessibilité. Les critères d'accessibilité que tu as définis peuvent inclure des problèmes tels que des liens hypertextes brisés, des images sans descriptions alternatives (ce qui pose des problèmes pour les personnes ayant une déficience visuelle), un contraste de couleur insuffisant pour les textes, ou encore des en-têtes mal formatés dans les tableaux.

Une fois que l'analyse est effectuée, la classe "CSVAnalyser" énumérera les problèmes détectés selon leur niveau de gravité : problèmes graves, majeurs ou mineurs. Les problèmes graves pourraient inclure des erreurs critiques, comme des images sans description, qui compromettent fortement l'accessibilité. Les problèmes majeurs concernent des aspects importants mais pas nécessairement bloquants, tandis que les

problèmes mineurs représentent des défauts d'accessibilité qui, bien que non essentiels, devraient être corrigés pour améliorer l'expérience utilisateur.

Les résultats de l'analyse sont ensuite renvoyés sous la forme d'un fichier JSON, un format de données facile à traiter, qui contient la liste des problèmes détectés avec une description détaillée de chacun et leur niveau de priorité. Ce fichier JSON peut être exploité par le frontend de l'application pour afficher ces informations de manière lisible, comme sous forme de tableaux, graphiques ou listes.

L'interface utilisateur du prototype se compose d'une mise en page claire et conviviale, conçue pour faciliter l'interaction de l'utilisateur avec le chatbot. En haut de la page, une barre de navigation propose des liens vers différentes sections, telles que la communauté, les exemples, la documentation et l'accueil, permettant aux utilisateurs de naviguer facilement. L'en-tête présente un message d'accueil engageant, accompagné d'une liste de suggestions pertinentes, offrant ainsi des pistes de discussion aux utilisateurs.



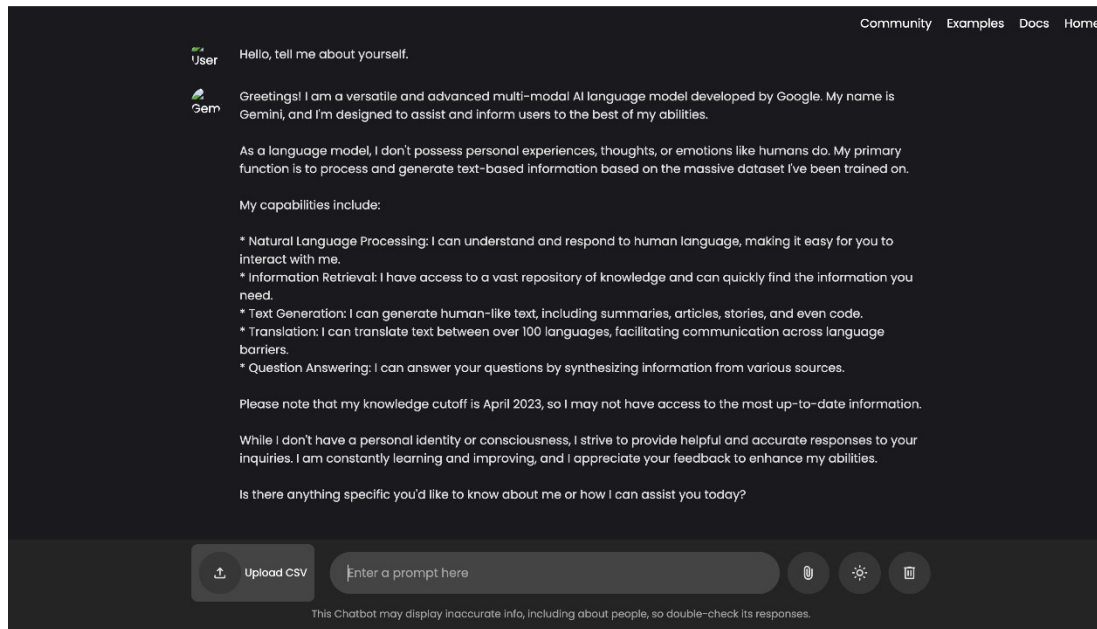
Le conteneur de chat situé au centre de l'écran affiche les échanges entre l'utilisateur et le chatbot, favorisant une communication fluide. En bas de la page, une zone de saisie permet à l'utilisateur d'entrer des messages ou d'importer des fichiers CSV, avec des icônes intuitives pour le téléchargement et l'envoi de messages.

Des boutons d'action supplémentaires pour basculer entre les thèmes et supprimer l'historique de discussion sont également inclus, garantissant une expérience utilisateur enrichie et personnalisable.

De plus, nous avons lié cette interface à l'API Gemini, ce qui permet d'améliorer l'interaction de l'utilisateur en fournissant des recommandations personnalisées et des analyses basées sur les données traitées. Cela enrichit l'expérience globale et répond aux

besoins spécifiques des utilisateurs en matière d'accessibilité. L'ensemble de l'interface est conçu pour être accessible et réactif, répondant aux besoins des utilisateurs tout en restant esthétiquement plaisant.

Voici l'interface utilisateur de notre premier prototype :



E.1 (3) - Essai(s) pour le(s) prototype(s)

Test #1 – Temps de réponse - Composante IU, API Flask et “CSVAnalyzer”

	Tests – Grandeur du fichier CSV téléchargé varie - Temps (s)		
	< 500 rangées	< 1000 rangées	< 5000 rangées
Essai #1	0.02345	0.04314	0.09890
Essai #2	0.01830	0.04663	0.07209
Essai #3	0.02455	0.04396	0.08623
Essai #4	0.02422	0.03476	0.07761
Essai #5	0.02534	0.03752	0.05961
Essai #5	0.02390	0.06133	0.09470
Essai #6	0.02709	0.04790	0.08910
Essai #7	0.02999	0.02668	0.07315
Essai #8	0.02406	0.03023	0.07376
Essai #9	0.02483	0.04291	0.07373
Essai #10	0.02625	0.02514	0.08339
Moyenne	0.02473	0.04002	0.08021
Rapport établi (rangées : seconde)	44839 rangées : 1 seconde		

Test #2 – Temps de réponse - Composante IU, API Gemini

	Tests – Nombre de mots dans les questions posées varie - Temps (s)					
	Sans Image			Accompagné d'une image		
	< 25 mots	< 50 mots	< 100 mots	< 25 mots	< 50 mots	< 100 mots
Essai #1	6.32	8.57	8.75	N/A (Cette composante du ChatBot n'est pas complètement fonctionnelle; toutefois, elle sera fonctionnelle sous peu)		
Essai #2	5.71	7.10	8.01			
Essai #3	7.66	8.14	10.82			
Essai #4	7.95	7.16	8.83			
Essai #5	7.27	8.66	10.22			
Essai #5	5.95	8.03	10.13			
Essai #6	6.89	7.89	9.32			
Essai #7	7.45	7.45	8.96			
Essai #8	8.01	8.87	10.34			
Essai #9	7.65	8.56	10.22			
Essai #10	7.32	8.23	9.73			
Moyenne	7.06	8.07	9.55			

Malgré que ce n'est pas un test, il est important de considérer le temps et les connaissances nécessaires à installer l'environnement informatique nécessaire à utiliser notre prototype. Voici une liste de ces éléments nécessaires (les valeurs associées aux colonnes "Facilité d'installation" et "Facilité d'utilisation" reflètent un utilisateur qui n'a pas des connaissances techniques):

Éléments importants	Coût (\$)	Facilité d'installation	Facilité d'utilisation
Visual Studio Code	0	Très facile	Légèrement difficile
Python 3.X	0	Facile	N/A
Flask API	0	Légèrement difficile	Difficile
Github	0	N/A	Difficile
Git	0	Facile	Légèrement difficile

E.2 - Présentation sur le progrès du projet

E.2 (2) - Premier 5 minutes:

(a) Composantes clés

- **Besoins du client :** Le département de la justice aux États-Unis a introduit de nouvelles règles concernant l'accessibilité web et mobile. Le client, Rutgers University, doit adapter les cours de leur LMS (système de gestion de l'apprentissage) Canvas (équivalent de Brightspace aux États-Unis) pour se conformer à ces règles. Nous sommes alors chargés de créer un prototype incluant un chatbot capable d'analyser le contenu des cours et fournir des recommandations de priorités pour améliorer l'accessibilité. Les besoins du client incluent la conformité aux normes WCAG 2.1 comprenant des règles pour

l'accessibilité sur le web, la personnalisation de l'interface et la capacité du ChatBot à fournir des recommandations adaptées aux utilisateurs.

- **Études d'étalonnage :** Nous avons étudié l'outil existant Ally, qui fournit des recommandations sur l'accessibilité, et identifié la nécessité d'un outil complémentaire, plus simple à utiliser et focalisé sur les priorités.
- **Spécifications cibles :**
 1. **Accessibilité :** Le prototype doit identifier les problèmes d'accessibilité dans les fichiers de cours extraits de Canvas, en se basant sur les recommandations d'Ally.
 2. **Facilité d'utilisation :** L'interface utilisateur doit être intuitive, accessible et réactive, même sur des ordinateurs de faible performance.
 3. **Interactivité :** L'application doit intégrer un chatbot capable de répondre aux questions des utilisateurs et de leur fournir des recommandations basées sur les résultats de l'analyse.
 4. **Compatibilité avec les systèmes d'accessibilité :** Intégration de l'API Gemini pour l'analyse des contenus.
 5. **Performance :** Temps de réponse du chatBot et de l'algorithme d'analyse des fichiers.csv.
 6. **Compatibilité :** Le prototype doit être compatible avec les fichiers CSV générés par Canvas et doit pouvoir traiter des fichiers de différentes tailles (petits, moyens, grands).
 7. **Classement des priorités :** Les problèmes d'accessibilité détectés doivent être classés en trois niveaux : graves, majeurs, et mineurs.
- **Concepts développés :** Nous avons développé une solution en Flask et Python, qui prend en charge l'analyse des fichiers CSV extraits de Canvas. Le prototype interagit également avec l'API Gemini pour obtenir des recommandations d'accessibilité.

(b) Résumé et présentation de la rétroaction du client

Le client, nous a dans un premier temps, présenter leur solution actuelle pour assurer l'accessibilité des cours de l'Université Rutgers avec l'outil d'accessibilité Ally. Il nous a ensuite fait part des différents problèmes qu'engendre cette solution et des retours des utilisateurs (les professeurs) n'étants pas toujours à l'aise avec cet outil.

L'outil Ally n'est pas facile d'utilisation pour les utilisateurs n'y étant pas formé. Le format des rétroactions qu'il produit est brute et occasionne une baisse de motivation des professeurs vis à vis des tâches à accomplir. Ces tâches sont présentées sous forme d'une longue liste sans priorisation des modifications les plus importantes à effectuer.

Suite à nos différentes rencontres avec le client, il nous a validé notre idée d'interface et avons établis quelles implémentations seraient intéressantes et réalisables dans le cadre du projet.

Notamment le fonctionnement du chatBot, qui restera simple (commande textuelle seulement) pour ne pas encombrer la solution et limiter les erreurs.

Le client nous a également mis à disposition des fichiers csv et des comptes Canvas factices avec lesquels nous pouvons nous entraîner et mieux aborder le sujet pour développer une solution de qualité.

Nous avons également abordé la possibilité de déployer le chatBot sur un serveur afin que les conversations des utilisateurs soient sauvegardées dans une base de données, cependant cela reste une option du fait de la complexité de la tâche et des moyens requis.

Bref, la rétroaction des clients a été favorable, en particulier concernant la simplicité d'utilisation du prototype. Notre feuille de route est respectée.

(c) Plan à venir et analyse de notre capacité à suivre le plan jusqu'à date

Jusqu'à présent, nous avons été en mesure de suivre le plan initialement établi de manière efficace. Chaque étape prévue, de la conception initiale à la création du premier prototype fonctionnel, a été réalisée dans les délais prévus. Notre équipe a réussi à surmonter les défis technique ("merge conflicts"), ainsi que la mise en place de la communication avec l'API Gemini.

Nous avons également respecté les délais pour la validation de nos hypothèses critiques, en particulier l'exécution locale du prototype, et avons pu réaliser les tests de performance sur les fichiers CSV et les réponses API, comme initialement planifié.

Pour les prochaines étapes, nous nous concentrerons sur l'amélioration de certaines fonctionnalités clés. Tout d'abord, nous allons optimiser le fonctionnement du bouton "Upload CSV" afin de garantir que les documents puissent être correctement téléchargés et analysés sans erreur. Ensuite, nous procéderons à un changement de design pour améliorer l'apparence des avatars de l'utilisateur et de l'IA, offrant ainsi une meilleure expérience visuelle.

Nous ajouterons également une barre de feedback, qui permettra au chatbot d'affiner ses réponses en fonction des retours des utilisateurs, rendant ainsi l'interaction plus précise et utile. De plus, nous veillerons à rendre les boutons "Envoyer" et "Attacher" bien distincts, afin d'éviter toute confusion lors de l'utilisation de ces fonctionnalités. Enfin, bien que le déploiement du chatbot sur un serveur soit envisagé, il dépasse largement notre budget actuel, et nous continuerons à explorer des alternatives viables à moindre coût.

Notre planification pour les deux prochaines semaines est attachée à la soumission.

E.2 (3) - Deuxième 5 minutes:

(a) Présentation des éléments de E.1

Lors de la présentation, nous allons nous concentrer les sur deux hypothèses critiques suivantes que nous avons écrites sous-forme de questions:

Serait-il préférable d'exécuter notre code de production localement sur les ordinateurs des utilisateurs, par l'entremise d'un API Flask ? et

Pourrons-nous en équipe faire recours à HTML, CSS et JavaScript afin de concevoir notre prototype?

Nous allons aussi expliquer le fait que ces deux hypothèses critiques concernent 3 des 5 CPX que nous avons choisi pour la conception du prototype final : facilité d'utilisation, l'exportation et la lecture des fichiers CSV et le support interactif fourni à l'utilisateur par l'entremise d'un "ChatBot".

L'interface est pensée pour offrir une expérience intuitive, avec une zone de chat claire, des suggestions automatiques, et des options d'upload de fichiers CSV. Elle inclut également plusieurs sections comme Home, Community, Docs et Exemples, chacune apportant des fonctionnalités spécifiques pour améliorer l'expérience utilisateur et la gestion des documents. Le design est responsif, accessible sur tous types d'écrans, avec des thèmes clairs et sombres pour répondre aux préférences de l'utilisateur. Actuellement, le prototype permet d'envoyer des messages, de sauvegarder les conversations localement, et d'analyser des fichiers CSV (cette fonction doit être optimisée). L'une des fonctionnalités permet de copier les réponses du chatbot en un seul clic pour une utilisation plus rapide et pratique.

Concernant les prochaines étapes, une idée potentielle serait de déployer le chatbot sur un serveur, mais cela impliquerait un coût élevé, bien au-delà de notre budget. Nous allons également améliorer l'interface pour distinguer plus clairement le bouton d'envoi du bouton d'attache de fichier, et revoir le design des avatars utilisateurs et IA pour offrir une expérience plus personnalisée.

Dans la présentation, nous allons aussi inclure un court résumé de la première série d'évaluation de nos prototypes. À cause que la majorité des spécifications cibles que nous avons choisies concernent des attributs qualitatifs ces tests concerneront un aspect qualitatif du prototype: le temps de réponse du "ChatBot". Une version raccourcie des deux tableaux (les versions originales se trouvent dans la section E.1.3) sera présente dans les diapositives.

(b) Démonstration en temps réel de(s) prototype(s)

Lors de la présentation, nous allons effectuer une démonstration exhaustive de nos premiers prototypes. Nous allons d'abord commencer par expliquer le processus d'analyse

des fichiers CSV. Cela inclut l'algorithme d'analyse écrit sous la classe "CSVAnalyzer" ainsi que le fonctionnement de notre Flask API.

Nous allons par la suite pivoter vers le côté opposé de notre schéma de composantes (qui fera partie de notre présentation et qui est inclut dans le livrable D). Pour cette section, nous allons montrer comment un utilisateur pourra interagir avec le "ChatBot" et montrer toutes les fonctions que nous avons inclus dans le site internet. Cela nous permettra de donner à notre audience un bon aperçu sur le progrès de notre prototype. La prochaine étape sera d'intégrer la partie d'analyse et l'interface du "ChatBot" ensemble.

Après que nous avons démontré notre prototype, nous présenterons les informations que nous avons besoin de recueillir lors de notre prochaine rencontre avec notre client. Ces informations sont résumées par les 4 énoncés suivants:

- Notre prototype n'est pas parfait: nous aurons besoin de pistes d'amélioration.
- Le client aimera certaines composantes. Il faut savoir les plus importantes à améliorer.
- En général, les clients changent souvent d'opinion. Nous devons connaître si des nouvelles fonctionnalités doivent être conçues.
- Nous avons établi une liste exhaustive de spécifications cibles, mais le client mettra de l'emphase sur certaines et pourrait nous fournir des autres.

E.3 - Rétroaction des pairs et dynamiques d'équipes

Nous avons tous complétés l'évaluation intitulée : *RÉTROACTIONS DES PAIRS ET DYNAMIQUES D'ÉQUIPE* sur la plateforme ITP Metrics.

Conclusion

Nous croyons que nous avons réussi à élaborer sur nos hypothèses critiques, concevoir un premier prototype très fonctionnel, effectuer des tests appropriés sur notre conception et de créer une présentation qui précise tous les détails généraux de notre projet jusqu'à date. Nous sommes présentement en train d'intégrer les deux souscomposantes du prototype ensemble afin de créer une application complète pour la prochaine rencontre avec notre client.

Livrable F : Contraintes de conception et prototype 2

Introduction :

Jusqu'à présent, nous avons établis nos CPX, notre conception détaillée initiale et nous avons conçu notre prototype #1. Dans ce livrable, nous allons nous concentrer sur deux contraintes non fonctionnelles qui joueront un rôle important dans la conception de notre prototype 2. Pour ce faire, nous allons modifier notre conception détaillée et réévaluer notre prototype après qu'il a été mis à date. Cela va nous permettre de déterminer si nous sommes présentement sur la bonne piste. Par la suite, nous allons avoir l'opportunité de présenter notre nouvelle conception à notre client.

F.1 : Contraintes de conception :

1.1. Identifiez les deux contraintes de conception non fonctionnelles (facteurs CPX) plus importantes qui jouent un rôle important dans le développement de vos prototypes. Justifiez votre raisonnement.

Les deux contraintes de conception non fonctionnelles (CPX) les plus importantes sont :

- 1) **La performance** : Il est important d'avoir des temps de réponses, des délais d'exécution et de chargements suffisamment courts. Cela permet une utilisation fluide et une meilleure expérience pour l'utilisateur. Si le logiciel prend du temps à analyser, traiter et fournir des réponses, cela impacterait la productivité de l'utilisateur ce qui est incompatible avec les besoins du client.
- 2) **La fiabilité** : Le logiciel doit être consistant et ne pas présenter d'erreur spontanée ou de bug, il faut également qu'il soit capable d'un pourcentage d'exactitude très élevé. Cela permet une utilisation simple, efficace et assurera la satisfaction de notre client.

Ces deux contraintes sont extrêmement importantes car elles imposent un standard qui fait de notre logiciel un outil robuste et performant.

1.2. Pour chaque contrainte de conception, expliquez en détail les changements qui devraient être apportés à votre concept pour satisfaire la contrainte (le cas échéant) ou ce qui existe déjà dans la conception pour respecter la contrainte.

- 1) **Performance** : Afin d'améliorer la performance de notre prototype (et en même temps améliorer sa facilité d'utilisation), nous avons décidé de remplacer notre API Flask et le script d'analyse CSV écrit en Python avec un script homologue rédigé en JavaScript. Cela va nous permettre d'exécuter le script d'analyse directement sur l'onglet navigateur. Conséquemment, le ChatBot devrait recevoir le résultat de l'analyse plus rapidement. Aussi, un utilisateur potentiel n'aura plus besoin d'accéder à une interface de programmation afin d'accéder au ChatBot.
- 2) **Fiabilité** : Afin d'améliorer la fiabilité de notre prototype, nous avons décidé d'empêcher le téléchargement d'image. Cela nous permettra de nous concentrer sur les fonctionnalités les plus importantes notamment, la communication à l'écrit avec le ChatBot et le fait de fournir le ChatBot directement le résultat de l'analyse. Cette décision résultera en une réduction des comportements non-attendues de notre interface d'utilisateur.

1.3. Fournissez des preuves (p.ex. analyse, calculs et/ou simulations simples, recherche) pour démontrer l'habileté de votre conception à satisfaire les contraintes. Justifiez le processus et les méthodes que vous avez utilisées.

Afin de satisfaire les deux contraintes qui concernent la performance et la fiabilité, nous avons effectués des simulations avec notre prototype. Les processus et méthodes que nous avons choisies reflètent directement le type de contrainte sous une évaluation. Les tests quantitatifs comme la performance concernent le temps de réponse mesuré avec un chronomètre. Toutefois, pour les évaluations qualitatives comme la facilité d'utilisation, nous baserons nos tests sur un échantillonnage d'opinions subjectives d'utilisateurs.

Série de tests #1 - Performance

Pour cette contrainte, nous avons évalué le temps de réponse de notre prototype. Nous avons fait des tests initiaux pour notre première architecture logicielle et les résultats étaient très impressionnants. En moyenne, un utilisateur devra uniquement attendre moins que 10 secondes à chaque fois qu'il utilise le prototype. Ce temps est associé au délai de réponse du Gemini API ainsi que le temps d'exécution du script d'analyse CSV. Nous avons récemment modifié

l'architecture logicielle de notre prototype, alors nous avons dû refaire ces tests.
Voici les résultats :

Test #1 – Temps de réponse - Composante IU et CSVAnalyzer JS (Mis à jour)

	Tests – Grandeur du fichier CSV téléchargé varie - Temps (s)		
	< 500 rangées	< 1000 rangées	< 5000 rangées
Essai #1	0.01054	0.02356	0.05841
Essai #2	0.00984	0.03456	0.04789
Essai #3	0.01554	0.02864	0.05990
Essai #4	0.01897	0.03234	0.05452
Essai #5	0.09944	0.02615	0.04872
Essai #5	0.01145	0.19840	0.05705
Essai #6	0.01678	0.02643	0.06541
Essai #7	0.01953	0.02967	0.04535
Essai #8	0.02004	0.02478	0.06501
Essai #9	0.01908	0.02105	0.04051
Essai #10	0.01652	0.02874	0.05245
Moyenne	0.02343	0.04312	0.05411
Rapport établi (rangées : seconde)	<u>49726 rangées : 1 seco de</u>		

Test #2 – Temps de réponse - Composante IU, API Gemini (Cette partie n'a pas changé)

	Tests – Nombre de mots dans les questions posées varie - Temps (s)					
	Sans Image			Accompagné d’une image		
	< 25 mots			< 50 mots	< 100 mots	< 100 mots
Essai #1	6.32	8.57	8.75	N / A (Fonctionnalité a été retirée de notre prototype).		
Essai #2	5.71	7.10	8.01			
Essai #3	7.66	8.14	10.82			
Essai #4	7.95	7.16	8.83			
Essai #5	7.27	8.66	10.22			
Essai #5	5.95	8.03	10.13			
Essai #6	6.89	7.89	9.32			
Essai #7	7.45	7.45	8.96			
Essai #8	8.01	8.87	10.34			
Essai #9	7.65	8.56	10.22			
Essai #10	7.32	8.23	9.73			
Moyenne	7.06	8.07	9.55			

Série de tests #2 - Fiabilité

Afin de déterminer si notre prototype satisfait cette contrainte, nous avons élaboré la procédure de test suivante :

- (1) Télécharger sur notre site internet un fichier CSV à analyser, pour lequel nous avons identifié les composantes qui devaient être priorisés
- (2) Demander au “ChatBot” d’énumérer tous les problèmes les plus importants concernant l’accessibilité des composantes du site internet
- (3) Déterminer si les problèmes que nous avons énumérés correspondent à ceux que le “ChatBot” nous a fourni. Cela va nous permettre d’établir un pourcentage d’exactitude du “ChatBot” qui devrait être à plus de 90%.
- (4) Le pourcentage d’exactitude sera calculé avec cette formule:
 - a. Pourcentage d’exactitude = (# de problèmes identifiés par le script d’analyse) / (# de problèmes identifiés par le “ChatBot”) x 100%

Essais et moyenne	Pourcentage d’exactitude (%)
Essai #1	50%
Essai #2	40%
Essai #3	50%
Essai #4	33%
Essai #5	33%

Essai #5	20%
Essai #6	50%
Essai #7	33%
Essai #8	20%
Essai #9	50%
Essai #10	20%
Moyenne	34.9%

1.4. Mettez à jour votre concept détaillé en conséquence.

Voir la section 2.4 pour le nouveau concept détaillé et pour une explication de sa raison d'être.

F.2 : Prototype 2 :

2.1. Résumez toute nouvelle rétroaction des clients que vous avez reçus ou tous nouveaux résultats d'essais et énoncez clairement ce qui doit être changé ou amélioré par rapport à votre concept. Mettez à jour votre concept détaillé en conséquence.

Lors de la dernière rencontre avec notre client, nous avons présenté les deux prototypes que nous avons conçu jusqu'à date. Nous avons présenté ce dernier avec les résultats de nos tests et nous avons déterminé avec le client qu'il fallait apporter quelques changements à notre prototype #2.

D'abord, le modèle Gemini 1.5 semble donner des réponses très abstraites et vagues lorsqu'il est posé des questions. Alors, avec le client nous avons décidé de pivoter vers un autre modèle d'IA, soit GPT-4.

Aussi, le client a mis beaucoup d'emphasis sur la priorisation des tâches par le ChatBot. C'est pour cette raison que nous avons besoin de modifier légèrement le format du résultat de l'analyse CSV. Au lieu de simplement fournir au ChatBot une liste exhaustive de toutes les composantes et les problèmes d'accessibilité liés à celles-ci, nous allons simplement donner au ChatBot de l'information sur les 5 composantes les plus problématiques.

La conception détaillée la plus récente se retrouve ci-dessous. Les schémas représentés dans la section 2.4 sont les conceptions détaillées qui ont été présentés au client avant de recevoir la rétroaction. Donc, notre prochain prototype sera basé sur la conception suivante:

Diagramme conceptuel de l'implémentation technique des composants du "LMS Tool"

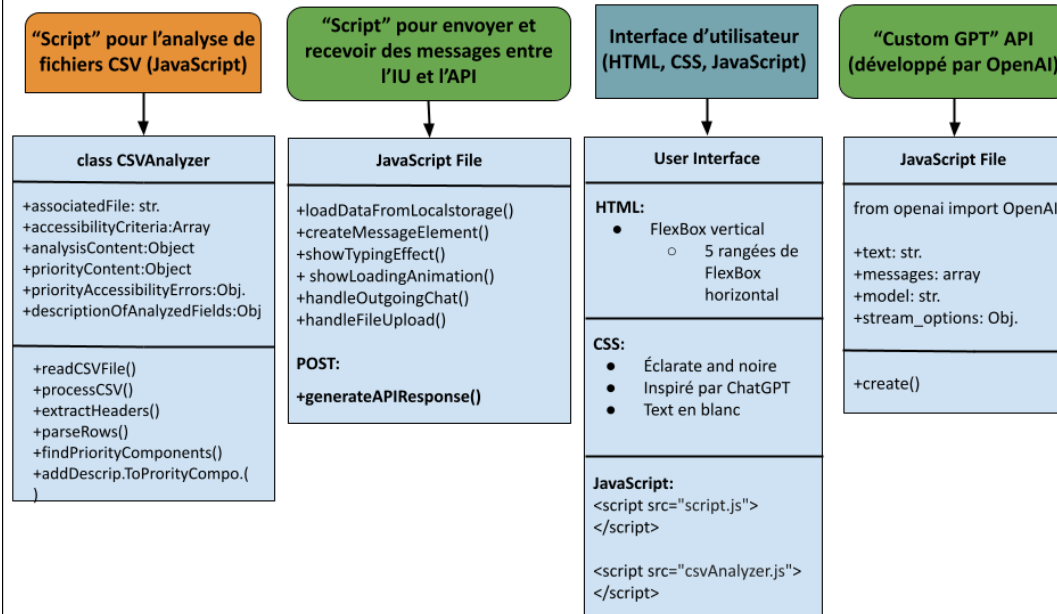
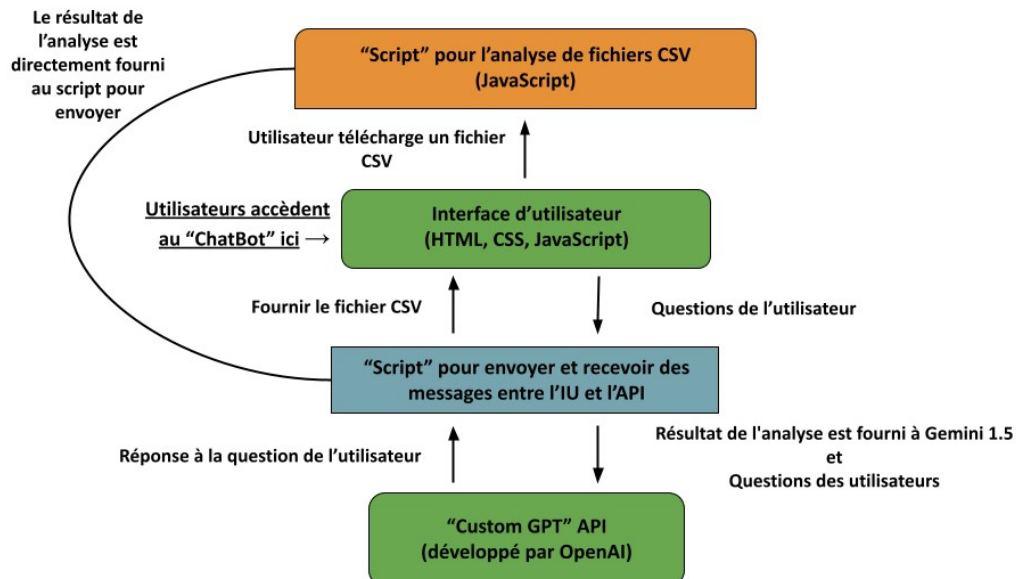


Diagramme conceptuel des composants du "LMS Tool"



2.2. Définissez vos hypothèses de produit les plus critiques que vous n'avez pas encore confirmées avec des essais. Expliquez quels facteurs CPX du Livrable du projet B cet(s) hypothèse(s) se rapportent.

Nous n'avons jusqu'à date pas confirmé certaines hypothèses critiques qui sont très importantes pour le succès de notre projet, avec des tests.

Tout d'abord, il faut vérifier si notre prototype a la capacité de prioriser certaines tâches à accomplir afin d'améliorer l'accessibilité du cours sur Canvas. Cela répond directement à notre CPX #3: *la priorisation des tâches par le logiciel*. En évaluant cette CPX à l'aide de tests, nous pourrions confirmer que notre prototype permettra d'évaluer correctement l'accessibilité des composantes des cours. Les tests que nous avons effectués concernant cette hypothèse critique sur notre prototype #2 se trouvent dans la section 2.5.

Une autre hypothèse critique qui doit être respectée afin d'assurer le succès de notre projet est sa simplicité d'utilisation. Comme pour l'hypothèse critique précédente, celle-ci répond directement à une CPX, soit *le logiciel est simple d'utilisation*. Toutefois, cette CPX concerne un aspect qualitatif de notre prototype alors il faudra changer notre méthodologie d'évaluation. Nos tests pour cette hypothèse critique se trouvent dans la section 2.5.

2.3. Développez un deuxième ensemble de prototypes qui vous aideront dans votre cheminement vers la création de votre prototype final et faites l'essai des hypothèses de produit critique au fur et à mesure.

Pour notre second prototype, nous avons décidé de modifier l'architecture logicielle de notre prototype 1. Nos changements consistent principalement d'un remplacement de l'API Flask avec un script local écrit en JavaScript. Nous avons mis à jour notre conception nos concepts détaillé en conséquence et avons effectué des nouvelles séries de tests aussi. Les concepts détaillés se retrouvent dans la section 2.4 et l'évaluation de notre prototype est démontrée dans la section 2.5.

2.4. Documentez votre dernier prototype(s) en utilisant autant d'esquisses/diagrammes/ photos que nécessaire et expliquez le but et le fonctionnement du(es) prototype(s).

Les deux images représentées ci-dessous présentent le concept détaillé de notre prototype #2. Comme mentionné précédemment, nous avons modifié l'architecture logicielle de notre prototype initial afin d'améliorer sa performance et fiabilité.

La performance a été directement améliorée en réécrivant le script d'analyse CSV en JavaScript afin de permettre la communication instantanée de notre interface d'utilisateur et ce dernier.

Nous avons aussi décidé d'empêcher les utilisateurs de télécharger des images par l'entremise de notre interface d'utilisateur afin d'améliorer la fiabilité de notre prototype.

Voici les nouveaux schémas:

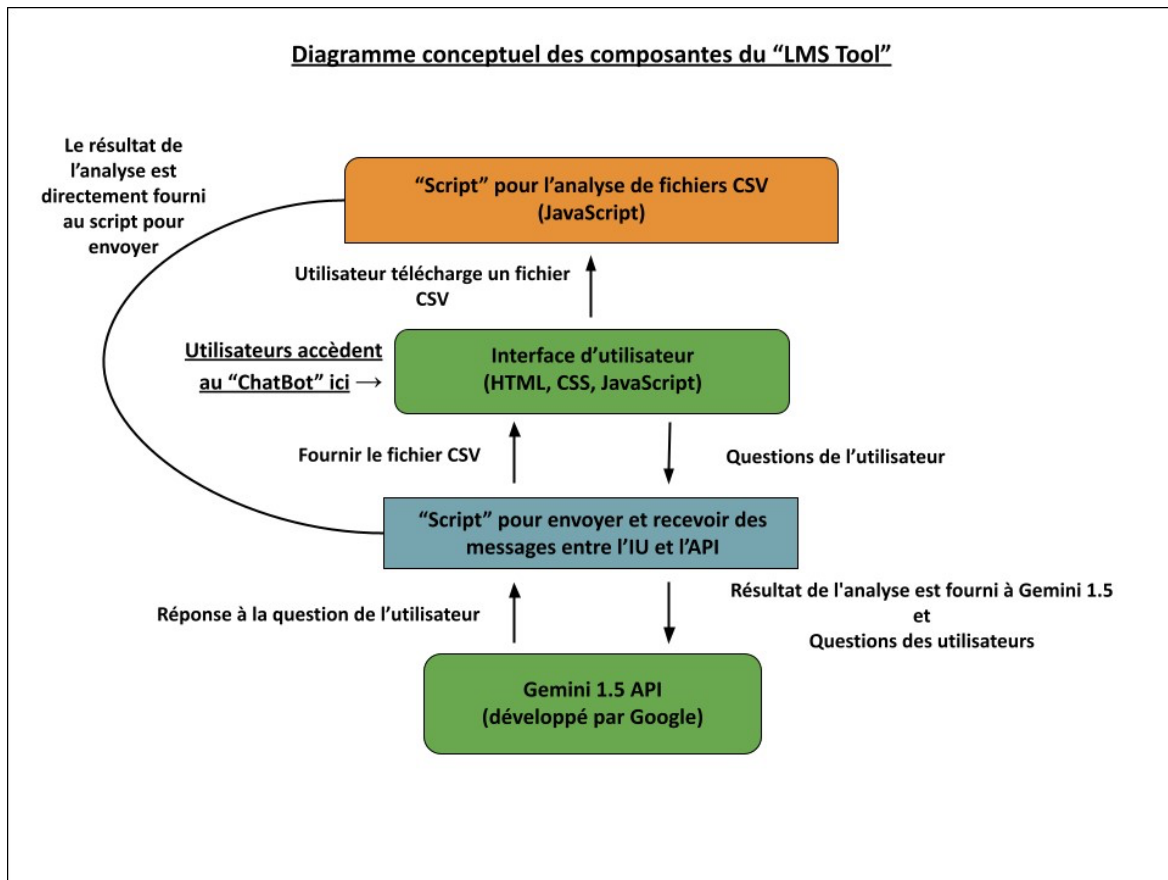
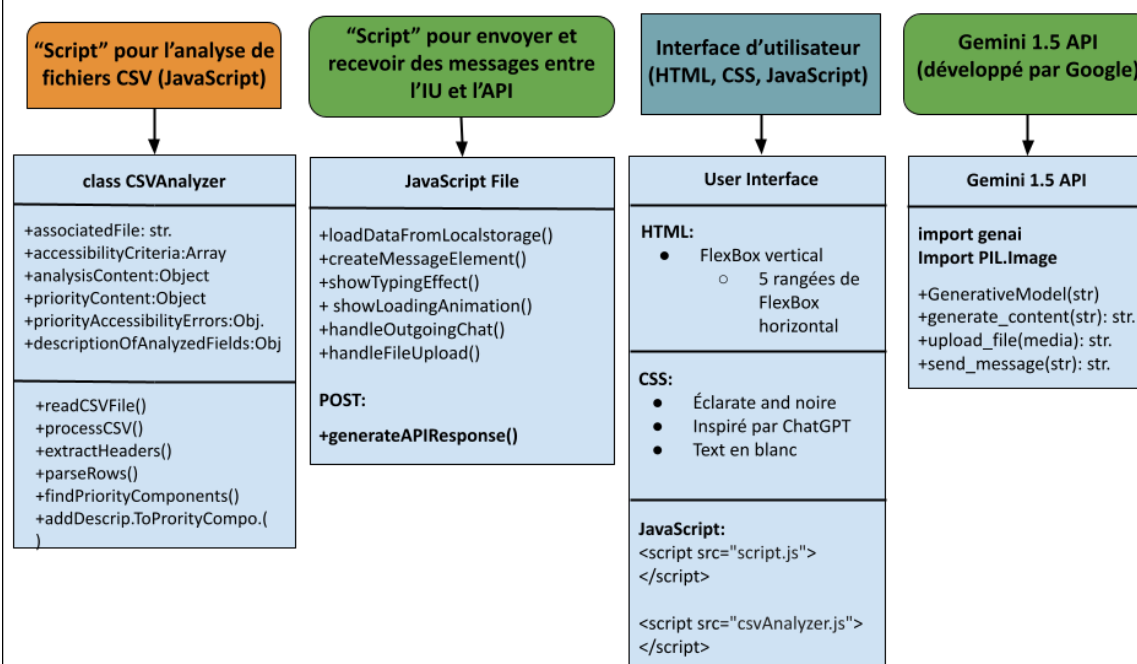


Diagramme conceptuel de l'implémentation technique des composantes du "LMS Tool"



2.5. Faites l'essai de votre prototype, analysez et évaluez sa performance par rapport aux spécifications cibles mises à jour développées en premier pour le Livrable de projet C et documentez tous les résultats de vos essais, ainsi que les spécifications de votre prototype. Présentez vos essais en forme de tableau bien organisé qui démontre les résultats attendus comparés aux résultats réels (c.-à-d. faire la comparaison des spécifications mesurées de votre prototype à vos spécifications cibles en incluant les deux dans un tableau similaire à celui que vous avez développé pour le Livrable de projet C).

Tableau 1 : Facilité d'utilisation

Utilisateur	Facilité d'utilisation	Taux d'achèvement des tâches (minutes)
Personne 1	Facile	<1
Personne 2	Facile	<1
Personne 3	Moyen	<2

Personne 4	Facile	<1
Personne 5	Moyen	<3
Personne 6	Facile	<1
Personne 7	Facile	<1
Personne 8	Facile	<1
Personne 9	Facile	<1
Personne 10	Facile	<1

Tableau 2 : Priorisation des tâches par le logiciel

Prompt: Give me recommendations on how to improve the accessibility components of “...” while adhering to WCAG 2.1.

Composantes	# de problème analysées par l'analyseur CSV	# de problèmes adressé par le ChatBot	Taux finals de problème adressé efficacement
Composante 1	2	4	2/4
Composante 2	1	4	1/4
Composante 3	1	4	1/4

On peut déduire que les recommandations fournies par le ChatBot sont trop vagues et ne sont pas adaptées pour chacune des composantes. Il faudra alors considérer une nouvelle solution pour la communication avec l'utilisateur.

2.6. Exposez les grandes lignes sur ce que votre équipe a l'intention de présenter à vos clients, ainsi que l'information que vous voulez recueillir lors de votre prochaine rencontre client.

Pour la prochaine rencontre avec notre client, notre équipe a préparé les grandes lignes suivantes de la présentation et les points d'information que nous souhaitons clarifier. L'objectif est de présenter l'avancement du projet, obtenir des retours constructifs et valider certaines décisions pour la suite.

1. Présentation des Progrès du Projet

- **Objectif principal :** Jusqu'à présent, notre équipe a mis en place un premier prototype fonctionnel de l'application. Nous avons intégré une interface utilisateur de base qui permet le téléchargement de fichiers CSV, et développé le backend avec csvAnaylzer.js pour traiter ces fichiers.

- **Démo du prototype actuel** : Nous allons montrer comment le système permet de télécharger un fichier CSV, de l'analyser, et d'afficher les résultats.
- **Explication technique** : Présenter brièvement l'architecture backend avec csvAnalyzer et la manière dont nous gérons les fichiers CSV, y compris l'analyse de base des données.
- **Interface utilisateur** : Présenter la structure actuelle de l'interface, incluant les fonctionnalités de téléchargement de fichier et de visualisation des résultats.

2. Fonctionnalités à Développer

- **Feedback utilisateur** : Présenter la possibilité d'un système de feedback intégré pour aider à améliorer l'interaction avec le ChatBot et l'analyse des données.
- **Amélioration de la fonctionnalité actuelle du prototype d'analyse**: rendre le ChatBot capable de répondre de manière spécifique et ciblée lorsqu'il reçoit des données ou des informations particulières. L'objectif est de lui permettre de détecter des éléments clés dans les messages utilisateurs pour répondre de façon précise, en s'éloignant des réponses génériques.

3. Points à Valider avec le Client

- **Scénarios d'utilisation** : Obtenir des informations sur des cas d'usage concrets ou des scénarios typiques pour mieux adapter le développement aux besoins réels des utilisateurs.
- **Préférences de design** : Avoir des retours sur l'interface utilisateur, notamment en termes de simplicité, de navigation, et d'esthétique.
- **Souhaits pour la documentation finale** : Connaître les attentes du client en termes de documentation technique ou de guides utilisateurs qui accompagneront le produit final.
- **Pistes d'améliorations**: demander au client des retours spécifiques sur la convivialité et l'efficacité de l'analyse des fichiers CSV.

Conclusion :

Pour conclure, nous avons réussi à établir les deux contraintes de conception (performance et fiabilité) qui doivent être respectés pour assurer la réussite de notre

prototype. Cela nous a permis d'établir une série de tests pour évaluer la capacité de notre produit à répondre aux deux contraintes. Suite à ces tests, nous avons établis une nouvelle conception détaillée que nous avons suivie afin de développer notre prototype 2. Maintenant que nous avons conçu le prototype 2, nous avons effectué une autre série d'évaluations afin de vérifier sa performance.

Lors de notre rencontre #3 avec notre client, nous avons présenté la conception détaillée de notre prototype 2, la conception elle-même ainsi que les résultats des tests. Cela nous a permis d'établir une troisième conception détaillée qui se retrouve dans la section 2.1.