

Jules HIRTZ
Yann MIJATOVIC
Théo PINCHON
Alexandre PERROT

Mimir
Tuteur : Yann BONIFACE
Année 2023 - 2024

MIMIR

Récapitulatif 4^{ème} itération

Sommaire

Planning prévisionnel.....	3
• Itération 4 (28h du 29/01 au 01/02) attendu.....	3
Gestion des Decks.....	3
Réponse à un deck (killer feature).....	3
Vérificateur de réponse.....	4
Modification de decks existant.....	5
Importation d'un deck.....	5
Remaniement de la structure des decks.....	6
Génération des cartes via un PDF.....	6
Tests de l'application.....	8
Routing global, navbar et style.....	8
Consultation du profil d'un autre utilisateur.....	8
Récapitulatif.....	9

Planning prévisionnel

- **Itération 4 (28h du 29/01 au 01/02) attendu**

Tableau de bord d'analyse statistique. (Jules / Théo)

Finaliser les fonctionnalités importantes (Toute l'équipe)

Tester l'application de fond en comble (unit tests, end test) et régler les bugs (Alexandre / Yann)

Ajouter les fonctionnalités optionnelles si possible (Toute l'équipe)

- Question par image
- live-code

Préparer soutenance fin de semestre (Toute l'équipe)

- **Itération 4 réel**

Routing et redirection (Jules)

Bar de navigation (Jules)

Profil d'un autre utilisateur (Jules)

Intégration génération de cartes (Alexandre)

Code refactor (Alexandre)

Test App (Alexandre & Yann)

Tags sur les decks / trie et recherche (Yann)

Page d'exploration (Yann)

Importation d'un deck (Alexandre)

Vérificateur de réponse (Yann & Alexandre)

Réponse à un deck avec mise à jour de la bdd et résumé de fin de réponse (Théo)

Modification de decks existants (Théo)

Gestion des Decks

Durant les itérations précédentes, nous avons géré la création de decks, leur structure ainsi que leur preview. Afin d'aboutir à une application minimaliste mais fonctionnelle, la principale fonctionnalité de réponse à un deck se voit toujours plus demandée.

Réponse à un deck (killer feature)

Le système de réponse à un deck est simple.

Tout d'abord, il est accessible depuis n'importe quelle page présentant des decks publiques, comme par exemple la page "Accueil", "Mes Decks" ou encore "Explorer".

L'utilisateur se retrouve alors rediriger vers une nouvelle page "q&a" pour "Question and Answer". Cette page ne fonctionne qu'avec l'utilisation d'un id dans les paramètres de recherche. Le deck souhaité sera alors chargé et la session de réponse peut alors débuter.

Avant le début de la session, un timestamp t_0 est enregistré afin de pouvoir chronométrer la performance de l'utilisateur. Lors du lancement de la session, l'application doit lister les

cartes qui sont à répondre, c'est-à-dire l'ensemble des cartes dont la date de dernière révision ajoutée à la durée de son palier respectif est antérieure à la date de l'instant t_0 . Cette liste de cartes est parcourue dans un ordre naturel, de la première à la dernière.

Lors de la réponse à une carte, actuellement 3 modes de réponse ont été pensés : saisie de texte, auto-évaluation de l'utilisateur et choix-multiple. Seuls les deux premiers sont déjà implémentés et le dernier ne saurait tarder.





Evaluez votre connaissance sur la question : 90%

0% 100%

Votre réponse ici

Par définition de sa structure, chaque carte possède 2 faces : soit la question (recto) soit la réponse verso (verso). La majorité des modes de réponse requiert une validation entre la réponse attendue et celle donnée, c'est pour cette raison que le verso de la carte n'est accessible à l'utilisateur qu'après avoir validé sa réponse. Une seule exception persiste avec l'auto-évaluation. Dans ce dernier, c'est l'utilisateur lui-même qui est responsable de la validation de sa réponse mais pour cela, il doit obligatoirement avoir accès à la réponse attendue.

Afin de promouvoir nos statistiques, la fin de la session de réponse se représente par un composant global dit résumé.

Votre progression		Que voulez-vous faire ?
 0 mauvaises réponses	 1 bonne réponse	+1 Augmenter le palier
 0m 11s		 Recommencer

Ce dernier composant ne se réduit pas qu'en son but de statistique. Bien qu'il permet à l'utilisateur de visualiser sa performance par son temps passé sur le deck (Date actuelle- t_0) ou son nombre de bonnes et mauvaises réponses, il laisse également le choix à l'utilisateur d'augmenter le palier des cartes bien répondues selon sa satisfaction ou non de ses résultats. A ce stade, l'utilisateur peut également décider de rejouer exactement la même liste.

La dernière fonctionnalité réside dans la proposition de decks similaires pouvant intéresser l'utilisateur dans ses révisions.

Vérificateur de réponse

Comme expliqué dans la partie précédente, il y a plusieurs mode de réponse possible à une carte, dont l'une qui est la saisie de texte, la problématique suivante était que certaines questions sont plus ouvertes que d'autres et ainsi on ne pouvait pas simplement indiquer que la réponse de l'utilisateur à la réponse donnée soit strictement identique.

Par exemple, “Comment fonctionne la descente de gradient”, l'utilisateur peut répondre avec certains mots différents de la réponse attendue mais le sens de la réponse est le même.

Il fallait donc trouver un moyen de comparer les réponses sémantiquement et non syntaxiquement parlant. Mais il existe une librairie nommée SentenceBERT qui permet de faire exactement ce que je viens de décrire, en comparant deux réponses entre elles et donne un pourcentage de similarité.

Cette librairie est en Python, ainsi on a pu l'implémenter rapidement en passant par l'API de l'extracteur. Et nous avons défini le seuil à 75%, cela signifie que si une réponse est similaire à 75% ou plus, alors on considère que la réponse est correct par rapport à celle attendu (cette valeur peut être modifiée au cours du projet)

Modification de decks existant

Un seul et même deck peut être perçu sous 3 formes différentes : public (lien vers la session de réponse), personnel (accès à la modification) ou statistique (affichage de selon un utilisateur donné). La première forme est décrite au-dessus et la dernière n'est pas encore implémentée. Pour ce qui en est de la seconde, elle est accessible uniquement par son créateur.

Pour comprendre comment s'effectue la modification d'un deck, il faut comprendre que cette action n'utilise aucune nouvelle page. La modification est intégralement prise en charge par la page de création de deck en y ajoutant l'id du deck dont il est question dans les paramètres de l'url.

De cette façon, chaque deck modifié est tout d'abord chargé par l'application puis utilisé afin de remplir chaque champ d'entrée de la page. Cette surcouche sur la page “newDeck” est simple et efficace. De plus, elle permet de pouvoir annuler ces changements à tout moment puisque le deck n'est pas directement mis à jour.

Afin de remplacer l'ancien deck par sa version modifiée, il est impératif de stocker à minima l'id BDD du deck dès son chargement sur la page de modification.

Importation d'un deck

Dans notre application, nous avons mis à disposition d'une bibliothèque de decks qui permet d'accéder à tous les decks publics de l'application afin que tous les utilisateurs puissent y accéder. C'est un moyen de partage rapide et efficace.

Il a fallu donc implémenter la fonction d'importation de deck, en effet, quand un utilisateur clique sur un deck du marketplace, cela va le rediriger et lui montrer le contenu du deck, exactement comme dans la création ou modification de deck, mais lors de l'importation, l'utilisateur ne pourra que visualiser le contenu, il ne pourra rien modifier tant qu'il n'a pas importé dans sa bibliothèque personnel.

Une fois le deck importé, le deck est copié et mis dans la bibliothèque de l'utilisateur, il peut modifier le deck comme bon lui semble.

Remaniement de la structure des decks

Au long de notre développement de l'application web, on a procédé au développement des pages en fonction de ce qui était le plus logique pour nous à réaliser. A un moment donné, nous nous rendons compte que la page statistiques est un problème assez complexe à résoudre. En effet, pour avoir des statistiques comme nous l'avions prévu dans l'étude préalable, il faut pouvoir enregistrer les performances de chaque utilisateur.

Notre idée de départ était que quand un utilisateur décide d'importer un deck, il clone le deck et se l'attribue personnellement.

C'est une bonne idée sur le concept de notre application qui est le suivi personnel de sa progression, mais on perd de l'information, au niveau de statistiques générales de chaque deck notamment. En effet, on voudrait garder les statistiques d'un deck pour que les utilisateurs se fassent une idée de la difficulté et d'où il se situe par rapport aux autres utilisateurs.

Le remaniement de la base de données Decks fut notre première idée, pour assigner des statistiques à chaque question identifiée par un utilisateur pour pouvoir faire des statistiques générales ainsi que récupérer les statistiques personnelles de chaque utilisateur pour pouvoir avoir le suivi.

On s'est rendu compte après-coup que ce système n'était pas le bon, puisque si on laisse la possibilité aux utilisateurs d'importer des decks publics, on leur laisserait la possibilité de les modifier comme bon leur semble et cela ne serait pas cohérent avec les statistiques de chaque deck, puisque le deck copié ne correspond pas forcément au deck original.

La nouvelle structure, elle, correspond beaucoup plus au système de l'application. Nous avons juste envisagé de rajouter une collection "statistiques" dans la base de données. Celle-ci permettrait d'assigner les statistiques à chaque deck, avec des statistiques pour chaque utilisateur. La structure des statistiques serait la même à chaque fois car un deck copié pourrait être publié avec des changements

Génération des cartes via un PDF

Lors de la première itération, nous avons fait une preuve de concept pour pouvoir convertir des PDF de cours vers des cartes, elle a été validée et nous avons mis en place l'extracteur de texte ainsi que l'intelligence artificielle qui s'occupe de générer les questions/réponse en fonction du contenu du PDF.

Maintenant, il a fallu l'intégrer dans notre application Web. Lorsque l'utilisateur souhaite créer un deck, il a fallu ajouter un bouton "Générer" pour qu'il puisse générer des cartes à partir d'un PDF (dans le futur, on implémentera d'autres formats tels que des Word et des PowerPoint)

Il a fallu implémenter la partie visuelle, où l'utilisateur doit déposer son fichier PDF, et implémenter un système de chargement pour indiquer à l'utilisateur que l'IA génère les cartes pour lui, et si jamais il y avait un problème quelconque, il a fallu mettre en place un système d'erreur pour signaler à l'utilisateur qu'il a eu un problème (mauvais fichier PDF, l'IA qui a planté...)

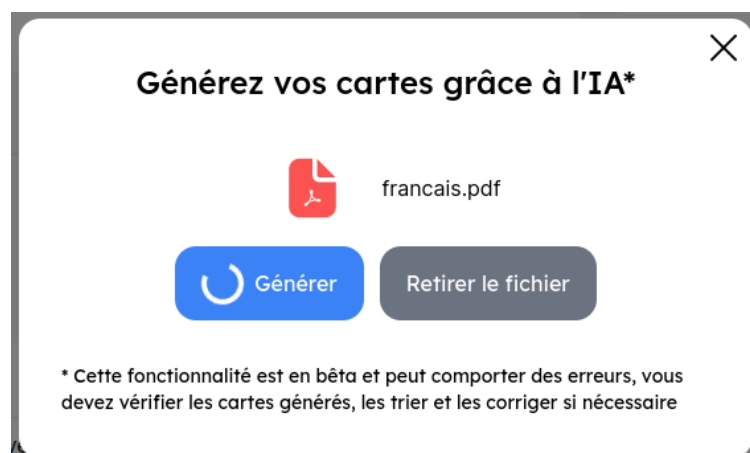
Ensuite, comme toute la partie back-end a été faite lors des précédentes itérations, il a fallu juste faire une requête depuis le serveur ayant l'extracteur de texte et l'IA, il y a eu quelques problèmes notamment à cause du CORS, le programme Python tournant sur un autre serveur que celui du site Internet, le navigateur refusait d'effectuer la requête en raison des risques de sécurité, mais nous avons pu résoudre ce problème rapidement en indiquant qu'il pouvait faire confiance à ce serveur.

Après que les questions/réponses soient générées, le serveur les renvoie au client et nous avons pu afficher directement dans de nouvelles cartes de l'utilisateur.

Cette expérience dure quelques secondes et est très fluide pour l'utilisateur. On s'attendait à ce que ça prenne plus de temps à générer. En revanche, à cause du manque de ressources que nous avons et de temps, nous n'avons pas pu prendre un modèle plus puissant pour l'IA et ainsi, cela génère des réponses peu cohérente par rapport au PDF, les questions restent plus pertinentes que la réponse qui elle, est complètement aberrante.

Et ainsi, on signale bien à l'utilisateur qu'il doit corriger par derrière ou supprimer la carte si cela ne lui va pas.

Voici quelques exemples visuels





1

Comment s'écrit le verbe "être" au présent simple à la première personne du singulier

QUESTION

je suis

RÉPONSE

+ Ajouter une nouvelle carte

Créer

2

Quelles sont les relations entre les deux axes ?

QUESTION

Les deux axes doivent être séparés, mais ils sont indissociables, les mots n'étant pas isolés et donnés sous forme de liste.

RÉPONSE

Tests de l'application

Au cours de cette itération, en plus d'implémenter de nouvelles fonctionnalités, nous avons également décidé de tester l'application afin de déceler les bugs qu'il pourrait avoir, nous avons pu optimiser le code également.

Tout d'abord, il y avait un souci visuel majeur, la barre de navigation à gauche qui était mal mise en place, en effet quand un utilisateur n'est pas connecté, il n'a pas besoin de voir cette barre de navigation, mais elle était quand même visible sur toutes les pages même quand elle n'a pas lieu d'être. On a pu régler ce problème visuel gênant.

On a également pu résoudre de nombreuses erreurs que la console de navigateur nous indiquait, où l'on utilisait mal certains composants.

Et enfin nous avons pu mettre en place certaines sécurité, comme un mauvais ID de deck lors d'une importation qui indique à l'utilisateur que le deck est introuvable au lieu d'une erreur qui faisait planter toute l'application.

Routing global, navbar et style

Après avoir repensé rapidement la navigation d'un utilisateur dans l'application en fin d'itération 3 nous, nous avons eu l'occasion de réimplémenter les différents accès à l'application en fonction de l'état d'un utilisateur. Cela a par exemple compris le routing d'un utilisateur si il est déjà connecté (utilisation de la session) pour éviter d'avoir des non sens (comme une page de connexion par exemple). De plus, le changement de ces routes nous ont permis de remarquer des anomalies dans l'expérience utilisateur, notamment la présence d'espacement pour accueillir la barre de navigation latérale (et son chargement), alors que l'utilisateur n'est pas censé pouvoir la voir. Cela à donc ensuite grandement modifié la perception et la conception que nous avions des différents layouts de l'application, entraînant une refonte de multiples parties des composants pour obtenir un résultat satisfaisant.

Consultation du profil d'un autre utilisateur

L'architecture a une nouvelle fois été repensée pour accueillir un nouveau système de contact. Pour l'instant nous disposons simplement d'une liste de contacts, mais cela manquait de clarté et de modularité. Voulant axer l'application sur le côté social pour se départager de nos concurrents, nous avons décidé de créer deux branches distinctes : followers (les utilisateurs qui nous suivent) et following (ceux que l'on suit). Cela permet donc de suivre quelqu'un et potentiellement plus tard d'avoir un système de notifications lorsqu'il crée un deck, sans qu'il ait besoin de nous avoir également en contact. Cette mise à jour permet à notre sens de gamifier le travail et de pousser à l'ouverture du contenu d'apprentissage à travers la reconnaissance de la qualité de son travail. Concrètement, nous avons donc dû inclure un moyen de voir les profils d'autres personnes et d'y incorporer les decks créés par cet utilisateur.

Récapitulatif

A l'heure où le projet prend forme, chaque problème ou modification apportées peuvent avoir un impact important sur le reste du projet. C'est pour cette raison que notre principale stratégie reste le merge avec le main et push le plus régulier possible, avancer doucement mais sûrement.

Comme prévu, nous finissons cette itération avec une application fonctionnelle et un design des plus agréables. Cependant quelques points restent à éclaircir afin de rendre le produit le plus complet possible, notamment les statistiques ou l'élaboration de couples question/réponse plus complexes. Et nous verrons tout cela dans les itérations suivantes.