

Jules HIRTZ  
Yann MIJATOVIC  
Théo PINCHON  
Alexandre PERROT

Mimir  
Tuteur : Yann BONIFACE  
Année 2023 - 2024

# **MIMIR**

## Récapitulatif 2<sup>ème</sup> itération

# Sommaire

<b>Planning prévisionnel</b>	<b>3</b>
• Itération 2 (28h du 15/01 au 18/01)	3
<b>Gestion des topics (In App)</b>	<b>3</b>
Decks	3
Cartes	4
<b>Amélioration de l'IA et lien avec la BDD</b>	<b>4</b>
<b>Maquette de l'application</b>	<b>6</b>
<b>Profil utilisateur et session</b>	<b>7</b>
<b>Récapitulatif</b>	<b>7</b>

# Planning prévisionnel

- **Itération 2 (28h du 15/01 au 18/01)**

Gestion des cartes et decks depuis l'interface Mimir. (Théo)

Système d'authentification et création de api. (Jules)

Design de la nouvelle interface et test de l'extraction de pdf en passant par xml.  
(Alexandre)

Test IA en fluctuant la taille des entrées et génération des cartes avec IA directement en BDD (Yann)

## Gestion des topics (In App)

### Decks

Bien que la principale action se limite à répondre son contenu, il en est tout autre quand celui-ci est un deck personnel. En fait, depuis la page "profil" l'utilisateur peut visualiser les decks qu'il a créés mais également les modifier.

Pour se faire, l'intégralité du deck se charge dans la page de création de deck comme un deck fraîchement construit. De cette façon, son créateur peut ainsi en modifier tous les paramètres sans pour autant impacter directement la base de données.

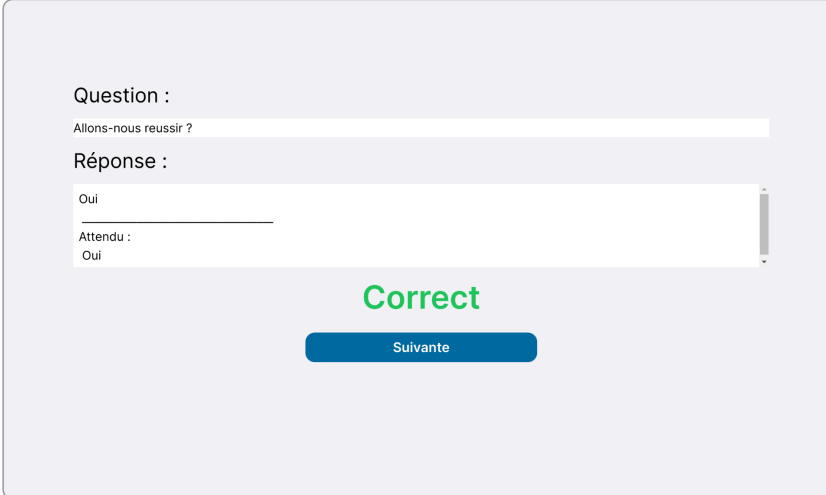
Une fois toutes les modifications terminées, l'utilisateur n'a plus qu'à valider l'ajout du deck comme lors de sa création. Le système de chargera de vérifier si le deck existe déjà avant de le mettre à jour dans la BDD.

### Cartes

Selon notre implémentation des éléments dans la base de données, aucune carte n'est accessible seule. Le seul chemin d'accès à une carte précise est l'utilisation du deck auquel elle appartient, ainsi qu'un id ou index pour extraire l'élément voulu.

Afin de prévenir des déconnexions spontanées, toute carte répondue est automatiquement mise à jour dans la base de données. Cela passe tout d'abord par la mise à jour en local dans la liste de cartes puis le deck en cours est re-construit pour finalement remplacer son prédécesseur toujours situé dans la base.

Lorsqu'un utilisateur répond à une carte, il doit être informé de la justesse de sa réponse ainsi que celle attendue. Actuellement représenté comme suit dans l'application :



The screenshot shows a quiz interface with a light purple background. It contains the following elements:

- Question :** A text input field containing "Allons-nous réussir ?".
- Réponse :** A text input field containing "Oui".
- Attendu :** A text input field containing "Oui".
- Correct :** A green text label indicating the answer is correct.
- Suivante :** A blue button with white text to proceed to the next question.

## Amélioration de l'IA et lien avec la BDD

Pour cette itération, nous avons voulu essayer tous les moyens à notre disposition avec les moyens techniques nous étant disponibles d'améliorer les résultats de sorties du modèle de l'IA.

Pour cela, nous avons décidé de porter notre attention sur deux choses :

- Le texte transmis au prompt
- Le modèle d'IA

Au niveau du modèle d'IA, nous avons essayé d'utiliser un modèle plus performant, notamment les modèles Mixtral 8X7B et Vigogne 13B ( les B représentent le nombre de milliards de paramètres que le modèle prend ). Pour avoir tenté l'expérience avec les deux modèles, nous nous sommes rendus compte que le traitement du prompt prenait beaucoup plus de temps et que le logiciel permettant de faire tourner l'IA est plus apte à bugger. Nous avons donc abandonné l'idée de changer de modèle d'IA et conserver celle qu'on utilise actuellement.

Au niveau du changement du texte transmis au prompt, nous avons essayé deux choses :

- Transformation du texte du PDF en XML
- Séparation du texte en plusieurs parties

Pour la transformation en XML, nous pensions que ce format allait définir chaque partie du texte du PDF, comme l'entête et le pied de page pour enlever les

informations inutiles. Nous nous sommes rendus compte que ce format n'apporte pas réellement ces informations dont nous avons tant besoin pour réguler le texte transmis au prompt. Nous avons certaines informations, comme la taille en centimètres de l'entête, mais cela ne nous sert absolument à rien.

Au niveau de la séparation du texte en plusieurs parties, nous avons essayé de séparer le texte en plusieurs parties ainsi que les questions/réponses voulues. C'est-à-dire que pour 10 questions/réponses voulues sur un pdf de 5 pages, nous prenons par exemple une division du texte en 5 parties et on génère 2 questions/réponses pour chaque partie. On se rend compte après de nombreux tests sur différents PDF et des nombres différents de parties que les résultats sont similaires à la génération de questions/réponses sur un gros texte non séparé.

Suite à ces différents résultats peu concluants, nous avons décidé de garder le programme de départ.

Nous avons donc aussi lié le texte de sortie de l'IA qui est un JSON avec la base de données. Pour ça, nous modifions à la main la sortie pour le transformer en format de deck attendu par la BDD et transmettre ce résultat à la base de données en passant par l'API que nous avons mis en place. Cela fonctionne en localhost à l'aide de flask, mais il faudra lier ce programme à notre application web une fois que l'environnement sera un peu mieux défini.

## Maquette de l'application

Durant la première itération, nous avons implémenté l'application web selon la maquette qu'on avait défini lors de l'étude préalable pour montrer l'idée du projet et d'avoir un prototype rapide à mettre en place pour qu'on puisse tester les fonctionnalités directement sans s'attarder sur les détails.

Dans cette itération, nous avons refait une maquette qui est beaucoup plus complète et plus détaillée avec une charte graphique qu'on a établi au sein de l'équipe.

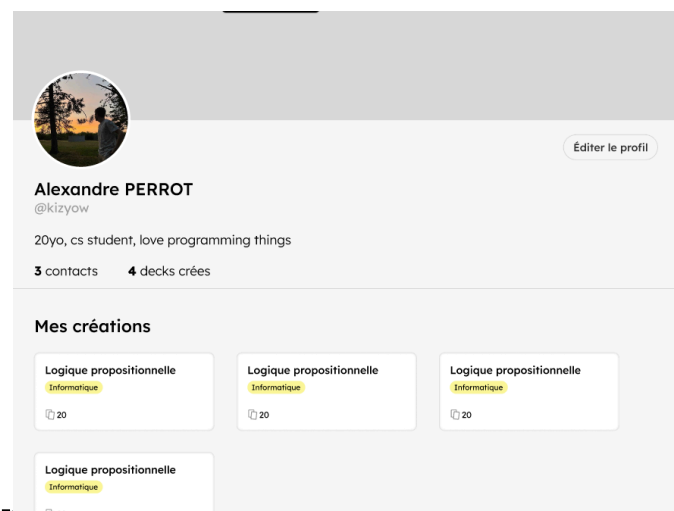
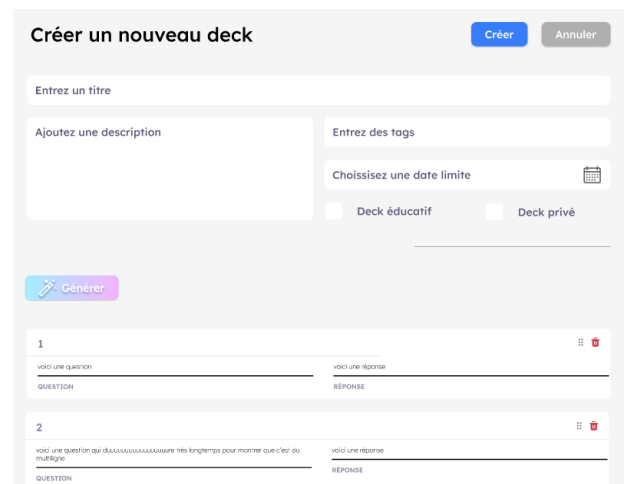
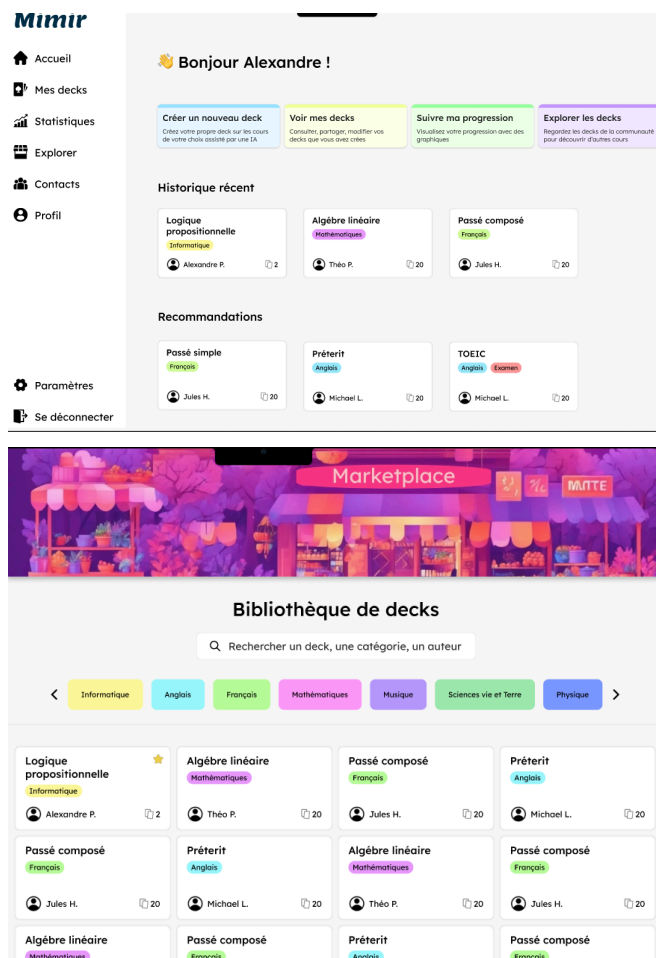
La maquette a été réalisée sur Figma et va nous permettre de la recopier au détail près sans qu'on ait besoin de penser à quelle couleur ce texte et sous quel format on le met en place. Cette maquette sert à cela, à avoir un design visuel entièrement complet

On a créé 25 pages sur la maquette ce qui signifie 25 pages à recoder dans l'application Web.

L'avantage de React est qu'on peut décomposer certains éléments comme des cartes en composant, ce qui nous permet de réutiliser autant de fois qu'on veut sur n'importe quelle page Web en le codant une seule et unique fois.

La programmation front-end est prévue lors de l'itération 3 en respectant la maquette.

Quelques images de notre maquette :



## Profil utilisateur et session

A l'aide de la librairie next-auth nous avons pu créer des profils utilisateurs complets dans la bdd, ajouter la couche d'interactivité du formulaire de login et d'enregistrement de compte. Ces nouvelles données nous ont permis de mettre en place une session utilisateur (récupérer les données du client en cours d'utilisation) et par extension de protéger notre application d'explorations sans compte utilisateur et / ou de pouvoir jouer sur certains leviers marketing. En effet les routes mises en places, permettent (ou non) d'accéder à certaines parties du site et peuvent par

exemple inciter un internaute à découvrir l'interface et à cliquer sur un deck, mais le redirige automatiquement vers la page de login s'il n'est pas connecté.

## Récapitulatif

Pour cette deuxième itération, l'apparition de conflits lors de la liaison de l'interface à la base de données nous a fait perdre un temps précieux.

Cependant, le malheur des uns a permis de libérer du temps pour les autres. Ce temps a été utilisé à bon escient puisque nous clôturons cette nouvelle itération avec un logique d'application fonctionnelle, une api vers la BDD également opérationnelle mais surtout avec une toute nouvelle maquette plus représentative de l'esprit de l'application.

La prochaine étape est donc la fusion parfaite de ces différents modules en vue de finir l'itération 3 avec une application opérationnelle. La suite nous servira à l'implémentation des nombreuses autres fonctionnalités.