Ergonomie & développement d’une SPA animée

QuizWiz

|  |  |
| --- | --- |
| **Auteur 1** | Asmae Hammouten |
| **Auteur 2** | Dania Ouarrad |
| **Auteur 3** | Eda Copur |
| **Auteur 4** | Ibrahim Bekkari |
| **Auteur 5** | Lars Hanquet |
| **Date** | 17.12.2023 |
| **Référence** | WEB2-2023-PROJECT-GROUP-10 |
| **Version** | 1.0 |

Contents

[1 Consignes et évaluations 3](#_Toc147308156)

[1.1 Consignes générales 3](#_Toc147308157)

[1.1.1 Création des groupes sur le site du cours 3](#_Toc147308158)

[1.1.2 Création d’un groupe sur GitHub Classroom et du web repo associé 4](#_Toc147308159)

[1.1.3 Projet 4](#_Toc147308160)

[1.2 JavaScript & Node.js : consignes techniques, timing et évaluations 6](#_Toc147308161)

[1.3 Ergonomie : consignes techniques, timing et évaluations 11](#_Toc147308162)

[2 Objectif du projet 11](#_Toc147308163)

[3 Mind map du projet 12](#_Toc147308164)

[4 Persona 12](#_Toc147308165)

[5 Axiomes de Morville 13](#_Toc147308166)

[6 Planning des tâches et cas d’utilisation 13](#_Toc147308167)

[7 Besoins techniques 16](#_Toc147308168)

[7.1 Système 16](#_Toc147308169)

[7.2 Frontend 17](#_Toc147308170)

[7.3 API 17](#_Toc147308171)

[8 Choix technologiques 18](#_Toc147308172)

[8.1 Frontend 18](#_Toc147308173)

[8.2 RESTful API 18](#_Toc147308174)

[8.3 Wireframe 18](#_Toc147308175)

[9 Conception & Implémentation 19](#_Toc147308176)

[9.1 Code repositories 19](#_Toc147308177)

[9.2 Secrets éventuels pour vos API ou base de données 19](#_Toc147308178)

[9.3 Documentation de votre API 20](#_Toc147308179)

[9.4 Déploiement de vos applications 20](#_Toc147308180)

[9.5 Code réutilisé 20](#_Toc147308181)

[10 Analyse des résultats par le groupe 21](#_Toc147308182)

[10.1 Évaluation du résultat par rapport au planning des tâches et des cas d’utilisation 21](#_Toc147308183)

[10.2 Audit ergonomique de votre projet 21](#_Toc147308184)

[10.3 Difficultés techniques rencontrées 21](#_Toc147308185)

[10.4 Conseils pour appliquer cette technologie 22](#_Toc147308186)

[10.5 Quels sont les points positifs à la manière dont s’est déroulée la collaboration au sein du groupe ? 22](#_Toc147308187)

[10.6 Quels sont les points qui seraient à améliorer pour de futures collaborations ? 22](#_Toc147308188)

[11 Analyses individuelles des résultats 22](#_Toc147308189)

[12 Présentation vidéo 22](#_Toc147308190)

[13 Revues de projets par les pairs 24](#_Toc147308191)

# Consignes et évaluations

## Consignes générales

### Création des groupes sur le site du cours

Veuillez former un groupe de 5 étudiants sur le site associé au cours : <https://e-vinci.github.io/web2>. Pour ce faire, veuillez-vous authentifier en cliquant sur l’icône . Rendez-vous sur l’onglet **Projets** (<https://e-vinci.github.io/web2/project-page>). Il est recommandé que l’attribution des **groupes** se fasse par **discussions** entre les **étudiants**. Lorsque 5 étudiants ont **un intérêt commun** pour un **projet**, ils s’inscrivent au sein d’un groupe en cliquant sur l’icône .

Pour aider à la création de groupes, il est aussi possible de vous inscrire :

* **à un groupe vide**. Cela permettra à tous d’identifier les partenaires potentiels.
* **à un groupe où il y a déjà un ou plusieurs étudiants**. Dans ce cas, veuillez-vous entretenir avec ces potentiels partenaires sur le **sujet de votre projet**.

Si nécessaire, vous pouvez vous désinscrire d’un groupe où vous n’avez pas trouvé de sujet commun dans le but de rejoindre un autre groupe. Il suffit de cliquer sur l’icône .

A la date ultime de création de groupe (Séance 9 du cours de JS, 16/10 ou 17/10 selon la série), pour les étudiants toujours en recherche de partenaires, nous faciliterons (ou imposerons si nécessaire) la création des groupes, mais pas des sujets de projet.

Une fois tous les groupes de 5 étudiants remplis, il restera maximum 4 étudiants non liés à un projet. Si nécessaire un ou plusieurs groupes de 4 étudiants seront créés.

### Création d’un groupe sur GitHub Classroom et du web repo associé

Pour chaque groupe de projet, vous allez héritez d’un web repository contenant un boilerplate via GitHub classroom.

**Veuillez passer à cette étape qu’une fois votre groupe déjà finalisé sur le site du cours.**

#### Création de l’équipe associée à un projet

Veuillez identifier le membre qui créera votre équipe sur GitHub.

Ce membre accédera à l’assignement via : <https://classroom.github.com/a/zJz7A4kY>

Ce membre devra créer une équipe reprenant le numéro de projet donné sur <https://e-vinci.github.io/web2/project-page> : si le nom de projet indiqué est **Projet N°4 : …** , il créera une équipe portant le nom **group-04** puis cliquera sur **Create team**.

Ce membre devra encore cliquer par la suite sur **Accept this assignment**.

Un web repository aura été créé pour votre équipe.

#### Joindre une équipe existante

Une fois l’équipe d’un projet créée, les autres membres accéderont aussi à l’assignement via : <https://classroom.github.com/a/zJz7A4kY>.

Ces membres joindront l’équipe existante en cliquant sur **Join** au sein de la bonne équipe. Par exemple, pour les membres du **Projet N°4**, ils cliqueront sur **Join** dans l’équipe **group-04**.

Si vous le souhaitez, vous pouvez visualiser cette vidéo qui montre [comment Joindre un GitHub Classroom Group Assignment](https://youtu.be/3TIVVCkne_0).

### Projet

Vous allez créer une SPA mettant en œuvre :

* Des sujets et technologies qui vous tiennent à cœur ;
* Une RESTful API tournant sous Node.js & Express ;
* Un frontend animé ;
* Un frontend consommant votre RESTful API et éventuellement des APIs tierces ;
* Au moins une librairie JS non vue en cours pour le frontend (anime.js ou phaser.io sont autorisées) ainsi qu’une librairie non vue pour l’API.

Pour votre frontend animé, l’animation peut être 2D, 3D, sous forme de jeux ou de simples effets visuels…

Au niveau de la présentation de votre projet, veillez à :

* Prendre en compte l’expérience utilisateur dès le début
* Optimiser le choix de vos technologies en fonction de l’expérience utilisateur
* Appliquez un maximum de théorème psycho-marketing
* Respectez les règles de Usability et auditez votre projet
* Respectez le GDPR

## 

## JavaScript & Node.js : consignes techniques, timing et évaluations

| **Tâche** | **Compétences** | **Critères** | **Deadline** | **Pt** | **Consignes** |
| --- | --- | --- | --- | --- | --- |
| Objectif du projet | C6) Documenter et présenter en vidéo le développement d'une SPA |  | **Séance 9 : 16/10 ou 17/10** |  | Donnez un nom à votre projet et décrire l’objectif de votre projet au §2 de ce document ainsi que sur <https://e-vinci.github.io/web2/project-page>, complétez :   * Le nom du projet : Projet N°X : Nom de votre projet * le champs « Description ».   Discuter de votre objectif avec un enseignant et assurer vous que cet objectif soit validé avant d’aller plus loin dans votre projet. |
| Planning des tâches et cas d’utilisation | C6) |  | **Séance 12 : 24/10 ou 27/10** |  | Décrire le planning des tâches et cas d’utilisation selon les instructions données au §6.  Présenter votre planning à un enseignant, afin qu’il puisse vous aider à bien prioriser les tâches. |
| Indiquer l’URL de votre code repository | C6) |  | **Séance 12 : 24/10 ou 27/10** |  | Votre code doit être accessible par tout le monde via un web repository public qui vous sera assigné par GitHub Classroom. Cela permettra notamment aux enseignants de suivre vos avancées tout au long de votre projet. Veuillez indiquer votre URL sur <https://e-vinci.github.io/web2/project-page>.  Plus d’information aux §1.1.2 et §9.1. |
| Choix technologiques | C6) |  | **Séance 15 : 13/11 ou 14/11** |  | Compléter le §8.  Discuter de vos choix technologiques avec un enseignant. |
| Rapports individuels d’activités | C6) | Rapports de qualité *Indicateurs : formulation de qualité, analyse de qualité, respect des consignes* | **12/11**  **19/11**  **26/11**  **3/12**  **10/12**  **17/12** | **1 solo** | Des sessions individuelles de feedback sont organisées via TEAMMATES permettant à chacun de répondre à des questions dont les réponses sont confidentielles ou anonymisées au sein d’un groupe. Des e-mails seront envoyés vous invitant à compléter un formulaire hebdomadaire, à compléter pendant le WE.  À partir de la 2ème soumission, tout formulaire hebdomadaire non complété amènera à une pénalité individuelle de 0.5 point.  Si vous manquez deux soumissions, vous aurez l’obligation de montrer que vous êtes actif sur le projet sous risque d’être écarté du projet. |
| Soumission du rapport de groupe | C6) | Idem | **17/12** | **1** | Compléter le §10 ainsi que tous les paragraphes qui n’auraient pas été finalisés de ce document.  Soumettre ce document, via Moodle (un devoir sera créé) ainsi que dans le répertoire **/report** de votre repo.  Effacer toutes les consignes mises *en grisé* dans ce document avant de soumettre ce rapport sur Moodle. |
| Soumission de la vidéo | C6) | Vidéo de qualité  *Indicateurs : présentation du projet de qualité, analyse de qualité, respect des consignes* | **17/12** | **2** | Présenter votre projet selon les exigences du §12. |
| Soumission du code du frontend | C2) Création d'IHM pour SPA  inclus :  C5 : Intégrer au développement d'une SPA des technologie non vues en cours  inclus si nécessaire :  C3) Sécurisation de SPA | Qualité de l’IHM produite  *Indicateurs : esthétique, fonctionnel, codage de qualité, respect des consignes, ambitieux & original, utilisation d’une librairie pour l’IHM non vue en cours* | **17/12** | **8** | Réaliser un frontend et un backend de Qualité : Code bien structuré, UI et UX de qualité, API bien documentée (documentation des opérations de votre API, requêtes permettant de tester votre API…).  Être ambitieux et original.  Démontrer une appropriation personnelle du code (via commentaires dans le code, discussion lors des cours...).  Respecter les spécifications techniques décrites dans ce document.  Déployer votre frontend et votre backend chez un provider gratuit.  NB : votre RESTful API doit être un minimum différente des APIs fournies dans les démos du cours de JS. |
| Soumission du code du backend | C1 : Création de services web  inclus :  C5 : Intégrer au développement d'une SPA des technologie non vues en cours  inclus si nécessaire :  C3 : Sécurisation de SPA | Qualité du web service produit  *Indicateurs : fonctionnel, codage de qualité, respect des consignes, ambitieux & original, utilisation d’une librairie pour le service web non vue en cours* | **17/12** | **5** |
| Déploiement tant de votre frontend que backend | C4) Déploiement d'applications web | Déploiement de la SPA sur le cloud  *Indicateurs : fonctionnel, performances de chargement acceptables* | **17/12** | **2** |  |
| Réaliser un minimum de 5 revues sur le site web | C7) Analyser le développement de SPA faites par des pairs | Revues de projets compréhensibles & constructives  *Indicateur : présence d’un minimum de 5 revues* | **Avant examen de 1ère session** | **1**  **solo** | Via <https://e-vinci.github.io/web2/my-reviews-page>, vous devez revoir les vidéos de présentation de 5 groupes (sauf le vôtre), exécuter leurs applications, et fournir votre critique de chacun de ces projets. Vous pourrez fournir la critique d’autant de projets que vous le souhaitez. Plus d’info sur la revue de projet au §13. |
|  | **TOTAL POINTS** |  |  | **20** | Il est à noter que des membres d’un même groupe pourront être côtés différemment en fonction de leur engagement sur le projet. L’engagement d’un étudiant est visible via les rapports individuels d’activités (outil TEAMMATES)via GitHub (GitHub Project, Issues, Milestones, commits…) et lors des sessions de cours.  Les étudiants non actifs risquent d’être écarté du projet, spécialement s’ils ne soumettent pas leurs rapports individuels.  Les étudiants n’ayant pas réalisé au moins un use case significatif seront considérés inactifs.  Les étudiants n’ayant pas participés significativement au projet recevront d’office une lourde pénalité au niveau de leurs points, voire un 0/20. |

## Ergonomie : consignes techniques, timing et évaluations

Les « deadlines » données ci-dessous sont les dates où au plus tard l’avancement des tâches doivent être présentables à un enseignant pendant le cours.

| **Compétence** | **Tâches** | **Deadline** | **Points** | **Consigne** |
| --- | --- | --- | --- | --- |
| Reporting & présentation | Objectif du projet | **17/10** |  | Décrire l’objectif de votre projet au §2 de ce document.  Discuter de votre objectif avec un enseignant et assurer vous que cet objectif soit validé avant d’aller plus loin dans votre projet. |
| Conception | Définir la vision marketing | **5/11** | 6 | Décrire le Mind map du projet. Créer le persona de (s) l’utilisateur (s) ciblé (s) par le projet. Répondre aux axiomes de Morville. |
| Analyse d’applications web | Architecture UX | **5/11** | 4 | Construire les wireframes détaillés de votre application. |
|  | Analyse des résultats et rapport associé | **17/12** | 2 | Auditez votre projet et vérifiez le respect des règles GDPR. |
|  | Présentation vidéo | **17/12** | 8 | Présenter votre projet en intégrant l’expérience utilisateur. |
|  | **TOTAL** |  | **20** | Il est à noter que différents membres d’un groupe pourront être côtés différemment en fonction de leur engagement sur le projet visible lors des sessions de cours. |

# Objectif du projet

Ce projet nous tient à cœur car il favorise l'apprentissage ludique et interactif, nous croyons que l'éducation peut être accessible, amusante et enrichissante, et c'est pourquoi nous nous engageons dans ce projet.

Notre plateforme propose une expérience enrichissante pour les utilisateurs. Ces derniers ont la possibilité de choisir parmi une variété de catégories prédéfinies, qu'il s'agisse de science, de culture, de sport ou d'autres sujets passionnants. Chacun peut trouver son domaine de prédilection.

Au sein de chaque catégorie, l'utilisateur a l’opportunité de tester ses connaissances en explorant et en participant à des quiz existants, créés par les utilisateurs eux-mêmes.

À la fin de chaque quiz, l'utilisateur reçoit un score. Si ce dernier est connecté, les points obtenus s'ajouteront à son nombre de points total. Aussi, pour pimenter l'expérience, les utilisateurs connectés ont la possibilité de gagner des badges. Chaque fois qu'ils cumulent 200 points, un badge est déverrouillé, ce qui va ainsi permettre de stimuler leur motivation à explorer et apprendre davantage.

De plus, lors des quiz, un minuteur est en marche, ajoutant un élément de défi à l'expérience. Cette fonctionnalité poussera l'utilisateur à vouloir s’améliorer.

L'utilisateur, s’il est connecté, a aussi la possibilité de créer de nouveaux quiz. Il a la liberté de concevoir ses propres quiz en choisissant la catégorie appropriée, en décidant du nombre de questions et en créant les questions et réponses possibles pour enrichir l'expérience.

Notre plateforme offre ainsi une opportunité d'apprentissage ludique et interactive, où les utilisateurs peuvent s'amuser tout en élargissant leurs connaissances.

# Mind map du projet

# Persona

# Axiomes de Morville

# Planning des tâches et cas d’utilisation

* URL vers votre GitHub Project public :

<https://github.com/orgs/e-vinci/projects/66>

# Besoins techniques

## Système

TRS01 : Vous devez développer une Single Page Application (SPA) à l’aide de JS et Node.js.

TRS02 : Votre RESTful API doit être indépendant de votre frontend ; vous aurez donc deux applications distinctes, une pour le frontend et l’autre pour la RESTful API.

TRS03 : Vous devez utiliser GitHub sur votre projet afin de gérer le développement de chacun des membres d’une équipe.

*Nous vous recommandons d’appliquer un workflow vu dans votre cours de DevOps : pour chaque cas d’utilisation / feature que vous développez, essayez de créer une branche correspondante. De plus, il serait intéressant que vous mettiez en œuvre des revues de code au sein de votre projet via des Pull Request sur Github.*

## Frontend

TRF01 : Votre frontend doit utiliser Webpack en tant que package bundler.

TRF02 : Le frontend, développé en HTML / CSS (bootstrap ou autre) / JavaScript, doit consommer au moins une de vos RESTful API.

*Votre frontend peut consommer des API externes, des APIs que vous n’avez pas développées vous-même (e.g. API de youtube, de google maps…)*

TRF03 : Votre frontend doit mettre en œuvre une librairie JS externe, ou l’API Canvas, afin de réaliser une animation.

*L’animation peut prendre la forme d’une animation 2D, 3D ou d’un jeu vidéo.   
Attention à ne pas juste offrir une minuscule animation à l’aide d’une librairie ne demandant aucun code JS, comme certaines librairies mettant tout en œuvre à l’aide de CSS.*

TRF04 : Votre frontend doit mettre en œuvre au minimum une librairie JS non vue en cours.

*Anime.js est autorisé pour votre animation.*

TRF05 : Votre frontend doit respecter les droits d’auteurs, que ça soit pour les éventuels sons, images, vidéos, librairies et morceaux de codes utilisés. Cela est de votre responsabilité et non pas de celle de vos enseignants.

TRF06 : Vous devez déployer votre frontend sur GitHub Pages ou d’autres providers gratuits supportant votre application.

## API

TRA01 : Vous devez créer une RESTful API afin d’offrir des opérations sur des ressources utiles à votre projet.

*La RESTful API ne peut pas être uniquement un « copier/coller » de ressources offertes dans le cours (notamment les ressources users et auths). Vous pouvez utiliser les ressources offertes dans le cours, mais vous devez y apporter des ajouts significatifs.*

TRA02 : Votre RESTful API doit mettre en œuvre au minimum un package non vu en cours.

TRA03 : Vous devez documenter les opérations de votre API conformément aux conventions REST.   
*Vous pouvez documenter votre API soit sous forme de tableau, comme vu dans le cours, soit à l’aide d’outils tel que Swagger.*

TRA04 : Les tests de votre API, les requêtes HTTP, doivent être données au sein de votre projet. Pour chaque opération de votre API, il doit exister au minimum une requête HTTP associée.

TRA05 : Votre API doit respecter les droits d’auteurs, que ça soit pour les éventuelles librairies utilisées, les morceaux de code, les sons, images, vidéos… Cela est de votre responsabilité et non pas de celle de vos enseignants.

TRA06 : Vous devez déployer votre backend sur Azure ou d’autres providers gratuits supportant votre application.

# Choix technologiques

## Frontend

La première librairie externe nous avons choisie d’intégrer à notre projet est « anime.js ».   
Les fonctionnalités variées qu’offre cette libraire, notamment la fonction ‘translateY’, ont été exploitée pour créer des animations agréables pour les utilisateurs, comme illustré dans la page d’inscription (RegisterPage) : <https://animejs.com/documentation/#CSStransforms.>

La deuxième librairie externe que nous avons utilisée est « Swal js » (SweetAlert2), qui offre une alternative aux boites de dialogue par défaut des navigateur web. L’utilisation de cette librairie ne répond pas directement à un besoin d’UC mais permet un rendu plus agréable pour l’utilisateur : [https://sweetalert2.github.io/.](https://sweetalert2.github.io/)

## RESTful API

Le package que nous avons décidé d’utiliser est « pg » (PostgreSQL pour Node.js).

En effet, pour la gestion des données persistantes dans notre RESTful API, nous avons choisi d’utilisé ElephantSQL, un service de base de données PostgreSQL dans le cloud. De ce fait, « pg » nous a permis d’assurer la gestion et la connexion vers la base de données depuis notre API. Ce package nous a permis d’établir une connexion sécurisée avec notre base de données et d’exécuter des requêtes SQL. : <https://www.npmjs.com/package/pg>

## Wireframe

# Conception & Implémentation

## Code repositories

* URL pour le web repository public associé à votre projet :

<https://github.com/e-vinci/web2-2023-project-group-10>

## Secrets éventuels pour vos API ou base de données

1. **Télécharger le Fichier** : Après avoir téléchargé le fichier « .env » depuis Moodle, placez-le à la racine du dossier de l'API.
2. **Utilisation** : Le fichier « .env » doit être présent à la racine du projet API avant de démarrer le serveur avec la commande « npm start ». Il sera automatiquement lu et ses variables seront intégrées à l'environnement de l'API.

## Documentation de votre API

* Tableaux représentant les opérations de votre API ou lien vers la documentation de votre API :

|  |  |  |  |
| --- | --- | --- | --- |
| Url | Method | Opérations | Test Rest Client |
| /quizzes | GET | Récupérer tous les quiz d'un utilisateur |  |
| /quizzes/readAllQuizzesByCategories/ ?label=Mathématiques | GET | Lisez tous les quiz par catégorie (ici mathématique) |  |
| /quizzes | POST | Créer un nouveau quiz |  |
| /quizzes/categories | GET | Récupérer toutes les catégories de quiz |  |
| /quizzes | DELETE | Supprimer un quiz par ID |  |
| /badges | GET | Récupérer tous les badges d'un utilisateur ou tous les badges |  |
| /badges | POST | Ajouter un badge à un utilisateur |  |
| /users | GET | Récupérer tous les utilisateurs |  |
| /users/details | GET | Récupérer les détails de l'utilisateur connecté |  |
| /users/login | POST | Authentifier un utilisateur |  |
| /users/register | POST | Enregistrer un nouvel utilisateur |  |
| /users | PATCH | Mettre à jour les points d'un utilisateur |  |

* Requêtes HTTP se trouvent dans : **/api/REST Client**

## Déploiement de vos applications

* URL de votre frontend déployé :

<https://e-vinci.github.io/Quizwiz/>

* URL de votre RESTful API déployée :

[*https://github.com/e-vinci/backendquizwiz*](https://github.com/e-vinci/backendquizwiz)

## Code réutilisé

| **Chemin du fichier où se trouve le code réutilisé** | **Auteur du code source réutilisé** | **URL où le code réutilisé est disponible** | **Raison de la réutilisation du code** |
| --- | --- | --- | --- |
| */frontend/src*  */Components/*  *pages/QuizPage* | Moataz Farid | https://www.delftstack.com/fr/howto/javascript/shuffle-array-javascript/ | Code pour randomiser un tableau |
|  |  |  |  |

# Analyse des résultats par le groupe

## Évaluation du résultat par rapport au planning des tâches et des cas d’utilisation

Les UC principales de notre projet étaient les suivantes :

* Faire la page d’accueil
* Faire la page de jeu quiz
* Faire la page de classement
* Faire la présentation vidéo
* Créer un quizz
* Afficher les quizz pour une catégorie
* Supprimer un quiz
* Afficher le score
* Gérer les badges
* Animation

Grâce à une bonne organisation de chacun des membres de l’équipe, nous avons pu finir ce que nous avions prévu de faire.

Quant aux UC changés en cours de projet, il n’y en a pas eu.

La seule modification par rapport à l’idée originale est le thème principal choisi du site.

## Audit ergonomique de votre projet

Pour notre site web nous avons choisi une lecture en Z. Tout ce qui est important pour jouer se trouve au centre de l’écran et vous ne devez pas cliquer sur plusieurs liens pour y accéder. En tête de page, avec une police suffisamment grande afin d’accrocher rapidement le regard, nous retrouvons tous les menus.

La loi de Fitts a été appliquée afin d’optimaliser la durée de manipulation de l’utilisateur. Notre site possède peu de bouton et chaque bouton est accessible et intuitif. Depuis la page de login, 3 clicks vous séparent de la page de jeu !

Les couleurs sont sobres, la police des caractères est très lisible et les images utilisées sont recherchées. Grâce à tous ces élément, l’environnement du jeu est agréable pour les utilisateurs.

Nous respectons également le RGPD. Vous avez le choix entre vous inscrire ou rester anonyme. Pour rappel la seule différence c’est qu’en anonyme votre score ne sera pas sauvegardé et vous ne pourrez pas créer vos propres quizz. Si vous vous inscrivez, vous devrez fournir un pseudo en plus d’un mot de passe et accepter les termes et conditions d’utilisation, une case est à cocher à cet effet. Le mot de passe sera bien sûr crypté dans la base de données. Par conséquent vous serez le seul à pouvoir vous connecter à votre compte.

La protection de vos données nous tient à cœur : à tout moment, vous pouvez faire une demande pour pouvoir supprimer votre compte.

## Difficultés techniques rencontrées

La première difficulté rencontrée était l’initialisation de la base de données (DB), étant donné qu’aux cours, nous n’avons pas vu comment nous connecter à une DB en ligne et comment y faire des requêtes. Après plusieurs tentatives infructueuses, nous avons pu trouver une solution sur internet.

La deuxième difficulté était liée au responsive du site, vu que chaque membre du groupe travaillait sur un écran avec des résolutions différentes, lorsqu’une page s’affichait correctement sur un écran, elle pouvait être différente sur un autre. Par exemple, la couleur en fonds d’écran de la home page n’apparaissait pas chez tout le monde.

La troisième difficulté rencontrée concernait le déploiement du backend en ligne. En effet, nous n’arrivions pas à avoir une communication avec notre DB. Après plusieurs tentatives, nous avons détectés d’où provenait le problème : il fallait configurer notre clé secrète sur Azur.

## Conseils pour appliquer cette technologie

Utiliser les Single Page Application : même si les SAP permettent d’améliorer et de fluidifier l’expérience utilisateur en évitant le chargement de page à chaque action, avoir une seule page web comme accès peut entraîner certains problèmes, c’est pour cela que nous conseillons de :

* Utiliser « eslint » qui évitera des erreurs indétectables, car cachées dans le code ;
* Faire le déploiement du site au début, ce qui vous évitera de tout devoir changer à la dernière minute à cause d’une erreur rencontrée pendant le déploiement ;
* Un des désavantages des SPA et que parfois, la page peut mettre du temps avant de charger et en sachant que la plupart des animations CSS3 peuvent être faites avec « anime.js » nous conseillons d’utiliser des librairies externes quand c’est possible.

## Quels sont les points positifs à la manière dont s’est déroulée la collaboration au sein du groupe ?

Tout au long de ce projet, la collaboration au sein du groupe a permis :

* D’établir une discipline de travail afin de permettre une progression de tous les membres.
* D’explorer des nouvelles technologies grâce à l’apport des membres du groupe.
* De progresser plus rapidement, en s’entraidant face aux problèmes techniques rencontrés.
* De se familiariser avec des outils permettant la collaboration en groupe.
* De s’améliorer individuellement, grâce aux critiques constructives et anonymes dans TEAMMATES.
* D’approfondir notre compréhension des concepts : en expliquant aux collaborateurs ce qu’on fait, on a une approche extérieure et donc une meilleure analyse.

## Quels sont les points qui seraient à améliorer pour de futures collaborations ?

Pour de futures collaborations il serait judicieux d’utiliser git plus intelligemment, avec des branches et pull request, car aux heures où nous travaillions ensemble la branche main était souvent embouteillée ce qui entrainait des conflits assez conséquents qu’il a fallu gérer.

Améliorer le timing pour les deadlines afin de pouvoir faire une relecture totale du travail à tête reposée.

# Analyses individuelles des résultats

# Présentation vidéo

Lien vers la vidéo youtube : <https://www.youtube.com/watch?v=zZg7_rE4RG4>

# Revues de projets par les pairs