

Ergonomie & développement d'une SPA animée

Zero-G odyssey

Auteur 1	Kevish Gawri
Auteur 2	Maxime Issa
Auteur 3	Lionel Janssens
Auteur 4	Edwin Musa Kalinda
Date	17.12.2023
Référence	WEB2-2023-PROJECT-GROUP-30
Version	1.0

Contents

1	Consignes et évaluations.....	3
1.1	Consignes générales.....	3
1.1.1	Création des groupes sur le site du cours.....	3
1.1.2	Création d'un groupe sur GitHub Classroom et du web repo associé.....	4
1.1.3	Projet.....	4
1.2	JavaScript & Node.js : consignes techniques, timing et évaluations.....	6
1.3	Ergonomie : consignes techniques, timing et évaluations.....	11
2	Objectif du projet.....	11
3	Mind map du projet.....	12
4	Persona.....	12
5	Axiomes de Morville.....	13
6	Planning des tâches et cas d'utilisation.....	13
7	Besoins techniques.....	16
7.1	Système.....	16
7.2	Frontend.....	17
7.3	API.....	17
8	Choix technologiques.....	18
8.1	Frontend.....	18
8.2	RESTful API.....	18
8.3	Wireframe.....	18
9	Conception & Implémentation.....	19
9.1	Code repositories.....	19
9.2	Secrets éventuels pour vos API ou base de données.....	19
9.3	Documentation de votre API.....	20
9.4	Déploiement de vos applications.....	20
9.5	Code réutilisé.....	20
10	Analyse des résultats par le groupe.....	21
10.1	Évaluation du résultat par rapport au planning des tâches et des cas d'utilisation.....	21

10.2	Audit ergonomique de votre projet.....	21
10.3	Difficultés techniques rencontrées.....	21
10.4	Conseils pour appliquer cette technologie.....	22
10.5	Quels sont les points positifs à la manière dont s'est déroulée la collaboration au sein du groupe ?.....	22
10.6	Quels sont les points qui seraient à améliorer pour de futures collaborations ?.....	22
11	Analyses individuelles des résultats.....	22
12	Présentation vidéo.....	22
13	Revue de projets par les pairs.....	24

1 Consignes et évaluations

1.1 Consignes générales

1.1.1 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.

1.1.2 Création d'un groupe sur GitHub Classroom et du web repo associé

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

Veillez passer à cette étape qu'une fois votre groupe déjà finalisé sur le site du cours.

1.1.2.1 Création de l'équipe associée à un projet

Veillez 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.

1.1.2.2 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](#).

1.1.3 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

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

Tâche	Compétences	Critères	Deadlin e	Pt	Consignes
Objectif du projet	C6) Documenter et présenter en vidéo le développement d'une SPA		Séance 9 : 16/10 ou 17/10		<p>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 :</p> <ul style="list-style-type: none"> - Le nom du projet : Projet N°X : Nom de votre projet - le champs « Description ». <p>Discuter de votre objectif avec un enseignant et assurer vous que cet objectif soit validé avant d'aller plus loin dans votre projet.</p>
Planning des tâches et cas d'utilisation	C6)		Séance 12 : 24/10 ou 27/10		<p>Décrire le planning des tâches et cas d'utilisation selon les instructions données au §6.</p> <p>Présenter votre planning à un enseignant, afin qu'il puisse vous aider à bien prioriser les tâches.</p>
Indiquer l'URL de votre code repository	C6)		Séance 12 : 24/10 ou 27/10		<p>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</p>

Tâche	Compétences	Critères	Deadlin e	Pt	Consignes
					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 <i>Indicateurs : formulation de qualité, analyse de qualité, respect des consignes</i>	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 2 ^{ème} 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.

Tâche	Compétences	Critères	Deadlin e	Pt	Consignes
					<p>Soumettre ce document, via Moodle (un devoir sera créé) ainsi que dans le répertoire /report de votre repo.</p> <p>Effacer toutes les consignes mises <i>en grisé</i> dans ce document avant de soumettre ce rapport sur Moodle.</p>
Soumission de la vidéo	C6)	<p>Vidéo de qualité</p> <p><i>Indicateurs : présentation du projet de qualité, analyse de qualité, respect des consignes</i></p>	17/12	2	Présenter votre projet selon les exigences du §12.
Soumission du code du frontend	<p>C2) Création d'IHM pour SPA inclus :</p> <p>C5 : Intégrer au développement d'une SPA des technologie non vues en cours inclus si nécessaire :</p> <p>C3) Sécurisation de SPA</p>	<p>Qualité de l'IHM produite</p> <p><i>Indicateurs : esthétique, fonctionnel, codage de qualité, respect des consignes, ambitieux & original, utilisation d'une librairie pour l'IHM non vue en cours</i></p>	17/12	8	<p>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...).</p> <p>Être ambitieux et original.</p>
Soumission du code du backend	<p>C1 : Création de services web inclus :</p> <p>C5 : Intégrer au développement d'une SPA des technologie non vues en cours inclus si nécessaire :</p> <p>C3 : Sécurisation de SPA</p>	<p>Qualité du web service produit</p> <p><i>Indicateurs : fonctionnel, codage de qualité, respect des consignes, ambitieux & original, utilisation d'une librairie pour le service web non vue en cours</i></p>	17/12	5	<p>Démontrer une appropriation personnelle du code (via commentaires dans le code, discussion lors des cours...).</p> <p>Respecter les spécifications techniques décrites dans ce document.</p>

Tâche	Compétences	Critères	Deadlin e	Pt	Consignes
					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.
Déploiement tant de votre frontend que backend	C4) Déploiement d'applications web	Déploiement de la SPA sur le cloud <i>Indicateurs : fonctionnel, performances de chargement acceptables</i>	17/12	2	
Réaliser un minimum de 5 revues sur le site web	C7) Analyser le développement de SPA faites par des pairs	Revue de projets compréhensibles & constructives <i>Indicateur : présence d'un minimum de 5 revues</i>	Avant examen de 1 ^{ère} 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 cotés 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

Tâche	Compétences	Critères	Deadline	Pt	Consignes
					<p>Project, Issues, Milestones, commits...) et lors des sessions de cours.</p> <p>Les étudiants non actifs risquent d'être écarté du projet, spécialement s'ils ne soumettent pas leurs rapports individuels.</p> <p>Les étudiants n'ayant pas réalisé au moins un use case significatif seront considérés inactifs.</p> <p>Les étudiants n'ayant pas participés significativement au projet recevront d'office une lourde pénalité au niveau de leurs points, voire un 0/20.</p>

1.3 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 cotés différemment en fonction de leur engagement sur le projet visible lors des sessions de cours.

2 Objectif du projet

Nous avons choisi de créer un jeu de plateforme dans l'espace où un vaisseau devra aller le plus loin possible en esquivant des obstacles et en collectant des ressources.

Les obstacles vont de plus en plus vite au fur et à mesure qu'on va loin dans l'espace. Il y a des étoiles à récolter. Ces étoiles permettront d'acheter des skins afin de rendre le vaisseau du joueur comme il le souhaite.

Nous voulions un jeu qui propose plus que simplement éviter des obstacles. C'est pourquoi nous avons rajouté la possibilité de tirer sur des obstacles à une certaine intervalle ainsi que la collection des étoiles, la monnaie du jeu.

Nous voulons rendre le jeu agréable visuellement c'est pourquoi nous avons décidé de donner la possibilité au joueur d'acheter des skins avec les étoiles qu'il récoltera en jeu.

3 Mind map du projet

Lien du mindmap du projet :

https://github.com/e-vinci/web2-2023-project-group_30/blob/main/ergonomics/marketing-view/MindMap_Ergo.pdf

4 Persona

Lien des persona :

https://github.com/e-vinci/web2-2023-project-group_30/blob/main/ergonomics/marketing-view/Personas_Ergo.pdf

5 Axiomes de Morville

Lien des axiomes :

https://github.com/e-vinci/web2-2023-project-group_30/blob/main/ergonomics/marketing-view/Morville.pdf

6 Planning des tâches et cas d'utilisation

- URL vers votre GitHub Project public : <https://github.com/orgs/e-vinci/projects/100>

7 Besoins techniques

7.1 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.

7.2 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.

7.3 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.

8 Choix technologiques

8.1 Frontend

Nous avons utilisé plusieurs librairies dans notre projet.

Pour commencer, nous avons utilisé la librairie phaser.io. Cette librairie s'occupe de toute la partie frontend du jeu. Nous avons pu réaliser toutes les actions que l'on voulait dans notre jeu uniquement grâce à cette librairie. Que ce soit les déplacements du vaisseau, les obstacles, les étoiles ou encore les obstacles.

Pour les animations dans plusieurs pages du frontend, nous avons utilisé animejs, que ce soit des animations en boucles comme le titre sur la homepage ou des animations déclenchés par certaines actions comme dans la boutique.

Animejs a été utilisé pour la homepage (titre et les boutons du menu), dans la page de la boutique des skins (les vaisseaux et le compte des étoiles en haut à droite) et lorsque le joueur perd dans la page du jeu..

8.2 RESTful API

Le package qu'on a utilisé est Helmet.js (<https://helmetjs.github.io/>).

8.3 Wireframe

Lien des wireframes :

https://github.com/e-vinci/web2-2023-project-group_30/tree/main/ergonomics/wireframes

9 Conception & Implémentation

9.1 Code repositories

- URL pour le web repository public associé à votre projet :
https://github.com/e-vinci/web2-2023-project-group_30

9.2 Secrets éventuels pour vos API ou base de données

9.3 Documentation de votre API

- Tableaux représentant les opérations de votre API ou lien vers la documentation de votre API : en local en faisant npm start sur l'api : <http://localhost:3000/api-docs/>
- Requêtes HTTP se trouvent dans : **/api/REST Client**

9.4 Déploiement de vos applications

*Veillez indiquer l'URL de votre frontend déployés sur <https://e-vinci.github.io/web2/project-page>.
Pour ce faire, veuillez modifier le champs « URL du site ».*

Veillez aussi indiquer ci-dessous deux URLs, comme par exemple

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

- URL de votre frontend déployé : <https://kevish-gawri-vinci.github.io/Zero-G-Odyssey>
- URL de votre RESTful API déployée : <https://zero-g-odyssey.azurewebsites.net/>

9.5 Code réutilisé

10 Analyse des résultats par le groupe

10.1 Évaluation du résultat par rapport au planning des tâches et des cas d'utilisation

Nous avons globalement atteint tous les objectifs que nous avons défini. Etant donné que nous ne connaissions pas encore les limites et possibilités des librairies choisies. Sur base de ce qu'on avait rapidement vu nous voulions au moins un jeu fonctionnel et propre.

Initialement nous voulions un jeu qui ressemble à « Geometry dash ». Mais nous avons décidé de changer car nous voulions ajouter des fonctionnalités en plus que la seule possibilité de sauter. C'est pourquoi nous avons décidé de partir sur un jeu qui avance tout seul comme « Geometry dash » mais avec des déplacements hauts et bas. Et plus tard dans le développement, la possibilité de tirer sur des obstacles.

Au fur et à mesure du développement, nous avons découvert les possibilités de Phaser.io. Nous avons donc décidé d'améliorer ce que nous avons déjà. Pour ce faire on a rajouté des sons d'ambiance et des mécaniques tels que le tir sur les obstacles.

10.2 Audit ergonomique de votre projet

Veillez utiliser les outils vus en cours pour analyser l'ergonomie de votre projet et si les règles de GDPR sont respectées. Créer ensuite ci-dessous un rapport à rédiger d'environ 10 lignes.

Tout d'abord pour ce qui est des règles de GDPR, elles sont totalement respectées grâce à nos conditions d'utilisation. En effet, nous parlons de : la collecte des données, le consentement, l'utilisation des données, le partage des données, le droit de retrait et de modifications et évidemment la sécurité des données.

10.3 Difficultés techniques rencontrées

Pour ce qui est du frontend avec phaser.io, nous n'avons pas eu de gros problèmes techniques je dirais. Ce qui nous a posé le plus de problèmes pour le frontend c'est avec github. Nous avons du mal à gérer les avancées des différents membres du groupe car on ne faisait pas tous correctement les push, pull et merge. Nous avons donc passé pas mal de temps à résoudre les conflits créés par des trop gros changements en une fois. Le moyen que nous avons trouvé à été de décortiquer tous les merge et push afin de comprendre ce qui était en conflit.

Cela a aussi été causé par une mauvaise utilisation des branches. En effet nous avons fait des branches pour les fonctionnalités mais on aurait pu encore faire des sous-branches pour éviter tout problèmes de conflits dans le code. Implémenter la librairie pour l'api était compliqué au début mais c'est car elle n'était pas adaptée pour notre projet, c'est pour cette raison qu'on s'est tourné vers helmet et ça a été directement plus simple à mettre en place. A part ça nous avons eu quelques bugs par ci par là mais ils ont souvent été rapidement corrigés. Le plus ennuyant d'entre eux était les boutons submit et les checkbox qui ne fonctionnaient pas car on avait mis un écouteur d'événement sur le main tout entier qui empêchait les comportements par défauts de tout et qui donc nous empêchait de submit.

10.4 Conseils pour appliquer cette technologie

Phaser est une librairie qui semble être presque sans limite mais qui est très simple pour des choses basiques. Il faut donc bien comprendre la base pour commencer et ensuite explorer les possibilités offertes par la librairie pour réaliser ce que l'on veut.

L'utilisation de la librairie phaser.io est relativement bien expliquée sur leur site. De plus il y avait dans le cours de JavaScript un module sur l'utilisation de cette librairie avec un cours d'un prof d'une autre école. En plus de ces documentations nous avons regardé sur internet pour trouver si des gens avaient des problèmes similaires aux nôtres.

Comme conseils on dirait avant de parler du code d'être vraiment à l'aise avec git et bien communiquer pour éviter le plus de problèmes lors des commit. Et surtout se mettre à jour avec le cours le plus rapidement possible !

Pour parler plus technologies, se renseigner sur les docs disponibles en ligne et regarder des vidéos est presque essentiel pour une bonne compréhension. Tous ces conseils sont les choses que l'on aurait préféré savoir directement ou mettre en place avant de commencer ce projet. Helmet.js n'est pas très difficile à comprendre et pour anime.js ça vient tout seul à force de pratiquer. Le meilleur conseil serait de pratiquer encore et encore. Peaufiner son projet petit à petit à force de travailler dessus et à force d'accumuler de nouvelles connaissances. Avec de la bonne volonté ça n'est pas si compliqué que ça d'appliquer ces technologies, il y a énormément de documentations et de vidéos ainsi que des outils comme les IA pour aider à comprendre tout ça.

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

Nous avons bien communiqué sur les avancées de chacun à partir d'un certain point dans le projet, ce qui nous a permis d'avancer relativement rapidement. Nous avons chacun notre partie du projet en quelque sorte. Deux personnes travaillaient sur le frontend pendant que deux personnes se concentraient sur le backend. Évidemment on s'est aidé en cas de souci sur le frontend ou backend. Si un problème persistait, on se mettait à plusieurs pour résoudre le problème et avoir des avis et manières de réfléchir différentes.

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

Nous avons mis pas mal de temps à vraiment se lancer sur le projet car nous manquions cruellement de communication au début. Ensuite notre travail avec github nous a posé beaucoup de problèmes car nous n'étions pas au point avec l'utilisation correcte de cet outil. Nous avons perdu beaucoup de temps à cause de ceci. Nous ne nous voyions pas souvent et la communication sur discord était presque inexistante.

Vers la fin du projet nous nous sommes bougés et on a fait en sorte d'aller aux cours de JS pour pouvoir se parler, et à partir de là nous avons communiqué presque tous les jours sur discord. Le travail a été beaucoup plus rapide et beaucoup plus clair. Ça a fait du bien à tout le groupe.

11 Présentation vidéo

Voici les exigences associées à votre présentation vidéo :

- *Elle doit viser une durée de 5 minutes, et ne peut pas dépasser 10 minutes.*
- *Elle doit être visible sous youtube par n'importe qui possédant son URL. Sa visibilité doit donc être en "Unlisted" ou "Public", mais pas « Private » !*
- *Elle se basera principalement :*
 - *sur la présentation de votre application web : exécution, en live, de votre API et du frontend ;*
 - *la présentation de l'expérience utilisateur ;*
- *Elle pourra aussi se baser sur d'autre(s) point(s) éventuel(s) vous permettant de vendre au mieux votre travail.*
- *Votre présentation devra être bien visible et audible.*
- *Il serait bien que celle-ci soit bien structurée, notamment via l'affichage éventuel de titres.*
- *Vous pouvez ajouter une bande son, et des images, mais seulement si celles-ci respectent les droits d'auteurs.*
- *Vous veillerez à ce que, dans la description de votre vidéo, vous fournissiez les liens vers le web repository associé à votre projet ainsi que l'URL vers le frontend déployé.*

L'idée est que si un projet intéresse des visiteurs de votre repo, ils aient accès à tout ce qui

est nécessaire pour bien le comprendre, voire pour le réutiliser, sous réserve de bien citer vos ressources.

En plus de ces exigences, la présentation vidéo a pour but de vendre un projet qui vous tient à cœur. Il est possible que si le résultat soit accrocheur, les enseignants demandent votre autorisation afin de rendre votre projet public. Avec votre autorisation, nous pourrions notamment présenter votre projet lors de salons d'étudiants, soit via votre vidéo, ou directement en exécutant votre application déployée sur le cloud.

Le site <https://e-vinci.github.io/web2/> présentera les projets qui auront été sélectionnés pour être publics. De plus, vous pourriez utiliser vos projets comme portfolio pour vos futurs employeurs.

Pour créer votre vidéo, avant de la mettre sous youtube, veillez à ce que celle-ci soit bien visible et bien audible. Nous vous recommandons :

- de la réaliser au format 1920 X 1080*
- d'utiliser un logiciel gratuit pour la réaliser. Voici ceux que nous pouvons vous conseiller :*
 - o <https://obsproject.com/> : logiciel open source demandant un temps d'adaptation, mais permettant de faire énormément*
 - o <https://www.loom.com/> : logiciel pouvant être utilisé gratuitement sous réserve d'accepter un logo. Très facile d'utilisation.*
 - o <https://screencast-o-matic.com/> : logiciel pouvant être utilisé gratuitement sous réserve d'accepter un logo. Très facile d'utilisation.*

Veillez indiquer le lien vers la vidéo youtube que vous avez créée sur le site <https://e-vinci.github.io/web2/project-page>. Pour ce faire, veuillez modifier le champ « Vidéo de présentation ».

De plus, veuillez indiquer ci-dessous ce lien :

Lien vers la vidéo youtube :