

# Ergonomie & développement d'une SPA animée

## DataMiner

<b>Auteur 1</b>	Nicolas Heymans
<b>Auteur 2</b>	Lotfi Mouayna
<b>Auteur 3</b>	Julien de Jacquier de Rosée
<b>Auteur 4</b>	Gauthier Collard
<b>Date</b>	29.11.2022
<b>Référence</b>	WEB2-2022-PROJECT-GROUP-15
<b>Version</b>	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.....	5
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.....	12
6	Planning des tâches et cas d'utilisation.....	12
7	Besoins techniques.....	12
7.1	Système.....	12
7.2	Frontend.....	13
7.3	API.....	13
8	Choix technologiques.....	14
8.1	Frontend.....	14
8.2	RESTful API.....	14
8.3	Wireframe.....	15
9	Conception & Implémentation.....	15
9.1	Code repositories.....	15
9.2	Secrets éventuels pour vos API ou base de données.....	15
9.3	Documentation de votre API.....	15
9.4	Déploiement de vos applications.....	16
9.5	Code réutilisé.....	17
10	Analyse des résultats par le groupe.....	18
10.1	Evaluation du résultat par rapport au planning des tâches et des cas d'utilisation.....	18

10.2	Audit ergonomique de votre projet.....	18
10.3	Difficultés techniques rencontrées.....	19
10.4	Conseils pour appliquer cette technologie.....	19
10.5	Quels sont les points positifs à la manière dont s'est déroulée la collaboration au sein du groupe ? .....	20
10.6	Quels sont les points qui seraient à améliorer pour de futures collaborations ? .....	20
11	Analyses individuelles des résultats.....	20
12	Présentation vidéo .....	20
13	Revue de projets par les pairs.....	21

## 1 Consignes et évaluations

### 1.1 Consignes générales

#### 1.1.1 Création des groupes sur le site du cours

Veillez former un groupe de 4 ou 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 4 ou 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 (**23/10**), 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 4-5 étudiants remplis, il restera maximum 3 étudiants non liés à un projet. Si nécessaire un ou plusieurs groupes de 3 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/7av06CzK>

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**.

e-vinci-web2

Accept the group assignment —  
web2-2022-project

Before you can accept this assignment, you must create or join a team. Be sure to select the correct team as you won't be able to change this later.

Create a new team

group-04

+ Create team


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

Après un refresh de la page qui suit, voilà ce qui apparaît :

You're ready to go —  
**group-04**

You accepted the assignment, **web2-2022-project**.

Your team's assignment repository has been created:

 <https://github.com/e-vinci/web2-2022-project-group-04>

We've configured the repository associated with this assignment ([update](#)).

Note: You may receive an email invitation to join **e-vinci** on your behalf. No further action is necessary.

Un web repository a 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'attribution via :  
<https://classroom.github.com/a/7av06CzK>.

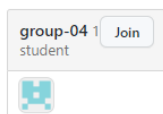
Ces membres rejoindront l'équipe existante en cliquant sur **Join** au sein de la bonne équipe. Par exemple, pour les membres du **Projet N°4**, ils cliqueront sur **Join** dans l'équipe **group-04**.

e-vinci-web2

Accept the group assignment —  
**web2-2022-project**

Before you can accept this assignment, you must create or join a team. Be sure to select the correct team as you won't be able to change this later.

Join an existing team



OR Create a new team

Create a new team

[+ Create team](#)

Si vous préférez, 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	Dead-line	Pt	Consignes
Objectif du projet	C7) Documenter et présenter en vidéo le développement d'une SPA		23/10		<p>Donnez un nom à votre projet et décrire l'objectif de votre projet au §2 de ce document ainsi que sur <a href="https://e-vinci.github.io/web2/project-page">https://e-vinci.github.io/web2/project-page</a>, complétez :</p> <ul style="list-style-type: none"><li>- Le nom du projet : Projet N°X : Nom de votre projet</li><li>- le champs « Description ».</li></ul> <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>

Tâche	Compétences	Critères	Dead- line	Pt	Consignes
Planning des tâches et cas d'utilisation	C7)		27/11		<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	C7)		27/11		<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 long de votre projet. Veuillez indiquer votre URL sur <a href="https://e-vinci.github.io/web2/project-page">https://e-vinci.github.io/web2/project-page</a>.</p> <p>Plus d'information aux §1.1.2 et §9.1.</p>
Choix technologiques	C7)		04/12		<p>Compléter le §8.</p> <p>Discuter de vos choix technologiques avec un enseignant.</p>
Rapports individuels d'activités	C7)	<p>Rapports de qualité</p> <p><i>Indicateurs : formulation de qualité, analyse de qualité, respect des consignes</i></p>	18/12	1 <b>solo</b>	<p>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.</p>

Tâche	Compétences	Critères	Dead-line	Pt	Consignes
					Tout formulaire hebdomadaire non complété amenera à une pénalité individuelle de 0.5 point.
Soumission du rapport de groupe	C7)	Idem	18/12	1	<p>Compléter le §10 ainsi que tous les paragraphes qui n'auraient pas été finalisés de ce document.</p> <p>Soumettre ce document, via Moodle (un devoir sera créé) ainsi que dans le répertoire <b>/report</b> 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	C7)	Vidéo de qualité <i>Indicateurs : présentation du projet de qualité, analyse de qualité, respect des consignes</i>	18/12	2	Présenter votre projet selon les exigences du §12.
Soumission du code du frontend	C1) Créer une IHM interactive, moderne & esthétique  Optionnel : C4) Intégrer l'authentification, l'autorisation et les sessions d'utilisateurs au sein d'une SPA	Qualité de l'IHM produite <i>Indicateurs : esthétique, fonctionnel, codage de qualité, respect des consignes, ambitieux &amp; original</i>	18/12	5	<p>Réaliser un frontend et un backend de Qualité : Code bien structuré, UI et UX de qualité Être ambitieux et original.</p> <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>
	C3) Créer une SPA intégrant	Qualité de l'intégration du	18/12	2	



Tâche	Compétences	Critères	Dead-line	Pt	Consignes
	une IHM & un web service	service web à l'IHM <i>Indicateurs : fonctionnel, codage de qualité, respect des consignes</i>			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	C2) Créer un service web de base  Optionnel : C4) Intégrer l'authentification, l'autorisation et les sessions d'utilisateurs au sein d'une SPA	Qualité du web service produit <i>Indicateurs : fonctionnel, codage de qualité, respect des consignes, ambitieux &amp; original</i>	18/12	4	
Utilisation d'une librairie pour des animations ou un jeu  Utilisation d'une librairie pour le service web	C6) Intégrer au développement d'une SPA des technologies non vues en cours	Intégration de librairies non vues en cours <i>Indicateurs : utilisation d'une librairie pour l'IHM, utilisation d'une librairie pour le service web</i>	18/12	2	
Déploiement tant de votre frontend que backend	C5) Déployer une SPA sur le cloud	Déploiement de la SPA sur le cloud <i>Indicateurs : fonctionnel, performances de chargement acceptables</i>	18/12	2	
Réaliser un minimum de 5	C8) Analyser le développement	Revue de projets	Avant examen	1 solo	Via <a href="https://e-vinci.github.io/web2/my-reviews-">https://e-vinci.github.io/web2/my-reviews-</a>

Tâche	Compétences	Critères	Dead-line	Pt	Consignes
revues sur le site web	de SPA faites par des pairs	compréhensibles & constructives <i>Indicateur : présence d'un minimum de 5 revues</i>	de 1 <sup>ère</sup> session		<a href="#">page</a> , 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.
	<b>TOTAL POINTS</b>			<b>20</b>	<p>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 Project, Issues, Milestones, commits...) et lors des sessions de cours.</p> <p>Les étudiants n'ayant pas participé activement au projet recevront d'office une lourde pénalité au niveau de leurs points, voire un 0/20.</p> <p>Les étudiants n'ayant pas réalisé au moins un use case significatif seront considérés inactifs.</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	<b>23/10</b>		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	<b>13/11</b>	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	<b>20/11</b>	4	Construire les wireframes détaillés de votre application.
	Analyse des résultats et rapport associé	<b>18/12</b>	2	Auditez votre projet et vérifiez le respect des règles GDPR.
	Présentation vidéo	<b>18/12</b>	8	Présenter votre projet en intégrant l'expérience utilisateur.
	<b>TOTAL</b>		<b>20</b>	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

Notre site web aura comme objectif un petit jeu de gestion en solitaire dont l'objectif sera déterminé par la personne qui joue, cette personne pourrait être n'importe qui, nous avons donc comme objectif de rendre le jeu clair et simple afin que tout le monde puisse y jouer. L'objectif (somme à atteindre) sera déterminé par l'utilisateur connecté qui joue, tout comme le temps attribué pour atteindre le dit objectif.

Dans le cas d'une victoire ou d'un échec nous proposerons à l'utilisateur de lancer une nouvelle partie en le faisant à nouveau choisir son niveau.  
Nous avons choisi ce projet car nous pensons qu'il serait intéressant de nous tourner vers un jeu de gestion en solitaire, en effet nous sommes tous des joueurs et avons voulu nous placer du côté du développement pour voir quelles seraient les réelles difficultés qui découlent du développement d'un jeu en équipe.

### 3 Mind map du projet

Vous trouverez le mind map dans le fichier /ergonomics/marketing-view de notre repo git.

### 4 Persona

Vous trouverez les personas dans le fichier /ergonomics/marketing-view de notre repo git.

### 5 Axiomes de Morville

Vous trouverez l'axiomes de Morville dans les fichier /ergonomics/marketing-view de notre repo git.

### 6 Planning des tâches et cas d'utilisation

- URL vers votre GitHub Project public :  
<https://github.com/e-vinci/web2-2022-project-group-15.git>

### 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 choisi d'utiliser la librairie externe animejs.

L'utilisation d'une librairie n'était pas essentielle dans le cadre de notre projet, néanmoins étant donné que les consignes nous demandaient d'en intégrer une nous avons choisi de prendre animejs.

Animejs est une petite bibliothèque JavaScript légère avec une API simple, petite et puissante. Elle fonctionne avec l'élément DOM, le CSS et JavaScript. Elle devait initialement être utilisée pour ajouter de petits effets sur notre logo ainsi que sur d'autres éléments HTML du site. Elle devait donc servir à rendre le site plus attrayant sans entraver la partie technique se trouvant derrière le site.

Malheureusement la personne en charge de la mise en place et l'utilisation de cette librairie n'est jamais parvenue à la faire fonctionner pleinement c'est pourquoi nous ne l'utilisons pas dans notre projet.

Lien vers animejs : <https://animejs.com/>

### 8.2 RESTful API

Pour la partie backend nous avons rencontré un problème lors de l'incrémentation des ressources que le joueur possède c'est alors que l'un des membres du groupe nous a proposé d'utiliser comme librairie externe de backend Lodash. Cette librairie s'est avérée sérieusement utile car elle avait une fonction add qui réglait le problème rencontré.

L'utilisation principale de Lodash est de faire des incréments. Nous l'avons donc implémentée de manière à incrémenter au fur et à mesure le nombre de ressources que le joueur possède.

Liens vers Lodash : <https://lodash.com/>

### 8.3 Wireframe

Vous trouverez le mind map dans les fichier /ergonomics/wireframes de notre repository git.

## 9 Conception & Implémentation

### 9.1 Code repositories

- URL pour le web repository public associé à votre projet :  
<https://github.com/e-vinci/web2-2022-project-group-15/tree/main>

### 9.2 Secrets éventuels pour vos API ou base de données

Nous n'avons pas de base de données SQL ou autre il n'y a donc pas de secret a vous transmettre.

### 9.3 Documentation de votre API

#### 9.3.1 Documentation des opérations de votre API

URI	Méthode HTTP	Opération
Auths/register	POST	Register a user : enregistre un user dans la base de donnée
Auths/login	POST	Login a user : connecte un user a son compte
Game/setCoal	POST	Add coal : Incrémente de 1 la ressource coal
Game/setIron	POST	Add iron : Incrémente de 1 la ressource iron
Game/setSilver	POST	Add silver : Incrémente de 1 la ressource silver
Game/setGold	POST	Add gold : Incrémente de 1 la ressource gold
Game/sellResources	POST	Add money : vent toutes les ressources
Game/lvlUp	POST	Lvl up : vous fait monter de niveau si vous avez assez d'argent

Game/getPlayer	POST	Get player : renvoie l'objet player
----------------	------	-------------------------------------

### 9.3.2 Documentation des tests de votre API

Auths/login	POST	Login : test si un utilisateur inconnu peut se connecter
Auths/login	POST	Login : test de connecter l'admin par défaut
Auths/register	POST	Register : test la création d'un nouvel utilisateur manager
Auths/login	POST	Login : test de connecter le manager
Game/getPlayer	POST	Get Player : test de récupérer les informations du player
Game/setCoal	POST	Add coal : test pour voir si le coal s'incrémente correctement
Game/setIron	POST	Add Iron : test pour voir si le iron s'incrémente correctement
Game/setSilver	POST	Add Silver : test pour voir si le silver s'incrémente correctement
Game/Gold	POST	Add Gold : test pour voir si le gold s'incrémente correctement
Game/sellResources	POST	Sell Resources : test pour voir si tout les ressources sont bien vendu
Game/lvlUp	POST	Lvl up : test pour voir si le player montes de niveau correctement

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

## 9.4 Déploiement de vos applications

Le déploiement fut infructueux



- URL de votre frontend déployé :
- URL de votre RESTful API déployée :

## 9.5 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
web2-2022-project-group-15/frontend/src/utils/auths.js	Raphael Baroni	<a href="https://github.com/e-vinci/js-demos/blob/main/frontend/frontend-essentials/cookies/src/utils/auths.js">https://github.com/e-vinci/js-demos/blob/main/frontend/frontend-essentials/cookies/src/utils/auths.js</a>	Utiliser des constantes pour la navbar
projet_web2/web2-2022-project-group-15/api/routes/auths.js	Raphael Baroni	<a href="https://github.com/e-vinci/js-demos/blob/main/backend-restful-api/restful-api-essentials/auths/routes/auths.js">https://github.com/e-vinci/js-demos/blob/main/backend-restful-api/restful-api-essentials/auths/routes/auths.js</a>	Code pour la création du login et register dans json
projet_web2/web2-2022-project-group-15/frontend/src/Components/Navbar/Navbar.js	Raphael Baroni	<a href="https://github.com/e-vinci/js-demos/blob/main/frontend/frontend-essentials/cookies/src/Components/Navbar/Navbar.js">https://github.com/e-vinci/js-demos/blob/main/frontend/frontend-essentials/cookies/src/Components/Navbar/Navbar.js</a>	Une navbar quand on est connecté
web2-2022-project-group-15/frontend/src/utils/path-prefix.js	Raphael Baroni	<a href="https://github.com/e-vinci/js-frontend-boilerplate/blob/main/src/utils/path-prefix.js">https://github.com/e-vinci/js-frontend-boilerplate/blob/main/src/utils/path-prefix.js</a>	Mise à jour des liens navbar

## 10 Analyse des résultats par le groupe

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

Au début de ce projet, nous avons pleins d'idées en tête pour ce qui était de la réalisation, mais au fur et à mesure que le travail avançait, nous nous rendions de plus en plus compte que toutes les fonctionnalités et autres détails auxquels nous avons (ou pas) pensés se sont avérés beaucoup plus complexes que prévus. Cette complexité s'est surtout reflétée dans la répartition du travail ainsi que la concrétisation des nombreux concepts auxquels nous avons pensés mais que nous n'avons pas pu réaliser par manque de temps ainsi que par limitation technique. Nous avons également pensé, dans un premier temps, à un concept plus simple que celui que nous avons tenté de mettre en place pendant ces 4 semaines. Le but initial était de créer un jeu d'arcade qui pourrait se terminer et qui afficherait donc une victoire si les objectifs demandés étaient remplis ou une défaite dans le cas contraire. C'est finalement un tout autre type de jeu que nous avons essayé de créer puisque celui-ci est infini, dans le sens où on pourrait y passer des centaines d'heures sans qu'il y ait une fin telle quelle au jeu, il n'y a donc ni potentielle victoire, ni défaite. Cette différence entre l'objectif initial et le résultat ne nous paraissait pas si grande au début mais a, au cours de notre travail posé beaucoup trop de questions sans réponses et de problèmes difficilement solvables.

### 10.2 Audit ergonomique de votre projet

Afin de respecter la loi RGPD nous voulions mettre en place un bouton « RGPD » dans la barre de navigation des utilisateurs connectés. Ce bouton RGPD aurait permis à l'utilisateur connecté d'être redirigé sur une nouvelle page et de voir en temps réel les données que nous avons collecté à son sujet, plus précisément les données du profil ainsi que celles du profil de joueur (le jeu).

L'utilisateur aurait ainsi pu voir son profil qui contient son nom d'utilisateur ainsi que son mot de passe (qui pour des raisons de sécurité serait resté chiffré) et ensuite son profil de joueur avec le nombre de ressources qu'il avait pour chacune d'entre elles ainsi que son niveau, son argent et l'argent nécessaire à ce qu'il monte de niveau.

Finalement en dessous de ces données il y aurait eu un bouton pour supprimer les données personnelles du compte ainsi qu'un autre bouton supprimant tout simplement le compte du jeu

comme la loi exige que les données collectées puissent être supprimées de notre base de données.

### 10.3 Difficultés techniques rencontrées

La première difficulté technique que nous avons rencontrée était de trouver les API, tout d'abord d'en trouver qui convient avec notre projet, qui faisaient sens et ne dénotaient pas totalement du reste du code, mais aussi une fois que l'on avait trouvé les deux, de les utiliser de façon consistante par rapport à l'implémentation globale.

Une autre difficulté rencontrée a été l'utilisation de Bootstrap, malgré les explications le concernant dans le cours, il était indispensable d'effectuer des recherches autour de toutes ses fonctionnalités afin de trouver une solution et ainsi d'avancer dans le CSS.

Il était aussi assez complexe avec Prettier/ESLint de correctement gérer les erreurs/warnings générés par exemple par des variables qui devraient en fait être des constantes mais également des « console.log » considérés comme des « warnings » par l'IDE.

La plus grande complication s'est avérée être d'envoyer des informations du backend vers le frontend concernant l'affichage du nombre de ressources alors qu'inversement, un bouton supposé incrémenter une valeur et donc passant des informations du frontend vers le backend fonctionnait parfaitement. Un dernier détail technique posant un problème était le fonctionnement de GitHub, celui-ci nous empêchait de travailler sur les mêmes fichiers en même temps et créait des « corruptions » de fichiers ce qui faisait que du travail était parfois perdu si nous n'étions pas méticuleux avec la façon dont la version « application bureau » s'utilisait.

### 10.4 Conseils pour appliquer cette technologie

La chose la plus importante que j'aurais souhaité connaître est le fait d'utiliser des méthodes POST et non GET dans les routes pour éviter de rafraîchir constamment la page et ruiner l'expérience utilisateur. Le fait de ne pas utiliser de base de données SQL rend également tous les processus moins lourds et fastidieux donc ne pas utiliser de base de données est dans notre cas bien mieux pour que le site soit bien réactif aux différentes actions de l'utilisateur.

Afin d'éviter de rencontrer les mêmes problèmes que nous il faut impérativement implémenter les fonctionnalités une par une. D'abord créer les fonction backend du jeu puis faire l'affichage du jeu en lui-même et ensuite créer les utilisateurs et donc une base de données pour enfin avoir un objet « player » unique pour chaque utilisateur. Une fois la structure bien mise en place on peut facilement implémenter des choses supplémentaires comme des améliorations de nouvelles ressources ou même de toutes nouvelles fonctionnalités.

Pour ce qui est du nombre de développeurs je pense qu'il faut une équipe restreinte du côté backend car toutes les différentes fonctionnalités vont être liées les unes aux autres, ce qui

nécessite une compréhension et une certaine maîtrise du code pour pouvoir l'utiliser a bon  
escient, éviter les redondances et éviter des bugs.

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

Beaucoup d'incompréhensions ainsi que de difficultés techniques (voir point 10.4) ont été  
rencontrées au cours de ce projet et malgré quelques complications pour ce qui est de l'organisation  
(voir point 10.6) une bonne ambiance était présente et tout le monde était ouvert à l'entraide dès  
que l'un d'entre nous en avait besoin. Cette entraide a grandement contribué à l'expansion de nos  
connaissances sur tout ce sur quoi on a travaillé durant ce projet (JavaScript, Bootstrap, GitHub,  
...). Comme dit ci-dessus, un très bon esprit d'équipe était présent ce qui exerçait une grande  
influence sur la motivation et la discipline lorsque nous étions réunis pour avancer pendant les  
sessions de travail. Pour faire court, la collaboration dans notre groupe tout au long du projet était  
très agréable, motivante et épanouissante malgré les difficultés techniques ainsi que les nombreux  
obstacles rencontrés.

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

L'organisation ainsi que la communication sont clairement des points essentiels à améliorer pour la  
suite. Étant donné l'outil que l'on utilisait pour travailler en même temps sur la partie code du projet  
(GitHub), il était indispensable de savoir quelles parties de code étaient travaillées sur le moment  
afin d'optimiser notre efficacité lors des séances de travail. De cette manière, notre organisation à  
ce niveau là n'était clairement pas au point et il fallait tout le temps communiquer quelles parties  
étaient travaillées. Concernant l'organisation, la répartition du travail était également assez floue  
durant le projet, il était ainsi difficile de différencier ce qui était fait de ce qui était encore à faire. Un  
autre point faible de notre groupe mettant tout autant à mal l'organisation serait la gestion du temps,  
étant donné le manque de clarté concernant la répartition du travail, il était très difficile d'estimer le  
temps nécessaire à la complétion du projet.

## **11 Analyses individuelles des résultats**

## **12 Présentation vidéo**

Lien vers la vidéo youtube : <https://youtu.be/b5KcEyAagBg>

## 13 Revues de projets par les pairs

Une fois la deadline de soumission des projets atteinte, la saison de revues des projets sera ouverte !

Le but ?

- Participer à la revue bienveillante, aux critiques constructives, de sites web faits par vos pairs ;
- Identifier les projets qui plaisent, notamment afin d'améliorer leur visibilité !

Pour vos revues de projets, voici les règles :

- 5 revues sont attribuées automatiquement à chaque membre d'un projet, ainsi qu'un coup de cœur lorsque vous accédez à <https://e-vinci.github.io/web2/my-reviews-page>.
- Toute revue doit comprendre au moins 1 point fort identifié et 1 point d'amélioration ; vous pouvez baser ces points suite au visionnage de la vidéo uniquement, mais nous vous recommandons de le faire après avoir exécuté l'application associée au projet ; vous pouvez bien sûr aussi accéder au code de l'application pour votre revue.
- Libre à vous de vous attacher au design, au gameplay, à l'ergonomie, au code, à la vidéo ou tout autre aspect dans votre revue. Chaque revue sera affichée – ainsi que votre nom – dans le détails d'une revue. Soyez donc bienveillants et constructifs dans votre analyse critique ; )
- Les résultats des revues ne sont accessibles qu'à vos pairs ! Votre analyse critique n'est donc pas publique, seuls les membres d'un projet de Web2 y ont accès, uniquement pour les membres ayant réalisé au moins 5 revues. Les projets sont listés par nombre de coups de cœur reçus, par nombre de revues faites, puis simplement par ordre alphabétique.
- Une revue soumise peut être modifiée. Nous vous conseillons, avant de donner un coup de cœur, d'avoir jeté un œil à tous les projets qui vous intéressent ; un coup de cœur donné ne peut pas être retiré ; )  
Par contre, vous pouvez faire vos revues tranquillement, puis mettre à jour celles-ci plus tard pour attribuer vos coups de cœur.
- Après avoir effectué vos 5 revues attribuées, vous obtenez :
  - un deuxième coup de cœur à offrir,
  - l'accès aux résultats généraux des revues et aux détails de chacune des revues.
  - Le pouvoir de revoir n'importe quel projet non encore revu.

Comme l'hébergement gratuit d'API devient de plus en plus compliqué, nous souhaitons garantir que les 3 projets les plus aimés puissent bénéficier d'un hosting offert et géré par la HE Vinci.

NB : Ce §13 peut être entièrement effacé du rapport que vous soumettrez sur Moodle.