Ergonomie & développement d’une SPA animée

**COVID CLICKER**

|  |  |
| --- | --- |
| **Auteur 1** | Ibrahim AYACHI |
| **Auteur 2** | Nicolae PASTRAMA |
| **Auteur 3** | Hamadi GADIH |
| **Auteur 4** | Teodor ARBUREANU |
| **Auteur 5** | Matteo ERISMANN |
| **Date** | 16.12.2023 |
| **Référence** | WEB2-2023-PROJECT-GROUP-02 |
| **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

Le projet du groupe 2 « Covid Clicker » est un idle game de type clicker. Il s’agit d’un jeu incrémental dont le but est de faire accroitre son score en cliquant, grâce au score accumulé l’utilisateur pourra acheter diverses améliorations augmentant la valeur de son clique ou générant des cliques automatiques par secondes.

L’utilisateur pourra également avoir accès a un tableau des joueurs ayant les scores les plus élevé afin de stimuler un esprit de compétition et son envie d’augmenter son score.

Cette application vise toute personnes voulant se détendre ou adepte des jeux incrémentaux. Le jeu est « Familly Friendly » et convient donc à toute personne de tour âge.

Ce projet nous tenait à cœur car il permet de ramener une touche de légèreté, grâce a un design arcade et des couleurs vive, le thème du covid qui était un élément de stresse pour la planète durant ces 3 dernières années.

# Mind map du projet

* <https://github.com/e-vinci/web2-2023-project-web-2023-project-group-2/blob/main/ergonomics/marketing-view/mindMap.png>

# Persona

* <https://github.com/e-vinci/web2-2023-project-web-2023-project-group-2/blob/main/ergonomics/marketing-view/Personas.pdf>

# Axiomes de Morville

# Planning des tâches et cas d’utilisation

* <https://github.com/orgs/e-vinci/projects/97>

# 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

Notre application étant principalement composé d’élément html, nous avons fait le choix d’utiliser la librairie externe Animejs. Cette libraire fonctionne à l’aide d’une API et permet l’animation 2D sur des éléments CSS, SVG, attribut DOM ou objets JavaScript.

Cette libraire nous a été utile pour les différentes animations présente sur le site, le jeu étant principalement fait d’élément html & CSS il nous fallait des indicateurs pour signaler au joueur que son action s’est produite. C’est pourquoi nous avons utilisé la librairie pour animer le covid sur lequel clique l’utilisateur et lui faire comprendre que sa clique a été pris en compte. Une autre utilisation est une animation de rafraîchissement de la boutique à chaque achat de l’utilisateurs, encore une fois dans le but d’indiquer visuellement que sont achat a été effectué, ainsi l’utilisateur ne se retrouva face à une simple page inanimée avec un compteur qui augmente.

Outre-le coté purement fonctionnel, nous nous sommes également servis de la libraire pour l’aspect esthétique qu’elle pouvait amener à notre site, les animations d’arrière-plan en sont un exemple.

Nous avons également utilisé la librairie Validator, Validator est une librairie permettant d’évaluer et de valider une grande quantité de donnée textuelle.

Dans le cadre de notre application nous avons utilisé Validator afin de s’assurer que le mot de passe entré par l’utilisateur lors de la création de son compte est assez puissant.

## Frontend

* Animejs : [*https://animejs.com*](https://animejs.com)

## RESTful API

* Validator : <https://www.npmjs.com/package/validator>

## Wireframe

* <https://github.com/e-vinci/web2-2023-project-web-2023-project-group-2/blob/main/ergonomics/wireframes/wireframes%20(1).pdf>

# Conception & Implémentation

## Code repositories

* <https://github.com/e-vinci/web2-2023-project-web-2023-project-group-2>

## Secrets éventuels pour vos API ou base de données

## Documentation de votre API

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

|  |  |  |
| --- | --- | --- |
| URI | Méthode http | Opération |
| Auths/register | POST | Enregistrer un utilisateur |
| Auths/login | POST | Loguer un utilisateur |
| Clicker/registerScore | POST | Enregistrer le score d’un utilisateur |
| Clicker/valueClickUser | POST | Permet de récupérer la valeur du click d’un utilisateur |
| Clicker/valueAutoClickUser | POST | Permet de récupérer la valeur de l’auto click d’un utilisateur |
| Clicker/scoreUser | POST | Permet de récupérer le score de l’utilisateur |
| leadrBord/ | GET | Permet de récupérer le score de tous les utilisateu |
| Upgades/readAll | POST | Lire toutes les amélioration d’un utilisateur |
| userUpgrades/ | PATCH | Permet d’améliorer une upgrade |

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

Auth.https

* Test de connexion pour un utilisateur inconnu
* ### Try to login an unknow user
* POST {{baseUrl}}/auths/login
* Content-Type: application/json
* {
* "username":"unknown",
* "password":"admin"
* }
* Connexion de l’admin par défaut
* ### Login the default admin
* POST {{baseUrl}}/auths/login
* Content-Type: application/json
* {
* "username":"admin",
* "password":"admin"
* }
* Création d’un utilisateur « manageur »
* ### Create the manager user
* POST {{baseUrl}}/auths/register
* Content-Type: application/json
* {
* "username":"ibrahil2",
* "password":"manager.2003",
* "confirmPassword":"Manager.2003"
* }
* Création d’un utilisateur « manageur » sans confirmation de mot de passe

Envoie une erreur

* ### Create the manager user
* POST {{baseUrl}}/auths/register
* Content-Type: application/json
* {
* "username":"manager",
* "password":"Manager.2003"
* }
* Test de connexion de l’utilisateur « manageur »
* ### Login the manager user
* POST {{baseUrl}}/auths/login
* Content-Type: application/json
* {
* "username":"manager",
* "password":"Manager.2003"
* }

Clicker.https

* Ajout de score pour « admin »
* ###Register score for admin
* POST {{baseUrl}}/clicker/registerScore
* Content-Type: application/json
* {
* "username":"admin",
* "nvxPoints": 125
* }
* Ajout de score pour « manager »
* ###Register score for manager
* POST {{baseUrl}}/clicker/registerScore
* Content-Type: application/json
* {
* "username":"manager",
* "nvxPoints": 323
* }
* Ajout d’une amélioration pour « admin »
* ####Upgrade clicker
* PATCH {{baseUrl}}/clicker/upgradeClicker
* Content-Type: application/json
* {
* "username": "admin",
* "upgradeID": 2
* }
* Récupération du score pour « admin »
* ####Get score
* post {{baseUrl}}/clicker/scoreUser
* Content-Type: application/json
* {
* "username": "admin"
* }

LeaderBoard.https

* Récupération de la liste des utilisateurs
* #### Get users table
* GET http://localhost:3000/leaderboard

Upgrades.https

* Récupération de toutes les améliorations
* ### Read all upgrades
* GET http://localhost:3000/upgrades
* Récupération des améliorations avec variables
* ### Read all upgrades with File variable
* @baseUrl = http://localhost:3000
* GET {{baseUrl}}/upgrades
* Tri des améliorations par ordre croissant des améliorations
* ### Read all upgrades sorted by operation (ascending)
* GET {{baseUrl}}/upgrades/?operation=add
* Création d’une amélioration en tant qu’admin
* #### Create a upgrade with the admin token
* POST {{baseUrl}}/upgrades
* Content-Type: application/json
* Authorization: {{admin.response.body.token}}
* {
* "title":"click: x10",
* "operation":"multiply",
* "cost": 2000,
* "upgradeClickerValue": 10
* }
* Suppression de l’amélioration 2 en tant qu’admin
* ### Delete upgrade identified by 2 with the admin token
* DELETE {{baseUrl}}/upgrades/6
* Authorization: {{admin.response.body.token}}
* Suppression de l’amélioration 6 en tant qu’admin
* ### Update the upgrade identified by 6 with the admin token
* PATCH {{baseUrl}}/upgrades/6
* Content-Type: application/json
* Authorization: {{admin.response.body.token}}
* {
* "title":"test"
* }
* Récupération des améliorations d’un utilisateur par son pseudo
* ### Read all upgrades from user by username
* POST {{baseUrl}}/upgrades/readAll
* Content-Type: application/json
* {
* "username":"admin"
* }

User\_upgrades.http

* Récupérations de toutes les améliorations d’un utilisateur
* ###Read all user\_upgrades from user
* GET {{baseUrl}}/userUpgrades/1
* Ajout d’une amélioration pour l’utilisateur « admind »
* ### Add one upgrade for user
* # *@name* *admin*
* POST {{baseUrl}}/userUpgrades/2
* Content-Type: application/json
* {
* "idUpgrade":4
* }
* Ajout d’une amélioration pour l’utilisateur 1
* ### Buy upgrade for user 1
* PATCH {{baseUrl}}/userUpgrades
* Content-Type: application/json
* {
* "idUpgrade":5,
* "username":"admin"
* }

## Déploiement de vos applications

* URL de votre frontend déployé : <https://github.com/e-vinci/web2-2023-project-web-2023-project-group-2/tree/main/frontend>
* URL de votre RESTful API déployée : <https://github.com/e-vinci/web2-2023-project-web-2023-project-group-2/tree/main/api>

## 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/stylesheets/main.css | Tahmid Ahmed | [*https://www.youtube.com/watch?v=LqRfQVcn3FY*](https://www.youtube.com/watch?v=LqRfQVcn3FY) | Car la vidéo donnait un code pour animer le background et on en avait besoin. On savait aussi pas comment faire |
| frontend/src/Components/Navbar/Navbar.js | DesignCourse | <https://www.youtube.com/watch?v=rfpRZ2t_BrQ&t=255s> | besoin d'un prototype de code pour changer le curseur et ce bout de code a permis de placer le curseur au même niveau que celui par défaut afin de pouvoir naviguer et cliquer correctement |
|  |  |  |  |

# Analyse des résultats par le groupe

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

Nous avons entrepris avec succès la réalisation de notre application en respectant la plus grande partie des objectifs fonctionnels fixé. L’inscription d’un nouvel utilisateur, la connexion, le click du joueur sur le covid, la gestion de son score, la possibilité d’achats d’amélioration et l’accès a une page de classement sont autant de cas d’utilisation que nous avons réussi à intégrer de manière satisfaisante.

En plus de ces objectifs fonctionnels, nous avons pu atteindre nos objectifs esthétiques grâce a l’intégration d’animations, de sons ou d’autres éléments visuels touchant plus au CSS qui rendent notre application plaisante et améliorent l’expérience utilisateurs.

Cependant il est important de noter que nous n’avons pas respecté un de nos objectifs fonctionnels qui était la possibilité de changer le « skin » du covid, fonctionnalité que nous avion prévu initialement. Nous avons fait le choix de laisser tomber cette fonctionnalité pour nous concentrer sur d’autres éléments plus importants du jeu tels que les améliorations. Il nous aurait également fallu trouver des images libres de droit s’inscrivant dans la direction artistique du site. Pour compenser nous nous sommes concentrés sur le fait renforcer l’identité visuelle de notre application et somme parvenu a un résultat qui nous satisfait.

## Audit ergonomique de votre projet

Pour ce projet, nous avons voulu le rendre le plus facile possible pour l’expérience utilisateur, étant donné que c’est un idle game. Nous nous sommes limités à 4 boutons pour la barre de navigation et un simple bouton géant avec un titre pour la page d’accueil pour ne pas fournir du contenu non utile à l’utilisateur. De même, pour la page de jeu, on s’est limité à 4 types d’objets : un bouton principal au centre pour cliquer, des améliorations, un score et une barre de progrès pour tenir compte des limitations de mémoire de l’utilisateur. Pour respecter le GDPR, nous avons choisis de créer une page qui explique les termes pour les données sur la vie privée qui est accessible à partir de la page d’enregistrement et qui doit être accepté avant de s’enregistrer.

## Difficultés techniques rencontrées

L’une des principales difficultés rencontrées lors du développement de l’application est la gestion des prix des amélioration. Une amélioration augmentant de prix à chaque fois que l’utilisateur l’achète, chaque utilisateur aura donc un prix différent pour chaque amélioration. Il a fallu trouver un moyen pour retenir à quel stade chaque utilisateur est, ce qui nous a fait repenser notre approche technique.

Un autre problème était la gestion de l’utilisateur, un utilisateur pouvant effectuer un grand nombre de cliques par seconde, il a fallu mettre en place un moyen pour ne pas surcharger de requête l’API tout en montrant visuellement a l’utilisateur que son score augmente bien.

Il nous a également fallu un temps d’adaptation avec la librairie animejs, apprendre une technologie totalement nouvelle fut un peu complexe au début et il nous a fallu repasser plusieurs fois sur l’une ou l’autre animation afin qu’elle corresponde exactement à ce que nous avions en tête.

## Conseils pour appliquer cette technologie

Pour développer un idle game, on pense que pour bien démarrer, il faut concevoir des beaux wireframes avec une bonne expérience utilisateur. Au début, lorsqu’on a énuméré nos taches, on n’a pas su penser qu’on allait devoir gérer le prix de chaque amélioration pour le clicker et par conséquent attribuer chaque amélioration achetée à l’utilisateur connecté pour ainsi changer son prix. Lorsqu’on a commencé, on avait aussi une vision trop ambitieuse en voulant avoir beaucoup d’animations différentes et de fonctionnalités difficiles pour le frontend ce qui nous a fait perdre du temps. Alors ça serait, peut-être, plus judicieux de commencer avec un frontend simple et d’ajouter plusieurs animations petit à petit après que les fonctionnalités principales soient développées. On conseil aussi, pour le score, de penser à son fonctionnement avant de commencer car il changera souvent dans ce genre de jeu avec des améliorations et la valeur du click qui change. On a trouvé que la meilleure façon de le développer c’est de le faire avec deux variables. Une pour le score affiché qui s’incrémente et qui est envoyé à l’api et une autre qui reprends le score du user depuis l’api et qui remplace le score affiché au début du chargement de la page.

Exemple de clicker :   
[981 cookies - Cookie Clicker (dashnet.org)](https://orteil.dashnet.org/cookieclicker/)  
[Jeux de Clicker 🖱️ Joue sur CrazyGames](https://www.crazygames.fr/c/clicker)

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

Dans l’ensemble, la collaboration au sein du groupe s’est bien passée et plusieurs points ont facilité cette entente :

Tout d’abord, l’un des principaux points positifs est que chaque membre avait une vision globale du projet en accord avec le reste du groupe, il n’y a pas eu de grands débats sur quelle direction devait partir le projet, les discussions étaient plutôt centrées sur de petits éléments.

Il y a eu du respect et de la collaboration dans le groupe, une personne ayant un problème ou une idée pouvait aller en voir une autre, personne ne restait de son coté à avancer seul.

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

Peut-être une meilleure utilisation de GitHub pour l’aspect branching, pull request etc…

Durant notre projet, nous n’avons travaillé qu’avec des push/pull sur la branche main, il n’y a pas eu de gros conflits mais cela aurait pu être optimisé.

Il faudrait peut-être aussi, dès le début, fragmenter les grande tâches en plus petites sous tâche de sort a avoir des valeur plus quantifiable su l’état d’avancement du projet, de mieux les répartir et de voir les liens éventuelles entre chacune d’elles.

# Présentation vidéo

Lien vers la vidéo youtube :   
https://youtu.be/GCFDOtr36BY