Ergonomie & développement d’une SPA animée

**INCLURE ICI LE NOM DE VOTRE PROJET**

Quiz en ligne

|  |  |
| --- | --- |
| **Auteur 1** | Younes Benbouchta |
| **Auteur 2** | Quoc An Duong |
| **Auteur 3** | Trong Nguyen |
| **Auteur 4** | Maxime Vlaminck |
| **Auteur 5** | Ke man Yi Nghi |
| **Date** | 18.12.2022 |
| **Référence** | WEB2-2022-PROJECT-GROUP-17 |
| **Version** | 1.0 |

Contents

[1 Consignes et évaluations 3](#_Toc115852290)

[1.1 Consignes générales 3](#_Toc115852291)

[1.1.1 Création des groupes sur le site du cours 3](#_Toc115852292)

[1.1.2 Création d’un groupe sur GitHub Classroom et du web repo associé 4](#_Toc115852293)

[1.1.3 Projet 5](#_Toc115852294)

[1.2 JavaScript & Node.js : consignes techniques, timing et évaluations 6](#_Toc115852295)

[1.3 Ergonomie : consignes techniques, timing et évaluations 11](#_Toc115852296)

[2 Objectif du projet 11](#_Toc115852297)

[3 Mind map du projet 12](#_Toc115852298)

[4 Persona 12](#_Toc115852299)

[5 Axiomes de Morville 13](#_Toc115852300)

[6 Planning des tâches et cas d’utilisation 13](#_Toc115852301)

[7 Besoins techniques 15](#_Toc115852302)

[7.1 Système 15](#_Toc115852303)

[7.2 Frontend 15](#_Toc115852304)

[7.3 API 16](#_Toc115852305)

[8 Choix technologiques 17](#_Toc115852306)

[8.1 Frontend 17](#_Toc115852307)

[8.2 RESTful API 17](#_Toc115852308)

[8.3 Wireframe 17](#_Toc115852309)

[9 Conception & Implémentation 17](#_Toc115852310)

[9.1 Code repositories 17](#_Toc115852311)

[9.2 Secrets éventuels pour vos API ou base de données 18](#_Toc115852312)

[9.3 Documentation de votre API 18](#_Toc115852313)

[9.4 Déploiement de vos applications 19](#_Toc115852314)

[9.5 Code réutilisé 19](#_Toc115852315)

[10 Analyse des résultats par le groupe 20](#_Toc115852316)

[10.1 Evaluation du résultat par rapport au planning des tâches et des cas d’utilisation 20](#_Toc115852317)

[10.2 Audit ergonomique de votre projet 20](#_Toc115852318)

[10.3 Difficultés techniques rencontrées 20](#_Toc115852319)

[10.4 Conseils pour appliquer cette technologie 20](#_Toc115852320)

[10.5 Quels sont les points positifs à la manière dont s’est déroulée la collaboration au sein du groupe ? 21](#_Toc115852321)

[10.6 Quels sont les points qui seraient à améliorer pour de futures collaborations ? 21](#_Toc115852322)

[11 Analyses individuelles des résultats 21](#_Toc115852323)

[12 Présentation vidéo 21](#_Toc115852324)

[13 Revues de projets par les pairs 22](#_Toc115852325)

# Consignes et évaluations

## Consignes générales

### Création des groupes sur le site du cours

Veuillez 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.

### 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/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**.

Graphical user interface, text, application

Description automatically generated

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

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

Graphical user interface, text, application

Description automatically generated

Un web repository a 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/7av06CzK>.

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**.

Graphical user interface, text, application, email

Description automatically generated

Si vous préférez, 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** | **Dead-line** | **Pt** | **Consignes** |
| --- | --- | --- | --- | --- | --- |
| Objectif du projet | C7) Documenter et présenter en vidéo le développement d'une SPA |  | **23/10** |  | 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 | C7) |  | **27/11** |  | 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 | C7) |  | **27/11** |  | 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 | C7) |  | **04/12** |  | Compléter le §8.  Discuter de vos choix technologiques avec un enseignant. |
| Rapports individuels d’activités | C7) | Rapports de qualité *Indicateurs : formulation de qualité, analyse de qualité, respect des consignes* | **18/12** | **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.  Tout formulaire hebdomadaire non complété amenera à une pénalité individuelle de 0.5 point. |
| Soumission du rapport de groupe | C7) | Idem | **18/12** | **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 | C7) | Vidéo de qualité  *Indicateurs : présentation du projet de qualité, analyse de qualité, respect des consignes* | **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  *Indicateurs : esthétique, fonctionnel, codage de qualité, respect des consignes, ambitieux & original* | **18/12** | **5** | Réaliser un frontend et un backend de Qualité : Code bien structuré, UI et UX de qualité  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. |
|  | C3) Créer une SPA intégrant une IHM & un web service | Qualité de l’intégration du service web à l’IHM  *Indicateurs : fonctionnel, codage de qualité, respect des consignes* | **18/12** | **2** |
| 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  *Indicateurs : fonctionnel, codage de qualité, respect des consignes, ambitieux & original* | **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  *Indicateurs : utilisation d’une librairie pour l’IHM, utilisation d’une librairie pour le service web* | **18/12** | **2** |  |
| Déploiement tant de votre frontend que backend | C5) Déployer une SPA sur le cloud | Déploiement de la SPA sur le cloud  *Indicateurs : fonctionnel, performances de chargement acceptables* | **18/12** | **2** |  |
| Réaliser un minimum de 5 revues sur le site web | C8) 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 n’ayant pas participés activement au projet recevront d’office une lourde pénalité au niveau de leurs points, voire un 0/20.  Les étudiants n’ayant pas réalisé au moins un use case significatif seront considérés inactifs. |

## 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 | **23/10** |  | 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 | **13/11** | 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 | **20/11** | 4 | Construire les wireframes détaillés de votre application. |
|  | Analyse des résultats et rapport associé | **18/12** | 2 | Auditez votre projet et vérifiez le respect des règles GDPR. |
|  | Présentation vidéo | **18/12** | 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

*Décrire le but du projet que vous souhaitez développer. Cette description (~15 lignes) doit notamment répondre à ces questions : quel type d’organisation (équipe sportive, bibliothèque, amis, famille…) sera prise en charge par votre application ?*

*Pourquoi cela vous tient à cœur ?*

*Vous pouvez envisager n’importe quel type d’application respectant les consignes du §1.1.*

*Veuillez résumer l’objectif de votre projet sur* [*https://e-vinci.github.io/web2/project-page*](https://e-vinci.github.io/web2/project-page) *dans le champ « Description ».*

*Veuillez aussi donner un nom à votre projet que vous compléterez sur* [*https://e-vinci.github.io/web2/project-page*](https://e-vinci.github.io/web2/project-page) *ainsi que sur la première page de ce rapport.*

*Pour décider de l’objectif de votre projet, vous pouvez vous inspirer de ce qui a été fait les années précédentes par les étudiants de web 2 en consultant la vitrine de projets offerte sur le site du cours.*

*Voici quelques autres exemples d’applications que vous pourriez réaliser :*

* *Une gestion de critiques de livres,*
* *Une gestion d’une équipe sportive (gestion des achats ou autres de l’équipe),*
* *Un jeux vidéo (multi-joueurs ou pas) dont les utilisateurs et leurs scores sont gérés par une RESTful API…*
* *Un jeu de devinette d’un mot sur base du dessin d’un des joueurs*
* *…*
* *Pour les étudiants qui n’ont toujours pas d’idées, voici ici la liste des projets proposés il y quelques années :*

**

# Mind map du projet

*Veuillez créer le Mind map de votre application en allant le plus en détails. N’oubliez pas de regrouper en catégories pertinentes comme vu en cours.*

*Veuillez soumettre votre mind map au sein du répertoire* ***/ergonomics/marketing-view*** *de votre repo.*

# Persona

*Veuillez créer au moins 2 personas de votre audience cible. Le but étant de représenter au mieux les personnes qui seront le plus engagés par votre application.*

*Veuillez soumettre vos persona au sein du répertoire* ***/ergonomics/marketing-view*** *de votre repo.*

# Axiomes de Morville

*Veuillez décrire chaque axiome de Morville de manière assez complète vis-à-vis de votre application. Ne pas donner de réponse basique mais allez plus loin dans les besoins.*

*Exemple : App de la ville de Bruxelles*

* *Valable : “parce que les gens en ont besoin et ça les aidera”*

*--> Une application aide toujours mais en quoi cela rajoute de la plus-value ?*

*--> En quoi l’application va*

*réellement changer l’expérience utilisateur déjà existante pour ce type de projet ? Et si cela ne change pas, il faudra aussi le justifier. Ne pas recréer la roue n’est pas un défaut en soi, ça s’appelle de l’efficience.*

*Veuillez soumettre vos axiomes au sein du répertoire* ***/ergonomics/marketing-view*** *de votre repo.*

# Planning des tâches et cas d’utilisation

*Au sein du repository GitHub de votre projet, vous devez créer un* ***Project*** *pour planifier les tâches, allouer les responsables, documenter vos avancées, visualiser vos* ***Milestones****...*

*Votre projet doit être* ***public*** *et doit suivre le template :* ***New project****,* ***Team backlog****.*

*Vous pouvez supprimer la colonne* ***New****.*

*Voici une vidéo expliquant la mise en place de votre GithHub Project :* [*Partie1 : Créer et configurer un GitHub Project*](https://youtu.be/y2e_ItjAZ2I)*.*

*Voici le workflow que nous souhaitons vous voir appliquer sur GitHub Project :*

* *Veuillez commencer votre projet en identifiant toutes les tâches principales à réaliser sur votre projet, principalement en soignant l'identification des use cases. Pour ce faire, vous pouvez visualiser la vidéo :* [*Partie 2 : planifier des tâches via GitHub Project*](https://youtu.be/hz0P-PfYqgo)*.*
* *Chaque****tâche****doit être couverte par une****draft Issue****au sein de GithHub que vous devrez convertir plus tard en****Issue****. Veuillez allouer une****Priority****à chaque****Issue****(ou****draft Issue****), ainsi qu'une****Size****.*
* *Lors de l'identification des tâches, les****Issues****associées se trouvent dans la colonne****Backlog****.*
* *Un****cas d'utilisation****doit être couvert par au moins une****Issue****avec un label nommé****enhancement****.*
* *Chaque****Issue****doit être associée à au moins un****Assignee****.*
* *S'il y a plusieurs****Assignees****associés à une****Issue****, celle-ci devra être découpée en suffisamment de tâches pour qu'il y ait un seul****Assignee****par tâche. Pour la découpe d’une Issue en tâches, la création de Label, et la gestion de Milestones, vous pouvez visualiser la vidéo :* [*Partie 3 : gestion approfondie des tâches via GitHub Project*](https://youtu.be/F8lPoL8qiZA)*.*
* *Dans un premier temps, une****Issue****associée à plusieurs****Assignees****peut simplement identifier les tâches associées au sein d'une****task list****. Plus tard, ces tâches devront être converties en nouvelles****Issues****associées à un seul****Assignee****.*
* *Lorsque vous travailler sur une****Issue****:*
  + *elle doit se trouver dans la colonne****In progress****ou****In review****si vous pensez avoir terminé mais que vous attendez le feedback d'un membre de votre projet.*
  + *pour chaque avancée significative sur une****Issue****, vous devez indiquer un commentaire via****Comment****résumant le travail effectué.*
* *Lorsque vous considérez qu'une****Issue****est terminée, faites la passer dans la colonne****Done****, indiquez un message via****Comment****résumant le travail effectué et cliquez sur****Close issue****.*
* *Pour facilement voir le pourcentage de progrès dans la fermeture des****Issues****qui vous sont associées, vous devez créer une****Milestone****par membre de projet et associer cette****Milestone****à toutes les****Issues****où le membre de projet est le seul****Assignee****.*

*Pour visualiser les tâches d’un seul membre de projet, il est possible de faire, via GitHub Project, autant de* ***view*** *qu’il y a de membres de projet, ou une seule vue pour chaque utilisateur connecté. Pour obtenir une vue pour un membre de projet, il suffit de cliquer sur la* ***view****, puis* ***filter****, puis d’écrire :* ***assignee:*** *et d’indiquer le username d’un membre du projet. Il est aussi possible de créer une vue des tâches de l’utilisateur connecté, qui pourra être réutilisée par tous les membres du groupe, en indiquant :* ***assignee:@me*** *pour le* ***filter****.*

*Pour votre développement, n’hésitez pas à être ambitieux, tout en restant réaliste. Comment faire ? Nous vous recommandons de spécifier les cas d’utilisations qui* ***doivent*** *être implémentés par une priorité «****Haute****». Ceux qui* ***pourraient*** *être implémentés, mais qui ne sont donc pas indispensables à l’application de base, vous devriez les catégoriser selon une priorité « Moyenne » ou « Basse ».*

*Afin de vous aider dans la création de votre planning au sein de GitHub Project, nous vous proposons une liste de base d’****Issues*** *à prendre en compte**:*

* *Create marketing vision (documentation)*
* *Create wireframes (documentation)*
* *Identify list of use cases (documentation)*
* *UC1 : ... (enhancement)*
  + *UC1-frontend : ... (enhancement)*
  + *UC2-api : ... (enhancement)*
    - *Create API HTTP requests (tests)*
* *UC2 : ... (enhancement)*
* *UC3 : ... (enhancement)*
* *UC...*
* *Document API (documentation)*
* *Deploy frontend (deployment)*
* *Deploy API (deployment)*
* *Create Project Report (documentation)*
* *Create Video (documentation)*

*Notons que les rapports individuels d’activités que vous devrez compléter chaque semaine, suite à invitation par e-mail, ne sont pas à reprendre dans votre planning de tâches.*

*Attention que chaque membre de projet doit avoir au moins un use case significatif pour être considéré actif.*

*Veuillez indiquer l'URL vers votre* ***Project*** *public sous GitHub ici :*

* URL vers votre GitHub Project public :

# Besoins techniques

*La spécification technique des besoins vous est donnée*

*N’hésitez pas à la compléter s’il manque qqch d’important.*

## 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

*Principalement pour la ou les librairies JS non vue en cours que vous avez ou allez appliquer, veuillez décrire la technologie choisie pour répondre à un ou plusieurs cas d’utilisation identifié(s) au §6. Ce paragraphe est à rédiger sur environ 20 lignes. Veuillez décrire les liens vers le ou les sites à utiliser (ou utilisés), les ressources principales utiles au développement.*

## Frontend

*Pour le frontend, voici des exemples de librairies open source qui pourraient être choisies, en fonction de votre objectif :*

* *pour la 2D :* [*https://animejs.com*](https://animejs.com)
* *pour la 3D :* [*https://threejs.org*](https://threejs.org)
* *pour les jeux :* [*https://phaser.io*](https://phaser.io)

## RESTful API

*Pour votre Restful API, vous devez découvrir un package non vu en cours pour réaliser l’une ou l’autre des fonctionnalités. Veuillez indiquer ici le ou les packages choisi(s) avec les liens vers les sites utilisés.*

## Wireframe

*Veuillez créer sur PowerPoint, des wireframes de votre application. Il en faut au minimum 3.*

*Exemple d’un jeu : il y aura au minimum, l’accueil inscription, le jeu, une page de résultat, page contact développeur, etc.*

*Veuillez soumettre vos wireframes au sein du répertoire* ***/ergonomics/wireframes*** *de votre repo.*

# Conception & Implémentation

## Code repositories

*Veuillez indiquer l’URL de votre web repository public, générée par GitHub Classroom, sur* *https://e-vinci.github.io/web2/project-page. Pour ce faire, veuillez au moins modifier le champs « Repo frontend » (et éventuellement « Repo backend »). L’URL du web repository de votre groupe devrait correspondre à qqch du style « https://github.com/e-vinci/web2-2022-project-group-04 » si vous êtes membres du Projet N°4.*

*Veuillez aussi indiquer cette URL ci-dessous.*

* URL pour le web repository public associé à votre projet : https://github.com/e-vinci/web2-2022-project-group-13

*Il est important que pour explorer vos projets, on puisse facilement installer votre frontend et backend. Veuillez veiller à ce que votre application s’installe et s’exécute simplement localement via ces actions :*

* *git clone de votre backend*
* *npm install et npm au niveau du backend*
* *git clone du frontend*
* *npm install et npm start au niveau du frontend*

## Secrets éventuels pour vos API ou base de données

*Si vous utilisez une base de données ou des API nécessitant des secrets, il est important de ne pas rendre public vos secrets. Dans ce cas :*

*- Votre application doit être sur le cloud pour que les autres étudiants puissent la revoir ; les étudiants ne pourront donc pas exécuter l'API localement. Veuillez clairement indiquer dans le README de votre projet si l’application ne peut pas être exécutée localement sans les secrets et veuillez indiquer l'URL tant de votre frontend que de votre API au sein de ce README.*

*- Vous devez mettre à disposition tous ces secrets (fichiers de configuration) à disposition de vos enseignants lors de la soumission de ce rapport. Le devoir Moodle vous permettra d’inclure les fichiers nécessaires.*

*- Pour la création de votre éventuelle DB, si elle ne se fait pas automatiquement lors du démarrage de votre API, vous devez offrir un script et le mettre au sein de votre projet. Dans ce cas, la procédure pour créer la DB doit être documenté au sein du README de votre projet.*

## Documentation de votre API

*Veuillez documenter les opérations de votre API, soit à l'aide de tableaux à donner ci-dessous, comme ceux vu dans le cours, soit en référençant la documentation qui aurait été générée à l'aide d'outils (Swagger par exemple). Votre fichier README de votre projet doit indiquer l'endroit où se situe la documentation de votre API.*

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

|  |  |  |  |
| --- | --- | --- | --- |
| URI | Méthode http | opération | Auths |
| /quiz | GET | READ ALL : lire toutes les ressources de type « quiz » vérifiés | Non |
| /quiz/id/ :id | GET | READ ONE : lire la ressource « quiz » vérifié | Non |
| /quiz/addQuiz | POST | CREATE ONE : créer la ressource « quiz » | JWT |
| /quiz/search | GET | SEARCH : recherche la ressource « quiz » vérifié | Non |
| /auths/register | POST | Créer une ressource de type "users" et renvoyer le username et un token JWT | Non |
| /auths/login | POST | Vérifier les credentials d'une ressource de type "users" et renvoyer le username et un token JWT si les credentials sont OK | Non |
| /admin | GET | READ ALL : lire toutes les ressources de type « quiz » non vérifiés | Non  JWT |
| /admin/all | GET | READ ALL : lire toutes les ressources de type « quiz » | JWT |
| /admin/remove/ :id | DELETE | DELETE ONE : Effacer la ressource identifiée | JWT |
| /admin/quiz/ :id | GET | READ ONE : lire la ressource « quiz » | JWT |
| /admin/validate / :id | POST | Valide la ressource « quiz » et met sa donnée « isVerified » à true | JWT |

*Veuillez aussi documenter les tests de votre API : les requêtes http doivent être données au sein de votre projet. REST Client devrait être utilisé, mais si vous préférez un autre client léger, vous devez vous mettre d’accord au sein de votre équipe de projet. Veuillez indiquer où se trouvent les requêtes HTTP si ça n’est pas dans le répertoire* ***/api/REST Client****.*

*Rappel : il ne peut pas y avoir d'opération de votre API sans au minimum une requête HTTP associée.*

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

## Déploiement de vos applications

*Veuillez 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 ».*

*Veuillez aussi indiquer ci-dessous deux URLs, comme par exemple https://e-vinci.github.io/wowapp :*

* URL de votre frontend déployé : https://github.com/e-vinci/web2-2022-project-group-13/tree/main/frontend
* URL de votre RESTful API déployée : https://github.com/e-vinci/web2-2022-project-group-13/tree/main/api

## Code réutilisé

*Il est important que vous citiez les parties de code que vous avez réutilisées, du code issu d’un tiers, au sein de votre code source. Pour ce faire, dans votre code source, utilisez des commentaires, en complétant au minimum les informations associées à l’auteur et à l’endroit où le code est disponible (URL). Voici le format que pourrait prendre votre commentaire:*

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
\*    Title: <title of program/source code>  
\*    Author: <author(s) names>  
\*    Date: <date>  
\*    Code version: <code version>  
\*    Availability: <where it's located, URL>  
\*  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

*Veuillez résumer tous les codes sources utilisés dans votre code, au sein de ce tableau :*

| **Chemin du fichier où se trouve le code réutilisé** | **Auteur du code source réutilisé** | **URL où le code réutilisé est disponible** | **Raison de la réutilisation du code** |
| --- | --- | --- | --- |
| /frontend/src/Components/QuizPage.js | Gabriel corti | https://codepen.io/borntofrappe/pen/LwZRON | Code pour avoir un timer + animation sur la quizPage |
| /frontend/src/Components/HomePage.js | Gabriel corti | https://codepen.io/borntofrappe/pen/LwZRON | Code pour avoir une animation sur la HomePage |
| /frontend/src/Components/QuizPage.js | In « w3schools » website | https://www.w3schools.com/js/js\_random.asp | Code pour randomiser les questions et les réponses d’un quiz |
| /frontend/src/Components/QuizPage.js | Tobiasahlin website | https://tobiasahlin.com/ | Code d’animation pour le titre du quiz |

# Analyse des résultats par le groupe

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

*Veuillez décrire si vous avez pu atteindre vos objectifs fonctionnels et autres tâches. Si vous avez dévié des objectifs fonctionnels identifiés initialement, veuillez décrire les raisons de ce changement.*

*Ce paragraphe est à rédiger sur environ 15 lignes.*

Une fonctionnalité que nous n’avons finalement pas ajouté sur le site, est le fait de pouvoir choisir la difficulté d’un seul quiz, et de créer un quiz avec différents niveaux de difficulté (la durée d’une question se réduisait si le niveau de difficulté augmentait et les questions devenait de plus en plus dur en fonction de la difficulté). Nous estimions que pour un utilisateur, la création d’un quiz ou l’expérience du quiz serait beaucoup trop longue (30 questions au total, 10 questions par difficulté), ce qui pourrait facilement l’ennuyer et finalement laisser la création ou abandonner le quiz. Nous avions décidé de faire finalement en sorte qu’un utilisateur puisse avoir une courte, et bonne expérience de notre application. Ainsi, rendu la création d’un quiz et la durée d’un quiz moins lourde. L’utilisateur ou un administrateur pourra ainsi créer, un quiz de 10 questions uniquement en choisissant son niveau de difficulté. Le niveau difficulté changera uniquement par rapport à la difficulté des questions et non du Timer. Nous avons limité le Timer à 10 secondes pour chaque question d’un quiz. Mise à part cela, nous avions pu réaliser nos différents objectifs fonctionnels et eu le résultat qu’on voulait avoir.

## Audit ergonomique de votre projet

*Veuillez 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.*

Les boutons principaux sont suffisamment gros pour une accessibilité facile au partie importante (quiz) et sont concentré vers le milieu de la page par rapport au menu qui sert à quitter les fonctionnalités principales du jeu.

L’interface du jeu est relativement épurée pour ne pas déconcentrer l’utilisateur et le fond est sur une couleur relativement neutre pour éviter de biaiser sa perception (perception de la couleur notamment qui peut induire sur ses choix). Les contrastes sont suffisants pour être clair.

L’utilisateur doit cependant déplacer significativement sa souris lors du choix des réponses et faire des actions inutiles (choix de réponse non modifiable mais obligation de cliquer sur suivant) mais cela est un choix actif de notre part. La raison est que notre concept d’application est orienté didactique. Ces temps de pause non nécessaire permettent à l’utilisateur d’avoir le contrôle et d’assimiler les bonnes réponses plutôt que faire un maximum de quiz en enchainant plus de questions rapidement.

Le bouton d’acceptation du RGPD n’est pas pré-coché et l’utilisateur à la possibilité de retirer son consentement.

## Difficultés techniques rencontrées

Nous avions rencontré plusieurs difficultés. Pour la HomePage, la fonctionnalité pour rechercher un quiz à eu beaucoup de problème, que ce soit en front-end ou back-end, tout d’abord pour le front-end, Quoc An Duong a eu du mal à créer un submit pour cette fonctionnalité, qui a finalement été avec un Event de type « keypress » et un « click ». la fonctionnalité a cessé de fonctionné lors de l’ajout du code de Ke Man, il a cherché pendant des heures à trouver le problème, le message d’erreur était un SyntaxError, hors, le problème se trouvait ailleurs. Il se trouvait du côté backend, sur les routes, une requête empêchait la fonctionnalité de fonctionner sans avoir le moindre détail. C’est en modifiant une certaine route, que la fonction recherche refonctionnait. Ke man et Quoc An ont eu aussi du mal à gérer l’animation du timer et aussi le timer pour la page quiz. En effet, lors de l’ajout de l’ajout de celui-ci, les boutons pour submit une réponse à une question ne fonctionnait plus, le problème a été résolu en ajoutant une nouvelle div pour l’animation de la bombe. Le timer a été fait étape par étape, ils ont d’abord créé une fonction afin de créer de compte à rebours de 10 secondes, puis ils l’ont implémenté sur la pageQuiz pour chaque question, ils ont tout d’abord eu comme idée de passer une page après les 10 secondes, mais finalement à la fin du compte à rebours, la réponse de la question est affiché après ce délais, et un boutton « next » est présent afin de passer à la question suivante. En cas de réponse trouver ou de mauvaise réponse, le timer s’arrête et la page indique si l’utilisateur a trouvé la bonne réponse ou non. Il peut ainsi aisément passer à la page suivante.

## Conseils pour appliquer cette technologie

*Veuillez résumer tout ce qui serait intéressant pour quelqu’un qui souhaiterait appliquer le même genre de technologie, sur base de ces questions :*   
*Qu’est-ce que vous auriez aimé savoir avant de démarrer ?*

*Des conseils pour éviter certains problèmes rencontrés ?*   
*Des liens vers les sites les plus intéressants ?*

*Etc.*

*Ce paragraphe est à rédiger sur environ 15 lignes.*

Nous pensions que les technologies que nous avions utilisés lors de notre projet n’a pas été réellement compliqués à assimiler. Effectuer des recherches sur internet avant de se lancer sur ce projet (et en général) peut être aussi une source pour ne pas avoir de doutes sur ce que vous allez faire et avant de poser une question à quelqu’un, effectuez des recherches par vous-même, internet est grand, puissant, et donc il y a casi toutes les réponses à vos questions mais aussi de comprendre l’anglais car la plupart des réponses sont en anglais. En tant cas que futur développeur, ou informaticien en général, nous vous conseillons le forum « Stackoverflow ». Casi 90% des réponses de questions que vous vous posez ou de problèmes que vous rencontrez se retrouve sur ce site. Nous vous conseillons aussi de rejoindre les serveurs Discord officiel de certaine technologie ou des serveurs discord de programmation en général. Ce sont en générales des serveurs où des personnes s’entraident, et où l’on peut poser tout type de questions. Il y a en général beaucoup d’informaticiens ayant déjà de l’expérience dans le milieu professionnel qui peuvent vous aider. Nous pensons ainsi, que l’une des qualités d’un informaticien est le fait d’être autonome. Le fait de savoir, apprendre une nouvelle technologie, d’arriver à s’informer seul, et effectuer ses recherches en anglais, est primordial. Cela évitera de déranger une personne pour rien.

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

Lors de notre projet, étant donné, que nous nous connaissions tous, l’équipe s’est montré très coopérative, Nous avions réparti les tâches, tout le monde a pu finalement finir ce qu’il devait réaliser. Nous nous sommes toujours fait confiance. Nous avions utilisé l’application Discord pour créer un groupe et ainsi pour pouvoir nous communiquer. Si une personne avait un problème sur son code, il pouvait toujours demander à une personne sur le groupe afin de pouvoir régler son souci. Nous n’avions aussi pas eu de problème technique. Chacun n’avait pas réellement de difficulté par rapport à la matière javascript, et donc, nous avions pu faire ce projet aisément et nous avions pu facilement résoudre les problèmes de conflits sur notre code. Le point clé de notre projet était surtout la communication et la répartition de nos tâches, cela a été un point primordial pour finir le code de notre projet à temps.

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

La gestion du temps a été un point négatif de notre projet. Nous pensions s’être pris trop tard pour le projet. Nous avions dû changer de projet quasiment à la dernière minute car nous avions eu beaucoup de mal à trouver une idée de projet qui conviendrait pour tout le monde, et cela s’est répercuté sur les choix technologiques à utiliser, le choix du design de notre application, nous pensions que nous aurions pu largement améliorer la qualité de nos wireframes et avoir une meilleure interface pour l’utilisateur, et sur la répartition des tâches, nous aurions pu avoir une meilleure répartition, car finalement, certaines tâches attribuées pour une seule personne par exemple, se sont finalement réalisées à 2. Nous tâcherons de décider une idée de projet plus tôt afin de ne plus être dans la même situation qu’on a été pour ce trimestre.