

# Ergonomie & développement d'une SPA animée

DON'T HACK ME !

<b>Auteur 1</b>	Wenlin CHEN
<b>Auteur 2</b>	Kilian Desmet
<b>Auteur 3</b>	Thomas Johnen
<b>Auteur 4</b>	Luca NICOLAS
<b>Auteur 5</b>	Lotfi MOUAYNA
<b>Date</b>	17.12.2023
<b>Référence</b>	WEB2-2023-PROJECT-GROUP-01
<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.....	4
1.2	JavaScript & Node.js : consignes techniques, timing et évaluations.....	6
1.3	Ergonomie : consignes techniques, timing et évaluations.....	11
2	Objectif du projet.....	11
3	Mind map du projet .....	13
4	Persona .....	13
5	Axiomes de Morville.....	13
6	Planning des tâches et cas d'utilisation .....	14
7	Besoins techniques.....	17
7.1	Système.....	17
7.2	Frontend .....	17
7.3	API .....	18
8	Choix technologiques.....	18
8.1	Frontend .....	18
8.2	RESTful API.....	19
8.3	Wireframe .....	19
9	Conception & Implémentation.....	19
9.1	Code repositories.....	19
9.2	Secrets éventuels pour vos API ou base de données.....	20
9.3	Documentation de votre API .....	20
9.4	Déploiement de vos applications .....	23
9.5	Code réutilisé.....	23
10	Analyse des résultats par le groupe .....	24
10.1	Évaluation du résultat par rapport au planning des tâches et des cas d'utilisation .....	24
10.2	Audit ergonomique de votre projet.....	24

10.3	Difficultés techniques rencontrées .....	25
10.4	Conseils pour appliquer cette technologie .....	25
10.5	Quels sont les points positifs à la manière dont s'est déroulée la collaboration au sein du groupe ? .....	26
10.6	Quels sont les points qui seraient à améliorer pour de futures collaborations ? .....	26
11	Analyses individuelles des résultats .....	26
12	Présentation vidéo .....	27
13	Revue de projets par les pairs .....	28

## 1 Consignes et évaluations


### 1.1 Consignes générales

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

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

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

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

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

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

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

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

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

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

#### 1.1.2.1 Création de l'équipe associée à un projet

Veillez identifier le membre qui créera votre équipe sur GitHub.

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

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

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

Un web repository aura été créé pour votre équipe.

#### 1.1.2.2 Joindre une équipe existante

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

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

Si vous le souhaitez, vous pouvez visualiser cette vidéo qui montre [comment Joindre un GitHub Classroom Group Assignment](#).

### 1.1.3 Projet

Vous allez créer une SPA mettant en œuvre :

- Des sujets et technologies qui vous tiennent à cœur ;
- Une RESTful API tournant sous Node.js & Express ;
- Un frontend animé ;
- Un frontend consommant votre RESTful API et éventuellement des APIs tierces ;

- Au moins une librairie JS non vue en cours pour le frontend (anime.js ou phaser.io sont autorisées) ainsi qu'une librairie non vue pour l'API.

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

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

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

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

Tâche	Compétences	Critères	Deadline	Pt	Consignes
Objectif du projet	C6) Documenter et présenter en vidéo le développement d'une SPA		<b>Séance 9 : 16/10 ou 17/10</b>		<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>
Planning des tâches et cas d'utilisation	C6)		<b>Séance 12 : 24/10 ou 27/10</b>		<p>Décrire le planning des tâches et cas d'utilisation selon les instructions données au §6.</p> <p>Présenter votre planning à un enseignant, afin qu'il puisse vous aider à bien prioriser les tâches.</p>
Indiquer l'URL de votre code repository	C6)		<b>Séance 12 : 24/10 ou 27/10</b>		<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>

Tâche	Compétences	Critères	Deadline	Pt	Consignes
Choix technologiques	C6)		Séance 15 : 13/11 ou 14/11		Compléter le §8.  Discuter de vos choix technologiques avec un enseignant.
Rapports individuels d'activités	C6)	Rapports de qualité <i>Indicateurs : formulation de qualité, analyse de qualité, respect des consignes</i>	12/11 19/11 26/11 3/12 10/12 17/12	1 solo	Des sessions individuelles de feedback sont organisées via TEAMMATES permettant à chacun de répondre à des questions dont les réponses sont confidentielles ou anonymisées au sein d'un groupe. Des e-mails seront envoyés vous invitant à compléter un formulaire hebdomadaire, à compléter pendant le WE. À partir de la 2 <sup>ème</sup> 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 <b>/report</b> de votre repo.

Tâche	Compétences	Critères	Deadline	Pt	Consignes
					Effacer toutes les consignes mises <i>en grisé</i> dans ce document avant de soumettre ce rapport sur Moodle.
Soumission de la vidéo	C6)	Vidéo de qualité <i>Indicateurs : présentation du projet de qualité, analyse de qualité, respect des consignes</i>	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 <i>Indicateurs : esthétique, fonctionnel, codage de qualité, respect des consignes, ambitieux &amp; original, utilisation d'une librairie pour l'IHM non vue en cours</i>	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...).
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 <i>Indicateurs : fonctionnel, codage de qualité, respect des consignes, ambitieux &amp; original, utilisation d'une librairie pour le service web non vue en cours</i>	17/12	5	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.



Tâche	Compétences	Critères	Deadline	Pt	Consignes
Déploiement tant de votre frontend que backend	C4) Déploiement d'applications web	Déploiement de la SPA sur le cloud <i>Indicateurs : fonctionnel, performances de chargement acceptables</i>	17/12	2	
Réaliser un minimum de 5 revues sur le site web	C7) Analyser le développement de SPA faites par des pairs	Revue de projets compréhensibles & constructives <i>Indicateur : présence d'un minimum de 5 revues</i>	Avant examen de 1 <sup>ère</sup> session	1 solo	Via <a href="https://e-vinci.github.io/web2/my-reviews-page">https://e-vinci.github.io/web2/my-reviews-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 non actifs risquent d'être écarté du projet, spécialement s'ils ne soumettent pas leurs rapports individuels.</p> <p>Les étudiants n'ayant pas réalisé au moins un use case significatif seront considérés inactifs.</p>

Tâche	Compétences	Critères	Deadline	Pt	Consignes
					Les étudiants n'ayant pas participé significativement au projet recevront d'office une lourde pénalité au niveau de leurs points, voire un 0/20.

### 1.3 Ergonomie : consignes techniques, timing et évaluations

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

Compétence	Tâches	Deadline	Points	Consigne
Reporting & présentation	Objectif du projet	17/10		Décrire l'objectif de votre projet au §2 de ce document.  Discuter de votre objectif avec un enseignant et assurer vous que cet objectif soit validé avant d'aller plus loin dans votre projet.
Conception	Définir la vision marketing	5/11	6	Décrire le Mind map du projet. Créer le persona de (s) l'utilisateur (s) ciblé (s) par le projet. Répondre aux axiomes de Morville.
Analyse d'applications web	Architecture UX	5/11	4	Construire les wireframes détaillés de votre application.
	Analyse des résultats et rapport associé	17/12	2	Auditez votre projet et vérifiez le respect des règles GDPR.
	Présentation vidéo	17/12	8	Présenter votre projet en intégrant l'expérience utilisateur.
	<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

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.*

*Veillez résumer l'objectif de votre projet sur <https://e-vinci.github.io/web2/project-page> dans le champ « Description ».*

*Veillez aussi donner un nom à votre projet que vous complétez sur <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 :



exemple de  
projets.docx

Le but de notre projet est de développer un site de gestionnaire de mot de passe. Ce site sera sécurisé et facile à utiliser, et permettra également d'avoir un top 10 des mots de passe les plus utilisés sur le net (nous n'irons bien évidemment pas chercher celui de nos utilisateurs) et donc n'étant pas sécurisé.

Le public cible sera les particuliers ainsi que les organisations. Le site permettra aux utilisateurs de générer, stocker et gérer leur mot de passe en les stockant à distance, et permettra aussi de connaître la force d'un mot de passe et les mots de passes identique utilisé sur plusieurs sites.

Les utilisateurs pourront créer des comptes individuels ou des comptes partagés pour les organisations. Cela nous intéresse car les sites de gestionnaire de mot de passe n'indiquent ni la force de nos mots de passe ni si des doublons s'y retrouvent, ce qui implique que dans le cas où l'un de nos mots de passe fuite il est beaucoup plus simple de se faire hacker d'autre de nos compte.

Nos objectifs sont donc :

- Créer un site de gestionnaire de mots de passe.
- Indiquer à l'utilisateur la force de ses mots de passe.
- Pouvoir générer un mot de passe fort.
- Pouvoir voir un tableau contenant les mots de passe les plus utilisés .

### 3 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.*

### 4 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.*

### 5 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*

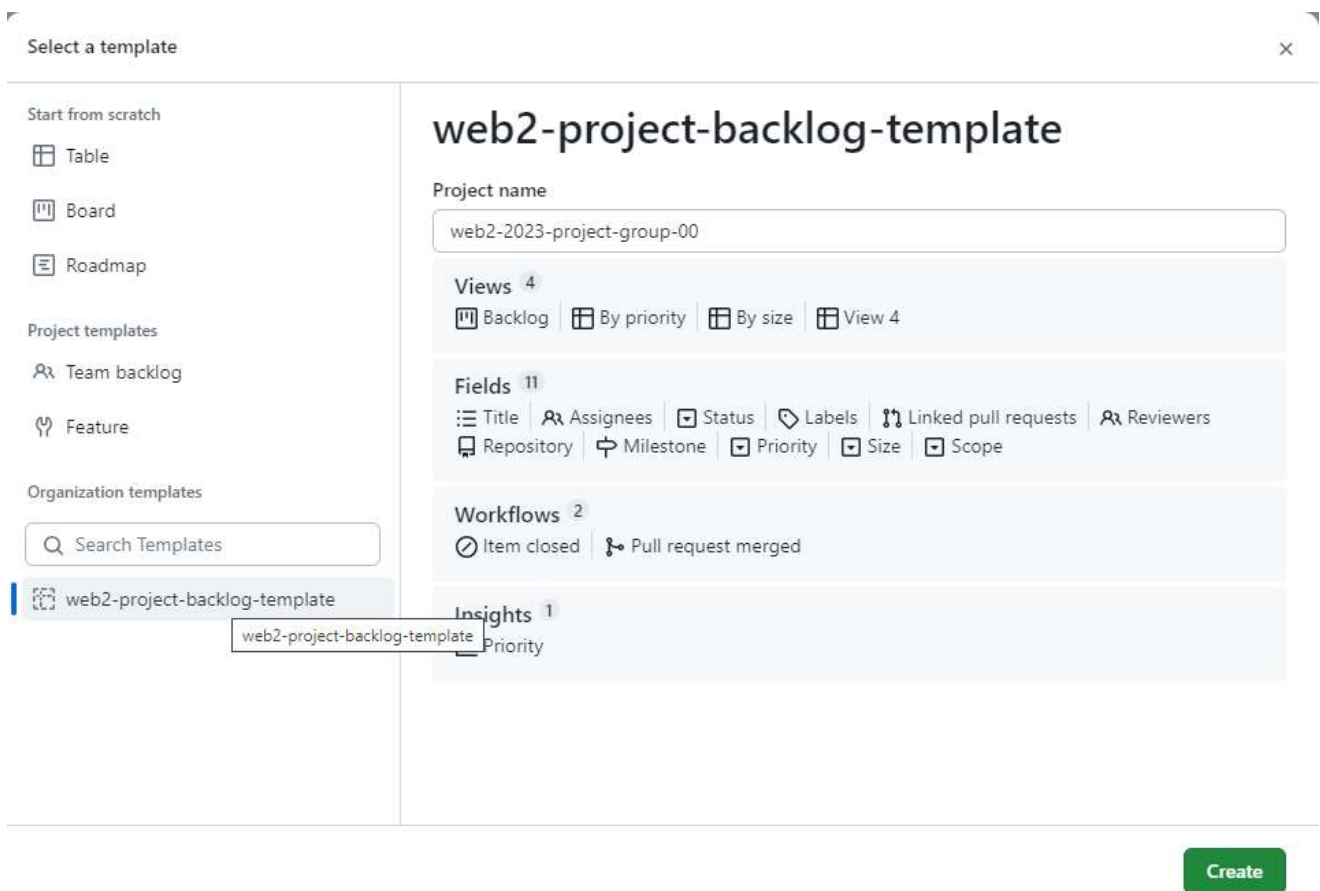
*réellement changer l'expérience utilisateur déjà existante pour ce type de projet ? Et si cela ne change pas, il faudra aussi le justifier. Ne pas recréer la roue n'est pas un défaut en soi, ça s'appelle de l'efficience.*

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

## 6 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**...

Veuillez regarder la vidéo qui suit pour mettre en place votre *GitHub Project*. Cette année, si vous souhaitez accélérer la création de votre *GitHub Project*, vous pouvez sélectionner comme *organization templates* : **web2-project-backlog-template** au lieu de **Team backlog** comme montré dans la vidéo.



Si vous préférez faire exactement comme dans la vidéo et créer manuellement toutes les tâches, vous utiliserez le template : **Team backlog**.

Voici la vidéo expliquant la mise en place de votre *GitHub Project* : [Partie 1 : Créer et configurer un GitHub Project](#).

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](#).  
NB : si vous avez accéléré la création de votre GitHub Project en sélectionnant le template **web2-project-backlog-template**, reste intéressante pour voir comment créer des drafts issues, convertir des draft issues en issues, ajouter des Labels aux issues...
- Chaque **tâche** doit être couverte par une **draft Issue** au sein de GitHub 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](#).
- 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 travaillez 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 (c'est les draft issues qui sont offertes dans le **web2-project-backlog-template**) :*

- 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 : <https://github.com/e-vinci/web2-2023-project-group-1>



## 7 Besoins techniques

*La spécification technique des besoins vous est donnée  
N'hésitez pas à la compléter s'il manque qqch d'important.*

### 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

*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.*

### 8.1 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>

- pour la 3D : <https://threejs.org>
- pour les jeux : <https://phaser.io>

## 8.2 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.

Nous avons choisis la librairie crypto-js avec le chiffrement AES-512 pour protéger les données sensibles, ce qui signifie que les mots de passe des utilisateurs sont stockés en toute sécurité et ne peuvent être déchiffrés que par eux-mêmes.

## 8.3 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.

# 9 Conception & Implémentation

## 9.1 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/web2-2023-project-group-1>

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*

## 9.2 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.*

## 9.3 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 référant 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/register	POST	REGISTER : Inscrire un utilisateur
auths/login	POST	LOGIN : Connecter un utilisateur
leaderboard/addLeaderboard	POST	CREATE/UPDATE ONE: Permet d'ajouter ou de modifier un mdp dans le leaderboard
leaderboard/get	GET	READ ALL : Lire tous les mdp du leaderboard
sitesPassword/addSite	POST	CREATE : Permet d'ajouter un site dans la base de données
sitesPassword/deleteSite	DELETE	DELETE ONE : Permet de supprimer le site de la base de données
sitesPassword/updateSite	PATCH	UPDATE ONE : Permet de modifier les données d'un site

sitesPassword/orderBySiteName	POST	READ ALL : Lire tous les sites triés par ordre alphabétique de noms de site d'un utilisateur
sitesPassword/orderByDate	POST	READ ALL : Lire tous les sites triés par ordre chronologique de création d'un utilisateur
sitesPassword/getSiteById	POST	READ ONE : Lire les informations d'un site d'un utilisateur

*Veillez 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. Veillez 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**

URL	Méthode http	Tests	Réponse	Paramètres
auths/login	POST	Connecter un utilisateur avec un nom inconnu	KO	username, password
auths/login	POST	Connecter l'utilisateur connu	OK	username, password
auths/register	POST	Inscrire un nouvel utilisateur	OK	login, email, password
auths/register	POST	Inscrire un utilisateur existant	KO	login, email, password
auths/readUserFromUsername	POST	Récupérer les informations d'un utilisateur avec son username	OK	username
auths/comparePassword	POST	Compare un mot de passe avec celui dans la base de donnée	OK	username, password

Auths/passwordCheck	POST	Compare son mot de passe avec celui de la base de donnée	OK	id, password
leaderboard/addLeaderBoard	POST	Ajouter un mdp dans le leaderboard	OK	password
leaderboard/leaderboard	GET	Afficher le leaderboard	OK	
sites/addSite	POST	Ajouter un site avec mdp	OK	userId, urlSite, siteName, usernameSite, passwordSite
sites/deleteSite	DELETE	Supprimer un site et ses informations d'un utilisateur	OK	userId, id
sites/updateSite	PATCH	Mets à jour les informations d'un site d'un utilisateur	OK	userId, login, url, site, id, password
sites/orderBySiteName	POST	Récupère par ordre alphabétique les noms de site d'un utilisateur	OK	username
sites/orderByDate	POST	Récupère par ordre chronologique de création de site d'un utilisateur	OK	username
sites/getSiteById	POST	Récupère toutes les informations d'un site d'un utilisateur	OK	userId, siteId

## 9.4 Déploiement de vos applications

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

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

- URL de votre frontend déployé : <https://iwezix.github.io/>
- URL de votre RESTful API déployée : donthackmeapi.azurewebsites.net

## 9.5 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>  
*  
******/
```

Veillez 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
e.g. /webApp/src/index.js	Dogan Erisen	<a href="https://github.com/Azure-Samples/active-directory-b2c-javascript-msal-singlepageapp">https://github.com/Azure-Samples/active-directory-b2c-javascript-msal-singlepageapp</a>	Code pour recevoir un access token via Azure AD

## 10 Analyse des résultats par le groupe

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

Au cours du développement de notre gestionnaire de mots de passe, notre équipe s'est efforcée de respecter le planning des tâches et des cas d'utilisation que nous avons initialement établi.

Nous avons su atteindre nos objectifs fonctionnels tels que la vérification de mots de passe avec un leaderboard associé pour les pires mots de passe du monde entier, et en y ajoutant les mots de passe ayant été effectué sur notre site.

Nous avons également pu mettre en place un générateur de mots de passe.

Quant à la page utilisateur, nous avons pu implémenter l'affichage, la modification, la suppression et l'ajout d'un site web, ainsi que qu'un filtre par nom alphabétique et un filtre par date d'ajout.

Sans oublier le cryptage et décryptage automatique des mots de passe de l'utilisateur pour ses sites. La seule fonctionnalité que nous n'avons pas pu mettre en place est le filtre par force de mots de passe en raison d'un manque de temps.

Dans l'ensemble, nous sommes satisfaits de notre progression et avons réussi à atteindre la plupart de nos objectifs fonctionnels.

### 10.2 **Audit ergonomique de votre projet**

Notre application, étant un gestionnaire de mots de passe, est adapté à toutes types de personnes sachant se servir d'un ordinateur.

Pour ce qui est des GDPR, notre application frontend ainsi que notre RESTful API respecte les différentes règles demandées par l'Europe. Nous avons équipé notre site d'un bouton menant vers des conditions d'utilisation, les données de nos utilisateurs sont stockées en toute sécurité sur les serveurs hébergeant le projet.

Les mots de passe stockés des sites de l'utilisateur, sont cryptés en arrivant sur la base de données depuis le frontend, pour empêcher toutes personnes de les récupérer.



### 10.3 Difficultés techniques rencontrées

Au début du projet, il nous a été compliqué de trouver un bon module de cryptage, ce qui nous a causé une perte considérable de temps sur plusieurs semaines.

Une autre difficulté que nous avons rencontrées durant le projet, il y a un package qui a été un peu plus compliqué que les autres. Ce fut « nodemailer », un package permettant d'envoyer un mail. L'implémentation du package en lui-même fut assez simple, mais nous avons rencontré un autre problème. Nous avons voulu créer une adresse Gmail pour notre site et nous sommes rendu compte que Gmail ne prenait plus en charge les applications tierces comme les API avec des comptes récents. Nous aurions donc dû, pour ce faire, acheter un nom de domaine. Nous avons donc décidé, pour le projet, d'utiliser l'une de nos adresses personnelles afin que tout soit fonctionnel.

Nous avons également rencontré des problèmes lors des animations selon la force du mot de passe, qui ne s'affichaient soit pas au bon endroit ou tout simplement pas.

Concernant le backend, nous avons rencontré quelques petites difficultés à certain moment mais rien n'ayant pu nous bloquer. Nos problèmes dans le backend ont souvent été résolus lors des cours de Javascript que nous avons continué d'avoir le long de notre projet ou simplement après une recherche sur StackOverflow ou encore sur ChatGPT.

À propos du déploiement de ce projet, ce fût une partie plus au moins compliquée surtout que nous ne nous étions absolument pas renseignés à ce sujet avant la dernière semaine. Le peu de temps restant en plus des petits problèmes visuels à régler nous a fait perdre un peu de temps sur le déploiement mais nous y sommes arrivés. Le plus compliqué a été de comprendre comment lier l'API avec le frontend, une fois ceci compris, le reste s'est dérouler sans problème.

### 10.4 Conseils pour appliquer cette technologie

Au commencement du projet, nous aurions dû nous dire que si nous n'arrivions pas à gérer un module de cryptographie, nous aurions dû directement voir les autres. Ce qui nous a bloqué pendant un long moment.

Un conseil que nous pouvons donner pour éviter certains problèmes rencontrés, c'est de bien lire la documentation et d'aller chercher sur stackOverFlow dès que nous rencontrons un problème ou encore ChatGPT. Les vidéos Youtube sont assez intéressantes aussi, ça nous a aidé pour le logo de notre site.

En général si vous avez un problème, vous pouvez trouver quelqu'un qui a déjà eu ce problème et quelqu'un a déjà répondu à ce problème. La communauté est souvent là pour aider les gens qui bloquent sur un problème ainsi que les professeurs.

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

La collaboration entre les membres de l'équipe a été très simple et il y a eu une bonne cohérence ainsi qu'une bonne communication et une bonne entraide au sein du groupe.

Lotfi et Thomas, qui ont déjà eu ce cours l'année dernière, n'hésitaient pas à nous aider et nous expliquer en faisant du pair-programming.

Luca qui se débrouillait très bien nous a souvent aidé à régler des problèmes durant le développement de notre code et arrivait toujours à nous motiver.

Kilian et Wenlin, se sont plus concentrés sur le design de notre site, et mis en place toutes les animations et ont donc aidé à respecter les fonctionnalités demandées.

Le fait de s'entraider nous a permis de ne pas rester coincé sur une tâche.

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

Dans de futures collaborations, une chose à améliorer, serait la planification des différentes tâches que le projet contiendra à sa version finale.

Une structure basique de l'application, une ébauche de ce à quoi ressemblera l'application. Une autre chose importante, c'est de d'abord essayer de créer une application simple pour un premier développement et de la complexifier après coup.

C'est-à-dire, à la place de créer quelque chose de compliqué, essayer plutôt de faire une application basique qui fonctionne et ensuite l'améliorer.

## 11 Analyses individuelles des résultats

*Les analyses individuelles sont à réaliser via les sessions de feedback qui seront à soumettre individuellement à la fin de chaque semaine de cours, pendant le WE, via l'outil TEAMMATES (<https://teammatesv4.appspot.com>).*

*Vous recevrez des e-mails vous invitant à compléter votre feedback hebdomadaire sur le projet. Attention aux pénalités si vous ne complétez pas votre feedback.*

*NB : Ce §11 peut être entièrement effacé du rapport que vous soumettrez sur Moodle.*

## 12 Présentation vidéo

Voici les exigences associées à votre présentation vidéo :

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

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

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

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

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

- <https://screencast-o-matic.com/> : logiciel pouvant être utilisé gratuitement sous réserve d'accepter un logo. Très facile d'utilisation.

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

De plus, veuillez indiquer ci-dessous ce lien :

Lien vers la vidéo youtube : <https://youtu.be/4-ChFg9kyZ8>

## 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.*