

Ergonomie & développement d'une SPA animée

INCLURE ICI LE NOM DE VOTRE PROJET

Auteur 1	Brahim ABID
Auteur 2	Youssef ABOUHAMID
Auteur 3	Soulaimane CHOUJAA
Auteur 4	Oussama EL BOUHTANI
Auteur 5	Matteo FIORE
Date	18.12.2022
Référence	WEB2-2022-PROJECT-GROUP-04
Version	1.0

Contents



1	Consignes et évaluations.....	3
1.1	Consignes générales	3
1.1.1	Création des groupes sur le site du cours	3
1.1.2	Création d'un groupe sur GitHub Classroom et du web repo associé	4
1.1.3	Projet.....	5
1.2	JavaScript & Node.js : consignes techniques, timing et évaluations.....	6
1.3	Ergonomie : consignes techniques, timing et évaluations.....	11
2	Objectif du projet.....	12
3	Mind map du projet	12
4	Persona	12
5	Axiomes de Morville.....	12
6	Planning des tâches et cas d'utilisation	13
7	Besoins techniques.....	15
7.1	Système.....	15
7.2	Frontend	15
7.3	API.....	16
8	Choix technologiques.....	17
8.1	Frontend	17
8.2	RESTful API.....	17
8.3	Wireframe	17
9	Conception & Implémentation.....	17
9.1	Code repositories.....	17
9.2	Secrets éventuels pour vos API ou base de données.....	18
9.3	Documentation de votre API	18
9.4	Déploiement de vos applications	19
9.5	Code réutilisé.....	19
10	Analyse des résultats par le groupe	20
10.1	Evaluation du résultat par rapport au planning des tâches et des cas d'utilisation	20
10.2	Audit ergonomique de votre projet.....	20

10.3	Difficultés techniques rencontrées.....	20
10.4	Conseils pour appliquer cette technologie	21
10.5	Quels sont les points positifs à la manière dont s'est déroulée la collaboration au sein du groupe ?	21
10.6	Quels sont les points qui seraient à améliorer pour de futures collaborations ?	21
11	Analyses individuelles des résultats.....	Erreur ! Signet non défini.
12	Présentation vidéo	22
13	Revue de projets par les pairs.....	Erreur ! Signet non défini.

1 Consignes et évaluations


1.1 Consignes générales

1.1.1 Création des groupes sur le site du cours

Veillez former un groupe de 4 ou 5 étudiants sur le site associé au cours : <https://e-vinci.github.io/web2>. Pour ce faire, veuillez-vous authentifier en cliquant sur l'icône . Rendez-vous sur l'onglet **Projets** (<https://e-vinci.github.io/web2/project-page>). Il est recommandé que l'attribution des **groupes** se fasse par **discussions** entre les **étudiants**. Lorsque 4 ou 5 étudiants ont **un intérêt commun** pour un **projet**, ils s'inscrivent au sein d'un groupe en cliquant sur l'icône .

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

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

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

A la date ultime de création de groupe (**23/10**), pour les étudiants toujours en recherche de partenaires, nous faciliterons (ou imposerons si nécessaire) la création des groupes, mais pas des sujets de projet.

Une fois tous les groupes de 4-5 étudiants remplis, il restera maximum 3 étudiants non liés à un projet. Si nécessaire un ou plusieurs groupes de 3 étudiants seront créés.

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

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

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

1.1.2.1 Création de l'équipe associée à un projet

Veillez identifier le membre qui créera votre équipe sur GitHub.

Ce membre accédera à l'assignement via : <https://classroom.github.com/a/7av06CzK>

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

e-vinci-web2

Accept the group assignment —
web2-2022-project

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

Create a new team

group-04

+ Create team

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

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

You're ready to go —
group-04

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

Your team's assignment repository has been created:

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

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

Note: You may receive an email invitation to join [e-vinci](#) on your behalf. No further action is necessary.

Un web repository a été créé pour votre équipe.

1.1.2.2 Joindre une équipe existante

Une fois l'équipe d'un projet créée, les autres membres accéderont aussi à l'assignement via : <https://classroom.github.com/a/7av06CzK>.

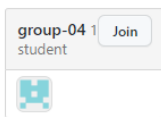
Ces membres joindront l'équipe existante en cliquant sur **Join** au sein de la bonne équipe. Par exemple, pour les membres du **Projet N°4**, ils cliqueront sur **Join** dans l'équipe **group-04**.

e-vinci-web2

Accept the group assignment —
web2-2022-project

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

Join an existing team



OR Create a new team

Create a new team + Create team

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

1.1.3 Projet

Vous allez créer une SPA mettant en œuvre :

- Des sujets et technologies qui vous tiennent à cœur ;
- Une RESTful API tournant sous Node.js & Express ;
- Un frontend animé ;
- Un frontend consommant votre RESTful API et éventuellement des APIs tierces ;
- Au moins une librairie JS non vue en cours pour le frontend (anime.js ou phaser.io sont autorisées) ainsi qu'une librairie non vue pour l'API.

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

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

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

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

Tâche	Compétences	Critères	Dead-line	Pt	Consignes
Objectif du projet	C7) Documenter et présenter en vidéo le développement d'une SPA		23/10		<p>Donnez un nom à votre projet et décrire l'objectif de votre projet au §0 de ce document ainsi que sur https://e-vinci.github.io/web2/project-page, complétez :</p> <ul style="list-style-type: none">- Le nom du projet : Projet N°X : Nom de votre projet- le champs « Description ». <p>Discuter de votre objectif avec un enseignant et assurer vous que cet objectif soit validé avant d'aller plus loin dans votre projet.</p>
Planning des tâches et cas d'utilisation	C7)		27/11		<p>Décrire le planning des tâches et cas d'utilisation selon les instructions données au §6.</p> <p>Présenter votre planning à un enseignant, afin qu'il puisse vous aider à bien prioriser les tâches.</p>
Indiquer l'URL de votre code repository	C7)		27/11		<p>Votre code doit être accessible par tout le monde via un web repository public qui vous sera</p>

Tâche	Compétences	Critères	Dead- line	Pt	Consignes
					<p>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.</p> <p>Plus d'information aux §1.1.2 et §9.1.</p>
Choix technologiques	C7)		04/12		<p>Compléter le §8.</p> <p>Discuter de vos choix technologiques avec un enseignant.</p>
Rapports individuels d'activités	C7)	Rapports de qualité <i>Indicateurs : formulation de qualité, analyse de qualité, respect des consignes</i>	18/12	1 solo	<p>Des sessions individuelles de feedback sont organisées via TEAMMATES permettant à chacun de répondre à des questions dont les réponses sont confidentielles ou anonymisées au sein d'un groupe. Des e-mails seront envoyés vous invitant à compléter un formulaire hebdomadaire, à compléter pendant le WE.</p> <p>Tout formulaire hebdomadaire non complété amènera à une pénalité individuelle de 0.5 point.</p>
Soumission du rapport de groupe	C7)	Idem	18/12	1	<p>Compléter le §10 ainsi que tous les paragraphes qui n'auraient pas été finalisés de ce document.</p> <p>Soumettre ce document, via Moodle (un devoir sera créé) ainsi que dans le répertoire /report de votre repo.</p>

Tâche	Compétences	Critères	Dead-line	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	C7)	Vidéo de qualité <i>Indicateurs : présentation du projet de qualité, analyse de qualité, respect des consignes</i>	18/12	2	Présenter votre projet selon les exigences du §11.
Soumission du code du frontend	C1) Créer une IHM interactive, moderne & esthétique Optionnel : C4) Intégrer l'authentification, l'autorisation et les sessions d'utilisateurs au sein d'une SPA	Qualité de l'IHM produite <i>Indicateurs : esthétique, fonctionnel, codage de qualité, respect des consignes, ambitieux & original</i>	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.
	C3) Créer une SPA intégrant une IHM & un web service	Qualité de l'intégration du service web à l'IHM <i>Indicateurs : fonctionnel, codage de qualité, respect des consignes</i>	18/12	2	NB : votre RESTful API doit être un minimum différente des APIs fournies dans les démos du cours de JS.
Soumission du code du backend	C2) Créer un service web de base Optionnel : C4) Intégrer	Qualité du web service produit <i>Indicateurs : fonctionnel, codage de qualité, respect des</i>	18/12	4	

Tâche	Compétences	Critères	Dead-line	Pt	Consignes
	l'authentification, l'autorisation et les sessions d'utilisateurs au sein d'une SPA	<i>consignes, ambitieux & original</i>			
Utilisation d'une librairie pour des animations ou un jeu Utilisation d'une librairie pour le service web	C6) Intégrer au développement d'une SPA des technologies non vues en cours	Intégration de librairies non vues en cours <i>Indicateurs : utilisation d'une librairie pour l'IHM, utilisation d'une librairie pour le service web</i>	18/12	2	
Déploiement tant de votre frontend que backend	C5) Déployer une SPA sur le cloud	Déploiement de la SPA sur le cloud <i>Indicateurs : fonctionnel, performances de chargement acceptables</i>	18/12	2	
Réaliser un minimum de 5 revues sur le site web	C8) Analyser le développement de SPA faites par des pairs	Revue de projets compréhensibles & constructives <i>Indicateur : présence d'un minimum de 5 revues</i>	Avant examen de 1 ^{ère} session	1 solo	Via https://e-vinci.github.io/web2/my-reviews-page , vous devez revoir les vidéos de présentation de 5 groupes (sauf le vôtre), exécuter leurs applications, et fournir votre critique de chacun de ces projets. Vous pourrez fournir la critique d'autant de projets que vous le souhaitez. Plus d'info sur la revue de projet au § Erreur ! Source du renvoi introuvable.
	TOTAL POINTS			20	Il est à noter que des membres d'un même groupe pourront être cotés différemment en fonction

Tâche	Compétences	Critères	Dead-line	Pt	Consignes
					<p>de leur engagement sur le projet. L'engagement d'un étudiant est visible via les rapports individuels d'activités (outil TEAMMATES) via GitHub (GitHub Project, Issues, Milestones, commits...) et lors des sessions de cours.</p> <p>Les étudiants n'ayant pas participés activement au projet recevront d'office une lourde pénalité au niveau de leurs points, voire un 0/20.</p> <p>Les étudiants n'ayant pas réalisé au moins un use case significatif seront considérés inactifs.</p>

1.3 Ergonomie : consignes techniques, timing et évaluations

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

Compétence	Tâches	Deadline	Points	Consigne
Reporting & présentation	Objectif du projet	23/10		Décrire l'objectif de votre projet au §0 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 cotés différemment en fonction de leur engagement sur le projet visible lors des sessions de cours.

2 Objectif du projet

Le but du projet était de créer une application mettant en relation des demandeurs d'emplois et des entreprises du monde de l'informatique. Elle permettrait aux demandeurs d'emplois s'inscrivant sur l'application, de pouvoir facilement trouver des offres qui leurs conviendraient aux mieux selon leurs critères et leurs compétences grâce à système de swipe dans lequel se trouveraient toutes les offres et à un bouton « j'aime » à côté de l'offre qui enverrait une petite notification aux entreprises, leur signalant les personnes intéressées par leur(s) offre(s). Une entreprise inscrite pourrait alors consulter le profil du chercheur d'emploi et y trouver ses informations générales ainsi que ses compétences. Si la personne en question intéresse l'entreprise, celle-ci peut alors décider de la garder à l'aide d'un bouton « garder » et créer ainsi un « match » le cas échéant. Sinon il suffit à l'entreprise de choisir « refuser », ou alors de ne rien faire pour la laisser en attente.

En tant qu'étudiant en informatique de gestion, le fait de pouvoir faciliter des personnes avec une application mettant en relation rapidement des chercheurs d'emplois et des entreprises seraient vraiment pratique.

3 Mind map du projet

Veillez 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.

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

4 Persona

Veillez 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.

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

5 Axiomes de Morville

Veillez 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.

Veillez 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**...

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 GitHub Project : [Partie1 : 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](#).
- 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 :

- 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**.*

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

- URL vers votre GitHub Project public : <https://github.com/e-vinci/web2-2022-project-group-04>

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

Pour le backend, nous avons mis en ligne notre base de données sur :

<https://www.elephantsql.com/>

Pour le frontend nous avons juste utilisé une librairie de swipe pour afficher les offres d'emploi :

<https://swiperjs.com/>

8.1 Frontend

Librairie utilisée : <https://swiperjs.com/>

8.2 RESTful API

Nous avons décidé d'utiliser une base de données pour stocker les données du site, pour cela nous avons utilisé elephantSQL. <https://www.elephantsql.com/>

8.3 Wireframe

Veillez 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.

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

9 Conception & Implémentation

9.1 Code repositories

Veillez 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.

Veillez 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-04>

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

Nous avons ajouté un fichier avec les secrets dans la soumission.

9.3 Documentation de votre API

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

URI	Méthode HTTP	Opération
compagnies	GET	READ ALL : Lire toutes les entreprises
compagnies/register	POST	CREATE ONE : Inscription d'une entreprise et crée un token de session
Compagnies/{id}	GET	READ ONE : Lire l'entreprise identifié
developers	GET	READ ALL : Lire tous les développeurs.
developers/{mail}	GET	READ ONE : Lire le développeur identifié par son mail
developers/login	POST	CREATE ONE : Crée un token de session
developers/registerDev	POST	CREATE ONE: Inscription d'un développeur
developers/profileDev/{id}	GET	READ ONE: Lire le profil d'un dev
developers/masteredLanguageDev/{id}	GET	READ ALL : Lire tous les langages maîtrisés par un développeur
developers/addLanguageProgramationToDev	POST	CREATE ONE : Ajout d'un langage aux langages maîtrisées
jobOffers	GET	READ ALL : Lire toutes les offres d'emplois
jobOffers/addToInterstedDev	POST	CREATE ONE : Crée un intérêt d'un développeur pour une offre
jobOffers/notInrestedDev	POST	DELETE : supprime l'intérêt d'un développeur pour une offre
jobOffers/allJobOfferFromCompany/{idCompany}	GET	READ ALL : Lire toutes les offres d'une entreprise identifié
jobOffers/allDevelopersInterstedOffer/{idOffer}	GET	READ ALL : Lire tous les développeurs intéressés par une offre identifié
jobOffers/create/{idCompany}	POST	CREATE ONE : Création d'une offre d'emploi pour une entreprise identifié
jobOffers/allTypeOffer	GET	READ ALL : Lire tous les types d'offres

jobOffers/likedOffers/{idCompany}	GET	READ ALL : lire toutes les offres likés par des développeurs d'un company identifié
JobsOffers /likeDev/{idDev}/{idOffer}	POST	UPDATE ONE : Like d'un développeur identifié par une entreprise
jobOffers/getMatchesDevAndCompnay/{idOffer}	GET	READ ALL : Lire les matches d'une offre identifié
jobOffers/getCompaniesMatchInfos/{idDev}	GET	READ ALL: Lire les matches qu'a eu un développeur identifié
jobOffers/getJobOffersMatchInfos/{idCompany}/{idDev}	GET	READ ALL : Lire les infos des offres d'emploi où il y'a un matche pour un développeur identifié
jobOffers/getLanguageRequired/{idOffer}	GET	READ ALL : Lire les langages requis par une offre d'emploi
jobOffers/getAllLanguages	GET	READ ALL: Lire les langages
jobOffers/addLanguageToOffer	POST	CREATE ONE: Crée un langage requis pour une offre

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

9.4 Déploiement de vos applications

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

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

- URL de votre frontend déployé : <https://youuss17abou.github.io/dev-job/>
- URL de votre RESTful API déployée : <https://api-devjob.azurewebsites.net/>

9.5 Code réutilisé

Nous n'avons réutiliser aucun code en dehors du cours de JS.

10 Analyse des résultats par le groupe

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

Nous avons réussi à sortir toutes les fonctionnalités principales du site, se connecter, s'inscrire, pouvoir swipe pour regarder les offres d'emplois, avoir un profil développeur et un profil d'entreprise, permettre aux entreprises de voir les intéressés pour ses offres et faire en sorte que les offres soient filtrées en fonction du profil du développeur. Nous avons aussi sécurisé les appels à l'API grâce au tokens.

Mais ce qui est sûr c'est que nous n'avons pas eu le temps de terminer totalement le site, il manque beaucoup de css par exemple et on aurait aimé avoir plus de temps pour bien finir ce projet.

Le site ne ressemble pas totalement aux wireframes mais est globalement identique.

Néanmoins nous sommes contents d'avoir pu développer les fonctionnalités que l'on voulait au départ.

10.2 Audit ergonomique de votre projet

Arrivé sur le site, on remarque première une grande image et ensuite un texte dessus assez descriptible mais pas entièrement et lorsque l'on glisse la souris dessus l'animation permet de découvrir ce qui s'y cache et le bouton s'inscrire maintenant se trouve tout à la fin de ce paragraphe avec une animation qui pousserait à cliquer dessus et ainsi permettre de s'inscrire. Au milieu du remplissage qui pourrait en intéresser quelques-uns, et tout à la fin les données de contacts afin pouvoir maximiser la prise de contact avec le site. La barre de menu se trouvant tout au-dessus avec à gauche un lien vers la page d'accueil et à droite un lien vers le login pour les utilisateurs déjà enregistrer (lecture en Z).

Quelques fonctionnalités possibles afin de ne pas décourager les utilisateurs et simplifier le processus décisionnel. Les formulaires d'inscriptions sont assez clairs et concis. Les offres sont affichées avec un système de swipe en affichant offre par offre afin que l'utilisateur soit concentré dessus et puis passez à la suivant facilement en swipant ou alors avec les touches gauche et droite de son clavier. Lorsque des actions doivent être explicitée cela se fait par message affichés sur la page et non en pop-up. La quasi-totalité des pages est identifiée par des titres lisibles et des informations utiles.

10.3 Difficultés techniques rencontrées

- Travailler avec une bd : les requêtes étaient assez compliquées à mettre en place au début.
- La gestion du temps
- L'utilisation d'une librairie pour le système de swipe

- Le déploiement nous a posé des problèmes avec les CORS
- Nous avons été trop gourmand dans notre projet, on n'a pas pu effectuer un travail fini à 100% mais cela nous a appris énormément de choses
- Nous n'avons pas eu le temps d'implémenter une messagerie intégrée au site
- Nous avons utilisé le mauvais boilerplate au début du projet pour le frontend ce qui nous a posé problème pour le déploiement.

10.4 Conseils pour appliquer cette technologie

Ce qui serait intéressant pour quelqu'un qui souhaiterait appliquer le même genre de technologie : Ce qu'on aurait aimé savoir avant de démarrer, c'est comment importer et utiliser des librairies externes à notre projet en respectant les manières modernes de coder car nous avons perdu beaucoup de temps à régler des problèmes concernant les librairies. Et il faut avoir un bon niveau en Vanilla JavaScript pour pouvoir faire des SPA. Il faut aussi avoir un bon niveau en base de données. Pour éviter les conflits sur git, nous vous conseillons de ne jamais travailler sur le même fichier pour ne pas perdre de temps à résoudre des conflits.

Les sites qui nous ont le plus aidé afin de comprendre et approfondir des choses vues au cours sont YouTube, Stack Overflow et les recherches sur google. Internet est une bonne ressource d'aide pour le développement en général. Mais attention à ne pas prendre n'importe quelle solution sur internet car certaines ne sont pas conformes à la manière moderne de coder.

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

Le groupe s'est globalement bien organisé, on a directement pensé à créer un discord avec différents salons textuels pour trier nos problèmes (backend, frontend, DB, etc..). On a aisément communiqué entre nous durant tout le projet. Quand quelqu'un rencontrait une difficulté, il y avait toujours une autre personne pour l'aider.

On a souvent pensé à faire relire notre code de chacun aux autres du groupe pour vérifier s'il était compréhensible, lisible et logique.

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

Le gros point à améliorer c'est la gestion du temps, avec les autres projets qui étaient au même moment que celui-ci on a eu du mal à savoir quand travailler sur ce projet.

Il faudrait faire un Scrum plus précis, le nôtre était quand même trop vague et n'était pas vraiment pris au sérieux.

S'améliorer sur l'utilisation des librairies, de Bootstrap et CSS. Notre site n'est pas aussi beau qu'on l'espérait mais on pense qu'avec plus de temps on pourrait faire quelque chose de bien plus ergonomique visuellement parlant.

11 Présentation vidéo

Voici les exigences associées à votre présentation vidéo :

- Elle doit viser une durée de 5 minutes, et ne peut pas dépasser 10 minutes.
- Elle doit être visible sous youtube par n'importe qui possédant son URL. Sa visibilité doit donc être en "Unlisted" ou "Public", mais pas « Private » !
- Elle se basera principalement :
 - 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

- <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.

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

De plus, veuillez indiquer ci-dessous ce lien :

Lien vers la vidéo youtube : <https://www.youtube.com/watch?v=LtVsCp5XNbc>