Projet 3

Currency converter API

Contrat de Professionnalisation Développeur Multimédia

L'École Multimédia

Sommaire

PRÉSENTATION

Vous venez d'être recruté comme développeur par Philippe - directeur d'une toute nouvelle startup nommée MoneyValue afin de développer une plateforme de conversion monétaire.

MoneyValue travaille dans le domaine de la finance.
L'objectif de ce service public et gratuit est d'acquérir des données sur les conversions les plus demandées.

L'objectif est de développer une API REST utilisable par des développeurs externes et qui permet de convertir des devises en d'autres devises.

Vous devez également développer l'administration de l'API qui permet de gérer les devises mises à disposition.

Contraintes

Vous travaillerez seul sur le projet.

Votre développement doit impérativement répondre à ces critères :

- L'API doit être développée avec le framework **Laravel** et la base de donnée **MySQL**
- L'administration doit être développée coté front avec le framework **Vue.js**
- Les produits doivent répondre aux demandes décrites dans le cahier des charges

Libertés

- Vous êtes libre d'utiliser les librairies tierces dont vous avez besoin
- Vous êtes libre d'utiliser les API et services tiers nécessaires
- Vous êtes libre d'ajouter des fonctionnalités EN PLUS de la demande client

CAHIER DES CHARGES

Le développement se divise en deux partie :

1. Une API REST public
2. Une administration privée permettant de gérer les données de l'API

Contrainte de développement

Le directeur technique a décidé que :

1. l'anglais sera toujours utilisé pour les champs et les variables
2. On utilisera le nommage en camelCase / PascalCase pour le nommage des fonctions et variables et nom des classes (gérées automatiquement par la CLI de Laravel)
3. les méthodes et propriétés doivent être toujours commentées
4. Vous utiliserez la technique du contrôleur de ressource de Laravel pour le CRUD de ressource.
5. Vous utiliserez le service de validation de Laravel pour la gestion des formulaires.
6. Les données seront récupérées/traitées dans le code à l'aide du composant *Eloquent* de Laravel.
7. Vous devez mettre en place un github pour versionner votre code
8. Vous devez mettre en place des *migrations* et des *seeders* pour la base de données

L'administration

L'administration doit permettre à l'administrateur de MoneyValue de :

1. S'identifier et accéder à l'administration privée
2. Visualiser la liste des pairs supportées
3. Ajouter, modifier ou supprimer une paire de conversion
4. Visualiser le nombre de requêtes effectuées pour chaque pair

L’API

L’API est de type REST et permet à un développeur tierce de convertir des devises en d’autres devises.

Le endpoint de l’API devra être : /api

Le développeur externe doit pouvoir :

1. Savoir si le service est fonctionnel
2. Récupérer la liste des pairs de conversion supportées
3. Convertir une quantité de devise suivant une paire existante

Les données

Les données à gérer doivent suivre les contraintes délivrées par Philippe :

Devises

Chaque devise est identifiée par 3 lettres
exemple : USD, EUR, BTC

Paires

Chaque paire possède un taux de conversion fixe entre deux devises
exemple : EUR -> USD : 1.1

Les conversions doivent pouvoir se faire dans les deux sens
exemple : EUR -> USD : 1.1, USD -> EUR : 0.90

Conversion

Le décompte du nombre de conversion est attaché à chaque paire
exemple : EUR -> USD : 53

LIVRABLES

Votre rendu final devra comporter les éléments suivants :

1. Votre cahier des charges complet (voir Annexe) - [Bloc RNCP 1]
2. Un dossier /admin comportant l'administration (front-end Vue) [Bloc RNCP 3]
3. Un dossier /api comportant l'api REST (back-end Laravel)[Bloc RNCP 4]

Vous devez livrer une archive de votre livrable avec l'ensemble des éléments, à l'exception des dossiers suivant :

- vendor
- node_modules

Cette archive aura comme titre votre nom et votre prénom.

Consignes

Vous devez également mettre en place des *seeders* adaptés pour pouvoir réaliser une démo auprès du client.

NOTATION

Critères

- La compréhension de la problématique client
- La qualité rédactionnelle du cahier des charges
- Vos choix technologiques et votre capacité à innover
- La qualité de l'administration et de son interface
- La qualité de l'architecture de l'API et des données
- La conformité de la réalisation au brief

Conseils

- Etablissez votre git dé le début, et faites des *commits* réguliers pour chaque fonctionnalité ou correction de bug
- Organisez-vous et planifiez votre travail : donnez vous des objectifs intermédiaires
- Bien prendre le temps d'analyser le cahier des charges
- Commencez toujours par la structure de vos données
- Ne jamais être trop ambitieux
- Mettez en oeuvre les bonnes pratiques vues en cours
- Refactoriser pour éviter le code redondant
- Soignez la qualité de votre code (commentaires, indentation)
- Pensez à la facilité d'utilisation et la qualité du résultat !

ANNEXE

Vous trouverez ci dessous un exemple de cahier des charges

Analyse client

Reformuler ici avec vos mots votre compréhension de moneyValue - votre client - et des implications techniques que cela engendre.

Choix technologiques

Motivez vos choix technologiques, aussi bien côté administration (front) que du côté de l'API (back)

Evaluation du temps de travail

Faites un petit tableau avec comme entrée chaque grand poste de développement, et comme sortie le nombre de jour de travail (1 journée = 7 heures)

Liste fonctionnelle

Listez de manière exhaustive les fonctionnalités à développer (administration et API)

Recettage

Faites un tableau avec comme entrée la liste fonctionnelle, et comme sortie si la fonctionnalité est opérationnelle ou non opérationnelle (ou possède un bug, une limitation …)

Diagramme de la base de données

Insérez ici un diagramme des tables MySQL (réalisé sur feuille de papier ou à l'aide d'un logiciel dédié, tel que mySql Worksbench)
Dans ce but expliciter les relations dans votre schéma : représentez les à l'aide de "flèches" normalisées entre les tables ou entités.

Documentation de l’API

Rédigez de manière succincte :

1. les différentes URL de l’API REST (avec le verbe HTTP)
2. Leur fonctionnalité
3. Les données retournées
4. Les message d’erreur possible

Adresse Github

L’adresse Github du projet (administration et API)

Wireframe de la partie front de l’administration

Les images des wireframes de la partie administration