

Table des matières

[Introduction 3](#_Toc536054075)

[Reformulation du besoin 3](#_Toc536054076)

[Devis 3](#_Toc536054077)

[Justification des choix techniques : 5](#_Toc536054078)

[Pourquoi Bootstrap ? 5](#_Toc536054079)

[Pourquoi SASS ? 5](#_Toc536054080)

[Pourquoi Express.JS ? 6](#_Toc536054081)

[Pourquoi Laravel ? 6](#_Toc536054082)

[MCD : 7](#_Toc536054083)

[Maquette graphique : 7](#_Toc536054084)

[Répartition des tâches : 8](#_Toc536054085)

[Planning des tâches : 8](#_Toc536054086)

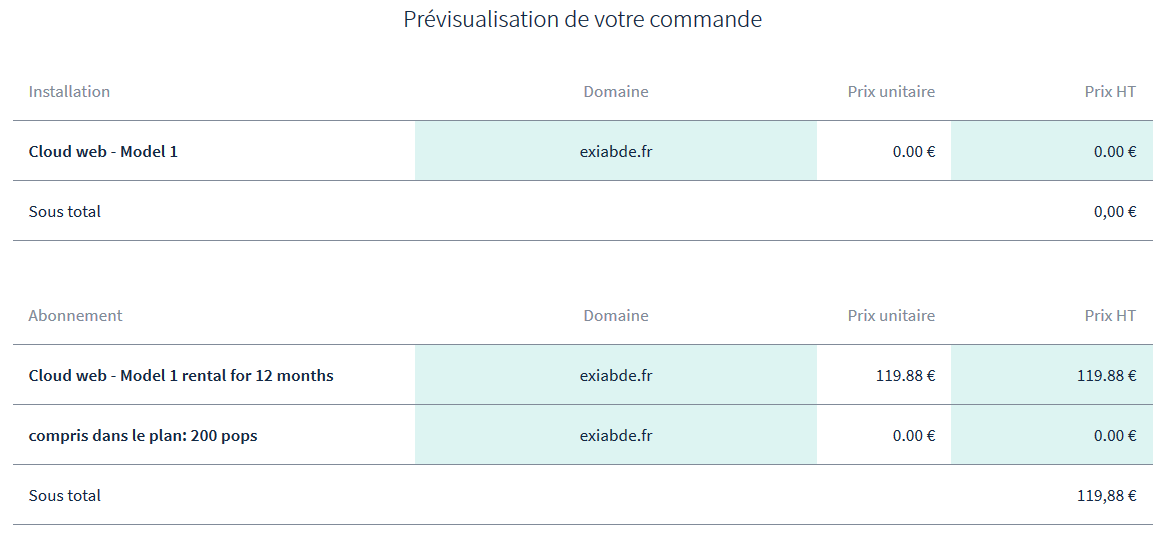
# Introduction

Pour cette mission, il nous a été demandé de créer le site internet du bureau des étudiants de l’Exia.Cesi, pour cela il nous a fallut utiliser différentes technologies récentes pour la plupart afin d’aboutir sur une architecture professionnelle et ergonomique facilement utilisable par n’importe quelle personne dépourvue de savoir dans le domaine, mais aussi manipulable par les membres de ce bureau des étudiants qui joueront le rôle d’administrateurs.

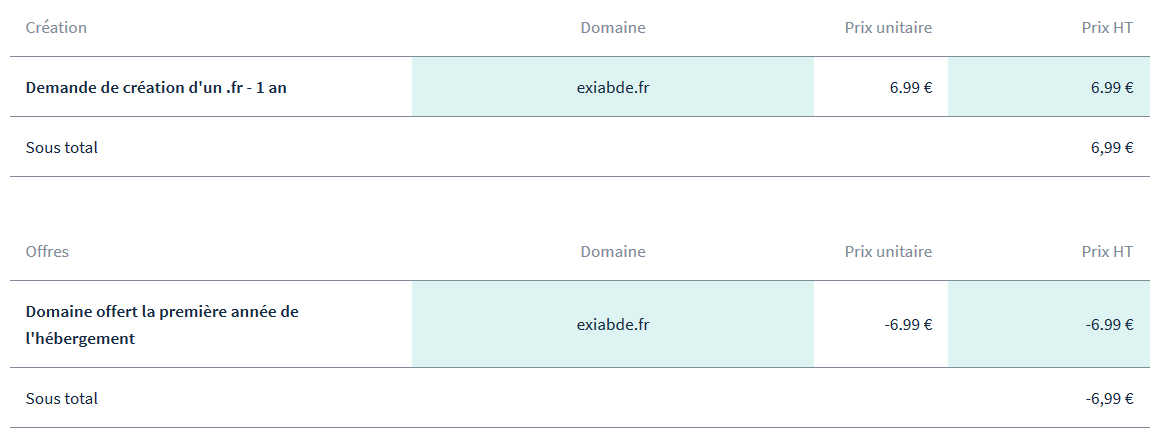
# Reformulation du besoin

Nous aurons donc besoin d’un site web responsive, à l’entrée du site nous aurons un site vitrine qui exposera les principaux contenus proposés par le site internet, tout visiteur aura accès au contenu mais nous pourra pas interagir avec, un étudiant pourra quant à lui se connecter, naviguer sur le site mais aussi proposer des idées, contribuer à ce qu’elles soient prises en compte mais aussi réagir aux précédents évènements et participer aux nouveaux, un salarié du CESI pourra signaler le contenu et les membres du bureau des étudiants administrer le site internet à l’aide d’outils permettent de supprimer certains contenus signalés ou indésirables.

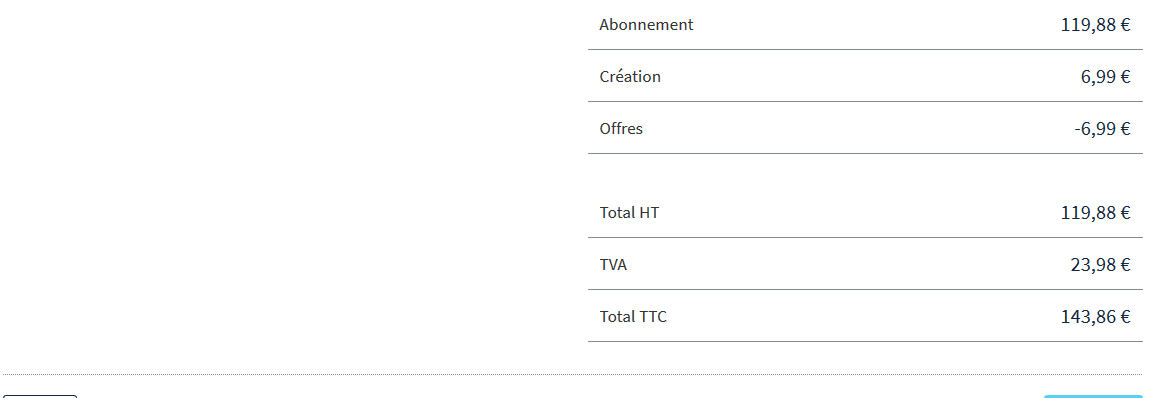
# Devis



Hébergement

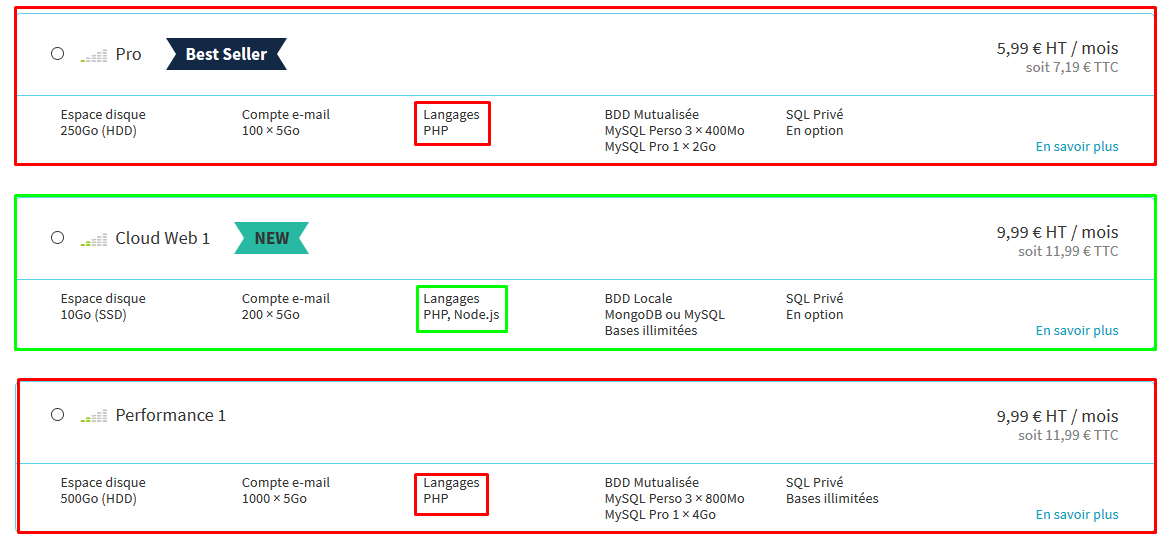


Nom de domaine



Total

Il a fallu pour cela choisir un serveur qui sera compatible avec PHP et Node.JS vous que nous avions mis en place deux serveurs :



Compatibilité Node.JS

# Justification des choix techniques :

## Image associéePourquoi Bootstrap ?

Et bien, car c’est un Framework c’est-à-dire orchestre de composants mis à disposition, qui sert à créer les fondations ainsi que les grandes lignes de tout ou d’une partie l’architecture d’un site internet, logiciel ou application. Par exemple, nous devons faire du routage pour un site, nous prenons un composant déjà tout prêt et qui a fait ses preuves et nous l'utilisons : gain de temps et fiabilité. L'utilité d'un framework est d'éviter de passer du temps à développer ce qui a déjà été fait par d'autres, souvent plus compétents.

Avec les avancées technologiques et les différents types d’appareils disponibles sur le marché, nous avons été obligés d’adapter nos interfaces web aux différentes tailles afin de correspondre à ces derniers, c’est possible de faire cela avec du simple CSS (Media Queries) mais cela prendrait beaucoup plus de temps, c’est là qu’intervient Bootstrap, a grille **Bootstrap** comporte par défaut 12 colonnes. Cependant, il faut savoir qu’elle est flexible et extensible dans la mesure où l’on peut réduire le nombre de colonnes ou en insérer de nouvelles dans chaque colonne existante et cela juste en modifiant certaines class de notre HTML.

Le système **Bootstrap** possède quatre tailles de grilles prédéfinies pour que la page se module suivant la taille de l’écran selon deux hypothèses : soit les éléments se redimensionnent en restant positionnés, soit ils s'empilent quand la fenêtre devient plus étroite et se positionne côte à côte quand elle s'élargit.

## Résultat de recherche d'images pour "Sass"Pourquoi SASS ?

Comme vous le savez, SASS est un préprocesseur CSS c’est-à-dire qu’il permet de générer dynamiquement des fichiers CSS et a pour objectif d'améliorer l'écriture de ces fichiers, en apportant plus de flexibilité.

Cet outil nous apporte beaucoup à notre manière d’écrire du CSS en nous facilitant la stylisation à l’aide de ces différentes fonctionnalités je cite :

**Les variables**  
Cela nous servira plus quand nous voudront respecter la charte graphique en les stockant dans nos variables, ce qui facilitera la tâche quand nous voudrions les appeler, pareil pour les polices d’écriture.

**Les mixins**  
Les mixins sont des fragments de CSS qui peuvent s’utiliser comme des fonctions.

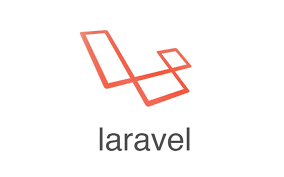
**Une facilité d’écriture, de lecture, et un code optimisé**  
L’écriture avec sass est mieux organisée, plus lisible, moins répétitive. Ce qui facilitera la lecture de notre code.

## Pourquoi Express.JS ?



Pour la configuration de notre serveur Node.JS, nous avons utilisé le framework Express. Le choix vis-à-vis de ce framework se justifie par plusieurs raisons : Le plus important est la simplicité de la syntaxe. N'étant pas encore habitués à travail coté-serveur sur Node, il nous fallait un framework flexible et simple d'utilisation. Les méthodes de réponses aux requetes HTTP telles que Get et Post sont assez faciles à réaliser. Le lien avec la base de données se fait assez facilement par la suite car on obtient facilement les chaines de caractères via les méthodes citées précedemment.

## Pourquoi Laravel ?



**Une documentation de qualité en cas de besoin**

Chaque développeur web utilisant Laravel a accès à une documentation particulièrement facile à comprendre, soignée et beaucoup plus organisée.

Laravel adopte le modèle contrôleur-vue-modèle . Comme il s’agit d’un cadre conforme à la norme MVC, il est livré avec de nombreuses fonctions intégrées et une architecture de développement robuste et améliorée.

**Mise à jour de la base de données**

La mise à jour de base de données est un processus automatisé. Les tables de base de données peuvent être mises à jour avec des données par défaut qui peuvent être utilisées pour la configuration préliminaire de l’application ou pour le test de l’application.

**Authentification et autorisation par hierarchie**

Il s’agit d’un framework PHP qui rend l’authentification des applications web très simple car tout est déjà intégré ou configuré dans le framework. Alors que la principale préoccupation de la plupart des propriétaires d’applications web concerne la vérification des utilisateurs et empêche l’accès des utilisateurs non autorisés, Laravel les soulage de cette tension. Il est relativement facile pour les propriétaires d’organiser la logique d’autorisation de leur application web et de restreindre l’accès des utilisateurs à toutes les ressources.

**Mise en cache**

Nous pouvons profiter du système de mise en cache de base fourni par Laravel pour réaliser des tâches de mise en cache simples comme le stockage d’objets dans un fichier ou une base de données. Simultanément, vous avez également le choix d’intégrer le framework web sans faille avec les systèmes de cache populaires tels que Redis, APC et Memcached. Ainsi, vous pouvez simplement augmenter les performances de l’application web en intégrant un système de mise en cache robuste dans Laravel.

**Artisan**

Laravel propose un outil intégré appelé Artisan, qui permet à l’utilisateur d’effectuer très rapidement de longues tâches de programmation qui peuvent être facilement réalisées par les développeurs de Laravel. Dans Laravel, le développeur doit interagir en utilisant une ligne de commande qui gère l’environnement du projet Laravel. Il est utilisé pour générer un code structuré et une structure de base de données pour faciliter la gestion du système de base de données.

**Moteur de templating Blade**

Laravel est livré avec le moteur de template Blade. Nous pouvons profiter de Blade pour inclure du code PHP simple dans la vue et compiler les vues en code PHP. Ainsi, nous pouvons donc améliorer les performances du site web en compilant les vues en code PHP, et mettre en cache le code jusqu’à ce que les vues changent. Le moteur de template lame vous permet en outre d’afficher plus facilement les données et d’étendre la mise en page sans affecter la vitesse de l’application.

**Système de routage**

Le système de routage est la caractéristique clé de Laravel qui est une méthode de routage facile à utiliser. Le chemin peut être défini sur l’application avec une couche de contrôle et une bonne flexibilité. Un répertoire est créé pour correspondre à l’URI.

# MCD :

En pièce jointe au dossier du projet.

# Maquette graphique :

En pièce jointe au dossier du projet.

# Répartition des tâches :

|  |  |
| --- | --- |
| **Membre** | **Tâches** |
| ECHAIB Walim | Front-end (HTML, CSS, JS) |
| ZIDI Mouhamed | Backend (PHP, Laravel) |
| LAHLALI Samy | Serveur Node.JS et API |
| MESBAH Zakaria | Base de données et requêtes (Modélisation, SQL) |

# Une image contenant capture d’écran Description générée automatiquementWBS :