ECOLE PRATIQUE DES HAUTES ETUDES COMMERCIALES

EPHEC

Avenue du Ciseau, 15
1348 Louvain-la-Neuve

Plateforme web et application mobile pour aider les jeunes et adolescents à réaliser des jobs pour des particuliers

Travail de fin d'études présenté en vue de l'obtention du diplôme de bachelier en Informatique et Systèmes orientation Technologie de l'Informatique

Thibaut Hermant

Rapporteur : Youcef Bouterfa

Année Académique 2019 – 2020

Préface

Suite à la crise sanitaire du Covid-19, certains points et difficultés rencontrées doivent être précisés.

Tout d'abord, j'ai fait le choix de développer une application mobile disponible pour IOS et Android. Néanmoins, pour permettre à mon application d'être installable sur les appareils IOS, je devais la publier sur la plateforme Apple Store. Cependant, pour effectuer cette démarche, il faut obligatoirement avoir à disposition un iMac ou tout autre ordinateur de la marque Apple. A cause de cette crise, je n'ai pas pu me rendre dans un endroit me permettant d'en avoir un à ma disposition. L'application n'a donc pas pu être déployée sur IOS pour le moment.

J'ai pu par contre la déployer sur la plateforme Play Store permettant aux appareils Android de la télécharger. Cependant, suite à cette crise des retards ont été pris au niveau de la vérification des applications par Play Store. Il est donc possible que lors de la présentation finale, l'application ne soit pas encore disponible sur cette plateforme.

Avant-propos

Je tiens à remercier toutes les personnes qui ont contribué à la réalisation de ce TFE.
Je tiens tout d'abord, à remercier Youcef Bouterfa, mon rapporteur. Il m'a supervisé, aidé et conseillé tout au long de ce TFE.
Je remercie également l'EPHEC qui m'a donné l'opportunité de pouvoir réaliser ce travail très enrichissant.
Enfin, je tiens aussi à remercier Martin Hermant et Camille Dardenne pour la relecture de ce rapport.

Table des matières

Introduction

Problématique

J’ai remarqué que de nombreux jeunes et adolescents souhaitent gagner de l'argent de poche en réalisant des petits travaux tels que du baby-sitting, de la garde d’animaux, du jardinage, …
Mais aussi que certains adultes n’ont pas toujours le temps ou sont simplement trop âgés pour réaliser ce genre de travaux.
Il existe donc une réelle demande pour les deux types de personnes comportant des avantages pour chacun.
C’est en voyant de nombreux messages sur les réseaux sociaux de jeunes proposant leurs services pour de petits jobs, que j’ai remarqué cette problématique.

Solutions existantes

Partant de ce constat, je me suis renseigné sur les solutions existantes.
Diverses solutions existent déjà sous différentes formes.

Certaines d’entre elles proposent des plateformes permettant de mettre en lien des employeurs et des jeunes pour des jobs d’étudiants à plus long terme avec un contrat. C’est par exemple le cas des plateformes Indeed, Student, Randstad, …

D’autres plateformes proposent des petits boulots entre particuliers sans contrat comme ListMinut ou Youpijob. Cependant, ce genre de plateforme autorise n’importe qui à proposer ses services. Cela ne privilégie pas le travail pour les jeunes et ceux-ci se sentent alors moins concernés.

Solution proposée

Pour ce faire, j’ai imaginé une plateforme permettant aux jeunes entre 15 et 25 ans de proposer leurs services à domicile pour des particuliers, et ce, dans différents domaines.
Je me suis alors renseigné sur les différents services dont avait besoin ma plateforme, pour satisfaire aux mieux tant les particuliers que les travailleurs.

En observant les solutions déjà existantes, j'ai vu que beaucoup d'entre elles faisaient ajouter les travaux par les particuliers, mais les rendaient visibles aux travailleurs seulement selon certains critères. Je me suis donc basé sur ce principe.

Les meilleurs critères de sélection des travaux, étaient selon moi :

- La distance entre le domicile du travailleur et le travail,
- Les compétences du travailleur,
- La disponibilité du travailleur,
- L'âge minimum du travailleur souhaité par le particulier.

J'ai donc fait en sorte de recueillir toutes ces informations auprès du travailleur, pour pouvoir lui proposer les travaux les mieux adaptés.

Il était également primordial pour les travailleurs et les demandeurs de pouvoir échanger par message ou par téléphone, s'il devait y avoir des informations complémentaires, des retards, des changements concernant un travail ...
J'ai donc décidé de mettre en place un système de messagerie créant une conversation entre le travailleur et le demandeur.

Toujours en me renseignant sur ce qui existe déjà en plateforme de petites prestations, j'ai remarqué que l'évaluation des prestataires était une chose très courante et appréciée. J'ai donc mis en place un système de notation du travailleur par le demandeur. A la fin de chaque travail, il sera demandé à la personne ayant reçu un service de noter la prestation du travailleur avec des étoiles de un à cinq. Grâce à cette évaluation, chaque travailleur obtient une moyenne des étoiles reçues.

Je me suis également dit qu'une personne souhaiterait peut-être refuser un travailleur, dans le cas où ce dernier aurait reçu trop de mauvaises évaluations et par conséquent posséderait une note assez basse. Ou encore, parce que cette personne aurait déjà travaillé avec lui et ne souhaiterait pas réitérer.
Pour ce faire, j'ai ajouté la possibilité à un demandeur de refuser un travail, ce dernier ne pourra alors plus l'effectuer.

Un autre fonctionnalité que j'ai imaginée, est l'affichage de statistiques. Différents graphiques sont visibles sur cette plateforme, permettant à l'utilisateur d'avoir une vue plus globale sur son activité. Je pense aussi que lorsqu'un travailleur va voir ses statistiques, cela va le pousser à toujours les améliorer et donc à utiliser plus régulièrement la plateforme.

J'ai fait le choix de développer une application mobile et une plateforme web pour permettre à un maximum de personnes d'utiliser cette solution. Cependant,

certaines personnes adultes sont parfois moins habituées à l'utilisation d'une application mobile et préfèreront peut être plus la plateforme web contrairement aux jeunes.
L'avantage d'une application mobile est d'avoir toutes les informations pour un travail à portée de main, directement sur son téléphone.

Cette solution est donc différente des autres déjà existantes, car elle propose des petits travaux à domicile tout en favorisant le travail des jeunes et des adolescents.

Comptes

Il y a deux types de comptes possibles: d'une part, les travailleurs réalisant les travaux. D'autre part, les demandeurs (personnes adultes) proposant des travaux aux jeunes.

Travailleur

Un travailleur a entre quinze et vingt-cinq ans. Il s'agit donc d'un adolescent ou d'un jeune adulte.
En s'inscrivant, un travailleur est amené à donner une série d'informations à son sujet.
Tout d'abord, des informations personnelles : nom, prénom, email, âge, numéro de téléphone, date de naissance et adresse complète, ainsi qu'un nom d'utilisateur unique et un mot de passe.
Ensuite, ce dernier doit donner une distance maximale de travail autour de son domicile. Cette distance doit être comprise entre un et cinquante kilomètres.
Après, le jeune travailleur indique également parmi les différents types de travaux possibles, ceux qu'il souhaite effectuer.
Enfin, Il doit entrer ses disponibilités selon les jours de la semaine.

Demandeur

Un demandeur est une personne ayant besoin d'un travailleur pour effectuer un travail.
Lors de son inscription, un demandeur devra indiquer ses données personnelles c'est à dire son nom, prénom, email, age, numéro de téléphone, date de naissance et adresse complète.
Il devra également choisir un nom d'utilisateur unique et un mot de passe.
Il existe neuf types de travaux possibles qu'un demandeur peut proposer:
Baby-Sitting, Travaux ménagers, Jardinage, Garde d'animaux, Bricolage, Faire des courses, Cours particuliers, Technologie, Autres.

Base de données

Choix du modèle

Il existe quatre modèles de bases de données : hiérarchique, réseau, relationnel et objet. J'ai choisi le modèle relationnel pour ma base de données. C'est un modèle qui organise les données en tables. Ce type de modèle me permet de maintenir et de faire évoluer facilement ma base de données.

Choix du langage

Au niveau du langage de ma base de données, j'utilise SQL. Ce langage est le plus répandu dans les bases de données relationnelles. Il permet d'effectuer des jointures, sélections, intersections, … entre les tables. Le SQL permet également de réaliser des modifications, ajouts et suppressions via de courtes requêtes.

Choix du SGBD

En langage SQL, il existe de nombreux systèmes de gestion de base de données (SGBD). Certains sont des systèmes propriétaires tels que Oracle Database, Microsoft SQL Server ou encore Sybase. D'autres sont libres tels que MySQL, MariaDB, PostgreSQL, ...
Mon choix s'est porté ici sur le système libre MySQL et cela pour diverses raisons. Tout d'abord, l'hébergement que j'ai choisi chez LWS offre une base de données MySQL5. Ensuite, il est facilement utilisable par la plupart des framework web. Et surtout parce que MySQL est l'un des SGBD les plus performants et rapides surtout en lecture de données.

Diagramme (annexe)

J'ai réalisé deux schémas de ma base de données, un schéma relationnel et un schéma entité-association. Le schéma relationnel me permet de voir ma base de données telle qu'elle existe sur le serveur. Le schéma entité-association permet de mieux comprendre les liens entre les tables. Ce dernier est composé d'actions faisant le lien entre les tables.

Explications

Ma base de données est composée de onze tables:

- account
- requester
- worker
- work
- worker_type_work
- type_work
- availability
- day
- message
- notification
- refused_worker

Les tables worker et requester contiennent les informations respectivement d'un travailleur et d'un demandeur. Elles héritent toutes les deux de la table account contenant les informations communes aux deux comptes.

La table work contient toutes les informations concernant un travail créé.

Les tables day et type_work contiennent des données prédéfinies. Day contient les jours de la semaine et type_work les différents types de travaux possibles. Elles ne sont donc pas modifiées. Pour relier ces 2 tables à un travailleur, j'ai dû créer 2 tables intermédiaires: availability et worker_type_work.

La table refused_work permet de retenir un travailleur s'il a été refusé pour un travail.

La table message est liée à un travail car des messages ne peuvent être envoyés que pour un travail.

Enfin, la table notification est liée, soit à un demandeur, soit à un travailleur en fonction duquel doit recevoir la notification.

Plateforme web

Technologies

Pour la réalisation de la plateforme web, j'ai décidé d'utiliser différentes technologies.

Front-End

Au niveau du Front-End, j'ai utilisé logiquement de l'HTML pour l'affichage du contenu et le CSS pour le design. Le design n'étant pas ma qualification principale, j'ai décidé d'utiliser un template gratuit pour avoir un rendu assez beau. J'ai cependant, modifié ce template pour pouvoir m'en servir pour réaliser ma plateforme.
C'est un template réalisé avec Bootstrap, un framework que je maîtrise correctement. Ce framework me permet d'ajouter et de modifier facilement des éléments avec un rendu propre.
J'ai également utilisé d'autres frameworks tels que sweetAlert pour afficher des popups et chartJs pour les graphiques..
Concernant les icônes dont je me suis servi, j'ai utilisé en grande partie les icônes fournies avec le template. J'ai aussi utilisé la librairie Font Awesome proposant des icônes web gratuites.

Toujours au niveau Front-End, j'ai fais le choix d'utiliser du javascript pour le côté dynamique de ma plateforme. Javascript me permettait de facilement communiquer avec mon backend via l'outil AJAX. C'est un outil fluide et rapide. Les autres raisons qui m'ont poussé à utiliser ce langage est le fait qu'il est compatible avec tous les supports numériques, qu'il me permet de recharger partiellement une page, de modifier n'importe quel contenu d'une page facilement et rapidement, …

Pour améliorer les performances du langage javascript, j'ai employé en grande partie la librairie JQuery. Cette librairie me permettait d'écrire du code javascript de façon beaucoup plus courte mais aussi d'en améliorer la rapidité.

Back-End

Au niveau Back-End, J'ai fais le choix d'utiliser le langage PHP. Il s'agit d'un langage spécialement conçu pour le développement d'applications web. Je le trouvais donc très approprié à cette réalisation. L'avantage du PHP est aussi que sa connexion avec une base de données est relativement simple. C'est un des langages backend les plus utilisés du marché.
Pour la connexion à la base de données, j'ai utilisé l'extension PDO (PHP Data Object). Cette extension m'a permis de faire tout type de requêtes SQL facilement.

Hébergement

Pour la phase de développement, ma plateforme et ma base de données étaient hébergées sur un serveur local (localhost).
Après avoir décidé du nom de ma plateforme et de mon application (Youngr), J'ai acheté un nom de domaine youngr.be ainsi qu'un hébergement. J'ai effectué cet achat via le service LWS. C'est un service que j'ai l'habitude d'utiliser et par conséquent que je connais assez bien. J'ai d'abord acheté un hébergement de premier niveau. Je passerai au hébergement supérieur lorsque sa capacité ne sera plus suffisante.
Il s'agit d'un serveur Linux basé en France avec un espace disque de 100Go. L'hébergement me permet également de créer 10 adresses mail et d'avoir un trafic mensuel illimité.

Outils

Différents outils m'ont permis de réaliser cette plateforme web.
Tout d'abord, au niveau de l'environnement de développement, j'ai décidé d'utiliser celui que j'emploie habituellement: Visual Studio Code de Microsoft. C'est un IDE gratuit et permettant d'installer des extensions de tout type. C'est pour moi le meilleur IDE pour le développement web.

J'ai utilisé Github pour la sauvegarde et le versionning de mon code. Cet outil est très utile et compatible avec de nombreux autres outils. Pour faciliter son utilisation j'ai installé GitKraken qui me permettait de "push" et de "pull" plus facilement mon code.

Pour gérer facilement ma base de données locale, j'ai utilisé le programme MySQL Workbench.
Une fois ma base de données mise en ligne sur mon serveur, j'ai utilisé phpMyAdmin qui me permettait de la gérer plus facilement depuis le serveur.

Pages

Connexion / Inscription

Il s'agit de la première page du la plateforme web. Cette page est divisée en deux. D'un côté l'inscription, avec un bouton pour s'incrire en tant que travailleur et un bouton pour s'inscrire en tant que demandeur. De l'autre côté, un formulaire de connexion où l'on doit entrer son nom d'utilisateur unique choisi lors de l'inscription et son mot de passe.
Il y a également un lien permettant de récupérer son mot de passe en cas d'oubli

Dashboard

Cette page est la page d'accueil sur laquelle on se retrouve après s'être connecté ou inscrit.
Elle diffère en fonction du type de compte.

Pour un compte de type travailleur, on trouve la partie la plus intéressante pour un travailleur : les propositions de travaux. Ces propositions qui s'affichent sont définies selon plusieurs critères: les disponibilités du travailleur, les types de travaux qu'il réalise, son âge et si la distance entre l'adresse du travail et celle du travailleur est inférieure ou égale à la distance maximale inscrite par le travailleur. C'est en fonction de tous ces critères que sont triés les travaux proposés ou non au travailleur.
Il est possible pour un travail proposé de l'accepter ou de voir l'ensemble des informations le concernant.
En dessous de ces propositions, se trouve les informations concernant le prochain travail à réaliser, si toutefois, il existe. J'ai décidé d'ajouter cette information car il est important pour un travailleur d'avoir rapidement les informations sur son prochain travail. Ainsi, en les mettant sur la première page, il peut les voir directement. Il s'agit des informations sur le travail et le demandeur, c'est-à-dire, la personne ayant posté le travail.

Pour un compte de type demandeur, le dashboard est composé de, tout d'abord, un formulaire lui permettant d'ajouter un nouveau travail. J'ai choisi de mettre ça en tout

premier sur la page d’accueil car c’est la principale fonctionnalité pour un demandeur.
Ce formulaire permet de rentrer différentes informations : le titre qui correspond à une courte description du travail, le type de travail parmi ceux possibles, l'âge minimum du travailleur, la date, l’heure de début et l’heure de fin, la rémunération par heure (entre 8 et 25€/h), l’adresse du travail, qui par défaut reste son adresse et une description plus complète du travail. Le prix du travail est ensuite défini par calcul avec l’heure de départ, l’heure de fin et la rémunération par heure.
Ensuite, comme pour le dashboard travailleur, il y a les informations concernant le prochain travail qui sera effectué ainsi que sur son travailleur.

Travaux

La page des travaux diffère aussi en fonction du type de compte.

Pour un compte travailleur, il y a deux parties à cette page. Une partie “Travaux à faire” contenant les travaux qu’il a acceptés de réaliser mais qui n’ont pas encore été faits. Pour chacun de ces travaux trois actions sont possibles : l’annuler, si le travailleur n’est plus en mesure de le réaliser, voir toutes les informations ou encore le marquer comme terminé lorsque le travail a été réalisé.
L’autre partie porte sur les ‘‘Travaux fini’’. Une fois qu’un travail “à faire” a été marqué comme fini et payé par le travailleur ou le demandeur, il est automatiquement déplacé dans cette partie. Un fois un travail terminé, il est uniquement possible d’en voir les informations.

Pour un compte de type demandeur, la page est divisée en trois parties : l’une permettant de voir ses travaux ajoutés mais encore libre, c’est-à-dire sans travailleur assigné. Sur ces travaux il est uniquement possible de l'annuler ou d’en voir les informations.
La deuxième partie contient les travaux à faire qui ont été assignés à un travailleur. Il est possible sur ces travaux d’en voir les informations, de l'annuler, de le marquer comme terminé et payé ou de supprimer le travailleur. Cette dernière option retirera le travailleur du travail et il ne sera plus possible pour lui de le réaccepter.
Enfin, la dernière partie de la page ‘‘travaux’’ pour demandeur est la liste de ses travaux terminés et payés. Pour celle-ci également, Il est uniquement possible pour lui d’en consulter les informations

Messages

Cette page est identique pour les deux types de comptes. Il s’agit d’une messagerie pour permettre aux travailleurs et demandeurs de discuter ensemble pour

s’échanger de plus amples informations si nécessaire.

Cette messagerie est composée à droite de la liste des conversations. Chaque conversation correspond à un travail à réaliser. Lorsque le travail est marqué comme terminé par une des deux personnes, la conversation n'apparaît plus et il devient alors impossible d’envoyer d’autres messages. Le nom de la conversation est d’ailleurs le titre du travail. Il peut donc y avoir plusieurs conversations avec les mêmes personnes si celles-ci ont plusieurs travaux à réaliser ensemble. Il est également affiché dans la conversation le dernier message, sa date et la photo du destinataire.

A gauche, on retrouve tous les messages d’une conversation lorsque l’on a cliqué sur celle-ci. On peut alors commencer à discuter avec la personne. Les messages sont reçus en temps réel.

Notifications

Cette page contient toutes les notifications reçues par l’utilisateur. Elle est également identique pour les deux types de compte et est divisée en 2 parties.

D’un côté, il y a les nouvelles notifications reçues et de l’autre les vingt dernières déjà lues. Lorsque l’on reçoit une nouvelle notification, elle s’ajoute dans la case de gauche et il est alors possible de soit l’indiquer comme lue grâce au bouton, soit la laisser dans cette partie si l’on souhaite la voir plus facilement. Il y a, dans la barre de navigation, le nombre de notifications non lues en permanence. Cela permet de ne pas oublier d’aller voir les nouvelles informations dans cette section. Une fois marquée comme lue, la notification va automatiquement s’ajouter dans la colonne de droite.
Une notification est composée de son texte et de sa date d’ajout.
Il existe différentes notifications en fonction du type de compte.
Un compte travailleur reçoit une notification lorsqu’un travail, lui étant attribué, est annulé ou marqué comme terminé et lorsqu’un particulier souhaite changer de travailleur.
Un compte demandeur reçoit une notification lorsqu’un de ses travaux a été attribué et lorsqu’un travailleur annule la réalisation de l’un de ses travaux.

Il existe également un code couleur pour ces notifications. Le bleu indique une notification positive comme un travail attribué ou terminé, l’orange est utilisé pour signaler qu’un travailleur annule la réalisation d’un travail et le rouge lorsqu'un travailleur est refusé ou quand un demandeur annule son travail.

Statistiques

J’ai fait le choix d’ajouter également une page de statistiques. Cela permet d’avoir une vue globale de ses activités sur la plateforme. Les statistiques changent également en fonction du type de compte.

Un travailleur pourra voir quatre graphiques. Le premier est un graphique linéaire qui permet de voir les revenus générés par mois sur l’année en cours. Le second est également linéaire, il montre les travaux totaux réalisés par mois sur l’année en cours. Le troisième est un graphique de type “donut”. Celui-ci montre par différentes couleurs le nombre total de travaux réalisés par type. Enfin, le dernier graphique se présente en bâtonnets horizontaux, il permet au travailleur de voir les villes dans lesquelles il a réalisé le plus de travaux. Tous ces graphiques se basent sur les travaux marqués comme terminés et payés.

Dans le cas du demandeur, on trouve deux graphiques à sa disposition. Le premier est un graphique linéaire montrant le total des travaux donnés par mois sur l’année en cours. Le second est un graphique “donut” montrant le total des travaux donnés par types. Ces données sont également basées sur les travaux terminés et payés.

Compte

La page ‘‘compte’’ permet à l’utilisateur de modifier ses informations personnelles. Toutes les informations sont modifiables hors le nom d’utilisateur unique qui n’est pas modifiable et le mot de passe qui lui, est seulement modifiable dans la section ‘‘mot de passe perdu’’ de la page de connexion.
Pour les travailleurs, il est également possible de changer ses disponibilités et ses types de travaux.
Cette page permet également aux utilisateurs d’ajouter leur photo de profil. Une image de profil par défaut s’affiche si aucune photo n’a été ajoutée.

Aide

Il s’agit d’une page pour aider les utilisateurs à se servir de la plateforme correctement. Elle décrit l’utilisation de l'ensemble des fonctionnalités disponibles pour chaque type de compte. Cette page diffère en fonction du type de compte utilisé

Technique

De nombreux aspects techniques sont à préciser dans cette plateforme.

Les sessions

Lors de la connexion la session de l'utilisateur est enregistrée en PHP.
La variable $_SESSION est une des sept variables superglobales que propose PHP. Elle est spécialement prévue pour l'enregistrement des sessions utilisateurs sur une application web. Sur chaque page de la plateforme, la session doit être démarrée grâce à la fonction prédéfinie session_start. Une fois la session lancée, on a accès au contenu de la variable superglobale. J'ai enregistré différentes informations dans cette variable : l'id du compte, le type de compte, l'id du type de compte, le prénom et le nom de l'utilisateur. Lorsque celui-ci souhaite se déconnecter, il lui suffit de cliquer sur le bouton de déconnexion et la fonction session_destroy sera alors appelée.
J'ai fait le choix d'utiliser les sessions en PHP car toutes les pages de ma plateforme sont en PHP. Les informations contenues dans la sessions sont alors accessibles partout.

API Geocoding Google

Lorsqu'un jeune s'inscrit, il doit entrer son adresse ainsi qu'un distance maximale de déplacement comprise entre un et cinquante kilomètres. Une fois que le travailleur arrive sur son dashboard, une liste de travaux possibles lui est proposée selon ses différents critères. La distance maximale est l'un des critères utilisés.
Lors du filtrage de ce critère, l'adresse du travailleur ainsi que l'adresse où se déroule le travail sont collectées. Ces adresses sont ensuite envoyées via une fonction PHP que j'ai réalisé à l'API Google Geocoding. Ma fonction récupère les deux adresses, elle les convertit pour les utiliser dans une URL, elle les envoie à l'API Google pour en récupérer des données en Geodata. De ces données sont, ensuite, extraites les latitudes et longitudes pour en calculer la distance. Si cette distance est inférieure ou égale à la distance maximale entrée par l'utilisateur, le travail pourra lui être proposé.
L'API Geocoding peut renvoyer les données soit sous forme JSON soit en XML. Chaque appel API est enregistré sur la plateforme Google Cloud Plateform. Celle-ci permet de voir des statistiques sur les appels effectués comme le temps, le traffic, les erreurs et la latence. Cette API, comme toutes les autres API de Google, est payante. Le prix est défini en fonction du nombre de requêtes effectuées. Ici, le prix est de 5$ pour 1000 requêtes. Cependant, je dispose actuellement d'un essai gratuit de 300€ et de 350 jours maximum.

Actualisation

Lors de l'envoi d'un message ou d'une notification, la plateforme en reçoit directement l'information. L'utilisateur n'est pas obligé de rafraîchir sa page pour avoir accès à ses nouveaux messages ou ses nouvelles notifications. Cette actualisation se fait toutes les trois secondes et affiche dans la barre de navigation un petit nombre sur l'onglet correspondant avec le nombre de messages non lus ou de notifications non lues. Lorsqu'on se trouve sur une conversation de la page ''messages'' et que l'on reçoit un message, celui-ci s'affiche automatiquement.
Cette actualisation permanente se fait grâce au javascript et plus précisément avec la méthode setIntervals. Cette dernière reçoit en paramètre une fonction à exécuter et une durée qui correspond au temps de rafraîchissement. La fonction exécutée effectue donc un appel ajax vers du PHP qui va chercher les nouveaux messages dans la base de données.
Pour ne pas alourdir trop le chargement de l'actualisation, la fonction va d'abord chercher uniquement le nombre de messages ou de notifications non lus et vérifie si elle correspond au nombre existant déjà. Si le nombre est identique elle ne fait rien mais si il est différent, elle ira chercher le contenu des nouveaux messages pour les afficher.

Appels AJAX

Pour permettre la liaison entre le javascript et le PHP, j'utilise des appels AJAX. AJAX signifie *Asynchronous JavaScript and XML*. Cet outil sert à envoyer des données du côté serveur(ici en PHP) pour y effectuer un traitement et en récupérer le résultat.
Je m'en sers ici pour envoyer les données d'un formulaire à un fichier PHP comme lors de la connexion, l'inscription ou l'ajout d'un travail. Je l'utilise également pour recuperer mes messages et mes notifications lors de l'actualisation.
L'AJAX a comme principale avantage de diminuer le temps de latence et ainsi d'augmenter la réactivité de ma plateforme.

Upload Image

Chaque utilisateur à la possibilité d'ajouter une photo de profil à son compte. Cette photo est téléchargée depuis un input HTML jusqu'au serveur. Cet upload passe par un fichier PHP vérifiant différentes choses.
Tout d'abord, il y a une vérification de l'extension du fichier ajouté. Seul les extensions JPG, JPEG, PNG et GIF sont acceptées.
Ensuite, le fichier vérifie si la taille n'est pas supérieure à la taille maximale d'upload du fichier de configuration de PHP. Cette taille est de deux mégaoctets. Cela permet de ne pas surcharger le serveur avec des images trop lourdes inutilement.

Enfin, si l'image est valide, elle sera renommée avec l'id de l'utilisateur et ajoutée sur le serveur.
Le lien relatif vers l'image sera lui, enregistré dans la base de données.

Sécurité

Diverses couches de sécurité ont été mises en place pour la réalisation de cette plateforme.

HTTPS

Lors l'achat de mon hébergement et de mon nom de domaine sur le service LWS, j'ai eu la possibilité de mettre en place un certificat SSL pour sécuriser mon site. Il s'agit d'un certificat de type Let's Encrypt gratuit. Celui-ci permet de sécuriser les connexions sur le site internet, chiffrer les contenus échangés mais aussi de s'assurer de l'identité du serveur.

PDO PHP

Pour recevoir les données sur mon site, j'utilise des requêtes SQL en PHP. Ces requêtes sont effectuées grâce à l'extension PDO. Elle fournit une connexion à la base de données et la possibilité d'effectuer tous types de requêtes SQL. Cette extension possède les méthodes *prepare* et *execute* qui permettent d'éviter les injections SQL lors des requêtes. J'ai donc utilisé ces méthodes pour sécuriser les appels à la base de données. Lorsque je n'utilisais pas ces méthodes je faisais appel à la méthode htmlspecialchars(expliquée ci-dessous) permettant également d'éviter ces injections.

htmlspecialchars

Cette méthode PHP sert à convertir certains caractères en codes pour leurs permettre de garder leurs significations. Cela permet également d'éviter les injections SQL dans la base de données. J'utilise ici cette méthode avant d'effectuer une requête SQL ou lorsque je n'utilise pas les méthodes *prepare* et *execute* de PDO.

Mot de passe

Lors de l'inscription, il est demandé à l'utilisateur d'entrer un mot de passe pour lui permettre de se connecter à la plateforme. J'ai fait le choix de forcer l'utilisateur à entrer un mot de passe avec au minimum une lettre, un chiffre avec huit caractères en tout. Cela permet d'éviter la découverte du mot de passe par des attaques de

force brute ou autres attaques.
Dans le formulaire d’inscription, il faut également confirmer son mot de passe pour que l’utilisateur ne se trompe pas lors de l’encodage de celui-ci.

Application mobile

Design

La réalisation du visuel de l'application a été réalisée avant de commencer le développement pour faciliter celui-ci. Pour la conception du design, j'ai utilisé Figma. Ce logiciel permet de créer les différentes vues de mon application mais aussi de créer les liens entre ces vues. L'intérêt de réaliser le design de l'application au préalable est d'avoir un premier aperçu de son rendu. Cela permet surtout de développer plus rapidement et de ne pas perdre du temps à penser au rendu.

Technologies

Front-End

J'ai fait le choix de créer une application mobile compatible à la fois sur Android et sur Apple. Pour ce faire, j'ai décidé de développer cette application en React Native.

Ce framework open source a été créé par Facebook en 2015. Il utilise le langage Javascript et permet de créer des applications cross-plateforme. A l'inverse des applications natives, une application cross-platform permet de ne devoir développer qu'une seule fois. Cependant l'avantage d'utiliser le react native est que celui-ci est cross-platform mais utilise les composants natifs de l'OS sur lequel il se trouve.

Pour améliorer le rendu de mon application mobile, j'ai fait le choix d'utiliser un framework principalement, Galio. Celui-ci permet d'utiliser des composants prêts à l'emploi tels que des boutons, des cartes, des champs de texte … Il me permet également de positionner et de redimensionner plus facilement mes composants sur l'écran.

Un autre framework que j'ai utilisé, est Native Base. Ce dernier donne la possibilité d'ajouter des composants déjà créer comme Galio. Je l'ai utilisé ici car il possède plus de composants de Galio. Je trouvais intéressant d'utiliser plusieurs frameworks de design react native pour pouvoir les comparer.
J'ai également fait appel à la librairie react native chart kit pour afficher le graphique des revenus pour les travailleurs sur cette application..

Back-End

Au niveau Backend, j'ai utilisé différents frameworks et langages de programmation.

Tout d'abord, pour permettre aux utilisateurs de naviguer sur l'application, j'ai fait appel à React Navigation et pour avoir accès à certaines informations depuis n'importe où sur mon application, j'ai fait appel à React Redux. Ces deux outils ont été créés spécifiquement pour le langage React.

Ensuite, le langage de programmation backend du React Native est le JSX. Il s'agit d'un langage très proche du Javascript.

Enfin, pour récupérer les données de ma base de données, j'ai décidé de créer une API. Mon application fait appel à cette dernière pour se connecter, afficher les informations ou encore envoyer des messages. J'ai fait le choix de réaliser cette API en PHP car la connexion à la base de données est rapide et sécurisée. C'est un langage très complet et très utilisé.

Hébergement

Il y a deux grandes plateformes de publications d'applications mobiles : Play Store de Google pour les appareils Android et autres et Apple Store d'Apple.

Pour pouvoir publier son application sur Apple Store, il est obligatoire d'avoir un ordinateur Apple en sa possession. N'ayant pas la possibilité d'en avoir un, je ne me suis pas concentré sur la publication de mon application sur ce store.
Je me suis plutôt penché sur la réalisation d'une APK permettant d'installer mon application sur n'importe quel appareil Android et de la publier sur Play Store.
Une APK est une version exécutable de mon application sur mobile android.
Elle est donc accessible à toutes les personnes possédant un appareil avec Android comme système d'exploitation.

Le coût de publication d'une application est de 25€ sur Play Store et de 100€ par an pour un compte sur Apple Store. La publication sur Apple Store viendra par la suite lorsque j'aurai la possibilité d'utiliser un ordinateur Apple.

Outils

J'ai utilisé différents outils pour la réalisation de cette application. Tout d'abord, pour le développement, j'ai continué à travailler sur l'IDE Visual Studio Code. Il permet d'installer des plugins aidant fortement pour ce développement.

Concernant la sauvegarde de mon code, j'ai utilisé Github tout comme pour ma plateforme web.

Ensuite, pour avoir un visuel de mon application, j'ai utilisé la plateforme. Au lancement d'Expo, celui-ci démarre un serveur NodeJS et génère un code QR. Il faut ensuite scanner ce code sur l'application Expo pour avoir un visuel de mon application. Les avantages de cette plateforme sont le fait que les changements se font en direct et que grâce à Expo, mon application est testable directement sur n'importe quels systèmes d'exploitation.

Puis, pour la réalisation de mon API, j'ai décidé d'utiliser Postman. C'est un logiciel permettant d'effectuer des appels à une API, de la documenter, d'effectuer des tests automatisés, … Je m'en suis servi dans le cadre de ce TFE pour tester mon API notamment grâce aux tests d'intégration.

Enfin, pour la création de mon application en version APK, j'ai dû utiliser le logiciel Android Studio. Pour la version Apple, il faut utiliser Xcode.

Pages

Connexion

La première page permet de se connecter à l'application avec son nom d'utilisateur unique et son mot de passe. Si la personne ne possède pas encore de compte, il y a un lien en-dessous permettant de la rediriger vers la page d'inscription de la plateforme web.

Dashboard

La page dashboard est différente suivant le type de compte. C'est la première page sur laquelle on arrive après la connexion.

Pour un compte travailleur, cette page contient tout d'abord, les trois prochains travaux à réaliser. J'ai décidé de mettre cette information en tout premier car le but premier de l'application mobile pour les travailleurs, est d'avoir les informations sur les prochains travaux à portée de main. En-dessous de ces trois travaux, j'ai ajouté un graphique montrant les revenus générés par le travailleur sur l'année en cours car je pense qu'en voyant cette information directement cela poussera le travailleur à effectuer plus de travaux pour gagner plus d'argent.

Pour un compte demandeur, le dashboard contient également ses 3 prochains travaux qui vont être réalisés car c'est aussi important pour eux d'avoir cette information sous la main. En-dessous se trouve les détails complets sur son prochain travail qui va être réalisé.

Travaux

La page ‘‘travaux’’ est également différente selon le type de compte.

Pour un compte travailleur, la page est composée d'abord, de la liste des propositions de travaux. Ces travaux sont évidemment définis selon les mêmes critères que sur la plateforme web. En-dessous de ces propositions, se trouve la liste des travaux qu'il a déjà acceptée et qu'il doit donc prochainement réaliser. En tout dernier se trouve la liste des travaux qu'il a déjà effectués.

Pour un compte demandeur, Cette page contient comme sur la plateforme web la liste de ses travaux libres, pris et terminés.

Détails

La page ‘‘détails’’ est une page qui est appelée lorsque l'utilisateur clique sur un travail. Le contenu de celle-ci va différer en fonction de l'état du travail et du type de compte. Les informations communes dans chaque page ‘‘détails’’, sont les informations du travail. C'est-à-dire : le titre, le type, la date, l'heure de début et de fin, le prix, l'adresse et la description.

Si l'on est sur un compte travailleur, lorsque l'on clique sur une proposition de travail on obtient, en plus des informations communes, les informations du demandeur avec la possibilité de lui envoyer un message ou de l'appeler, ainsi qu'un bouton pour accepter le travail. Pour les travaux pris d'un travailleur, on y retrouve en plus un bouton permettant d'annuler le travail.

Si l'on est sur un compte demandeur, lorsque l'on clique sur un travail pris, on obtient des informations sur le travailleur avec la possibilité de lui envoyer un message ou de l'appeler et un bouton pour annuler le travail, un pour refuser le travailleur et un pour marquer le travail comme terminé.

Messages

La page message est identique pour les deux types de compte. Lorsque l'utilisateur arrive dessus, il y a tout d'abord, la liste des conversations correspondant aux

travaux à réaliser. Sur chaque conversation est affichée le dernier message, la date de celui-ci ainsi que le titre du travail.
Une fois que l'utilisateur clique sur une conversation, celui-ci arrive sur les messages de cette conversation avec la possibilité de lui en envoyer un.

Technique

React Navigation

Il s'agit d'une librairie React Native permettant de naviguer entre différentes pages. Il existe plusieurs types de navigation. Ici deux types de navigation ont été mis en place. La première est une navigation de type Stack, les différents écrans vont se superposés pour chaque fois afficher le nouveau par dessus l'ancien. Cela permet également de revenir en arrière sur la vue précédente. J'ai utilisé cette navigation pour mes pages détails, la page d'une conversation et pour la page de connexion.
Le deuxième type de navigation utilisé dans mon application est la Bottom Navigation. Celle-ci crée une barre de navigation en bas de l'écran avec différents onglets. Cela permet de naviguer facilement entre plusieurs pages principales. La Bottom Navigation de mon application contient les pages dashboard, travaux et Messages.
Cette librairie est la plus utilisée et la plus performante dans les applications React Native.

React Redux

Cette librairie Javascript va permettre de créer une variable globale à toute l'application. En comparaison au langage PHP utilisé pour la plateforme web, Redux serait comme la variable SESSION. J'enregistre toutes les informations liées au compte dans cette variable globale. Lorsqu'une modification sera effectuée dans cette variable globale cela se modifiera directement sur toutes les pages. React Redux est le système de centralisation de données le plus performant en React Native.

API

L'API est réalisée en PHP et est hébergée sur le serveur LWS. Elle effectue des requêtes SQL sur la base de données et certains traitements sur ces données avant de les envoyer. L'API est appelée depuis l'application en Javascript via la méthode Fetch. Cette dernière est asynchrone et reçoit les informations de L'API sous forme JSON. J'ai utilisé Fetch car celle-ci est plus efficace que les autres telles que XHR.

Ejection

Une fois le visuel de mon application terminé, J'ai dû la formaliser pour utiliser des composants de l'appareil. Cette formalisation se fait automatiquement via une commande React Native. Cela va changer le fichier d'index de l'application et créer un dossier spécifique pour IOS et un pour Android. A partir de ce moment, l'application ne se lance plus sur Expo mais directement sur un simulateur ou sur le téléphone pour les appareils Android. Je devais donc lancer l'application sur mon téléphone ainsi qu'un serveur NodeJS pour continuer à développer mon application et voir les changements en direct.

Persistance

La dernière étape pour permettre à l'application de fonctionner correctement sur un appareil mobile sans serveur NodeJS est la persistance des données au sein de cette application. Cette étape consiste à l'enregistrement permanent des données contenues dans Redux et ainsi le maintien de la session utilisateur.

Sécurité

Connexion

L'application dispose d'une page permettant à l'utilisateur de se connecter sur le compte créé via la plateforme web. Les informations de cette page de connexion sont validés par l'API et cette dernière renvoie des messages d'erreur correspondants en cas d'erreur de connexion.

API

Au niveau de l'API, j'ai mis en place une sécurité d'authentification de l'utilisateur vu que cette dernière est en ligne sur le serveur et qu'elle fournit des informations personnelles telles que des messages, données personnelles ...
Cette authentification identifie l'utilisateur via un système de token. Celui-ci est généré automatiquement et permet à l'API de savoir de quel compte il s'agit.
Les données que renvoie cette API ne sont donc pas visibles directement depuis un navigateur ou depuis un quelconque logiciel permettant d'effectuer des appels API tels que Postman.

Site vitrine

Présentation

J’ai décidé de réaliser un site de vitrine pour pouvoir présenter le principe de Youngr aux personnes avant qu’elles ne s’inscrivent. Les gens préfèrent toujours d’abord avoir les informations sur le produit avant de devoir s'inscrire.
Ce site de présentation contient toutes les informations nécessaires aussi bien pour un jeune travailleur que pour une personne ayant besoin d’aide.
Il y a également les informations et les liens concernant le téléchargement de l’application mobile.

Pour permettre à l'utilisateur de naviguer sur ce site, une barre de navigation est mise en place. J’ai fait le choix d’utiliser un template gratuit lors de la réalisation de ce site pour avoir le meilleur rendu possible.
Il est important d’avoir un très bon rendu sur le site vitrine car celui-ci est la première chose que voit l’utilisateur en arrivant sur Youngr.be.
Pour permettre à l’utilisateur d’aller sur sa plateforme, un bouton est disponible sur la barre de navigation. La plateforme web, elle, est hébergée sur un sous domaine de Youngr.be.

Contact

Une page ‘‘contact’’ est également mise à disposition sur le site vitrine. Ce formulaire de contact permet à l’utilisateur d’envoyer un mail en cas de problèmes, questions ou tout autres demandes.
Une adresse mail a été créée à cet effet sur le domaine: help@youngr.be
Avec cet hébergement, j’ai la possibilité de créer dix adresses mail pour ce domaine. Je peux donc en créer de nouvelles plus spécifiques au besoin.

Légalité

Condition générale d'utilisation

Les conditions générales d'utilisations sont divisées en plusieurs parties.
Tout d'abord, il y a l'objet de la réalisation de ces conditions ainsi que des mentions légales de la personne ayant réalisée le site web. En l'occurrence, ici un personne physique et non une société.

Ensuite, les conditions d'accès au site sont définies ainsi que la création d'un compte. Ces conditions sont différentes en fonction du type de compte créé.
Pour un travailleur, il est stipulé qu'il doit avoir au minimum 15 ans, l'âge légal pour travailler en belgique. Si ce dernier n'est pas majeur, il doit avoir l'accord d'un parent ou d'un tuteur légal pour s'inscrire. Cette partie stipule également les accès aux différents services disponibles.

Puis, au niveau des responsabilités de l'utilisateur, il est stipulé que ses revenus ne peuvent pas excéder 6340€ annuel ni 528,33€ mensuel, conformément à la loi belge au sujet des services de citoyen à citoyen. Il est également indiqué que chaque travailleur doit déclarer lui-même ses revenus pour être en toute légalité et être couvert d'une assurance en responsabilité civile en cas de dommage.
Le demandeur, lui, s'engage à ne pas donner de travaux qualifiés comme dangereux pour le travailleur.

Après, sont stipulées les obligations de ces travailleurs. Cette partie indique qu'une fois un travail accepté et non annulé, le travailleur s'engage à réaliser la prestation et le demandeur à rémunérer ce travailleur.

Enfin, ces conditions générales indiquent que l'éditeur du site web ne peut en aucun cas être tenu responsable lors d'un accident ou d'un quelconque dommage.
Il s'agit ici, d'un résumé des conditions générales d'utilisation du site youngr.be

Politique de confidentialité

La politique de confidentialité concerne l'utilisation des données personnelles récoltées.
Il y est décrit les données de l'utilisateur susceptibles d'être traitées ainsi que le traitement qu'elles pourraient subir.

Il est également indiqué la durée de conservation des données à caractère personnel, conformément au nouveau règlement général sur la protection des données.
Enfin, il est signalé que le consentement de l'utilisateur est accordé pour l'utilisation de ses données personnelles.

Méthode de travail

Outils

J’ai utilisé plusieurs outils pour gérer au mieux ce travail.

Tout d’abord, pour avoir une vision globale du travail à accomplir, j’ai décidé d’utiliser le site Asana. Il s’agit d’un site ressemblant à Trello et permettant de créer un tableau avec différentes cartes. J’ai donc divisé mon travail en de nombreuses cartes que j’ai triées par degré de priorité. Une fois une carte terminée, je la déplaçais dans la colonnes voisine correspondant aux cartes terminées.
J’ai ajouté la plupart des cartes en début de projet mais lorsqu'une nouvelle idée ou une modification était à apporter j’ajoutais une nouvelle carte à la colonne “A Faire”.

Puis, l’utilisation de clockify m’a permis de compter mes heures de travail. Chaque jour après avoir fini de travailler, j’ajoutais le temps de travail sur clockify. Cet outil m’a surtout donné une vue d’ensemble sur le temps total de travail fourni et sa répartition.

Répartition

Au niveau de la répartition du travail, j’ai procédé en plusieurs étapes.

Tout d’abord, j’ai commencé par une période de recherche pour trouver l’idée de ce TFE, les différents services, le public ciblé. Cette étape précédait la réalisation de mon cahier des charges pour le défense technique de janvier.

Ensuite, je me suis mis à la réalisation du design de mon application sur Figma et de la recherche d’un template pour ma plateforme web.

Puis, sur base de toutes les informations que j’avais récoltées et des services que j’avais imaginés, j’ai pu me lancer dans la réalisation de ma base de données. Je l’ai par la suite encore modifiée pour avoir au final la base de données la plus optimale possible.

Après, j’ai pu me lancer dans le développement de mon application mobile et de ma plateforme web. J’ai fait le choix de réaliser les deux en même temps car cela me permettait de mieux voir si j’oubliais quelque chose d’important dans l’une ou l’autre.

Enfin, je me suis mis à la rédaction de ce rapport final.

Pour permettre de voir la répartition selon les différentes parties, j'ai créé sept catégories sur Clockify. Ces différentes parties étaient Web, Application, Documentation, Recherche, Base de données, Design et Api.

Gestion du temps

Au niveau de la gestion du temps, j'ai commencé le design assez tôt dans l'année scolaire pour pouvoir commencer mon développement au plus vite. Sachant que celui-ci était assez conséquent.

Etant en stage de mi-février à mi-mai, je pouvais uniquement travailler le soir sur mon TFE en rentrant du stage. Cette période était plus compliquée à gérer car cela me faisait des journées très longues et j'avais donc peu de temps en dehors. J'ai donc essayé de travailler un maximum chaque weekend sur mon TFE afin de fournir le travail nécessaire. A la fin de mon stage jusque début juin, j'ai pu me concentrer sur mon TFE et particulièrement sur le rédaction de ce rapport.

Je pense avoir su gérer mon temps comme je le souhaitais.Il n'a cependant pas été facile de travailler aussi bien durant la période de confinement obligatoire dû à la crise sanitaire du Covid-19. En effet, Il est plus simple pour moi de me mettre dans une ambiance de travail lorsque je suis avec d'autres personnes qui travaillent autour de moi. Malgré cela, j'ai su m'adapter et terminer dans les temps la réalisation de ce TFE.

Conclusion

Bilan

Je pense avoir pu fournir une solution fonctionnelle et attractive concernant la problématique de départ. Cette solution fournit des avantages pour chacun des utilisateurs. Elle est donc intéressante pour ceux-ci.
J'ai eu l'opportunité de découvrir de nouvelles technologies totalement inconnues comme le développement mobile et d'une API. J'ai pu également approfondir mes connaissances en développement web grâce à l'utilisation de nouveaux frameworks.

Au terme de ce TFE, j'ai pu non seulement développer une solution fonctionnelle mais aussi la mettre en ligne. Le site vitrine ainsi que la plateforme web est accessible sur le site youngr.be et l'application mobile est installable en téléchargeant l'APK ou directement depuis Play Store.

Améliorations

Certaines améliorations peuvent évidemment être faites sur cette solution.

Au niveau des fonctionnalités, la modification des informations d'un travail ou la possibilité de créer un travail sur plusieurs jours peuvent être des améliorations intéressantes. Le paiement en ligne peut également être une fonctionnalité pratique pour les utilisateurs.

Concernant l'application mobile, les meilleures améliorations selon moi seraient la possibilité de lancer automatiquement le GPS par défaut du téléphone lorsque l'on clique sur l'adresse ou encore la possibilité d'ajouter un travail directement depuis cette application.

Futur

Dans un futur proche, mes objectifs concernant Youngr sont de réaliser un maximum de publicité pour attirer les gens dessus. Je compte principalement sur les réseaux sociaux, le bouche à oreille et les médias pour augmenter son utilisation.
Je compte également me pencher sur la réalisation des améliorations décrites ci-dessus pour avoir une évolution permanente de cette solution.

Bibliographie

Web / Application

- https://www.php.net/
- https://developer.mozilla.org/fr/docs/Web/JavaScript
- https://stackoverflow.com/
- https://www.creative-tim.com/product/now-ui-dashboard
- https://redux.js.org/basics/usage-with-react
- https://reactnavigation.org/
- https://openclassrooms.com/fr/courses/4902061-developpez-une-application-mobile-react-native/4902068-decouvrez-le-developpement-mobile-actuel
- https://galio.io/docs/#/install
- https://nativebase.io/
- https://console.developers.google.com/?hl=FR&pli=1
- https://levelup.gitconnected.com/using-jwt-in-your-react-redux-app-for-authorization-d31be51a50d2
- https://sweetalert2.github.io/
- https://www.chartjs.org/
- https://getbootstrap.com/
- https://www.bootstraptoggle.com/
- https://github.com/
- https://fontawesome.com/
- https://www.figma.com/
- https://pixabay.com/fr/photos/poign%C3%A9e-de-main-les-mains-3382503/

Base de données

- https://dbdiagram.io/home
- http://www.mocodo.net/
- https://sql.sh/
- https://www.lws-hosting.be/

Légalité

- https://www.codeur.com/blog/cgu-site-internet/
- https://www.captaincontrat.com/articles-droit-commercial/cgu-conditions-generales-utilisation
- https://www.bijklussen.be/fr/service-de-citoyen-a-citoyen.html

Méthodes de travail

- https://clockify.me/
- https://app.asana.com