ProMatch

Préparé par Jyotsna Bhunjun, Éléa Charier, Eya Cherni, Aurélie Fidélia, Sara Hamed & Neda Javaheri

16 avril 2025

Table des matières

1.Introduction

But

Promatch est un réseau social permettant de faciliter la communication ainsi que la recherche des stages et de stagiaires entre l'employeur et l'étudiant avec une simple fonctionnalité de "swipe". La plateforme consiste d'une page web qui focalise sur les fonctionnalités en lien avec l'employeur en leurs offrant des outils ciblés pour publier, gérer et consulter les candidatures à leurs offres de stages et une application mobile qui contient les fonctionnalités en lien avec l'étudiant, leur permettant de découvrir, filtrer et postuler à des stages selon leurs besoins et études universitaires.

2. Processus d'affaires

L'application permettra aux étudiants de :

- Consulter des offres de stage à jour, centralisées dans une interface simple et attractive ;
- Filtrer les stages selon plusieurs critères : domaine (génie logiciel, mécanique, électrique...), niveau, compétences requises, durée, localisation, télétravail, etc. ;
- Postuler rapidement à l'aide d'un système de type « swipe » inspiré d'applications modernes (glisser vers la droite pour postuler, vers la gauche pour ignorer) ;
- Suivre leurs candidatures dans un tableau de bord intégré : statut des demandes (envoyée, en attente, refusée, acceptée), relances automatiques, rappels de suivi, notes personnelles ;
- Mettre à jour leur profil étudiant (CV, lettre de motivation, portfolio, liens GitHub ou LinkedIn) pour chaque candidature ;
- Favoriser des offres pour postuler plus tard (fonction "Favoris") ;
- Recevoir des recommandations personnalisées, en fonction des champs d'intérêt, des compétences, et de l'historique d'activité dans l'application.

Du côté des employeurs, la solution prévoit :

- Un espace administrateur pour publier des offres de stage, gérer les candidatures reçues, filtrer par compétences ou programme, et avec les candidats ;
- La possibilité de consulter les profils complets des étudiants (CV, parcours académique, disponibilité) ;
- Des outils d'analyse simples pour évaluer la visibilité et la performance de leurs offres (clics, candidatures, taux d'engagement).
- La possibilité de consulter son profil, de le modifier et de voir ses offres de stage déposées.

3.Conception et développement agile

Méthodologie choisie

Nous avons adopté la méthodologie **Scrum**, une méthode agile centrée sur une organisation itérative et incrémentale du développement logiciel. Cette approche favorise l'adaptabilité, la collaboration et la livraison fréquente de versions fonctionnelles du produit.

Organisation des sprints

- **Nombre de sprints:** [3 sprints]
- **Durée de chaque sprint** : 10 jours par sprints
- **Sprint planning** : Une séance de planification a été organisée en début de chaque sprint pour définir les objectifs et prioriser les tâches à réaliser.
- **Daily Scrum** : Des points quotidiens de 15 minutes ont été tenus pour assurer le suivi de l'avancement et lever les obstacles.
- **Sprint review et rétrospective** : À la fin de chaque sprint, une revue était effectuée pour évaluer les livrables suivie d'une rétrospective pour identifier les axes d'amélioration à l'aide de l'enseignant.

Outils utilisés

- **Gestion de projet** : Azure DevOps
- **Contrôle de version** : Git et GitHub
- **Communication** : Discord et imessage

Gestion des rôles Scrum

- **Product Owner (PO)** : L'enseignant Abderrazak Sahraoui, responsable de la définition des besoins et de la priorisation des éléments du backlog.
- **Scrum Master (SM)** : L'enseignant Abderrazak Sahraoui garant de l'application de la méthodologie Scrum, facilitateur entre les membres de l'équipe et le PO.
- **Équipe de développement** : Composée de notre équipe, chargée de transformer les besoins en fonctionnalités concrètes.

Fonctionnalités par composante

Composantes	Fonctionnalités principales
Base de données	- Stockage des utilisateurs, stages, CV, candidatures, etc - Requêtes relationnelles optimisées
API (REST)	- Authentification via JWT et session-express - Endpoints pour stages, profils, candidatures
Back-End	- Traitement des données - Gestion des rôles et des sessions - Chiffrement des mots de passe

Composantes	Fonctionnalités principales
Front-End Web	- Interface d'inscription/login employeur/étudiant - Dépôt d'offre de stage - Suivi des candidatures - Profil employeur
Front-End Mobile	- Interface d'inscription/login étudiant - Navigation dans les offres - Postuler et suivre les candidatures - Profil étudiant

Technologies et langages utilisés

Composante	Technologies / Logiciels utilisés
Base de données	SQL Server
API / Back-End	Node.js, Express.js, JWT, Bcrypt
Front-End Web	HTML, CSS, JavaScript, Tailwind CSS
Front-End Mobile	Android Studio, Java, Retrofit, ViewPager
Diagramme d'architecture	Réalisé avec Figma

Développement

L'ensemble du code source de notre projet est disponible sur GitHub :

- Frontend Web : https://github.com/elea-c/TCH099_Web.git
- Application mobile Android : https://github.com/rbhunjun22/MOBILE_PROJET.git

Références principales aux fichiers de code :

- index.js : Point d'entrée du serveur Express
- authController.js : Gère l'inscription, la connexion, et la génération des tokens JWT
- routes.js : Contient les routes Express
- RetrofitClient.java : Gère la communication entre l'application Android et le backend
- SwipingActivity.java : Activité principale pour swiper les stages
- StageAPI.java : Interface Retrofit pour appeler les routes /stages

4.Déploiement

Le déploiement de notre application **ProMatch** repose sur l'installation et la configuration de trois composantes principales : le serveur backend (API), la base de données SQL Server, et l'application mobile Android. L'objectif est d'assurer une communication fluide entre le site web (interface employeur) et l'application mobile (interface étudiant), les deux partageant la même base de données. Voici les étapes à suivre pour installer et exécuter correctement le système complet.

Dans un premier temps, il faut cloner le projet à partir du dépôt GitHub, en utilisant les commandes suivantes : git clone https://github.com/votre-repo/ProMatch.git et cd ProMatch. Ensuite, on procède à l'installation du **backend**, développé avec **Node.js** et **Express**. Pour cela, certains prérequis doivent être installés : Node.js (version 18 ou plus), un éditeur de code comme Visual Studio Code, ainsi qu'un accès valide à la base de données SQL Server (hébergée sur Azure dans notre projet).

Une fois les outils en place, il faut configurer les informations de connexion à la base de données. Cela se fait en créant un fichier .env dans le répertoire du backend, avec les paramètres suivants : Le nom du serveur SQL hébergé sur Azure, le nom d'utilisateur pour se connecter à la base SQL, le mot de passe associé à l'utilisateur pour l'authentification , le nom de la base de données ciblée sur le serveur, le port sur lequel la base écoute et une clé secrète.

On installe ensuite les dépendances nécessaires avec la commande :

```
npm install
```

Puis on lance le serveur à l'aide de :

```
node index.js
```

Le serveur démarre alors sur le port 3000 ou 8080, selon la configuration. Il expose plusieurs routes API essentielles, notamment pour l'authentification (/connexion, /inscription), la consultation des stages (/stages), et la gestion des candidatures (/candidatures/:id).

Concernant la **base de données**, celle-ci est déjà déployée sur Azure SQL Server. Elle comprend toutes les tables nécessaires à la gestion des utilisateurs, stages, candidatures, CVs et connexions. Les requêtes SQL intégrées dans le backend permettent d'interagir automatiquement avec cette structure, sans avoir besoin de migration manuelle si la base est déjà en place.

Du côté de l'application **Android**, développée avec Android Studio, l'utilisateur doit ouvrir le dossier du projet mobile, généralement situé dans /mobile/ProMatchApp. Il est également nécessaire d'avoir un émulateur ou un appareil physique connecté pour exécuter l'application.

Avant de lancer l'application, il faut s'assurer que l'URL de base du backend est bien configurée. Dans le fichier RetrofitClient.java, il est recommandé d'utiliser l'adresse suivante pour le développement local :

public static final String BASE_URL = http://10.0.2.2:3000/;

Cette adresse permet à l'émulateur Android d'accéder au backend hébergé sur l'ordinateur hôte. Si l'API est déployée sur un serveur distant, il suffit de remplacer cette URL par l'adresse publique correspondante.

Enfin, pour exécuter l'application, il suffit de cliquer sur "Run" dans Android Studio. L'application se lance et permet aux étudiants de :

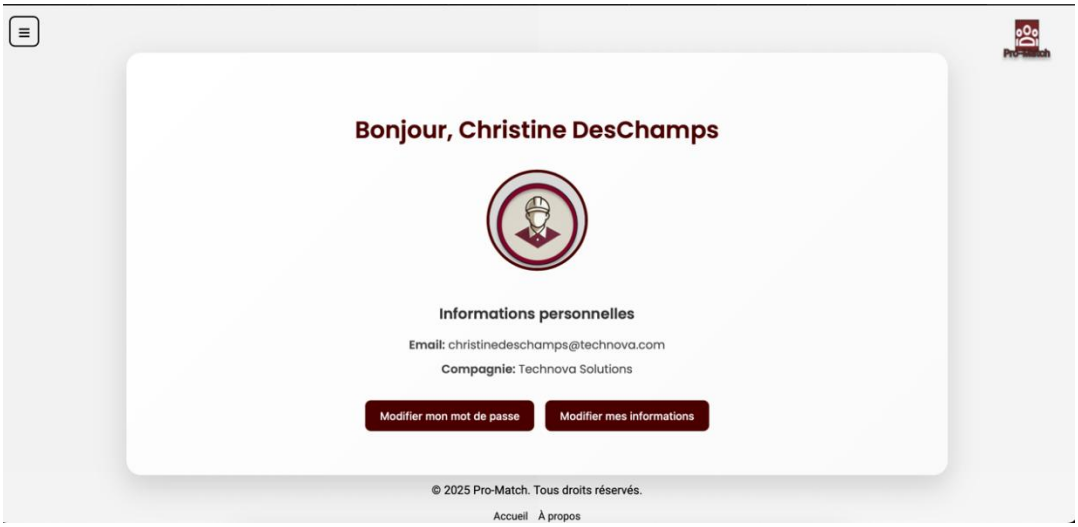
- Se connecter ou s'inscrire,
- Consulter et swiper des offres de stages,
- Postuler aux stages sélectionnés,
- Suivre l'état de leurs candidatures.

Grâce à cette architecture unifiée, notre application web et mobile accèdent en temps réel à la même base de données, ce qui permet une synchronisation efficace entre les deux interfaces.

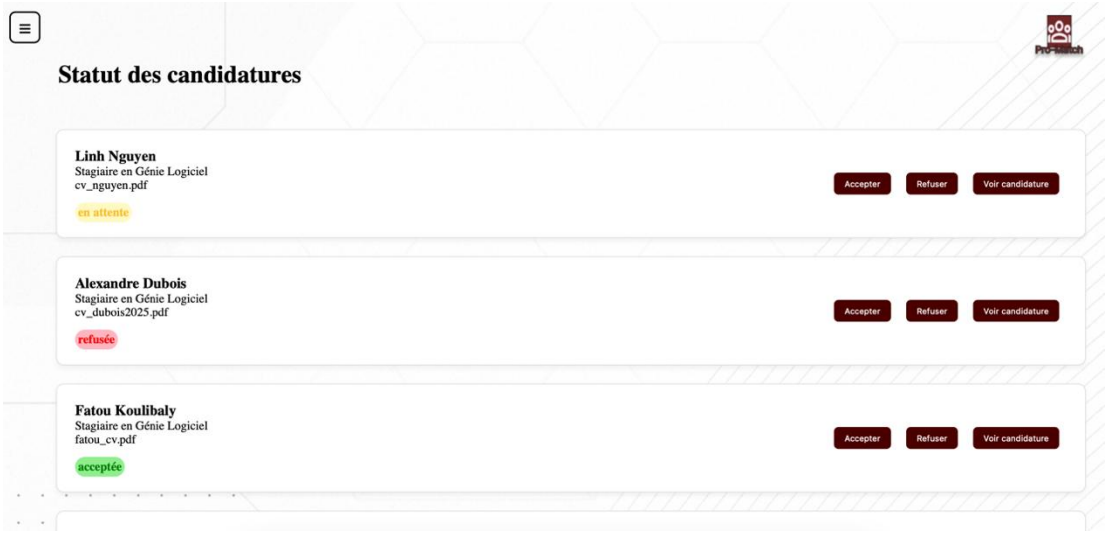
5.1 Pages Web développées

Page de connexion/inscription qui permet à un employeur de s’inscrire, se connecter.

Page de profil qui permet à l’employeur de voir ses informations et les modifier.

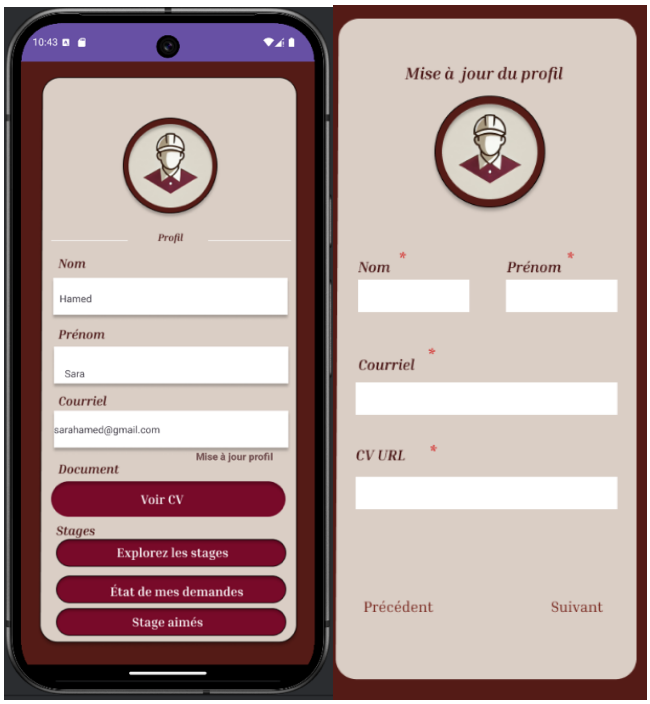


Page de candidature qui permet à l'employeur de voir les candidatures de étudiants à ses stages.

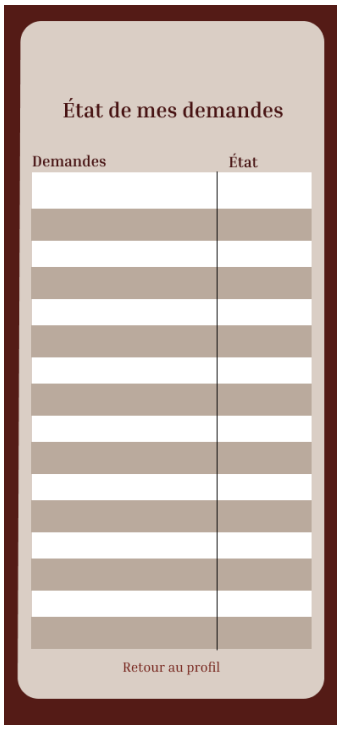


5.2 Pages Mobile développées

Page de profil qui permet à l'étudiant de confirmer ses informations\ Page de mise à jour du profil qui permet à l'étudiant de mettre à jour ses informations



Page d'état de demande qui permet à l'étudiant de consulter la réponse des employeurs



Page de filtre qui permet à l’étudiant de filtrer les stages selon ses préférences et ses critères de recherches