République Tunisienne
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
École Supérieur Privée d'ingénierie et de technologie
TEK-UP

Rapport de Projet de Service Web

Présenté en vue de l'obtention de la

Formation d'Ingénieur en Sciences du Numérique

Spécialité : Génie Logicielle et Système d'Information

Développement D'une Plateforme Freelance

Réalisé par

Kenza Naffeti et Mohamed Amine Riche
et
Aziz Selmi et Soulaymen Omrani

Encadrante académique : Mr. Med Amine Ben Rhouma

TEKUP

Année Universitaire : 2024/2025

Table des matières

Table des figures

chapitre 1

Cadre général du projet

1.1 Introduction

1.1.1 Contexte du projet

Dans un environnement de travail numérique en constante évolution, les freelances deviennent un acteur central du marché du travail. Toutefois, la gestion de leurs informations professionnelles est souvent dispersée (email, WhatsApp), rendant le suivi difficile et non structuré.

1.1.2 Objectif principal

L'objectif de ce projet est de concevoir une plateforme centralisée qui permet aux freelances de créer et gérer leur profil professionnel, et aux RH de superviser ces profils de manière sécurisée et efficace.

1.1.3 Problèmes identifiés

— Échange d'informations par des moyens non structurés (email, WhatsApp)

— Absence de base de données centralisée

— Difficile de trier, filtrer ou rechercher des freelances efficacement

1.1.4 Solution proposée

Développement d'un système web avec back-end en GraphQL + NestJS, supportant la gestion des utilisateurs et profils via un modèle de données structuré, une authentification JWT, et des rôles définis (FREELANCER et RH).

1.2 Analyse du problème

1.2.1 Description des utilisateurs

— **Freelance** : crée et gère son propre profil

— **RH** : visualise tous les profils et peut les modifier ou supprimer

1.2.2 Fonctionnalités principales

— **Création de profil** :
  — Un freelance peut créer un compte et renseigner ses informations personnelles.
  — Il peut lister ses compétences (avec niveaux, domaines...).
  — Il peut ajouter des liens vers ses profils professionnels (LinkedIn, Behance, GitHub, etc.).

— **Mise à jour du profil** :
  — Les freelances peuvent modifier leur profil à tout moment.
  — Ils peuvent ajouter de nouvelles compétences ou mettre à jour leurs expériences.

— **Suppression de profil** :
  — Possibilité pour un freelance de supprimer définitivement son profil.
  — Option de désactivation temporaire du compte si nécessaire.

— **Interface d'administration** :
  — Pour la startup : accès à une interface dédiée.
  — Permet la visualisation, la gestion et le suivi des profils freelances.
  — Inclut des fonctionnalités de modération et de vérification des données.

1.2.3 Contraintes techniques

— Authentification JWT
— Autorisation basée sur les rôles (Guard NestJS)
— Requêtes GraphQL et validation via DTO

1.3 Liste des fonctionnalités GraphQL

N°	Fonction	Rôle	Description
1	`createUser(input)`	Public	Créer un utilisateur (freelance ou RH)
2	`login(input)`	Public	Authentification, retourne un token JWT
3	`me()`	Authentifié	Retourne les informations de l'utilisateur connecté
4	`createProfile(input)`	Freelance	Créer son propre profil
5	`updateProfile(id, input)`	RH / Propriétaire	Mettre à jour un profil
6	`deleteProfile(id)`	RH / Propriétaire	Supprimer un profil
7	`profiles()`	RH uniquement	Voir tous les profils
8	`profile(id)`	RH / Propriétaire	Voir un profil par ID
9	`users()`	RH	Voir tous les utilisateurs
10	`user(id)`	RH	Voir un utilisateur précis

1.4 Architecture du système

1.4.1 Technologies utilisées

— **NestJS** – Framework back-end principal

— **GraphQL** – API pour les requêtes structurées
— **TypeORM** – ORM pour la persistance des données
— **PostgreSQL** – Base de données relationnelle

1.4.2 Organisation du code

Modularisation en 3 modules :

— **User** : création et login d'utilisateurs
— **Profile** : gestion des informations professionnelles
— **Auth** : JWT, Guards, décorateurs personnalisés

1.5 Modélisation des entités

1.5.1 Diagramme de classes

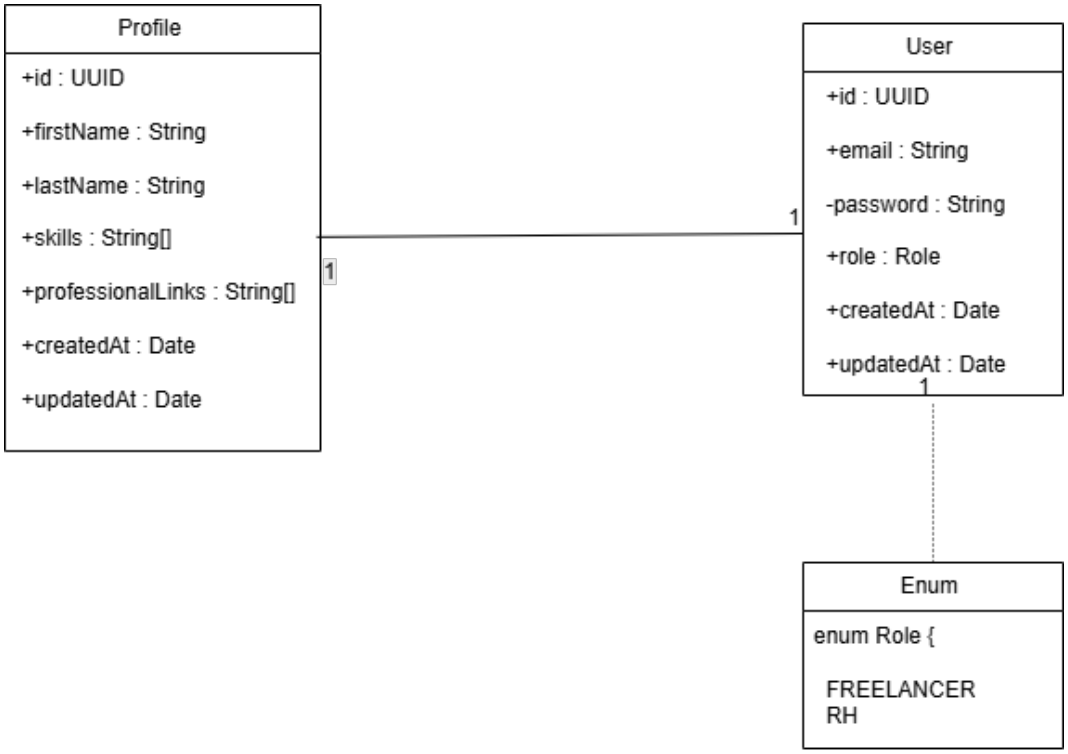

FIGURE 1.1 – Diagramme de classes / entités

1.5.2 Identification des entités et relations

— **User** :
  — Peut être un `FREELANCER` ou un `RH` (type d'utilisateur).
  — Chaque `User` peut avoir un seul `Profile`.
— **Profile** :
  — Appartient à un seul `User`.
  — Contient les données personnelles, les compétences, les liens professionnels, etc.

  - Peut être accepté ou non (`accepted = true/false`), visible uniquement pour les `RH`.

- **Relation entre entités** :
  - `User` **(1)** $\longleftrightarrow$ `(1) Profile`

1.6 API GraphQL : requêtes et mutations

1.6.1 Mutations

- `createUser(input)` : Créer un utilisateur (freelance ou RH)
- `login(input)` : Authentification JWT
- `createProfile(input)` : Créer son profil freelance
- `updateProfile(id, input)` : RH ou propriétaire met à jour
- `deleteProfile(id)` : RH ou propriétaire supprime un profil

1.6.2 Queries

- `me()` : Retourne l'utilisateur connecté
- `users()` : RH peut voir tous les utilisateurs
- `user(id)` : Voir un utilisateur spécifique
- `profiles()` : RH peut voir tous les profils
- `profile(id)` : Voir un profil spécifique (selon rôle)

1.7 Exemples de requêtes et réponses

Exemple : login

Mutation

```
mutation {
  login(input: { email: "freelance@example.com", password: "secret" })
}
```

Réponse

```
{
  "data": {
    "login": "eyJhbGciOiJIUzI1NiIsInR..."
  }
}
```

Exemple : get profiles (RH)

Query

```
query {
  profiles {
    id
    firstName
    skills
    professionalLinks
  }
}
```

Réponse

```
{
  "data": {
    "profiles": [
      {
        "id": "1234",
        "firstName": "Alice",
        "skills": ["NestJS", "GraphQL"],
        "professionalLinks": ["https://github.com/alice"]
      }
    ]
  }
}
```

1.8 Sécurité et gestion des rôles

— Authentification par JWT à chaque requête
— Accès protégé par '@UseGuards(JwtAuthGuard)'
— Vérification du rôle dans les services
— RH peut voir / modifier tous les profils

1.9 Tests et validations

— Création de profil avec user non authentifié ⇒ rejetée
— Freelance tente d'accéder à un autre profil ⇒ rejetée
— RH peut voir tous les utilisateurs
— Jeton expiré ⇒ accès interdit

1.10 Améliorations futures

— Ajout d'une interface front-end
— Recherche multicritère de profils
— Intégration LinkedIn/GitHub API
— Export PDF du profil

1.11 Conclusion

Le projet permet la gestion complète des profils freelances via une API moderne, sécurisée et modulable. Il améliore le suivi et centralise les informations de manière fiable.

Conclusion Générale

Au terme de ce travail, nous avons conçu et développé une plateforme e-learning gamifiée qui répond aux besoins pédagogiques actuels en combinant apprentissage, interactivité et suivi personnalisé. En adoptant une méthodologie agile basée sur des sprints successifs, nous avons pu structurer efficacement le projet et assurer une évolution progressive et maîtrisée des fonctionnalités.

Chaque sprint a apporté une contribution essentielle à l'architecture globale : de la gestion des utilisateurs à l'intégration des cours et évaluations, en passant par la gamification et la visualisation des statistiques. Les choix technologiques, notamment ASP.NET Core pour le backend et Angular pour le frontend, ont permis de garantir performance, modularité et maintenabilité de l'application.

Ce projet nous a permis de mettre en pratique des concepts théoriques tout en développant des compétences techniques, organisationnelles et méthodologiques, essentielles pour la conduite de projets logiciels à grande échelle.