**Rapport d'Avancement de Projet**

**Initialisation du Projet Node.js et React.js**

Pour initialiser le projet, j'ai utilisé Node.js avec React.js comme framework frontend. Voici les étapes que j'ai suivies :

**Initialisation du Projet Node.js :**

J'ai utilisé la commande **npm init** pour créer un nouveau projet Node.js.

Cette commande génère un fichier package.json qui contient les métadonnées du projet.

**Initialisation du Projet React.js :**

J'ai utilisé la **commande npx create-react-app** pour créer une nouvelle application React.

Cette commande génère une structure de projet React de base avec les dépendances nécessaires.

**Installation des Dépendances des Frameworks**

Après l'initialisation, j'ai installé les dépendances des frameworks suivants :

**Next.js :**

Commande d'installation : **npm install next**

Utilité : Next.js est un framework React.js qui facilite la création d'applications web évolutives, en offrant des fonctionnalités telles que le rendu côté serveur, le routage dynamique, et la génération de pages statiques.

**Prisma** :

Commande d'installation : **npm install @prisma/client**

qui facilite l'accès à la base de données et la manipulation des données. Il permet une interaction facile et sécurisée avec la base de données MySQL.

**TypeScript** :

Commande d'installation : **npm install typescript**

**Tailwind** **CSS** :

Commande d'installation : **npm install tailwindcss**

Utilité : Tailwind CSS est un framework CSS utilitaire qui permet de créer des

**Génération du Schéma de la Base de Données avec Prisma**

J'ai utilisé Prisma pour générer le schéma de la base de données. Voici les étapes suivies :

**Définition du Schéma Prisma :**

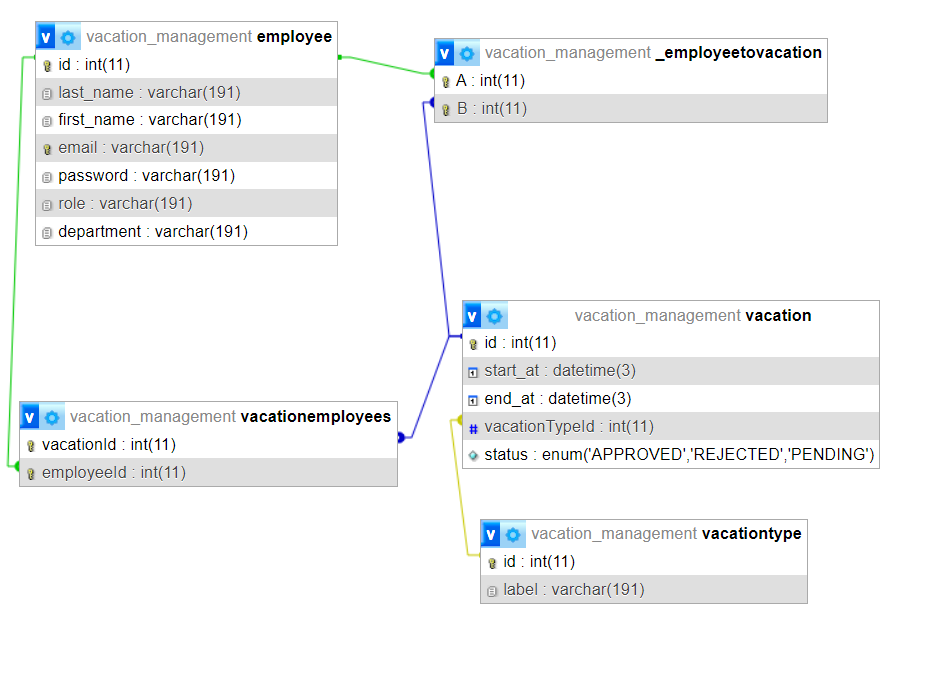
J'ai défini les modèles de données pour les employés, les congés et les types de congés dans des fichiers .prisma.

J ai aussi spécifier les relations entre les tables

**Génération de la Base de Données :**

J'ai exécuté la commande **npx prisma db push** pour générer la base de données à partir du schéma défini.

Cette commande crée les tables et les relations spécifiées dans le schéma Prisma dans la base de données MySQL.



**Développement de la Fonctionnalité de l employee**

Pour commencer le développement de l'application, j'ai implémenté le **CRUD** pour l’employé.

**Création de la Fonction createEmployee :**

J'ai créé une fonction ***createEmployee***.

Cette fonction reçoit les données de l'employé via une requête HTTP et les enregistre dans la base de données en utilisant Prisma.

**Fonctionnalité de Suppression d'Employé :**

Pour permettre la suppression d'un employé avec la fonction ***deleteEmployee***

**Fonctionnalité de Récupération d'Employé :**

Pour récupérer les détails d'un employé spécifique, j'ai créé une fonction ***getEmployee***

**Fonctionnalité de Récupération de la Liste des Employés :**

Pour récupérer la liste complète des employés avec la fonction ***getEmployees***

**Fonctionnalité de Mise à Jour d'Employé :**

Pour mettre à jour les informations d'un employé la fonction ***updateEmployee***

***Et j’ai aussi fait le CRUD pour vacations et vacationTypes***