Pour réussir ce déploiement, nous devons adapter légèrement le code pour qu'il fonctionne en production (fini le localhost), configurer le serveur et lier les deux parties avec un "Reverse Proxy" (Nginx).

Étape 1 : Préparation du Code Backend (NestJS)

Actuellement, votre backend écoute à la racine. Pour faciliter la configuration du serveur (Nginx), il est vivement conseillé d'ajouter un préfixe global (ex: /api) à toutes vos routes backend.

1. Modifier GestionAbsenceBack/src/main.ts

Ajoutez app.setGlobalPrefix('api'); avant le SwaggerModule et activez le CORS pour la production.

TypeScript

```
// Fichier: GestionAbsenceBack/src/main.ts

import { NestFactory } from '@nestjs/core';

import { AppModule } from './app.module';

import { SwaggerModule, DocumentBuilder } from '@nestjs/swagger';


async function bootstrap() {

 const app = await NestFactory.create(AppModule);


 // 1. Activer CORS pour accepter les requêtes du frontend

 app.enableCors();


 // 2. Ajouter un préfixe global pour distinguer l'API du Frontend sur le même serveur

 app.setGlobalPrefix('api');


 const config = new DocumentBuilder()

  .setTitle('API de gestion des absences')

  .setDescription('Cette API permet de gérer les absences de la licence MIASHS')

  .setVersion('1.0')

  // Le tag ne change pas, mais l'accès sera /api/student, etc.

  .addTag('Etudiant')

  .build();


 const document = SwaggerModule.createDocument(app, config);
```

SwaggerModule.setup('docs', app, document); // J'ai renommé 'api' en 'docs' pour éviter la confusion avec le préfixe

// Utiliser le port de l'environnement ou 3000 par défaut

await app.listen(process.env.PORT || 3000);

}

bootstrap();

Étape 2 : Préparation du Code Frontend (Vue.js)

Vos fichiers de fetch (comme students.js) contiennent des URLs "en dur" (http://localhost:3000). Cela ne fonctionnera pas sur le serveur.

1. Configurer l'URL de base

La méthode la plus propre avec Vite est d'utiliser une variable d'environnement ou un chemin relatif si le backend et le frontend sont sur le même domaine (ce qui sera le cas avec Nginx).

Option recommandée (Chemin relatif) :

Remplacez toutes les occurrences de http://localhost:3000 par /api dans vos fichiers src/shared/fetchers/*.js.

Exemple de modification pour students.js :

JavaScript

// AVANT

const response = await fetch("http://localhost:3000/student", { ... });

// APRÈS (Le navigateur complétera automatiquement avec le domaine du site)

const response = await fetch("/api/student", { ... });

Note: Faites cela pour tous les fetchers (groups.js, slots.js, etc.).

2. Créer le build de production

Sur votre machine locale (ou sur le serveur si vous envoyez le code source), générez les fichiers statiques :

Bash

cd GestionAbsenceFront

npm install

npm run build

Cela va créer un dossier dist/ contenant votre site web prêt à être déployé.

Étape 3 : Installation sur le Serveur Universitaire

Connectez-vous à votre serveur via SSH.

1. Installer les pré-requis (Node.js, PM2, Nginx)

Bash

Mettre à jour le système

sudo apt update && sudo apt upgrade -y

Installer Node.js (Version 20 ou 22 recommandée)

curl -fsSL https://deb.nodesource.com/setup_20.x | sudo -E bash -

sudo apt install -y nodejs

Installer PM2 (Pour garder le backend allumé en permanence)

sudo npm install -g pm2

Installer Nginx (Serveur web et reverse proxy)

sudo apt install -y nginx

2. Installer et Configurer le Backend

Copiez votre dossier GestionAbsenceBack sur le serveur (par exemple dans /var/www/gestion-absence/back).

Bash

cd /var/www/gestion-absence/back

Installer les dépendances

npm install

Générer le client Prisma

npx prisma generate

Créer le fichier de production .env

nano .env

Dans ce fichier .env, mettez :

Extrait de code

PORT=3000

DATABASE_URL="file:./dev.db"

Ou un chemin absolu pour être sûr : "file:/var/www/gestion-absence/back/production.db"

JWT_SECRET="votre_secret_super_securise"

3. Initialiser la Base de Données

Puisque vous utilisez SQLite, assurez-vous que le fichier de base de données est créé et que les migrations sont appliquées.

Bash

npx prisma migrate deploy

4. Démarrer le Backend avec PM2

Bash

npm run build

pm2 start dist/main.js --name "gestion-absence-api"

pm2 save

pm2 startup

Étape 4 : Déployer le Frontend

Copiez le dossier dist (généré à l'étape 2) vers le serveur, par exemple dans /var/www/gestion-absence/front.

Assurez-vous que les permissions sont correctes :

Bash

sudo chown -R www-data:www-data /var/www/gestion-absence/front

---

Étape 5 : Configuration Finale (Nginx)

C'est ici que nous lions le tout. Nginx va servir les fichiers du Frontend (Vue.js) et rediriger les requêtes commençant par /api vers votre Backend (NestJS).

Créez un fichier de configuration Nginx :

Bash

sudo nano /etc/nginx/sites-available/gestion-absence

Collez la configuration suivante (adaptez server_name avec votre domaine ou IP) :

Nginx

```
server {
    listen 80;
    server_name votre-domaine.univ.fr;  # Ou l'IP du serveur

    # 1. Servir le Frontend (Vue.js)
    location / {
        root /var/www/gestion-absence/front;
        index index.html;
        try_files $uri $uri/ /index.html; # Important pour le routage Vue.js (SPA)
    }

    # 2. Rediriger l'API vers NestJS
    location /api/ {
        proxy_pass http://localhost:3000/api/;  # Redirige vers le backend
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_cache_bypass $http_upgrade;
    }
}
```

Activez le site et redémarrez Nginx :

Bash

```
sudo ln -s /etc/nginx/sites-available/gestion-absence /etc/nginx/sites-enabled/
sudo nginx -t # Vérifie s'il y a des erreurs de syntaxe
sudo systemctl restart nginx
```

Résumé

1. Le backend tourne sur le port 3000 (géré par PM2).
2. Le code frontend ne contient plus de localhost, mais utilise des chemins relatifs /api/....
3. Nginx écoute sur le port 80 (Web).

- Si l'utilisateur demande /, il reçoit l'app Vue.js.
- Si l'app Vue.js demande /api/student, Nginx transmet à localhost:3000/api/student.

Votre application est maintenant accessible sur http://votre-domaine.univ.fr.