FICHE PROCÉDURE PROJET HELPDESK

FICHE PROCÉDURE PROJET HELPDESK

📄 Objectif du projet

Créer une application web HelpDesk qui permet à des utilisateurs de créer et suivre des tickets de support, et aux administrateurs de les gérer (modifier, changer le statut, supprimer, etc.). Le projet comporte un espace utilisateur simple et un panneau d'administration protégé par mot de passe.

🔧 Technologies utilisées

- Frontend : React (Vite), React Router DOM

- Backend : Express.js, Node.js

- Base de données : MySQL

- Styles : CSS classique

✅ Étapes du développement

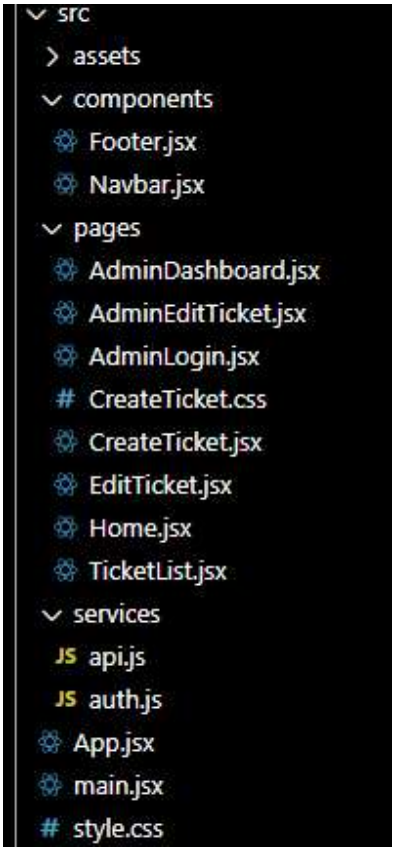
1. Initialisation du projet avec Vite

- Création d'un projet Vite + React (npm create vite@latest)

- Installation des dépendances :

```
PS C:\Users\Administrateur\Desktop\helpdesk-backend\helpdesk-frontend>   npm install react-router-dom axios
npm WARN EBADENGINE Unsupported engine {
npm WARN EBADENGINE   package: 'vite@6.2.5',
npm WARN EBADENGINE   required: { node: '^18.0.0 || ^20.0.0 || >=22.0.0' },
npm WARN EBADENGINE   current: { node: 'v21.7.1', npm: '10.5.0' }
npm WARN EBADENGINE }

removed 6 packages, changed 3 packages, and audited 179 packages in 7s

36 packages are looking for funding
  run `npm fund` for details

1 moderate severity vulnerability

To address all issues, run:
  npm audit fix

Run `npm audit` for details.
```

2. Structure du projet frontend



3. Création du système utilisateur

- Home.jsx : page d'accueil avec liens vers la création et la liste des tickets.

- CreateTicket.jsx : formulaire pour créer un ticket (titre, description, priorité).

- EditTicket.jsx : permet de modifier uniquement son propre ticket (statut non visible).

- TicketList.jsx : liste tous les tickets avec boutons Modifier/Supprimer.

4. Navigation utilisateur

- Mise en place de Navbar.jsx + Footer.jsx et intégration globale dans App.jsx.

- Utilisation de <Outlet /> dans App.jsx pour les routes imbriquées.

5. Connexion backend (Express + MySQL)

- Routes backend :

  - POST /tickets

  - GET /tickets et /tickets/:id

  - PUT /tickets/:id

  - DELETE /tickets/:id

- Stockage dans une table tickets avec les colonnes :

  id, title, description, priority, status, created_at (timestamp par défaut)

6. Sécurisation du côté admin

- Page AdminLogin.jsx avec mot de passe stocké en .env :

  VITE_ADMIN_PASSWORD=superadmin

- Enregistrement dans localStorage de isAdmin = true

- Utilitaire auth.js pour vérifier la connexion

- Protection des routes admin avec un composant AdminRoute

7. Tableau de bord administrateur

- AdminDashboard.jsx : tableau listant tous les tickets avec :

  - Filtres par statut

  - Boutons Modifier / Supprimer

- AdminEditTicket.jsx : formulaire pour éditer titre, description, priorité et statut

8. Ajout du Footer sur toutes les pages

- Intégration dans App.jsx

- Intégration manuelle dans toutes les pages admin (layout avec flexbox)

9. Améliorations apportées

- Filtrage dynamique par statut dans l'espace admin

- Affichage conditionnel du lien Admin / Déconnexion dans Navbar

- Mise en page adaptée pour éviter les grandes zones blanches (admin)

🔐 Accès admin (local)

- URL : /admin/login

- Mot de passe : défini dans .env via VITE_ADMIN_PASSWORD

📈 Fonctionnalités à venir (suggestions)

- Recherche par mot-clé

- Tri par priorité ou date

- Attribution d'agents (champ assigned_to)

- Export CSV ou PDF

- Système de notification

- Authentification admin connectée à une base de données

🎯 Conclusion

Ce projet HelpDesk offre une base solide pour la gestion de support interne. Il s'adapte facilement à des besoins plus complexes tout en respectant une séparation claire entre utilisateur et administrateur. Idéal pour un déploiement en intranet ou PME.