Voici la **liste complète des fichiers à développer**, classés par **dossier** et **ordre logique** (API → pages → composants → utilitaires).

**✅ 1. ROUTES API (Back-end)**

📁 app/api/projects/

* route.ts : GET (liste projets), POST (création projet)
* [projectId]/route.ts : PUT (édition), DELETE (suppression)

📁 app/api/files/

* route.ts : GET (liste fichiers par projectId), POST (upload fichier)
* [fileId]/route.ts : PUT, DELETE

📁 app/api/features/

* route.ts : GET (liste features par projectId), POST (création feature)
* [featureId]/route.ts : PUT, DELETE

📁 app/api/schema-fields/

* route.ts : GET (par projectId), POST (création field)
* [fieldId]/route.ts : PUT, DELETE

**✅ 2. PAGES UTILISATEUR (Front-end)**

📁 app/projects/

* page.tsx : page de liste des projets + bouton de création
* [projectId]/page.tsx : page de détails projet avec :
  + onglet Fichiers
  + onglet Features
  + onglet SchemaFields

📁 app/files/ *(optionnel si contenu dans la page projet)*

* [fileId]/page.tsx : page de détails ou aperçu d’un fichier

📁 app/features/ *(optionnel si contenu dans la page projet)*

* [featureId]/page.tsx : page de détails/édition d’une feature

📁 app/schema/ *(optionnel si contenu dans la page projet)*

* [schemaId]/page.tsx : page de détails ou aperçu de champ

**✅ 3. COMPOSANTS UI & MÉTIER**

📁 components/models/

* ProjectForm.tsx : formulaire de création/modification projet
* ProjectList.tsx : tableau ou cards des projets
* FilesForm.tsx : formulaire d’upload fichier
* FilesList.tsx : liste des fichiers (filtrée par projet)
* FeaturesForm.tsx : formulaire de création/modification feature
* FeaturesList.tsx : liste des features
* SchemaFieldsForm.tsx : formulaire de création champ de schéma
* SchemaFieldsList.tsx : liste hiérarchique des fields

📁 components/layout/

* Header.tsx, Footer.tsx, Sidebar.tsx (si navigation latérale)

**✅ 4. UTILS / LIB / STORES / HOOKS**

📁 lib/

* lib/prisma.ts : accès base de données
* lib/auth/auth-client.ts : gestion session (si auth)
* lib/utils.ts : fonctions helpers

📁 utils/

* getProjects.tsx
* getFiles.tsx
* getFeatures.tsx
* getSchemaFields.tsx
* uploadFile.tsx

📁 stores/ (si gestion état local/global via Zustand ou autre)

* useProjectStore.ts
* useFilesStore.ts *(optionnel)*
* useFeaturesStore.ts *(optionnel)*

📁 hooks/

* use-toast.ts : pour feedback utilisateur

Liste des fichiers de app-builder

# Les routes

## auth

### app/api/auth/[...all]/route.ts (DONE)

Description : Route catch-all pour l'authentification

Fonctionnalités : Login, logout, register, reset password, verify email

Utilise : lib/auth/auth-server.ts, lib/prisma.ts

Utilisé par : Toutes les pages d'authentification

Export : { GET, POST}

## Project

### app/api/projects/route.ts (DONE)

Description : CRUD des projets utilisateur

Fonctionnalités : GET (liste filtrée), POST (création)

Utilise : lib/prisma.ts, lib/auth/auth-server.ts, validators/projectSchemas.ts

Utilisé par : app/projects/page.tsx, components/models/ProjectList.tsx, stores/useProjectsStore.ts

Export : { GET, POST }

### app/api/projects/[projectId]/route.ts (DONE)

Description : Gestion d'un projet spécifique

Fonctionnalités : GET (détails), PUT (mise à jour), DELETE (suppression)

Utilise : lib/prisma.ts, lib/auth/auth-server.ts, validators/projectSchemas.ts

Utilisé par : app/projects/[projectId]/page.tsx, components/models/ProjectCard.tsx, stores/useProjectStore.ts

Export : { GET, PUT, DELETE }

## roadmap

### app/api/roadmap/route.ts

Description : CRUD des roadmaps

Fonctionnalités : GET (par projet avec phases), POST (création)

Utilise : lib/prisma.ts, validators/roadmapSchemas.ts, lib/auth/auth-server.ts

Utilisé par : app/roadmap/page.tsx, components/models/RoadmapList.tsx, stores/useRoadmapStore.ts

Export : { GET, POST }

### app/api/roadmap/[roadmapId]/route.ts

Description : Gestion d'une roadmap spécifique

Fonctionnalités : GET (détails), PUT (mise à jour), DELETE (suppression)

Utilise : lib/prisma.ts, validators/roadmapSchemas.ts, lib/auth/auth-server.ts

Utilisé par : app/roadmap/[roadmapId]/page.tsx, components/models/RoadmapForm.tsx

Export : { GET, PUT, DELETE }

## features

### app/api/features/route.ts

Description : CRUD des features

Fonctionnalités : GET (hiérarchie par projet), POST (création)

Utilise : lib/prisma.ts, validators/featureSchemas.ts, lib/auth/auth-server.ts

Utilisé par : app/features/page.tsx, components/models/FeaturesList.tsx, stores/useFeaturesStore.ts

Export : { GET, POST }

### app/api/features/ [featuresId]/route.ts

Description : Gestion d'une feature spécifique

Fonctionnalités : GET (détails), PUT (mise à jour), DELETE (suppression)

Utilise : lib/prisma.ts, validators/featureSchemas.ts, lib/auth/auth-server.ts

Utilisé par : app/features/[featureId]/page.tsx, components/models/FeaturesForm.tsx

Export : { GET, PUT, DELETE }

## tasks

### app/api/tasks/route.ts

Description : CRUD des tâches

Fonctionnalités : GET (avec filtres), POST (création avec assignation)

Utilise : lib/prisma.ts, validators/taskSchemas.ts, lib/auth/auth-server.ts

Utilisé par : app/tasks/page.tsx, components/models/TaskList.tsx, stores/useTasksStore.ts

Export : { GET, POST }

### app/api/tasks/[taskId]/route.ts

Description : Gestion d'une tâche spécifique

Fonctionnalités : GET (détails), PUT (mise à jour), DELETE (suppression)

Utilise : lib/prisma.ts, validators/taskSchemas.ts, lib/auth/auth-server.ts

Utilisé par : app/tasks/[taskId]/page.tsx, components/models/TaskForm.tsx

Export : { GET, PUT, DELETE }

## files

### app/api/files/route.ts

Description : Gestion des fichiers

Fonctionnalités : GET (liste par projet), POST (upload)

Utilise : lib/prisma.ts, lib/upload.ts, lib/auth/auth-server.ts

Utilisé par : app/files/page.tsx, components/models/FilesList.tsx, utils/uploadFile.tsx

Export : { GET, POST }

### app/api/files/[fileId]/route.ts (DONE)

Description : Gestion d'un fichier spécifique

Fonctionnalités : GET (détails), PUT (mise à jour), DELETE (suppression)

Utilise : lib/prisma.ts, lib/upload.ts, lib/auth/auth-server.ts

Utilisé par : app/files/[fileId]/page.tsx, components/models/FilesForm.tsx

Export : { GET, PUT, DELETE }

## User stories

### app/api/user-stories/route.ts

Description : CRUD des user stories

Fonctionnalités : GET (par projet avec hiérarchie), POST (création)

Utilise : lib/prisma.ts, validators/userStorySchemas.ts, lib/auth/auth-server.ts

Utilisé par : app/user-stories/page.tsx, components/models/UserStoryList.tsx, stores/useUserStoriesStore.ts

Export : { GET, POST }

### app/api/user-stories/[storyId]/route.ts

Description : Gestion d'une user story spécifique

Fonctionnalités : GET (détails), PUT (mise à jour), DELETE (suppression)

Utilise : lib/prisma.ts, validators/userStorySchemas.ts, lib/auth/auth-server.ts

Utilisé par : app/user-stories/[storyId]/page.tsx, components/models/UserStoryForm.tsx

Export : { GET, PUT, DELETE }

## sprints

### app/api/sprints/route.ts

Description : Gestion d'un sprint spécifique

Fonctionnalités : GET (détails), PUT (mise à jour), DELETE (suppression)

Utilise : lib/prisma.ts, validators/sprintSchemas.ts, lib/auth/auth-server.ts

Utilisé par : app/sprints/[sprintId]/page.tsx, components/models/SprintForm.tsx

Export : { GET, PUT, DELETE }

### app/api/sprints/[sprintId]/route.ts

Description : Gestion d'un sprint spécifique

Fonctionnalités : GET (détails), PUT (mise à jour), DELETE (suppression)

Utilise : lib/prisma.ts, validators/sprintSchemas.ts, lib/auth/auth-server.ts

Utilisé par : app/sprints/[sprintId]/page.tsx, components/models/SprintForm.tsx

Export : { GET, PUT, DELETE }

## shema

### app/api/schema-fields/route.ts

Description : CRUD des schémas de champs

Fonctionnalités : GET (par projet), POST (création)

Utilise : lib/prisma.ts, validators/schemaSchemas.ts, lib/auth/auth-server.ts

Utilisé par : app/schema/page.tsx, components/models/SchemaList.tsx, stores/useSchemaStore.ts

Export : { GET, POST }

### app/api/schema-fields/[fieldId]/route.ts

Description : Gestion d'un champ spécifique

Fonctionnalités : GET (détails), PUT (mise à jour), DELETE (suppression)

Utilise : lib/prisma.ts, validators/schemaSchemas.ts, lib/auth/auth-server.ts

Utilisé par : app/schema/[fieldId]/page.tsx, components/models/SchemaForm.tsx

Export : { GET, PUT, DELETE }

## teams

### app/api/teams/route.ts

Description : CRUD des équipes

Fonctionnalités : GET (liste avec membres), POST (création)

Utilise : lib/prisma.ts, validators/teamSchemas.ts, lib/auth/auth-server.ts

Utilisé par : app/teams/page.tsx, components/models/TeamList.tsx, stores/useTeamsStore.ts

Export : { GET, POST }

### app/api/teams/[teamId]/route.ts

Description : Gestion d'une équipe spécifique

Fonctionnalités : GET (détails), PUT (mise à jour), DELETE (suppression)

Utilise : lib/prisma.ts, validators/teamSchemas.ts, lib/auth/auth-server.ts

Utilisé par : app/teams/[teamId]/page.tsx, components/models/TeamForm.tsx

Export : { GET, PUT, DELETE }

## Time log

### app/api/time-logs/route.ts

Description : CRUD des logs de temps

Fonctionnalités : GET (statistiques), POST (enregistrement)

Utilise : lib/prisma.ts, validators/timeLogSchemas.ts, lib/auth/auth-server.ts

Utilisé par : app/roadmap/timelog/page.tsx, components/timelog/TimeLogList.tsx, utils/getTimeLogs.tsx

Export : { GET, POST }

### app/api/time-logs/[logId]/route.ts

Description : Gestion d'un log spécifique

Fonctionnalités : GET (détails), PUT (mise à jour), DELETE (suppression)

Utilise : lib/prisma.ts, validators/timeLogSchemas.ts, lib/auth/auth-server.ts

Utilisé par : components/timelog/TimeLogForm.tsx

Export : { GET, PUT, DELETE }

## comments

### app/api/comments/route.ts

Description : CRUD des commentaires

Fonctionnalités : GET (par entité), POST (création)

Utilise : lib/prisma.ts, validators/commentSchemas.ts, lib/auth/auth-server.ts

Utilisé par : app/comments/page.tsx, components/models/CommentList.tsx, stores/useCommentsStore.ts

Export : { GET, POST }

### app/api/comments/[commentId]/route.ts

Description : Gestion d'un commentaire spécifique

Fonctionnalités : GET (détails), PUT (mise à jour), DELETE (suppression)

Utilise : lib/prisma.ts, validators/commentSchemas.ts, lib/auth/auth-server.ts

Utilisé par : app/comments/[commentId]/page.tsx, components/models/CommentForm.tsx

Export : { GET, PUT, DELETE }

## user

### app/api/users/route.ts

Description : CRUD des utilisateurs (admin)

Fonctionnalités : GET (liste filtrée), POST (création admin)

Utilise : lib/prisma.ts, validators/userSchemas.ts, lib/auth/auth-server.ts

Utilisé par : app/admin/users/page.tsx, components/admin/UserList.tsx

Export : { GET, POST }

### app/api/users/[userId]/route.ts

Description : Gestion d'un utilisateur spécifique

Fonctionnalités : GET (profil), PUT (mise à jour), DELETE (admin)

Utilise : lib/prisma.ts, validators/userSchemas.ts, lib/auth/auth-server.ts

Utilisé par : app/admin/users/[userId]/page.tsx, app/user/profile/page.tsx

Export : { GET, PUT, DELETE }

## Activités

### app/api/activity-logs/route.ts

Description : Logs d'activité système

Fonctionnalités : GET (historique filtré), POST (création automatique)

Utilise : lib/prisma.ts, lib/auth/auth-server.ts

Utilisé par : app/activitylogs/page.tsx, app/admin/activity-logs/page.tsx, utils/getActivityLogs.tsx

Export : { GET, POST }

# Les pages

## app

### app/layout.tsx

Description : Layout racine avec providers

Fonctionnalités : Metadata, fonts, providers Auth/Zustand/Theme, navigation

Utilise : components/layout/Header.tsx, components/layout/Footer.tsx, components/providers/Providers.tsx, lib/auth/auth-client.ts

Utilisé par : Toutes les pages

Export : default function RootLayout

### App/page.tsx

Description : Dashboard principal/Landing page

Fonctionnalités : Vue d'ensemble, statistiques, actions rapides

Utilise : components/dashboard/DashboardStats.tsx, stores/useProjectsStore.ts, utils/getProjects.tsx

Utilisé par : Navigation principale

Export : default function HomePage

### App/loading.tsx

Description : Composant de chargement global

Fonctionnalités : Skeleton, spinner, état de chargement

Utilise : components/ui/skeleton.tsx

Utilisé par : Next.js automatiquement

Export : default function Loading

### not-found.tsx

Description : Page 404 personnalisée

Fonctionnalités : Message d'erreur, navigation retour

Utilise : components/ui/button.tsx

Utilisé par : Next.js automatiquement

Export : default function NotFound

## auth

### app/auth/forgot-password/page.tsx

Description :

Fonctionnalités :

Utilise :

Export :

### app/auth/forgot-password/page.tsx

Description : Demande de reset password

Fonctionnalités : Formulaire email, envoi lien reset

Utilise : lib/auth/auth-client.ts, validators/authSchemas.ts

Utilisé par : Page sign-in

Export : default function ForgotPasswordPage

### app/auth/reset-password/page.tsx

Description : Reset password avec token

Fonctionnalités : Validation token, nouveau mot de passe

Utilise : lib/auth/auth-client.ts, validators/authSchemas.ts

Utilisé par : Email de reset

Export : default function ResetPasswordPage

### app/auth/goodbye/page.tsx

Description :

Fonctionnalités :

Utilise :

Export :

### app/auth/sign-in/page.tsx

Description : Page de connexion

Fonctionnalités : Formulaire login, validation, redirection vers User/hello/page.tsx

Utilise : lib/auth/auth-client.ts, validators/authSchemas.ts, components/auth/SignInForm.tsx

Utilisé par : Navigation, redirections

Export : default function SignInPage

### app/auth/sign-up/page.tsx

Description : Page d'inscription

Fonctionnalités : Formulaire register, validation email

Utilise : lib/auth/auth-client.ts, validators/authSchemas.ts, components/auth/SignUpForm.tsx

Utilisé par : Page sign-in, navigation

Export : default function SignUpPage

### app/auth/verify-email/page.tsx

Description : Page de déconnexion/suppression

Fonctionnalités : Message de confirmation, nettoyage session

Utilise : lib/auth/auth-client.ts

Utilisé par : Actions de déconnexion

Export : default function GoodbyePage

## User

### User/hello/page.tsx

Description : Page d'accueil utilisateur connecté

Fonctionnalités : Informations de base, actions rapides

Utilise : lib/auth/auth-client.ts, stores/useAuthStore.ts

Utilisé par : Navigation utilisateur

Export : default function HelloPage

### User/profile/page.tsx

Description : Profil utilisateur détaillé

Fonctionnalités : Modification infos, paramètres, historique

Utilise : lib/auth/auth-client.ts, components/user/ProfileForm.tsx, app/api/users/[userId]/route.ts

Utilisé par : Menu utilisateur

Export : default function ProfilePageExport :

### User/dashboard

Description : Dashboard personnel utilisateur

Fonctionnalités : Mes tâches, mes projets, statistiques

Utilise : stores/useTasksStore.ts, stores/useProjectsStore.ts, components/dashboard/UserStats.tsx

Utilisé par : Navigation utilisateur

Export : default function UserDashboardPage

## comments

### app/comments/page.tsx

Description : Vue globale des commentaires

Fonctionnalités : Commentaires récents, par entité

Utilise : components/models/CommentList.tsx, stores/useCommentsStore.ts, utils/getComments.tsx

Utilisé par : Navigation, notifications

Export : default function CommentsPage

### app/comments/[commentId]page.tsx

Description : Détails d'un commentaire

Fonctionnalités : Thread de discussion, réponses

Utilise : components/models/CommentForm.tsx, stores/useCommentsStore.ts, app/api/comments/[commentId]/route.ts

Utilisé par : Liste commentaires

Export : default function CommentDetailPage

## Dev

### app/dev/architecture/page.tsx

Description :

Fonctionnalités :

Utilise :

Export :

### app/dev/link/page.tsx

Description :

Fonctionnalités :

Utilise :

Export :

### app/dev/roadmap/pages.tsx

Description :

Fonctionnalités :

Utilise :

Export :

### app/dev/schema/page.tsx

Description :

Fonctionnalités :

Utilise :

Export :

## Projects

### app/projects/page.tsx (Donne)

Description : Liste de tous les projets et un bouton « ajouter un projet »

Fonctionnalités :

🔍 Filtres par statut / recherche

👁️ Vue grille ou liste

🧰 permet de delectionner un projet en mettant à jour useProjectStore

➕ Création projet avec modale

♻️ Rafraîchissement via SWR

Utilise :

ProjectList.tsx, ProjectFilters.tsx, ProjectForm.tsx, Zustand (useProjectsStore), use-Modal.ts

Utilisé par : Navigation principale (HeaderNav), dashboard

Export : default function ProjectsPage

### app/projects/[projectId]/page.tsx

Description : Vue détaillée d'un projet

Fonctionnalités : Infos projet, équipe, métriques, navigation sous-sections

Utilise : components/models/ProjectCard.tsx, utils/getProjects.tsx, stores/useProjectStore.ts

Utilisé par : Liste projets, navigation

Export : default function ProjectDetailPage

## features

### app/ features/page.tsx

Description : Liste des features d'un projet

Fonctionnalités : Vue hiérarchique, création/édition, drag&drop

Utilise : components/models/FeaturesList.tsx, stores/useFeaturesStore.ts, utils/getFeatures.tsx

Utilisé par : Navigation projet

Export : default function FeaturesPage

### app//features/[ features]/page.tsx

Description : Détails d'une feature spécifique

Fonctionnalités : Infos feature, sous-features, user stories liées

Utilise : components/models/FeaturesForm.tsx, stores/useFeaturesStore.ts, app/api/features/[featureId]/route.ts

Utilisé par : Liste features

Export : default function FeatureDetailPage

### app/user-stories/page.tsx

Description : Liste des user stories

Fonctionnalités : Affichage par epic, priorisation, estimation

Utilise : components/models/UserStoryList.tsx, stores/useUserStoriesStore.ts, utils/getUserStories.tsx

Utilisé par : Navigation projet

Export : default function UserStoriesPage

### App/user-stories/[storyId]/page.tsx

Description : Détails d'une user story

Fonctionnalités : Critères d'acceptation, tâches, commentaires

Utilise : components/models/UserStoryForm.tsx, stores/useUserStoriesStore.ts, app/api/user-stories/[storyId]/route.ts

Utilisé par : Liste user stories

Export : default function UserStoryDetailPage

### app/sprints/page.tsx

Description : Gestion des sprints

Fonctionnalités : Board Kanban, planification, métriques

Utilise : components/sprints/SprintBoard.tsx, stores/useSprintStore.ts, utils/getSprints.tsx

Utilisé par : Navigation projet

Export : default function SprintsPage

### App/sprints/[sprintId]/page.tsx

Description : Détails d'un sprint

Fonctionnalités : Burndown, rétrospective, clôture sprint

Utilise : components/models/SprintForm.tsx, stores/useSprintStore.ts, app/api/sprints/[sprintId]/route.ts

Utilisé par : Liste sprints

Export : default function SprintDetailPage

### app /tasks/page.tsx

Description : Liste des tâches

Fonctionnalités : Filtres avancés, vue Kanban/liste, assignations

Utilise : components/models/TaskList.tsx, stores/useTasksStore.ts, utils/getTasks.tsx

Utilisé par : Navigation projet, sprints

Export : default function TasksPage

### app /tasks/[taskId]/page.tsx

Description : Détails d'une tâche

Fonctionnalités : Infos complètes, sous-tâches, temps passé

Utilise : components/models/TaskForm.tsx, stores/useTasksStore.ts, app/api/tasks/[taskId]/route.ts

Utilisé par : Liste tâches, board sprint

Export : default function TaskDetailPage

### app/ schema/page.tsx

Description : Gestion des schémas de données

Fonctionnalités : Création tables, champs, relations

Utilise : components/models/SchemaList.tsx, stores/useSchemaStore.ts, utils/getSchemaFields.tsx

Utilisé par : Navigation projet

Export : default function SchemaPage

### app/schema/[schemaId]/page.tsx

Description : Détails d'un schéma

Fonctionnalités : Edition champs, génération code

Utilise : components/models/SchemaForm.tsx, stores/useSchemaStore.ts, app/api/schema-fields/[fieldId]/route.ts

Utilisé par : Liste schémas

Export : default function SchemaDetailPage

### app /files/page.tsx

Description : Gestionnaire de fichiers

Fonctionnalités : Upload, organisation, preview

Utilise : components/models/FilesList.tsx, stores/useFilesStore.ts, utils/getFiles.tsx

Utilisé par : Navigation projet

Export : default function FilesPage

### app/ /files/[fileId]/page.tsx

Description : Détails d'un fichier

Fonctionnalités : Metadata, versions, partage

Utilise : components/models/FilesForm.tsx, stores/useFilesStore.ts, app/api/files/[fileId]/route.ts

Utilisé par : Liste fichiers

Export : default function FileDetailPage

### app/reports/activity/page.tsx

Description :

Fonctionnalités :

Utilise :

Export :

## teams

### app/teams/page.tsx

Description : Gestion des équipes

Fonctionnalités : Liste équipes, membres, permissions

Utilise : components/models/TeamList.tsx, stores/useTeamsStore.ts, utils/getTeams.tsx

Utilisé par : Navigation, admin

Export : default function TeamsPage

### app/teams/[teamId]/page.tsx

Description : Détails d'une équipe

Fonctionnalités : Gestion membres, projets assignés

Utilise : components/models/TeamForm.tsx, stores/useTeamsStore.ts, app/api/teams/[teamId]/route.ts

Utilisé par : Liste équipes

Export : default function TeamDetailPage

## Suivie d’activité

### app/roadmap/page.tsx

Description : Roadmap du projet

Fonctionnalités : Vue timeline, phases, dépendances

Utilise : components/roadmap/RoadmapView.tsx, stores/useRoadmapStore.ts, utils/getRoadmaps.tsx

Utilisé par : Navigation projet

Export : default function RoadmapPage

### app/roadmap/timelog/page.tsx

Description : Suivi du temps par roadmap

Fonctionnalités : Rapports temps, comparaison estimations

Utilise : components/timelog/TimeLogList.tsx, utils/getTimeLogs.tsx

Utilisé par : Navigation roadmap

Export : default function TimeLogPage

### app/roadmap/activitylogs/page.tsx

Description : Historique des activités

Fonctionnalités : Filtres par utilisateur, type, date

Utilise : components/activity/ActivityList.tsx, utils/getActivityLogs.tsx

Utilisé par : Navigation, monitoring

Export : default function ActivityLogsPage

## Admin du projet

### app/admin/page.tsx

Description :

Fonctionnalités :

Utilise :

Export :

### app/admin/users/page.tsx

Description :

Fonctionnalités :

Utilise :

Export :

### app/admin/users/[userId]/page.tsx

Description :

Fonctionnalités :

Utilise :

Export :

### app/admin/teams/page.tsx

Description :

Fonctionnalités :

Utilise :

Export :

### app/admin/projects/page.tsx

Description :

Fonctionnalités :

Utilise :

Export :

### app/admin/activity-logs/page.tsx

Description :

Fonctionnalités :

Utilise :

Export :

### app/admin/settings/page.tsx

Description :

Fonctionnalités :

Utilise :

Export :

# components

## layout

### components/layout/header/Header.tsx

Description : En-tête principal

Fonctionnalités :

Logo et navigation principale

Menu utilisateur

Notifications

Recherche globale

Utilise : components/layout/header/Logo.tsx, components/layout/header/HeaderNav.tsx, compoents/layout/header/IsConnected

Export : default function Header

### components/layout/header/Logo.tsx

Description :

Fonctionnalités :

Utilise :

Export :

### components/layout/header/HeaderNav.tsx

Description :

Fonctionnalités :

Utilise :

Export :

### components/layout/header/Isconected.tsx

Description :

Fonctionnalités :

Utilise :

Export :

### components/layout/footer.tsx

Description :

Fonctionnalités :

Utilise :

Export :

### components/layout/Background.tsx

Description :

Fonctionnalités :

Utilise :

Export :

## models

### components/models/Projects/ProjectList.tsx (DONE)

Description : Liste des projets avec filtres

Fonctionnalités :

Affichage grille/liste

Filtres par statut, équipe, date

Pagination

Actions rapides (favoris, archiver)

Utilise : components/models/ProjectCard.tsx, stores/useProjectStore.ts, hooks/use-Modal.ts

Export : default function ProjectList

### components/models/projects/ProjectFilters.tsx (DONE)

utilisé par app/projects/page.tsx

### components/models/projects/ProjectCard.tsx

Description :

Fonctionnalités :

Utilise :

Export :

### components/models/ProjectForm.tsx (DONE)

Description : Formulaire projet

Fonctionnalités : Création/édition, Validation

Utilise : lib/validators/projectSchemas, app/api/Project

Export :

### components/models/ FeaturesList.tsx

Description : Liste hiérarchique des features

Fonctionnalités :

Vue arborescente avec indentation

Expand/collapse des branches

Drag & drop pour réorganiser

Actions contextuelles

Utilise : components/models/FeaturesForm.tsx, stores/useFeatureStore.ts, hooks/use-Modal.ts

Export : default function FeaturesList

### components/models/ FeaturesForm.tsx

Description :

Fonctionnalités :

Utilise :

Export :

### components/models/ TaskList.tsx

Description :

Fonctionnalités :

Utilise :

Export :

### components/models/TaskForm.tsx

Description :

Fonctionnalités :

Utilise :

Export :

### components/models/ FilesList.tsx

Description :

Fonctionnalités :

Utilise :

Export :

### components/models/ FilesForm.tsx

Description :

Fonctionnalités :

Utilise :

Export :

### 

### components/sprints/SprintBoard.tsx

Description : Board Kanban pour sprints

Fonctionnalités : Colonnes par statut, Drag & drop des tâches

Utilise : components/sprints/SprintColumn, stores/useSprintStore

Export :

### components/sprints/SprintColumn.tsx

Description : Colonne du board

Fonctionnalités : Affichage des tâches, Drop zone

Utilise : components/tasks/TaskCard, hooks/useTaskMutations

Export :

### components/models/UserStoryList.tsx

Description : Liste des user stories

Fonctionnalités :

Affichage par epic/feature

Prioritization drag & drop

Estimation en story points

Statuts visuels

Utilise : components/models/UserStoryForm.tsx, stores/useUserStoryStore.ts

Export : default function UserStoryList

### components/sprints/SprintBoard.tsx

Description : Board Kanban pour sprints

Fonctionnalités :

Colonnes par statut (TODO, IN\_PROGRESS, REVIEW, DONE)

Drag & drop des tâches

Compteurs par colonne

Filtrages rapides

Utilise : components/sprints/SprintColumn.tsx, stores/useSprintStore.ts, stores/useTaskStore.ts

Export : default function SprintBoard

### components/models/UserStoryForm.tsx

### components/models/SprintList.tsx

### components/models/SprintForm.tsx

### components/models/RoadmapList.tsx

### components/models/RoadmapForm.tsx

### components/models/CommentList.tsx

### components/models/CommentForm.tsx

### components/models/TeamList.tsx

### components/models/TeamForm.tsx

### components/dashboard/DashboardStats.tsx

### components/providers/Providers.tsx

## ui

accordion

alert-dialogue

alert

avatar

badge

button

calendar

card

checkbox

dialog

dropdown-menu

form

input

label

popover

progress

radio-group

select

sheet

skelton

sonner

switch

table

tabs

textarea

toast

toaster

# lib

## aut

### lib/auth/auth.ts (DONE)

Description :

Fonctionnalités :

Utilise :

Export :

### lib/auth/auth-client.ts (DONE)

Description :

Fonctionnalités :

Utilise :

Export :

### lib/auth/auth-server.ts (DONE)

Description :

## generated

### lib/generated/prisma/client.ts (DONE)

Description :

Fonctionnalités :

Utilise :

Export :

## autres

### lib/prisma.ts (DONE)

Description :

Fonctionnalités :

Utilise :

Export :

### lib/utils.ts

Description :

Fonctionnalités :

Utilise :

Export :

### lib/constants.ts

- Constantes globales

# stores

### stores/useAuthStore.ts

Description : Store d'authentification

Fonctionnalités :

État utilisateur connecté

Permissions et rôles

Actions de connexion/déconnexion

Persistance de session

Utilise : lib/auth/auth-client.ts

Export : useAuthStore

### Stores/useProjectStore

Description : pour un projet

Fonctionnalités :

Utilise :

Export :

### stores/useProjectsStore.ts (DONE)

Description : Store de gestion des projets

Fonctionnalités :

📌 Stocke le projet actuellement sélectionné (`project`)

🧭 Stocke son identifiant (`projectId`) pour navigation ou lookup rapide

🔄 Permet de mettre à jour ou réinitialiser le projet actif

🧹 Peut être vidé via `clearProject` (ex. : changement d’espace de travail)

Export : useProjectStore

### stores/useFeaturesStore.ts ()

Description : Store des features

Fonctionnalités :

Features par projet avec hiérarchie

Gestion des relations parent/enfant

Calcul de progression

Cache avec invalidation

Utilise : utils/getFeatures.tsx, API routes

Export : useFeatureStore

### stores/useSprintStore.ts

Description :

Fonctionnalités :

Utilise :

Export :

### stores/useTasksStore.ts

Description : Store des tâches

Fonctionnalités :

Tâches par contexte (projet, sprint, feature)

Gestion des assignations

Suivi des statuts

Métriques de performance

Utilise : utils/getTasks.tsx, API routes

Export : useTaskStore

# Hooks

### Hooks/use-toast.ts (DONE)

Description : Hook pour notifications toast

Fonctionnalités :

Gestion des messages de succès/erreur

Auto-dismiss configurable

Types de toast (info, warning, error)

Utilise : components/ui/toast.tsx

Export : useToast

### Hooks/use-Modal.ts (DONE)

Description : Hook pour gestion des modales

Fonctionnalités :

État d'ouverture/fermeture

Gestion du focus et accessibility

Fermeture par ESC/backdrop

Utilise : components/ui/dialog.tsx

Export : useModal

# Utils

### utils/getRoadmaps.tsx

### utils/getTeams.tsx

### utils/getUsers.tsx

### Utils/getProjects.tsx (byUser) (donne)

Description : utilise useSession de lib/auth/auth-client.ts

Fonctionnalités :

Utilise :

Export :

### Utils/getFeatures.tsx (byProjectId)

Description :

Fonctionnalités :

Utilise :

Export :

### Utils/getUserStories.tsx (byProjectId)

Description :

Fonctionnalités :

Utilise :

Export :

### Utils/getSprints.tsx (byProjectId)

Description :

Fonctionnalités :

Utilise :

Export :

### Utils/getTasks.tsx (byProjectId)

Description :

Fonctionnalités :

Utilise :

Export :

### Utils/getSchemafields.tsx (byProjectId)

Description :

Fonctionnalités :

Utilise :

Export :

### Utils/getFiles.tsx (byProjectId)

Description :

Fonctionnalités :

Utilise :

Export :

### Utils/getComments.tsx

Description :

Fonctionnalités :

Utilise :

Export :

### Utils/getTimeLogs.tsx (byTaskId, byUserId)

Description : Fonction pour récupérer les suivis de temps par tâche ou utilisateur.

### Utils/getActivityLogs.tsx (byUserId)

Description : Fonction pour récupérer les journaux d’activité par utilisateur ou type.

Fonctionnalités :

Utilise :

Export :

### Utils/uploadFile.tsx

Description : Fonction pour gérer le téléversement de fichiers

Fonctionnalités :

Utilise :

Export :

# Validators

### validators/projectSchemas.ts

Description : Schémas de validation Zod pour projets

Fonctionnalités :

Validation création/édition projet

Règles métier (dates, priorités)

Validation des relations équipes

Utilise : zod

Export : createProjectSchema, updateProjectSchema

### validators/authSchemas.ts

### validators/taskSchemas.ts

### validators/featureSchemas.ts

### validators/sprintSchemas.ts

# Autres

### middleware.ts

Description :

Fonctionnalités :

Utilise :

Export :

### next.config.js

Configuration Next.js

### tailwind.config.js

- Configuration Tailwind CSS

### tsconfig.json

- Configuration TypeScript

### package.json

- Dépendances et scripts

### .env

- Variables d'environnement

### prisma/schema.prisma

- Votre schéma Prisma

### components.json

- Configuration shadcn/ui