A white and blue cover with hexagons

AI-generated content may be incorrect.

Contents

[Présentation du Projet 3](#_Toc195743441)

[Objectif 3](#_Toc195743442)

[Fonctionnalités 4](#_Toc195743443)

[Fonctions secondaires (priorité moyenne) : 4](#_Toc195743444)

[Technologies et Outils 5](#_Toc195743445)

[Planning Prévisionnel 6](#_Toc195743446)

[Modèles et Données 7](#_Toc195743447)

[Idées Bonus à Ajouter 8](#_Toc195743448)

[Prochaines étapes immédiates 8](#_Toc195743449)

[Cyle de vie du projet 9](#_Toc195743450)

[Diagramme de Gant 9](#_Toc195743451)

[Diagramme de classe 9](#_Toc195743452)

[Diagramme de cas d'utilisation 9](#_Toc195743453)

[Diagramme de sequences 9](#_Toc195743454)

[Lien des comptes 9](#_Toc195743455)

[Conclusion 10](#_Toc195743456)

Cahier des Charges

Plateforme de Gestion des Anciens Sujets Universitaires

# Présentation du Projet

**Nom du projet** : Plateforme de gestion des anciens sujets universitaires

**Responsables du projet** :

* -Chef de projet / Responsable technique :5 Dabo Ali
* Développeur Back-end : py
* Développeur Front-end :Dabo Ali
* Responsable design UI/UX : Dabo Ali et py

Objectif:

Créer une plateforme web sécurisée permettant aux étudiants d'accéder aux anciens sujets d’examens et corrigés via un espace personnel, et aux administrateurs d’ajouter et gérer ces ressources.

**Objectifs Détailés**

* Centraliser et sécuriser les anciens sujets et corrigés.
* Faciliter la révision des étudiants via une interface de recherche intuitive.
* Mettre en place une gestion d’utilisateurs basée sur les rôles (étudiant/admin).
* Permettre une gestion simplifiée des sujets et des corrigés pour les administrateurs.

## Fonctionnalités

**Pour les étudiants :**

* Connexion sécurisée avec identifiant unique et mot de passe.
* Accès à un tableau de bord personnel.
* Recherche de sujets par matière, année, semestre.
* Téléchargement des sujets et corrigés (PDF).
* Consultation de l’historique de téléchargement.

**Pour les administrateurs :**

* Connexion à un espace admin.
* Ajout de nouveaux sujets et corrigés via un formulaire (upload de fichiers, métadonnées associées).
* Modification / suppression de sujets et corrigés.
* Gestion des utilisateurs (ajout/suppression étudiants, réinitialisation de mot de passe).

### Fonctions secondaires (priorité moyenne) :

* Statistiques : nombre de téléchargements par sujet, par utilisateur.
* Gestion des demandes de sujets non disponibles (boîte de suggestions).
* Authentification par rôles : distinction Étudiant / Admin.
* Historique de connexions et actions.

## Technologies et Outils

|  |  |  |  |
| --- | --- | --- | --- |
| Côté Client (Frontend) | Côté Serveur (Backend) | Base de Données | Outils Divers |
| React JS | Django (Python) | PostgreSQL / MySQL | Postman (tests API) |
| Figma (UI Design) | Django REST Framework |  | Git & GitHub (versioning) | |
| Tailwind CSS / Bootstrap | JWT pour l’authentification |  | Netlify/Vercel (front) |
|  |  |  | Render/Railway (back) |

## Planning Prévisionnel

|  |  |  |  |
| --- | --- | --- | --- |
| Phase | Responsable(s) | Durée estimée | Statut |
| Création du dépôt GitHub | Dabo Ali | 1 jour |  |
| Rédaction du cahier des charges | Dabo Ali | 1 jours |  |
| Conception des maquettes (Figma) | Dabo Ali & Py |  |  |
| Mise en place environnement | Dabo Ali & Py |  |  |
| Définition des modèles et migrations | Py |  |  |
| Développement de l’API (auth, sujets) | Py |  |  |
| Intégration du front React (connexion, tableau de bord) | Dabo Ali |  |  |
| Intégration API / Front | Py |  |  |
| Tests et ajustements | Dabo Ali & Py |  |  |
| Déploiement | Dabo Ali & Py |  |  |

# Modèles et Données

**Utilisateur (User)**

|  |  |  |
| --- | --- | --- |
| Champ | Type | Obligatoire |
| id | Auto (UUID) | Oui |
| identifiant | String | Oui |
| email | String (unique) | Oui |
| mot\_de\_passe | Hashé | Oui |
| rôle | String (étudiant/admin) | Oui |

**Sujet**

|  |  |  |
| --- | --- | --- |
| Champ | Type | Obligatoire |
| id | Auto (UUID) | Oui |
| matière | String | Oui |
| année | String | Oui |
| semestre | String |  |
| fichier\_sujet | Fichier (PDF) | Oui |
| fichier\_corrigé | Fichier (PDF) |  |
| ajouté\_par | Référence admin | Oui |

**Sécurité**

* Authentification sécurisée avec JWT.
* Hashage des mots de passe avec bcrypt.
* Rôles et permissions clairement séparés (admin / étudiant).
* Données et fichiers protégés via des accès restreints et serveurs HTTPS.

## Idées Bonus à Ajouter

Tu peux décider d’intégrer certaines de ces idées plus tard :

* Téléchargements limités: par nombre ou période (ex : 5 par jour).
* Commentaires ou notation des sujets: pour aider les autres étudiants.
* Demandes de sujets spécifiques: si non disponibles.
* Notifications email: lors de la mise en ligne de nouveaux sujets.

## Prochaines étapes immédiates

1. Créer le dépôt GitHub commun

2. Finaliser ce cahier des charges ensemble

3. Designer la page d’accueil et page de recherche sous Figma

4. Définir les modèles Django et lancer les premières migrations

5. Créer l’API Django REST

6. Développer l’interface React et connecter l’API

7. Effectuer les tests et déployer

# Cyle de vie du projet

# Diagramme de Gant

Diagramme de classe

# Diagramme de cas d'utilisation

# Diagramme de sequences

# Lien des comptes

Github :

Py 🡪 <https://github.com/AgohChris>

Dabo Ali 🡪 <https://github.com/alsondab>

Depot Git du projet 🡪 <https://github.com/alsondab/Plateforme_de_Gestion_des_Anciens_Sujets_Universitaires->

# Conclusion

Ce cahier des charges vous servira de guide pour structurer le projet, répartir les tâches et suivre votre avancement. Vous pouvez bien sûr l’adapter et l’ajuster selon l’évolution du projet.