ECOLE MAROCAINE DES SCIENCES DE L'INGENIEUR
Membre de HONORIS UNITED UNIVERSITIES

HONORIS UNITED UNIVERSITIES

Rapport de projet
3ème année
Ingénierie Informatique et Réseaux

Sous le thème

Gestionnaire de contact avec Django

Réalisé par :

Aya Allaoui
Othmane Chougrad

Année UNIVERSITAIRE: 2024-2025

Sommaire

I - Introduction

Presentation de l'utilite principale du projet

Ce projet consiste en une application web de gestion de contacts, développée avec Django, permettant aux utilisateurs de stocker, organiser et retrouver facilement leurs contacts. Les fonctionnalités principales incluent :

- **L’ajout**, la **modification** et la **suppression** de contacts (nom, email, numéro de téléphone).
- Une **authentification** sécurisée par numéro de téléphone et mot de passe.
- La **recherche** rapide d’un contact par son numéro.
- Le **filtrage** des contacts par catégorie (domicile, professionnel, mobile).
- Un **stockage** sécurisé dans une base de données SQLite pour éviter les pertes de données.

Objectifs du projet

1. Gérer efficacement les données des utilisateurs.
2. Proposer une solution robuste pour éviter la perte de contacts.
3. Faciliter la recherche et l'organisation des contacts.
4. Répondre aux besoins des utilisateurs (particuliers et professionnels).

Technologies utilisées

- **Backend** : Django (framework Python)
- **Base de données** : SQLite3
- **UI** : HTML/CSS, Bootstrap
- **Sécurité** : Protection contre les injections SQL

II - Contexte général & Analyse des besoins

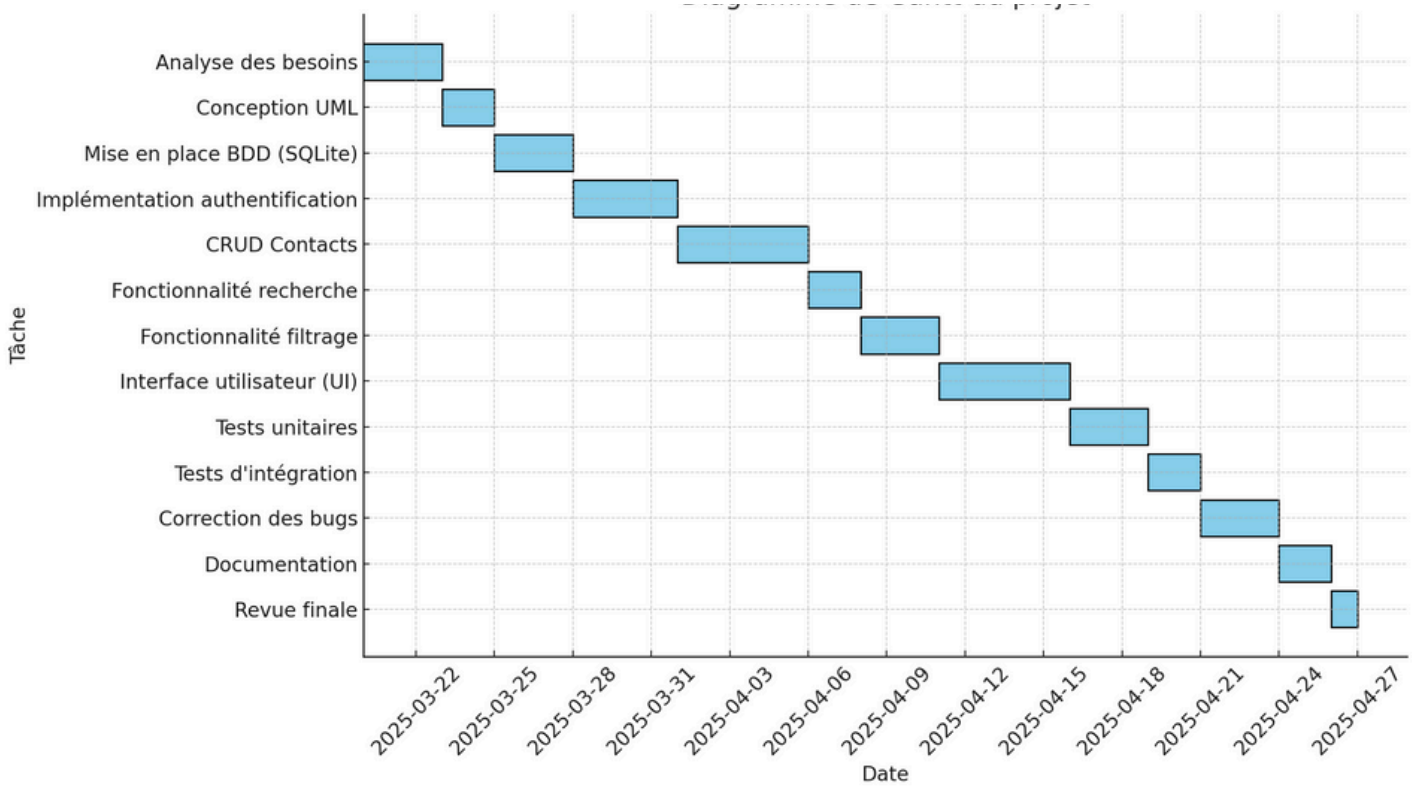
Exigences fonctionnelles

Fonctionnalité	Description	Priorité
Authentification	Connexion via numéro de téléphone et mot de passe	Haute
Ajout d'un contact	Saisie du nom, email, numéro et catégorie	Haute
Modification d'un contact	Mise à jour des informations	Moyenne
Suppression d'un contact	Suppression avec confirmation	Moyenne
Recherche par numéro	Recherche instantanée	Haute
Filtrage par catégorie	Affichage par "Domicile", "Professionnel", "Mobile"	Moyenne
Stockage sécurisé	Sauvegarde en base de données	Haute

Exigences non fonctionnelles

- **Performance** : Temps de réponse < 2 secondes.
- **Sécurité** : Protection contre les injections SQL.
- **Fiabilité** : Aucune perte de données.
- **Interface intuitive** : Design clair et convivial.
- **Maintenabilité** : Code structuré et documenté.

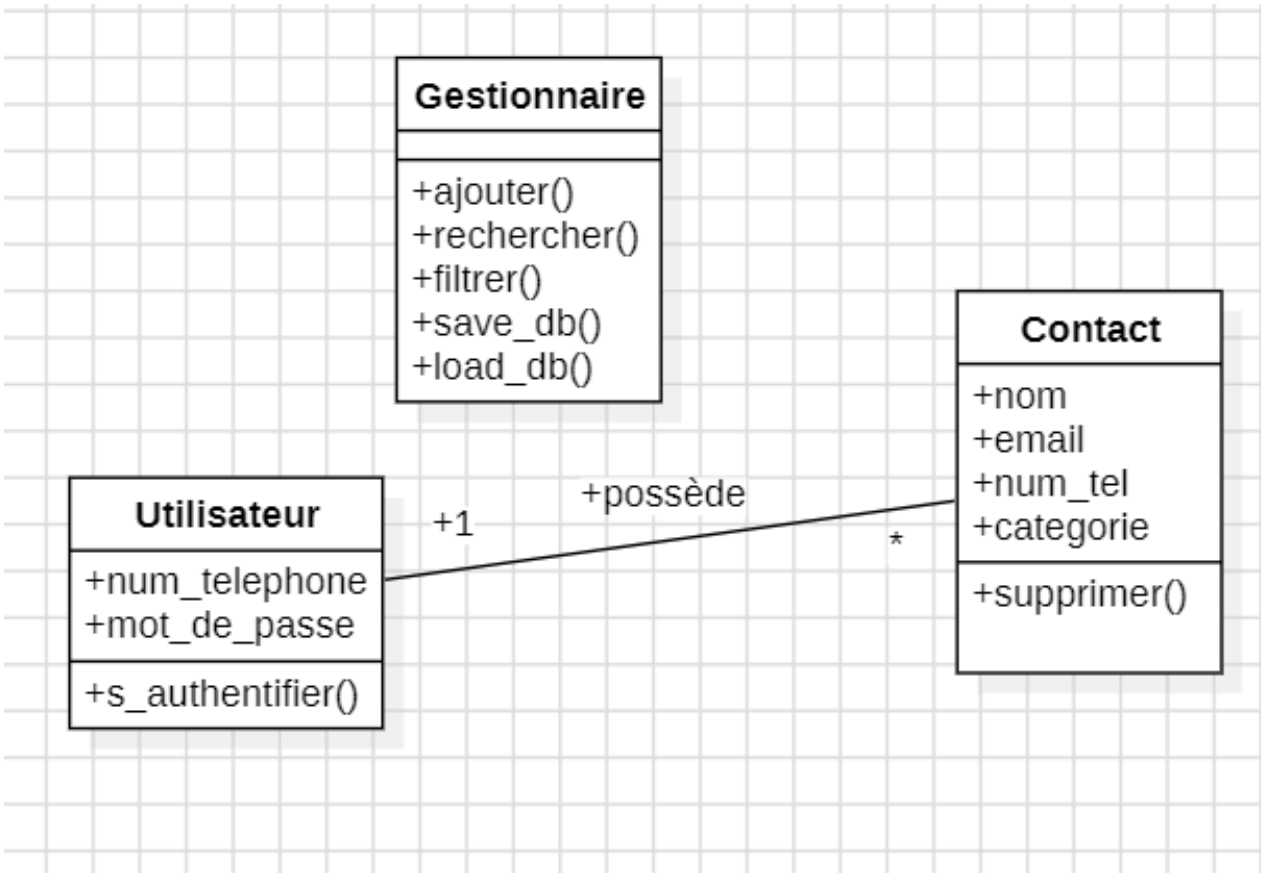
Voici un diagramme de Gantt montrant l'analyse complète par rapport à la durée de chaque tâche :



III - Conception du système

Modélisation des classes et objets

Voici un diagramme de classe illustrant la structure du projet en utilisant StarUML :



IV- Détails d'implémentation

Code source commenté

Exemple :

```python
from django.db import models
from django.contrib.auth.models import AbstractBaseUser, BaseUserManager, PermissionsMixin

#créer utilisateurs normaux et superutilisateurs
class CustomUserManager(BaseUserManager):
    #créer utilisateur
    def create_user(self, phone_number, password=None, **extra_fields):
        """
        Crée et enregistre un utilisateur avec le numéro de téléphone et mot de passe donnés.
        """
        if not phone_number:  # Vérifie input telephone
            raise ValueError("Ecrivez votre numéro de téléphone !!!!")

        # crée & enregistre l'instance utilisateur
        user = self.model(phone_number=phone_number, **extra_fields)
        user.set_password(password)  #hash le mdp ( normalement ..)
        user.save(using=self._db)  # database saving
        return user
```

Description de l'implémentation

- **Authentification** : Utilisation de Django AbstractUser.
- **Recherche** : Filtrage avec Contact.objects.filter(numero=query).
- **Filtrage par catégorie** : Utilisation de GET parameters.

Description de l'implémentation

Défis rencontrés et solutions

- **Problème** : Authentification corrompue à cause de la mal configuration du code
- **Solution** : Création de nouvelles classes correspondantes dans le modèle

V - Tests et validation

Scénarios de test

Test	Résultat attendu
Ajout d'un contact	Le contact s'affiche dans la liste
Recherche par numéro	Retourne le bon contact en < 2s
Filtrage par catégorie	Affiche uniquement les contacts de la catégorie sélectionnée

Résultats des tests

- Toutes les fonctionnalités principales validées.
- Performance respectée (< 2s pour la recherche).

VI - Conclusion & Perspectives

Bilan du projet

Application fonctionnelle répondant aux besoins initiaux.

Limites du projet

Pas d'interface graphique avancée (possible amélioration avec React et CSS moderne).

Perspectives d'amélioration

- Interface responsive pour téléphone.
- Synchronisation cloud (passage à MongoDB).

VII - Annexes

Autres ressources

Documentation :

Cours professeur
https://www.djangoproject.com
https://www.getbootstrap.com
https://www.w3school.com
https://www.github.com

Réalisation :

https://www.canva.com