  
  
  
  
  
  
  
APPLICATION WEB : CHAT\_APP

Réalisé par :

LIDALT Omar  
NABI Maya  
OZKOSAR Enes

Année universitaire :

2024/2025

Table des matières

* Partie 1 : Introduction
* Partie 2 : Technologies utilisées
* Partie 3 : Analyse et conception
* Partie 4 : Implémentation
* Partie 5 : Adresse du serveur
* Partie 6 : Conclusion

**INTRODUCTION :**

Dans le contexte du cours Technologies Web II / Programmation Web, nous avons mené à bien un projet dont l'objectif était de concevoir une application de messagerie centralisée. Ce projet représente une occasion de mettre en pratique nos compétences en développement web, tout en découvrant les échanges entre le serveur, dirigé par Django, et le client, piloté par JavaScript.

Ce projet vise principalement à concevoir une application qui autorise la communication simultanée de plusieurs utilisateurs à travers divers salons de discussion. Cette application comprend des caractéristiques clés comme l'enregistrement et la connexion d'utilisateurs, la gestion de salons, l'insertion d'émojis ainsi que des outils de modération pour garantir un contexte sécuritaire et plaisant.

Outre l'aspect technique, ce projet nous a donné l'occasion de collaborer en équipe et de nous acclimater aux meilleures pratiques de gestion de projets collaboratifs. Nous avons utilisé des outils modernes et observé des normes de codage strictes afin d'assurer la qualité et la clarté de notre travail.

Cette introduction expose le cadre général du projet et sa finalité globale. Les parties suivantes fourniront une description détaillée des technologies utilisées, des décisions prises en matière de conception, de l'implémentation et des résultats obtenus.

**Technologies utilisées :**

Pour la réalisation de ce projet, nous avons utilisé les technologies suivantes :

* **Python** : Langage de programmation principal pour le développement du côté serveur.
* **Django** : Framework web permettant une gestion efficace des modèles, des vues et des contrôleurs.
* **JavaScript** : Utilisé pour dynamiser les pages côté client et gérer les interactions utilisateur.
* **Channels / WebSockets** : Implémentés pour assurer une communication bidirectionnelle en temps réel entre le client et le serveur.
* **SQLite** : Base de données légère pour le stockage des informations des utilisateurs et des messages

**Analyse et conception :**

**Analyse et conception**

**Dans le cadre de la conception de l’application, nous avons structuré les données en utilisant trois modèles principaux dans Django : User, Room et Message. Ces modèles permettent de gérer les utilisateurs, les salons de discussion et les messages échangés, tout en respectant une approche orientée objet. Voici les détails des choix de conception et des modèles :**

**1. Modèle User**

* **Extends : AbstractUser de Django pour profiter des fonctionnalités d’authentification intégrées.**
* **Champs ajoutés :**
  + **profile\_pic : Permet à chaque utilisateur de personnaliser son profil avec une image.**
* **Méthodes :**
  + **get\_full\_name() : Retourne le nom complet de l'utilisateur ou, à défaut, son nom d'utilisateur.**

**2. Modèle Room**

* **Champs :**
  + **name : Nom unique pour identifier le salon.**
  + **moderator : Utilisateur responsable de la modération du salon (relation ForeignKey vers User).**
  + **users : Liste des utilisateurs ayant rejoint le salon (relation ManyToManyField).**
  + **date\_created : Enregistre automatiquement la date de création du salon.**
* **Méthodes :**
  + **\_\_str\_\_ : Retourne le nom du salon pour une représentation lisible.**

**3. Modèle Message**

* **Champs :**
  + **room : Salon associé au message (relation ForeignKey vers Room).**
  + **sender : Utilisateur ayant envoyé le message (relation ForeignKey vers User).**
  + **content : Contenu textuel du message.**
  + **timestamp : Enregistre automatiquement la date et l’heure de l’envoi.**
* **Méthodes :**
  + **\_\_str\_\_ : Affiche un aperçu du message, incluant le nom de l’expéditeur et une partie du contenu.**

**4. Choix de conception**

* **Relations entre modèles :**
  + **L’utilisation des relations ForeignKey et ManyToManyField permet de refléter les interactions réelles entre les utilisateurs, les salons et les messages.**
* **Gestion des droits :**
  + **Le champ moderator dans le modèle Room permet d’attribuer des privilèges spécifiques à un utilisateur.**
* **Évolutivité :**
  + **La structure actuelle peut facilement être étendue pour intégrer des fonctionnalités supplémentaires, comme des salons privés ou des fichiers joints dans les messages.**

**Une image contenant texte, diagramme, capture d’écran, ligne

Description générée automatiquement**

**IMPLEMENTATION :**

**Le projet est basé sur une architecture solide qui associe Django côté serveur à JavaScript côté client. Les modèles établissent la structure des données et simplifient leur gestion grâce à l'ORM intégré de Django, tandis que les vues et les templates offrent la possibilité de créer des pages dynamiques. L'emploi de Django Channels et WebSockets assure une communication instantanée, indispensable pour l'opération sans accroc des salons de discussion. Du point de vue du client, JavaScript permet le renouvellement dynamique des messages et des utilisateurs connectés, offrant ainsi à l'utilisateur une expérience à la fois interactive et réactive. On a minutieusement testé tous les composants pour assurer un rendement optimal.**

**Une image contenant texte, capture d’écran, logiciel, Page web

Description générée automatiquement**

**Une image contenant texte, capture d’écran, logiciel, Système d’exploitation

Description générée automatiquement**

**Une image contenant texte, capture d’écran, logiciel, Icône d’ordinateur

Description générée automatiquement**

**Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

Description générée automatiquement**

**Une image contenant texte, capture d’écran, logiciel, Icône d’ordinateur

Description générée automatiquement**

**Une image contenant texte, capture d’écran, logiciel, Icône d’ordinateur

Description générée automatiquement**

**Une image contenant texte, capture d’écran, logiciel, Système d’exploitation

Description générée automatiquement**

**Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

Description générée automatiquement**

Adresse du serveur

**CONCLUSION**

**Le développement de cette application de messagerie centralisée a été une aventure gratifiante, nous offrant l'opportunité de renforcer nos aptitudes en programmation web tout en mettant en pratique des notions essentielles telles que la gestion de données, l'échange client-serveur et la communication en direct. En utilisant Django, JavaScript et WebSockets, nous avons réussi à élaborer une solution à la fois opérationnelle et adaptable qui répond aux critères spécifiés dans le cahier des charges.**

**Cette réalisation, au-delà des compétences techniques, nous a instruits sur l'importance de la collaboration en équipe, l'adhésion aux bonnes méthodes de développement et la prévision des challenges associés à la conception et à la mise en œuvre d'un projet.  
  
Les caractéristiques mises en place, y compris la gestion des salons, les émoticônes et les droits de modération, établissent un socle robuste pour des évolutions futures, comme l'intégration de fonctionnalités plus poussées telles que les notifications ou le cryptage des messages.**

**Ce projet s'avère être une réussite aussi bien sur le plan technique qu'éducatif, consolidant notre assurance dans l'exécution de projets complexes au sein d'un environnement collaboratif.**