

Automatique et Systèmes  
Embarqués

1ère année ENSISA

Sujet

**MeetingPro – Application  
de gestion de réservation  
de salles**

Présenté Par :

ABAGA Wayne

AGBO Florian

Encadré Par :

M. Loïc Riegel

M. François Ludwinski



# 1. Introduction

Ce projet a été réalisé dans le cadre de l'unité d'enseignement "Analyse et Organisation Orientée Objet (AOO)" en 1ère année Automatique & Systèmes Embarqués à l'ENSISA. L'objectif était de développer une application Python permettant à une entreprise fictive, "MeetingPro", de gérer la réservation de salles via une interface graphique moderne et simple à utiliser.

## 2. Objectifs et cahier des charges

L'application MeetingPro propose la location de trois types de salles : standard, conférence, et informatique.

Les utilisateurs doivent pouvoir :

- Ajouter un nouveau client
- Ajouter une nouvelle salle
- Afficher les salles réservables
- Visualiser les réservations pour un client
- Vérifier la disponibilité d'une salle à un créneau donné
- Voir les salles disponibles sur un créneau
- Réaliser une réservation

Le tout devait être encapsulé dans une interface graphique intuitive comportant ces fonctionnalités sous forme d'onglets, avec persistance des données via un fichier data.json.

### 3. Démarche et architecture

Nous avons opté pour une architecture MVC (Modèle-Vue-Contrôleur) :

- Modèle (model/) : contient les classes Client, Salle, et Reservation
- Contrôleur (controller/) : gère la logique métier, l'ajout d'objets, la persistance JSON
- Vue (view/) : l'interface utilisateur avec customtkinter dans menu\_principal.py

Le fichier main.py est le point d'entrée de l'application.

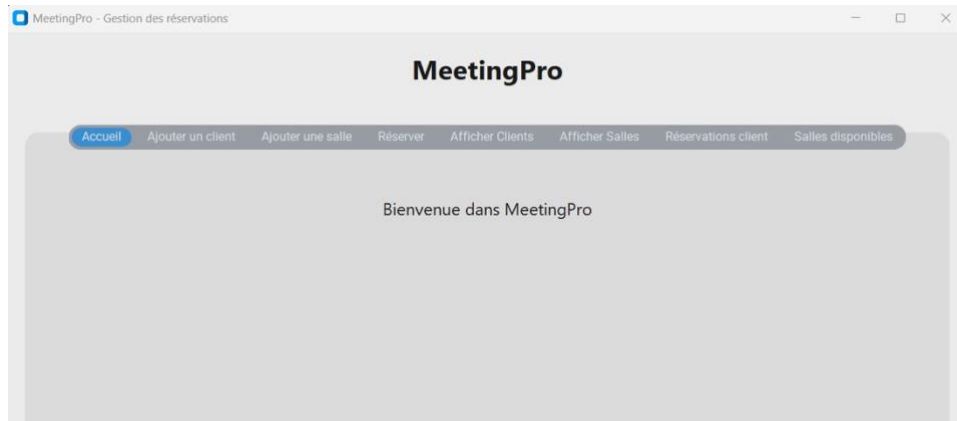
Les données sont stockées dans data.json et mises à jour automatiquement après chaque action.

Tâche	Réalisée par
Implémentation des modèles	Wayne
Contrôleurs JSON & réservations	Wayne
Interface graphique (Tk puis customtkinter)	Florian
Tests unitaires	Wayne + Florian
README et Documentation	Florian
Coordination Git + Branches	Wayne (repo principal)

## 4. Description de l'application MeetingPro

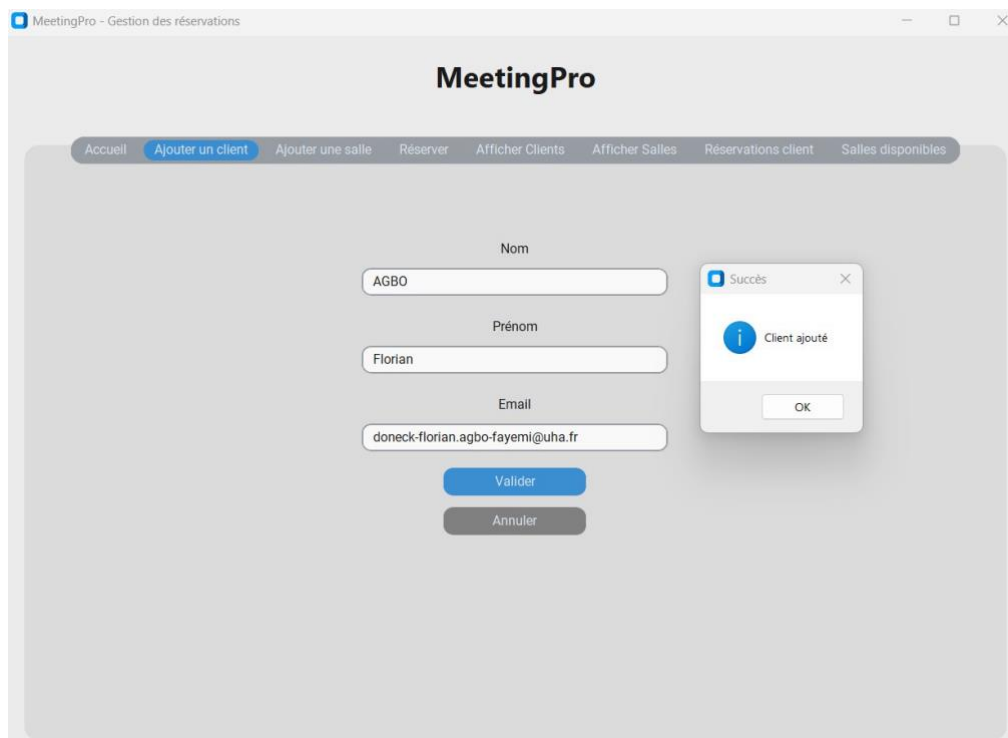
- Onglet “Accueil” :

Cet onglet affiche un message de bienvenue. Il sert de page d'accueil simple et visuellement claire à chaque lancement.



- Onglet “Ajouter un client” :

Permet d'ajouter un nouveau client dans le système. L'utilisateur doit remplir les champs suivants :

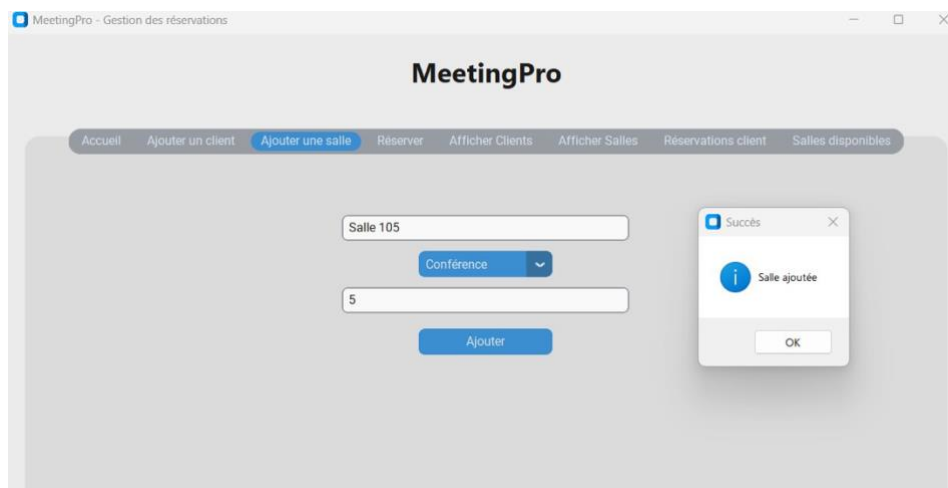


- Onglet “Ajouter une salle” :

Permet d’enregistrer une nouvelle salle avec :

- Un identifiant (ex : SALLE101)
- Un type (Standard, Conférence, Informatique)
- Une capacité (nombre de places)

Cliquez sur Valider pour enregistrer. En cas de doublon d’identifiant, une erreur est affichée.

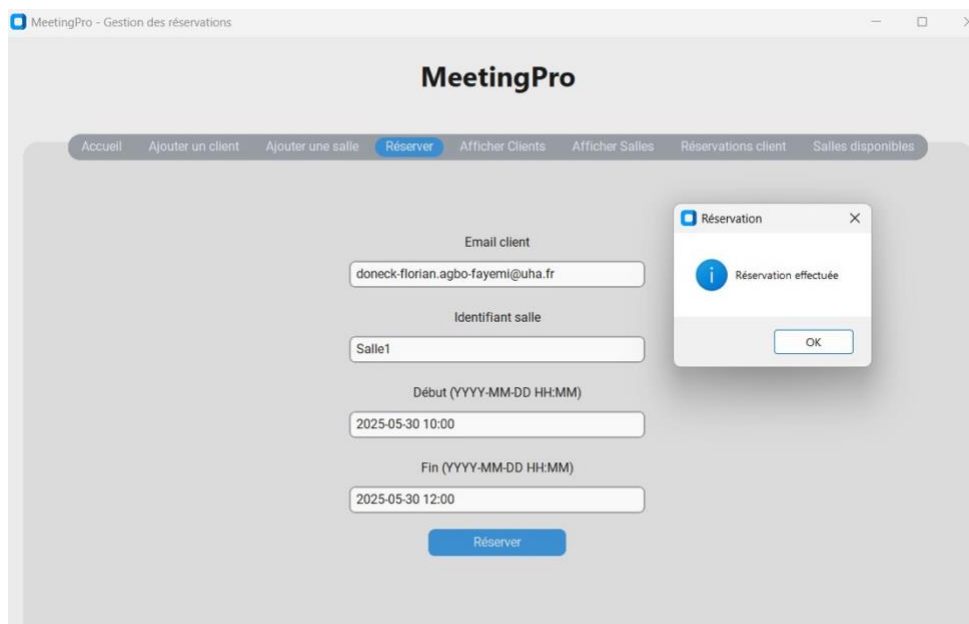


- Onglet “Réserver” :

Pour effectuer une réservation, l’utilisateur doit saisir :

- L’email du client
- L’identifiant de la salle
- La date et l’heure de début (au format YYYY-MM-DD HH:MM)
- La date et l’heure de fin (même format)

Le système vérifie automatiquement les conflits d'horaires et refuse une réservation si la salle est déjà prise. Une alerte de confirmation ou d'erreur s'affiche selon le cas.



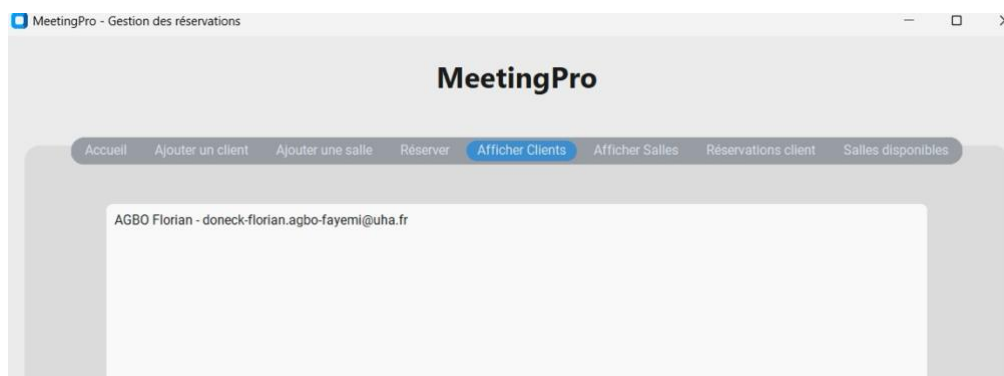
The screenshot shows the 'MeetingPro - Gestion des réservations' application. The 'Réserver' tab is active in the top navigation bar. The form contains the following fields: 'Email client' (filled with 'doneck-florian.agbo-fayemi@uha.fr'), 'Identifiant salle' (filled with 'Salle1'), 'Début (YYYY-MM-DD HH:MM)' (filled with '2025-05-30 10:00'), and 'Fin (YYYY-MM-DD HH:MM)' (filled with '2025-05-30 12:00'). A blue 'Réserver' button is at the bottom. A modal window titled 'Réservation' is open, displaying an information icon and the text 'Réservation effectuée', with an 'OK' button.

- Onglet : “Afficher Clients”

Liste tous les clients enregistrés avec :

- Nom
- Prénom
- Email
- ID unique (UUID)

Utile pour vérifier rapidement les clients en base.

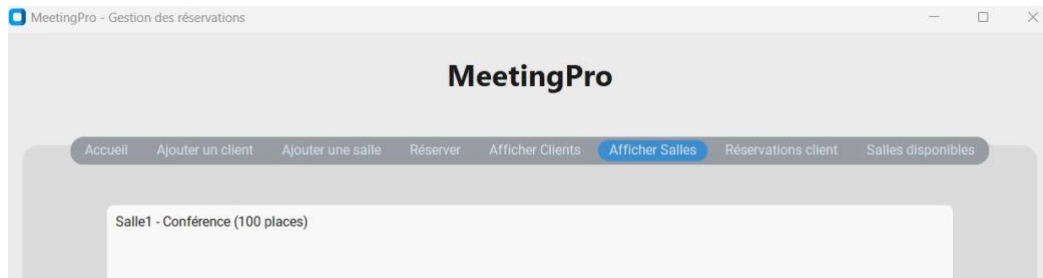


The screenshot shows the 'MeetingPro - Gestion des réservations' application with the 'Afficher Clients' tab selected in the top navigation bar. The main content area displays a single client entry: 'AGBO Florian - doneck-florian.agbo-fayemi@uha.fr'.

- Onglet : “Afficher Salles” :

Liste toutes les salles enregistrées avec :

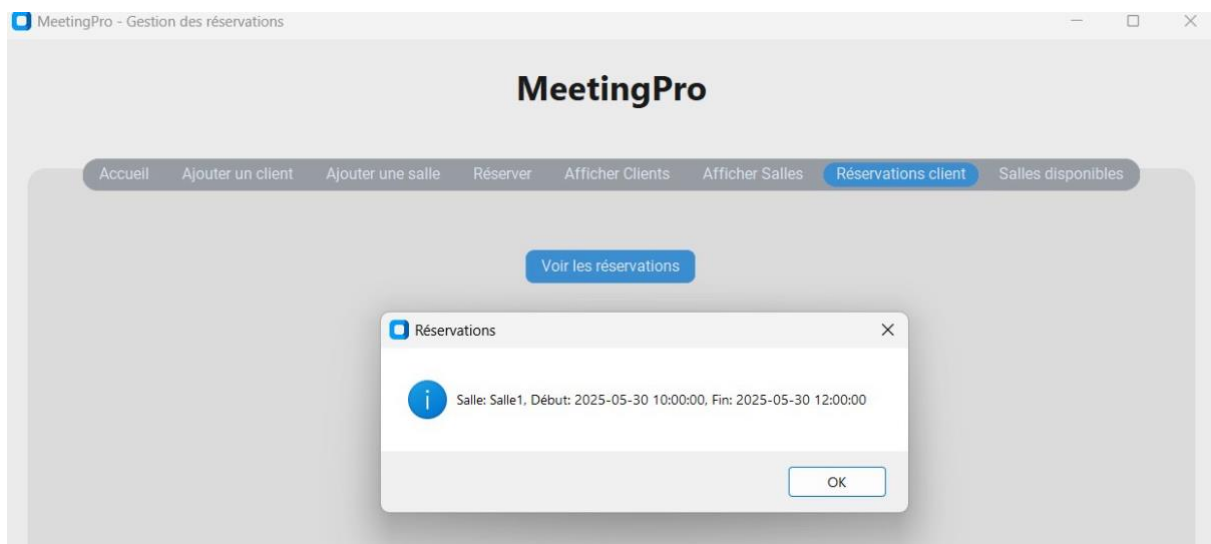
- Identifiant
- Type de salle
- Capacité



- Onglet : “Réservations client” :

Après avoir saisi l’email d’un client, cet onglet affiche toutes ses réservations passées et à venir :

- Identifiant de la salle
- Date et heure de début
- Date et heure de fin





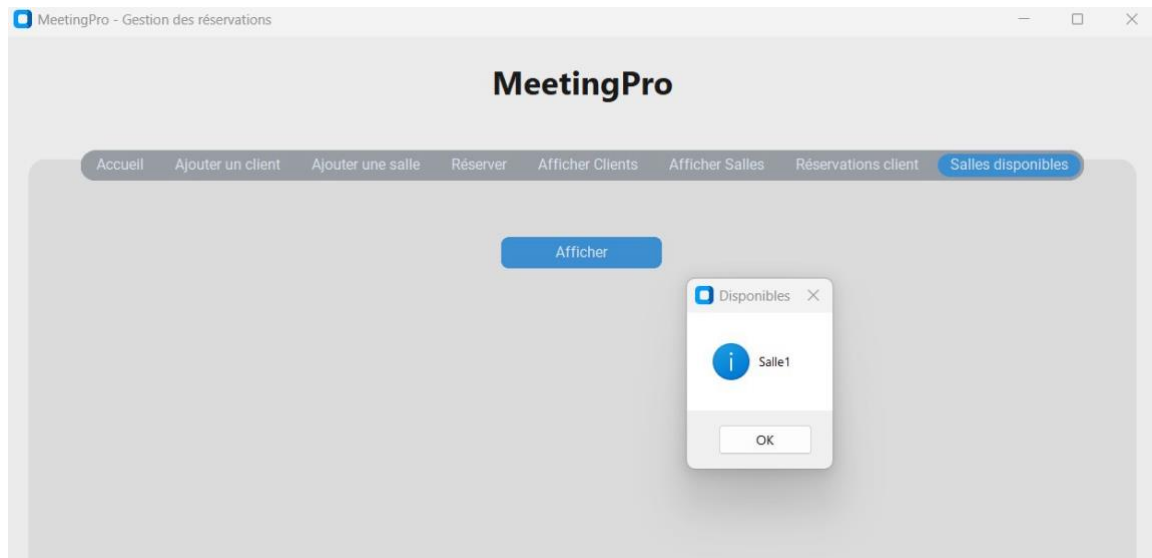
- Onglet : “Salles disponibles”

Permet de rechercher quelles salles sont libres sur un créneau donné.

L'utilisateur saisit :

- Une date/heure de début
- Une date/heure de fin
- 

Le système liste toutes les salles non réservées pendant cette période.



- Navigation et retour à l'accueil

Après chaque opération réussie (ajout, réservation), l'application revient automatiquement à l'onglet d'accueil pour plus de lisibilité.

Tous les champs sont réinitialisés pour éviter les doublons ou erreurs de saisie.

- Notes techniques

Les dates doivent toujours être saisies au format YYYY-MM-DD HH:MM  
(ex : 2025-06-01 14:00)

Le fichier data.json contient toutes les données enregistrées

Le système empêche les conflits de réservation et les doublons d'identifiants

## 5. Problèmes rencontrés et solutions

Problème	Solution apportée
Conflits entre créneaux	Implémentation de vérifications strictes dans le contrôleur
Intégration customtkinter	Refonte complète de la vue, remplacement de tous les widgets Tk
Structuration Git à deux	Création de branches wayne-ui et florian-ui avec pull requests claires
Débogage JSON et typage	Utilisation de uuid, typage dynamique et debug rigoureux

## 6. Ressources utilisées

- [Documentation Python officielle](#)
- [customtkinter GitHub](#)
- [YouTube – MVC Pattern](#)
- Stack Overflow, GitHub Copilot, ChatGPT (OpenAI)

## 7. Conclusion et perspectives

Le projet a été mené de façon structurée et collaborative, avec une interface aboutie et fidèle à la maquette proposée.

L'utilisation de GitHub en équipe a permis une organisation claire des modifications. Si nous devons poursuivre ce projet, nous ajouterions :

- Un système de connexion utilisateur
- Une base de données SQL pour gérer des volumes plus importants
- Un export PDF des réservations ou factures

## 8. Annexes

- README complet (voir dépôt GitHub)
- Code source et architecture
- Tests unitaires exécutables via unittest