📅 JLPT Appointments

Application de gestion de rendez-vous pour les examens JLPT avec réservation de créneaux, vérification par email et gestion via Supabase.

🚀 Installation

1️⃣ Cloner le dépôt

```
git clone https://github.com/yanis-san/jlpt-appointments.git
cd jlpt-appointments
```

2️⃣ Activer bash

```
bash
```

3️⃣ Installer pipenv (si nécessaire)

```
pip install pipenv
```

4️⃣ Installer les dépendances

```
pipenv install
```

5️⃣ Activer l'environnement virtuel

```
pipenv shell
```

🗄️ Configuration de la base de données (Supabase)

📋 Tables

```
-- Création des tables
CREATE TABLE slots (
id bigint GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,
date text NOT NULL,
time text NOT NULL,
available boolean DEFAULT true
);

CREATE TABLE appointments (
id bigint GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,
date text NOT NULL,
time text NOT NULL,
name text NOT NULL,
phone text NOT NULL,
email text NOT NULL,
jlpt_level text NOT NULL,
created_at timestamp with time zone DEFAULT timezone('utc'::text, now()) NOT NULL
);
```

⚡ Index

-- Optimisation avec des index
CREATE INDEX idx_slots_date_time ON slots(date, time);
CREATE INDEX idx_appointments_email ON appointments(email);

🔒 Sécurité et Policies

-- Activer la sécurité au niveau des lignes (RLS)
ALTER TABLE slots ENABLE ROW LEVEL SECURITY;
ALTER TABLE appointments ENABLE ROW LEVEL SECURITY;

-- Policies pour slots
CREATE POLICY "Enable insert for authenticated users only"
ON slots FOR INSERT WITH CHECK (true);

CREATE POLICY "Enable read access for all users"
ON slots FOR SELECT USING (true);

CREATE POLICY "Enable update for authenticated users only"
ON slots FOR UPDATE USING (true) WITH CHECK (true);

-- Policies pour appointments
CREATE POLICY "Enable insert for authenticated users only"
ON appointments FOR INSERT WITH CHECK (true);

CREATE POLICY "Enable read access for all users"
ON appointments FOR SELECT USING (true);

📄 Variables d'environnement .env
📎 Supabase

SUPABASE_URL=votre_url_supabase
SUPABASE_KEY=votre_cle_supabase

✉ Configuration Email (Gmail recommandé)

MAIL_SERVER=smtp.gmail.com
MAIL_PORT=587
MAIL_USE_TLS=True
MAIL_USERNAME=votre_email@gmail.com
MAIL_PASSWORD=votre_mot_de_passe_app
MAIL_DEFAULT_SENDER=votre_email@gmail.com

🔑 Flask

SECRET_KEY=votre_cle_secrete_aleatoire

▶ Démarrage de l'application

pipenv run python app.py

🧪 Tests
🔬 Exécuter tous les tests

pipenv run pytest

📊 Voir la couverture des tests

pipenv run pytest --cov=app tests/

🎯 Exécuter un test spécifique

pipenv run pytest tests/test_app.py::test_save_appointment_form -v

🧰 Structure du projet

jlpt-appointments/
├── app.py # Application principale
├── templates/ # Templates HTML
│ ├── content.html # Formulaire principal
│ ├── error.html # Page d'erreur
│ ├── index.html # Template de base
│ ├── language_buttons.html# Sélecteur de langue
│ ├── slots.html # Créneaux horaires
│ ├── success.html # Confirmation
│ └── verify.html # Vérification email
├── tests/ # Tests unitaires
│ ├── conftest.py # Configuration de test
│ └── test_app.py # Tests principaux
└── .env # Fichier de configuration

✅ Couverture des fonctionnalités et tests
📆 Gestion des créneaux

```
test_get_available_slots
    Vérifie que les créneaux disponibles sont bien retournés
    Exclut les créneaux non disponibles
```

📄 Soumission du formulaire

```
test_save_appointment_form
    Vérifie la soumission du formulaire
    Vérifie l'affichage de la page de vérification après soumission
```

🔐 Génération et vérification de code

```
test_verification_code_generation
    Vérifie que le code généré est bien de 6 chiffres
test_verify_code
    Vérifie la validation d'un code correct
test_invalid_verification_code
    Vérifie que les codes incorrects sont rejetés avec un message adapté
```

✉ Envoi d'email

```
test_email_sending_with_code
    Vérifie que l'email de vérification est bien envoyé
```

📅 Disponibilités

```
test_unavailable_dates
    Vérifie la récupération correcte des dates indisponibles
```

🛠 Initialisation de la base

```
test_initialize_slots
    Vérifie que les créneaux sont bien créés dans Supabase
```

🔄 Flux complet

```
test_save_appointment_email_flow
    Teste le parcours complet de réservation
    De la soumission à la réception de l’email de vérification
```

🤝 Contribution

Les contributions sont les bienvenues !
N’hésitez pas à ouvrir une issue ou proposer une pull request.

📪 Notes

```
Pour les emails via Gmail, pensez à créer un mot de passe d’application
(indispensable si la double authentification est activée).
Supabase est utilisé comme backend sans serveur pour la gestion des créneaux et
des rendez-vous.
Flask est utilisé pour le serveur principal et la logique de vérification.
```