Struktur Folder Project

Ini dia struktur folder yang bisa kamu gunakan untuk proyek ini:

```
instagram-cupit-series/
├── static/
│   ├── css/
│   │   └── style.css          # File CSS untuk styling umum
│   ├── js/
│   │   └── script.js          # File JavaScript untuk interaksi di sisi
user
│   └── images/
│       └── background.jpg     # Gambar-gambar untuk "coffee shop vibes"
├── templates/
│   ├── admin/
│   │   └── index.html         # Halaman admin untuk generate OTP
│   ├── user/
│   │   ├── otp.html           # Halaman input OTP untuk user
│   │   ├── form_step1.html    # Form input username IG
│   │   ├── form_step2.html    # Form pilih gender
│   │   ├── form_step3.html    # Form pilih rentang usia
│   │   └── result.html        # Halaman hasil pairing
│   └── base.html              # Template dasar (header, footer, dll.)
├── app.py                     # File utama aplikasi Flask
├── database.py                # Skrip untuk inisialisasi database
├── requirements.txt           # Daftar library Python yang dibutuhkan
└── README.md                  # Penjelasan project
```

Skema Tabel Database

Kita akan menggunakan SQLite karena lebih mudah untuk pengembangan awal dan cukup kuat.

Tabel `otps`

Tabel ini akan menyimpan OTP yang dibuat oleh admin.

Kolom	Tipe Data	Deskripsi
`id`	INTEGER PRIMARY KEY AUTOINCREMENT	ID unik untuk setiap OTP
`otp_code`	TEXT NOT NULL	Kode OTP 6 digit
`created_at`	TIMESTAMP DEFAULT CURRENT_TIMESTAMP	Waktu OTP dibuat
`expires_at`	TIMESTAMP NOT NULL	Waktu OTP kedaluwarsa
`is_used`	BOOLEAN DEFAULT 0	Status OTP sudah digunakan atau belum (0=belum, 1=sudah)

Tabel `users`

Tabel ini akan menyimpan data pengguna yang mengisi form.

Kolom	Tipe Data	Deskripsi
`id`	INTEGER PRIMARY KEY AUTOINCREMENT	ID unik untuk setiap pengguna
`instagram_username`	TEXT NOT NULL	Username Instagram yang diinput
`gender`	TEXT NOT NULL	Gender pengguna (Laki-laki/Perempuan)
`age_range`	TEXT NOT NULL	Rentang usia pengguna (misal: 17-19)
`created_at`	TIMESTAMP DEFAULT CURRENT_TIMESTAMP	Waktu data pengguna disimpan

---

Alur Pembuatan Menggunakan Flowchart A-B-C

Flowchart A: Admin (Barista) Membuat OTP

Cuplikan kode

```
graph TD
    A[Mulai] --> B{Admin Akses Halaman OTP};
    B --> C[Klik Tombol "Generate OTP"];
    C --> D[Sistem Membuat OTP 6 Digit Acak];
    D --> E[Sistem Menentukan Waktu Kedaluwarsa (5 Menit dari Sekarang)];
    E --> F[OTP dan Waktu Kedaluwarsa Disimpan ke Database 'otps'];
    F --> G[OTP Ditampilkan di Halaman Admin];
    G --> H[Admin Mencetak/Memberikan OTP ke Customer];
    H --> I[Selesai];
```

Flowchart B: Customer Memasukkan OTP dan Mengisi Form

Cuplikan kode

```
graph TD
    A[Mulai] --> B{Customer Scan QR Code/Akses Website};
    B --> C[Customer Diarahkan ke Halaman Input OTP];
    C --> D[Customer Memasukkan OTP];
    D --> E{Validasi OTP: <br/>1. OTP Ada di Database? <br/>2. Belum
Digunakan? <br/>3. Belum Kedaluwarsa?};
    E -- Tidak Valid --> C;
    E -- Valid --> F[OTP Ditandai Sebagai 'Digunakan' di Database];
    F --> G[Customer Diarahkan ke Form Step 1: Input Username IG];
    G --> H[Customer Diarahkan ke Form Step 2: Pilih Gender];
    H --> I[Customer Diarahkan ke Form Step 3: Pilih Rentang Usia];
    I --> J[Data Customer (Username, Gender, Usia) Disimpan ke Database
'users'];
    J --> K[Selesai];
```

Flowchart C: Alur Logic Pairing (Setelah Form Diisi)

Cuplikan kode

```
graph TD
    A[Mulai Pairing] --> B{Sistem Mengambil Data Customer Saat Ini};
    B --> C{Sistem Mencari Data di Tabel 'users'};
    C --> D{Filter Data: <br/>1. Gender BERLAWANAN dengan Customer Saat
Ini <br/>2. Rentang Usia SAMA dengan Customer Saat Ini <br/>3. TIDAK
TERMASUK Customer Saat Ini};
```

```
D --> E{Apakah Ada Hasil Pairing?};
E -- Tidak Ada --> F[Tampilkan Pesan "Belum Ada Pasangan yang Cocok"];
E -- Ada --> G[Tampilkan Daftar Username Instagram yang Cocok];
G --> H[Selesai Pairing];
```

Fitur Tambahan dan Pertimbangan Keamanan

- **QR Code**: Setelah website di-hosting, kamu bisa membuat QR Code yang mengarah ke URL website-mu.
- **Desain Modern & UI Smooth**: Ini akan lebih banyak diatur di sisi **CSS** dan **JavaScript**. Gunakan framework CSS seperti Bootstrap atau Tailwind CSS untuk mempercepat proses desain. Untuk smooth step-by-step UI, kamu bisa pakai JavaScript untuk menampilkan/menyembunyikan form langkah per langkah.
- **Keamanan Ringan**:
    - **Batas Waktu OTP**: Sudah kita masukkan di skema database.
    - **Validasi Username**: Lakukan validasi sederhana di sisi frontend (JavaScript) dan backend (Python) untuk memastikan format username Instagram valid (misal: tidak ada spasi, karakter tertentu).
    - **Tidak Perlu Login IG Asli**: Konsep ini sudah sesuai, hanya input string username.

Oke, mari kita jabarkan alur pembuatan isi setiap file dan folder sesuai struktur yang sudah disepakati. Ini akan memberimu gambaran langkah demi langkah apa saja yang perlu diisi di setiap bagian.

Alur Pembuatan Isi File dan Folder

A. Persiapan Awal (File Pendukung)

1. `requirements.txt`:
    - **Tujuan**: Daftar library Python yang dibutuhkan agar proyek bisa berjalan.
    - **Isi**: Cukup tambahkan `Flask` dan `Flask-SQLAlchemy` (jika pakai SQLAlchemy untuk database) atau `Flask-Session` (untuk sesi jika perlu).
    - **Contoh**:
    - `Flask`
    - `Flask-SQLAlchemy # Jika menggunakan ORM SQLAlchemy`
    - `# Atau jika tidak menggunakan ORM, cukup Flask saja`
2. `database.py`:
    - **Tujuan**: Skrip untuk membuat tabel `otps` dan `users` di database SQLite (atau MySQL). Ini hanya perlu dijalankan sekali saat setup awal.
    - **Isi**:
        - Import modul `sqlite3`.
        - Fungsi untuk membuat koneksi ke database.
        - Fungsi untuk membuat tabel `otps` dengan kolom `id`, `otp_code`, `created_at`, `expires_at`, `is_used`.
        - Fungsi untuk membuat tabel `users` dengan kolom `id`, `instagram_username`, `gender`, `age_range`, `created_at`.
        - Logika utama untuk memanggil fungsi pembuatan tabel saat skrip dijalankan.
    - **Contoh Fungsi (Skema Dasar)**:

Python

```python
import sqlite3

DATABASE = 'cupit_series.db'

def create_tables():
    conn = sqlite3.connect(DATABASE)
    cursor = conn.cursor()

    cursor.execute('''
        CREATE TABLE IF NOT EXISTS otps (
            id INTEGER PRIMARY KEY AUTOINCREMENT,
            otp_code TEXT NOT NULL,
            created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
            expires_at TIMESTAMP NOT NULL,
            is_used BOOLEAN DEFAULT 0
        )
    ''')

    cursor.execute('''
        CREATE TABLE IF NOT EXISTS users (
            id INTEGER PRIMARY KEY AUTOINCREMENT,
            instagram_username TEXT NOT NULL,
            gender TEXT NOT NULL,
            age_range TEXT NOT NULL,
            created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
        )
    ''')
    conn.commit()
    conn.close()

if __name__ == '__main__':
```

```
    create_tables()
    print("Database tables created successfully.")
```

3. `README.md`:
    - **Tujuan**: Penjelasan singkat tentang proyek, cara menjalankan, dan fitur-fiturnya.
    - **Isi**:
        - Judul proyek "Instagram Cupit Series".
        - Deskripsi singkat (Digital Matchmaking untuk Pengunjung Coffee Shop).
        - Fitur utama (sisi admin, sisi user, pairing logic).
        - Cara menjalankan (install requirements, inisialisasi database, run app.py).
        - Link penting (misal: `/admin` dan `/user`).

B. Desain Tampilan (Folder `static/` dan `templates/`)

1. `static/css/style.css`:
    - **Tujuan**: Mengatur tampilan visual keseluruhan website.
    - **Isi**:
        - Styling dasar untuk body (font, background `background.jpg`).
        - Styling untuk form, tombol, input, dan kontainer agar terlihat modern dan "coffee shop vibes".
        - Desain responsif untuk berbagai ukuran layar.
        - Animasi atau transisi untuk UI smooth (misal: saat pindah step form).
2. `static/js/script.js`:
    - **Tujuan**: Menambahkan interaktivitas di sisi klien, terutama untuk form multi-step.
    - **Isi**:
        - Fungsi untuk menangani perpindahan antar langkah form (sembunyikan/tampilkan div langkah).
        - Validasi input sederhana di sisi klien (misal: cek format username IG).
        - Mungkin interaksi AJAX sederhana untuk mengirim data form tanpa reload halaman (opsional, bisa juga dengan submit form biasa).
3. `static/images/background.jpg`:
    - **Tujuan**: Gambar latar belakang atau ikon yang mendukung tema coffee shop.
    - **Isi**: File gambar (`.jpg`, `.png`).
4. `templates/base.html`:
    - **Tujuan**: Template dasar HTML yang akan diwarisi oleh halaman-halaman lain. Ini menghindari pengulangan kode HTML yang sama (header, footer, link CSS/JS).
    - **Isi**:
        - Deklarasi `<!DOCTYPE html>`, `<html>`, `<head>`, `<body>`.
        - Link ke `style.css`.
        - Blok untuk `title` dan `content` yang akan diisi oleh template turunannya.
        - Link ke `script.js` di bagian bawah `<body>`.
    - **Contoh Struktur**:

```
HTML
<!DOCTYPE html>
<html lang="id">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
```

```
    <title>{% block title %}Cupit Series{% endblock %}</title>
    <link rel="stylesheet" href="{{ url_for('static',
filename='css/style.css') }}">
</head>
<body>
    <div class="container">
        {% block content %}{% endblock %}
    </div>
    <script src="{{ url_for('static', filename='js/script.js')
}}"></script>
</body>
</html>
```

5. **`templates/admin/index.html`**:
   - **Tujuan**: Halaman untuk admin (barista) membuat dan melihat OTP.
   - **Isi**:
     - Mewarisi dari `base.html`.
     - Tombol "Generate OTP".
     - Area untuk menampilkan OTP yang baru dibuat.
     - Mungkin daftar OTP yang masih aktif (opsional).
     - Instruksi atau informasi untuk admin.
   - **Contoh**:

     HTML
     ```
     {% extends 'base.html' %}
     {% block title %}Admin - Generate OTP{% endblock %}
     {% block content %}
         <h1>Panel Admin</h1>
         <button id="generate-otp-btn">Generate OTP</button>
         <p>OTP Anda: <strong id="otp-display"></strong></p>
         <p>Masa berlaku: 5 menit</p>
         <p>Berikan OTP ini kepada customer.</p>
     {% endblock %}
     ```

6. **`templates/user/otp.html`**:
   - **Tujuan**: Halaman pertama yang diakses user untuk memasukkan OTP.
   - **Isi**:
     - Mewarisi dari `base.html`.
     - Form input untuk OTP.
     - Tombol "Verifikasi OTP".
     - Pesan error jika OTP salah/kedaluwarsa.
   - **Contoh**:

     HTML
     ```
     {% extends 'base.html' %}
     {% block title %}Verifikasi OTP{% endblock %}
     {% block content %}
         <h1>Selamat Datang di Cupit Series!</h1>
         <p>Masukkan kode OTP dari barista:</p>
         <form action="{{ url_for('verify_otp') }}" method="POST">
             <input type="text" name="otp_code"
     placeholder="Masukkan OTP" required maxlength="6">
             <button type="submit">Verifikasi</button>
         </form>
         {% if error %}
             <p class="error-message">{{ error }}</p>
         {% endif %}
     {% endblock %}
     ```

7. `templates/user/form_step1.html` - `form_step3.html`:
    - **Tujuan**: Masing-masing halaman untuk satu langkah pengisian form.
    - **Isi**:
        - Mewarisi dari `base.html`.
        - Input khusus untuk setiap langkah:
            - `form_step1.html`: Input teks untuk **username Instagram** (@username).
            - `form_step2.html`: Radio button untuk **Gender** (Laki-laki/Perempuan).
            - `form_step3.html`: Dropdown atau radio button untuk **Rentang Usia** (17-19, 20-22, dst.).
        - Tombol "Lanjut" dan "Kembali" (jika pakai JS untuk multi-step).
        - Ini bisa jadi 3 file terpisah, atau satu file dengan JavaScript yang mengatur visibilitas setiap langkah. Untuk kesederhanaan awal, 3 file terpisah dengan redirect lebih mudah.
8. `templates/user/result.html`:
    - **Tujuan**: Menampilkan hasil pairing.
    - **Isi**:
        - Mewarisi dari `base.html`.
        - List atau kartu yang menampilkan username Instagram yang cocok.
        - Pesan jika tidak ditemukan pasangan.
    - **Contoh**:

HTML
```
{% extends 'base.html' %}
{% block title %}Hasil Pairing{% endblock %}
{% block content %}
    <h1>Pasangan Potensial Anda:</h1>
    {% if matches %}
        <ul>
            {% for user in matches %}
                <li><a href="https://instagram.com/{{
user.instagram_username }}" target="_blank">@{{
user.instagram_username }}</a></li>
            {% endfor %}
        </ul>
    {% else %}
        <p>Maaf, belum ada pasangan yang cocok saat ini. Coba
lagi nanti!</p>
    {% endif %}
    <p>Tetap semangat dan nikmati kopimu!</p>
{% endblock %}
```

C. Logika Aplikasi (File `app.py`)

Ini adalah inti aplikasi Flask yang akan menghubungkan semua bagian.

1. **Inisialisasi Flask & Database**:
    - Import `Flask`, `render_template`, `request`, `redirect`, `url_for`, `session`, `uuid` (untuk OTP), `datetime`.
    - Inisialisasi aplikasi Flask.
    - Konfigurasi database (misal: `app.config['DATABASE'] = 'cupit_series.db'`).
    - Setup sesi Flask (untuk menyimpan status OTP valid sementara).
2. **Fungsi Pembantu Database**:

- Fungsi untuk mendapatkan koneksi database (`get_db()`).
- Fungsi untuk menutup koneksi database (`close_db()`).
- Fungsi untuk menjalankan query database.

3. **Rute untuk Sisi Admin (`/admin`)**:
   - **`GET /admin`**:
     - Render `templates/admin/index.html`.
   - **`POST /admin/generate_otp`**:
     - Generate OTP 6 digit acak (misal: pakai `random.randint`).
     - Hitung `expires_at` (current_time + 5 menit).
     - Simpan OTP, `created_at`, `expires_at`, `is_used=0` ke tabel `otps`.
     - Return OTP yang digenerate (bisa dalam bentuk JSON atau langsung render ulang halaman admin).
4. **Rute untuk Sisi User (`/`)**:
   - **`GET /`**:
     - Render `templates/user/otp.html`.
   - **`POST /verify_otp`**:
     - Ambil OTP dari form.
     - Query database `otps`: cari OTP yang `otp_code`nya cocok, `is_used=0`, dan `expires_at` belum lewat.
     - **Jika valid**:
       - Update `is_used=1` di tabel `otps` untuk OTP tersebut.
       - Set sesi `otp_valid = True`.
       - Redirect ke `/form/step1`.
     - **Jika tidak valid**:
       - Render `templates/user/otp.html` lagi dengan pesan error.
5. **Rute untuk Form Multi-Step (Setelah OTP Valid)**:
   - **Middleware/Decorator**: Buat fungsi pembantu untuk mengecek `session['otp_valid']` sebelum mengakses rute form. Jika tidak valid, redirect ke `/`.
   - **`GET /form/step1`**:
     - Cek sesi `otp_valid`.
     - Render `templates/user/form_step1.html`.
   - **`POST /form/step1`**:
     - Ambil username IG dari form.
     - Validasi format username (regex sederhana).
     - Simpan username ke sesi sementara.
     - Redirect ke `/form/step2`.
   - **`GET /form/step2`**:
     - Cek sesi `otp_valid`.
     - Render `templates/user/form_step2.html`.
   - **`POST /form/step2`**:
     - Ambil gender dari form.
     - Simpan gender ke sesi sementara.
     - Redirect ke `/form/step3`.
   - **`GET /form/step3`**:
     - Cek sesi `otp_valid`.
     - Render `templates/user/form_step3.html`.
   - **`POST /form/submit`** **(atau `/form/step3` yang menangani submit akhir)**:
     - Ambil rentang usia dari form.
     - Ambil semua data (username, gender, age_range) dari sesi.
     - Simpan data lengkap ke tabel `users`.

        - **Lakukan Logic Pairing**:
            - Ambil gender pengguna saat ini.
            - Ambil rentang usia pengguna saat ini.
            - Query tabel `users`:
                - `WHERE gender != current_user_gender`
                - `AND age_range = current_user_age_range`
                - `AND id != current_user_id` (untuk memastikan tidak menampilkan diri sendiri)
            - Ambil hasil pairing.
        - Render `templates/user/result.html` dengan data hasil pairing.
        - Hapus sesi `otp_valid` (agar OTP tidak bisa dipakai lagi).
6. **Penanganan Error**:
    - Tambahkan rute untuk error 404 (Not Found).
    - Error handling umum untuk database.
