TalentScout Intelligent Hiring Assistant

TalentScout is an **AI-powered recruitment screening assistant** built with **Streamlit** and **Google Gemini API**. It simulates an initial recruiter conversation: gathering candidate details, asking tailored technical questions, analyzing sentiment, and storing structured candidate data.

Features

- **Conversational Chatbot**: Conducts a natural interview-like conversation with candidates.
- **Candidate Data Capture**: Collects name, email, phone, experience, location, desired role, and tech stack.
- **AI-Generated Technical Questions**: Creates 3–5 tailored technical questions per candidate's skills.
- **Sentiment Analysis**: Analyzes candidate responses using TextBlob (with option to integrate Hugging Face models).
- **Admin Panel (Sidebar)**: View collected candidate data, sentiment logs, and clear/reset the chat.
- **Dark Mode UI**: Custom CSS styling for a sleek, modern appearance.
- **Data Persistence**: Saves candidate information locally (`candidate_data.json`).

Project Structure

```
project-root/
|-- app.py                  # Main Streamlit app
|-- candidate_data.json     # Stores anonymized candidate data
|-- requirements.txt        # Python dependencies
|-- README.md               # Project documentation
```

Installation & Setup

1. Clone Repository

```
git clone https://github.com/yourusername/talentscout-assistant.git
cd talentscout-assistant
```

2. Install Dependencies

```
pip install -r requirements.txt
```

3. API Key Setup

Obtain a **Google Gemini API key** from Google AI Studio.

- Option 1: Use **Streamlit Secrets Manager** Add in `.streamlit/secrets.toml`:

```
GEMINI_API_KEY = "your_api_key_here"
```

- Option 2: Use `.env` file

```
GEMINI_API_KEY=your_api_key_here
```

Running the App

Start the Streamlit app:

```
streamlit run app.py
```

Then open your browser at http://localhost:8501.

How It Works

1. Candidate is greeted with a **welcome message**.
2. Assistant asks for candidate details step by step (name, email, phone, etc.).
3. Based on candidate's tech stack, **tailored technical questions** are generated.
4. Candidate responses are analyzed (with **sentiment analysis**) and logged.
5. Collected candidate profile is stored in `candidate_data.json`.
6. Admins can view logs and reset sessions from the sidebar.

Admin Panel

The sidebar includes:

- **Candidate Data** → JSON view of extracted details.
- **Sentiment Log** → Input + sentiment classification.
- **Reset Button** → Clears all stored data and restarts conversation.

UI Styling

- **Dark mode** with black background.
- **Sidebar** styled with blue highlights (`#1a73e8`).
- **Buttons** with gradient hover effect.

Future Enhancements

- Use **Hugging Face pre-trained models** for:
  - Sentiment (BERT-based)
  - Named Entity Recognition (NER) for auto-extracting candidate info
  - Zero-shot classification for inferring desired roles and skills
- Store data in a **database** (e.g., MongoDB, Firebase) instead of JSON file.
- Add **email notifications** to recruiters after candidate completion.
- Multi-language support.

License