# EVAT Chatbot (Australia) — Setup & Run Guide

## Overview

A sprint-built assistant that finds EV charging stations across Australia, personalises results, and plans trips with charging stops and sightseeing.  
  
- Sprint 1: NLP for POIs/landmarks  
- Sprint 2: Charger search + geocoding + AU bounding  
- Sprint 3: Personalisation (preferences + history)  
- Sprint 4: Trip planning (OSRM routing, corridor/detour, sightseeing via Overpass)

## Features

- Understands queries like 'charge near Melbourne Airport'  
- Searches nationwide EV chargers (Open Charge Map), caches to CSV  
- Personalised ranking: distance, plug types, kW, and recent use  
- Trip planner: shortest route vs. enhanced with charging stops  
- Sightseeing suggestions near stops / along the route  
- Simple web UI with chat, history, and a 'Searching…' loader

## Repo Layout

EVAT\_Flask/  
├─ app.py # Flask server + routes  
├─ evat\_core.py # NLP, data fetch/cache, ranking, routing, sightseeing  
├─ templates/  
│ └─ index.html # Chat UI  
├─ static/ # CSS/JS  
├─ Key.env.example # Sample env file (copy to Key.env)  
├─ ev\_data/ # Cached CSVs (auto-created)  
└─ user\_state/ # User profiles/history (auto-created)

## Prerequisites

- Python 3.9–3.11 (Conda recommended)  
- Internet access for first data fetch  
- Optional API keys:  
 - OCM\_API\_KEY – Open Charge Map  
 - GOOGLE\_API\_KEY – Google Maps Geocoding (OSM fallback exists)

## Quick Start

1) Create environment & install:  
 conda create -n evat\_chatbot python=3.11 -y  
 conda activate evat\_chatbot  
 pip install flask spacy geopy googlemaps pandas requests python-dotenv  
 python -m spacy download en\_core\_web\_sm  
  
2) Configure keys: Create Key.env file in the project folder.  
  
3) Run:  
 conda activate evat\_chatbot  
 python app.py  
  
 Open http://127.0.0.1:5000 in your browser.

## Trimester Summary

- Sprint 1 – NLP: spaCy NER + regex + filler removal to capture POIs/landmarks.  
- Sprint 2 – Data & Search: Open Charge Map fetch with pagination, AU bounding box filter, geodesic distance ranking.  
- Sprint 3 – Personalisation: User preferences (plug types, min kW, max distance), recency boost, JSON profile/history.  
- Sprint 4 – Trip Planning: OSRM routing, range/reserve/corridor/detour logic, sightseeing, dual plan output.