AI Astrologer Application: Project Summary and Setup

August 18, 2025

1 Project Overview

The AI Astrologer is a web application that collects user birth details (Name, Date, Time, Place) through a clean UI, calculates astrological signs (Sun, Moon, Ascendant) and personality traits, and provides astrology-based responses to free-text questions. It meets requirements for a user-friendly interface, rule-based and AI-driven output, and question handling

2 Technical Components

2.1 Frontend (`index.html`)

- Built with HTML, CSS, and JavaScript.
- Features a form for birth details and a question input field with a placeholder (e.g., "Visakhapatnam, India") to guide correct input.
- Uses AJAX (`fetch`) to send data to `/process` and `/process_message` routes.
- Styled with responsive, inline CSS.

2.2 Backend (`app.py`)

- **Framework**: Flask with CORS for frontend communication.
- **Astrology**: `pyswisseph` calculates Sun, Moon, and Ascendant positions using Julian date.
- **Geolocation**: `geopy` (Nominatim, 10-second timeout) converts place names to coordinates with a cache, spelling corrections, and country fallback; `timezonefinder` and `pytz` handle timezone conversions.
- **Traits**: PyTorch neural network (`TraitModel`) predicts five traits using dummy data.
- **Questions**: `nltk` tokenizes questions for rule-based responses based on keywords.

- **Error Handling**: Logging, try-catch blocks, and rate limit handling manage errors.
- **Storage**: Global `user_data` dictionary stores user details.

2.3 Dependencies (`requirements.txt`)

- Includes `flask`, `pyswisseph`, `geopy`, `torch`, `nltk`, `gunicorn`, `numpy`.
- Requires Swiss Ephemeris files in `ephe/`.

3 Setup Instructions

1. **Verify Project Structure**: Ensure `app.py`, `requirements.txt`, `static/index.html`, and `ephe/` (with `sepl_18.se1`, `semo_18.se1`, `seas_18.se1`) are in `C:\Users\admin\D`
2. **Set Up Virtual Environment**:
   - Run: `python -m venv env`
   - Activate: `env\Scripts\activate`
3. **Install Dependencies**: Run `pip install -r requirements.txt`.
4. **Download Ephemeris Files**: Place `sepl_18.se1`, `semo_18.se1`, `seas_18.se1` in `ephe/` from `ftp://ftp.astro.com/pub/swisseph/ephe/`.
5. **Run Application**: Run `python app.py` and open `http://127.0.0.1:5000`.
6. **Test**: Use inputs like Name: Srikar Pilla, Date: 2004-09-12, Time: 22:30, Place: Visakhapatnam, India. Wait 1–2 seconds between submissions.

4 Challenges and Solutions

- **Syntax Error**: Fixed a malformed string in `horoscopes`.
- **Geolocation Issues**: Specific places (e.g., "Vishakapatanam") failed due to spelling errors or Nominatim timeouts. Solutions:
  - Added a geolocation cache to reduce repeated requests.
  - Implemented spelling corrections (e.g., "Vishakapatanam" to "Visakhapatnam, India").
  - Used country fallback (e.g., "India") if specific places fail.
  - Increased timeout to 10 seconds and enforced rate limits.

5 Key Features

- **Input**: UI for birth details and questions, with placeholder for correct place format.

- **Output**: Sun Sign, Moon Sign, Ascendant, and AI-predicted traits.
- **Questions**: Personalized responses based on keywords and user data.
- **Robustness**: Handles errors with spelling corrections and fallback.

6 Deliverables

- **Codebase**: Zip or GitHub repository with all files.
- **Demo Video**: 2–5 minute video showing input (e.g., "Visakhapatnam, India"), results, and question handling, recorded with OBS Studio and uploaded to Google Drive.
- **PDF Report**: Generated from this LaTeX document.