Treg Research Assistant - Walkthrough

Overview

This project implements a **Treg Research Assistant**, a "Concierge Agent" for bioinformaticians. It uses a RAG (Retrieval-Augmented Generation) pipeline to answer questions based on PubMed abstracts and ClinicalTrials.gov data.

Project Structure

The project is divided into `backend` (Python/FastAPI) and `frontend` (React/Vite).

```
capstone_project/
├── backend/
│   ├── data/
│   │   └── raw_data.json       # Ingested data
│   ├── api_server.py           # FastAPI server
│   ├── data_ingestion.py       # Data fetching script
│   ├── rag_agent.py            # LlamaIndex agent logic
│   └── requirements.txt        # Python dependencies
└── frontend/
    ├── src/
    │   ├── App.jsx             # Chat Interface
    │   └── ...
    ├── package.json
    └── ...
```

Setup & Verification

1. Backend Setup

Prerequisites: Python 3.9+, Google API Key.

1. Navigate to `backend`:

   ```
   cd backend
   ```

2. Install dependencies:

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

3. Set API Key:

```
export GOOGLE_API_KEY="AIza..."
```

4. Run Data Ingestion (already run, but to refresh):

   ```
   python data_ingestion.py
   ```

   Verification: Check `backend/data/raw_data.json` exists and contains data.
5. Start the API Server:

   ```
   python api_server.py
   ```

   Verification: Server starts at `http://0.0.0.0:8000`. Visit `http://localhost:8000/health` to confirm.

2. Frontend Setup

Prerequisites: Node.js 18+.

1. Navigate to `frontend`:

   ```
   cd frontend
   ```

2. Install dependencies:

   ```
   npm install
   ```

3. Start the Development Server:

   ```
   npm run dev
   ```

   Verification: App opens at `http://localhost:5173`.

Features Implemented

- **Multi-Agent System**:
  - **Orchestrator**: Gemini 1.5 Pro agent that plans and delegates.
  - **Researcher**: Gemini 1.5 Flash agent for PubMed/ClinicalTrials retrieval.
  - **Analyst**: Gemini 1.5 Flash agent for data analysis and code execution.
- **Tools**:
  - **Retrieval**: Custom tools for PubMed and ClinicalTrials.gov.

    - **Analysis**: Python code execution sandbox.
- **Configurable Models**: Switch between Pro and Flash models via UI.
- **Architecture**: Google ADK patterns with Vertex AI Reasoning Engine.

Tech Stack

- **Backend**: Python, FastAPI, Vertex AI SDK, Google ADK.
- **AI Engine**: Google Gemini 1.5 Pro & Flash.
- **Frontend**: React, Vite, TailwindCSS.
- **Data**: PubMed E-utilities, ClinicalTrials.gov API.