🛰️ Requirements Document – AstroPath

📌 Project Title

AstroPath – Intelligent Multi-Agent Platform for Space Mission Planning & Astronomical Event Guidance

---

👩‍💼 Stakeholders

Stakeholder	Role/Interest
Client (SpaceEdTech)	Wants a tool to teach and simulate space missions for students & enthusiasts
End Users	Students, educators, astronomy hobbyists, and space enthusiasts
Project Team	Developers, AI/ML engineers, designers
API Providers	NASA, OpenWeather, Google Maps, etc.

🎯 Objectives

- Create a **web-based platform** that uses **multi-agent AI** to help users:
    - Plan missions (e.g., Earth to Mars)
    - Visualize orbits and launch windows
    - Get alerts for real-time astronomical events
    - Interact with a chatbot for mission advice
- Make it **educational, interactive, and based on real-world data**.

💡 Core Features / Modules

1. 🚀 Trajectory Agent

- Calculates optimal interplanetary travel routes using physics (e.g., Hohmann transfer).
- Uses libraries like **Skyfield**, **Astroquery**, or **NASA Horizons API**.
- Supports real-time trajectory plotting (via **Plotly** or **Three.js**).

2. 📆 Launch Window Agent

- Calculates ideal launch windows based on planetary alignment.
- Uses **ephemeris data** via Skyfield/Astroquery.
- Visualizes date ranges on an interactive timeline.

3. 🧮 Fuel Agent

- Computes fuel requirements using Δv calculations.
- Takes spacecraft type into account.

- Displays simulated fuel consumption curve.

4. 🌦️ Weather Agent

- Fetches weather for selected launch sites (Cape Canaveral, ISRO SHAR).
- Uses **OpenWeatherMap** API.
- Shows clear/cloudy conditions and alerts for storm conditions.

5. 🧑‍🚀 Mission Control Agent

- Main user-facing chatbot interface.
- Accepts mission specs (origin, destination, type, constraints).
- Coordinates other agents and returns mission plan summary.
- Optionally powered by **Dialogflow CX + Gemini API**.

6. 🌌 Event Agent (for AstroAid)

- Shows upcoming events (e.g., ISS pass, eclipses).
- Uses **Heavens Above, NASA APIs**, or **timeanddate.com** data.
- Suggests observation tips based on user location + weather.

🖥️ User Interface Requirements

- **Dashboard**: Mission planner with orbital visualizations, fuel meters.
- **Chatbot UI**: Human-like, educational conversation agent.
- **Mission Timeline**: Gantt or step-based display of the mission journey.
- **Observation Panel**: List of events with visibility/weather status.

- **Responsive design**: Mobile/tablet/desktop support.
- **Dark/Light modes**.

📚 Technical Stack

Area	Technology
Frontend	Next.js, TailwindCSS, Plotly
Backend	Python (FastAPI), Flask (optional)
Database	Firebase Firestore or Realtime DB
Hosting	Firebase Hosting or Google App Engine
APIs	NASA APIs, Skyfield, OpenWeatherMap
AI/Agents	Python agents, Dialogflow, Gemini (if allowed)
3D/Charts	Plotly, Three.js (for orbit views)

🔐 Security Requirements

- User authentication via **Firebase Auth**.
- Role-based access control (basic users vs admin).
- Input validation & protection from injection attacks.

📈 Performance & Scalability

- Real-time computation under 3s for mission plan output.
- Scalable Firebase backend with fallback cache (e.g., planetary data cache).

- Lazy loading of assets and charts.

🎓 Educational Mode (Bonus)

- Guided walkthrough for space physics basics.
- Definitions and animations for key terms (Δv, aphelion, perihelion).
- Quizzes or challenges based on missions planned.

🛠️ Deliverables

Deliverable	Description
AstroPath Web Application	Full frontend and backend platform
User Manual/Documentation	Setup guide, user walkthrough, tech doc
Launch Demo Video	2–3 minute demo video of full system
Presentation Deck (Client Format)	Executive summary + diagrams
GitHub Repo	Clean, well-commented code with README

🧪 Testing & QA

- Unit tests for all agents
- Functional test of mission planner
- UI/UX feedback loop from 5 beta testers

📅 Timeline (Client Milestone Proposal)

Week	Milestone
1	UI Mockups + Agent architecture
2	Trajectory + Launch Window agent ready
3	Weather + Fuel agent integrated
4	Chatbot + AstroAid features added
5	Testing + UX polish + Hosting setup
6	Final presentation + Client walkthrough

✅ Success Criteria

- Users can simulate Earth-to-Mars missions successfully.
- Chatbot can respond to 90% of queries without errors.
- Events and launch suggestions work in real-time.
- 3D orbit visuals render smoothly on devices.