Poppy Universe :
Step-by-Step Action Plan

0. Plan & Prep (1 – 2 days)

1) **MVP vs stretch goals**
   - **MVP** = approximate recommendations only.
   - **Stretch** = Exact positions + interactive sky map
2) Database structure
   - Users, objects, interactions, etc.
   - Think about what fields each table needs for filtering and personalization.
3) **wireframe/flow diagram**
   - Login → recommendations → interaction history
   - Optional: profile page or settings for stretch goal.
4) Data sources list
   - NASA open dataset, open star catalogs, Messier objects, and any other astronomy CSV/JSON files.
5) Technical stack decisions
   - C# for the model
   - Python for Data Science

1. Data Collection (2 – 3 days)

1) **Reliable open datasets**
   - ANSA Open Data portal
   - Star catalogs (bright stars, Messier objects, planets)
   - Astronomy APIs (optional for later)
2) **Download datasets in CSV/JSON format**
3) **Folder Structure**
   - Make a folder structure for the project: raw_data/, cleaned_data/, db_seed/.
4) **Dataset glance**
   - Sneak peek at the dataset: number of objects, fields available, missing values, etc.

2. Data Exploration & cleaning (1 – 2 days)

1) **Inspect Each Dataset**
   - Missing values, duplicates, weird formats
   - Normalize fields:
     - months → Jan/Feb/etc.
     - difficulty → beginner/intermediate / advanced.
2) **Approximate-only vs exact / calculated later**
   - Decide which columns are approximate-only (MVP) and which will be exact / calculated later.
3) **Merge datasets**
   - Merge datasets if needed to have one master object list.
4) **Optional visualization**
   - Make basic visualizations (histograms of objects per hemisphere, month, difficulty) to understand distribution.

---

3. Seed Database (1 – 2 days)

1) **DB schema**
   - Create a DB schema (Users, Objects, Interactions).
2) **Objects table**
   - Populate objects table with 10-20 initial entries: planets, bright stars, a few constellations.
3) **Add a test user account.**
4) **Database Queries**
   - Make sure the database can be queried easily for filtering by month, hemisphere, direction, and difficulty.
5) **Testing that inserting and querying interactions work**

---

4. C# Recommendation Engine (3 – 5 days)

1) **Start a new C# project for the engine**
2) **Implement filtering logic for:**
   - Month visibility
   - Hemisphere
   - Direction (east, north, etc.)
   - Difficulty
   - Popularity
3) **Personalization**
   - Ad optional personalization using interactions: basic ranking based on past likes.
4) **Testing Engine**
   - Test engine with seed DB objects: make sure recommendations make sense.
5) **Engine calls**
   - Ensure that the engine can be called by the backend dynamically

---

5. Backend Setup (3 – 5 days)

1) **Set up JavaScript / Node.js backend**
2) **Connect the backend to the Database**
3) **Login system**
   - Implement login system: hashed passwords, real accounts, no exact location stored.
4) **Recommendation Engine**
   - Connect the backend to the C# recommendation engine so requests return recommendations
5) **Test backend API endpoints with sample queries**

---

6. Basic Frontend MCP (3 – 5 days)

1) **Create simple HTML/CSS/JS pages:**
   - Login/register page
   - Recommendations page showing object name, month, hemisphere, difficulty, direction
   - Optional → allow users to "like" objects to log interactions
2) **Connect frontend to backend API**
3) **Testing**
   - Full login flow
   - Recommendations
   - Interaction logged

---

7. Optional: Exact Positions (2 – 3 days)

1) **Astronomy API**
   - Integrate Astronomy API (lat, long, datetime → altitude, azimuth, magnitude).
2) **Update recommendation engine**
   - Update recommendation engine to use exact position as extra features if available
3) **Ensure fallback to approximate DB values if API fails.**

---

8. Optional: Interactive Frontend (5 – 7 days)

1) **Profile, recommendations, history**
   - Build a Vue + Vite frontend with multiple pages
2) **Interactive Sky map**
   - Implement an interactive sky map showing recommended objects
3) **Appealing UI**
   - Make UI visually appealing, like a Poppy Universe app.
4) **Connection**
   - Connect all frontend features to backend endpoints.

---

9. Optional: Advanced ML Personalization (3 – 5 days)

1) **User interaction**
   - Analyze user interactions to predict preferred objects.
2) **Popularity trends**
   - Track popularity trends per skill level and hemisphere.
3) **Complete the loop**
   - Implement this info back into the recommendation ranking.

---

10. Deployment (2 – 3 days)

1) **Backend & Database**
   - Deploy Backend and Database on Azure (using free student credits).
2) **Host Frontend**
3) **Secure API keys.**
4) **Testing**
   - Test full flow on Azure: Login, Recommendations, interactions, fallback behavior.

END