**🗓️ Week 1 Draft Implementation Plan**

**Day 1: Define Roles and Tools**

1. **Define Agent Roles:**
   * **PM-Agent**: Task coordinator, receives goals from user and delegates work.
   * **Market Research Agent**: Gathers information.
   * **Dev-Agent**: Writes and tests code.
   * **Design-Agent**: Creates UI layouts or visual content.
   * **Analysis-Agent**: Evaluates results, summarizes findings.
   * **Trend**-**Agent:** Evaluates current internet trends.
   * **Content Creation-Agent:** Creates content for social media.
2. **Choose tools:**
   * Language: Python (or JS if using Node)
   * API Comm: HTTP (Flask or FastAPI)
   * Workflow: File-based messages, local API calls, or Firebase Realtime DB
   * IDE: Firebase Studio / VS Code

**Day 2: Set Up Local Agent Framework**

1. **Create folders or modules for each agent** (e.g. pm\_agent/, research\_agent/ etc.)
2. **Install dependencies:**

bash

CopyEdit

pip install flask openai (or gemini, langchain, etc.)

1. **Basic structure for each agent (Python + Flask)**:

python

CopyEdit

from flask import Flask, request, jsonify

app = Flask(\_\_name\_\_)

@app.route('/run', methods=['POST'])

def run\_task():

data = request.json

# Do the task...

return jsonify({'response': 'Task done'})

if \_\_name\_\_ == '\_\_main\_\_':

app.run(port=5001) # Each agent runs on different port

**Day 3: Build the Project Manager Agent**

* Give PM-Agent a UI or terminal interface to receive instructions.
* It will:
  + Receive user goals
  + Decide which agent does what
  + Send instructions to others using HTTP requests

python

CopyEdit

import requests

def assign\_task(agent\_url, payload):

res = requests.post(f"{agent\_url}/run", json=payload)

return res.json()

# Example

assign\_task('http://localhost:5002', {'task': 'Find top 5 AI APIs'})

**Day 4: Implement 2 Specialized Agents**

* **Research-Agent**: Connects to web, summarizes results.
* **Dev-Agent**: Generates or edits code using Gemini/OpenAI.

Include logs or messages the PM can access.

**Day 5: Implement Remaining Agents**

* **Design-Agent**: Generates UI mockups, descriptions.
* **Analysis-Agent**: Evaluates output quality or summaries.

They all receive tasks and return structured responses.

**Day 6: Workflow & Communication Logic**

* Create a **shared log file** or **Firebase Realtime DB** as a message board.
* PM-Agent:
  + Logs tasks given
  + Waits for replies
  + Sends results to next agent

This is your **"workflow chain"**.

Example Firebase logic:

python

CopyEdit

# PM writes task to /tasks/research

# Research-Agent listens to that path and writes response to /results/research

**Day 7: Testing & Optimization**

* Run a real project idea through the agents (e.g. "Build a landing page for AI course")
* Debug task flow
* Add timeout/retry logic
* Optional: Add simple dashboard (React or Terminal UI)

**✅ Final Goal Example Flow**

sql

CopyEdit

User → PM-Agent → Research-Agent → Dev-Agent → Design-Agent → Analyst-Agent → PM-Agent → User