Personal Health Coach — AI Agent

GenAI 4 GenZ · Session 2 Challenge in AI Agents

A health monitoring agent that compresses medical history and wellness data, providing personalized recommendations with lower API processing costs.

1. Project Overview

This project builds an **AI-powered health coaching agent** that accepts a user's medical history and daily wellness data (sleep, steps, mood, meals), **compresses** that data via summarization to reduce token usage, sends the compressed context to Claude for analysis, and returns **personalized health recommendations** covering diet, exercise, and sleep.

The key challenge is *efficient context compression* — enabling the agent to handle weeks or months of health data without hitting token limits or incurring high API costs.

Tech Stack

Layer	Tool
Language	Python 3.10+
LLM API	Anthropic Claude `claude-sonnet-4-5`
Data Storage	JSON / SQLite
Web Interface	Streamlit
Environment	`python-dotenv`

Project Structure

```
personal-health-coach/
          main.py           # Entry point
          agent.py          # Core agent logic
          compressor.py     # Compression module
          recommender.py    # Recommendations
          app.py            # Streamlit web UI
          data/
                sample_health_data.json
          requirements.txt
          .env              # Never commit!
```

2. Step-by-Step Build Guide

Step 1 — Environment Setup

```
# Virtual environment
python -m venv venv
source venv/bin/activate
# Windows: venv\Scripts\activate

# Install packages
pip install anthropic python-dotenv
    streamlit
```

`.env` file:

```
ANTHROPIC_API_KEY=your_api_key_here
```

`requirements.txt`:

```
anthropic>=0.25.0
python-dotenv>=1.0.0
streamlit>=1.30.0
```

Step 2 — Health Data Schema

`data/sample_health_data.json`:

```
{
 "user_id": "user_001",
 "profile": {
   "age": 28, "weight_kg": 70,
   "height_cm": 175,
   "conditions": ["mild anxiety",
                  "iron deficiency"],
   "goals": ["improve sleep",
             "increase energy"]
 },
 "daily_logs": [{
   "date": "2025-02-01",
   "sleep_hours": 6.5,
   "steps": 4200,
   "water_liters": 1.5,
   "mood": "tired",
   "meals": ["oatmeal","pizza","chips"],
   "exercise": "none",
   "notes": "Felt sluggish all day"
 }]
}
```

Step 3 — Data Compressor (Most Important Module)

Key insight: Compressing raw daily logs into a ~300-word snapshot before calling Claude reduces token usage

by **60–80%**, dramatically cutting API costs for long-term users.

compressor.py:

```python
import anthropic, json, os
from dotenv import load_dotenv

load_dotenv()
client = anthropic.Anthropic(api_key=os.getenv("ANTHROPIC_API_KEY"))

def compress_health_data(health_data: dict) -> str:
    """Compresses daily health logs into a concise summary using Claude."""
    raw_logs = json.dumps(health_data["daily_logs"], indent=2)
    profile  = json.dumps(health_data["profile"], indent=2)

    prompt = f"""You are a medical data summarizer. Create a CONCISE health
summary (max 300 words). Focus on: average sleep/steps/hydration, mood
patterns, nutrition quality trends, exercise consistency, concerning patterns.

User Profile: {profile}
Daily Logs ({len(health_data['daily_logs'])} days): {raw_logs}

Output a compressed health snapshot only. No extra commentary."""

    response = client.messages.create(
        model="claude-sonnet-4-5-20250929",
        max_tokens=400,
        messages=[{"role": "user", "content": prompt}]
    )
    return response.content[0].text
```

Step 4 — Recommendation Agent

recommender.py:

```python
import anthropic, os
from dotenv import load_dotenv

load_dotenv()
client = anthropic.Anthropic(
  api_key=os.getenv("ANTHROPIC_API_KEY"))

SYSTEM_PROMPT = """You are a compassionate
expert health coach. Provide:
1. Top 3 personalized recommendations
2. One positive observation
3. One 7-day priority focus area
Tone: warm and non-judgmental."""

def get_recommendations(
        summary: str,
        question: str = None) -> str:
    msg = f"Health summary:\n\n{summary}"
    if question:
        msg += f"\n\nQuestion: {question}"
    response = client.messages.create(
        model="claude-sonnet-4-5-20250929",
        max_tokens=600,
        system=SYSTEM_PROMPT,
        messages=[{"role":"user",
                   "content": msg}]
    )
    return response.content[0].text
```

Step 5 — Agent Pipeline

agent.py:

```python
from compressor import compress_health_data
from recommender import get_recommendations
import json

def run_health_agent(
        data_path: str,
        user_question: str = None):
    """Load -> Compress -> Recommend."""
    with open(data_path, "r") as f:
        health_data = json.load(f)

    compressed = compress_health_data(
        health_data)
    recs = get_recommendations(
        compressed, user_question)

    raw  = len(json.dumps(
        health_data["daily_logs"]))
    comp = len(compressed)
    pct  = (raw - comp) / raw * 100
    print(f"Saved: {pct:.1f}%")

    return {
      "compressed_summary": compressed,
      "recommendations":    recs,
      "compression_ratio":  f"{pct:.1f}%"
    }
```

main.py:

```python
from agent import run_health_agent

if __name__ == "__main__":
    run_health_agent(
      data_path=
        "data/sample_health_data.json",
      user_question=
        "Why am I tired after 7h sleep?"
    )
```

Step 6 — Streamlit Web UI

app.py:

```python
import streamlit as st, json, os
from agent import run_health_agent

st.set_page_config(page_title="Health Coach",
                   page_icon="*")
st.title("Personal Health Coach")

uploaded = st.file_uploader(
  "Upload health_data.json", type="json")
question = st.text_input(
  "Ask a question (optional)")

if uploaded and st.button(
      "Get Recommendations",
          type="primary"):
  data = json.load(uploaded)
  os.makedirs("data", exist_ok=True)
  with open("data/temp.json", "w") as f:
    json.dump(data, f)

  with st.spinner("Analysing..."):
    result = run_health_agent(
      "data/temp.json", question or None)

  st.subheader("Health Summary")
  st.info(result["compressed_summary"])
  st.subheader("Recommendations")
  st.success(result["recommendations"])
  st.caption(result["compression_ratio"])
```

```
streamlit run app.py
```

Step 7 — Testing

```
# Run CLI agent
python main.py

# Quick inline test
python -c "
from␣agent␣import␣run_health_agent
run_health_agent(
␣␣'data/sample_health_data.json',
␣␣'How␣can␣I␣improve␣energy?'
)"
```

Test scenarios (edit the JSON):

- **Poor sleep:** `sleep_hours` set to 4–5
- **Sedentary:** `steps` under 2 000
- **Healthy habits:** 8h sleep, 10k steps

Expected console output:

```
Loading health data...
Compressing health history...

--- Compressed Summary ---
[200-300 word health snapshot]

--- Health Coach Says ---
[3 recs + positive note + focus]

Compression: 65.3% reduction
```

Verify: aim for >**50%** size reduction before submitting.

3. Core Features, Checklist & Bonus Ideas

Core Features

- **Data Compression** — 60–80% token reduction via ˜300-word snapshots
- **Personalized Recommendations** — Tailored to individual profile and trends
- **Pattern Recognition** — Detects correlations (e.g. low sleep ⇒ poor mood)
- **Q&A Support** — Context-aware follow-up health questions
- **Web UI** — Streamlit interface for non-technical users

Bonus Ideas to Stand Out

- **Trend detection** — "Sleep improved 3 days in a row!"
- **Multi-week compression** — chain weekly summaries
- **Health score** (0–100) based on averages
- **Nutritional gap flagging** from meal logs
- **Weekly reminders** via a task scheduler

Submission Checklist

- Agent loads & processes health data correctly
- Compression achieves >**50%** reduction
- Recommendations are personalized, not generic
- Code is clean and well-commented
- `.env` is in `.gitignore` and NOT committed
- `sample_health_data.json` included for reviewers
- README is complete with setup instructions
- *(Bonus)* Streamlit app runs without errors

Security reminder: Always add `.env`, `venv/`, and `__pycache__/` to your `.gitignore` before pushing to GitHub. Never expose your Anthropic API key in a public repository.