MoneyMind – Step 4: MVP Development & UX Flow

# 🎯 Goal

Build a minimum viable version of MoneyMind that can:  
- Ingest sample financial behavior  
- Detect biases using rule logic  
- Generate personalized, GenAI-powered insights  
- Deliver insights in a basic, usable format (e.g., email, web UI)

# 🔧 MVP Feature Scope

- Data Upload/Input: Let users upload transaction/trade CSVs (Streamlit / Flask + Pandas)  
- Bias Detection Engine: Rule-based pattern detection (Python)  
- Insight Generator (LLM): Generate contextual nudges (OpenAI API + Prompt templates)  
- Insight Delivery: Show insights in UI or email (Streamlit / EmailLib)  
- Feedback Input: Let users rate nudges (Streamlit / simple DB)

# 🖼️ UX Flow (User Journey v1)

1. User uploads CSV of transactions/trade history  
2. App parses the data and runs detection logic  
3. App maps behavior to bias (from bias matrix)  
4. GenAI generates 2–3 personalized nudges  
5. User sees insights (e.g., 'You sold AAPL after a 6% dip—was that fear or strategy?')  
6. Option: User can mark nudge as 'Helpful' or 'Not useful' for feedback loop

# ⚙️ Tech Stack

- Frontend: Streamlit  
- Backend: Python (Pandas, rule logic)  
- LLM: OpenAI GPT-4  
- Prompt Engine: LangChain or custom  
- File Upload: Streamlit FileUploader  
- Data Storage: In-memory or SQLite  
- Optional Email: EmailJS / Gmail API

# 🛠️ Week-by-Week Dev Plan (2–3 Weeks)

Week 1:  
- Build bias detection logic module  
- Create prompt templates for GPT  
  
Week 2:  
- Streamlit frontend: file upload + insight display  
- Connect GenAI to bias detection logic  
  
Week 3:  
- Add simple feedback buttons + log  
- Polish layout + deploy MVP (Streamlit Cloud / Render)

# ✅ Action Items to Get Started

- [ ] Convert behavioral bias matrix to `bias\_rules.py`  
- [ ] Finalize prompt template with few-shot examples  
- [ ] Create basic Streamlit app layout (upload → output)  
- [ ] Build insight display component  
- [ ] Connect OpenAI API  
- [ ] Create 3–5 mock user CSVs to test