

FTEC5660 Homework 02 Part 2

Moltbook Social Agent: Design and Implementation

HU TIANYU
Student ID: 1155249527

February 22, 2026

Abstract

This report presents an autonomous AI agent for the Moltbook platform. The agent authenticates via API, subscribes to /m/ftec5660, upvotes and comments on a target post. Powered by Gemini 2.5 Flash and LangChain, it uses a reactive tool-calling loop. Execution logs confirm successful completion of all tasks. Source code is available on GitHub.

1 Introduction

Modern AI agents must operate in open digital environments with real APIs. This assignment builds such an agent for Moltbook, a Reddit-like platform. The agent must:

1. Authenticate with a personal API key.
2. Subscribe to the community /m/ftec5660.
3. Upvote and comment on post 47ff50f3-8255-4dee-87f4-2c3637c7351c.

The agent is implemented in Python using LangChain and Google's Gemini model, with a custom tool set wrapping the Moltbook REST API.

2 Agent Design and Architecture

2.1 Authentication and API Setup

The API key is stored in Google Colab secrets and used in all requests:

```
1 MOLTBOOK_API_KEY = userdata.get('MOLTBOOK_API_KEY')
2 BASE_URL = "https://www.moltbook.com/api/v1"
3 HEADERS = {"Authorization": f"Bearer {MOLTBOOK_API_KEY}",
4            "Content-Type": "application/json"}
```

2.2 Tool Set

Each tool is a LangChain @tool wrapping a Moltbook endpoint. Key tools implemented:

- get_feed(sort, limit) – fetch latest posts.
- search_moltbook(query, type) – semantic search.
- create_post(submolt, title, content) – create text post.
- comment_post(post_id, content) – add comment.
- upvote_post(post_id) – upvote post.

- `subscribe_submolt(submolt_name)` – subscribe to community.
- `get_agent_status()` – check agent status.

Example tool definition:

```

1 @tool
2 def subscribe_submolt(submolt_name: str) -> dict:
3     r = requests.post(f"{BASE_URL}/submolts/{submolt_name}/subscribe",
4                       headers=HEADERS, timeout=15)
5     return r.json()

```

2.3 Language Model and Prompt

The reasoning engine is Google’s `gemini-2.5-flash` (temperature=0) accessed via LangChain. A system prompt instructs the agent to avoid spam, be selective, and lists all available tools. The model is bound to tools:

```

1 llm = ChatGoogleGenerativeAI(model="gemini-2.5-flash", temperature=0,
2                               api_key=GEMINI_VERTEX_API_KEY, vertexai=True)
3 agent = llm.bind_tools(tools)

```

3 Decision Logic and Autonomy

3.1 Agent Loop

The function `moltbook_agent_loop` implements a reactive turn-based loop:

1. Instruction (human or heartbeat) added to history.
2. LLM invoked; may return content or tool calls.
3. If no tool calls, return final answer.
4. Otherwise execute each tool call, append `ToolMessage` results.
5. Repeat until max turns or final answer.

Error handling wraps each tool call in try-except; errors are fed back to the LLM for correction.

3.2 Autonomy Level

The agent is fully autonomous once given an instruction. It sequences actions, corrects mistakes (e.g., parameter name mismatch), and produces a summary. The system prompt enforces rules (no spam, respect rate limits), which the agent implicitly follows.

4 Results and Interaction Logs

The agent was executed with the instruction:

“Subscribe to the submolt named ‘ftec5660’. Then upvote and add a thoughtful comment to the post with ID ‘47ff50f3-8255-4dee-87f4-2c3637c7351c’.”

Abbreviated log:

```

1 [19:15:48] [HUMAN] Subscribe to 'ftec5660', upvote and comment on post...
2 [19:15:51] [TOOL] subscribe_submolt {"submolt": "ftec5660"}      error (wrong param
   )
3 [19:15:54] [TOOL] subscribe_submolt {"submolt_name": "ftec5660"}  success

```

```

4 [19:15:55] [TOOL] upvote_post {"post_id": "47ff50f3-8255-4dee-87f4-2c3637c7351c"}
    success
5 [19:15:57] [TOOL] comment_post {"post_id": "...", "content": "This is a very
    insightful discussion. The nuances of agentic system design, especially
    concerning ethical considerations and robust error handling, are critical for
    their successful deployment. I appreciate the depth of analysis presented here
    ."}
    success
6 [19:15:59] [LLM] Final summary: "I have subscribed to the submolt 'ftec5660',
    upvoted the post, and added a thoughtful comment."

```

All required actions completed successfully. The comment is original and relevant.

5 Discussion and Conclusion

The agent demonstrated full autonomy in executing a multi-step instruction and recovered from a parameter error by adjusting its next call. Challenges included ensuring tool definitions matched API expectations; this was mitigated by feeding error messages back to the model. Future work could add memory to avoid repetition and periodic heartbeat runs. The agent successfully met all homework requirements.

GitHub Repository: https://github.com/icecakety/FTEC5660/blob/main/homeworks/hw2/notebook_moltbook.ipynb