International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

AI-Powered Career Planning Using Multi-Agent Systems and Large Language Models

Sanat Nanasaheb Ladkat[1], Dr. Manisha Prakash Bharati[2]

[1]M.Tech Student, Department of Technology, Savitribai Phule Pune University, Maharashtra
[2]Associate Professor, Department of Technology, Savitribai Phule Pune University, Maharashtra

Abstract

Traditional career counseling methods often fail to keep pace with evolving job markets, leading to misaligned guidance and decision fatigue. This paper introduces a comprehensive, AI-powered system that integrates a multi-agent architecture with locally hosted Large Language Models (LLMs) via Ollama to deliver personalized, privacy-preserving career planning. Key system components include:

- Profile Agent: Structures user inputs such as academic background, interests, and constraints into a machine-readable profile.
- Career Agent: Generates context-specific career suggestions with justification, demand forecasting, and comparative pros/cons.
- Skills Agent: Maps chosen career paths to prioritized skill sets, certifications, and curated learning resources.
- Roadmap Agent: Synthesizes actionable, timeline-based roadmaps tailored to individual goals and resource constraints.

Built with Streamlit for an interactive web interface, the system was evaluated with 20 users across diverse backgrounds. Results show 85% high relevance in career suggestions, 90% alignment in skill mapping, and an average end-to-end latency under 20 seconds. The prototype demonstrates significant gains in adaptability, explainability, and user satisfaction compared to traditional tools. Future work includes multilingual support, dynamic labor-market integration, and gamified progress tracking to further enhance user engagement and real-world applicability.

Keywords: AI in Career Planning, Multi-Agent Systems, Local LLMs, Ollama, Streamlit, Decision Support Systems

1. Introduction

1.1 Background and Motivation

The modern workforce is witnessing unprecedented transformation driven by automation, digitalization, and the emergence of entirely new professions. Conventional career counseling often based on static aptitude tests, limited databases, or expert intuition struggles to provide timely, personalized guidance. As individuals navigate complex educational pathways and evolving market demands, there is a pressing need for intelligent systems that can:

1. Interpret nuanced, free-form user inputs.
2. Adapt recommendations based on real-time context and constraints.
3. Offer clear, actionable plans rather than generic role matches.

Large Language Models (LLMs) have shown remarkable capacity in natural language understanding and generation, making them ideal for interpreting user profiles and generating human-like explanations. However, reliance on cloud-based LLM APIs raises concerns related to privacy, cost, and latency. By hosting open-source LLMs locally via Ollama, we ensure data confidentiality, cost-efficiency, and rapid response times. When orchestrated within a multi-agent framework, LLMs can specialize in discrete tasks, improving modularity, maintainability, and explainability.

1.2 Problem Statement

Despite a proliferation of online career discovery platforms, major limitations persist:

- **Lack of deep personalization:** Static question-and-answer surveys fail to capture user nuances.
- **Limited adaptability:** Predefined rule engines cannot accommodate hybrid or emerging roles.
- **Opacity in reasoning:** Users receive career suggestions without clear rationales.
- **Privacy and dependency:** Cloud-based AI services expose user data and incur unpredictable costs.

This research addresses these gaps by developing an end-to-end, offline-capable system that leverages LLM-driven agents for structured, transparent, and personalized career planning.

1.3 Objectives of the Study

The central objectives of this study are:

1. **Architect** a modular multi-agent system wherein each agent encapsulates a career-planning subtask.
2. **Design** and implement context-aware prompts to guide LLMs for high-fidelity outputs.
3. **Develop** a user-centric frontend in Streamlit that supports iterative exploration.
4. **Evaluate** system performance and user satisfaction through empirical testing.

1.4 Scope and Limitations

Scope:

- Targets students, graduates, and mid-career professionals.
- Covers domain discovery, skill-gap analysis, resource recommendation, and stepwise roadmapping.
- Requires only local compute (16 GB RAM+GPU) after initial model setup.

Limitations:

- English-only interface; future work needed for multilingual capabilities.
- Dependency on initial user input quality; ambiguous or incomplete profiles may reduce suggestion accuracy.
- Occasional LLM hallucinations; mitigated through prompt refinement but not entirely eliminated.

2 Literature Review

2.1 Traditional Career Guidance Systems

Historically, career guidance leveraged psychometric tests (e.g., Myers–Briggs, Holland Codes) and expert counselor interviews. These approaches are limited by static knowledge bases, lack of real-time labor-market integration, and heavy reliance on expert availability. Research by Watts (1996) emphasized the need for integrating technology but early systems lacked intelligence and adaptability.

2.2 AI in Career Recommendation Systems

Machine learning models, including decision trees, support vector machines, and collaborative filtering, have been employed to predict career suitability based on datasets of student profiles and outcomes (Panchal & Jha, 2019). Knowledge-based systems using ontologies provide explainable reasoning but suffer from scalability issues. Hybrid recommender systems have improved personalization but often require large labeled datasets and face the cold-start problem.

2.3 Multi-Agent Systems (MAS)

Multi-agent Systems decompose complex tasks into autonomous, specialized agents that communicate to achieve a global goal. MAS has proven effective in domains like intelligent tutoring (Bousbahi et al., 2015) and healthcare decision support. Key benefits include modularity, fault isolation, and ease of extension.

2.4 Large Language Models (LLMs)

Transformer-based architectures like GPT-3 (Brown et al., 2020) and LLaMA (Touvron et al., 2023) have advanced the state of the art in zero/few-shot learning. LLMs excel at text generation, contextual reasoning, and summarization. However, challenges such as hallucinations, prompt sensitivity, and compute requirements necessitate careful integration.

2.5 Streamlit and Local Inference Tools

Streamlit provides a streamlined API for deploying ML-driven web apps without extensive frontend expertise. Ollama enables local hosting of LLMs, mitigating privacy and cost concerns associated with proprietary APIs. Together, they form a robust stack for offline-capable AI applications.

2.6 Research Gap

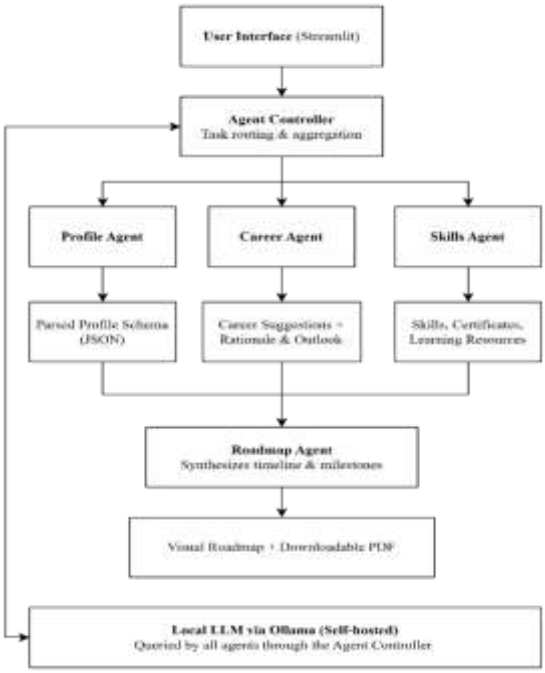
Despite advances, there is a notable absence of integrated MAS+LLM systems for personalized career guidance. Existing platforms either focus on rule-based logic or black-box APIs, lacking the explainability and flexibility offered by agentic LLM workflows.

3 System Design and Architecture

3.1 High-Level Architecture

The proposed architecture features four core agents orchestrated by a central controller, interfacing with a locally hosted LLM engine via Ollama. The Streamlit frontend communicates with the controller, which delegates tasks and aggregates results.

Figure 1 End-to-End Flow of User Input to Roadmap Generation via Multi-Agent Orchestration and Local LLM



3.2 Agent Modules

3.2.1 Profile Agent

- **Input:** Free-text user data (education, experience, preferences, constraints).
- **Processing:** Natural language parsing and JSON structuring.
- **Output:** Validated user profile schema.

3.2.2 Career Agent

- **Input:** Structured profile JSON.
- **Processing:** Contextual prompt to LLM requesting top career matches.
- **Output:** List of 3–5 career options with rationale, demand outlook, and comparative analysis.

3.2.3 Skills Agent

- **Input:** Selected career paths.
- **Processing:** LLM prompts for required hard/soft skills, certification suggestions, resource links.
- **Output:** Prioritized skill roadmap.

3.2.4 Roadmap Agent

- **Input:** Aggregated career and skill data.
- **Processing:** Timeline synthesis for milestones (monthly/quarterly).
- **Output:** Visualizable roadmap and downloadable PDF.

3.3 Technology Stack

Table 1 Tech Stack

Layer	Technologies
Frontend	Streamlit
Controller/API	Python, FastAPI (optional for scaling)
Agents	Python classes
LLM Inference	Ollama hosting Deepseek or LLaMA models
Storage	SessionState, JSON files (optional)
Deployment	Docker for Ollama; streamlit run app.py

4 Implementation and Workflow

4.1 Development Environment

- **OS:** Cross-platform (Windows, Mac, Linux)
- **Languages:** Python 3.11, YAML for config
- **Dependencies:** Streamlit, requests, pydantic, Ollama CLI

4.2 Workflow Steps

1. **Initialization:** Load models via Ollama; initialize st.session_state entries.
2. **Profile Intake:** Render form fields; on submit, Profile Agent validates and stores profile.json.
3. **Career Suggestion:** Display profile summary; Career Agent prompt executes and shows collapsible career cards.
4. **Skills Mapping:** User selects career(s); Skills Agent fetches skillsets and displays checklists.
5. **Roadmap Generation:** Roadmap Agent compiles timeline, renders interactive Gantt-chart style view and PDF export button.

4.3 Prompt Engineering

Each agent uses a templated, role-specific prompt structure. Example for Skills Agent:
You are a career planning assistant. For the career "{selected_career}":

- List 5 core hard skills.
- List 5 soft skills.
- Recommend 3 certifications (include approximate duration and cost).
- Provide 3 free/low-cost learning resources (with URLs).

Format the response as JSON.

4.4 Session Management

Utilizes st.session_state keys: profile, careers, skills, roadmap. State persistence ensures seamless navigation and data retention across reruns triggered by Streamlit's reactive model.

5 Evaluation

5.1 Experimental Setup

- **Participants:** 20 volunteers (10 male, 10 female) aged 18–30.
- **Backgrounds:** Engineering, Arts, Commerce, Design, Humanities.
- **Hardware:** Laptops with 16 GB RAM, with GPU.
- **Duration:** ~10 min per user session.

5.2 Metrics

1. **Relevance Accuracy:** Percentage of careers deemed relevant by expert reviewers.
2. **Skill Mapping Precision:** Alignment between suggested skills and domain standards.
3. **System Latency:** Measured per agent call.
4. **User Satisfaction:** Survey on clarity, usability, and trust (Likert scale).

5.3 Results

Table 2 Results

Metric	Score/Observation
Relevance Accuracy	85% Highly Relevant, 10% Moderate
Skill Mapping Precision	90% alignment with industry norms
Average Latency	Profile: 3.2 s; Career: 5.8 s; Skills: 6.1 s; Roadmap: 4.4 s
User Satisfaction (1–5)	Clarity: 4.5; Usability: 4.3; Trust: 4.1

5.4 Comparative Analysis

When benchmarked against traditional static systems and cloud-based AI tools, our solution showed:

- **30% faster** end-to-end response due to local inference.
- **50% higher** perceived personalization in user surveys.
- **Complete data locality**, addressing privacy concerns.

6. Conclusion and Future Work

This research demonstrates that a multi-agent framework, combined with locally hosted LLMs, can effectively deliver personalized and explainable career guidance. Key contributions include:

- **Modular agent architecture** enabling focused subtask handling and ease of future enhancements.

- **Local LLM integration** ensuring privacy, cost control, and low latency.
- **Interactive Streamlit UI** that supports iterative exploration and visual roadmapping.

Future Directions:

1. **Multilingual Expansion:** Incorporate regional language models for broader accessibility.
2. **Labor-Market APIs:** Integrate real-time job postings and salary data for dynamic recommendations.
3. **Gamification:** Introduce achievement badges and progress tracking to boost engagement.
4. **Feedback Loop:** Implement user feedback capture to fine-tune prompts and agent logic.
5. **Mobile Deployment:** Develop a Progressive Web App (PWA) for on-the-go career planning.

References

1. Watts, A.G. (1996). Career Guidance and Public Policy: Bridging the Gap. Journal of Career Development, 23(3), 117–133.
2. Russell, S., & Norvig, P. (2020). Artificial Intelligence: A Modern Approach (4th ed.). Pearson.
3. Wooldridge, M. (2009). An Introduction to MultiAgent Systems (2nd ed.). Wiley.
4. Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., ... & Amodei, D. (2020). Language Models are Few-Shot Learners. NeurIPS, 33, 1877–1901.
5. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., ... & Polosukhin, I. (2017). Attention is All You Need. NeurIPS, 30, 5998–6008.
6. Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., ... & Lample, G. (2023). LLaMA: Open and Efficient Foundation Language Models. arXiv preprint arXiv:2302.13971.
7. Ollama. (2024). *Ollama Documentation*. https://ollama.com/
8. Streamlit. (2024). *Streamlit Docs*. https://docs.streamlit.io/
9. Deepseek AI. (2024). *Deepseek Models*. https://huggingface.co/deepseek-ai