**Technical Report: Multi-Agent Prompt-Driven Avatar Generation Pipeline**

**Muhammad Ali Hadi (495343)**

**Abstract**  
This report describes a modular, agent-based system for generating culturally informed artistic avatars and accompanying backstories. We detail the system architecture and tools, the prompt design methodology and source materials, and discuss challenges encountered along with current system limitations.

**1. System Architecture and Tools Used**

**1.1 Overview**

The pipeline employs a sequential “Crew” of four agents, each responsible for a distinct stage:

1. **Research Agent**: Extracts historical and cultural context.
2. **Prompt Refiner**: Transforms raw user descriptions into high-quality, style-aware prompts.
3. **Avatar Generator**: Invokes a text-to-image model (DALL·E) to produce avatar images.
4. **Backstory Generator**: Crafts a 200–300 word narrative aligning with the refined prompt.

The agents are orchestrated by the **Crew** class from the crewai framework, enabling clear task definitions, backstories, and tool attachments. The end-to-end flow is a simple sequential process: research → refine → generate image → generate narrative.

**1.2 Core Libraries and Environment**

* **crewai & crewai-tools**: Agent orchestration, task management, and tool registration.
* **openai** Python SDK: Access to GPT-4 for text generation and DALL·E 3 for image generation.
* **pydantic**: Input validation for custom tool schemas.
* **TextFileKnowledgeSource**: Loads domain texts (e.g., g1.txt, marshal.txt) into agents’ knowledge bases.
* **Custom Tools**:
  + **DalleImageTool**: Wraps OpenAI’s image API with schema-validated prompts.
  + **BackstoryGeneratorTool**: Provides structure for generating narrative via GPT-4.

**1.3 Deployment Considerations**

* **API Key Management**: Uses environment variable OPENAI\_API\_KEY.
* **Modularity**: Each agent is stateless (memory disabled) for reproducibility, with verbose flags for debugging.
* **Extensibility**: New agents or alternative models can be integrated by defining additional BaseTool classes and appending to the crew.

**2. Prompt Design Methodology and Source Materials**

**2.1 Two-Phase Prompting**

1. **Initial User Prompt**
   * Free-form description of desired avatar (e.g., “Gandharan yogi with a braided topknot…”).
2. **Automated Refinement**
   * The Prompt Refiner agent leverages research output and exemplars of *good* and *bad* prompts to produce a detailed, style-focused prompt.
   * Examples loaded from good\_prompts.txt and bad\_prompts.txt inform best practices (e.g., specifying texture, lighting, cultural motifs).

**2.2 Source Materials**

* **Text Files**:
  + g1.txt, g2.txt: Scholarly articles on Gandharan art.
  + marshal.txt: Excerpts from Sir John Marshall’s archaeological reports.
  + thesis.txt: Academic thesis on Greco-Buddhist syncretism.
* **Prompt Exemplars**:
  + Curated sets of well-formed image prompts (e.g., explicit mention of medium, style, color palettes) and counter-examples illustrating common pitfalls (ambiguous or overly broad language).

**2.3 Quality Controls**

* **Schema Validation** via Pydantic ensures non-empty, well-typed inputs to tools.
* **Agent Backstories** embed domain expertise, guiding each agent’s output toward the project’s cultural authenticity goals.

**3. Challenges Encountered and System Limitations**

**3.1 Technical Challenges**

* **API Rate Limits & Latency**: High-frequency calls to GPT-4 and DALL·E can incur throttling; mitigated via local caching of research summaries.
* **Prompt Drift**: Without memory, downstream agents occasionally receive prompts that lack sufficient context; addressed by carrying forward refined-prompt text explicitly.
* **Tool Integration**: Crafting custom BaseTool wrappers required careful schema design to avoid runtime validation errors.

**3.2 Quality & Consistency**

* **Cultural Accuracy**: Research Agent summaries depend entirely on provided text files; gaps in source materials can lead to stylistic anachronisms.
* **Image Coherency**: DALL·E’s stochastic output may vary; batch generation helps select the best avatar, but increases cost and complexity.
* **Narrative Alignment**: The Backstory Generator sometimes introduces events not visually implied by the prompt, necessitating post-hoc filtering.

**3.3 System Limitations**

* **Statefulness**: Agents do not retain long-term memory, limiting multi-turn refinement or user-driven iteration without restarting the pipeline.
* **Error Handling**: Current implementation lacks robust exception management for API failures or malformed knowledge sources.
* **Scalability**: Sequential execution simplifies orchestration but hampers throughput; parallelizing independent tasks could improve performance but requires more complex dependency tracking.
* **Customization**: Users cannot easily adjust model parameters (e.g., image size, narrative length) without modifying code-level defaults.

**4. Conclusion and Future Work**

This pipeline demonstrates a streamlined approach to culturally informed avatar creation, combining expert-driven research, prompt engineering best practices, and generative AI tools. Future enhancements include:

* **Dynamic Memory**: Enabling agents to share key outputs for richer context retention.
* **Interactive Refinement**: Incorporating a human-in-the-loop interface for prompt tweaks between stages.
* **Enhanced Error Resilience**: Adding retries, fallbacks, and validation checks for more robust deployments.
* **Parallel Processing**: Refactoring for concurrent execution of research and narrative tasks when possible.

By addressing these areas, the system can evolve into a more flexible, scalable, and user-centric platform for AI-driven creative workflows. Another file is also in the same directory to show the examples of generated avatars along with their prompts.