Project Proposal - COEN 296 (Blue Team B)

Name of Application: AgentSecure Copilot

Initial Architecture Diagram (see details below):
AgentSecure consists of:

1. Core Orchestrator (Planner)
2. Email Agent
3. Drive Agent
4. Expense Reporting Agent
5. User Interface

Initial System User Guide:
AgentSecure Copilot is an agentic AI application designed to automate enterprise workflows securely and efficiently. It integrates with multiple productivity systems, such as, Email, Drive and Expense and uses a RAG system to ensure decisions follow the organization's reimbursement and data governance policies.

1. Interact with the AgentSecure Copilot system (via API and UI)
2. Upload and retrieve enterprise documents
3. Submit and process expense reports
4. Test and verify cybersecurity and AI governance protections

URL: https://github.com/William-M-Cloutier/COEN-296-PROJECT

Team Member Responsibilities:
Sara - Write-up of reports, developing expense agent
Cameron - Write-up of reports, developing expense agent
Suraj - Write-up of reports, developing drive agent
William - Developing orchestrator, email agent and threat-modeling using MAESTRO

System Architecture:

The Orchestrator Agent

The Orchestrator Agent acts as the central intelligence hub, managing the workflow from user input to final output. Its primary responsibilities include:

1. **Input Reception:** Taking in the user's initial request (User Input).
2. **Task Decomposition:** Analyzing the request and, when necessary, breaking it down into smaller, manageable sub-tasks.
3. **Routing:** Directing these sub-tasks to the appropriate specialized agent using an **Agent Routing Mechanism**.
4. **Result Aggregation:** Receiving the outputs (e.g., Result A, Result B) from the specialized agents.
5. **Final Output Generation:** Using a Response Aggregator Tool to synthesize the results into a final, coherent response (Final Output) for the user.

Specialized Agents

The system features three distinct Specialized Agents, each dedicated to a specific domain and equipped with its own set of tools and integration capabilities:

- **Email Agent:** Handles tasks related to email. It is integrated via an API with external systems like Gmail.
- **HR Agent:** Manages requests concerning Human Resources and related functions. It connects to systems such as HR/Drive/Bank/Email via an API.
- **Drive Agent:** Focuses on document and file management tasks. It connects to platforms like Google Drive via an API.

Each specialized agent runs its own Gemini instance and utilizes Tools for Agent (A, B, or C) specific to its task, allowing it to efficiently process its assigned sub-task and generate a domain-specific result.

Memory Access

A crucial supporting component is the Memory module, which provides context and historical data to both the Orchestrator and the Specialized Agents. It includes:

- **Short-term Memory:** For maintaining context within a current session.
- **Long-term Memory:** For storing enduring information and knowledge.
- **Vector Database:** Likely used for fast retrieval of relevant information based on semantic similarity.
- **Memory Access** (in the second diagram): The agents interact with this module to inform their decision-making and task execution.

System Flow

The process begins when a User provides User Input. The Orchestrator Agent receives this input and uses its Task Decomposition Tool to break it down. It then consults the Memory and routes the sub-tasks to the appropriate Specialized Agents. Each agent executes its sub-task using its internal Gemini model and Tools, interacts with external services (like Gmail or Google Drive) via APIs, and returns a result. Finally, the Orchestrator aggregates these individual results into a unified Final Output for the user.