Assignment Brief — Module 6: Building and Deploying AI Agents

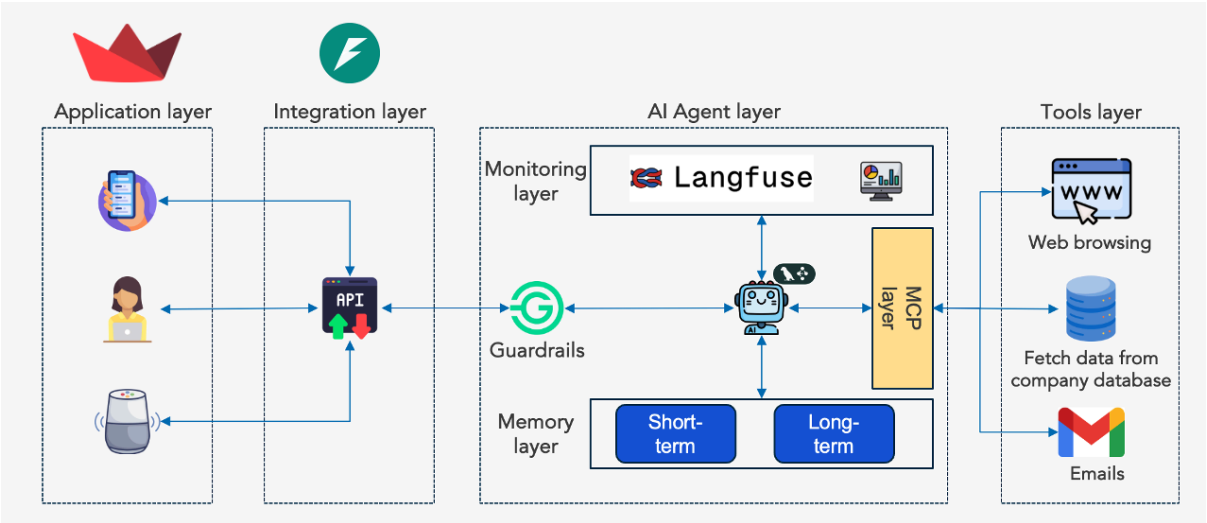
Objective:

This assignment complements your **Capstone Project** by focusing on the **technical quality, reliability, and responsible design** of the **AI Agent** you are developing.

Your mission is to **design, implement, deploy, monitor, and evaluate** a LangGraph-based AI Agent that integrates with an MCP server, enforces guardrails for safety, and demonstrates observability through Langfuse.
This assignment will be graded as a **standalone technical evaluation** of your Capstone's agentic component.

By completing this assignment, you will:

- Build an AI Agent using **LangGraph (Python)** and integrate it with tools and at least one **MCP server**.
- Deploy the agent as an **API on Render** and connect it to a **Streamlit Community Cloud UI**.
- Implement **monitoring and evaluation using Langfuse** and **safety mechanisms using Guardrails AI**.
- Deliver complete **source code, documentation, and a technical report** demonstrating your engineering decisions and learnings.



Scope

This assignment focuses on the **Agent layer** of your Capstone Project, its architecture, deployment, evaluation, and guardrails. Your work must demonstrate the following **five technical pillars**:

Technical Requirements

1. Agent Architecture & MCP Integration

- The Agent must be implemented in Python using LangGraph.
- Integrate at least one tool into the agent.
- Integrate at least one MCP server.
- Define a clear agent structure: nodes, tools, state, and control flow.
- Explain design decisions including:
    - Model and tool selection
    - Context and memory management
    - Orchestration logic and execution reasoning

Include an architecture diagram showing how the LangGraph agent interacts with the MCP server, external tools, and other components.

2. Deployment Setup

The agent must be deployed in a fully functional end-to-end configuration:

Component	Requirement	Deployment Target
Backend (Agent API)	Host your LangGraph Agent as an accessible API	Render
Frontend (User Interface)	Build an interactive Streamlit UI for users to query the Agent	Streamlit Community Cloud
Integration	Demonstrate working communication between UI → API → LangGraph Agent → MCP	Must show working deployment

You must provide a deployment diagram and links to both deployed components.

3. Evaluation & Monitoring

- Integrate Langfuse to observe and evaluate your Agent's behaviour.

- Track and report metrics such as:
    - Request latency
    - Tool invocation rate and success/failure ratio
    - Response coherence and quality (via LLM-as-a-Judge)
- Present key insights from Langfuse dashboards or exported logs, highlighting where your Agent performs well and where it fails.

4. Safety & Guardrails

- Implement at least one Guardrails AI mechanism. Examples include:
    - Input validation or sanitisation
    - Output moderation or filtering
    - Policy enforcement or restricted tool usage
- Explain:
    - The risk scenario being mitigated
    - The guardrail implementation and configuration
    - The impact on safety and UX

5. Reflection & Future Work

Reflect critically on:

- What design trade-offs you faced (simplicity vs flexibility, speed vs safety, etc.)
- What worked and what didn't during development or deployment
- How you would evolve your system to a production-level agent (e.g., security, scaling, caching, authentication)

Deliverables

Submit a single report (PDF) that combines both documentation and analysis. Additionally, provide links to your code, deployments, and dashboards.

Deliverable	Description
1. GitHub Repository	Complete, well-organised source code including the LangGraph agent, MCP integration, and Streamlit UI. Must include detailed `README.md` summarizing the project.
2. Report (PDF)	A structured report that integrates documentation, evaluation, and reflection, including all sections listed below, and limited to a maximum of 10 pages.
3. Working Deployments	- Render API endpoint - Streamlit Community Cloud app - Langfuse dashboard (public view or screenshots)

Engineering Report Structure

Section	Description
1. Overview & Context	Briefly describe your capstone use case and the Agent's purpose.
2. Architecture & MCP Integration	LangGraph structure, components, and design rationale.
3. Deployment Setup	Render API, Streamlit UI, and integration workflow.
4. Evaluation & Monitoring	Langfuse metrics, evaluation strategy, and insights.
5. Safety & Guardrails	Implementation and relevance of Guardrails AI.
6. Reflection & Future Work	Lessons learned, trade-offs, and next steps.

Submission Guidelines

Send deliverables by email

- **Deadline:** December 20th, 2025 (aligned with Capstone Demo Day)
- **Language:** English
- **Submission Package Must Include:**
    - GitHub link to source code
    - Render API link
    - Streamlit UI link
    - Langfuse dashboard (or screenshots)
    - PDF Engineering Report

Evaluation Criteria (Total: 20 Points)

Criterion	Description	Points
Architecture & LangGraph Implementation	Sound design, modularity, correct LangGraph and MCP integration	4
Deployment & Integration	Functional end-to-end system on Render and Streamlit Cloud	4
Evaluation & Monitoring (Langfuse)	Insightful use of metrics and performance analysis	4

Safety & Guardrails (Guardrails AI)	Effective risk mitigation and proper configuration	3
Documentation & Reflection	Clarity, completeness, and depth of reflection	3
Creativity & Innovation Bonus	(Optional +2) Originality in architecture, features, or user experience	+2