Scenario- Agentic AI set question 1

1. Scenario:
   Your company is building a smart customer support chatbot to handle FAQs, ticket creation, and escalation.

   Question:
   Which Multi-Agent architecture would work best?

Steps:

1. Use a Master Agent to control the conversation.
2. Create specialized agents for FAQs, ticketing, and escalation.
3. Route user queries to the correct sub-agent.
4. Maintain shared context across agents.
5. Escalate to a human agent when needed.

---

2. Scenario:
   In a Nested Agent setup, one Sub-Agent fails to complete its sub-task.

   Question:
   What should the Master Agent ideally do?

Steps:

1. Detect sub-agent failure or timeout.
2. Log the failure for monitoring.
3. Retry the task or assign it to another agent.
4. Simplify or reframe the task if needed.
5. Notify the user or fallback safely.

---

3. Scenario:
   You're building a conversational AI for a healthcare app. One module checks symptoms, and another suggests next steps.

   Question:
   How should the agent system be designed for smooth user interaction?

Steps:

1. Use a single front-facing conversational agent.
2. Internally call symptom-check and guidance agents.

3. Share context between medical agents.
4. Sequence responses clearly.
5. Add safety and disclaimer checks.

---

4. Scenario:
   During real-time chat, the agent responds slowly and often delays the next message.

   Question:
   What could be the reason and how to solve it?

Steps:

1. Check for heavy model or tool latency.
2. Identify blocking synchronous calls.
3. Use lightweight models where possible.
4. Enable async or streaming responses.
5. Cache frequent responses.

---

5. Scenario:
   An agent-based project for a smart assistant is expanding. You now need to add calendar handling, email reminders, and weather updates.
   Question:
   How will you structure your agent system?

Steps:

1. Introduce a Master Orchestrator Agent.
2. Add domain-specific agents for calendar, email, and weather.
3. Use tool/API integrations per agent.
4. Share user context centrally.
5. Keep agents loosely coupled.

---

6. Scenario:
   Your Conversational Agent often gives generic replies and doesn't adapt to user preferences over time.

   Question:
   What’s missing in the agent’s design?

Steps:

1. Lack of user memory or personalization.
2. No long-term context storage.
3. Add preference and history memory.
4. Update responses based on past interactions.
5. Periodically refresh stored context.

---

7. Scenario:
   Your chatbot is repeating the same answers even when the context changes.

   Question:
   What's possibly wrong and how to fix it?

Steps:

1. Context window is not updated.
2. Old messages are being reused.
3. Improve context management logic.
4. Reset or prune chat history.
5. Pass updated state to the agent.

---

8. Scenario:
   An AI agent is making decisions based on user financial data.
   Question:
   What design considerations should you follow?

Steps:

1. Ensure data privacy and security.
2. Apply strict access control.
3. Use explainable decision logic.
4. Add human-in-the-loop approval.
5. Follow regulatory compliance.

---

9. Scenario:
   You're building a travel assistant agent that pulls flight, hotel, and weather data.

   Question:
   How will you manage multi-source interaction?

Steps:

1. Use a central coordinator agent.
2. Create separate agents for each data source.
3. Standardize data formats.
4. Aggregate responses before replying.
5. Handle API failures gracefully.

---

10. Scenario:
    Your chatbot was used for FAQs before. Now it's becoming a full personal assistant.

    Question:
    What changes will you make to the agent design?

Steps:

1. Move from single-agent to multi-agent architecture.
2. Add memory and personalization.
3. Introduce task-planning capabilities.
4. Integrate tools and external APIs.
5. Support long-term conversations.