Name: Shekhar Chaudhary

Date: 10/07/2025

## **Part One: Emerging and Disruptive Technologies**

### **Technology 1: Agentic Artificial Intelligence Systems**

**1. Identification and Description**  
Agentic Artificial Intelligence (AI) systems represent the next generation of intelligent computing. Unlike traditional AI models that respond to prompts or perform isolated tasks, agentic AI systems can reason, plan, and take autonomous actions toward a defined goal. They integrate perception, decision-making, and tool use into one continuous workflow, making them capable of executing complex, multi-step operations with minimal human supervision. Examples include **real agent-driven applications** such as **OpenAI’s GPT-powered assistants**, **Google DeepMind’s AlphaCode**, and **Microsoft’s Copilot**, which leverage autonomous reasoning and multi-step task execution to automate complex business and creative processes across industries.

**2. Likely Impacts on Computer Science or My Career**  
Agentic AI is transforming computer science by merging software engineering, cognitive science, and machine learning into one discipline. For my career, this technology perfectly aligns with my goal of building scalable, intelligent systems that automate decision workflows. It shifts the focus from programming static logic to engineering adaptive agents that can reason and act dynamically. As I continue developing my own multi-agent architectures, I am gaining advanced experience in AI model orchestration, safety validation, and reinforcement learning techniques—all crucial skills for the future of AI engineering.

**3. Impact on Humans, Communities, or the World**  
Agentic AI can dramatically enhance productivity by automating repetitive, cognitive tasks, freeing humans to focus on creativity and problem-solving. On a community scale, it can make data analysis, healthcare triage, and education more accessible. However, it also introduces ethical considerations—such as accountability, bias mitigation, and transparency—which must be addressed to ensure that AI benefits all of society responsibly.

**4. Course Outcomes Achieved and Remaining**  
Through implementing AI-driven components in my projects, I’ve strengthened my ability to design modular architectures, integrate APIs, and optimize algorithms for adaptive learning. I’ve achieved outcomes related to **software design, data structures, and scalability**. Moving forward, I plan to deepen my understanding of **AI governance, data security, and ethical deployment frameworks** to responsibly scale autonomous systems in production environments.

### **Technology 2: Cloud-Edge Integration and Distributed Intelligence**

**1. Identification and Description**  
Cloud-edge integration connects the computational power of cloud systems with the responsiveness of edge devices. Instead of sending all data to remote servers, this approach processes data locally and synchronizes insights back to the cloud. It combines the advantages of real-time decision-making with the scalability of centralized infrastructure. Technologies like AWS Greengrass and Azure IoT Edge exemplify this hybrid model, enabling continuous, low-latency processing even in limited connectivity environments.

**2. Likely Impacts on Computer Science or My Career**  
For computer scientists, this paradigm represents a major shift from monolithic and centralized systems to distributed, event-driven architectures. Personally, this strengthens my capacity to design applications like Collage AI and SipPal that can run efficiently on both local and cloud networks. It also reinforces my expertise in **database design, microservices, and performance optimization**, preparing me to engineer systems capable of serving millions of users globally with minimal latency.

**3. Impact on Humans, Communities, or the World**  
Cloud-edge systems can benefit communities by improving access to technology in remote areas, enhancing healthcare through real-time monitoring, and enabling sustainable computing practices that reduce bandwidth consumption. On a global scale, distributed intelligence ensures that data privacy is respected, as more processing occurs locally rather than in centralized data centers.

**4. Course Outcomes Achieved and Remaining**  
This technology directly relates to course outcomes in **algorithms, database efficiency, and client/server interaction**. I’ve already achieved strong outcomes in designing reliable, scalable APIs and data models. The next step is refining my skills in **distributed system fault-tolerance and data synchronization** between cloud and edge nodes, ensuring consistency and resilience across architectures.

Part Two: Status Checkpoints Table

|  |  |  |  |
| --- | --- | --- | --- |
| Checkpoint | Software Design and Engineering | Algorithms and Data Structures | Databases |
| Checkpoint | Software Design and Engineering | Algorithms and Data Structures | Databases |
| Name of Artifact Used | Software engineering project: inventory tracker app | A sorting/searching program from CS-300 (Data Structures & Algorithms | Database design from CS-340 (Client/Server Development |
| Status of Initial Enhancement | Draft completed with initial documentation | Draft completed with initial documentation | Draft completed with initial documentation |
| Submission Status | Submitted for initial review | Submitted for initial review | Submitted for initial review |
| Status of Final Enhancement | Improve the app’s modularity and add error handling, logging, and test | Improve the app’s modularity and add error handling, logging, and test | Improve the app’s modularity and add error handling, logging, and test |
| Uploaded to ePortfolio | |  | | --- | | Planned for next module |  |  | | --- | |  | | |  | | --- | | Planned for next module |  |  | | --- | |  | | |  | | --- | | Planned for next module |  |  | | --- | |  | |
| Status of Finalized ePortfolio | |  | | --- | | Not finalized yet |  |  | | --- | |  | | |  | | --- | | Finalized and polished |  |  | | --- | |  | | Not finalized yet |

### ****Reflection****

While enhancing these artifacts, I noticed testing complexity in asynchronous Angular API calls and database latency during CRUD validation. I plan to further refine logging and mock-data testing to ensure reliability. Any feedback on advanced testing strategies or ePortfolio presentation formatting would be valuable to strengthen the final submission.

### ****References****

* Google Developers. (2024). Angular documentation. https://angular.dev
* Amazon Web Services. (2024). AWS Greengrass developer guide. https://docs.aws.amazon.com/greengrass
* OpenAI. (2024). Model Context Protocol (MCP) documentation. <https://platform.openai.com/docs/mcp>
* Express.js. (2023). Application structure and middleware. https://expressjs.com
* Mozilla Developer Network (MDN). (2024). Single Page Applications (SPAs). https://developer.mozilla.org/en-US/docs/Glossary/SPA