# NebulaByte: Multi-Agent System Design

## NebulaByte Multi-Agent Systems Architecture

### Introduction to Multi-Agent Systems

Multi-agent systems consist of multiple autonomous agents that coordinate to solve complex problems. This approach offers:

- Modularity: Each agent handles specific tasks
- Scalability: Easy to add new capabilities
- Robustness: Failure of one agent doesn't crash system
- Specialization: Agents optimize for specific domains

### Controller Patterns:

1. Centralized: Single controller routes all requests
2. Decentralized: Agents negotiate directly
3. Hierarchical: Multiple levels of control
4. Hybrid: Combine approaches based on needs

### Agent Communication:

Agents exchange information through:

- Message passing
- Shared memory
- Event systems
- REST APIs

### Routing Strategies:

- Rule-based: Use explicit conditions
- LLM-based: Let AI decide routing
- Hybrid: Combine rules with AI reasoning
- Learned: Use ML to optimize routing

### Common Agent Types:

- Information Retrieval: Search and fetch data
- Processing: Transform and analyze information
- Decision Making: Choose actions based on context
- Execution: Carry out selected actions

### Design Considerations:

Consider failure modes, logging for transparency, performance optimization, and user experience when building multi-agent systems.