System Architecture Design Checklist

1. Requirements Alignment
- ☐ Have I clearly identified functional requirements (what the system must do)?
- ☐ Have I identified non-functional requirements (performance, security, usability, etc.)?
- ☐ Does my architecture map directly to these requirements?

2. Structure & Separation of Concerns
- ☐ Have I separated the system into layers (UI, business logic, data, etc.)?
- ☐ Are modules/components focused on a single responsibility?
- ☐ Is the diagram hierarchical and easy to follow?

3. Modularity & Coupling
- ☐ Are components loosely coupled (minimal dependencies)?
- ☐ Are components highly cohesive (do one job well)?
- ☐ Could I swap out a component (e.g., database, payment service) without breaking everything?

4. Performance & Scalability
- ☐ Can the system handle expected user load?
- ☐ Have I considered caching, indexing, or other performance optimizations?
- ☐ Can the system scale up or out if needed?

5. Security & Privacy
- ☐ Are authentication and authorization clearly placed in the design?
- ☐ Is sensitive data encrypted in transit and at rest?
- ☐ Am I following the Principle of Least Privilege (minimal access rights)?

6. Reliability & Fault Tolerance
- ☐ What happens if a service or component fails?
- ☐ Do I have redundancy or fallback mechanisms?
- ☐ Can the system recover gracefully from errors?

7. Observability
- ☐ Does the architecture include logging and monitoring?
- ☐ Can I track performance metrics (latency, error rates)?
- ☐ Are there health checks or alerts for failures?

8. Standards & Integration
- ☐ Am I using standard protocols/APIs (HTTP, REST, OAuth, JSON, etc.)?
- ☐ Is the system designed to integrate with external tools/services?
- ☐ Am I avoiding unnecessary complexity (YAGNI principle)?

9. Simplicity & Communication
- ☐ Is the design as simple as possible while meeting requirements?
- ☐ Could someone new to the project understand my diagram in 5 minutes?
- ☐ Have I documented assumptions, decisions, and trade-offs?