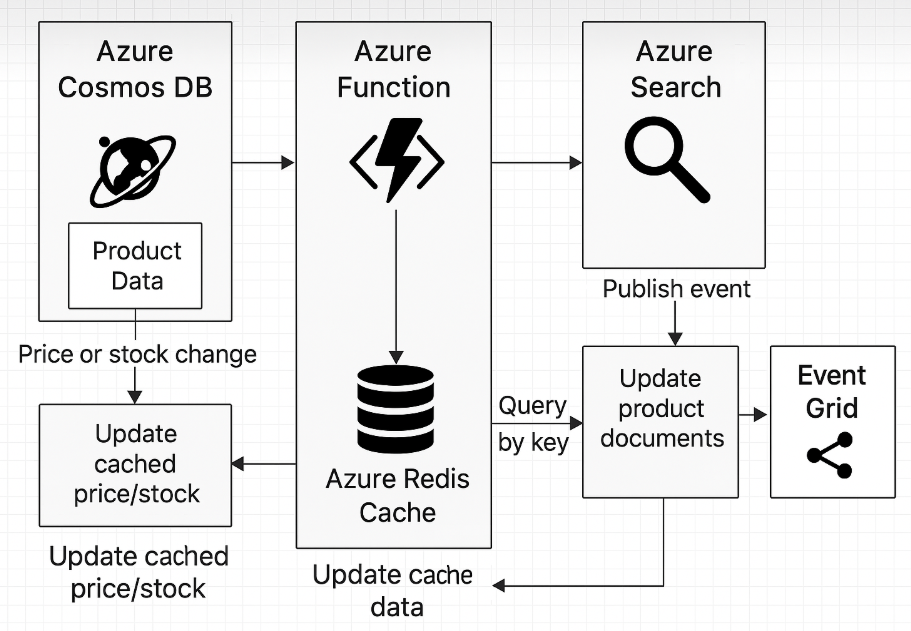
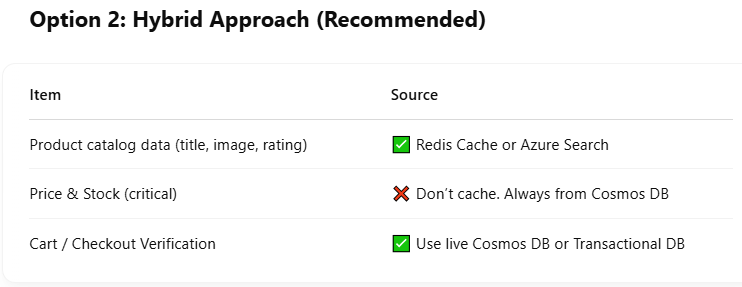
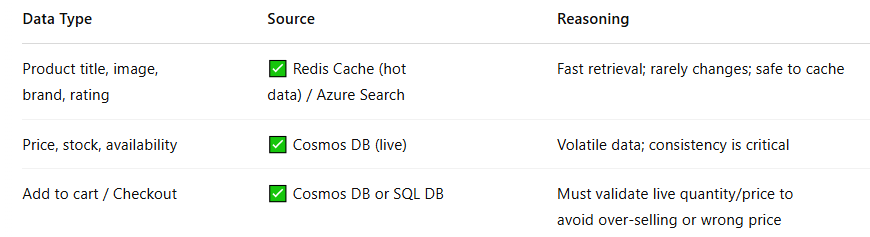
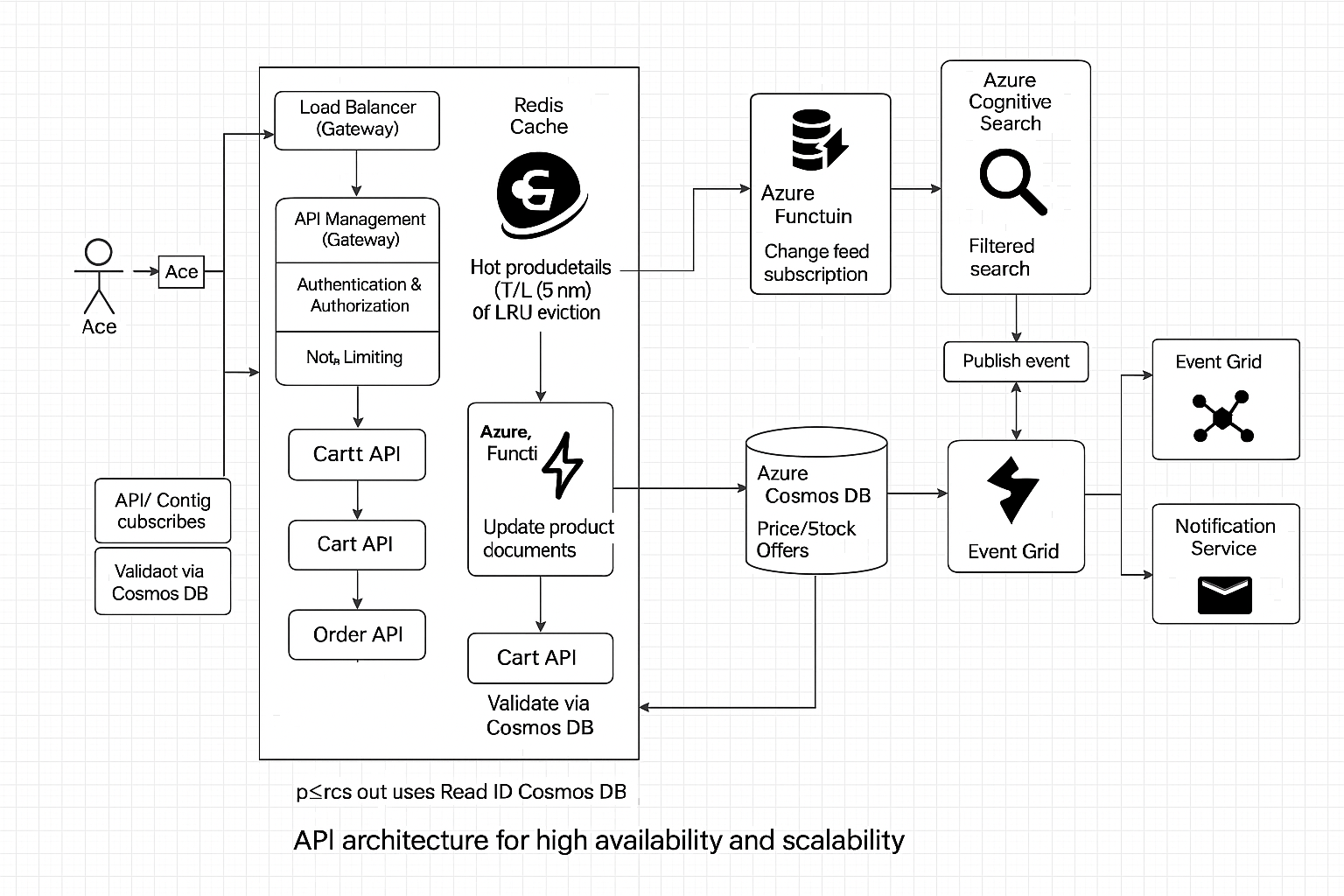
**Solution Architect**

**Job Description:**

* Previous experience in a Solution Design, or Architect Role with the ability to translate user requirements into detailed design documentation.

****

* Experience in the use of cloud technologies and in architecting solutions for the cloud, AWS and Microsoft Azure technology stack very desirable.
* Demonstrate knowledge of integration technologies and services with the ability to design and document integration architectures.
* Experience of working across domains such as Business, Data, Applications and Technology.
* Exposure to technologies such as big data, analytics, artificial intelligence, automation and Internet of Things - (desirable).
* Demonstrate clear technology thinking and a sound understanding of technology solution design principles.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **✅ Technology Solution Design Principles (Enterprise-Proven)**   | **Principle** | **Description** | **Example / Tooling** | | --- | --- | --- | | **1. Modularity** | Design systems as loosely coupled, independently deployable units | Microservices, Clean Architecture | | **2. Scalability** | Design to handle increased load without degrading performance | Azure App Service auto-scale, Cosmos DB partitioning | | **3. Resilience** | Ensure the system recovers gracefully from failures | Circuit Breaker (Polly), Retry, Outbox, DR setup | | **4. Observability** | Build visibility into systems through logging, metrics, and tracing | Azure Monitor, App Insights, OpenTelemetry | | **5. Security by Design** | Embed security across all layers from day one | OAuth2, RBAC, WAF, Key Vault, DAST/SAST | | **6. API First** | Design APIs first and treat them as products | OpenAPI (Swagger), Azure APIM | | **7. Event-Driven** | Enable decoupling via events and async communication | Event Grid, Service Bus, Kafka | | **8. Domain-Driven Design** | Align software design with business domain | Aggregates, Bounded Contexts, Ubiquitous Language | | **9. Fail-Fast** | Detect errors early and terminate quickly to reduce damage | Exception handling with alerts, circuit breakers | | **10. Cost Optimization** | Architect to balance performance, availability, and cost | Azure Cost Management, Reserved Instances, Cosmos RUs | | **11. Portability** | Minimize vendor lock-in and enable cross-platform deployment | Containerization, .NET Core, Terraform | | **12. Maintainability** | Easy to understand, test, extend, and operate over time | SOLID, Clean Code, DevOps pipelines | | **13. DR & HA Readiness** | Design for high availability and disaster recovery | RTO/RPO definition, Multi-region Cosmos/SQL setup | | **14. CI/CD Enablement** | Build pipelines for automated testing, build, and deploy | Azure DevOps, GitHub Actions | | **15. Compliance & Audit** | Ensure the system meets legal and regulatory standards | GDPR, PCI-DSS, Logging with correlation ID | |

* The Candidate should be able to provide technical direction, architect, and design solutions to meet functional and non-functional requirements within the Azure Cloud Environment
* Enforce adherence to architectural standards/principles, product-specific guidelines, usability design standards which applies to Networking, Security and Data flow modelling design.
* Proactively provide guidance on engineering methodologies, standards and best practices for all the designs produced.

|  |
| --- |
| 1. Architectural Standards **2. Code Quality & Maintainability**  **3. Security Best Practices**  **4. Performance, Observability & Operations**  **5. DevOps & Governance** |

* Provide insight and directional support to the Support Team staffs.
* Identify and mitigate Risks, Assumptions, Issues and Decisions throughout the application full life cycle.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **How I Handle RAID Throughout the Application Lifecycle:**  | **Lifecycle Stage** | **Risks** | **Assumptions** | **Issues** | **Decisions (ADR)** | | --- | --- | --- | --- | --- | | **1. Requirement Analysis** | Misalignment with business goals | APIs will be reused from legacy | Unclear NFRs | Use API-first design | | **2. Architecture Design** | Wrong tech stack choice | Cosmos DB performance fits needs | Disagreement on microservices boundary | Use Cosmos DB with Change Feed (ADR #12) | | **3. Implementation** | Developer misunderstanding of patterns | Dev team familiar with DDD | Code quality drops in modules | Enforce Clean Architecture & reviews | | **4. Testing** | Environment not matching prod | Load simulates real traffic | Flaky tests due to timing | Use isolated test environment | | **5. Deployment** | Downtime in production | Blue-green rollout will succeed | DNS cutover failed | Use Front Door with staged routing | | **6. Operations** | No alerts for latency spikes | Redis will cache high-traffic data | Logs not visible in App Insights | Add OpenTelemetry and health probes | |

* Demonstrate strong analytical and technical problem-solving skills
* Ability to analyse and operate at various levels of abstraction
* Ability to balance what is strategically right with what is practically realistic
* Excellent technical skills, enabling the creation of future-proof and complex solutions
* Excellent interpersonal communication and organizational skills that are required to operate at a commanding position with the distributed workstreams that deliver quality services and solutions.
* Ability to rapidly gain knowledge of the organizational structure to facilitate work with groups outside of the immediate technical team.
* Familiar with solution implementation/management, service/operations management, etc.

|  |
| --- |
| **1. Solution Implementation Management**   * Worked closely with developers to **translate HLD/LLD into working microservices** * Used **Azure DevOps CI/CD pipelines** to ensure repeatable, gated deployments * Defined **infra as code (Bicep/Terraform)** for environment provisioning * Ensured alignment with:   + **Security policies** (e.g., RBAC, OAuth2, WAF)   + **Deployment strategies** (Blue-Green, Canary)   + **Testing** (Unit, Integration, Load)   🎯 *Example*: “I implemented a Cosmos DB-based Order API using Clean Architecture and validated it in UAT with performance and DR drills.”  **🔹 2. Service & Operations Management**   * Set up **monitoring/alerts** via Azure Monitor, App Insights, Log Analytics * Enabled **health probes, tracing, and dashboards** using OpenTelemetry * Conducted **DR drills** and defined **RTO/RPO objectives** * Handled **production incident triage**, root cause analysis, and post-mortems * Maintained **support runbooks** and operational documents for handover   🎯 *Example*: “In a global e-commerce solution, I ensured 99.95% uptime using Azure Front Door, Redis caching, and automated scaling policies.”  **🔹 3. Service Handoff & Support Enablement**   * Created **runbooks and architecture guides** for L2/L3 teams * Ensured **SLA documentation** and alert-based escalation workflows * Integrated **change management and deployment approval** into CI/CD |

* Maintains close awareness of new and emerging technologies and their potential application for service offerings and products.
* Create Design documents with reference and integration architecture models.

|  |
| --- |
| “I’ve created 15+ full architecture designs — at least 8 of them included detailed integration architecture, and 3–4 were developed as reusable reference models for enterprise patterns like microservices, identity federation, and event-driven architecture. Each design included visual diagrams, traceable ADRs, and security/compliance alignment.” |

* Strong exposure on security best practices and apply the latest industry standards in the design.