Below is a **clear, executive + architect–level comparison of Enterprise, Solution, and Application Architecture**, tailored for **large healthcare multi-product initiatives** (like the one you've been designing).
This is **interview-ready**, **delivery-ready**, and **governance-ready**.

Enterprise vs Solution vs Application Architecture — Comparison

---

1️⃣ High-level purpose (simple explanation)

Architecture	**Primary Question Answered**	**Simple Meaning**
Enterprise Architecture (EA)	*What should the whole organization look like to achieve strategy?*	The **city plan**
Solution Architecture (SA)	*How do we implement a specific initiative correctly?*	The **building blueprint**
Application Architecture (AA)	*How do we build a system or service right?*	The **room-by-room design**

2️⃣ Scope & Time Horizon

Dimension	**Enterprise Architecture**	**Solution Architecture**	**Application Architecture**
Scope	Entire organization	One program / initiative	One system / service
Time horizon	3–5 years	6–18 months	Sprint-level to yearly
Change frequency	Low	Medium	High
Stability	High	Medium	Low
Audience	Executives, portfolio, EA	Product, architects, PMs	Engineers, SRE, QA

---

3 Ownership & Accountability

Architecture	Owner	Accountability
Enterprise	Chief Architect / EA	Strategy alignment, reuse, governance
Solution	Solution / Product Architect	Delivery success, risk, compliance
Application	Tech Lead / App Architect	Code quality, reliability, performance

---

4 Primary Outputs (Artifacts)

Architecture	Typical Artifacts
Enterprise	Reference architecture, capability maps, roadmaps, standards, governance model
Solution	Solution diagrams, ADRs, integration patterns, NFRs, security model, migration plan
Application	Component diagrams, APIs, DB schemas, sequence diagrams, CI/CD, runbooks

---

5 Level of Abstraction

Architecture	Level
Enterprise	Conceptual + logical
Solution	Logical + physical
Application	Physical + implementation

6️⃣ Decision Types

Architecture Decisions Made

Enterprise	Platforms, standards, cloud, data strategy, security posture
Solution	Integration patterns, NFRs, product boundaries, migration approach
Application	Frameworks, libraries, data models, code structure

---

7️⃣ Governance & Control

Architecture Governance Mechanism

Enterprise	Architecture Review Board, standards, guardrails
Solution	ADRs, design reviews, contract testing
Application	CI/CD gates, code reviews, SRE checks

8️⃣ Example (Healthcare)

Enterprise Architecture

- Adopt platform-first model
- Central identity & consent
- Lakehouse + data mesh
- Zero Trust security
- Event-driven integration
- GenAI governance

Solution Architecture (Telehealth Program)

- Use platform identity & consent
- API + event integration
- Façade for EHR

- NFRs: 99.9% availability
- Audit all PHI flows
- Safe GenAI summarization

Application Architecture (Televisit Service)

- Clean architecture
- OAuth2 enforcement
- Domain model for visits
- Event emission
- Circuit breakers
- Runbooks

9️⃣ How they relate (this is key)

Enterprise Architecture

↓ sets constraints

Solution Architecture

↓ applies constraints

Application Architecture

↓ implements constraints

🔟 Common failure modes

Failure	Cause	Fix
EA is ignored	Too abstract	Reference architectures
SA is bloated	Too detailed	Clear scope
AA diverges	No enforcement	Automated governance

Failure	Cause	Fix
Duplication	Weak EA	Platform enforcement
Slow delivery	Heavy reviews	Self-service guardrails

1️⃣1️⃣ Interview-ready one-liner

“Enterprise Architecture sets direction, Solution Architecture ensures safe delivery, and Application Architecture ensures quality execution. When aligned, teams move fast without breaking safety or governance — which is critical in healthcare.”

1️⃣2️⃣ When to use which (quick rule)

- **Enterprise Architecture** → Strategic, cross-product, irreversible
- **Solution Architecture** → Program-level, high risk, high integration
- **Application Architecture** → System-level, implementation detail

---

Below is a **practical, one-page cheat sheet for new architects** joining a **large healthcare, multi-product platform program**. This is designed to help them **avoid common mistakes, know what to do when, and speak the right language from day one**.

🧭 Architect Cheat Sheet (Enterprise • Solution • Application)

For Healthcare Multi-Product Platforms

1️⃣ Architecture at a Glance

Level	What You Do	Your Focus	Typical Mistake
Enterprise Architecture	Set direction	Strategy, platforms, guardrails	Too abstract
Solution Architecture	Enable delivery	Integration, risk, NFRs	Too detailed
Application Architecture	Build correctly	Code, quality, resilience	Too isolated

---

2️⃣ The Golden Rule

Enterprise constrains → Solution adapts → Application implements

If you break this order, chaos follows.

3️⃣ What to Produce at Each Level

Enterprise Architecture

- Capability map
- Reference architecture
- Standards & guardrails
- Roadmap
- Governance model
- Security posture
- Data strategy

Solution Architecture

- Solution diagram (C4 L2/L3)
- ADRs
- Integration patterns
- NFRs
- Security model
- Migration plan
- Risk register

Application Architecture

- Component diagram
- API specs
- Domain model
- DB schema
- CI/CD pipeline
- Runbooks
- Resilience patterns

4️⃣ When to Create an ADR (Always Ask This)

Create an ADR if:

- Decision is hard to reverse
- It affects security, data, or PHI
- Multiple teams are involved
- It will be questioned later
- It sets a new pattern

If you don’t write an ADR, you will re-fight the same battle.

5️⃣ Healthcare Non-Negotiables

Area	Rule
PHI	Always encrypted, audited
Identity	Platform IAM only
EHR	Façade + strangler only
Analytics	Never query EHR directly
GenAI	RAG + audit + human approval
Security	Zero Trust always
Downtime	Manual fallback required

6️⃣ Integration Patterns (Use These Only)

✅ API-first (sync)
✅ Event-driven (async)
✅ Façade + strangler
✅ Read/write separation
✅ Batch for claims
❌ Direct DB access
❌ Point-to-point spaghetti

7️⃣ Architecture Review Checklist (5-minute version)

Before approving:

- Uses platform services?
- PHI flows documented?
- ADR created?
- NFRs defined?
- Security model approved?

- Integration pattern standard?
- Runbook exists?

If any answer is “no” → stop.

8️⃣ Metrics Architects Must Watch

- Platform reuse %
- API bypass %
- MTTR
- PHI access violations
- DQ score
- Exception age
- Cost per integration

Metrics tell you when architecture is failing **before incidents happen**.

9️⃣ Common Traps & How to Avoid Them

Trap	How to Avoid
Over-design	Timebox decisions
Under-design	Use reference architecture
Being ignored	Speak in business risk
Slowing teams	Automate guardrails
Shadow IT	Make platform easier
Losing trust	Document decisions

🔟 How to Influence Without Authority

- Bring options, not opinions
- Translate tech risk to business risk

- Quantify impact
- Document decisions
- Let data speak
- Align to patient safety

1️⃣1️⃣ First 30–60–90 Day Plan for New Architects

30 days

- Learn domain & constraints
- Read reference architectures
- Map stakeholders
- Observe reviews

60 days

- Own a solution
- Write ADRs
- Lead design reviews
- Enforce patterns

90 days

- Improve governance
- Simplify platforms
- Mentor others
- Influence roadmap

1️⃣2️⃣ Interview One-Liner (use this)

“I use enterprise architecture to set guardrails, solution architecture to reduce risk, and application architecture to ensure quality — especially critical in healthcare where safety and compliance are non-negotiable.”

Below are **role-based architecture checklists** plus a **review automation checklist** that turns governance into repeatable, low-friction execution.
This is exactly what mature healthcare platforms use to **scale architecture without slowing teams**.

🧩 Role-Based Architecture Checklists

1️⃣ Enterprise Architect (EA) Checklist

(Strategy, platforms, guardrails)

Strategy & Alignment

- ☐ Business capability map updated (L1–L3)
- ☐ Portfolio OKRs mapped to capabilities
- ☐ Platform vs product boundaries defined
- ☐ Regional/regulatory differences identified
- ☐ Exec sponsors confirmed

Standards & Reference Architecture

- ☐ Enterprise reference architecture published
- ☐ Technology standards catalog updated
- ☐ Approved integration patterns defined
- ☐ Data architecture standards defined
- ☐ Security posture (ZTA) enforced

Governance

- ☐ ARB cadence active
- ☐ ADR template enforced
- ☐ Exception process with expiry
- ☐ Automated policy checks enabled
- ☐ Architecture health metrics reviewed quarterly

Risk & Compliance

- ☐ HIPAA controls mapped
- ☐ GenAI governance defined
- ☐ Vendor lock-in risks assessed
- ☐ Audit evidence automated

2️⃣ Solution Architect (SA) Checklist

(Program / initiative delivery)

Scope & Context

- ☐ Problem statement clear
- ☐ Product boundaries defined
- ☐ Dependencies mapped
- ☐ Legacy impact assessed
- ☐ Migration plan defined

Solution Design

- ☐ Solution diagram (C4 L2/L3)
- ☐ Enterprise standards applied
- ☐ Platform services reused
- ☐ Integration patterns approved
- ☐ No direct EHR or DB access

Security & Compliance

- ☐ PHI flows documented
- ☐ Threat model created
- ☐ IAM integrated
- ☐ Audit logging defined
- ☐ Consent enforced

Quality & NFRs

- ☐ Availability SLO defined
- ☐ Performance targets set
- ☐ DR/RTO/RPO defined
- ☐ Manual fallback for clinical flows
- ☐ Cost estimate reviewed

Delivery Governance

- ☐ ADRs written
- ☐ API contracts defined
- ☐ Data contracts defined
- ☐ CI/CD with security gates
- ☐ Runbooks planned

3️⃣ Application Architect / Tech Lead Checklist

(System / service-level execution)

Design

- ☐ Single bounded context
- ☐ Clean/hex architecture
- ☐ One DB per service
- ☐ Domain events defined
- ☐ No shared schemas

API & Integration

- ☐ OpenAPI documented
- ☐ Versioning defined
- ☐ Contract tests implemented
- ☐ Idempotency handled

- ☐ Retries & timeouts set

Security

- ☐ OAuth2/OIDC enforced
- ☐ RBAC/ABAC applied
- ☐ Secrets in vault
- ☐ PHI masked in logs
- ☐ Audit events emitted

Resilience

- ☐ Circuit breakers
- ☐ Bulkheads
- ☐ Graceful degradation
- ☐ Health checks
- ☐ Chaos tests planned

DevOps & Ops

- ☐ CI/CD implemented
- ☐ SAST/DAST/SCA enabled
- ☐ IaC used
- ☐ Canary/blue-green deploy
- ☐ Runbooks ready

⚙️ Review Automation Checklist

(Turn governance into code)

---

1. Pre-Commit Automation

- ☐ Linting
- ☐ Secret scanning
- ☐ Dependency vulnerability scanning
- ☐ IaC scanning
- ☐ Schema validation

---

2. CI Pipeline Automation

- ☐ Build + unit tests
- ☐ Contract tests
- ☐ SAST / DAST
- ☐ Container image scan
- ☐ License compliance check

---

3. Architecture Policy Checks (Critical)

- ☐ Approved frameworks only
- ☐ Platform services used
- ☐ No direct EHR endpoints
- ☐ No shared DB access
- ☐ Encryption enabled
- ☐ Logging enabled

(Use OPA / Sentinel / policy-as-code)

---

4. Data Governance Automation

- ☐ Schema registered
- ☐ Lineage captured
- ☐ Data quality checks
- ☐ Access policy enforced
- ☐ PHI classification applied

---

5. Security Automation

- ☐ IAM policy validation
- ☐ Least privilege check
- ☐ JIT access enforced
- ☐ DLP scans
- ☐ Threat model uploaded

---

6. Deployment Automation

- ☐ Blue/green or canary
- ☐ Health check validation
- ☐ Rollback automation
- ☐ Observability hooks enabled

---

7. Post-Deploy Automation

- ☐ SLO monitoring
- ☐ Cost anomaly detection
- ☐ Drift detection
- ☐ Access review automation
- ☐ Evidence archived for audit

🧠 How Mature Teams Use This

- EA checklist → quarterly governance
- SA checklist → design review gate
- App checklist → PR / pipeline gate
- Automation checklist → **no-human gatekeeping**

🎯 Interview-ready summary (use this)

“I scale architecture by giving each role a checklist and automating the review gates. Architects define the rules, pipelines enforce them, and teams move fast without breaking compliance or safety.”
