Observability & Engineering Effectiveness Architecture Teachers Timetable Management System

Principal Architect View • Production-Grade • Observability + DORA Metrics

This document defines a comprehensive observability and engineering effectiveness strategy for the Teachers Timetable Management System. It combines logs, metrics, traces, health checks, alerting, incident management, and DORA metrics into a single, coherent operating model.

1. Observability Vision & Goals

- End-to-end transparency across the system
- Early detection of failures and performance degradation
- Reduced Mean Time To Detect (MTTD) and Mean Time To Recover (MTTR)
- Actionable insights for engineers, leadership, and business stakeholders
- Objective measurement of delivery performance using DORA metrics

If you cannot observe a system, you cannot reliably operate or scale it.

2. The Three Pillars of Observability

2.1 Logs – What happened?

Logs provide a chronological narrative of system behavior. Logs must be structured, searchable, and correlated.

- Structured JSON logs
- CorrelationId / TraceId
- TimetableId (Aggregate Identifier)
- UserId, Role, Operation Name
- Clear severity levels (Info, Warn, Error, Critical)

2.2 Metrics – How is the system behaving?

Metrics are aggregated numerical signals sampled over time.

- Golden signals: latency, traffic, error rate, saturation
- Infrastructure metrics: CPU, memory, disk, queue depth

- Business metrics: timetables created, conflicts detected
- AI metrics: OCR accuracy, AI rejection rate

2.3 Traces – Where did time go?

Distributed tracing shows the end-to-end execution path of a request.

- W3C Trace Context propagation
- Cross-service tracing (API → Domain → SQL → Service Bus → AI)
- Async trace continuation for background processing

3. Observability Tooling Strategy

3.1 Azure-Native Tooling

- Azure Application Insights – application-level telemetry
- Azure Monitor – platform and infrastructure metrics
- Log Analytics Workspace – centralized log storage
- Azure Alerts & Action Groups – alerting and notifications

3.2 Datadog (Optional / Advanced)

- Unified dashboards across cloud and environments
- Advanced APM and distributed tracing
- Anomaly detection using ML
- SLO-based alerting and error budgets

Tools collect signals; architecture gives those signals meaning.

4. Health Checks & Dependency Monitoring

- Liveness checks – is the process running?
- Readiness checks – can the service accept traffic?
- Dependency checks – SQL, Blob Storage, Service Bus, AI services

5. Alerting & Stakeholder Notification Model

- Threshold-based alerts for known limits
- Anomaly-based alerts for unknown failure modes
- Composite alerts to reduce noise and alert fatigue

Who Gets Notified?

- Engineers – PagerDuty / Microsoft Teams
- Operations – On-call escalation rotation
- Business stakeholders – SLA & availability dashboards

6. Incident Management Workflow

- Alert fired by monitoring system
- On-call engineer notified
- Root cause analysis using logs, metrics, and traces
- Mitigation or rollback
- Post-incident review and action items

Every incident is a learning opportunity. Blameless postmortems are mandatory.

7. DORA Metrics – Measuring Engineering Effectiveness

DORA metrics provide an objective, research-backed view of software delivery performance. In this system, DORA metrics are derived directly from observability and CI/CD data.

7.1 Deployment Frequency

- Definition: How often the organization deploys to production
- Derived from: CI/CD pipeline deployment events
- Value: Measures delivery velocity and confidence

7.2 Lead Time for Changes

- Definition: Time from code commit to production
- Derived from: Git commit timestamps + pipeline execution time
- Value: Indicates efficiency of delivery pipeline

7.3 Change Failure Rate

- Definition: Percentage of deployments causing incidents or rollbacks
- Derived from: Deployment logs + incident alerts
- Value: Measures release quality

7.4 Time to Restore Service (MTTR)

- Definition: Time to recover from production incidents
- Derived from: Alert fired → alert resolved timestamps
- Value: Measures operational resilience

High-performing teams optimize for stability and speed simultaneously.

8. Mapping Observability Signals to DORA Metrics

- Logs and traces identify failed deployments (Change Failure Rate)
- Alerts and incidents determine MTTR
- Pipeline telemetry feeds Deployment Frequency
- Commit-to-prod tracing feeds Lead Time for Changes

9. Security, Privacy & Compliance

- PII masking and redaction in logs
- Role-based access to observability dashboards
- Log retention and data residency policies
- Audit-ready trails for compliance

10. Closing Summary

High-performing systems observe both production behavior and engineering behavior.

This observability architecture integrates system telemetry with DORA metrics, enabling the Teachers Timetable Management System to achieve reliability, fast recovery, continuous delivery excellence, and executive-level visibility.