Standard for SaaS Enterprise Applications

Contents

1. Purpose

This standard sets the premise of Enterprise Applications as IT systems, which serve in the execution of business processes and roles in executing the processes. Generally, the criticality (see Section 9) of a business process drives certain expectations from the "serving" application.

The standard addresses Enterprise Applications provided via Software as a Service. This document captures Information Technology related guidelines, requirements and expectations for enterprise and customer-facing applications, for their ongoing support and operations.

Detailed specifications are identified under the following technology domains:

- Application Non-Functional Requirements (NFR)
- Application Hosting
- Embedded Intelligence
- Modeling Business and Operations

2. Scope

This statement of Information Technology guidelines and expectations applies to all enterprise and customer-facing applications, whether owned and operated by the University, or used for university business through

contractual arrangements.

3. Standard

3.1. Application Non-Functional Requirements (NFR)

3.1.1. Disaster Recovery and Continuity of Operations

(see Section 9 explaining availability, RTO and RPO requirements)

1. Availability during business hours (including planned and unplanned downtime) measured and reported monthly: 99.90 percent (allowing for up to 2 hours 5 minutes of downtime annually)
2. Availability overall (including planned and unplanned downtime overall (i.e. including business hours)) measured and reported monthly: 99.90 percent (allowing for up to 8 hours 45 minutes of downtime annually)
3. Recovery Time Objective (RTO): 12 hours
4. Recovery Point Objective (RPO): 1 hour
5. Disaster Recovery Test frequency: Annual testing and certification. Open to Customer participation.
6. Notification, escalation and recovery time: Proactive notifications initiated by the service provider, sent to subscribed recipients. See further details about Incident Management, Change Management, and Self Reporting.

3.1.2. Operational Support

Incident Management

(see Section 9 explaining severity levels)

1. Support ticket requests accepted from all customer segments e.g. faculty, students and administrators 24x7x52 for rapid categorization and prioritization by severity. Each ticket is assigned to a severity level between 1 through 4.
2. Tickets assigned Severity Level 1 are to be acknowledged in 15 minutes and fixed within 4 hours
3. Tickets assigned Severity Level 2 are to be acknowledged in 60 minutes and fixed within 24 hours
4. Tickets assigned Severity Level 3 are to be acknowledged in 24 hours and fixed within 5 working days
5. Tickets assigned Severity Level 4 are to be acknowledged in 48 hours and fixed within 10 working days

Self-Reporting

1. Operational metrics: e.g. Performance and Usage reported within 24 hours, User activity, User behavior

2. Outages and exceptions notified within 30 minutes
3. Incident tracking: Realtime dashboards, Trends, Escalations to Problems

Change Management

1. Release upgrades: 1-month prior notice.
2. Minor version upgrades: 1-week prior notice.
3. Emergency fix: 4 hours prior notice.
4. Automated deployment of production and non-production environments
    a. Automated deployment scripts should be used to include configuration of product application, data and infrastructure.
    b. All deployments should refresh non-production environment(s), prior to production.
5. Non-production environment(s) should be accessible for testing and training purposes.

3.1.3. Product Performance — User Experience

1. Interaction cost and time
    a. Not more than 3 clicks
    b. Below 30 seconds to initiate action
2. Response time (Latency) - Browser
    a. 2.5 sec Largest Contentful Paint (LCP)
    b. 0.5 sec Interaction to Next Paint (INP)
    c. 0.1 sec Cumulative Layout Shift (CLS)
3. Response time (Latency) - Mobile App
    a. 1 sec refresh on mobile app
    b. 0.1 sec API response
    c. 0.01 sec event delivery

3.1.4. Product Performance — System Performance

1. System elasticity
    a. Ability to scale 10x on-demand, including disk and bandwidth
2. Data consistency across integrated systems (System integrity)
    a. Less than 60 seconds for Mission critical data

b. Less than 5 minutes for Business-critical data

c. Less than 1 hour for Business-support data

d. Less than 24 hours for Business-convenience data

This includes Virginia Tech data related to product transactions, logs, audit-trail, usage available for API or bulk exchange (e.g. ETL)

3. Continuous improvement

a. Meeting cadence: monthly

Regular meetings between Contact Administrators and Virginia Tech Technical Leads. Technical Leads will discuss functional and non-functional product features, product roadmaps, use cases, and business process guidance or best practices.

4. Access to Data

a. Virginia Tech will have complete and full access to data on the system with real-time access (e.g. via APIs) and for bulk download (e.g. via batch). This includes application data, transactional data, logs, data related usage (e.g. audit trails) and user experience. (measured monthly)

5. API performance

a. API latency: Average latency of 100ms with a maximum p95 and p99 latency of 200ms and 500ms

b. API error rate: Less than 0.5 percent failed requests

c. API Throughput: 1,000+ requests per second (RPS)

d. API Availability: (includes planned and unplanned downtime) measured and reported monthly: 99.90 percent (allowing for up to 8 hours 45 minutes of downtime annually)

3.2. Application Hosting

3.2.1. Integrate-ability

1. Applications should provide standards-based integration capabilities using documented and supported interfaces such as REST APIs, webhooks, event streams, batch interfaces, and file-based exchange mechanisms where appropriate.
2. Integration interfaces should support secure authentication and authorization mechanisms consistent with university standards (e.g. OAuth2, OpenID Connect, SAML, API keys, mutual TLS).
3. Application Programming Interfaces (APIs) should be versioned, documented, and backward compatible for a reasonable transition period following release upgrades.
4. Applications should support event-driven integration patterns to enable near real-time processing of operational and business events across enterprise platforms and services.

5. Integration capabilities should support extraction and exchange of transactional data, master/reference data, audit records, operational metrics, and configuration data.
6. Bulk import and export capabilities should support commonly adopted structured data formats (e.g. JSON, CSV, XML, parquet or equivalent) and frequencies at nominal predefined costs, if any.
7. Applications should provide mechanisms for integration monitoring, error handling, retry processing, and dead-letter or failed transaction reporting.
8. Applications should support integration with enterprise workflow, IT service management, observability, identity management, and data integration platforms.

3.2.2. Observability

1. Applications should provide operational visibility into application health, availability, performance, capacity, and usage.
2. Applications should expose logs, metrics, events, and audit records through accessible interfaces or export mechanisms suitable for enterprise monitoring and analysis platforms.
3. Operational telemetry should support integration with enterprise observability and monitoring platforms (e.g. SIEM, APM, log aggregation, event management, and analytics platforms).
4. Applications should support proactive alerting for service degradation, outages, threshold breaches, integration failures, and abnormal operational behavior.
5. Operational events and alerts should include sufficient contextual information to support incident triage, troubleshooting, root cause analysis, and operational reporting.
6. Applications should retain operational logs, audit records, and telemetry for a customer-defined retention period consistent with university policy and regulatory requirements.
7. Applications should support correlation between operational events and transactions across integrated systems using trace identifiers, transaction identifiers, or equivalent mechanisms.
8. Observability capabilities should support both real-time operational monitoring and historical trend analysis.

3.2.3. Auditability

1. Applications should maintain audit records for authentication events, authorization changes, configuration changes, administrative actions, data access, and data modifications.
    a. Audit records should include timestamp, initiating identity, originating system or source, action performed, and affected object or record.
        i. Audit records should be immutable.
        ii. Retention of audit records should be configurable to align with Virginia Tech policy.

iii. Audit records should be exportable for compliance, forensic investigation, and operational analysis purposes.

3.2.4. Proactive Reporting

1. Applications should provide configurable dashboards and reporting capabilities for operational health, service usage, integration activity, and business transaction monitoring.
2. Service providers should proactively communicate known service degradation, outages, maintenance activities, security events, and operational risks affecting university services.
3. Applications should support scheduled and on-demand reporting through APIs, dashboards, and export capabilities.

3.2.5. Configurability

1. The system should allow for meaningful process and behavior changes without code modification e.g. support for vanity URLs.
    a. These configuration changes should survive product upgrades.
    b. Configuration changes should be auditable (see Section 3.2.3)

3.2.6. Extensibility

The system should support separate configurations per environment, and these configurations should be promotable across environments.

1. The system should define supported extension points, such as APIs, SDKs, event subscriptions, and webhooks.
2. The system should remain operationally separate from any extensions such that the system and any extensions can be deployed and upgraded independently of one another.
    a. Any compatibility requirements should be clearly documented and advanced notice given for deprecated extension points.

3.2.7. Information Security

See standards related to Identity and Access Management, Data Security and Data Privacy.

3.2.8. Access and Authorization

1. Access policy and authorization rules must be managed centrally within Grouper
    a. Authentication should be via an authorized university identity provider whenever possible
        i. Single Sign-On (SSO) integrations should support SAML or OIDC

ii. SSO integrations should support group-based (RBAC/ABAC) authorization mechanisms

b. Each user should only receive the access they require based on their affiliation or job duties and not more

c. Security Group, Class, and Role naming will conform to established standards

d. Creation of new Security Groups will start with minimum access and expand as necessary

e. Virginia Tech objects will be kept separate from delivered objects, whenever possible

f. Delivered Security Groups, Classes and Roles will be removed from the VT environment whenever possible

3.3. Embedded Intelligence

Enterprise applications and software-as-a-service offerings that include embedded intelligence, artificial intelligence, machine learning, generative AI, or agent-based capabilities should document and disclose how those capabilities are used in support of university business processes.

At a minimum, application owners, service providers, or vendors should provide the following information.

3.3.1. Model Orchestration and Governance

The application should describe how AI or machine learning capabilities are implemented, managed, and maintained within the service. This includes:

- the methods used for model registry, versioning, training pipeline management, packaging, and deployment;
- any orchestration frameworks, model-serving interfaces, or OpenAI-compatible APIs used by the application;
- any agent integration or interoperability protocols used by the application, including protocols such as Model Context Protocol (MCP) or Agent-to-Agent (A2A), where applicable;
- the methods used for prompt management, context assembly, retrieval-augmented generation (RAG), or other context-injection techniques, where applicable;
- the method AI agents connect to user-facing applications enabling real-time, multi-modal or interactive experiences such as AG-UI.

The application should include an evaluation process for AI guardrails, policy enforcement, and ongoing review of system behavior. This process should align with the University’s Responsible and Ethical Artificial Intelligence Framework and other applicable university guidance.

The application should identify the business use cases supported by embedded intelligence. Examples may include prediction, forecasting, planning, decision support, classification, recommendation, intelligent automation, anomaly detection, content generation, summarization, conversational interfaces, and knowledge

discovery. The application should also identify known limitations, prohibited uses, and relevant antipatterns where established best practices indicate heightened risk.

3.3.2. Observability and Monitoring

The application should support operational monitoring of AI-enabled services. At a minimum, this includes standard service metrics such as latency, throughput, error rates, and timeout rates.

The application should also support monitoring of model behavior and output quality, as appropriate to the use case. This includes, where applicable:

- token usage and associated cost;
- response length;
- output format compliance;
- safety compliance;
- detection or monitoring of bias, hallucinations, toxicity, and exposure of personally identifiable information or other sensitive data.

Where embedded intelligence materially influences application outputs or decisions, the application should support diagnostic and explainability methods appropriate to the use case. Examples may include techniques such as LIME or SHAP, as well as monitoring for model drift or other degradation in performance over time.

3.3.3. Data Security and Privacy

All university data used, processed, transmitted, stored, or generated by embedded intelligence capabilities should be protected in accordance with applicable Virginia Tech policies and standards.

At a minimum, applications that handle university data should meet the requirements of the Standard for High-Risk Digital Data Protection, where applicable.

Application owners and service providers should ensure that embedded intelligence capabilities do not weaken university requirements for data security, privacy, confidentiality, access control, retention, or appropriate handling of sensitive information.

3.4. Architecture Models

ArchiMate is the preferred modeling language for architecture artifacts because it provides a consistent, vendor-neutral way to describe and relate business, application, data, and technology elements. Using ArchiMate helps ensure models can be understood, compared, and reused across domains and over time. These will include developing Solution Architecture models and Support models.

3.4.1. Capability Models

The EDUCAUSE Higher Education Reference Model (HERM), including the Business and Service Reference Models, serves as the primary reference point for classifying higher-education capabilities and services. Aligning architecture artifacts to HERM promotes sector relevance, shared understanding, and external comparability. Alternatives or extensions are acceptable when they provide clearer insight, with the expectation that their relationship to HERM concepts is explained.

4. Compliance and Enforcement

It is the responsibility of University IT service providers to ensure that the controls described in this document are implemented. University departments undergo periodic audits. These audits sometimes include an analysis of the processes and controls used by departments to secure and manage end user devices, servers, and applications. The department is responsible for remediation of any findings of noncompliance with this standard within the time frame agreed to with the auditors.

5. Definitions

See the IT Glossary and Division of IT Acronyms Glossary for definitions and acronyms not defined here.

6. References

- *IT Disaster Recovery Standard* — https://it.vt.edu/content/dam/it_vt_edu/policies/IT_Disaster_Recovery_Standard-v1-3-2026.pdf
- *Standard for High Risk Digital Data Protection* — https://it.vt.edu/content/dam/it_vt_edu/policies/Standard-for-High-Risk-Digital-Data-Protection.pdf
- *IT Glossary* — https://it.vt.edu/content/dam/it_vt_edu/policies/IT-Glossary-for-policies-and-standards.pdf
- *Division of IT Acronyms Glossary* — https://it.vt.edu/resources/acronyms.html

7. Maintenance of Standard

The Enterprise Architecture office is responsible for the development and maintenance of the standard. The implementation and compliance of the standard is the responsibility of University Department Heads within their respective departments, and IT service providers. Questions may be directed to Enterprise-Architecture-g@vt.edu.

8. Revision History

Version	Date	Description
1.0	May 2026	Initial version of the standard developed in May 2026. Reviewed by the Enterprise Architecture Community of Practice in May 2026. Approved by the Associate VP for IT Governance, Planning, and Strategy on May 20, 2026.

9. Endnotes

9.1. Process Criticality

Criticality Level	Description
Mission critical function	Disruptions of Mission critical functions (also referred to as Essential Functions) will have a crippling effect on operations. Considered a showstopper with no practical workarounds. Impact is generally pervasive across one or more divisions or colleges.
Business critical function	Disruptions of Business-critical functions will have a significant operational impact. Work-arounds available to maintain operational stability for a brief period of time (up to 72 hours), require extensive manual interventions. Impact is generally pervasive across one or more divisions or colleges.
Business support function	Disruptions of Business support functions will lead to considerable operational impact. Work-arounds are available to maintain operational stability over a short period of time (duration to be defined). Impact is generally local to a division or college.
Business convenience function	Disruptions of Business convenience functions will lead to tolerable operational impact. Work-arounds are available to maintain operational stability over an extended period of time (duration to be defined). Impact is generally local to a department or functional unit in a division or college.

9.2. Disaster Recovery and Continuity of Operations

Criticality Level	Availability (Business Hours)	Availability (Overall)	RTO	RPO
Mission critical function	99.90%	99.90%	12 hours	1 hour
Business critical function	99.50%	99.50%	72 hours	1 hour

Criticality Level	Availability (Business Hours)	Availability (Overall)	RTO	RPO
Business support function	97.00%	97.00%	To be defined	To be defined
Business convenience function	95.00%	95.00%	To be defined	To be defined

9.3. Support Ticket Severity

The purpose of these descriptions is to align incident severities generally associated with applications to the business processes served by these applications.

The severity of an incident, in this context, considers the impact of an outage of the application or an application feature, on the ability to execute the business process, and the level of effort required to temporarily overcome the disruption.

Severity Level	Applicable Function
Severity Level 1	Mission critical function(s)
Severity Level 2	Business-critical function(s)
Severity Level 3	Business support function(s)
Severity Level 4	Business convenience function(s)