Civitas AI – Product Requirements Document (PRD)

1. Product Overview

Product Name: Civitas AI (working name)

Tagline: Real-time, explainable, policy-driven text moderation for web, mobile, and AI systems.

Civitas AI is a web and mobile platform that provides real-time text moderation using the Hugging Face Friendly Text Moderator API. The product is designed as trust-and-safety infrastructure, not merely an API wrapper. It supports configurable policies, explainable decisions, human-in-the-loop review, and audit-grade evidence suitable for enterprise and regulated environments.

---

2. Problem Statement

Organizations hosting user-generated content or deploying AI-driven conversational systems face increasing risk from toxic, abusive, or policy-violating text. Existing moderation solutions often lack transparency, configurability, and auditability. Teams need a solution that operates in real time, adapts to different policies and jurisdictions, and provides clear evidence for why moderation actions were taken.

---

3. Goals and Objectives

Primary Goals

- Moderate text in real time with low latency
- Provide explainable moderation outcomes
- Support configurable, policy-driven enforcement
- Enable human review and auditability

Success Metrics

- Median latency under 200 ms
- 99.9% service availability
- Sub-1% sustained false-positive override rate
- Developer integration time under 30 minutes

---

4. Target Users

- End users submitting or consuming text (chat, comments, reviews)

- Trust & Safety teams
- Moderators
- Compliance and Legal teams
- Product and Engineering teams

---

5. Core Use Cases

1. Real-time moderation of chat and messaging
2. Pre-submission filtering for comments or posts
3. Guardrails for AI assistants (prompt and response moderation)
4. Post-hoc moderation review and dispute handling
5. Compliance reporting and audit review

---

6. Functional Requirements

6.1 Real-Time Text Moderation

- Integrate with the Hugging Face Friendly Text Moderator API
- Multi-label classification (toxicity, hate, harassment, sexual content, violence, profanity)
- Configurable confidence thresholds

6.2 Explainability

- Category scores per moderation request
- Clear, human-readable explanations
- Optional LLM-generated rationale for reviewers

6.3 Policy Engine

- Policy definition by content type, user type, and region
- Actions: allow, warn, block, escalate
- Versioned policies with effective dates

6.4 Human-in-the-Loop Review

- Queue for flagged content
- Review actions: approve, reject, edit, escalate
- Feedback captured as structured data

6.5 User Feedback Experience

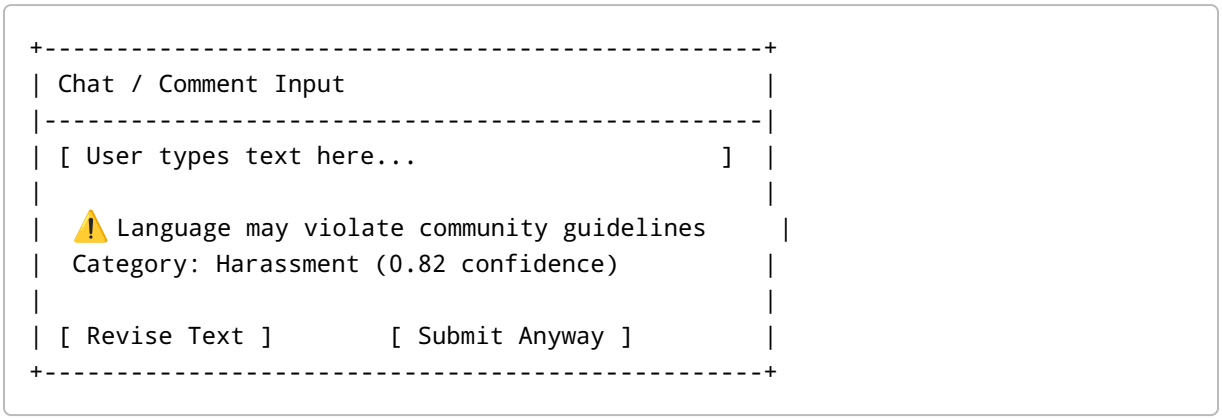
- Inline warnings during text entry
- Non-punitive, educational messaging
- Optional rewrite suggestions

---

7. Wireframe and UX Flow

7.1 End User Text Submission Flow

Screen: Text Input / Chat Composer

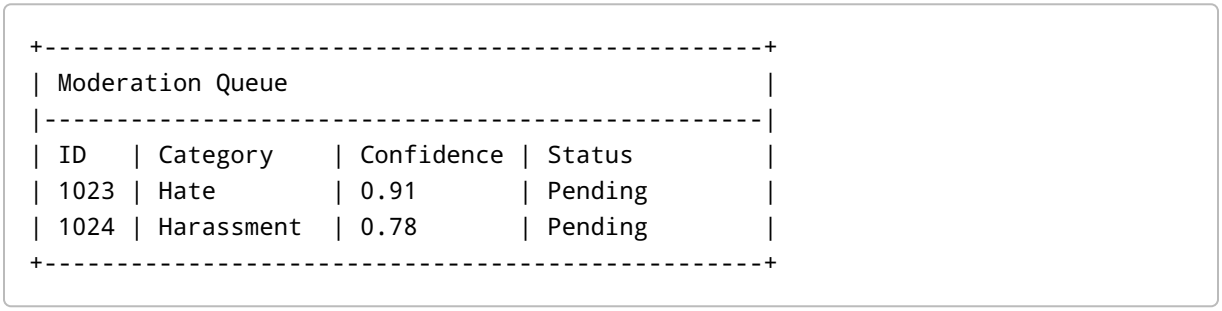


States: - Neutral: No warnings, submit enabled - Warning: Inline explanation, soft block - Blocked: Submission disabled, clear reason shown

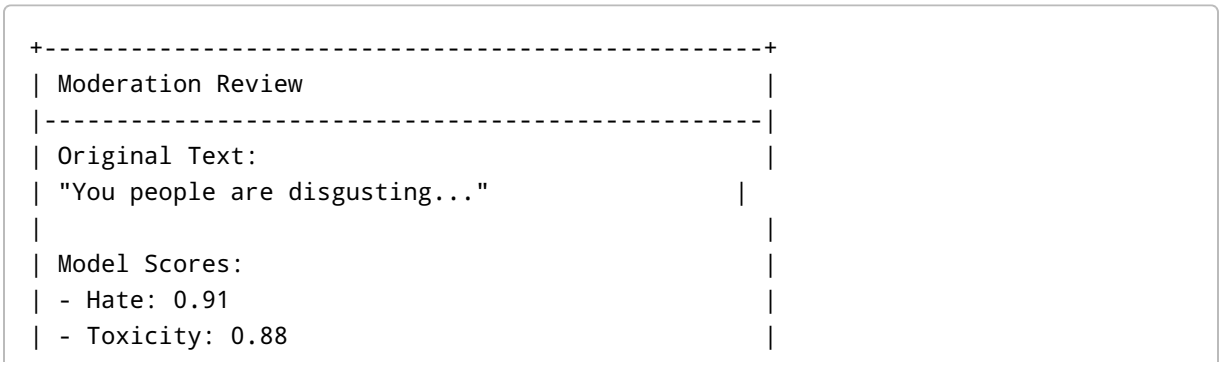
---

7.2 Moderator Review Flow

Screen: Moderator Queue



Screen: Moderation Detail View



```
|                                                  |
| Policy Triggered: Social-Standard v1.3           |
| Action Suggested: Block                          |
|                                                  |
| [ Approve Block ] [ Override ] [ Escalate ]      |
+--------------------------------------------------+
```

7.3 Admin / Policy Management Flow

Screen: Policy Editor

```
+--------------------------------------------------+
| Policy Editor – Youth Safe Mode                  |
|--------------------------------------------------|
| Category        | Threshold | Action             |
| Hate            | 0.60      | Block              |
| Harassment      | 0.50      | Warn               |
| Profanity       | 0.70      | Warn               |
|                                                  |
| Effective Date: 2026-03-01                        |
| Version: 1.0                                     |
|                                                  |
| [ Save Draft ]        [ Publish Policy ]         |
+--------------------------------------------------+
```

These wireframes are designed for direct handoff to Figma, with each screen mapping cleanly to a component-based design system.

---

8. Agentic SDLC Mapping (Policy → Control → Evidence)

8.1 Policy

- Content moderation policies defined declaratively
- Policies aligned to enterprise and regulatory frameworks
- Versioned, time-bound, and jurisdiction-aware

8.2 Controls

- Automated moderation control via Hugging Face Friendly Text Moderator
- Deterministic policy evaluation engine
- Human-in-the-loop review as compensating control
- Separation of duties between policy authoring and review

8.3 Evidence

- Every moderation decision generates audit evidence
- Evidence artifacts include:
- Content hash or encrypted content
- Model name and version
- Category scores
- Policy ID and version
- Automated decision
- Human override (if any)
- Timestamp and actor

This structure allows Civitas AI to operate as a governed AI control system inside an agentic SDLC, where agents execute controls continuously and humans supervise exceptions.

---

9. Non-Functional Requirements

- Scalability to millions of requests per day
- Horizontal scaling of moderation service
- Strong observability (logs, metrics, traces)
- Configurable data retention

---

10. Security and Compliance

- TLS for all data in transit
- Role-based access control
- API key and OAuth authentication
- GDPR and CCPA-aligned deletion workflows
- No customer data used for model training

---

11. Controls Matrix (NIST / ISO / AI RMF Aligned)

Control Area	Framework Reference	Control Description	Civitas AI Implementation	Evidence Generated
Harmful Content Detection	NIST AI RMF – MAP	Identify harmful content risks	Automated text classification	Model scores, logs
Policy Enforcement	ISO/IEC 42001 – Clause 8	Enforce AI usage policies	Policy engine with thresholds	Policy version, decision
Human Oversight	NIST AI RMF – GOV	Human-in-the-loop review	Moderator workflows	Review actions

Control Area	Framework Reference	Control Description	Civitas AI Implementation	Evidence Generated
Transparency	ISO/IEC 23894	Explain AI decisions	Explainable outputs	Explanation text
Auditability	NIST AI RMF – MEASURE	Enable audits and monitoring	Immutable audit logs	Evidence records
Continuous Improvement	NIST AI RMF – MANAGE	Improve system performance	Feedback loops	Override analytics

This matrix enables direct traceability from regulatory expectations to implemented controls and generated evidence.

---

12. Analytics and Reporting

- Moderation volume by category
- False positive and false negative trends
- Policy effectiveness metrics
- Exportable audit logs

---

12. Roadmap

Phase 1

- Core moderation service
- Web demo application
- Basic admin dashboard

Phase 2

- Policy engine
- Human review workflows
- Mobile SDKs

Phase 3

- Multi-model orchestration
- Adaptive thresholds
- Advanced trust and safety analytics

---

13. Risks and Mitigations

- False positives mitigated through human review
- Latency mitigated via async processing and caching
- Regulatory risk mitigated through strong auditability

---

14. Product Backlog (Pre-Jira)

This backlog represents implementation-ready work items written in a tool-agnostic format. Items are grouped by theme and ordered roughly by delivery sequence. Each item includes acceptance criteria to support agentic and human validation before promotion.

Epic A: Core Moderation Platform

A1. Text Moderation API Integration
Acceptance Criteria: - Moderation requests successfully call the Hugging Face API - Responses include category scores and confidences - Timeouts and API failures are handled gracefully

A2. Moderation Request Pipeline
Acceptance Criteria: - Pipeline supports synchronous and async calls - Requests are traceable via correlation IDs - Throughput meets defined latency SLAs

A3. Latency Optimization & Caching
Acceptance Criteria: - Repeated text yields cached results when enabled - Cache invalidation respects policy changes

---

Epic B: Policy Engine

B1. Policy Data Model
Acceptance Criteria: - Policies include thresholds, actions, version, and scope - Policies are immutable once published

B2. Policy Evaluation Service
Acceptance Criteria: - Given model scores, correct action is deterministically selected - Evaluation output references policy version

B3. Policy Versioning & Rollback
Acceptance Criteria: - New policy versions do not affect historical decisions - Rollback restores previous effective behavior

B4. Region and Context Awareness
Acceptance Criteria: - Policies resolve correctly based on region and context metadata

Epic C: End-User Experience

C1. Real-Time Inline Feedback
Acceptance Criteria: - Feedback appears within 200 ms of user input - Warn and block states match policy outcomes

C2. User Messaging Content
Acceptance Criteria: - Messages are human-readable and policy-aligned - No internal model details are exposed

C3. Optional Rewrite Suggestions
Acceptance Criteria: - Suggestions align with policy intent - Feature can be disabled per deployment

---

Epic D: Moderator Experience

D1. Moderation Queue
Acceptance Criteria: - Queue displays only actionable items - Sorting and filtering function correctly

D2. Moderation Detail View
Acceptance Criteria: - Displays text, scores, policy, and recommendation

D3. Review Actions
Acceptance Criteria: - Actions are persisted and auditable

D4. Feedback Capture
Acceptance Criteria: - Overrides are recorded as structured data

---

Epic E: Admin & Governance

E1. Policy Management UI
Acceptance Criteria: - Draft and published states are supported

E2. Role-Based Access Control
Acceptance Criteria: - Only authorized roles can modify policies

E3. Audit Log Viewer
Acceptance Criteria: - Evidence can be searched and exported

Epic F: Agentic SDLC & Evidence

F1. Evidence Schema Definition
Acceptance Criteria: - Evidence objects reference policy, control, and outcome

F2. Immutable Evidence Storage
Acceptance Criteria: - Stored evidence cannot be altered

F3. Evidence Export
Acceptance Criteria: - Exports match stored evidence exactly

Epic G: Analytics & Monitoring

G1. Moderation Metrics
Acceptance Criteria: - Metrics update in near real time

G2. False Positive / Override Analytics
Acceptance Criteria: - Overrides are clearly attributable

G3. Policy Effectiveness Dashboard
Acceptance Criteria: - Trends are viewable over time

Epic H: Platform & Security

H1. Authentication & API Keys
Acceptance Criteria: - Unauthorized access is blocked

H2. Data Retention Controls
Acceptance Criteria: - Retention settings are enforced

H3. Observability
Acceptance Criteria: - Logs and metrics are available for all services

---

15. Domain Model

Core Entities

TextSubmission - id - content (or hash) - context metadata - timestamp

ModerationDecision - id - submission_id - model_name and version - category_scores - policy_id and version - automated_action

Policy - id - version - thresholds - actions - scope

ReviewAction - decision_id - reviewer_id - action - rationale

EvidenceRecord - policy_reference - control_reference - decision_reference - timestamp

16. RACI and Agent Responsibility Map

Activity	AI Agent	Human Moderator	Admin	Auditor
Text Classification	R	I	I	I
Policy Evaluation	R	I	A	I
Inline User Feedback	R	I	I	I
Moderation Review	C	R	I	I
Policy Authoring	I	I	R	A
Evidence Generation	R	I	I	A
Audit Review	I	I	C	R

R = Responsible, A = Accountable, C = Consulted, I = Informed

---

17. Agentic Acceptance Validation Model

Each backlog item's acceptance criteria is validated by a combination of automated agents and human review, forming promotion gates across environments.

Validation Layers

- **Static Validation Agent**: Verifies schemas, policy references, and control IDs
- **Behavioral Test Agent**: Executes functional tests against acceptance criteria
- **Policy Compliance Agent**: Confirms actions align with active policy versions
- **Evidence Verification Agent**: Confirms evidence records are generated and immutable
- **Human Review (Exception-Based)**: Triggered only when agents disagree or thresholds are crossed

Promotion Rules: - All automated agents must pass for standard promotion - Human approval required only for overridden or escalated items

---

18. Control ID Taxonomy

All controls use a deterministic ID scheme to support traceability across policy, code, and evidence.

Format: `MOD-XXX`, `POL-XXX`, `GOV-XXX`, `AUD-XXX`

Examples: - MOD-001: Automated text classification - MOD-002: Real-time user feedback enforcement - POL-001: Threshold-based moderation policy - POL-003: Regional policy resolution - GOV-002: Human-in-the-loop review - AUD-001: Immutable evidence storage

Control IDs are referenced in: - Policy definitions - Code annotations - Evidence records - Audit exports

---

19. Threat Model and Abuse Cases

Key Threat Scenarios

1. **Prompt Evasion**: Users attempt to bypass moderation through obfuscation
2. **Over-Moderation Risk**: Legitimate speech incorrectly blocked
3. **Under-Moderation Risk**: Harmful content passes undetected
4. **Policy Tampering**: Unauthorized changes to moderation thresholds
5. **Evidence Manipulation**: Attempts to alter audit records

Mitigations

- Ensemble and threshold tuning
- Human review as compensating control
- RBAC and separation of duties
- Append-only evidence storage
- Full policy version traceability

---

20. Promotion Gates and Environment Progression

Environments

- Development
- Staging
- Production

Gate Requirements

Gate	Requirement
Dev → Staging	All agent validations pass
Staging → Prod	Evidence verification + no unresolved overrides
Emergency Patch	Human approval + post-hoc audit

---

21. Reference Implementation Structure

```
/civitas-ai
  /apps
    /web
```

```
    /mobile
  /services
    /moderation
    /policy-engine
    /review
  /agents
    /validation
    /policy-compliance
    /evidence-checker
  /schemas
    policy.json
    decision.json
    evidence.json
  /controls
    control-registry.yaml
  /docs
    architecture.md
    threat-model.md
    controls-matrix.md
```

This structure supports agentic validation, clean ownership boundaries, and direct traceability from requirements to running code.

---

22. Control-to-Code Annotation Example

To ensure traceability from requirements to running systems, controls are explicitly annotated in code and configuration.

Example: Moderation Decision Service (pseudo-code)

```
// Control: MOD-001 Automated Text Classification
// Policy Reference: POL-001 Threshold-Based Moderation
// Evidence Output: AUD-001 Immutable Evidence Record

function moderateText(inputText, context) {
  const modelResult = runModeratorModel(inputText); // MOD-001
  const policy = resolvePolicy(context);             // POL-001
  const decision = evaluatePolicy(policy, modelResult);

  writeEvidence({                                    // AUD-001
    controlId: "MOD-001",
    policyId: policy.id,
    modelVersion: modelResult.version,
    decision
  });
```

```
  return decision;
}
```

This pattern ensures every execution path can be traced to a defined control and produces audit-ready evidence.

---

23. Sample Evidence Record (JSON)

```
{
  "evidence_id": "ev-8f23a1",
  "timestamp": "2026-03-12T14:22:09Z",
  "submission_id": "sub-10293",
  "control_id": "MOD-001",
  "policy_id": "POL-001",
  "policy_version": "1.3",
  "model": {
    "name": "hf-friendly-text-moderator",
    "version": "2025-11"
  },
  "scores": {
    "hate": 0.91,
    "toxicity": 0.88
  },
  "automated_action": "block",
  "human_override": null,
  "immutable": true
}
```

This record represents the minimum viable evidence required to support audit, investigation, and regulatory review.

---

24. Executive Summary (Non-Technical)

Civitas AI is a trust and safety platform designed for organizations deploying user-generated content or AI-driven conversations at scale. Unlike traditional moderation tools, Civitas AI embeds governance directly into the system by linking policies to automated controls and continuously generating audit-grade evidence.

The platform uses AI to moderate content in real time while preserving transparency, accountability, and human oversight. Every moderation decision is explainable, policy-driven, and traceable, enabling organizations to demonstrate responsible AI use without slowing down innovation.

By integrating agent-based validation, human review where necessary, and immutable evidence generation, Civitas AI allows enterprises to scale safely, meet emerging AI governance expectations, and maintain user trust.

---

25. Reusable PRD & Governance Template

This document is designed to function as a **repeatable template** for governed AI systems beyond text moderation.

Template Sections Intended for Reuse

- Product Overview & Problem Statement
- Policy → Control → Evidence model
- Agentic Acceptance Validation Model
- Control ID Taxonomy
- Threat Model & Abuse Cases
- Promotion Gates
- Control-to-Code Annotation Pattern
- Evidence Schema

To reuse: 1. Replace domain-specific categories (e.g., moderation) with new AI capability 2. Retain governance, agent, and evidence sections unchanged 3. Generate new control IDs scoped to the domain

This enables consistent governance across multiple AI products with minimal overhead.

---

26. Agent Rules Generated from PRD

The PRD directly informs system-wide agent rules used in the Agentic SDLC.

Example Agent Rules

Validation Agent Rules - Every backlog item must reference one or more control IDs - Every control must generate evidence - Promotion is blocked if evidence is missing or mutable

Policy Compliance Agent Rules - Policy version used in execution must match published policy - Deprecated policies cannot be evaluated

Evidence Agent Rules - Evidence records must be append-only - Evidence must reference control and policy IDs

These rules can be expressed as configuration, Rego, or embedded agent logic and are derived directly from this PRD.

---

27. NIST AI RMF Artifact Mapping

This section maps Civitas AI artifacts to NIST AI RMF documentation expectations.

NIST AI RMF Function	Artifact Produced
GOVERN	Policy definitions, RACI, control taxonomy
MAP	Threat model, abuse cases, risk scenarios
MEASURE	Model scores, override metrics, analytics
MANAGE	Promotion gates, agent rules, human review

This mapping allows Civitas AI outputs to be reused directly in AI governance documentation without rework.

---

28. Auto-Generated Governance Artifacts

This section demonstrates how governance artifacts can be mechanically generated from this PRD, eliminating manual translation between product, security, and compliance.

28.1 Control Registry (YAML)

The control registry is derived from the Control ID Taxonomy and backlog.

```
controls:
  - id: MOD-001
    name: Automated Text Classification
    type: automated
    description: Classifies submitted text using an approved moderation model
    evidence: AUD-001

  - id: POL-001
    name: Threshold-Based Moderation Policy
    type: policy
    description: Maps model scores to enforcement actions
    evidence: AUD-001

  - id: GOV-002
    name: Human-in-the-Loop Review
    type: compensating
    description: Requires moderator approval for escalated content
    evidence: AUD-002
```

```
  - id: AUD-001
    name: Immutable Evidence Storage
    type: audit
    description: Stores append-only moderation evidence
```

This registry can be loaded directly by agents, CI pipelines, or policy engines.

28.2 Rego Policy Suite (OPA)

The following Rego policies represent a **production-grade baseline** derived directly from Civitas AI acceptance criteria, agent rules, and promotion gates. These policies are intended to run inside OPA as part of CI/CD, runtime enforcement, or both.

28.2.1 Promotion Authorization Policy

```
package civitas.promotion

default allow := false

# Allow promotion only when all controls and evidence are satisfied
allow {
  valid_policy
  valid_controls
  evidence_complete
  no_open_overrides
  environment_transition_allowed
}

valid_policy {
  input.policy.status == "published"
  input.policy.id == input.decision.policy_id
  input.policy.version == input.decision.policy_version
}

valid_controls {
  startswith(input.decision.control_id, "MOD-")
}

evidence_complete {
  input.evidence.immutable == true
  input.evidence.control_id == input.decision.control_id
  input.evidence.policy_id == input.decision.policy_id
}

no_open_overrides {
```

```
  not input.overrides[_].status == "open"
}

environment_transition_allowed {
  input.env_from == "staging"
  input.env_to == "prod"
}
```

28.2.2 Human-in-the-Loop Enforcement Policy

```
package civitas.human

default allow := false

# Escalated or overridden decisions require human approval
allow {
  input.action != "escalate"
}

allow {
  input.action == "escalate"
  input.human_approved == true
}
```

28.2.3 Evidence Integrity Policy

```
package civitas.evidence

default valid := false

valid {
  input.evidence.immutable == true
  input.evidence.timestamp != ""
  input.evidence.control_id != ""
  input.evidence.policy_id != ""
}
```

28.2.4 Deny Reasons (Developer Feedback)

```
package civitas.deny

deny[msg] {
  input.evidence.immutable == false
  msg := "Evidence must be immutable"
```

```
}

deny[msg] {
  input.policy.status != "published"
  msg := "Policy must be published before enforcement"
}

deny[msg] {
  input.action == "escalate"
  not input.human_approved
  msg := "Escalation requires explicit human approval"
}
```

rego package civitas.moderation

Block promotion if required evidence is missing

deny[msg] { input.control_id == "MOD-001" not input.evidence.generated msg := "MOD-001 failed: missing immutable evidence" }

Require human approval for escalations

deny[msg] { input.action == "escalate" not input.human_approved msg := "Escalated content requires human approval" }

````
This policy is derived directly from acceptance criteria and agent rules.

---

### 28.3 Agent Configuration (YAML)

Agents are configured declaratively using rules sourced from the PRD.

```yaml
agents:
  validation-agent:
    responsibilities:
      - verify_control_ids
      - verify_policy_versions
      - verify_evidence_presence

  policy-compliance-agent:
    responsibilities:
      - enforce_rego_policies
````
````

```
      - block_promotion_on_violation

  evidence-agent:
    responsibilities:
      - validate_immutability
      - verify_audit_fields
```

This configuration allows agents to operate consistently across environments.

28.4 CI/CD Promotion Gate Example

```
promotion_gates:
  staging:
    require:
      - validation-agent: pass
      - policy-compliance-agent: pass

  production:
    require:
      - validation-agent: pass
      - policy-compliance-agent: pass
      - evidence-agent: pass
      - human_approval_if_escalated: true
```

This ensures that governance is enforced automatically before deployment.

---

29. Final Executive Closure

Civitas AI is specified as both a product and a system of governance. This document enables automated generation of controls, policies, agent behavior, and audit artifacts from a single source of truth.

As a result, governance is no longer a parallel effort or afterthought—it is an emergent property of the system itself. This approach reduces risk, accelerates delivery, and provides defensible assurance of responsible AI operation at scale.