PIRANHA CMS

André Oliveira 107637
Alexandre Cotorobai 107849
Hugo Correia 108215
Joaquim Rosa 109089

Scenario 2

Table of contents

01.
System Analysis & Project Vision

Current State Analysis

- Piranha CMS is a lightweight, flexible CMS built on ASP.NET Core
- Strong in simplicity and efficiency for basic content management
- Supports traditional CMS and headless implementations

Current limitations:

- Basic content states (New, Draft, Unpublished, Published)
- No multi-stage approval workflows
- Lacks collaborative editing features
- Basic permission model without state-specific permissions

Strategic Goals & Quality Attributes

Vision: Transform Piranha CMS into an enterprise-grade editorial platform orchestrating complex content approval workflows while maintaining performance excellence.

Strategic Goals:

- Advanced Multi-stage Content States and Intelligent Permissions
- Seamless, Context-aware Content Handoffs
- Enterprise-grade Scalability and Performance

02. Architectural Design Methodology

Architectural Design Methodology

Selection of Attribute-Driven Design (ADD)

Why ADD was chosen:

- Explicitly prioritizes quality attributes from the outset
- Provides iterative approach for progressive evolution
- Integrates stakeholders throughout the design process
- Preserves integrity of existing architectural layers

Comparison with alternatives:

- Architecture-Centric Design Method (ACDM) - Less explicit focus on quality attributes
- TOGAF ADM - Excessive formality and documentation for our medium-scale project

Implementation Process:

1. Reviewing inputs
2. Iteration goal & Input Selection
3. Choosing Elements to Refine
4. Choosing Design Concepts
5. Instantiate Architectural Elements
6. Sketching Views & Recording Design Decisions
7. Performing Analysis & Review

03. Domain-Driven Design Approach

Strategic Domain Analysis

Core Domain:

- Editorial Workflow

Supporting Domains:

- Content Management
- User & Permissions
- Audit & History

Generic Domain:

- Notification

Domain Models

Core Domain:

Editorial Workflow: most significant business value and competitive differentiation

- Workflow definition and configuration;
- State management and transitions;
- Approval processes and validation rules.

Domain Models

Supporting Domain:

Content Management (existing Module)

- Added extension that links the Content to the Editorial Workflow.

User & Permissions (existing Module)

Audit & History

- Tracking content state changes;
- Enables compliance and traceability.

Domain Models

Generic Domain:

Notification

- Manages communication about workflow events

04.

Cross-Cutting Concerns

Cross-Cutting Concerns

Logging and Error Handling

- Standardized log formats and structured errors
- Centralized aggregation for traceability
- Uniform capture of failures and state changes

Security

- Role-based access control (RBAC) across services
- Centralized authentication and authorization policies

Performance

- Shared performance metrics across modules

Scalability

- Stateless processing and asynchronous communication
- Horizontal scalability at service boundaries

Observability

- Unified collection of metrics, logs, and traces
- Integration with Prometheus, Jaeger, and Grafana
- End-to-end visibility for monitoring and root cause analysis

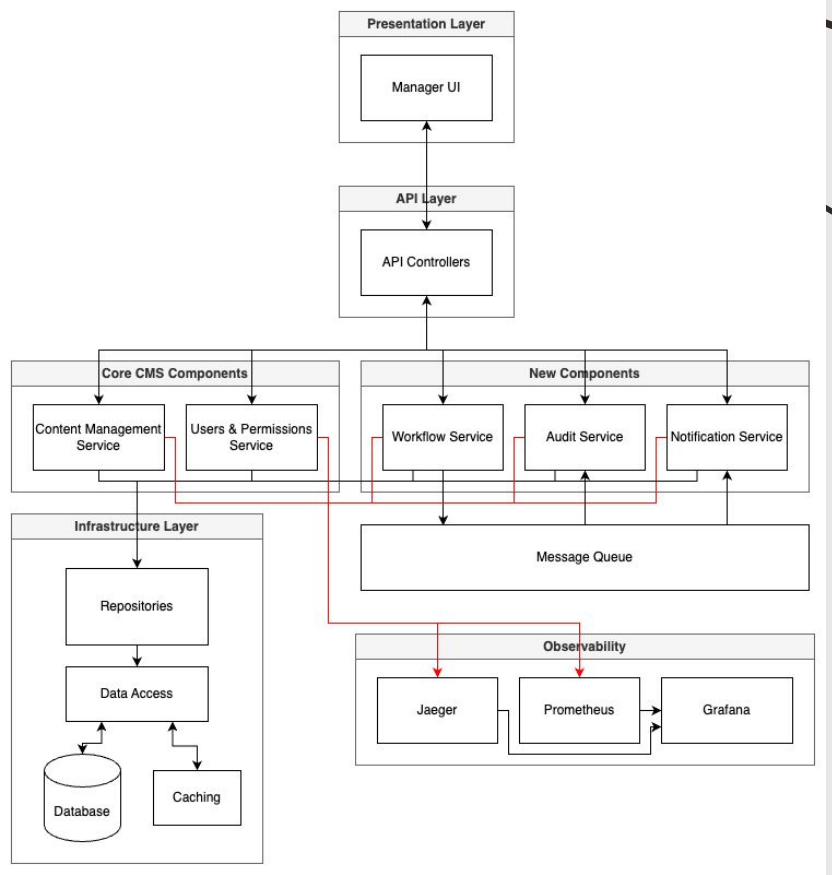
05.

Architecture

Architecture

High-Level Architecture

- **Presentation Layer:** Manager UI
- **API Layer:** RESTful endpoints
- **Application Services Layer:** Core CMS Components and New Components
- **Infrastructure Layer:** Data access, caching, repositories
- **Message Queue**
- **Observability**



06.

Goals Achieved

Goals Achieved

Editorial Workflow

- Workflow definition and configuration; ✔️
- State management and transitions rules (with User & Permissions Module); ✔️
- Approval processes; ✔️

Audit & History

- Tracking content state changes; ✔️
- Enables compliance and traceability. ✔️

Notification

- Manages communication about workflow events ✔️

DEMO

Software Architecture 2025

Lista DEMO

Pré:

- Criar 2 user: Editor, Reviewer
- Criar 1 workflow
  - States: Draft, Review, Ready, Published
  - Transitions:
    - Draft->Review (Editor),
    - Review->Ready (Reviewer),
    - Ready->Published (Admin),
    - Review->Draft (Reviewer, Admin),
    - Ready -> Draft (Reviewer, Admin)
    - Published->Draft (Admin).
- 1 Janela no privado (Editor)

Pós:

- Editor cria uma página
- Associa o workflow
- Mostra a Instance
- Transição para Review

- Reviewer abre a previsualização
- Transição para Ready
- Logout de Reviewer

- Admin transita para Published (verifica se está published)
- Admin vê algo mal e dá reject
- Editor mostra Audit History
- Mostrar notifications
- Mostrar criar workflow