■ ISMS Accelerator for Confluence — Technical Documentation

1. Purpose

The ISMS Accelerator is a Confluence Cloud app, built with Atlassian Forge, designed to automatically scaffold a complete ISO/IEC 27001:2022 Information Security Management System (ISMS) workspace. Instead of starting from scratch, organizations instantly get root pages aligned with Clauses 4–10, Annex A control pages with structured templates, and guidance from ISO/IEC 27002:2022.

2. Core Features

Root Pages

Each core ISMS section is generated as a Confluence page with Purpose, Guidance, To Complete sections with examples, and Evidence prompts. Pages include: Scope, Responsibilities, Risk Management, SoA, Operations, Documentation, Evidence & Audits.

Annex A Controls

One page per control (A.5.1 – A.8.93), each including Control Summary, Expectations & Guidance, Business completion fields (owners, procedures, metrics), Risk & Gap tracking, and Evidence.

Other Features

Ultra-batched creation (1 page per call), professional UI with space key input and options, status updates, and Marketplace-friendly (no external fetch).

3. Architecture

Technology

• Atlassian Forge platform • Custom UI (Vite) for frontend • Forge Resolver backend with Node.js 20 • Confluence v2 REST API with route`` and api.asUser()

Components

1. manifest.yml – declares modules and permissions 2. src/index.js – resolver functions, helpers, templates 3. ui/index.html – input form and layout 4. ui/main.js – frontend logic with @forge/bridge 5. vite.config.js – build pipeline 6. package.json – dependencies and scripts 7. README.md – developer instructions

4. Development Workflow

1. Bootstrap with forge create, then modified for Custom UI. 2. UI developed in HTML/CSS/JS, built with Vite. 3. Backend developed with resolver functions calling Confluence v2 API. 4. Tested with forge tunnel and installed with forge deploy/install. 5. UI clipping solved with padding and sentinel spacer.

5. Design Decisions & Lessons Learned

• v2 API chosen to replace deprecated v1. • asUser() required to avoid app-level permission issues. • Ultra-batching avoids Forge timeout. • Custom UI replaces deprecated UI Kit 1. • Sentinel spacer avoids iframe clipping.

6. Deployment Checklist

1. manifest.yml – correct scopes 2. forge lint – validate 3. npm run build:ui 4. forge deploy 5. forge install 6. Test in Confluence

7. Marketplace Considerations

• Restrict scopes for production. • App only stores minimal IDs in Forge storage. • No external data flows. • Documentation required: privacy, security, SLA.

8. Blueprint for Future Frameworks

Same architecture can be reused for NIST CSF, HITRUST, SOC 2, PCI DSS. Replace templates for new frameworks while keeping UI + batching logic. Root pages = framework domains; Control pages = criteria. UI and backend stay largely unchanged.

9. Example: Adapting to SOC 2

Root pages map to Trust Services Criteria (Security, Availability, Confidentiality, Processing Integrity, Privacy). Annex-like pages created for each criterion. Same structure: Purpose, Guidance, Business Sections, Evidence.