Functional Specification: CumulusPro Digital-Gemeinde.ch

Table of Contents

[Version 3](#_Toc201920099)

[1. Introduction 4](#_Toc201920100)

[1.1 Purpose 4](#_Toc201920101)

[1.2 Scope 4](#_Toc201920102)

[1.3 Intended Audience 4](#_Toc201920103)

[2. General Conditions (Rahmenbedingungen) 5](#_Toc201920104)

[RBG-001: Legal Jurisdiction 5](#_Toc201920105)

[RBG-002: Language Requirements 5](#_Toc201920106)

[RBG-003: Hosting Requirements 5](#_Toc201920107)

[RBG-004: Change Management 5](#_Toc201920108)

[2. Functional Requirements 6](#_Toc201920109)

[2.1 Form Structure and Creation 6](#_Toc201920110)

[FR-001: Multi-page and Section-based Forms 6](#_Toc201920111)

[FR-002: Form Templates 6](#_Toc201920112)

[FR-003: Draft and Published States 6](#_Toc201920113)

[2.2 Supported Field Types 6](#_Toc201920114)

[FR-004: Basic Input Fields 6](#_Toc201920115)

[FR-005: Special Fields 7](#_Toc201920116)

[FR-006: Field Customization and Attributes 7](#_Toc201920117)

[2.3 Field Validation and Logic 7](#_Toc201920118)

[FR-007: Required Fields 7](#_Toc201920119)

[FR-008: Regex Validation 7](#_Toc201920120)

[FR-009: Conditional Logic 8](#_Toc201920121)

[2.4 Form Behaviour and Actions 8](#_Toc201920122)

[FR-010: Submission Handling 8](#_Toc201920123)

[FR-011: Redirect or Confirmation Message 8](#_Toc201920124)

[2.5 Workflow and Integration 8](#_Toc201920125)

[FR-013: Workflow Integration 8](#_Toc201920126)

[FR-014: External APIs 8](#_Toc201920127)

[FR-015: Payment Gateway 9](#_Toc201920128)

[2.6 User and Tenant Management 9](#_Toc201920129)

[FR-015: Role-Based Access Control 9](#_Toc201920130)

[FR-016: Tenant Customization 9](#_Toc201920131)

[3. Non-Functional Requirements 11](#_Toc201920132)

[NFR-001: Performance 11](#_Toc201920133)

[NFR-002: Availability 11](#_Toc201920134)

[NFR-003: Security 11](#_Toc201920135)

[NFR-004: Scalability 11](#_Toc201920136)

[NFR-005: Auditability 11](#_Toc201920137)

[4. Standards 12](#_Toc201920138)

# Version

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Description | Author |
| 1.0 | 25/06/2025 | Initial Version | Roger Vial |
|  |  |  |  |

# 1. Introduction

## 1.1 Purpose

This document constitutes the functional requirements specification for the CumulusPro Digital-Gemeinde.ch platform.

## 1.2 Scope

The specification covers all core components of the Digital-Gemeinde.ch system, including:

- Form creation and configuration

- Field types and validation

- Dynamic logic and conditions

- Submission processing and export

- Multi-tenant support and branding

- Workflow integration

- Role-based access and user management

## 1.3 Intended Audience

This document is intended for software developers, QA engineers, architects, DevOps teams, and stakeholders involved in the development, maintenance, and integration of the Digital-Gemeinde.ch solution.

# 2. General Conditions (Rahmenbedingungen)

### RBG-001: Legal Jurisdiction

The solution shall comply with the legal framework of the Swiss Confederation. All contractual and operational matters are subject to Swiss law.

### RBG-002: Language Requirements

The user interface shall be available in German and English. Administrative and technical documentation should also be available in English.

### RBG-003: Hosting Requirements

The platform shall be hosted in data centers located within Switzerland or the European Union if they are hosted by CumulusPro. The platform can also be hosted in local (on-premise) Datacenters.

### RBG-004: Change Management

Any changes to core functionality must be documented, communicated in advance, and subject to approval by the responsible authority.

# 2. Functional Requirements

## 2.1 Form Structure and Creation

### FR-001: Multi-page and Section-based Forms

The system shall allow users to create forms composed of one or more pages. Users can add, remove, rename, and reorder sections.

### FR-002: Form Templates

The system shall support the creation of form templates which can be reused across multiple forms. Templates can be edited and versioned.

### FR-003: Draft and Published States

Forms can exist in a draft or published state. Only published forms are available for end-user submission. Editing a published form creates a new version.

## 2.2 Supported Field Types

### FR-004: Basic Input Fields

The following input types shall be supported:

- Text (single-line)

- Textarea (multi-line)

- Number (with min/max/step)

- Email (validated format)

- Date (calendar picker)

- Time (HH:mm format)

- Dropdown (single selection)

- Checkbox (boolean or multi-choice)

- Radio buttons (single-choice)

- File Upload (PDF, images, DOCX, etc.)

### FR-005: Special Fields

The following domain-specific fields shall be supported:

- Municipality selector (Gemeinde): Populates options based on canton

- Separator and title elements: Non-input visual structure

- Read-only text blocks: For help and description

### FR-006: Field Customization and Attributes

Each field (where applicable) shall support:

- Label, placeholder, tooltip

- Mandatory flag

- Default value (static or dynamic)

- Custom validation rules (regex)

- Conditional visibility based on rules

- Error message customization

## 2.3 Field Validation and Logic

### FR-007: Mandatory Fields

Form designers shall be able to mark fields as required. Required fields must be completed before submission.

### FR-008: Regex Validation

Fields shall support pattern-based validation using regular expressions. Designers can define patterns and associated error messages. Example:

- Swiss phone: `^\+41\d{9}$`

- Postal code: `^\d{4}$`

### FR-009: Conditional Logic

Fields and sections can be conditionally shown or hidden based on values of other fields.

## 2.4 Form Behaviour and Actions

### FR-010: Submission Handling

Upon submission, the form data shall be:

- Persisted in the database

- Associated with user and tenant metadata

- Trigger optional email notifications

### FR-011: Redirect or Confirmation Message

Post-submission actions shall support:

- Redirecting to a custom URL

- Showing a configurable confirmation message

## 2.5 Workflow and Integration

### FR-013: Workflow Integration

Forms may be tied to workflows in the Straatos platform. Upon submission:

- Tasks are created in defined workflows

- Approvers can view submission data

- Form actions (e.g. approval) are recorded in audit trail

### FR-014: External APIs

The platform shall support API endpoints for:

- Retrieving form definitions

- Submitting form responses

- Exporting submission data

### FR-015: Payment Gateway

The platform shall support payment options of the form using Payrexx with the option to integrate further payment gateways in the future.

## 2.6 User and Tenant Management

### FR-015: Role-Based Access Control

Users can be assigned to roles:

- Administrator: Full access to tenant configuration and forms

- Designer: Can create and publish forms

- Viewer/Processor: Can only see submitted form data

### FR-016: Tenant Customization

Each tenant may configure branding elements such as Logo, colors of UI elements, header, footer and fonts.

# 3. Non-Functional Requirements

### NFR-001: Performance

Form rendering shall occur in under 4 seconds for 95% of cases.

### NFR-002: Availability

The system shall be available 99.9% of the time excluding planned maintenance if the appropriate infrastructure is provided. As such, the system shall support a High Availability infrastructure.

### NFR-003: Security

All data in transit must use TLS 1.2+. Data at rest is encrypted. Role-based access and tenant isolation are enforced.

### NFR-004: Scalability

The platform must support scaling to handle increased users and data volume without degradation in performance. The platform shall support up to 1,000 form submissions per hour with an average of 2 attachments with an average file size of 200 KB.

### NFR-005: Auditability

All user interactions, configuration changes, and submissions shall be logged with timestamp and user context.

# 4. Standards

**STD-001: Security Standards**The system shall follow OWASP Top 10 guidelines and encrypt data in transit using TLS 1.2 or higher.

**STD-002: Software Development Standards**The platform shall be developed including automated tests, and enforce code quality via reviews and static analysis.

**STD-003: Interoperability and Integration Standards**RESTful APIs using JSON shall be provided. All endpoints shall be documented using OpenAPI 3.0. UTF-8 encoding must be used throughout.

**STD-004: Usability and UI Standards**The application shall use responsive design principles, support consistent UI frameworks, and provide language options in English and German as well as the possibility to add further languages.

**STD-005: Hosting and Infrastructure Standards**If the platform is hosted by CumulusPro, the underlaying datacentre hosting provider must be ISO/IEC 27001 compliant and ensure all data resides within Switzerland or EU jurisdictions.

If the platform is hosted on premise, the customers are responsible for the infrastructure but must be able to host the application within their own datacentres.

**STD-006: Documentation Standards**  
Documentation shall be written in a common format such as Microsoft Word, markdown or HTML5, and available in English.