# **Operations Documentation – GemeindeConnect**

# **Version**

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

## **1. Introduction**

This document describes the technical and organizational aspects necessary for the secure, stable, and maintainable operation of the Form Designer platform. It is intended for administrators, IT operations personnel, and support teams.

## **2. System Overview**

**System Name:** Gemeinde Connect – Form Designer  
**Purpose:** A web-based platform for creating, managing, and executing digital forms.  
**Components:**

* FormDesigner Frontend (Angular SPA)
* FormDesigner Backend (.NET Web API)
* SQL Database
* Azure Blob Storage or Local Network storage
* OAuth 2 / OpenID Connect Identity Provider)
* Mailserver

## **3. Operating Environment**

**3.1 Infrastructure**

* **Web Hosting:** IIS on Windows Server or Azure App Service (alternatively via Docker/IIS container)
* **Database:** Azure SQL (Elastic Pool, max. 250 GB) or local SQL Server 2019 or 2022
* **File Storage:** Azure Blob Storage or Local network storage
* **Identity Provider:** Azure Entra or other OAuth2 / OpenID Connect Identity Provider
* **Email Service:** SendGrid or local SMTP Mailserver
* **Monitoring:** Azure Application Insights

**3.2 Network Configuration**

* HTTPS only (TLS 1.2+)
* Inbound traffic controlled via API Gateway or Load Balancer
* Database access restricted to internal services (non-public)

## **4. System Components in Operation**

**4.1 Backend (.NET Web API)**

* REST-based endpoints
* Authentication using JWT tokens
* Access to Azure SQL using managed identity or connection string
* Form data stored as structured JSON + metadata

**4.2 Frontend (Angular)**

* Static Single Page Application served via web server or CDN
* Configuration via environment.ts specifying API base URL

**4.3 Database**

* Name: formdesigner-cpro-ch
* Key Tables:
  + FormDesigns, FormDatas, FormTemplates, Users, etc.as per Database Create script in repo.
* Versioning and history stored in FormDesignsHistory, DesignerHistory

## **5. Operational Procedures**

**5.1 Deployment**

* CI/CD pipeline via Azure DevOps
* Automated deployment of frontend and backend
* Database migrations via EF Core Migrations
* Backup: automated daily via Azure SQL

**5.2 Monitoring & Logging**

* Application telemetry via Azure Application Insights or local application logs
* Error tracking and performance metrics
* Health checks exposed via dedicated API endpoints (e.g., /health)

**5.3 Maintenance**

* Regular patching of system components
* Log rotation via Azure Log Analytics
* Automated index maintenance via Azure SQL configuration

## **6. Security & Data Protection**

**6.1 Authentication & Authorization**

* Using OAuth2 & OpenID Connect with Azure Entra or other Identity Provider
* Role-based access control implemented in the backend

**6.2 Data Privacy**

* Form data stored in Azure Blob Storage
* Access restricted via SAS tokens or private containers
* Data encryption:
  + At Rest: Transparent Data Encryption (TDE) for SQL
  + In Transit: TLS 1.2+

## **7. Backup & Recovery**

**7.1 Database**

* Azure SQL Point-in-Time Restore (PITR) or local SQL with custom backup.
* Retention period: 7–35 days or depending on customers requirements

**7.2 Form Data (Blob Storage)**

* Snapshot-based recovery available on request
* If local storage is used, then Backup based on customers backup policy

**7.3 Disaster Recovery**

* Documented disaster recovery plan in place
* Target Recovery Time Objective (RTO): < 2 hours