## Business Requirements

**Purpose and Objectives**

* **Primary Goal:** Data downloading with geospatial and non-geospatial filtering without the need to download single or multiple large files
* **Target Users:** Internal teams / Public users
* **Expected Usage Volume:** Daily requests with possible concurrent users
* **Business Value:** Facilitate data access, remove technical complexity to handle large files

**Budget and Resource Constraints**

* **Project Budget**: Zero budget allocated - no additional funding available from requester
* **Implementation Cost**: All development, licensing, infrastructure, and deployment costs must be absorbed by the implementing department within existing operational budgets
* **Resource Allocation**: Implementation must utilize existing IT departmental staff and resources

## High-Level System Components

**Data Storage Layer**

* PostGIS-enabled PostgreSQL database as the primary spatial data store
* PostGIS tables for large countries are partitioned (compatibility with AGIS Enterprise to be checked)
* Handles geometric/geographic data types, spatial indexing, and complex spatial queries
* Stores both spatial geometries and associated attribute data with full support for non-spatial attributes

**API Gateway Layer**

* RESTful API endpoints for external clients
* Database-first, language-agnostic, no preference amongst Python, Node.js, Java, etc..
* Handles authentication, rate limiting, and request routing
* Provides standardized interfaces for both spatial and non-spatial data operations

**Business Logic Layer**

* Spatial processing and analysis services
* Non-spatial data filtering and search capabilities
* Data validation and transformation
* Query optimization and result formatting
* Multi-format output generation (CSV, GeoJSON, GeoParquet, GeoPackage)
* **No need to provide OGC Standard** formats (e.g. WMS, WFS, etc..) **but would be a plus**

**Caching Layer**

* In-memory caching for frequently accessed datasets or larger countries (spatial and non-spatial)
* Pre-computed analysis results (e.g. country stats, NUTS level stats, etc..)

## Core Functional Areas

### Data Management

* **No need of CRUD operations** (read, update, etc..)
* **No need for bulk data import** capabilities data are update unregularly with an annual frequency

### Query Engine

* **Spatial Queries**: Proximity searches, geometric operations, spatial aggregations
* **Non-Spatial Queries**: Attribute-based filtering, text search, numerical ranges, date filtering
* **Combined Queries**: Spatial and attribute filters working together

### API Services

* Feature retrieval with flexible filtering options
* Multi-format response handling non spatial (csv, json) and spatial (geojson) formats
* Real-time analysis endpoints

## Query Capabilities

### Non-Spatial Filtering Options

* Attribute value matching (exact, partial, regex)
* Numerical range queries
* Date/time range filtering
* Text search and full-text search

### Spatial Filtering Options

* **Unified Data Access:** the API will provides seamless access to data distributed across multiple tables and sources, presenting a single, coherent interface to users without exposing underlying data architecture complexity

### Combined Query Types

* Spatial boundaries + attribute filters (e.g. Building in NUTS FR72 with height > 10 meters)
* Proximity search + property matching (e.g. buildings in Rome with area greater than 500 square meters)
* Time-based + location-based filtering (e.g. buildings built before 1995 in Berlin)

## Output Format Support

### CSV Format

* Flattened attribute data
* Coordinate fields for point geometries
* Header customization

### JSON Format

* Structured attribute data
* Nested object support
* No geometry representation
* Metadata inclusion

### GeoJSON Format

* Full spatial geometry preservation
* Feature collections
* Standards-compliant output (EPSG:4326 re-projection)

### GeoParquet and GeoPackage

* Full spatial geometry preservation
* Feature collections
* Custom Projection (EPSG:4326 and EPSG:3035 Minimum)

## Integration Points

### Client Applications

* **No Web mapping** applications (**NO NEED** for a Dashboard/Geoportal)
* Desktop GIS
* Spreadsheet software, etc…

### Supporting Services

* No Authentication and authorization systems, data are freely available
* Monitoring and logging infrastructure

### API Endpoint Examples

It would be preferable to use custom endpoints avoiding the ArcGIS REST API Naming Conventions.

**Query Endpoints to be adopted:**

* /api/buildings?limit=100&format=csv [ Non-spatial filter, CSV output ]
* /api/countries/malta/stats &format=csv [ Non-spatial filter, CSV output ]
* /api/ buildings /buffer?building\_id=1231l&distance=5km&format=geoparquet [Combined filter ]

**Instead of:**

* Predefined structure: /rest/services/{serviceName}/{serviceType}/{operation}

With service types limited to MapServer, FeatureServer, GPServer, ImageServer and operation names mostly predefined (query, identify, export, etc.)

**If possible, additional ArcGIS limitations should be avoided:**

* Layer numbers instead of descriptive names in URLs
  + /rest/services/MyData/FeatureServer/**0**/query
* Required authentication tokens in URLs or headers

## Scalability Considerations

### Performance Optimization

* Indexing strategies for both spatial and non-spatial columns
* Asynchronous processing for large exports (e.g. entire NUTS 1/2/3 regions)

### Data Organization

* Partitioning strategies for large datasets (Non-spatial partitioning)

## Data are updated annually or even multi-year frequency

**Summary of endpoints (Up-to-date but not exhaustive) to be replicated in ArcGIS Enterprise**

| **Method** | **Path** | **Function** |
| --- | --- | --- |
| GET | /buildings/within/10?lat=16&lon=41&limit=100 | Find all buildings within a certain distance from a pair of coordinates |
| GET | /buildings/nearest/{lat}/{lon} | Find nearest building the a pair of coordinates |
| GET | /buildings/buffer/{building\_id}/{distance}?lat=16&lon=41&limit=100 | Find all buildings within a certain distance from a certain building |
| GET | /buildings/distance/{building\_id1}/{building\_id2} | Distance between building one and building two |
| GET | /buildings/contains/{lat}/{lon} | Find the building containing a pair of coordinates |
| GET | /buildings/size-distribution/{country} | Fined the distribution of areas of buildings divide in ten classes |
| GET | /buildings/largest/{country}/{limit} | Finde largest buildings in a country limited to (default 10) |
| GET | /buildings/export/{format} | Export buildings fetched from a query in a specific format (csv, geoparquet,etc..) |
| GET | /buildings/compare/countries | Compare distribution of data between two countries |
| GET | /countries | The list of countries contained in the database with metadata on accuracy, completeness, statistics,etc.. |
| GET | /health | Helth of the api server |
| GET | / | Home message |