D6.3 Web-service performance report

Deliverable for the Horizon Europe Project BirdWatch

Version 1.1

Funded by the European Union

Legal Disclaimer

Funded by
the European Union

Document Information

GA Number	101082634	**Type of Action**	Horizon-IA
Full Title	BirdWatch - a Copernicus-based service for the improvement of habitat suitability of farmland birds via satellite-enabled monitoring, evaluation and optimisation of CAP greening measures		
Project Acronym	BirdWatch		
Start Date	February 1st, 2023	**Duration**	36 Months
Project URL	https://birdwatch-europe.org/		
Deliverable	D6.3 Web-service performance Report		
Work Package	WP6		
Project Month of Delivery	**Contractual** M28	**Actual**	M33
Nature	OTHER	**Dissemination Level**	PU
Lead Beneficiary	LUP		
Responsible Author	Nastasja Scholz, LUP		
Contributions from	Ursula Plötz, LUP Laura Panner, LUP		

Funded by the European Union

History of Changes

Version	Issue Date	Stage	Description	Comments	Contributor
1.0	01.10.2025	First version	First draft		Laura Panner Ursula Plötz
1.1	14.10.2025	Second version		In response to our reviewer's comments, a conclusion section was added to the deliverable	Laura Panner

Funded by the European Union

Funded by the European Union

Funded by
the European Union

Introduction

BirdWatch's aim is to provide an EU-wide service supporting the monitoring and improvement of farmland habitat suitability for bird species which breed or forage on agricultural land.

The BirdWatch service will consist of a geospatial data-based monitoring service which evaluates the habitat suitability of farmland parcels for specific bird species as well as of an optimisation workflow, serving as a decision-support for the identification of appropriate policy-supported agri-environmental measures.

The service relies on geospatial data which partly consists of environmental parameters derived from the Sentinel-1 and Sentinel-2 of the Copernicus Program of the European Space Agency (ESA) and auxiliary spatial datasets, most importantly the national or regional Land Parcel Identification System (LPIS) data.

The service will be made available to users via a web-based GIS application.

The heart of this GIS application is a backend database where all necessary data storage and management takes place. This includes the data components necessary to calculate the state of habitat suitability and its temporal evolution as well as the various constraints and the associated optimised habitat suitability, provided by VITO's MooV service. The database scheme is described in Deliverable D6.1.

Built on top, there is a web-based user interface which accesses this database for providing the results to the users. The first iteration of the web-based user interface was described in Deliverable D6.2.

Since the submission of D6.2, the BirdWatch platform has progressed and interface functionalities have been extended. These functionalities, together with the output of the monitoring and optimisation service are going to be tested together with various stakeholders in our four test regions.

In preparation for the test activities, the platform performance was tested and adjustments made, where necessary.

This deliverable reflects the current stage of platform performance and the changes which have been made since the submission of D6.1 and D6.2.

Funded by
the European Union

1 Performance Analysis GeoServer

1.1 Migration from WMS to WMTS

Summary

The WMS (Web Map Service) response times for multiple vector layers often exceeded 60 seconds at peak and repeatedly led to timeouts. The main causes are computationally expensive database queries on large tables. Switching to WMTS (Web Map Tile Service) layers shifts the rendering workload to pre-rendering/seeding and enables massive caching. As a result, query times now typically drop from multiple seconds to the sub-second range, error rates decrease, and scalability under peak loads improves significantly, since images only need to be delivered instead of being rendered anew each time.

Implementation Aspect

- **Services:** GeoServer (WMS), GeoWebCache available; vector layers served directly from large database tables.
- **Problem Description:** Long rendering times in WMS (partly > 60 s) leading to timeouts; unreliable user experience in detailed views.
- **Goal:** Stable, fast delivery of map rendering by migrating to WMTS.

Technical Context: WMS vs. WMTS

- **WMS (on-the-fly):** For each request, the map is fully generated (DB query, styling, rendering). Requests (sometimes > 60 s) up to timeout; unreliable user experience in detailed views.
- **WMTS (tiled):** Fixed tile pyramid (TileMatrixSet). Identical requests → high cache hit rate. Rendering can be done in advance (seeding) or on-demand; subsequent reuse from cache.
- **Effect:** CPU/DB intensity per user request is replaced by a network request for cached files; load peaks are significantly mitigated.

Effects of the Migration

- **Latency:** TTFB (Time To First Byte) typically < 1 s (from cache) instead of > 60 s (WMS rendering).
- **Throughput & Load Behavior:** More concurrent users without degradation (noticeable performance drop of the system under load).
- **Error Rate:** Significantly fewer timeouts/500 errors, since complex renderings are performed in advance.
- **Cost/Benefit:** Server CPU and database load decrease; bandwidth increases moderately (more, but smaller files).

Funded by
the European Union

Risks & Limitations

- **Static representation per style:** Ad-hoc filters and dynamic SLD (Styled Layer Descriptor) parameters are only partially supported in WMTS (each variant requires its own cache keys/tile sets).
- **Seeding effort at high zoom levels:** The number of tiles increases significantly.
- **Data currency:** Data changes only become visible after seeding → seeding must always be executed after data updates.
- **FeatureInfo/Identify limited in WMTS:** WMTS GetFeatureInfo is optional/inconsistent and in some cases only available via WMS proxy (no caching benefit), or not at all — a direct click on a parcel therefore cannot be reliably captured/processed. A workaround is required for this (an invisible WMS layer delivering the features).

Results

- **Perceptible acceleration:** Map tiles consistently appear “immediately”; no timeouts.
- **Stability under load:** No rendering queues, low error rate in logs.
- **Consistency:** No or only minimally visible tile artifacts; styles match the WMS reference.

Note on Data Basis

This decision template is based on repeatedly observed behaviour (WMS generation partly > 60 s with timeout) and established practice for performance improvement through tile caches. Measurement data are not required but can later improve fine-tuning (e.g., seeding limits, tile size).

Funded by
the European Union

1.2 Impact of Zoom Levels on Map Layers with Large Database Tables

Summary

The map's performance noticeably decreases when zooming out—especially for layers such as Birdwatch that are rendered directly from very large database tables. At low zoom levels, WMTS pre-generation (seeding) sometimes fails, as both the number of tiles to be generated and the geometry complexity per tile exceed the available resources. This results in long loading times and even timeouts. Against this background, we deliberately limited the zoom levels and focused seeding spatially on the project area of Europe. In doing so, we reduce rendering load and storage requirements, stabilize response times, and at the same time ensure the required level of detail in the relevant area.

Implementation Aspect

- **Data Source:** PostGIS database; tables with a high number of objects and partly complex geometry.
- **Services:** GeoServer WMS/WMTS (GeoWebCache) for an interactive map from overview to detail.
- **Observation:** At low zoom levels, WMTS tiles could not be pre-rendered for several data-intensive layers.

Observed Effects

- Noticeably longer loading times at high magnifications (detailed zooms) without affecting the Birdwatch application, since high zoom levels (e.g., outside Europe) are not within the project area.
- Partial failures/error messages during tile requests.
- WMTS seeding fails at large zoom levels or would take excessively long.
- At medium zoom levels, operation is possible, but with a significant increase in latency compared to the overview.

Technical Context

- **Geometry Complexity:** Dense line networks, large polygons, and many vertices per feature increase rendering and placement effort (e.g., labelling) per tile.
- **Database Path:** Queries on very large tables become I/O-intensive; without appropriate spatial indexes, partitioning, and generalization, response times and resource consumption increase significantly.

Funded by the European Union

Risks Without Zoom Level Limits

- Persistently poor user experience at certain zoom levels.
- High infrastructure costs without proportional benefit (inefficient seeding).
- Operational instability under peak loads.

Note on Data Basis

No measurement data were available for this analysis. The assessment is based on established behaviour patterns of GeoServer/GeoWebCache in combination with large database tables and on the observed symptoms (in particular failed WMTS seeding at high zoom levels).

Funded by
the European Union

2 Database Performance Analysis

2.1 Parcel Information Query

Summary

Previously, there were two ways to retrieve parcel information and display it in the status: via an input field and by clicking on the map. The map variant used getFeatureInfoUrl (WMS GetFeatureInfo) and had very long response times of around ~13 seconds. Regional performance differences were also observed.

Implementation Aspect

For Germany and Flanders (0–14), the click query on **geo_parcel_data** averages 10.07 s (median 9.85 s, p95 11.86 s).
For South Tyrol and Lithuania (15–20), we measure an average of 19.91 s (median 19.72 s, p95 22.39 s).
Thus, latency in South Tyrol/Lithuania is about twice as high (median comparison).

Parcel ID	Time (s)
3307290	13.78
3307291	9.39
3338480	9.44
3228003	9.37
3311329	9.13
6060817	9.09
6151939	9.24
3368127	10.24
3682537	9.99
3399281	10.27

Funded by the European Union

1149116	9.70
1461178	10.32
1355292	10.22
682930	10.82
3321094	19.04
3858124	17.23
1258722	18.22
616244	22.10
1035337	22.48
179018	20.40

Technical Context

- **Table SRID:** EPSG:4326; the click point is correctly set using ST_SetSRID(ST_MakePoint(lon, lat), 4326).
- **GiST Index:** Present on bwdata.geo_parcel_data(aoi_shape) — the query is index-supported.

Applied Measure

Implementation of a lightweight API endpoint. The client now sends the lon/lat of the click point on the map, and a direct PostGIS query on *bwdata.geo_parcel_data* (Point-in-Polygon via ST_Intersects) is executed server-side. The results are processed further without WMS overhead.

Result

Response time dropped from ~13 s to under 1 s — (13× faster). Interaction is significantly more responsive, and both network and server overhead were reduced (including in Lithuania and South Tyrol).

Funded by the European Union

2.2 Area Tables & Geometry Queries

Summary

In several parts of the application, geometry comparisons are performed to retrieve the required data or to process data (e.g., generating the GeoJSON for the project's own areas).

Implementation Aspect

- **Data Source:** PostGIS database; tables with a high number of objects and partly complex geometry
- **Observation:** Geometric intersection queries (AOI ↔ Parcel) led to long runtimes and even timeouts without appropriate indexes.

Applied Measure

- Creation of a spatial index (GiST) on aoi_shape (see below).
- **Ensure SRID** consistency column in 4326; inputs transformed to 4326 with ST_Transform if necessary).

Result

The query execution time is approximately 1000× faster with a GiST index compared to without indexing.

```
SELECT parcel_id
FROM bwdata.geo_parcel_data
WHERE ST_Intersects(
  aoi_shape,
  CASE
    WHEN :srid = 4326 THEN ST_GeomFromText(:wkt, :srid)
    ELSE ST_Transform(ST_GeomFromText(:wkt, :srid), 4326)
  END
);
```

Parcel ID	Country	Year	Area	Region
4424856	italy	2022	3,866.65	southtyrol
3542089	italy	2022	30,235.49	southtyrol
555159	lithuania	2022	12,201.91	Jonava District

Funded by the European Union

				Municipality
555471	lithuania	2022	44,908.69	Raseiniai District Municipality
3044986	germany	2022	57,950.00	Bavaria
3044992	germany	2022	12,650.00	Bavaria
1036418	belgium	2018	13,528.02	flanders
384832	belgium	2018	19,305.05	flanders

2.3 Data Aggregation

Summary

To obtain the average habitat suitability per region, all regional data are currently aggregated, and the average is calculated within the SQL query. The table being accessed contains all bird species data for all regions and is therefore correspondingly large.

Observed Effects

- **Long query** load times with extreme cases around **~35s**
- **Backlog of requests**
- **Noticeably poor user experience:** Sluggish interaction, delayed map rendering.

Applied Measure

The aggregations that were previously calculated at runtime (AVG per region and species) are now precomputed per region and stored in a compact aggregate table, which reduces response times from seconds to milliseconds. This aggregate table is implemented as a physical table with aggregated data and indexes and is automatically updated through a function and trigger whenever the underlying base data changes. In this way, the values remain up to date without the overhead of recalculating all aggregates during query execution.

Advantages:

- **Very fast queries (millisecond range)** thanks to small result sets and appropriate indexes
- **Stable latency** independent of dataset size.
- **Predictable load** (computational effort shifted to off-peak jobs).

Funded by the European Union

Disadvantages:

- **Data currency:** After data imports, re-aggregation is required (MV refresh or batch).
- **Complexity:** Invalidation/refresh logic is needed (but can be well automated).

Result

By precomputing the aggregations and storing them in an incremental aggregate table, response times were reduced from several seconds to milliseconds. This leads to a significantly more responsive user experience and ensures stable system performance even under higher load. At the same time, the database is relieved of expensive runtime calculations, allowing resources to be used more efficiently.

Funded by
the European Union

3 Login Performance

Summary

On 10 September 2025, a total of 15 login attempts were recorded: 10 successful logins and 5 deliberately triggered failed attempts (test cases). For the production assessment, only the successful logins are considered; after adjustment, this results in a success rate of 100% (10/10). The duration of successful logins had a median (P50) of about 392 ms, with the P95 around 415 ms. The largest share of the total duration was taken by the authentication phase (checkUser) at around 373 ms (P50); the subsequent database update of *last_login* was negligible at about 18 ms (P50). The five test failures—four of them under artificial network throttling (3G/4G)—reflected mainly authentication latency as expected, and measured around ~237 ms (throttled) or once ~475 ms (without throttling). Overall, the sample confirms a stable and fast login path.

Technical Context

- **Measurement Path:** Click on login → checkUser (auth) → UPDATE last_login → session active/redirect.
- **Bottleneck Share:** The largest share lies in auth/checkUser; the database is not a limiting factor in the login path.
- **Stability:** Narrow distribution (P50 ≈ 392 ms, P95 ≈ 415 ms) indicates a consistent path; outliers are rare.
- **Failed Attempt:** One failed login (~475 ms) raises the overall P95 across all attempts to ~447 ms; causes should be reported separately (e.g., wrong password vs. policy).

Risks & Limitations

- **Sample Size & Time Window:** Daily/weekly fluctuations and IdP peak load are not covered.
- **Network Variability:** Results aggregate different network profiles; without separate evaluation, differences between 3G/4G/no throttling may be hidden.
- **Context Changes:** Modifications to the IdP, policies, or rate limits can temporarily affect login times.

Note on Data Basis

- **Method:** Server-side timing (monotonic clock) with logging per attempt (*login_attempt*).
- **Network Profiles:** Sample includes simulated throttling (3G, 4G) as well as unthrottled connections; the reported metrics aggregate across all three profiles. For higher significance, it is recommended to tag the network profile in the event (e.g., net_profile: '3g' | '4g' | 'unthrottled') and evaluate separately.

Funded by the European Union

4 Layer Performance

Summary

Tile latencies show clear differences by layer: OSM (OpenStreetMap) serves as a stable network baseline (typically ~25–120 ms). **“geo_parcel_hs_species_0”** (Germany majority) usually lies in the low 3-digit range (approx. 80–300 ms) with recurring outliers reaching into the seconds (once ~14 s). **“geo_parcel_hs_lt_species_0”** (Lithuania majority) and **“ortho_lt”** (Orthophoto Lithuania) stand out with frequent second-level values (typically 0.4–1.7 s; repeatedly 2–5 s, with some measurements at 8–15 s). Since OSM remains consistently fast at the same time, this points to cache misses, seeding gaps, or grid mismatches in the Lithuania parcel and orthophoto layers—rather than general network issues.

Technical Context

Measurements were taken client-side via OpenLayers *tileload* events (metric tile_latency_ms). The values represent the loading time of individual tiles, including server rendering time, caching, and network share. Requests are made in EPSG:3035; for WMTS, an exact match of MatrixSet, resolutions, tile size, STYLE, and FORMAT between the client (OpenLayers-WMTSTileGrid) and server-side seeding is critical. Any mismatch results in divergent cache keys and thus a MISS instead of a HIT, even if tiles are “pre-generated.”

The observed fluctuations in the Germany parcel layer (GER) indicate typical seeding gaps. Whether this is also the case for the Lithuania parcel layer (LT) still needs to be verified (seeding status & parameter alignment). The variance in the Lithuania orthophoto layer can also be explained: it is WMS, i.e., rendered on the fly without pre-generated tiles. This layer should be migrated into tile caching (e.g., GeoWebCache as WMTS/XYZ or tiled WMS with metatiling) to achieve consistent latencies and high cache hit rates.

Applied Measure

- **Measurements on the Server (production-near):** Perform client- and server-side measurements directly against the production GeoServer and collect P50/P95 values per layer/zoom/AOI for tile_latency_ms, TTFT, cache hit rate (GWC), error rate, and server metrics (CPU/IO).
- **Check caching configurations of layers:** Compare WMTS layer settings (zoom levels, tile size, etc.) to rule out configuration errors.
- **Expand seeding:** Extend pre-generation (seeding) of tiled layers—especially for Lithuania—in the relevant AOIs and zoom levels to increase the cache hit rate and reduce latency/variance spikes.

Funded by the European Union

- **Migrate orthophoto layer to WMTS/XYZ:** Evaluate whether the current WMS orthophoto layer can be migrated to a tiled service (WMTS or XYZ) with an active tile cache (e.g., GWC) to avoid on-the-fly rendering and significantly reduce latencies through cache hits.
- **Restrict region layers in the frontend (set extent):** Currently, multiple region layers are visible in parallel; OpenLayers loads tiles for each visible layer—even outside the relevant area. By defining a region-appropriate extent (EPSG:3035) per WMTS/XYZ layer and optionally enabling auto-visibility based on the current view, unnecessary requests can be avoided. This reduces server load, bandwidth, and latency without changing functionality.
- **Switch layer SQL views to materialized views:**
  Advantages: Faster seeding, better performance when not everything is cached.
  Disadvantages: MVs must be manually refreshed when data is updated.
- **Enable memory cache:** Tiles are held not only on disk but also in RAM.
  Expected Result: Significantly reduced response times for repeated access (< 50 ms).
- **Proxy optimization:**
  Adjust the PHP proxy to transparently pass through GeoWebCache headers (geowebcache-cache-result, geowebcache-cache-time, etc.).
  Set custom cache-control headers for image/* to enable browser caching (Cache-Control: public, max-age=604800, immutable).
  Remove the GeoServer default headers no-cache/pragma, which block browser caching.
- **Frontend asynchronization:**
  Load WMTS layers in parallel instead of sequentially, so slow layers do not block the map.
  Set updateWhileAnimating: false / updateWhileInteracting: false to avoid unnecessary tile requests and prevent the request queue from filling up.

Several performance optimization measures for the map layers have already been implemented or initiated. The restriction of region layers in the frontend as well as frontend asynchronization have been successfully deployed and are showing positive effects. The switch from SQL views to materialized views has been started on a test basis to evaluate performance gains. Enabling the memory cache and optimizing the proxy are currently under review. This phased implementation ensures that quick wins are realized immediately while more complex adjustments are being validated step by step.

Note

The measurements were carried out on a local system, where the seeding status was lower than on the server. Individual higher latencies may therefore have resulted from on-the-fly generation. No measurements with artificial network throttling (e.g., 3G/4G) were performed. The values should be understood as an initial orientation and highlight optimization potential. On the server, better performance can be expected due to more extensive pre-seeding and caching.

Funded by
the European Union

It should also be noted that no measurements are yet available for the measures currently in progress (e.g., memory cache, proxy optimization, materialized views). The reported times therefore reflect the situation **before** the implementation of these optimizations.

Layer	Messungen	Min (ms)	P50 (ms)	P95 (ms)	Max (ms)	Mittel (ms)
osm	133	16	38	117	207	50
GER Majorität	61	66	207	3083	14430	695
Litauen Majorität	35	54	239	5113	12117	2223
Ortho LT	51	181	1293	4167	15614	2880

Funded by the European Union

Conclusion and Next Steps

The conducted performance analyses demonstrate significant improvements across all main components of the system — from map rendering to database interaction and authentication. The migration from WMS to WMTS proved to be the most effective optimization, reducing map response times from more than 60 s to sub-second levels and stabilizing performance under peak load. Database optimizations, including spatial indexing and precomputed aggregations, yielded substantial gains, with query runtimes reduced by factors of 10 – 1000×. The login path shows consistent, low latency and can be considered stable.

Remaining optimization potential exists primarily in the area of map layer performance. Differences in latency between regions (e.g., Lithuania) indicate further potential for improvement through more consistent WMTS configuration, extended seeding, and migration of WMS-based layers (e.g., orthophotos) into tile caches. Planned measures such as enabling in-memory caching, optimizing proxy headers, and switching SQL views to materialized views are expected to further stabilize and accelerate system response.

Next Steps

- Validate and fine-tune caching configurations (GeoWebCache parameters, memory cache).
- Implement proxy and browser cache optimizations to leverage client-side caching.
- Finalize and benchmark materialized views for frequently accessed SQL layers.
- Establish automated seeding and re-aggregation routines to maintain data freshness.
- Extend performance monitoring (P50/P95 metrics per layer/query) for continuous optimization.

Overall, the system is now in a much more stable and performant state, with a clear roadmap for further fine-tuning and sustainable scalability.

Funded by
the European Union