GlossAPI: ML-assisted Anonymization Layer and Targeted Pipeline Improvements for Greek Datasets

---

Project Page: GlossAPI

Repository: https://github.com/eellak/glossAPI

Mentors: Dimitrios Athanasopoulos (https://github.com/jimmmyss),
Nikos Tsekos (http://github.com/nikostsekos),
Panagiotis Skarvelis (https://github.com/sl45sms)

Project Size: 350 hrs

Table of Contents

1. Basic Details

Full Name: Khushi Agrawal

Email and GitHub Username: khushisaritaagrawal@gmail.com | khushiiagarwal

Communication: Discord - khushi_41383 , LinkedIn - khushiagrawal028

Your first language: Hindi / English (Fluent)

Location and Timezone: Bangalore, India IST (UTC+5:30)

Share links, if any, of your previous work on open source projects:

Kubeflow

PR #12598: Resolved IfPresent conditions in component.yaml for the backend. (Merged)

PR #12607: Added Literal parameter validation in the API Server and Driver. (Merged)

PR #3080: Added Helm chart configuration for the data cache. (Merged)

PR #3124: Added support for ClusterTrainingRuntimes in the Helm chart. (Merged)

+ ***6 more*** *pull requests across pipelines, trainer, and website documentation.*

KubeStellar (GitHub)

PR #2295: Implemented fixed node styling in the treeView component. (Merged)

PR #2259: Added validation and fixed access permission reset issues. (Merged)

PR #2253: Updated overflow properties and enhanced styling in HelmTab components. (Merged)

+ ***14 more*** *pull requests improving UI components, documentation, and user management features.*

Kserve (GitHub)

PR #5009: Enabled label and annotation propagation for llmisvc. (Merged)

PR #608: Added documentation detailing label and annotation propagation. (Open)

Mesa (GitHub)

PR #155: Fixed apply_plan() so it successfully flattens multiple tool call results. (Merged)

PR #138: Prevented move_one_step from crashing when used with OrthogonalMooreGrid. (Merged)

PR #123: Fixed CoT and ReWOO reasoning passing str to add_to_memory() where a dict is expected. (Merged)

PR #118: Aligned STLTMemory.get_prompt_ready() return type with the Memory ABC. (Merged)

Krkn-chaos (GitHub)

PR #93: Resolved a CSV parser error that occurred when scenarios had varying SLO columns. (Merged)

PR #79: Added a namespace parameter to node hog scenarios. (Merged)

PR #96: Added a stopping criteria framework for the genetic algorithm. (Merged)

+ ***10 more*** *pull requests focusing on chaos testing enhancements and metadata tracking.*

Kgateway-dev (GitHub)

PR #13395: Refactored and consolidated duplicate TLS logic into pluginutils. (Merged)

PR #13393: Added error string assertions in the inference extension validation tests. (Merged)

PR #13424: Refactored functions by consolidating them into a new generic helper. (Merged)

Knative (GitHub)

PR #8866: Suppressed verbose OTEL logging in EventTransform. (Merged)

For a complete log of all my open-source contributions, including pull requests and issues across all organizations, please view my ***GitHub Gist***

2. Your Motivation

2.1 What is your motivation to take part in Google Summer of Code?

I want to contribute to real production software that acts as the bedrock for researchers and institutions. As an AI Engineer who frequently builds Retrieval-Augmented Generation (RAG) pipelines and on-premise LLM infrastructure, I understand that models are only as good as the data feeding them. GSoC provides the rigorous structure I need to write high-impact, scalable code at the intersection of data security and machine learning, guided by experienced maintainers.

2.2 Why did you choose Open Technologies Alliance (GFOSS)?

Data privacy is the single biggest bottleneck in open dataset publication today. GFOSS is doing critical infrastructure work to democratize data for languages like Greek. By supporting tools like GlossAPI, GFOSS ensures that localized NLP resources can be safely published, which aligns perfectly with my commitment to scalable, open data ecosystems.

2.3 Why do you want to work on this particular project?

Building an anonymization layer is fundamentally a data-security and systems-engineering challenge. My background involves building scalable ML architectures and DevSecOps tools (like my work in the CNCF ecosystem), which directly translates to this problem.
Furthermore, I have spent significant time analyzing not just the current codebase, but the project's historical trajectory. The 2025 GSoC project (by Dimitrios Athanasopoulos) successfully built the OCR backbone and validated that academic papers and open books are the highest-value data sources for Greek NLP. However, publicly releasing those specific corpora is currently blocked by the personally identifiable information (PII) inherently present in academic PDFs, author contacts, ΑΦΜ tax IDs, and institutional affiliations. My project directly unblocks the publication step that the 2025 project made possible, making the full pipeline end-to-end usable for open dataset release.

2.4 What are your expectations from us during and after successful completion of the program?

During the program, I expect prompt feedback on architectural design documents, early flags if a direction is incompatible with the broader GFOSS ecosystem, and rigorous code reviews. After completion, I expect to remain a core contributor of this module.

3. Project Details

3.1 What are you making?

An ML-assisted anonymization layer for the GlossAPI dataset production pipeline. The system is a new `Corpus.anonymize()` phase that runs after `Corpus.clean()` and before `Corpus.section()`. It reads cleaned markdown files, detects sensitive entities using a hybrid rule and ML approach, replaces them with consistent placeholders, and generates an audit sidecar.

3.2 How will it impact Open Technologies Alliance (GFOSS)?

It will unlock terabytes of previously unpublishable Greek text (including the **~300GB of OpenArchives data** processed in **GSoC 2025**). By providing an automated, scalable way to mask sensitive PII, GFOSS can legally and ethically release datasets that are desperately needed for training localized **Greek LLMs**. Furthermore, the **sidecar JSON logs** generated by this module will automatically compile a massive annotated dataset of Greek named entities over time.

3.3 What technologies (programming languages, etc.) will you be using?

- **Python 3.11+** (Core implementation language)
- **spaCy 3.x** (el_core_news_lg for highly-optimized, CPU-capable Greek ML NER)
- **Regex (re)** (Deterministic detection for structured PII: ΑΦΜ, AMKA, phones)
- **pandas / pyarrow** (Metadata parquet integration for state tracking)
- **bisect** ($O(\log N)$ span-to-line offset remapping to preserve markdown layout)

3.4 The Visibility & Security Gap

Currently, when a document passes through `Corpus.clean()`, the sanitized markdown is passed directly to `Corpus.section()`. If the document contains a Greek Tax ID (ΑΦΜ) or a personal phone number, it is permanently baked into the downstream **JSONL export**. To fix this, the anonymization logic must be injected **before** the layout is classified into sections, but **after** the OCR noise has been cleaned.

3.5 Technical Architecture & Implementation plan

To fully explain how this implementation will work, I have mapped out the pipeline flow and the exact code structures I plan to introduce.

I. The Pipeline Injection Point

The anonymization logic must be injected before the layout is classified into sections, but after the OCR noise has been cleaned. All downstream stages will automatically inherit the anonymized text through the existing `self.markdown_dir` redirect mechanism.

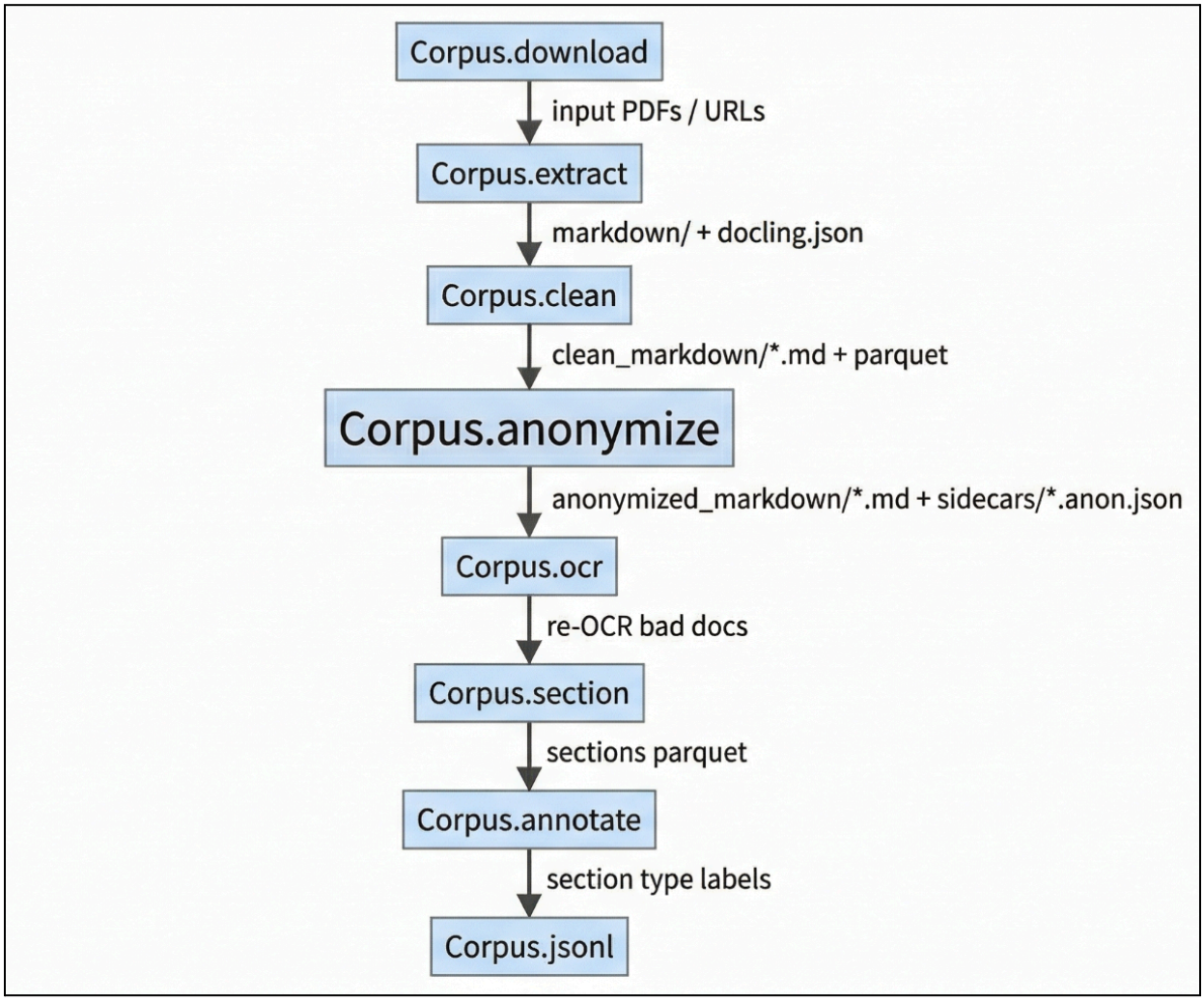

***Fig. 3.3.1**: GlossAPI Complete Pipeline. Shows the new `Corpus.anonymize()` phase injected between `clean()` and `ocr()`*

II. The Dual-Engine Detection Pipeline

Processing text line-by-line breaks the context window for ML models. Instead, the detection phase splits into two parallel engines operating on the full document string:

1. **Deterministic Rule Engine (`rule_detector.py`):** Handles Greek-specific regex patterns (ΑΦΜ, AMKA, phones). It includes an OCR normalization step to catch errors like a Latin 'O' replacing a Greek 'O' in an ID number.
2. **ML-Assisted NER Engine (`ml_detector.py`):** Integrates `spaCy` to process the cleaned markdown and extract `PERSON` and `ORG` entities.

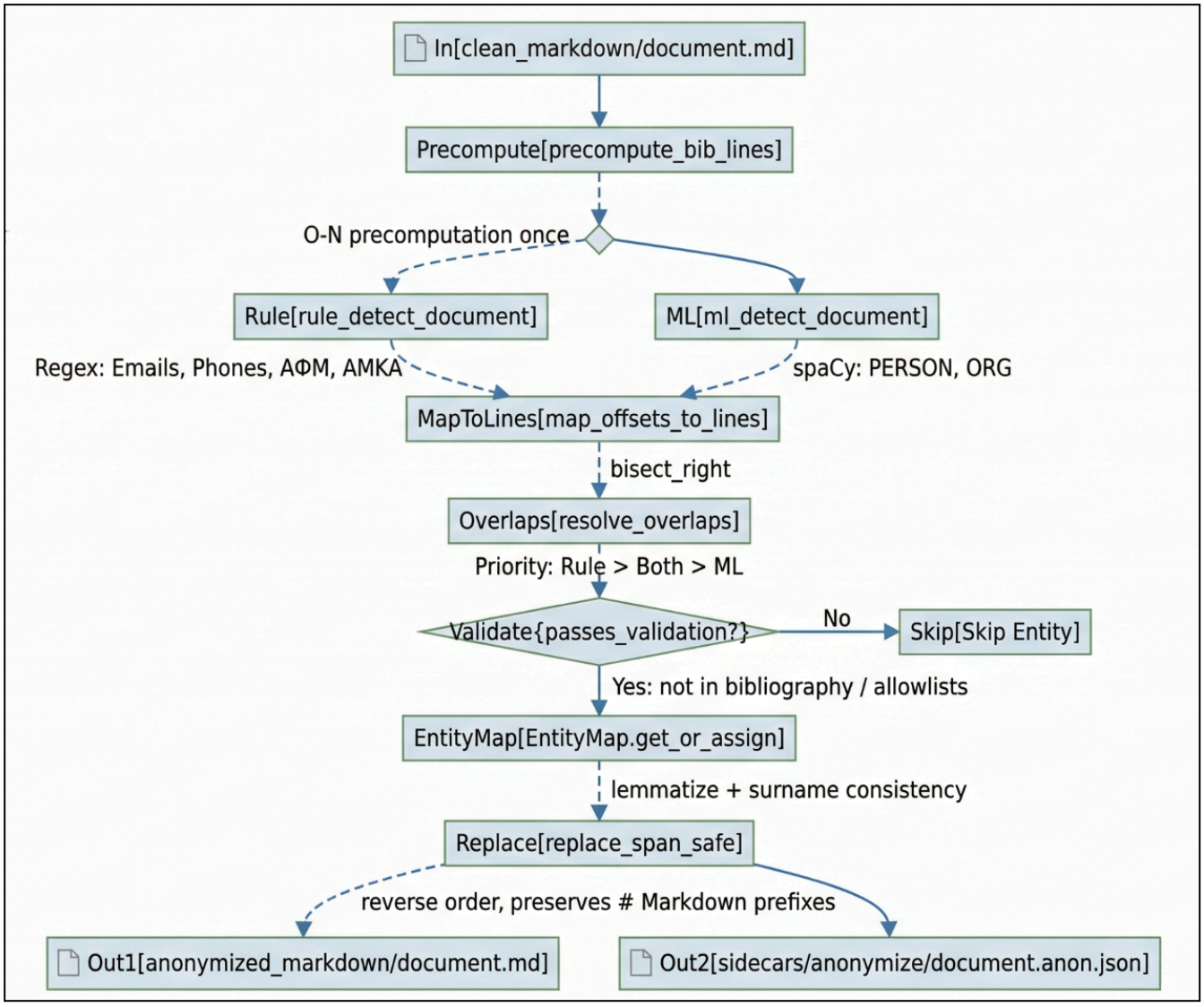

***Fig. 3.3.2**: Anonymization Phase Internal Pipeline showing parallel detection, overlap resolution, and replacement.*

III. Core Backend Implementation

A. The Orchestrator Mixin (`phase_anonymize.py`)

I will implement `AnonymizePhaseMixin` to handle state recovery and coordinate the anonymizer. It reads the existing Parquet metadata to skip previously processed files, preventing the pipeline from starting over if it crashes halfway through a large dataset.

```python
class AnonymizePhaseMixin:
    def anonymize(self, config: Optional[AnonymizeConfig] = None, skip_existing: bool = True) -> None:
        self.anonymized_markdown_dir.mkdir(parents=True, exist_ok=True)
        sidecars_dir = self.output_dir / "sidecars" / "anonymize"

        # State Recovery & Resumability Check via Parquet
        df = pd.read_parquet(self._get_cached_metadata_parquet())
        already_done = set(df[df["anonymization_status"] == "success"]["filename"]) if skip_existing else set()

        updates = []
        for md_path in self.cleaned_markdown_dir.glob("*.md"):
            if md_path.name in already_done: continue

            try:
                # Core Hybrid Execution
                result = self.anonymizer.anonymize_document(
                    md_path,
                    out_path=self.anonymized_markdown_dir / md_path.name,
                    sidecar_path=sidecars_dir / f"{md_path.stem}.anon.json"
                )
                updates.append({"filename": md_path.name, "anonymization_status": "success"})
            except Exception as e:
                updates.append({"filename": md_path.name, "anonymization_status": "failed"})

        _merge_anonymization_updates(df, updates, self._get_cached_metadata_parquet())

        # CRITICAL: Redirect downstream pipeline stages
        self.markdown_dir = self.anonymized_markdown_dir
```

B. Overlap Resolution

If the Regex layer detects `[10:20]` and the ML layer detects `[15:30]`, replacing both will corrupt the text. I will implement a resolution function that prioritizes deterministic rules over ML predictions.

```python
def resolve_overlaps(rule_entities: List[Entity], ml_entities: List[Entity]) -> List[Entity]:
    """ Resolves span collisions to prevent markdown corruption. """
    all_entities = sorted(rule_entities + ml_entities, key=lambda e: e.start_char)
    resolved = []
    for current in all_entities:
        if not resolved:
            resolved.append(current)
            continue
        previous = resolved[-1]
        # Check for span collision
        if current.start_char < previous.end_char:
            # PRIORITY: Deterministic Rule (0) always overrides ML (1)
            if current.source == "RULE" and previous.source == "ML":
                resolved[-1] = current
            elif current.source == previous.source:
                # Tie-breaker: Longer span wins
                if (current.end_char - current.start_char) > (previous.end_char - previous.start_char):
                    resolved[-1] = current
        else:
            resolved.append(current)
    return resolved
```

C. Heading-Safe Replacement (Preventing Downstream Classifier Failure)

The pipeline's section classifier (`gloss_section_classifier.py`) relies on exact markdown syntax. If we strip the `#` tokens during anonymization, the classifier fails. `replace_span_safe` parses the token before replacing the sensitive text.

```python
def replace_span_safe(line: str, start: int, end: int, placeholder: str) -> str:
    """ Preserves structural markdown prefixes during replacement. """
    heading_match = re.match(r"^(#+\s+)(.*)", line)
    if heading_match:
        prefix, payload = heading_match.groups()
        # Adjust offsets to account for the stripped prefix
        adjusted_start = start - len(prefix)
        adjusted_end = end - len(prefix)
        safe_payload = payload[:adjusted_start] + placeholder + payload[adjusted_end:]
        return f"{prefix}{safe_payload}"
    return line[:start] + placeholder + line[end:]
```

3.6 Test Plan

To ensure absolute reliability without causing regressions in GlossAPI's native extraction, I will enforce three testing layers:

- **Unit Tests (`test_anonymize_logic.py`):** I will rigorously verify the classification logic against synthetic, OCR-corrupted Greek strings. (e.g., Testing if the Regex correctly normalizes a Greek 'o' confused with a Latin 'o' inside a tax ID).
- **Structural Invariant Tests:** I will use pytest fixtures to assert that `Corpus.anonymize()` never alters the total line count of a document, ensuring `_format_academic_document()` (which computes physical page layout fractions) does not silently drift.
- **Integration Tests (`test_corpus_flow.py`):** I will execute the full pipeline from `download()` to `jsonl()` export, verifying that the final JSON outputs correctly inherit the anonymized text via the updated `markdown_dir` fallback chain.

3.7 Implementation & Output Verification

To definitively validate this architecture and eliminate technical risk prior to the coding period, I have already implemented a fully functional, local Proof of Concept (PoC) integrated directly into the `eellak/glossAPI` codebase.

The complete source code, including the `AnonymizePhaseMixin`, the hybrid detection engine, the bisection mapping logic, and the automated test suite, can be reviewed on my fork:

Link to your github branch - **feature/anonymization-poc**

Output 1: Test Suite Validation & Edge Case Handling

I made a suite of 12 targeted `pytest` cases to verify that the anonymization layer behaves predictably under stress and does not cause regressions in the extraction pipeline. The tests explicitly validate:

1. **Deterministic Accuracy:** Verifies 100% catch rates for Greek formatting of emails, +30 phone numbers, and ΑΦΜ/ΑΜΚΑ variants, including synthetic strings heavily corrupted by simulated OCR noise (e.g., Latin 'O' instead of Greek 'O').
2. **Structural Invariants:** Asserts that `replace_span_safe()` strictly maintains the total document line count and preserves markdown table structures (`|---|`), preventing the `_format_academic_document()` layout fraction logic from silently drifting.
3. **Overlap Resolution:** Forces collisions between the Regex and spaCy ML engines to verify the `Rule(0) > Both(1) > ML(2)` priority queue successfully prevents string corruption.
4. **$O(N)$ Context Filtering:** Verifies that entities detected within the `precompute_bib_lines()` boundaries (e.g., authors in the `References/Bibliography` section) are safely suppressed from redaction.
5. **State Resumability:** Confirms the mixin correctly reads Parquet metadata to skip previously processed documents.

```
~/Desktop/glossAPI- feature/anonymization-poc*
❯ PYTHONPATH=src .venv/bin/pytest tests/test_anonymize_phase.py -v

========================= test session starts =========================
platform darwin -- Python 3.11.13, pytest-9.0.2, pluggy-1.6.0 -- /Users/khushiagrawal/Desktop/glossAPI-/.venv/bin
/python3
cachedir: .pytest_cache
rootdir: /Users/khushiagrawal/Desktop/glossAPI-
configfile: pyproject.toml
collected 12 items

tests/test_anonymize_phase.py::test_corpus_exposes_anonymize PASSED [  8%]
tests/test_anonymize_phase.py::test_rule_detector_email_and_phone PASSED [ 16%]
tests/test_anonymize_phase.py::test_heading_prefix_preserved PASSED [ 25%]
tests/test_anonymize_phase.py::test_entity_inside_heading_prefix_not_replaced PASSED [ 33%]
tests/test_anonymize_phase.py::test_bibliography_not_anonymized PASSED [ 41%]
tests/test_anonymize_phase.py::test_line_count_unchanged PASSED  [ 50%]
tests/test_anonymize_phase.py::test_entity_map_surname_consistency PASSED [ 58%]
tests/test_anonymize_phase.py::test_dry_run_no_modification PASSED [ 66%]
tests/test_anonymize_phase.py::test_sidecar_written PASSED       [ 75%]
tests/test_anonymize_phase.py::test_overlap_resolution_rule_wins PASSED [ 83%]
tests/test_anonymize_phase.py::test_anonymize_creates_output_dir PASSED [ 91%]
tests/test_anonymize_phase.py::test_anonymize_redirects_markdown_dir PASSED [100%]

========================= 12 passed in 1.43s =========================

~/Desktop/glossAPI- feature/anonymization-poc*
❯
```

***Fig. 3.7.1**: Demonstrates that the anonymization layer integrates with Corpus and passes isolated test cases, including heading structural safety and $O(N)$ bibliography suppression.*

Output 2: Greek Input vs Anonymized Output

To demonstrate the precision of the text replacement, below is a simulated representation of the PoC's text transformation. Notice how the inline PII is masked while the surrounding markdown syntax (`# and **`) and bibliography context remain completely untouched.

Raw Input (`clean_markdown/doc_001.md`) and **Anonymized Output (`anonymized_markdown/doc_001.md`)**

Fig. 3.7.2: *This shows the exact text transformation. Emails, phones, and ΑΦΜ (tax IDs) are replaced inline. The Markdown table structure remains intact, and the author's name in the References section is safely ignored based on context filtering.*

Output 3: JSON Audit Sidecar

Data destruction without an audit trail is unacceptable for production datasets. For every processed document, the PoC generates an `.anon.json` sidecar file alongside the markdown. This provides exact character offsets, entity types, and algorithmic confidence scores, ensuring maintainers retain total visibility into what the pipeline altered.

Generated Sidecar (`sidecars/anonymize/doc_001.anon.json`):

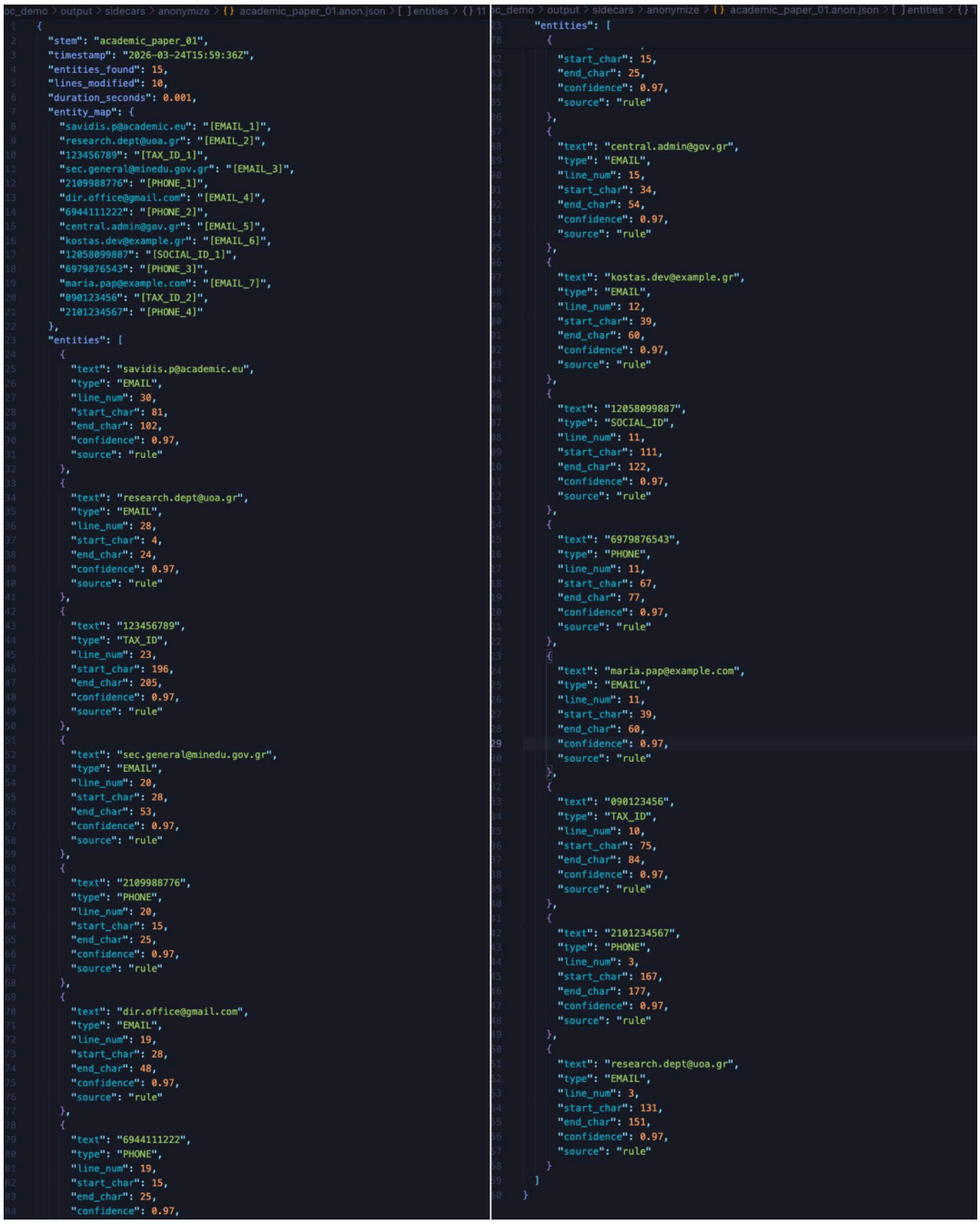

Fig 3.7.3: *This standardized sidecar format creates a rich, highly structured dataset perfectly primed for fine-tuning future GreekBERT NER models.*

4. Timeline

Weekly Time Commitment & Planned Absences (Exams):

- **Standard Capacity:** I will dedicate **~40-45 hours per week** to this project.
- **Off-the-Grid:** My university end-semester exams begin on May 18th and will last for approximately 10 days. To mitigate this, I have explicitly built a buffer week into my project timeline to accommodate this without impacting the final deliverables.

4.1 Community Bonding Period (May 1 – May 24)

- **Week 1 & 2**
  Deep-read the remaining unread pipeline files (`gloss_section_classifier.py` and the Docling OCR adapters) to map out exactly how line fractions are computed. Finalize the **AnonymizeConfig API** surface with mentors and curate a **50-document gold-standard evaluation dataset**.
- **Week 3 & 4**
  Light, asynchronous communication with mentors to finalize the codebase structure. Set up isolated testing environments and ensure `pytest tests/test_corpus_flow.py` passes cleanly on my local machine.

4.2 Phase 1: Core Scaffolding & Deterministic Rules (May 25 – June 21)

- **Week 1**
  Create the `src/glossapi/anonymize/` package scaffolding. Inject **AnonymizePhaseMixin** into the Corpus Method Resolution Order (**MRO**). Write initial blank pytest fixtures.
- **Week 2**
  Develop the deterministic **Greek Regex Engine** (`patterns_el.py`) for emails, +30 phone numbers, ΑΦΜ (Tax IDs), and AMKA. Implement basic OCR confusable-character normalization.
- **Week 3**
  Build the mathematically complex **$O(\log N)$ bisection mapping logic** to translate absolute character offsets back to exact line numbers. Implement the heading-safe `replace_span_safe()` function.

- **Week 4**
  Implement sidecar JSON logging (`.anon.json`). Update Parquet metadata schema (`parquet_schema.py`) to include **anonymization_status** for pipeline resumability.

4.3 Phase 2: ML Integration & Contextual Resolution (June 22 – July 19)

- **Week 5**
  Implement the **$O(N)$ single-scan pre-computation** (`precompute_bib_lines()`) to suppress over-redaction in Bibliography and Reference sections.
- **Week 6**
  Integrate spaCy (`el_core_news_lg`) via lazy-loading for **batched NLP inference** over full document chunks.

MIDTERM DELIVERABLES (6 – July 10)

Prior to the midterm evaluation, I will deliver a fully functional, resumable pipeline that successfully executes the Deterministic Rule Layer, logs sidecars, and runs initial batched ML inference without breaking the markdown layout.

- **Week 7**
  Implement the strict `resolve_overlaps()` **priority algorithm (Rule > ML)** to prevent string corruption during span collisions.
- **Week 8**
  Develop the EntityMap for **cross-inflection Greek surname matching** (ensuring variations like Παπαδόπουλος and Παπαδόπουλου map to the exact same placeholder token).

4.4 Phase 3: Export Fix, Evaluation & Hardening (July 20 – August 24)

- **Week 9**
  Execute surgical updates to the `_iter_jsonl_records()` fallback chain in `phase_export.py` to natively prioritize the protected text during final dataset generation.
- **Week 10**
  Run the full hybrid engine against the 50-document gold-standard set. Generate a comprehensive **Precision/Recall/F1 evaluation report** contrasting false-positive and false-negative leak rates.

- **Week 11**
  Draft extensive user-facing documentation in `docs/anonymization.md`. Write the `scripts/verify_anonymization.py` script for maintainers to easily audit dataset leakage.
- **Week 12 (BUFFER WEEK)**
  Dedicated exclusively to addressing pending PR reviews, squashing commits, fixing newly discovered edge-case bugs, and covering any tasks that spilled over from previous weeks.
- **Week 13**
  Final end-to-end integration testing. Ensure all CI/CD pipelines pass perfectly before the Coding Ends deadline.

FINAL EVALUATION DELIVERABLES (August 24 – August 31)

Prior to the final evaluation, I will deliver the fully merged, production-ready hybrid anonymization layer, accompanied by the mathematical evaluation report, maintainer audit scripts, and comprehensive documentation.

4.5 Post-GSoC Plans

Do you plan to continue working on the project after GSOC ends?
Absolutely. I plan to stay as a long-term, active contributor for GlossAPI. Once this anonymization pipeline is in production, it will begin automatically generating thousands of highly accurate .anon.json sidecar files. My immediate next goal post-GSoC is to utilize these logs to **fine-tune a specialized GreekBERT model** for academic Named Entity Recognition (NER), contributing a powerful, open-source AI asset back to the GFOSS ecosystem.

5. Motivation

Convince us that you will be a good fit for the GFOSS project you have selected:

Building production infrastructure requires anticipating how a system will fail at scale. My active track record in the **Cloud Native Computing Foundation (CNCF)** proves my ability to navigate massive codebases. I have successfully merged pull requests involving full-stack parameter validation and pipeline optimization in complex projects like **Kubeflow** and **Kgateway**.

More importantly, I understand the GlossAPI codebase at the level of **individual line references**. For instance, I know exactly how `_format_academic_document()` computes layout fractions in `gloss_section.py`. This deep understanding is exactly why my anonymization design strictly preserves document line counts to prevent downstream classifier failures.

I have been in constant communication with the maintainers (**@jimmys and @tsksn**). I proactively built a working **Proof of Concept (PoC)** and translated their feedback into architectural fixes.

This feedback driven approach and my deep familiarity with the codebase ensure I can hit the ground running and deliver a production-ready anonymization layer this summer.

Khushi Agrawal

+91-9109356699 | khushisaritaagrawal@gmail.com | GitHub: khushiiagrawal | LinkedIn: khushiagrawal028 | Portfolio: khushiagrawal.tech

EXPERIENCE

Klugsys (Hayy.ai) **November 2025 – Present**

AI Engineer Intern: LLM Systems, RAG Pipelines, Model Fine-Tuning, Kubernetes Deployment *Germany (Remote)*

- Built an **on-premises LLM infrastructure** using **FastAPI**, supporting secure inference and **retrieval-augmented** workflows with strong data isolation and no cloud dependency.
- Engineered LLM services using **Docker and Kubernetes**, enabling **scalable GPU-backed inference** and improving system throughput by **2x**.

KratiTech Pvt Ltd **June 2025 – August 2025**

AI Developer Intern: Computer Vision, LLMs, NLP, RAG, Model Deployment *Kanpur (Remote)*

- Led a team of **8 interns** to develop a **Retrieval-Augmented Generation**-based chatbot leveraging **vector databases** to answer industry-specific queries and **automate processes** for Uttar Pradesh State Industrial Development Authority.
- Developed a **CNN-based face recognition system** achieving **95% accuracy** in feature extraction and recognition.

QuantArena **December 2024 – February 2025**

Web Developer Intern: React.js, Tailwind CSS, Redux, Node.js, Express.js, MySQL *Pune (Remote)*

- Developed an **analytics and risk management platform** with **three levels of access control** and secure data access.
- Deployed the application on **AWS EC2**, managed **DNS with Route 53** and configured **Nginx** for optimized performance.

PROJECTS

Adaptive Threat Modeler | *Go, TypeScript, Semgrep, Docker, Kubernetes, Helm, Prometheus, Grafana* **Demo** | **GitHub**

- Architected an **AI-driven DevSecOps platform** to statically scan over 11+ languages for **OWASP Top 10** vulnerabilities, generating context-aware fixes, automated **GitHub issues**, and real-time **Slack alerts** via **MCP**.
- Deployed a production cloud-native stack and implemented full observability using **Prometheus metrics** and **Grafana dashboards** for latency, throughput, and vulnerability severity tracking.

MCP Research Assistant | *Python, Model Context Protocol, arXiv API, Claude Desktop, REST APIs* **GitHub**

- Built an **AI research assistant** with modular **research, file system, and fetch services** for automated academic workflows.
- Integrated with **arXiv API**, Claude Desktop, and custom REST APIs to fetch, analyze, and organize **1,000+ research papers** and files.

ACHIEVEMENTS & OPEN SOURCE

- **Patent Published:** Co-inventor of "Sustainability Analysis Method and System for Item Classification and Disposal Recommendations" **(Application No. 202541098856 A)**, filed October 2025, published November 2025. **Link**
- **Open Source Contributions:** Active contributor to **Kubeflow**, **KubeStellar**, **krkn-chaos**, **kgateway-dev**, **kserve**, **knative** and **OpenMS**, with multiple merged PRs. **Link**
- **Selected for ACM Winter School:** Chosen among **40 students across India** to study **Responsible AI** at XIMB, under faculty from **IIT Kharagpur**.
- **Winner, FantomCode'25:** Won a 24-hour national-level hackathon among **400** participants at RVITM, Bangalore.
- **Top 10 Finalist at IITM:** Ranked in Top 10 among **1200** participants in Global Hyperloop Conference 2024, IIT Madras.
- **Runner-up, UI/UX:** Secured 2nd place in UI/UX among **80** participants at Catalysis 3.0, DSCE.
- **CodeChef 2-Star:** Global Rank: **1220**, Max Rating: **1438**

CERTIFICATIONS

Hugging Face: Agents Course – Hands-on training in building AI agents using Hugging Face libraries. **Link**

Infosys: Microsoft SQL from A to Z Course – SQL queries, optimization and database management. **Link**

EDUCATION

Dayananda Sagar College of Engineering, Bangalore **CGPA: 9.30/10**

Bachelor of Engineering in Information Science and Engineering *2023 – 2027*

Indian Institute of Technology, Mandi **CGPA: 9.02/10**

Minor in Artificial Intelligence and Data Science *2025 – 2026*

SKILLS

Programming Languages: C, C++, Python, Golang, TypeScript

Frameworks: FastAPI, Flask, React.js, Next.js, Node.js, Express.js

Libraries: TensorFlow, PyTorch, Scikit-learn, Pandas, NumPy, Matplotlib, Data Visualization, LangChain, LangGraph

Tools & Platforms: Git, Docker, Kubernetes, Prometheus, Grafana, AWS, CI/CD, MongoDB, MySQL, Vector Databases