NLP Project Issues List - Phase 2 Period 5

Organized by component and team member, with a focus on UI tasks for Noam

Table of Contents

- Milestones
- Database & Data Pipeline Integration
- Natural Language Processing & Classification
- Smart Response & Summarization
- Backend Logic & API
- Desktop GUI – C++ Interface

Milestones

- **M1:** Database Schema & Initial Setup complete (Friday, April 11, 2025)
- **M2:** Core ML models Integrated (Tuesday, April 22, 2025)
- **M3:** NLP pipeline Working (Tuesday, April 29, 2025)
- **M4:** Backend API & Processing Complete (Tuesday, May 6, 2025)
- **M5:** Complete System integration & Testing (Tuesday, May 13, 2025)

1. Database & Data Pipeline Integration

1.1 Finalize and Implement the Email Database

1.1.1 Design the core SQLite schema for emails and metadata (Esteban)

- ☐ IS-001: Research efficient schema designs for email storage
- ☐ IS-002: Create entity-relationship diagrams for the database
- ☐ IS-003: Document schema design decisions and constraints

1.1.2 Build C++ bindings for DB (pybind11) (Noam)

- ☐ IS-004: Research pybind11 integration best practices for SQLite
- ☐ IS-005: Set up environment for pybind11 development
- ☐ IS-006: Create wrapper class for SQLite database operations
- ☐ IS-007: Implement basic CRUD operations via pybind11
- ☐ IS-008: Add transaction support and error handling
- ☐ IS-009: Write unit tests for C++ bindings
- ☐ IS-010: Optimize binding performance for large datasets

1.1.3 Implement DB schema in code and validate it (Esteban)

- ☐ IS-011: Convert schema design to SQLite DDL statements
- ☐ IS-012: Implement schema migration system
- ☐ IS-013: Create validation tests for schema integrity

1.1.4 Stress test DB with large email volume (Esteban)

- ☐ IS-014: Generate synthetic email dataset for testing
- ☐ IS-015: Develop benchmarking framework for DB operations
- ☐ IS-016: Optimize indexing for common query patterns

1.2 Dataset Processing System

1.2.1 Preprocess Enron dataset for NLP tasks (Jiang)

- ☐ IS-017: Clean and normalize Enron dataset text
- ☐ IS-018: Extract metadata from email headers
- ☐ IS-019: Create training/testing splits for ML models

1.2.2 Add multithreaded loading for >10k emails (Jiang)

- ☐ IS-020: Design thread pooling strategy for data loading
- ☐ IS-021: Implement thread-safe database operations
- ☐ IS-022: Add progress reporting for bulk operations

1.2.3 CLI/GUI dataset import interface (Noam)

- ☐ IS-023: Design CLI command structure for dataset imports
- ☐ IS-024: Implement file format validation
- ☐ IS-025: Create progress reporting for long imports
- ☐ IS-026: Design GUI import dialog interface
- ☐ IS-027: Implement Qt-based import wizard
- ☐ IS-028: Add error handling and validation feedback

1.2.4 Implement anonymization filter for personal data (Jiang)

- ☐ IS-029: Research and select PII detection algorithms
- ☐ IS-030: Implement name and email address anonymization
- ☐ IS-031: Add phone number and address detection/anonymization

1.2.5 Extend support for Kaggle/UCI datasets (Esteban)

- ☐ IS-032: Research common email dataset formats
- ☐ IS-033: Create format converters for compatible datasets

2. Natural Language Processing & Classification

2.1 Email Category Classification System

2.1.1 Migrate baseline to RoBERTa/BERT (Giorgos)

- ☐ IS-034: Set up HuggingFace transformers environment
- ☐ IS-035: Implement BERT-based classification pipeline
- ☐ IS-036: Compare performance with existing baseline

2.1.2 Benchmark vs hybrid model (accuracy/speed) (Giorgos)

- ☐ IS-037: Design hybrid model architecture
- ☐ IS-038: Implement benchmarking framework
- ☐ IS-039: Document performance tradeoffs

2.1.3 Add confidence thresholds via config (Giorgos)

- ☐ IS-040: Implement configurable confidence scoring
- ☐ IS-041: Add multi-label classification support
- ☐ IS-042: Create visualization for classification confidence

2.1.4 Serialize model predictions into DB (Jiang)

- ☐ IS-043: Design database schema for classification results
- ☐ IS-044: Implement efficient bulk storage of predictions

2.2 Named Entity Recognition Integration

2.2.1 Integrate HuggingFace NER pipeline (Remi)

- ☐ IS-045: Configure NER model for email-specific entities
- ☐ IS-046: Implement entity extraction pipeline
- ☐ IS-047: Optimize for performance in email context

2.2.2 Tagging output mapping and entity types (Esteban)

- ☐ IS-048: Define entity type taxonomy for email domain
- ☐ IS-049: Create mapping between NER outputs and DB schema

2.2.3 Visualize NER tags in GUI (Octavian)

- ☐ IS-050: Design interactive entity highlighting UI
- ☐ IS-051: Implement hover tooltips for entity information

2.2.4 Store and index entities in DB (Jiang)

- ☐ IS-052: Extend database schema for entity storage
- ☐ IS-053: Implement efficient entity indexing and search

2.3 Emotion & Tone Detection

2.3.1 Replace TextBlob with VADER/FinBERT (Remi)

- ☐ IS-054: Evaluate VADER and FinBERT performance on email data
- ☐ IS-055: Implement selected sentiment analysis model
- ☐ IS-056: Create compatibility layer with existing pipeline

2.3.2 Add tone classification (casual, formal, sarcasm) (Remi)

- ☐ IS-057: Research tone classification techniques
- ☐ IS-058: Create training data for email tone classification
- ☐ IS-059: Implement and evaluate tone classifier

2.3.3 Integrate tone into reply and GUI (Octavian)

- ☐ IS-060: Design UI elements for tone display
- ☐ IS-061: Connect tone analysis to reply generation system

3. Smart Response & Summarization

3.1 Summarization Engine

3.1.1 Build logic to choose best summarization method (Giorgos)

- ☐ IS-062: Research email-specific summarization techniques
- ☐ IS-063: Implement selection logic based on email properties
- ☐ IS-064: Create evaluation framework for summarization quality

3.1.2 Implement extractive summarization (TextRank or BERTSum) (Noam)

- ☐ IS-065: Research and compare TextRank vs BERTSum implementations
- ☐ IS-066: Implement text preprocessing for summarization
- ☐ IS-067: Integrate selected algorithm with Python bindings
- ☐ IS-068: Optimize for speed with longer documents
- ☐ IS-069: Test summarization quality on Enron dataset

3.1.3 Integrate summarization into pipeline output (Noam)

- ☐ IS-070: Design summarization output format
- ☐ IS-071: Add configuration options for summary length
- ☐ IS-072: Implement pipeline connector for summarization module
- ☐ IS-073: Create summarization caching layer
- ☐ IS-074: Add serialization of summaries to database
- ☐ IS-075: Write integration tests for summarization pipeline

3.1.4 Add summary method selector (based on email length) (Giorgos)

- ☐ IS-076: Define threshold criteria for summary method selection
- ☐ IS-077: Implement fallback mechanisms for summarization
- ☐ IS-078: Evaluate performance across different email lengths

3.2 Smart Reply Generator

3.2.1 Add urgency/tone awareness to reply logic (Remi)

- ☐ IS-079: Develop urgency detection algorithm
- ☐ IS-080: Implement tone-aware reply templates
- ☐ IS-081: Create rule system for urgent email handling

3.2.2 Implement thread-based context memory (Remi)

- ☐ IS-082: Design conversation thread data structure
- ☐ IS-083: Implement thread context extraction algorithm
- ☐ IS-084: Add persistent storage for thread context

3.2.3 Fine-tune small LLM for reply suggestions (Giorgos)

- ☐ IS-085: Research suitable small LLMs for email replies
- ☐ IS-086: Create fine-tuning dataset from email corpora
- ☐ IS-087: Implement model training and evaluation pipeline

3.2.4 Connect smart reply to GUI approve/edit UI (Octavian)

- ☐ IS-088: Design reply suggestion interaction UI
- ☐ IS-089: Implement approval/edit workflow
- ☐ IS-090: Add template management for recurring responses

4. Backend Logic & API

4.1 Modular Pipeline & Logic System

4.1.1 Modularize main flow (Loader –> Classifier –> Summary) (Octavian)

- ☐ IS-091: Design pipeline architecture with clear interfaces
- ☐ IS-092: Implement plugin system for pipeline components
- ☐ IS-093: Create visualization for pipeline data flow

4.1.2 Add config system for thresholds (Noam)

- ☐ IS-094: Design configuration file format (YAML/JSON)
- ☐ IS-095: Implement configuration parsing and validation
- ☐ IS-096: Create default configurations for all modules
- ☐ IS-097: Add runtime configuration update capability
- ☐ IS-098: Implement configuration UI in settings panel
- ☐ IS-099: Test configuration changes across pipeline

4.1.3 Add async processing queue (multiprocessing/Celery) (Noam)

- ☐ IS-100: Research best approach (multiprocessing vs Celery)

- ☐ IS-101: Design queue architecture and task structure
- ☐ IS-102: Implement worker pool and task distribution
- ☐ IS-103: Add progress monitoring and reporting
- ☐ IS-104: Implement failure handling and retries
- ☐ IS-105: Test performance with large email batches

4.2 REST API for Predictions

4.2.1 Add endpoints for predictions and metadata search (Jiang)

- ☐ IS-106: Design REST API specification
- ☐ IS-107: Implement prediction request/response format
- ☐ IS-108: Add authentication and rate limiting

4.2.2 Create backend API (Flask or FastAPI) (Jiang)

- ☐ IS-109: Set up API framework and environment
- ☐ IS-110: Implement route handlers for core functionality
- ☐ IS-111: Create API documentation with Swagger/OpenAPI

4.2.3 Add endpoints for metadata search (Esteban)

- ☐ IS-112: Design search query format and capabilities
- ☐ IS-113: Implement efficient metadata search algorithms
- ☐ IS-114: Add result pagination and sorting options

5. Desktop GUI – C++ Interface

5.1 Interface Design & Tagging

5.1.1 Design inbox layout and mock UI (Octavian)

- ☐ IS-115: Create wireframes for main application screens
- ☐ IS-116: Define application style guide and theme
- ☐ IS-117: Create mockups for approval

5.1.2 Design tags for urgency, tone, and category (Octavian)

- ☐ IS-118: Research effective visual tagging systems
- ☐ IS-119: Design tag color scheme and iconography
- ☐ IS-120: Create interactive tag filtering concept

5.1.3 Implement Qt layout with dummy data (Noam)

- ☐ IS-121: Set up Qt development environment
- ☐ IS-122: Create main application window and layouts
- ☐ IS-123: Implement inbox view with dummy emails
- ☐ IS-124: Add email detail view components
- ☐ IS-125: Create placeholder visualization widgets

- ☐ IS-126: Implement dark/light theme support
- ☐ IS-127: Add responsive layout adjustments

5.2 GUI Logic & Integration

5.2.1 Display email + model outputs in GUI (Noam)

- ☐ IS-128: Design email detail view layout
- ☐ IS-129: Create component for displaying email content with HTML support
- ☐ IS-130: Implement visualization of classification results
- ☐ IS-131: Add NER tag highlighting in email content
- ☐ IS-132: Create emotion/tone display component

5.2.2 Connect GUI with backend predictions (Giorgos)

- ☐ IS-133: Implement API client for prediction requests
- ☐ IS-134: Add caching layer for frequently accessed predictions
- ☐ IS-135: Create real-time update mechanism

5.3 Smart User Interactions

5.3.1 Add smart reply approve/edit dropdown (Octavian)

- ☐ IS-136: Design inline reply suggestion UI
- ☐ IS-137: Implement edit/approve interaction flow
- ☐ IS-138: Add keyboard shortcuts for common actions

5.3.2 Build inbox analytics view (Esteban)

- ☐ IS-139: Design analytics dashboard layout
- ☐ IS-140: Create email volume and category visualizations
- ☐ IS-141: Implement time-based analytics views

5.3.3 Add filters and search bar (Remi)

- ☐ IS-142: Design advanced search syntax
- ☐ IS-143: Implement search highlighting in results
- ☐ IS-144: Create saved search/filter functionality