ShipCube Intern Task Document

Project: Email Add-on for Supply Chain Client Management

Nishant **Priyanka**

October 6, 2025

1. Objective

Develop an intelligent **email add-on** for supply chain client handling that can:

1. Read and process a high volume of client emails,
2. Filter them on a priority basis using sentiment analysis,
3. Summarize client messages, and
4. Suggest intelligent answers for client queries.

The goal is to improve client communication efficiency and enable quick triage of urgent issues in supply chain management workflows.

2. Role Division

Nishant: Backend, NLP/ML modeling, and API development.
Priyanka: UI/UX, integration with Gmail/Outlook, testing, and documentation.
Both will collaborate on dataset creation, labeling, evaluation, and end-to-end system integration.

3. Phase 1 — Setup and Data Preparation

Joint Tasks

- Create GitHub repository with directories: `/docs`, `/src/backend`, `/src/models`, `/src/addon`, and `/tests`.
- Collect or synthesize representative email datasets with fields: sender, timestamp, subject, thread ID, and content.

Nishant

- Set up the backend environment (Python, Conda, Docker).
- Build an email ingestion prototype supporting IMAP and Microsoft Graph API.
- Define a data schema for emails: `(raw_text, cleaned_text, sender_role, timestamp, thread_id)`.

Priyanka

- Create UI wireframes for inbox visualization: priority badges, summaries, and suggested replies.
- Build a local JSON-based mailbox simulator to test early backend endpoints.

4. Phase 2 — Preprocessing and Labeling

Nishant

- Implement preprocessing: remove HTML tags, signatures, normalize text, detect language.
- Add Named Entity Recognition (NER) for supply chain entities (PO#, ETA, product codes).
- Build a labeling tool to tag emails for priority, sentiment, intent, and SLA urgency.

Priyanka

- Design labeling guidelines with examples and edge cases.
- Coordinate labeling sessions and maintain labeled datasets in CSV/JSONL format.

5. Phase 3 — Priority and Sentiment Pipeline

Nishant

- Implement a sentiment classifier using transformer models (e.g., `sentence-transformers`).
- Design prioritization logic based on:
  - Sentiment polarity and intensity,
  - Intent type (complaint/query/status),
  - Customer tier and response delay,
  - Presence of urgency keywords.
- Develop REST endpoint `/classify` returning priority, sentiment, and intent.

Priyanka

- Integrate priority and sentiment display in UI with badges and color coding.
- Implement a rule-editor interface for modifying priority logic weights.

6. Phase 4 — Email Summarization

Nishant

- Implement abstractive summarization using T5/BART-based transformer.
- Generate structured output: short subject line, 3-bullet summary, and extracted entities.
- Add post-processing to redact PII and maintain numeric accuracy.
- Expose API endpoint `/summarize`.

Priyanka

- Embed summaries into the add-on interface with expandable detail view.
- Highlight extracted entities and add an "Edit Summary" option for user correction.

7. Phase 5 — Suggested Replies (Answer Drafting)

Nishant

- Design the reply suggestion pipeline:
  1. Intent recognition and slot filling,
  2. Retrieval of similar past replies using FAISS/ElasticSearch,
  3. Generation of formal, concise, and actionable drafts.
- Build endpoint `/suggest_reply` returning multiple draft options with evidence sources.

Priyanka

- Create a compose-pane integration for one-click reply insertion.
- Add inline evidence markers (e.g., "Source: KB#42").
- Implement user feedback logging (accept/edit/reject statistics).

8. Phase 6 — Integration, Testing, and Evaluation

Joint Tasks

- Define test metrics for classification, summarization, and reply quality.

Nishant

- Implement unit tests for sentiment and intent classification.
- Evaluate summaries using ROUGE metrics and human scoring.
- Log model accuracy, latency, and confidence metrics.

Priyanka

- Conduct user acceptance tests (UAT) with sample email batches.
- Collect qualitative feedback for UI/UX improvement.
- Simulate complete flow: ingest → classify → summarize → suggest reply.

9. Phase 7 — Deployment and Monitoring

Nishant

- Containerize all backend components using Docker and write deployment scripts.
- Implement logging, monitoring, and alerting for API performance.

Priyanka

- Package the prototype as a Gmail/Outlook add-on.
- Prepare installation and user demo instructions.
- Develop sample email scenarios (urgent complaint, general query, multi-thread summary).

10. Phase 8 — Documentation and Final Demo

Priyanka (Lead)

- Write user manual describing each feature, interpretation of priority tags, and editing options.
- Prepare PowerPoint slides for final demo: problem, solution workflow, and live demonstration.

Nishant

- Draft technical documentation (API endpoints, model architectures, retraining procedure).
- Write detailed README for developers.

11. Evaluation Metrics

- **Classification:** Precision, recall, F1-score for priority and intent.
- **Sentiment:** Confusion matrix against labeled data.
- **Summarization:** ROUGE/ROUGE-L scores and readability assessment.
- **Reply Suggestions:** Human acceptance rate and edit distance from final replies.
- **System Performance:** Throughput (emails/sec), latency per request, and failure rate.

12. Technology Stack

- **Languages:** Python (backend), JavaScript/React (frontend).
- **Frameworks:** FastAPI, Hugging Face Transformers, FAISS, spaCy.
- **Databases:** PostgreSQL / MongoDB for metadata, S3 for storage.
- **Deployment:** Docker, GitHub Actions (CI/CD).
- **Security:** OAuth2 for email APIs, PII redaction, encrypted data handling.

13. Privacy and Compliance

- Redact or hash personally identifiable information (PII) before training.
- Maintain compliance with GDPR and Indian Data Protection Acts.
- Exclude real credentials from repositories and use secure secrets management.

14. Initial 5-Task Checklist

1. Initialize GitHub repository and invite team members.
2. Nishant: Implement mock endpoints `/classify`, `/summarize`, `/suggest_reply`.
3. Priyanka: Build a basic inbox UI wired to these endpoints.
4. Both: Prepare labeled dataset (50–200 emails) and define labeling rules.
5. Schedule daily 15-minute stand-ups and weekly review meetings.