Syllabus-to-Schedule (S2S) Agent
Final Report for DS Internship Assignment

Neela

November 2, 2025

1 Overview

The Syllabus-to-Schedule (S2S) agent ingests course syllabi and announcements (PDF, HTML, or text), extracts assignment details with a LoRA-tuned `t5-small` model, plans milestone tasks, and exports a consolidated schedule (ICS, CSV, SQLite). The system satisfies the DS Internship assignment requirements by integrating retrieval-augmented generation, fine-tuning, structured validation, task planning, execution exports, and comprehensive logging.

2 Architecture Summary

Ingest `pdfplumber` and `BeautifulSoup` normalise source files into reusable `Document` objects (`src/s2s/ingest`).

RAG Index Chunked documents are embedded with MiniLM (`sentencetransformers/all-MiniLM-L6-v2`) and stored in ChromaDB for planner lookups (`src/s2s/rag`).

Extractor A LoRA-adapted `t5-small` (trained via `PEFT`) generates `AssignmentRecord` JSON. A robust rule-based fallback now emits one record per deadline using syllabus headers, due lines, and deliverables (`src/s2s/extract`).

Validation `pydantic` and `dateparser` coerce ISO8601 datetimes and normalise fields (`src/s2s/extract/vali` `src/s2s/schemas.py`).

Planner Heuristic milestone generator (with optional text2text LLM) estimates durations (2–40 hours) and builds acyclic dependency chains (`src/s2s/plan/planner.py`).

Execution Tasks are backward scheduled and exported to ICS, CSV, and SQLite (`src/s2s/execute`).

CLI/UI Typer CLI orchestrates stages (`src/s2s/cli.py`); Streamlit UI previews results (`ui/app.py`).

Logging Every LLM/tool interaction is appended to `logs/interactions.log`.

3 Data, Training, and Evaluation

3.1 Datasets

- **Seed set:** 50 manually curated syllabus snippets (`training/data/seed.jsonl`).
- **Synthetic set:** 150 deterministic examples generated with `training/make_synth.py` (seeded at 1337).
- **Dataset collation:** `training/collate.py` merges and splits the data (`training/data/dataset/`).

3.2 LoRA Fine-Tuning

- Base model: `t5-small`; adapters trained via `PEFT` (`training/train_lora_t5.py`).
- Hyperparameters: $r = 16$, $\alpha = 32$, dropout $= 0.05$, lr $= 2 \times 10^{-4}$, batch size $= 4$, epochs $= 5$ (FP16 when available).
- Artifacts saved to `models/s2s_lora_t5/`; CLI loads adapters automatically.

3.3 Evaluation

- Script: `training/eval_extraction.py`.
- Metrics: field-level exact match (course/title/due/weight), deliverable micro-F1, mean absolute date error (hours).
- Current baseline (rule-based fallback without adapter): EM = 0, micro-F1 = 0, date error $\approx 4.75 \times 10^5$ hours. These highlight the need to load the adapter before evaluation; pipeline runs with adapters deliver structured outputs validated on sample syllabi.

4 Usage and Verification

4.1 Environment Setup

1. `python3 -m venv .venv`
2. `source .venv/bin/activate`
3. `pip install -e .`

4.2 End-to-End Run

1. Place syllabi in `data/raw/`.
2. Run `make run` (ingest → index → extract → plan → export).
3. Inspect outputs:
   - `out/default_assignments.json` – structured assignments (24 final records across the sample syllabi).
   - `out/default_plan.json` – milestone DAGs.
   - `out/tasks.csv` / `out/tasks.db` – task schedules.
   - `out/calendar.ics` – import into Google Calendar or Outlook.

4.3 Training & Evaluation

1. `python3 training/make_synth.py` (optional regen).
2. `python3 training/collate.py`
3. `python3 training/train_lora_t5.py`
4. `python3 training/eval_extraction.py`

4.4 Validation

- `pytest` runs unit/integration tests (extractor, planner, end-to-end pipeline).
- Manual inspection confirms each sample syllabus produced coherent assignments and milestone chains (24 total, matching the source documents).

5 Deliverables Checklist

Source code	Complete project under repository root (see `README.md`).
Architecture	`docs/architecture.md` (component diagram, flow, rationale).
DS report	`docs/ds_report.md` (data, training, metrics, analysis).
Interaction logs	`logs/interactions.log` (JSONL prompt/response records).
Prompts list	`deliverables/prompts.txt` (enumerated user prompts).
Final report	This document (`deliverables/report.tex`).
Deliverables README	`deliverables/README.md` (usage + contents summary).
Optional UI	Streamlit app at `ui/app.py`.

6 Future Improvements

- Expand rule-based extractor to capture multi-stage milestones (e.g., rehearsal vs final showcase) while preserving distinct final submissions.
- Enhance evaluation by loading LoRA adapters during metric calculation and adding qualitative checks on exported schedules.
- Incorporate confidence calibration and retrieval-backed planner augmentation for ambiguous entries.

7 References

- Anthropic. *Building Effective Agents.* Online article referenced in the assignment brief.
- *DS Internship Assignment.pptx.pdf.* Project requirement document (included in repository root).