

Snipster Cohort – Syllabus

Program outcome

Build **Snipster**, a full-featured code snippet manager with a clean, real-world architecture:

- Python package + tested core domain
 - CLI tool for managing snippets
 - Repository pattern with `SQLModel` + at least DB implementation (and optional JSON one)
 - FastAPI backend & Streamlit (or NiceGUI) front-end prototype
 - Optional: auth, fuzzy search, integrations, cloud deploy, CI/CD, docs
-

Tech Stack & Practices

- **Python 3 + uv** for envs & packaging
 - **SQLModel** (+ PostgreSQL/SQLite)
 - **pytest** (+ fixtures & coverage)
 - **ruff + mypy (or ty) + pre-commit**
 - **Typer** (CLI) & **Rich** (CLI UX)
 - **FastAPI** (API) & **Streamlit / NiceGUI** (UI prototype)
 - **Git & GitHub** (branches, PRs, collaboration)
 - **GitHub Actions**, PyPI, basic cloud deploy (e.g. [Fly.io](https://fly.io) or Heroku)
-

Pre-work / Intro (on enrollment)

- Welcome & cohort expectations
 - Join group calls & introduce yourself
 - Walkthrough of the Snipster app mindmap (overall architecture)
 - GitHub repo setup, branching & PR workflow (one per week)
 - How we use AI tools responsibly during the cohort
 - “Wins file” – simple system to track progress
-

Week 1 – First Steps & Core Model

Goal: Repo running, tooling in place, first domain model.

- Clone repo, set up project with **uv**
 - Create **Snippet** model with **SQLModel**
 - First **pytest** + coverage run
 - Configure **ruff** + **pre-commit** (and mypy or ty)
 - (Bonus) Alternate constructor with `@classmethod` + basic validation
 - Weekly milestone checklist
-

Week 2 – Architecture & Testing

Goal: Introduce clean architecture via Repository pattern.

- Define `SnippetRepository` interface (ABC)
 - Implement in-memory and DB repositories (`SQLModel`, optional JSON)
 - Apply **Repository pattern** to decouple storage from logic
 - Pytest II: fixtures, shared state, high coverage test suite
 - Add structured exception handling
 - (Bonus) **Alembic** for database migrations
 - Weekly milestone checklist
-

Week 3 – Features & UX

Goal: Turn Snipster into a useful snippet manager.

- Plan & implement features: search, favourites, tags
 - Guidance on scoping & iterating features
 - (Bonus) Use **TDD** for new features
 - (Bonus) Fuzzy search using `difflib`
 - Weekly milestone checklist
-

Week 4 – CLI & Developer Experience

Goal: Wrap the app in a solid CLI and improve DX.

- Use environment variables for config (DB URL, etc.)
 - Build CLI with **Typer** (add/list/get/delete snippets)
 - Use **Rich** for better CLI output (tables, colours, prompts)
 - (Bonus) End-to-end tests for the CLI
 - (Bonus) “Run code” feature (e.g. via piston API or AWS Lambda)
 - Weekly milestone checklist
-

Week 5 – API & Front-End Prototype

Goal: Expose Snipster over HTTP and add a UI.

- Build REST API with **FastAPI** (CRUD endpoints)
 - Prototype UI with **Streamlit** or **NiceGUI** (no JS required)
 - (Bonus) FastAPI testing and/or basic auth
 - (Bonus) Integrations:
 - Export snippets as images using **pybites-carbon**
 - Export as gists / via GitHub API
 - Weekly milestone checklist
-

Week 6 – Deployment, CI & Documentation

Goal: Ship, document and showcase your work.

- Deploy FastAPI app to cloud (e.g. [Fly.io](https://fly.io) or Heroku; optionally use Docker)
- Package and publish CLI to **PyPI** (using uv)
- Add **GitHub Actions** to run tests (and optionally deployment + PyPI push)
- Documentation:
 - Strong `README`
 - Docstrings
 - (Bonus) Sphinx or MkDocs site
- Share your work: PyBites community, social media, resume / LinkedIn
- Final milestone checklist