Engineering Open Source Projects (EOSP)

Series of lectures and seminars by Anton Grishin
To be presented to high school students from CPM in CU campus.
The all course materials in the alchemmist/eosp GitHub repository.

Abstract

This project builds a system to evaluate developer contributions based on GitHub activity, implemented as a Python library, a CLI, and a Telegram bot. Students practice modular design, testing, CI/CD, documentation, and real-world Open Source workflows, focusing on usability and clean, extensible solutions.

What is the goal?

The goal of this project is to build a practical system for assessing developer or team contributions based on GitHub activity. The system includes a Python library to calculate contribution metrics, a CLI for interacting with the library, and a Telegram bot for convenient access.

Who is the teacher?

Anton is a hands-on software developer and Python instructor with over four years of experience, contributing to dozens of repositories. Anton is a finalist of the Russian Championship in Competitive Programming. And winner of the «Science for Life» conference. Author of blog: alchemmist.xyz. More experience in CV.

Why this course?

HR processes and IT management are increasingly complex, making it difficult to evaluate developers across productivity, code quality, skills, and contributions without clear metrics. This course teaches students real-world Open Source practices while providing hands-on experience in designing modular, maintainable, and well-documented systems.

What's the methodology?

The course is practice-driven. From the very first lecture, students receive a concrete task: to design and build a real project. All subsequent lectures, materials, and activities are structured to support this goal, gradually adding the knowledge and tools needed to complete and refine the project.

Learning Outcomes

By the end of the course, students will know:

- How the Open Source development process works in practice (GitHub Flow, issues, pull requests, reviews).
- How to design software systems with clear architecture and separation of concerns (`lib` → `cli` → `bot`).
- How to write meaningful tests and use testing as a foundation for code quality and refactoring.

- How to build and maintain `CI` and `CD` pipelines for testing, releases, and deployments.
- How to develop and publish libraries, CLI tools, and services ready for real users.
- How to document projects properly (`README`, wiki, licenses) to support contributors and users.
- How to work with secrets, automation, and deployment in production-like environments.
- How to present, pitch, and publicly share an Open Source project.

Schedule

`Week 1:`	Course overview, Open Source fundamentals, project idea and GitHub workflow
`Week 2:`	Library-first design, testing mindset, and Test Driven Development in Python
`Week 3:`	CI/CD pipelines, releases, licensing, and introduction to CLI tools
`Week 4:`	CLI development, documentation, licensing decisions, and repository structure
`Week 5:`	Writing effective READMEs and improving project documentation
`Week 6:`	Deployment fundamentals, Docker basics, and production automation
`Week 7:`	Pitch preparation, slide decks, and project presentation
`Week 8:`	Final presentations, project retrospective, and future directions