Add New Dependency – Team Guide

Goal

Keep everyone on the same, reproducible Python environment. Nobody installs one-off packages; all changes are tracked in requirements.txt and enforced by CI.

When to use this

• You need a new library for the project. • requirements.txt changed and you need to sync locally.

Author workflow (you are adding a dependency)

1) Activate the project virtual environment.

```
source .venv/bin/activate    # Windows: .venv\Scripts\activate
```

2) Install the package (pin will be captured by freeze).

```
pip install <package>
# optional: pin a version directly
# pip install <package>==<version>
```

3) Update requirements.txt with the exact resolved versions.

```
pip freeze > requirements.txt
```

4) Commit and push the change.

```
git add requirements.txt
git commit -m "Add <package>"
git push
```

5) Open/Update your PR with a short note like:

```
Adds <package> for <1■line reason>. Team: pull & run pip install -r requirements.txt
```

Teammate workflow (requirements.txt changed)

1) Pull latest changes and activate venv.

```
git pull
source .venv/bin/activate    # Windows: .venv\Scripts\activate
```

2) Install the exact versions from requirements.txt.

```
pip install -r requirements.txt
```

3) Verify tooling still runs:

```
flake8 src tests
mypy src
pytest
```

Why this matters

• Reproducibility: everyone runs identical versions → fewer “works on my machine” bugs. • Reviewability: dependency diffs are visible in PRs. • CI consistency: the same requirements drive local and GitHub Actions runs.

Do & Don’t

DO	Use pip install + pip freeze to update requirements.txt and commit it.
DO	Explain briefly in the PR why the new package is needed.
DO	Remove unused deps later (pip uninstall <pkg> → freeze → commit).
DON'T	pip install packages locally without updating requirements.txt.
DON'T	Edit requirements.txt by hand unless you know exactly why.

Common pitfalls & fixes

• On shared servers (e.g., eceprog), disk quota errors with mypy caches: run *mypy --no-incremental src* or configure *cache_dir = "/tmp/mypy_cache"* in pyproject.toml.

• If install fails due to old pip/setuptools: run *python -m pip install --upgrade pip wheel setuptools*.

• If the venv seems broken: deactivate → remove .venv → recreate → reinstall from requirements.txt.

Quick copy/paste blocks

```
# Add a new dependency (author)
source .venv/bin/activate
pip install <package>
pip freeze > requirements.txt
git add requirements.txt
git commit -m "Add <package>"
git push

# Sync after someone changed requirements.txt (teammate)
git pull
source .venv/bin/activate
pip install -r requirements.txt
flake8 src tests && mypy src && pytest
```