

Lecture 7: Packaging and Automation

CEE 690

Packaging

What is packaging?



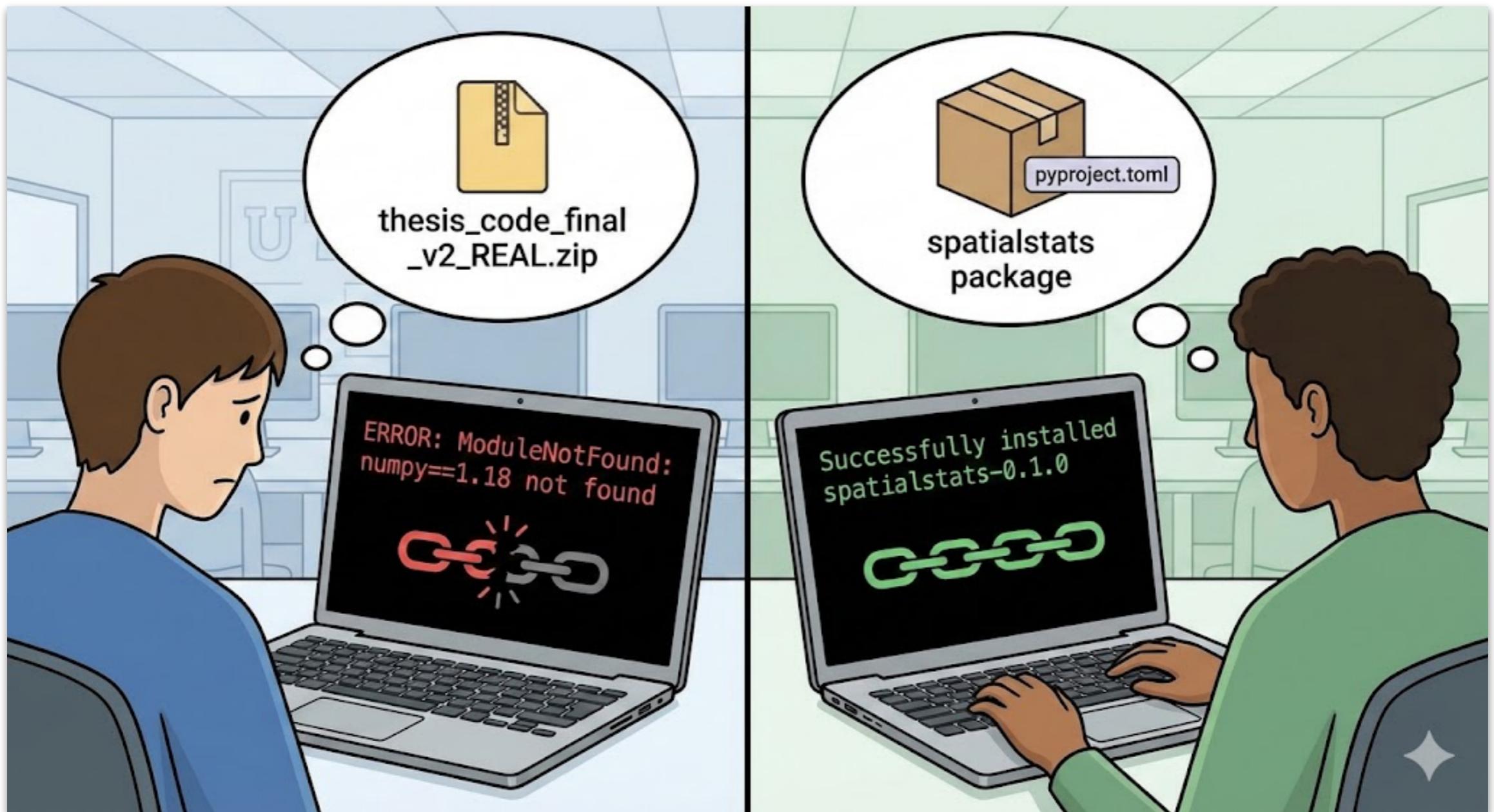
The Script Approach (No Packaging)



The Package Approach (Standardized)

Source: Gemini

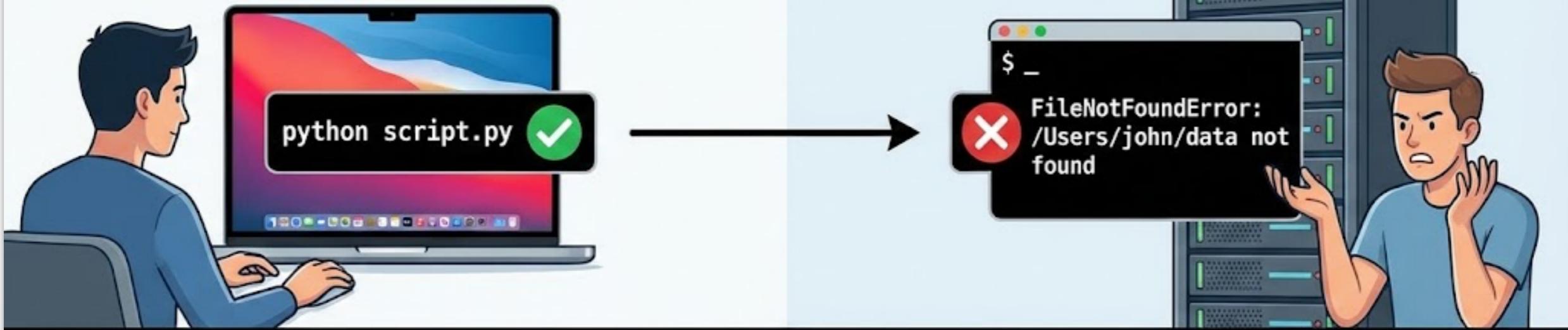
Reason 1: Reproducibility



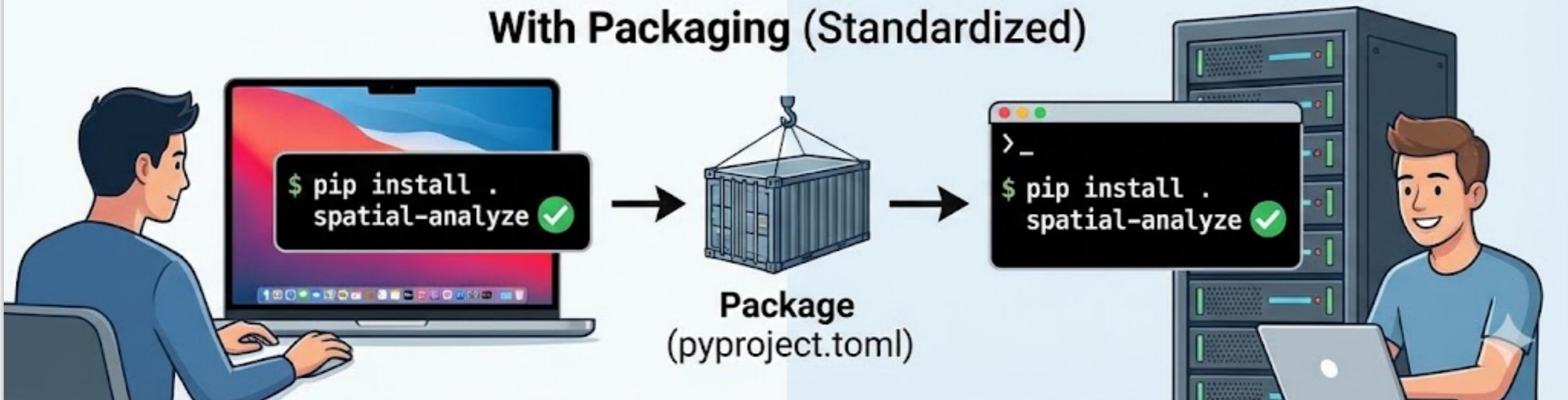
Source: Gemini

Reason 2: Portability

Without Packaging (Raw Script)

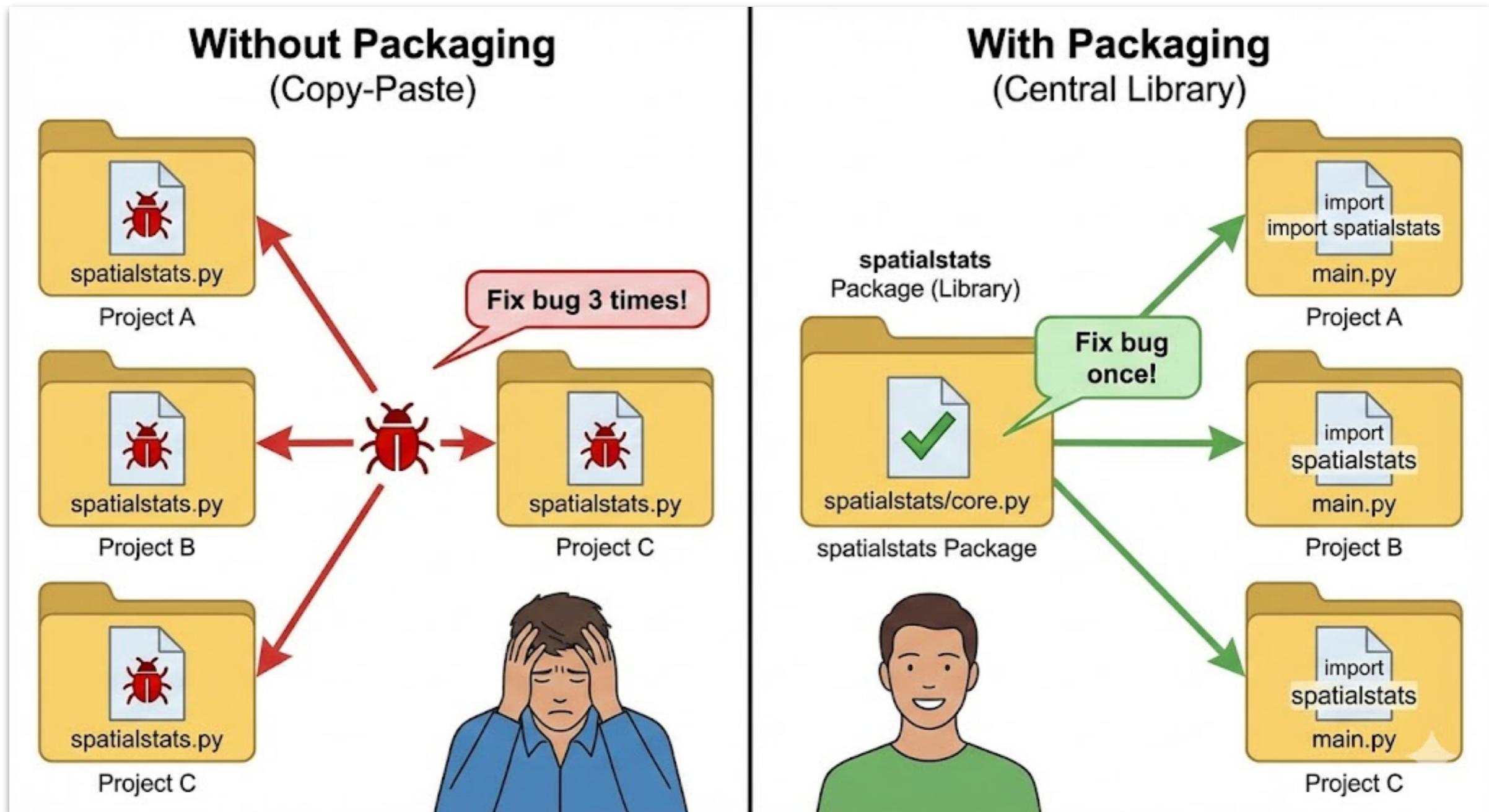


With Packaging (Standardized)



Source: Gemini

Reason 3: Usability



Source: Gemini

Getting ready for packaging



```
(cee690) nc153 $ tree -L 3
.
└── docs
    ├── _build
    │   └── doctrees
    │       └── html
    ├── conf.py
    ├── index.rst
    ├── make.bat
    └── Makefile
└── era_interim_annual_197901_201512_upscaled_subset.nc
└── example.json
└── README.md
└── spatialstats.py
```

Most of this is just organizing it clearly, however, there are few things that we need to discuss



```
(cee690) nc153 $ tree -L 3
.
└── data
    └── era_interim_annual_197901_201512_upscaled_subset.nc
└── docs
    ├── _build
    │   └── doctrees
    │       └── html
    ├── conf.py
    ├── index.rst
    ├── make.bat
    └── Makefile
└── Makefile
pyproject.toml
README.md
└── src
    └── spatialstats
        ├── core.py
        └── __init__.py
└── tests
    └── test_spatialstats.py
```

Getting ready for packaging



```
(cee690) nc153 $ tree -L 3
.
└── docs
    ├── _build
    │   └── doctrees
    │       └── html
    ├── conf.py
    ├── index.rst
    ├── make.bat
    └── Makefile
└── era_interim_annual_197901_201512_upscaled_subset.nc
└── example.json
└── README.md
└── spatialstats.py
```

Most of this is just organizing it clearly, however, there are few things that we need to discuss



```
(cee690) nc153 $ tree -L 3
.
└── data
    └── era_interim_annual_197901_201512_upscaled_subset.nc
└── docs
    ├── _build
    │   └── doctrees
    │       └── html
    ├── conf.py
    ├── index.rst
    ├── make.bat
    └── Makefile
└── Makefile
└── pyproject.toml
└── README.md
└── src
    └── spatialstats
        ├── core.py
        └── __init__.py
└── tests
    └── test_spatialstats.py
```

Getting ready for packaging



```
(cee690) nc153 $ tree -L 3
.
└── docs
    ├── _build
    │   └── doctrees
    │       └── html
    ├── conf.py
    ├── index.rst
    ├── make.bat
    └── Makefile
└── era_interim_annual_197901_201512_upscaled_subset.nc
└── example.json
└── README.md
└── spatialstats.py
```

Most of this is just organizing it clearly, however, there are few things that we need to discuss



```
(cee690) nc153 $ tree -L 3
.
└── data
    └── era_interim_annual_197901_201512_upscaled_subset.nc
└── docs
    ├── _build
    │   └── doctrees
    │       └── html
    ├── conf.py
    ├── index.rst
    ├── make.bat
    └── Makefile
└── Makefile
└── pyproject.toml
└── README.md
└── src
    └── spatialstats
        ├── core.py
        └── __init__.py
└── tests
    └── test_spatialstats.py
```

Pyproject.toml

This is what tells pip “what to do.

```
1 [build-system]
2 requires = ["setuptools>=61.0"]
3 build-backend = "setuptools.build_meta"
4
5 [project]
6 name = "spatialstats"
7 version = "0.1.0"
8 description = "A package for spatial statistics on NetCDF data"
9 authors = [{name = "Nate", email = "nc153@duke.edu"}] # Updated example
10 dependencies = [
11     "numpy",
12     "netCDF4",
13     "matplotlib"
14 ]
15
16 # This maps the command "spatialstats" to the main() function in core.py
17 [project.scripts]
18 spatialstats = "spatialstats.core:main"
19
20 [tool.setuptools.packages.find]
21 where = ["src"]
```

Installing package

```
(cee690) nc153 $ pip install -e .
Obtaining file:///hpc/home/nc153/spatialstats
Installing build dependencies ... done
Checking if build backend supports build_editable ... done
Getting requirements to build editable ... done
Preparing editable metadata (pyproject.toml) ... done
Requirement already satisfied: numpy in /hpc/home/nc153/miniconda3/
0) (2.4.1)
Requirement already satisfied: netCDF4 in /hpc/home/nc153/miniconda
1.0) (1.7.4)
Requirement already satisfied: matplotlib in /hpc/home/nc153/minic
=0.1.0) (3.10.8)
Requirement already satisfied: contourpy>=1.0.1 in /hpc/home/nc153/
lib->spatialstats==0.1.0) (1.3.3)
Requirement already satisfied: cycler>=0.10 in /hpc/home/nc153/min
>spatialstats==0.1.0) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /hpc/home/nc153/
tlib->spatialstats==0.1.0) (4.61.1)
Requirement already satisfied: kiwisolver>=1.3.1 in /hpc/home/nc153/
tlib->spatialstats==0.1.0) (1.4.9)
Requirement already satisfied: packaging>=20.0 in /hpc/home/nc153/n
ib->spatialstats==0.1.0) (25.0)
Requirement already satisfied: pillow>=8 in /hpc/home/nc153/minicor
atialstats==0.1.0) (12.1.0)
Requirement already satisfied: pyparsing>=3 in /hpc/home/nc153/min
```

No need to carry the script around anymore or make sure it is in a certain directory. It is now part of your conda environment.

```
(cee690) nc153 $ python
Python 3.14.2 | packaged by conda-forge | (main, Dec 6 2025, 11:21:58) [GCC 14.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import spatialstats
>>>
```

Pip install . Vs. Pip install -e .

| Feature | <code>pip install .</code> | <code>pip install -e .</code> |
|---------------------|-------------------------------|--------------------------------------|
| Full Command | Regular Install | Editable (Developer) Install |
| Mechanism | Copies files to site-packages | Creates a link to your source folder |
| Updates | Requires reinstall | Instant |
| Speed | Slower (builds wheel) | Fast (just links) |
| Use Case | Production / Final Release | Active Coding / Debugging |

Two valid options...



```
(cee690) nc153 $ tree -L 3
.
└── data
    └── era_interim_annual_197901_201512_upscaled_subset.nc
└── example.json
└── docs
    ├── _build
    │   └── doctrees
    │       └── html
    ├── conf.py
    ├── index.rst
    ├── make.bat
    └── Makefile
└── Makefile
└── pyproject.toml
└── README.md
└── src
    └── spatialstats
        ├── core.py
        └── __init__.py
└── tests
    └── test_spatialstats.py
```



```
(cee690) nc153 $ tree -L 3
.
└── data
    └── era_interim_annual_197901_201512_upscaled_subset.nc
    └── example.json
└── docs
    ├── _build
    │   └── doctrees
    │       └── html
    ├── conf.py
    ├── index.rst
    ├── make.bat
    └── Makefile
└── Makefile
└── pyproject.toml
└── README.md
└── src
    └── spatialstats.py
└── tests
    └── test_spatialstats.py
```

Both options are ok. Why is the first option preferred?

Two valid options...



```
(cee690) nc153 $ tree -L 3
.
└── data
    └── era_interim_annual_197901_201512_upscaled_subset.nc
└── docs
    ├── _build
    │   └── doctrees
    │       └── html
    ├── conf.py
    ├── index.rst
    ├── make.bat
    └── Makefile
└── Makefile
pyproject.toml
README.md
└── src
    └── spatialstats
        ├── core.py
        └── __init__.py
└── tests
    └── test_spatialstats.py
```



```
(cee690) nc153 $ tree -L 3
.
└── data
    └── era_interim_annual_197901_201512_upscaled_subset.nc
└── docs
    ├── _build
    │   └── doctrees
    │       └── html
    ├── conf.py
    ├── index.rst
    ├── make.bat
    └── Makefile
└── Makefile
pyproject.toml
README.md
└── src
    └── spatialstats.py
└── tests
    └── test_spatialstats.py
```

Both options are ok. Why is the first option preferred?

Two valid options...



```
(cee690) nc153 $ tree -L 3
.
└── data
    └── era_interim_annual_197901_201512_upscaled_subset.nc
└── docs
    ├── _build
    │   └── doctrees
    │       └── html
    ├── conf.py
    ├── index.rst
    ├── make.bat
    └── Makefile
└── Makefile
pyproject.toml
README.md
└── src
    └── spatialstats
        ├── core.py
        └── __init__.py
└── tests
    └── test_spatialstats.py
```



```
(cee690) nc153 $ tree -L 3
.
└── data
    └── era_interim_annual_197901_201512_upscaled_subset.nc
└── docs
    ├── _build
    │   └── doctrees
    │       └── html
    ├── conf.py
    ├── index.rst
    ├── make.bat
    └── Makefile
└── Makefile
pyproject.toml
README.md
└── src
    └── spatialstats.py
└── tests
    └── test_spatialstats.py
```

Both options are ok. Why is the first option preferred?

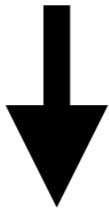
`__init__.py`

```
1 # src/spatialstats/__init__.py
2 from .core import SpatialAnalyzer, get_args
3
4
```

Why make our lives so complicated?

spatialstats executable

```
10 dependencies = [  
11     "numpy",  
12     "netCDF4",  
13     "matplotlib"  
14 ]  
15  
16 # This maps the command "spatialstats" to the main() function in core.py  
17 [project.scripts]  
18 spatialstats = "spatialstats.core:main"  
19
```



```
(cee690) nc153 $ spatialstats --JSON_FILE spatialstats/data/example.json  
Configuration loaded from spatialstats/data/example.json  
Computing the statistics  
Visualizing the data  
Plot saved to output_plot.png  
Saving statistics to output_temporal_stats.nc  
Processing complete.
```

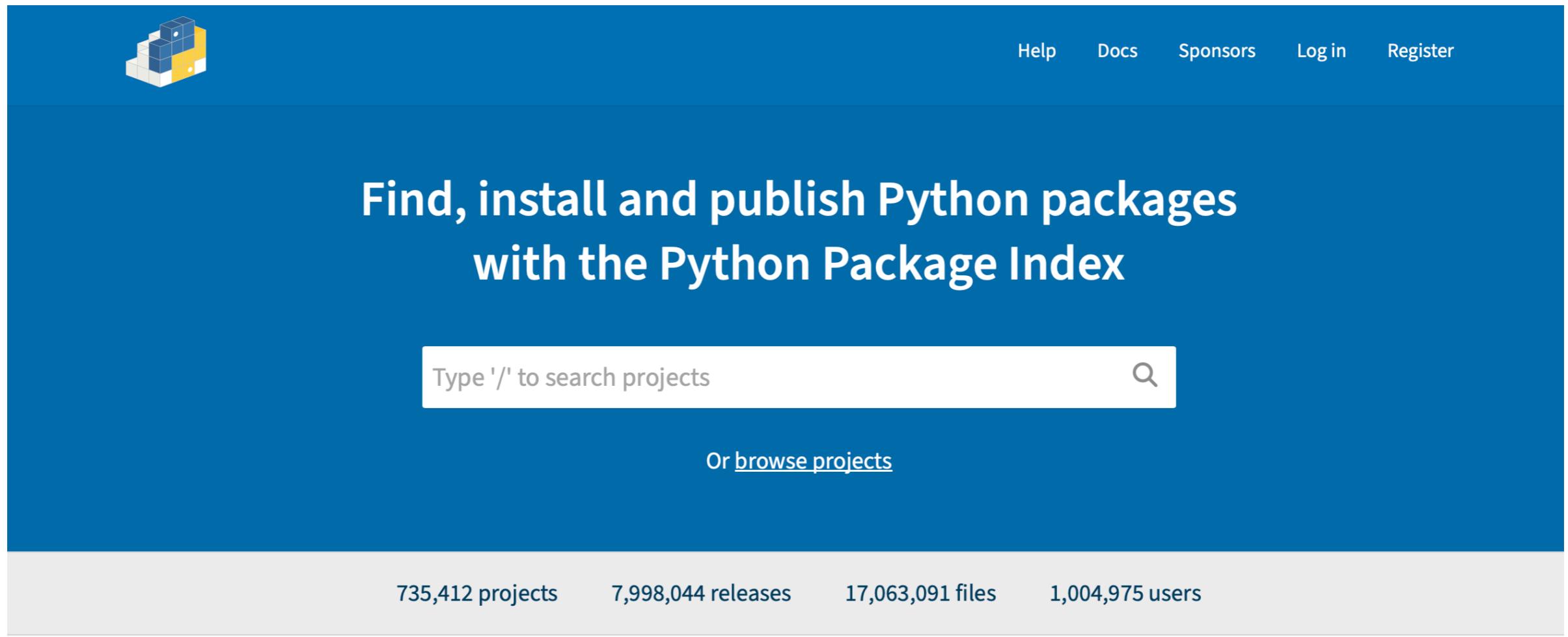
Makefile

```
1 .PHONY: install run clean
2
3 # Install the package in "editable" mode (-e)
4 # This means changes in src/ apply immediately without reinstalling
5 install:
6     pip install -e .
7
8 # Run the tool using the new command we created in pyproject.toml
9 # Note: We point to the data in the new data/ folder
10 run:
11     spatialstats --INPUT_FILE data/era_interim_annual_197901_201512_upscaled_subset.nc
12
13 clean:
14     rm -rf build dist src/*.egg-info
15     find . -type d -name "__pycache__" -exec rm -rf {} +
16     find . -type f -name "*.pyc" -delete
```

```
(cee690) nc153 $ make install
pip install -e .
Obtaining file:///hpc/home/nc153/spatialstats
Installing build dependencies ... done
Checking if build backend supports build_editable ...
Getting requirements to build editable ... done
Preparing editable metadata (pyproject.toml)
Requirement already satisfied: numpy in /hpc/h0 (2.4.1)
Requirement already satisfied: netCDF4 in /hpc
```

```
(cee690) nc153 $ make clean
rm -rf build dist src/*.egg-info
find . -type d -name "__pycache__" -exec rm -rf {} +
find . -type f -name "*.pyc" -delete
```

Ready for Pypi



The screenshot shows the PyPI homepage with a blue header bar. On the left is a logo of a 3D cube composed of smaller cubes in white, blue, and yellow. On the right are navigation links: Help, Docs, Sponsors, Log in, and Register. Below the header is a large white search bar containing the placeholder text "Type '/' to search projects" and a magnifying glass icon. To the right of the search bar is the text "Or [browse projects](#)". At the bottom of the page, a light gray footer bar displays statistics: 735,412 projects, 7,998,044 releases, 17,063,091 files, and 1,004,975 users.

Find, install and publish Python packages
with the Python Package Index

Type '/' to search projects

Or [browse projects](#)

735,412 projects 7,998,044 releases 17,063,091 files 1,004,975 users

It's better to avoid adding anything
to Pypi. Use GitHub instead

```
(cee690) nc153 $ pip install git+https://github.com/chaneyn/spatialstats.git
```

Automation

Can something/someone else run all
checks for me?

Continuous integration

Before code is allowed to be merged on the main branch, it should pass these tests:

Continuous integration

Before code is allowed to be merged on the main branch, it should pass these tests:

1. **Linting** - Is the code clean?

Continuous integration

Before code is allowed to be merged on the main branch, it should pass these tests:

- 1. Linting** - Is the code clean?
- 2. Unit Tests** - Does the math work?

Continuous integration

Before code is allowed to be merged on the main branch, it should pass these tests:

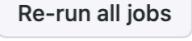
- 1. Linting** - Is the code clean?
- 2. Unit Tests** - Does the math work?
- 3. Installation** - Does pip install actually work?

Automation: GitHub Actions

| | | | | |
|---|--|---|-------------------------|--|
|  Nathaniel Chaney | Added script for GitHub actions | ✓ | 18207a0 · 4 minutes ago |  27 Commits |
|  .github/workflows | Added script for GitHub actions | | 4 minutes ago | |
|  data | cleaned up some issues with the Makefile | | 20 minutes ago | |

← Python Package Tests

 **Added script for GitHub actions #1**

| | |
|---|---|
|  Summary |  |
| All jobs | |
|  build-and-test | |
|  Usage | |
|  Workflow file | |

build-and-test
succeeded 2 minutes ago in 25s

| | |
|--|-----|
| >  Set up job | 1s |
| >  Checkout Code | 2s |
| >  Set up Python | 0s |
| >  Install Dependencies | 15s |
| >  Run Unit Tests | 3s |
| >  Post Set up Python | 0s |
| >  Post Checkout Code | 0s |
| >  Complete job | 0s |

Example for spatialstats

```
(cee690) nc153 $ vi .github/workflows/tests.yml
```

```
1 name: Python Package Tests
2
3 on: [push, pull_request]
4
5 jobs:
6   build-and-test:
7     runs-on: ubuntu-latest
8
9   steps:
10    - name: Checkout Code
11      uses: actions/checkout@v4
12
13    - name: Set up Python
14      uses: actions/setup-python@v4
15      with:
16        python-version: '3.10'
17
18    - name: Install Dependencies
19      run: |
20        python -m pip install --upgrade pip
21        pip install .[test] # Installs the package + test dependencies
22
23    - name: Run Unit Tests
24      run: |
25        python -m unittest discover tests
```