Python Packaging Project Setup Guide – wellsfargo\_1pii\_kpi\_automation

# Introduction

This document provides guidelines for setting up a Python packaging project in a Wells Fargo internal context, using the `wellsfargo\_1pii\_kpi\_automation` project as an example. It covers two standard methods of packaging a Python project:  
- Setuptools with setup.cfg + pyproject.toml – the traditional packaging approach using Setuptools.  
- Poetry with pyproject.toml – a modern all-in-one approach using Poetry.

We compare these methods, highlight why mixing both systems in one project is discouraged, and present recommended configurations for each. Explanations are provided for each configuration section (metadata, options, dependencies, build system, etc.), along with notes on using Artifactory as a private PyPI repository. Finally, we outline best practices regarding versioning (semantic versioning), CI/CD pipeline compatibility, and dependency locking for reproducible environments.

## Is Mixing Setuptools and Poetry Allowed?

No. Mixing Setuptools (setup.cfg) and Poetry (pyproject.toml) in the same project is discouraged. Each tool expects to manage your project independently, and using both can lead to inconsistent builds and conflicting metadata.

## Setuptools Recommended Configuration

setup.cfg

[metadata]  
name = wellsfargo\_1pii\_kpi\_automation  
version = 1.1.0  
description = A Python project for KPI automation  
  
[options]  
packages = find:  
install\_requires =  
 jira  
 psycopg2-binary  
 pymssql  
 pytest  
 pytest-cov  
 coverage  
zip\_safe = True  
include\_package\_data = True  
python\_requires = >=3.11  
  
[options.package\_data]  
\* = README.md  
  
[options.packages.find]  
where = .

pyproject.toml

[build-system]  
requires = ["setuptools", "wheel"]  
build-backend = "setuptools.build\_meta"

## Poetry Recommended Configuration

pyproject.toml

[tool.poetry]  
name = "wellsfargo\_1pii\_kpi\_automation"  
version = "1.1.0"  
description = "A Python project for KPI automation"  
authors = ["K154844 <mastinder.singh@wellsfargo.com>"]  
readme = "README.md"  
  
[tool.poetry.dependencies]  
python = ">=3.11,<3.12"  
jira = "^3.4.0"  
psycopg2-binary = "^2.9.5"  
pymssql = "^2.3.0"  
  
[build-system]  
requires = ["poetry-core"]  
build-backend = "poetry.core.masonry.api"

## Best Practices Summary

- Stick to one tool: Setuptools or Poetry.  
- Use semantic versioning (MAJOR.MINOR.PATCH).  
- Keep lock files under version control for reproducibility.  
- Avoid duplicate metadata in multiple config files.  
- Use CI/CD to test, build, and publish artifacts.