Supporting Python-Based Research Projects

For Principal Investigators, Supervisors & Managers

A practical guide to roles, responsibilities, and self-assessment tools for research projects that leverage Python workflows — from data analysis to reproducible simulations.

Part I: Roles & Responsibilities

Successful Python-based research depends on the right support at every stage. This section outlines the core responsibilities of principal investigators (PIs), supervisors, project managers, and research support staff.

Roles at a Glance

Role	Stage	Key Task
👩‍🔬 **Principal Investigator**	Grant Application	Build scalable, reproducible workflows
👩‍🏫 **Supervisor – Early PhD**	Project Kickoff	Establish good coding & data habits
👨‍🎓 **Supervisor – Late PhD**	Pre-Submission	Polish code for sharing & citation
🎓 **Supervisor – Master's**	Thesis Development	Scope realistically & deliver a clean repo
👩‍💼 **Division/Project Manager**	Program Oversight	Provide standards, training & infrastructure

Each role contributes uniquely to the success of the project. Below are detailed descriptions of the responsibilities and support expectations for each.

👩‍🔬 Principal Investigator (PI) – Grant Application Stage

Key Task: Enable scalable, reproducible, and fundable Python-based research workflows.

Top Responsibilities:

1. **Draft a Data Management Plan**: Plan for structured, accessible data including formats, sources, backup strategies, and long-term storage solutions.
2. **Budget for Technical Staff & Compute**: Ensure funding proposals include adequate resources for programming effort, compute environments, and RSE support.
3. **Embed Open Science Goals**: Set expectations for sharing code and data early. Identify target platforms such as GitHub, Zenodo, or JOSS.
4. **Define Team Roles Early**: Avoid ambiguity by assigning ownership of coding, testing, documentation, and infrastructure tasks.
5. **Recognize Infrastructure as Output**: Treat scripts, pipelines, and tools as valid

👩‍🏫 Supervisor – Early-Stage PhD

Key Task: Establish foundational habits and workflows.

Top Responsibilities:

1. **Clarify Python's Purpose**: Discuss Python's role in the research—data cleaning, modeling, visualizations—and match expectations accordingly.
2. **Provide Onboarding Resources**: Recommend accessible learning material and domain-specific examples to shorten the learning curve.
3. **Set Coding Standards**: Introduce basic best practices such as naming conventions, consistent folder layout, and commenting.
4. **Co-create a Project Repo**: Start the research with a working, version-controlled template to encourage reproducibility.
5. **Diagnose Common Blockers**: Help students differentiate between programming issues and theoretical confusion, reducing frustration and delays.

👩‍🎓 Supervisor – Late-Stage PhD

Key Task: Help finalize the research product with shareable, citable code.

Top Responsibilities:

1. **Guide Packaging**: Help students prepare their work for archiving and citation, including exporting notebooks, writing documentation, and assigning DOIs.
2. **Prioritize Refactoring Time**: Allocate time for cleaning up code, reorganizing functions, and removing redundant or experimental files.
3. **Define Code Value**: Clarify how the code supports the thesis—whether it's a deliverable or a means to an end.
4. **Encourage Reproducibility**: Ensure results can be regenerated using provided scripts and clear instructions.
5. **Support Publication Readiness**: Assist with making the repository public, licensing, and writing metadata or README files.

🎓 Supervisor – Master's Thesis

Key Task: Scope the project realistically and guide toward a clean final product.

Top Responsibilities:

1. **Avoid Overengineering**: Keep the student focused on solving the research problem without adding unnecessary complexity.
2. **Provide Templates**: Share lightweight starter code and data that reflect good structure and minimal viable functionality.
3. **Teach the Basics**: Reinforce the importance of clear variable names, inline comments, and using functions to avoid repeated code.
4. **Require a Final Repo**: Expect a deliverable GitHub repo that includes a README, instructions, and cleaned datasets.
5. **Time Guidance**: Remind students that data cleaning and debugging often take longer than expected, and should be accounted for early.

👨‍💼 Division/Project Manager

Key Task: Support multiple projects through shared infrastructure and policy.

Top Responsibilities:

1. **Create Standards**: Define coding norms for the division including naming, formatting, and documentation guidelines.
2. **Support GitHub/GitLab Orgs**: Provide access and oversight for centralized repositories and collaborative workflows.
3. **Train Researchers**: Offer internal courses or onboarding materials for students and staff new to programming.
4. **Institutionalize Documentation**: Require teams to document handover procedures, workflow diagrams, and dependencies.
5. **Enable Collaboration**: Encourage reuse by rewarding shared solutions, modular scripts, and cross-project contributions.

Optional Roles: RSEs, Data Stewards, Research Assistants

- 👨‍💻 **Research Software Engineer (RSE):**
  - Provide expertise on software design, testing frameworks, performance, and continuous integration.
  - Ensure code is scalable, modular, and production-grade when needed.
- 📊 **Data Steward / Data Manager:**
  - Handle acquisition, conversion, storage, and ethical/data policy compliance.
  - Work to make data FAIR (Findable, Accessible, Interoperable, Reusable).
- 👨‍💻 **Research Assistant / Collaborator:**
  - Contribute scripts, exploratory notebooks, or documentation.
  - Communicate blockers and coordinate closely with leads to align on tasks.

Part II: Self-Assessment Checklists

Use the tables to reflect on project readiness. Tick ✓ or ✗, and jot brief notes if needed.

1. Principal Investigator (PI)

Question	✓ Yes	✗ No	? Not Sure
Included a clear Data Management Plan?	☐	☐	☐
Budgeted time & resources for coding, documentation, and testing?	☐	☐	☐
Justified the need for RSEs/HPC resources?	☐	☐	☐
Specified expected data formats and sources?	☐	☐	☐
Embedded open-science or reproducibility deliverables (JOSS, Zenodo)?	☐	☐	☐
Defined roles around code development, review, and maintenance?	☐	☐	☐

2. Supervisor – Early PhD

Question	✓ Yes	✗ No	? Not Sure
Clarified Python's role (analysis vs. pipeline)?	☐	☐	☐
Provided domain-specific Python learning resources?	☐	☐	☐
Shared a repo/folder template?	☐	☐	☐
Introduced basic coding practices (README, modularity, comments)?	☐	☐	☐
Can distinguish blocker types (syntax vs. concept)?	☐	☐	☐
Discussed reproducibility vs. "just working"?	☐	☐	☐
Encouraged peer-sharing of notebooks?	☐	☐	☐

3. Supervisor – Late PhD

Question	✓ Yes	✗ No	? Not Sure
Code reviewed/refactored for clarity and reuse?	☐	☐	☐
README, docstrings, or user guide present?	☐	☐	☐
Planned code/data deposits (GitHub release, Zenodo DOI)?	☐	☐	☐
Considering a JOSS or methods paper submission?	☐	☐	☐
Time allocated for final code/data polishing?	☐	☐	☐
Reflects strong research-software practices?	☐	☐	☐

4. Supervisor – Master's Thesis

Question	✓ Yes	✗ No	? Not Sure
Scope of Python work matches thesis timeline?	☐	☐	☐
Starter templates provided?	☐	☐	☐
Importance of data structure emphasized?	☐	☐	☐
Final repo with README required?	☐	☐	☐
Code quality reviewed in assessment?	☐	☐	☐
Pipeline documentation guided?	☐	☐	☐

5. Division/Project Manager

Question	✓ Yes	✗ No	? Not Sure
Central GitHub/GitLab org exists?	☐	☐	☐
Code templates/folders available?	☐	☐	☐
Internal Python training supported?	☐	☐	☐
RSEs/technical roles recognized in budgets?	☐	☐	☐
Documentation for handover of code/data?	☐	☐	☐
Publishing shared tools/datasets encouraged?	☐	☐	☐

6. Research Assistant / Collaborator

Question	✓ Yes	✗ No	? Not Sure
Understand project's reproducibility expectations?	☐	☐	☐
Using version control (Git)?	☐	☐	☐
Documenting scripts/pipelines with comments or markdown?	☐	☐	☐
Asking for help on data or analysis clarifications?	☐	☐	☐
Flagging unclear/inconsistent data to the team?	☐	☐	☐

Appendix: Templates & Resources

- How to structure a repository for a python project
- Sample README.md
- JOSS Submission Guidelines
- Zenodo Deposits