



Project Documentation Guide

For Computational Biology Projects
(2-page printable guide)

Why Documentation Matters

1. What is Project Documentation?

Project documentation is a structured collection of information that describes the *goals, methods, tools, data, code, and results* of a project. It explains **what was done, how, and why**—making your work reusable and understandable.

2. Why Is It Important?

Reproducibility

- In computational biology, reproducibility is key.
- Others should be able to **replicate** your results from your documentation alone.

Collaboration

- Helps team members understand the project quickly.
- Avoids miscommunication or duplicated work.

Knowledge Retention

- Future you (or another researcher) can pick up where you left off—even months later.

Publication & Review

- Clean documentation makes your project easier to convert into a **paper, report, or thesis**.
- Journals and grant reviewers often require detailed methods and code availability.

Code Reuse & Sharing

- Well-documented code can be reused in future projects or shared with the broader scientific community.

Page 2: How to Document a Computational Biology Project

1. Project Structure (Recommended Folders)

```
/project-name
├── README.md           # Overview of the project
├── environment.yml      # Conda or pip requirements
├── data/               # Raw and processed data
├── notebooks/          # Jupyter notebooks
├── scripts/            # Python/R scripts
├── results/            # Output data/plots
├── docs/               # Extended documentation
└── LICENSE / .gitignore # Optional
```

2. Essential Documentation Files

README.md (*Always include*)

- Project title and brief description
- Authors and collaborators
- Project goals
- How to run it (e.g., script or notebook)
- Required dependencies

environment.yml OR requirements.txt

- List of software/packages used (e.g., numpy, biopython, matplotlib)
- Enables others to recreate your computational environment.

docs/ folder (*Optional, for detailed methods*)

- Extended explanation of:
 - Data preprocessing
 - Statistical tests or models used
 - Algorithms or bioinformatics tools applied
 - Interpretations of results
-

3. Tips for Good Documentation

- ✓ Use **clear and simple language**
 - ✓ Add **comments** in your code and notebooks
 - ✓ Keep a **CHANGELOG** for updates
 - ✓ Use **Markdown** for clean formatting
 - ✓ Include **citations** to tools, datasets, or papers
 - ✓ If using Jupyter notebooks, add text cells to explain steps
-

Example: Documenting a Genomic Data Analysis

```
# Gene Expression Analysis of Lung Cancer Samples

## Project Overview
This project identifies differentially expressed genes between tumor and
normal lung tissue using RNA-seq data from TCGA.

## Tools Used
- Python 3.9
- Pandas, Seaborn, DESeq2, Biopython

## How to Run
1. Clone the repo
2. Install dependencies: `conda env create -f environment.yml`
3. Run analysis: `python scripts/analyze_expression.py`

## Data
- Raw data from TCGA (see `data/README.md`)
```
