Process Signals Dashboard – Cheatsheet (Python, Bash, Git, CI)

A compact reference for your Streamlit + data processing workflow: structure, debugging, caching, tests, and CI.

1. Project Structure

- app.py – Streamlit UI and orchestration (gating with st.stop()).
- src/io_excel.py – Excel ingest, header discovery, numeric cleaning.
- src/processing.py – Moving averages and FFT utilities.
- src/plotting.py – Matplotlib (PNG) + Plotly (interactive) figures.
- tests/ – pytest unit tests; keeps regressions from sneaking in.
- .github/workflows/tests.yml – CI pipeline (GitHub Actions).

2. Debugging Workflow (VS Code)

- Breakpoints stop only in Debug mode (F5). Running normally ignores them.
- Core controls: Step Over F10, Step Into F11, Continue F5.
- Use a local harness (e.g., debug_runner.py) to debug without Streamlit reruns.
- Watch df.columns, NaN counts, array lengths, and time monotonicity.

3. Moving Averages (MA)

- Centered SMA: past+future; no phase shift; not causal.
- Trailing SMA: current+past; causal but introduces lag.
- EMA: causal, smoother, less lag than trailing SMA.
- Edge padding avoids artificial drops at boundaries.

4. FFT Sanity Checks

- FFT assumes monotonic, roughly uniform time (check np.diff(t) > 0).
- Sampling rate fs ≈ 1 / median(dt); Nyquist ≈ fs/2.
- Large low-frequency spikes often indicate drift/trend.

5. Streamlit Mental Model

- App reruns top-to-bottom on every interaction.
- Guard expensive steps with gating and caching.
- Cache I/O and preprocessing; avoid caching plots unless needed.

6. Python Essentials for Robust Projects

- pathlib.Path for portable paths.
- typing and type hints for readability and IDE help.
- dataclasses for small data containers.
- logging instead of print; try/except + raise for predictable failures.

7. Bash / CLI Commands You Will Use Constantly

- Create venv: python -m venv .venv
- Activate (PowerShell): .\.venv\Scripts\Activate.ps1
- Install deps: pip install -r requirements.txt
- Run tests: pytest -q; Run app: streamlit run app.py

8. Git Commands (Team-Ready Workflow)

- git checkout -b feature/my-change
- git add . && git commit -m "message"
- git push -u origin feature/my-change
- Protect main with CI + PR reviews.

9. CI Basics (GitHub Actions)

- Workflows run on push and pull requests.
- CI should run pytest and block merges when red.
- Branch protection enforces quality gates.

10. Refactoring (Key Learnings)

- Refactor structure without changing behavior; tests must stay green.
- Replace tuples and magic indices with domain objects (e.g., dataclasses).
- Refactor in small, reversible steps to reduce risk.

Mindset shift: from “does it run?” to “is it correct, testable, and robust?”