Process Guide: AI-first Project and Bug Management

Paul Doyle

2025-12-22
Version 1.7

Overview

How to use AI_first for process, project management, and bug management. Use support views for context, personas for decisions, and stage action files to capture evidence.

Activation (First Session)

- Copy the `AI_first/` directory into your repo root.
- Open `AI_first/ui/index.html` as the bundle landing page.
- Open `AI_first/ui/PM.html` and `AI_first/ui/bugmgmt_issues.html` to identify active work and open issues.
- Choose the active project, bug, or action you are taking on.
- Create or update the stage action file at `AI_first/projects/<project>/phases/phase<NN>/actions/<project>_phase<NN>_stage_<name>_action.md`.
- Record decisions and updates in the stage action file as you go.

Context Launch Prompt

Use this before selecting any persona so every session starts from the same entry point. Use the Context Launch prompt in `AI_first/ui/index.html` or `AI_first/ui/process_guide.html` and copy it into your AI session.

- `Context Launch: Review AI_first/ui/index.html, AI_first/ui/PM.html, AI_first/ui/bugmgmt_issues.html, AI_first/docs/process.md, AI_first/docs/projectplan.md, and AI_first/docs/project_wide_docs/personas.md. Identify the active project, phase, and stage action file; list open bugs; summarize current status; and ask which persona to activate next (default order: Project Creator/Owner -> Project/Process Manager -> Developer -> QA Lead; add optional personas from AI_first/docs/project_wide_docs/personas.md only when scope triggers them). If no stage action file exists, propose the file path using the naming convention.`

Repo Layout

- Support tooling lives under `AI_first/` (process docs, Bug Management (BugMgmt), UI reports).
- Project planning docs live under `AI_first/projects/`.
- Add product workspace directories in downstream repos as needed (outside `AI_first/`).

Optional Automation

Use scripts only when you want to regenerate UI views. The day-to-day workflow is AI-first and doc-driven.

- `python3 AI_first/scripts/render_docs.py` to render markdown into `AI_first/ui/docs/`.
- `python3 AI_first/scripts/watch_docs.py` to auto-render while editing.
- `python3 AI_first/scripts/init_project.py --project <project> --prefix <PREFIX> --owner "Name"` to scaffold a project.
- `python3 AI_first/scripts/issues.py list --format json --output AI_first/bugmgmt/exports/json/bugmgmt_issues.json` to regenerate JSON.
- `python3 AI_first/scripts/issues.py list --format html --output AI_first/ui/bugmgmt_issues.html` to regenerate the HTML report.

Notes

- Keep content PII-free unless explicitly required.
- Prefer local assets and deterministic outputs.
- Document versioning: 1.x; increment the minor number for each change to this guide.
- Bug Management (BugMgmt) is optional; remove or ignore `AI_first/bugmgmt/`, `AI_first/ui/bugmgmt_issues.html`, and `AI_first/scripts/issues.py` if you do not need it, and remove the Bug Management link from navigation if desired.
- Run scripts from the repo root; use `python3` (or `python` if mapped to Python 3).
- Update this guide when the process changes.

AI-first Loop

- Review Context: ask Codex to review the AI_first context before work begins.
  Output: current phase and active stage action file summary.
- Activate Personas: use personas to define actions, acceptance criteria, and risks before implementation.
  Output: decisions recorded in the stage action file.
- Plan and Update: update the stage action file and phase docs to reflect the next actions.
  Output: updated phase definition/action plan and stage action log.
- Execute: execute actions, validate outcomes, and record results.
  Output: validation notes and completion status logged.

AI Prompt Examples

- Review Context: `Context Launch: Review AI_first/ui/index.html, AI_first/ui/PM.html, AI_first/ui/bugmgmt_issues.html, AI_first/docs/process.md, AI_first/docs/projectplan.md, and AI_first/docs/project_wide_docs/personas.md.`

`Identify the active project, phase, and stage action file; list open bugs; summarize current status; and ask which persona to activate next (default order: Project Creator/Owner -> Project/Process Manager -> Developer -> QA Lead; add optional personas from AI_first/docs/project_wide_docs/personas.md only when scope triggers them). If no stage action file exists, propose the file path using the naming convention.`

- Activate Personas: `Activate Project/Process Manager Persona. Review the active stage action file and propose 3 next actions with acceptance criteria and risks.`
- Plan and Update: `Draft updates for the stage action file and related project docs. Note next steps and owners.`
- Execute: `Act as Developer Persona. Implement the chosen action, summarize changes, and draft closure notes for the stage action file (and Bug Management entry if applicable).`

Artifact Map (Source of Truth)

This map mirrors the AI-first Loop and points to the source-of-truth artifacts for each step.

- Review Context: `AI_first/ui/PM.html`, `AI_first/ui/bugmgmt_issues.html`, `AI_first/docs/process.md`, `AI_first/docs/projectplan.md`.
- Activate Personas: `AI_first/docs/project_wide_docs/personas.md`, `AI_first/docs/process.md`. Record decisions in the stage action file.
- Plan and Update: `AI_first/docs/templates/phase_definition_template.md`, `AI_first/docs/templates/phase_action_plan_template.md`, `AI_first/docs/templates/action_stage_template.md`.
- Execute: `AI_first/ui/PM.html` (and Bug Management when work is bug-driven). Log validation notes in the stage action file.

Personas and Windows

Default to three terminals or windows: Project Creator/Owner (switching to Project/Process Manager once the project is defined), Developer, and QA. Keep each perspective visible while AI drafts and you validate the work.

- Terminal 1 (Project Creator/Owner then Project/Process Manager): define the project setup, then review the project plan, active phase docs, and action plan in one view; log decisions and keep project actions aligned.
- Terminal 2 (Developer): work on implementation, track local changes, and validate results; update stage action files as work completes.
- Terminal 3 (QA): draft and verify bug entries, confirm reproduction steps, and log test notes; use Bug Management to track open and closed issues.

AI Prompt Examples

- **Project Creator/Owner Persona**:
  `Activate Project Creator/Owner Persona. Confirm project name, purpose, owner, prefix, and initial phases to create.`

- **Project/Process Manager Persona**:
  `Activate Project/Process Manager Persona. Select the active project and phase, review the stage action file, and continue from the last update with 3 next actions and acceptance criteria.`
- **Developer Persona**:
  `Activate Developer Persona. Select the active project and phase, continue from the last update, and carry out the actions already identified or fix a logged bug. Summarize changes and draft updates for the stage action file and bug entry.`
- **QA Lead Persona**:
  `Activate QA Lead Persona. Select the active project and phase, continue from the last update, and either log a bug or outline a test automation suite. Draft validation notes for the stage action file or bug entry.`

Persona Responsibilities

- **Project Creator/Owner Persona**: establish project name, purpose, owner, and prefix; define initial phases and files to create.
- **Project/Process Manager Persona**:
  define phases and actions in `AI_first/docs/projectplan.md` and phase docs; confirm scope and acceptance criteria; use `AI_first/ui/PM.html` as the portfolio view.
- **Developer Persona**: execute the active stage action file; analyze root cause and propose fixes in Bug Management; update status as work moves to `in_progress` and `closed`.
- **QA Lead Persona**: draft bug entries with AI and confirm reproduction steps; validate fixes and update notes for closed issues; use `AI_first/ui/bugmgmt_issues.html` for audit visibility.

Optional Personas (Scope-triggered)

Use optional personas only when the scope or risk warrants specialized review. See `AI_first/docs/project_wide_docs/personas.md` for full prompts.

- **Product Manager**: define acceptance criteria and scope boundaries.
- **Repository Steward**: protect the template structure and AI_first/project separation.
- **Docs Expert**: keep docs accurate, consistent, and linked.
- **UI/Accessibility**: review usability, accessibility, and style consistency.
- **Bug Triage**: validate IDs, required fields, and owners after bug updates.
- **Automation/Tooling**: list commands needed to refresh exports and rendered docs.
- **Architect**: identify cross-boundary risks and acceptance conditions.
- **Security**: flag threats, secrets/PII handling, and mitigations.
- **Ops/Observability**: note runbook gaps, monitoring needs, and rollback notes.
- **Performance/Cost**: note performance or cost risks when scale matters.
- **DBA**: note data integrity and schema risks when a database is introduced.

Project Planning & Delivery

Keep project planning AI-first: define phase intent, plan actions, then deliver and review outcomes while PM.html mirrors progress. Use the appropriate persona to draft, execute, and validate

updates. Primary artifacts: `phase_definition.md`, `action_plan_phaseNN.md`, the active stage action file, and `AI_first/ui/PM.html`.

- Define Phase: update phase goals, scope, and acceptance criteria. Output: refreshed phase definition.
- Plan Actions: break work into actions in the action plan; create or update the active stage action file. Output: actionable plan and stage log.
- Deliver & Review: execute actions, validate outcomes against acceptance criteria, and update PM.html and next actions.

AI Prompt Examples

- Define Phase: `Select the active project and phase, continue from the last update, and draft updates to the phase definition with goals, scope, and acceptance criteria.`
- Plan Actions: `Using the current phase definition, draft or update the action plan and the active stage action file with next actions, owners, and checkpoints.`
- Deliver & Review: `Summarize delivery progress and validation results, update the stage action file with outcomes, and list the next actions to reflect in PM.html.`

Bug Management Workflow

Keep bugs AI-first: capture the issue, work the fix, and confirm closure with evidence. Use the appropriate persona to draft, execute, and validate updates. Primary artifacts: `issues.jsonl`, the active stage action file, and `AI_first/ui/bugmgmt_issues.html`.

- Open: draft a clear bug report with reproduction steps and impact; confirm summary and severity before logging. Output: new entry in `issues.jsonl`.
- Work: analyze root cause and propose a fix plan; track progress and decisions in the stage action file. Output: status updated as work moves forward.
- Close: record closure notes and validation results; reflect outcomes in project actions if needed. Output: bug closed with evidence recorded.

AI Prompt Examples

- Open: `Draft a bug entry with required fields, clear reproduction steps, and impact notes. Suggest severity and summary for review.`
- Work: `Given this bug entry, analyze root cause, propose a fix plan, and draft updates for the stage action file and bug status.`
- Close: `Summarize fix outcomes, draft closure notes, and list any project plan updates needed.`