

# Process Guide: Bug Management and Project Management

Vibe Template Team

2025-12-22

## Purpose

This guide explains how to use this repository for bug management and project management. It also outlines how to start a new project repo from this template.

## Quick Start

1. Read the process and personas: `support/docs/process.md`, `support/docs/project_wide_docs/personas.md`.
2. Review the project plan: `support/docs/projectplan.md`.
3. Start the doc watcher for automatic updates while editing (from the repo root):
  - `python3 support/scripts/watch_docs.py`
4. Update issues in `support/bugmgmt/issues/issues.jsonl` and regenerate exports:
  - `python3 support/scripts/issues.py list --format json --output support/bugmgmt/exports/json/bugmgmt_issues.json`
  - `python3 support/scripts/issues.py list --format html --output support/ui/bugmgmt_issues.html`
5. Open `support/ui/PM.html` to review project status and key links.

## Repo Layout

- Support tooling lives under `support/` (process docs, BugMgmt, UI reports).
- Project planning docs live under `projects/`.
- Product work goes under `src/`, `tests/`, `config/`, `data/`, `assets/`, `scripts/`, `docs/`.

## Multi-Context Workflow

- Keep three windows open: the project plan, the active phase definition/action plan, and the current stage action file.
- Keep personas visible and log decisions in the stage action file as you work.
- Use the stage action file as the single place to capture scope, acceptance, and validation notes.

## Bug Management Workflow

- Source of truth: `support/bugmgmt/issues/issues.jsonl` (one JSON object per line).
- Use project-based IDs (example: `BMG-2025-12-001`).

- Required fields:
  - `id`, `date`, `project`, `phase`, `stage`, `area`, `status`
  - `severity`, `summary`, `owner`
- Detail fields: `root_cause`, `proposed_fix`, `qa_reproduction`.
- After any bug change, regenerate the JSON and HTML exports.

## Project Management Workflow

- Maintain project structure in `projects/<project>/`.
- Track phases in `phase_definition.md` and `action_plan_phaseNN.md`.
- Log progress in stage action files under `actions/`.
- Update `support/ui/PM.html` with current phase status, owners, and next actions.
- Render formatted docs for offline viewing:
  - `python3 support/scripts/render_docs.py`

## Starting a New Project Repo

1. Copy this repository to a new location (or clone and rename). Treat the template as read-only.
2. Optional: run `python3 support/scripts/init_project.py --project <project> --prefix <PREFIX> --owner "Name"`.  
This scaffolds docs and updates `support/docs/projectplan.md` and `support/ui/PM.html`.
3. If you did not use the script, update `support/docs/projectplan.md` and create `projects/<project>/`.
4. Add phase definitions and action plans using templates in `support/docs/templates/`.
5. Set a project prefix in `support/scripts/issues.py` (`PROJECT_PREFIXES`).
6. Regenerate BugMgmt exports and open `support/ui/PM.html` to verify.

## Notes

- Keep content PII-free unless explicitly required.
- Prefer local assets and deterministic outputs.
- Update this guide when the process changes.