JAM.26

DAY 1 WORKFLOW CHECKLIST

Day 1 - Planning & Development Workflow

Requirements, functional planning, UI/UX, local development, GitHub, and AI-assisted engineering.

01 Understand the Requirement

Requirement > Users > Pages > Features > Actions > Data

- [ ] Main user is identified
- [ ] Required pages or screens are identified
- [ ] Core features are separated from optional features
- [ ] Important user actions are listed
- [ ] Required data is identified
- [ ] Unclear requirements are written as questions
- [ ] Assumptions are identified before implementation
- [ ] Task is small enough to explain clearly

CHECKPOINT:

You can explain what the application must do before asking AI to build it.

02 Functional Planning with Excalidraw

User > Interface > Action > Application Logic > Data > Result

[ ]	Main user flow is diagrammed
[ ]	Important screens or interfaces are represented
[ ]	User actions are connected to application behavior
[ ]	Data movement is shown when relevant
[ ]	Major components and interactions are visible
[ ]	Diagram focuses on function rather than visual styling
[ ]	Missing interactions or unclear decisions are identified
[ ]	Plan can be explained to another person

CHECKPOINT:

The system behavior is understandable before implementation begins.

03 UI/UX Planning with Variant

Requirements > Layout > Components > States > Review

- [ ] Design prompt describes the application clearly
- [ ] Required pages or views are represented
- [ ] Core components are visible
- [ ] Primary actions are easy to find
- [ ] Empty states are considered
- [ ] Form fields and required information are clear
- [ ] Design matches the functional plan
- [ ] Generated design is reviewed instead of accepted automatically

CHECKPOINT:

You have a usable interface direction that supports the required workflow.

04 Development Environment & Git

Project > Install > Run > Inspect > GitHub > Commit

- [ ] Project opens correctly in Visual Studio Code
- [ ] Dependencies install successfully
- [ ] Development server starts
- [ ] Application opens in the browser
- [ ] Browser Developer Tools can be used
- [ ] Git repository is initialized or connected
- [ ] Git status is understood before committing
- [ ] Project is connected to the correct GitHub repository
- [ ] Meaningful working checkpoint is committed
- [ ] Latest commit is pushed to GitHub

CHECKPOINT:

The project runs locally and has a recoverable version-control checkpoint.

05 AI-Assisted Implementation

Describe > Generate or Modify > Review > Run > Test > Debug > Verify

- [ ] AI receives one clear objective
- [ ] Existing project is inspected before major changes
- [ ] Generated or modified files are reviewed
- [ ] Application is run after changes
- [ ] Requested behavior is tested manually
- [ ] Errors are read before asking AI to fix them
- [ ] AI receives evidence and project context
- [ ] Proposed fixes are reviewed
- [ ] Feature is tested again after the fix
- [ ] Working result is verified before moving on

CHECKPOINT:

AI accelerates implementation without replacing review, testing, or engineering judgment.

06 Day 1 Working Checkpoint

Plan > Design > Build > Test > Commit

[ ]	Functional diagram is complete enough to explain the feature
[ ]	UI design supports the planned workflow
[ ]	Application runs locally
[ ]	At least one meaningful feature works
[ ]	Feature has been tested
[ ]	Browser console has been checked
[ ]	Any AI-generated changes were reviewed
[ ]	Known issues are understood
[ ]	Working state is committed to Git
[ ]	Repository is pushed to GitHub

CHECKPOINT:

You leave Day 1 with a planned, designed, tested, and committed local application.

FINAL DAY 1 PATH

REQUIREMENT > FUNCTIONAL PLAN > UI DESIGN > PROJECT

AI-ASSISTED BUILD > REVIEW > RUN > TEST > DEBUG > VERIFY > COMMIT

DAY 1 COMPLETE

You can move from a requirement to a functional plan, translate it into a UI direction, build with AI assistance, review and test the result, debug failures, and preserve a working checkpoint in GitHub.