M01 Daily Workflow

Methodology Beginner

The optimal routine for a productive development session

The interaction loop

Every Claude Code session follows this pattern:

```
1. DESCRIBE  → Explain what you want
2. ANALYZE   → Claude explores the codebase
3. PROPOSE   → Claude proposes a diff
4. REVIEW    → Read and evaluate
5. DECIDE    → Accept / Reject / Modify
6. VERIFY    → Run the tests
7. COMMIT    → Save (optional)
```

Claude proposes, you decide. Never skip the review step, even on changes that seem obvious.

Session start

Before typing the first request:

1. Verify CLAUDE.md is loaded: ask "What's the test command for this project?" and see if Claude answers with the configured command
2. Set a clear intention: one single objective per session
3. Start with `/plan` if the task touches multiple files

```
cd your-project
claude
# Check with /status or a question about the project config
```

During the session

Review the diff at each step: do not let Claude chain multiple changes without intermediate validation. On significant modifications, ask explicitly:

```
You: Show me what you changed before continuing
```

Context management (personal reminders, not quality thresholds):

Context	Action
0-50%	Work freely
50-75%	Be selective about open files
75%+	`/compact` now
90%+	Save decisions, then compact or hand off as needed

Anti-Pattern: Vibe coding

Letting Claude iterate unsupervised across multiple files, then discovering problems an hour later. The end-of-session diff becomes unreadable and rolling back is costly.

Concrete countermeasures:

- Validate after each modified file, not after an entire batch
- Run tests after each significant change
- Commit regularly to have clean rollback points

End of session

Before closing:

```
# Check the repo state
git diff --stat

# Commit if everything is clean
git commit -m "feat: short description"
```

If important decisions were made during the session (architecture choices, adopted conventions), note them in CLAUDE.md so the next session starts from the right context. Sessions do not transfer implicit decisions on their own.

Effective prompt format

```
WHAT:   Fix the timeout bug on login
WHERE:  src/auth/session.ts
HOW:    Increase expiry from 1h to 24h
VERIFY: Login persists after browser refresh
```

This framework avoids ambiguities and reduces clarification back-and-forth.