Software Development Process

Or how the He LLMs do I do this?

Luke Herbert

January 21, 2025

Presentation Overview

The Issue with LLMs

I would like a three legged stool or frame so that I can show that each of the three legs of my process - LLM, meta documents and testing framework - are equally important.

The Issue with LLMs

I would like a three legged stool or frame so that I can show that each of the three legs of my process - LLM, meta documents and testing framework - are equally important.

Contents

Integrated Development Approach

Three Parallel Tools

1. Meta Documents

Integrated Development Approach

Three Parallel Tools

1. Meta Documents
2. LLM Integration

Integrated Development Approach

Three Parallel Tools

1. Meta Documents
2. LLM Integration
3. Testing Framework and Continuous Testing

Integrated Development Approach

Three Parallel Tools

1. Meta Documents
2. LLM Integration
3. Testing Framework and Continuous Testing

Contents

Meta Documents Role

Living Documentation

- Initial project scoping and design
- Document ongoing issues and challenges
- Record development ideas and iterations
- Track tried and abandoned approaches
- Serve as documentation when complete

Contents

LLM Usage

Intelligent Assistant

- Interprets meta documents for context
- Uses design guidelines for implementation
- References history of approaches
- Follows specific development instructions

Advantages of using meta documents and LLM

Break Revision Cycles

- Sometimes the LLM will go round in circles!
- Provides alternative approaches
- Use meta documents to steer the process
- Use the LLM to update the meta files

Contents

Pre-commit Configuration

Automated Quality Checks

- Pre-commit hooks run before each commit
- Enforces code quality standards
- Runs automated tests
- Checks formatting and linting

How Pre-commit Hooks Work

Workflow

1. Developer stages changes with git add
2. Pre-commit runs configured hooks
3. Each hook checks specific aspects:
   - Code formatting
   - Linting rules
   - Test execution
4. Commit blocked if any hook fails
5. Developer fixes issues and tries again

Benefits of Pre-commit Hooks

Key Advantages

- Catches issues before they enter the codebase
- Ensures consistent code style across team
- Automates repetitive quality checks
- Reduces code review overhead
- Prevents broken tests from being committed

Pre-commit Configuration Example

```
repos:
-   repo: https://github.com/pre-commit/pre-commit-hooks
    rev: v4.4.0
    hooks:
    -   id: trailing-whitespace
    -   id: end-of-file-fixer
    -   id: check-yaml
    -   id: check-added-large-files

-   repo: https://github.com/psf/black
    rev: 23.3.0
    hooks:
    -   id: black

-   repo: local
    hooks:
```

Contents

Testing Approach

Flexible Testing Strategy

- Initial minimal working application tests
- Parallel development with main code
- Focus on problem areas
- Regular updates to match running code

Test-Driven Development

TDD Cycle

1. Write failing test first
2. Implement minimal code to pass test
3. Refactor while keeping tests green
4. Repeat for next feature

Benefits

- Ensures code is testable by design
- Prevents over-engineering
- Documents expected behavior
- Catches regressions early

Test Organization

Directory Structure

- Tests mirror source code structure
- Component tests in ___tests___ directories
- Shared test utilities and helpers
- Clear separation of concerns

Test Categories

- Unit tests for individual components
- Integration tests for component interactions
- End-to-end tests for complete workflows
- Performance and accessibility tests

Test Maintenance

Continuous Improvement

- Regular test suite review and updates
- Refactor tests as code evolves
- Monitor test coverage and quality
- Document test patterns and practices

Common Challenges

- Keeping tests up to date with code changes
- Managing test data and dependencies
- Balancing test coverage and maintenance
- Handling flaky or unreliable tests