

AI-Driven Development - 30-Day Challenge-Task-7

Name: ZoyaAfzal

Slot:(Friday 6–9 PM)

Task Overview :

What is SPECKit Plus? – Write a short note Students must download SPECKit and review the 5 core concepts included in it:

1. /constitution
2. /specify
3. /pla
4. /tasks
5. /implement

What is SPECKit Plus(ShortNote):

Spec-Kit Plus is an advanced toolkit for Spec-Driven Development (SDD), where clear specifications are written before any code is generated. It builds on the original GitHub Spec Kit and provides a structured workflow for creating AI-native applications, including multi-agent systems. In Spec-Kit Plus, specifications, architecture decisions, plans, tasks, and prompt histories are treated as first-class project files, ensuring clarity and consistency. The toolkit offers templates, automation scripts, and AI-tool integrations that guide you from writing specs to planning, task generation, and implementation. It is installed via Python and initialized with simple CLI commands, making setup easy. Overall, Spec-Kit Plus enforces structured development, reducing errors and producing more maintainable, reliable AI-generated software.

Core Concepts of SPECKit Plus



Constitution Phase:

The Constitution Phase in Spec-Kit Plus sets the foundational rules for a project by defining coding standards, testing guidelines, and documentation requirements in a single `constitution.md` file. All future specs and code must follow these rules, ensuring consistency and quality. It acts as the guiding framework for the entire development workflow.



Specify Phase (Specification Phase)

In Spec-Kit Plus the **Specify Phase** is where you turn an idea into a clear, structured set of requirements, you answer *what* needs to be built and why. Using the `/sp.specify` command, you generate a `spec.md` file (plus requirement-checklist) describing the feature's functionality, user stories, acceptance criteria, edge cases, and constraints.



Plan Phase

The **Plan Phase** translates the specification into a clear technical roadmap. Using the `/sp.plan` command, it generates files like `plan.md` and `data-model.md` that outline architecture, data models, APIs, and implementation details. This phase ensures all project rules are followed and technical decisions are clarified, providing a structured guide for development.



Tasks Phase

The **Tasks Phase** breaks the plan into actionable work units using the `/sp.tasks` command, generating a `tasks.md` file. Each task is small, test-driven, and clearly ordered, making implementation manageable and traceable. This phase ensures all work aligns with the specification and plan, ready for structured development.



Implement Phase

The **Implement Phase** turns the plan and tasks into working code using the `/sp.implement` command. It follows a Test-Driven Development workflow: write tests first (RED), code to pass them (GREEN), then refactor. Tasks are executed in order, producing source code, tests, and updated progress aligned with the specification and plan.