Day 1: Foundation & Setup Mastery

GitHub Spec Kit Intensive Training - Foundation Day

Duration: 6-8 hours
Objective: Master environment setup, understand SDD fundamentals, complete first spec-driven project
Success Criteria: 100% environment validation, successful project initialization, basic workflow proficiency

---

🌅 Morning Session (3-4 hours)

Hour 1: Environment Validation & Setup

1.1 Complete Environment Validation (30 minutes)

```
# Run comprehensive validation
cd /home/ubuntu/github_spec_training
./validate_environment.sh

# If any failures, address immediately
./validate_environment.sh --install-help
```

Validation Checklist:
- [ ] Python 3.11+ installed and accessible
- [ ] Git configured with your credentials
- [ ] UV package manager working
- [ ] WSL2 properly configured (Windows 11)
- [ ] AI coding agent accessible (GitHub Copilot/Claude Code)
- [ ] Network connectivity to GitHub and PyPI
- [ ] HX-Infrastructure-Knowledge-Base repository cloned

1.2 GitHub Spec Kit Installation & Verification (30 minutes)

```
# Test Spec Kit installation
uvx --from git+https://github.com/github/spec-kit.git specify init test_foundation_pro
ject

# Verify installation success
cd test_foundation_project
ls -la
cat README.md
```

Expected Outputs:
- Project directory created with proper structure
- Configuration files present (.specify/, prompts/, etc.)
- AI agent integration working
- Slash commands accessible in your AI agent

Hour 2: Spec-Driven Development Fundamentals

2.1 Understanding the SDD Philosophy (45 minutes)

Core Concepts to Master:
1. **Intent-First Development:** Specifications capture “what” and “why” before “how”
2. **AI-Assisted Implementation:** Leverage AI agents for code generation from clear specs
3. **Iterative Refinement:** Specifications evolve through feedback loops
4. **Quality Through Clarity:** Better specs = better code = fewer bugs

Practical Exercise:

```
# Create your first specification
uvx --from git+https://github.com/github/spec-kit.git specify init day1_learning_proje
ct --ai copilot

cd day1_learning_project
```

In your AI agent, use the `/specify` command to create a specification for:
Project: “Personal Task Management System for HX-Infrastructure Projects”

Specification Requirements:
- User can create, edit, and delete tasks
- Tasks have priority levels and due dates
- Integration with HX-Infrastructure project categories
- Simple web interface for task management

2.2 The Four Phases Deep Dive (30 minutes)

Phase 1: Specify
- Define user journeys and success criteria
- Focus on business value and user experience
- Avoid technical implementation details
- Create clear acceptance criteria

Phase 2: Plan
- Technical architecture and technology choices
- Integration with existing systems (HX-Infrastructure)
- Security and compliance considerations
- Resource and timeline estimates

Phase 3: Tasks
- Break down into atomic, testable units
- Define clear completion criteria
- Sequence dependencies properly
- Enable parallel development where possible

Phase 4: Implement
- Execute tasks using AI assistance
- Validate against specifications continuously
- Refactor and optimize iteratively
- Document decisions and learnings

Hour 3: First Hands-On Project

3.1 Complete Specify Phase (45 minutes)

Using your AI agent with the `/specify` command, create a comprehensive specification for the Personal Task Management System. Your specification should include:

User Stories:
- As an HX-Infrastructure team member, I want to track project tasks
- As a project manager, I want to see task priorities and deadlines
- As a developer, I want to categorize tasks by project area

Success Criteria:
- System handles 100+ tasks without performance issues
- Interface is intuitive for non-technical users
- Data persists between sessions
- Mobile-responsive design

Acceptance Tests:
- User can create a task in under 30 seconds
- Task filtering works across all categories
- Due date notifications are accurate
- Data export functionality works

3.2 Validation Checkpoint (15 minutes)

Self-Assessment Questions:
1. Does your specification clearly define the “what” without the “how”?
2. Can someone else understand the requirements without technical knowledge?
3. Are success criteria measurable and testable?
4. Does the specification align with HX-Infrastructure needs?

🌞 Afternoon Session (3-4 hours)

Hour 4: Plan Phase Mastery

4.1 Technical Planning with AI Assistance (45 minutes)

Use the `/plan` command in your AI agent to create a technical plan for your task management system:

Planning Focus Areas:
- **Technology Stack:** Choose appropriate technologies for HX-Infrastructure environment
- **Architecture:** Design scalable, maintainable system architecture
- **Data Model:** Define entities, relationships, and storage strategy
- **Integration Points:** How system connects with existing HX-Infrastructure tools
- **Security:** Authentication, authorization, and data protection
- **Deployment:** How system will be deployed and maintained

Expected Deliverables:
- Detailed technical architecture document
- Technology justification and alternatives considered
- Data flow diagrams and system interactions

- Security and compliance considerations
- Deployment and maintenance strategy

4.2 Plan Validation and Refinement (15 minutes)

Plan Quality Checklist:
- [ ] Technology choices align with HX-Infrastructure standards
- [ ] Architecture supports specified requirements
- [ ] Security considerations are comprehensive
- [ ] Deployment strategy is practical and tested
- [ ] Plan includes monitoring and maintenance procedures

Hour 5: Tasks Phase Implementation

5.1 Task Breakdown with AI (45 minutes)

Use the `/tasks` command to break down your plan into actionable tasks:

Task Categories:
1. **Setup and Configuration**
- Environment setup
- Database initialization
- Basic project structure

1. **Core Functionality**
   - Task CRUD operations
   - User interface components
   - Data persistence layer

2. **Advanced Features**
   - Priority and categorization
   - Due date management
   - Search and filtering

3. **Integration and Deployment**
   - HX-Infrastructure integration
   - Testing and validation
   - Deployment configuration

Task Quality Standards:
- Each task is completable in 2-4 hours
- Clear acceptance criteria for each task
- Dependencies clearly identified
- Tasks can be validated independently

5.2 Task Prioritization and Sequencing (15 minutes)

Prioritization Framework:
1. **Critical Path:** Tasks that block other work
2. **Risk Mitigation:** High-risk tasks tackled early
3. **Value Delivery:** User-facing features prioritized
4. **Learning Curve:** Complex tasks when energy is highest

Hour 6: Implement Phase Introduction

6.1 Implementation Strategy (30 minutes)

Implementation Best Practices:
- Start with highest-risk tasks
- Validate frequently against specifications
- Use AI assistance for code generation
- Document decisions and learnings
- Refactor continuously for quality

AI-Assisted Implementation:
- Use `/implement` command for code generation
- Provide context from specifications and plans
- Validate generated code against requirements
- Iterate based on testing and feedback

6.2 First Implementation Sprint (30 minutes)

Implement the first 2-3 tasks from your task list:

Suggested Starting Tasks:
1. Project setup and basic structure
2. Database schema creation
3. Basic task model implementation

Implementation Validation:
- Code runs without errors
- Basic functionality works as specified
- Code follows HX-Infrastructure standards
- Documentation is clear and complete

🌆 Evening Session (1-2 hours)

Hour 7: HX-Infrastructure Integration

7.1 Archive Content Analysis (45 minutes)

Analyze your HX-Infrastructure-Knowledge-Base repository to understand:

Integration Opportunities:
- Existing project categories and structures
- Common task types and workflows
- Integration points with current tools
- Lessons learned from past projects

Practical Exercise:

```bash
cd /home/ubuntu/github_spec_training/HX-Infrastructure-Knowledge-Base
# Analyze repository structure
find . -name "*.md" | head -10
# Look for project patterns and categories
grep -r "project" . | head -5
```

7.2 Specification Refinement (15 minutes)

Update your task management system specification to better integrate with HX-Infrastructure:

Refinement Areas:
- Task categories that match HX-Infrastructure projects
- Integration with existing documentation standards
- Workflow alignment with current processes
- Data export formats compatible with existing tools

Hour 8: Day 1 Validation & Preparation

8.1 Proficiency Self-Assessment (30 minutes)

Foundation Skills Checklist:
- [ ] Environment fully validated and working
- [ ] GitHub Spec Kit installation successful
- [ ] Understanding of all four SDD phases
- [ ] Completed first specification document
- [ ] Created technical plan with AI assistance
- [ ] Generated task breakdown
- [ ] Implemented basic functionality
- [ ] Integrated HX-Infrastructure considerations

Proficiency Levels:
- **Novice (0-40%):** Can follow instructions but needs guidance
- **Beginner (41-60%):** Understands concepts, needs practice
- **Intermediate (61-80%):** Can work independently with occasional help
- **Advanced (81-100%):** Can teach others and troubleshoot issues

Target for Day 1: 60-70% proficiency in foundation skills

8.2 Day 2 Preparation (15 minutes)

Tomorrow's Focus: Intermediate Application with Archive Integration

Preparation Tasks:
- [ ] Review today's learnings and document insights
- [ ] Identify specific HX-Infrastructure use cases for tomorrow
- [ ] Prepare questions about advanced Spec Kit features
- [ ] Set up development environment for more complex projects

8.3 Lessons Learned Documentation (15 minutes)

Create a lessons learned document for Day 1:

```
# Create lessons learned file
touch /home/ubuntu/github_spec_training/day1_lessons_learned.md
```

Document the following:
- What worked well in your learning process
- Challenges encountered and how you solved them
- Insights about Spec-Driven Development
- Questions for further exploration
- Improvements for teaching others

🎯 Day 1 Success Validation

Mandatory Completion Criteria:

- [ ] Environment validation passes 100%
- [ ] GitHub Spec Kit successfully installed and tested
- [ ] Complete specification created for practice project
- [ ] Technical plan generated with AI assistance
- [ ] Task breakdown completed and prioritized
- [ ] At least 3 tasks implemented successfully
- [ ] HX-Infrastructure integration considerations documented
- [ ] Lessons learned documented for future reference

Proficiency Indicators:

- Can explain SDD methodology to someone else
- Comfortable using AI agents with Spec Kit commands
- Understands the value of specifications before implementation
- Can troubleshoot basic environment and installation issues
- Ready to tackle more complex projects tomorrow

If You're Behind Schedule:

- Focus on environment validation and basic workflow
- Skip advanced implementation tasks if needed
- Ensure you understand the four phases conceptually
- Document specific areas where you need more practice
- Plan extra time for catch-up in Day 2 morning session

📚 Additional Resources for Day 1

Essential Reading:

- GitHub Spec Kit Documentation (https://github.com/github/spec-kit)
- Spec-Driven Development Blog Post (https://github.blog/ai-and-ml/generative-ai/spec-driven-development-with-ai-get-started-with-a-new-open-source-toolkit/)

Video Resources:

- Search for "GitHub Spec Kit tutorial" on YouTube
- VS Code GitHub Copilot setup videos

Community Support:

- GitHub Spec Kit Issues and Discussions
- Stack Overflow spec-driven-development tag

End of Day 1
Next: Day 2 - Intermediate Application with Archive Integration

Estimated Completion Time: 6-8 hours
Success Rate Target: 60-70% proficiency in foundation skills

Remember: This is intensive training. Take breaks, stay hydrated, and don’t hesitate to revisit concepts as needed. The goal is deep understanding, not just completion.