Phase 0: Foundation & Setup - COMPLETE ✅

Date Completed: October 16, 2025
Duration: Initial implementation session
Status: All Phase 0 objectives achieved

🎯 Phase 0 Objectives

Phase 0 focused on establishing the foundational infrastructure for Solo Git:

- ✅ Project directory structure and scaffolding
- ✅ CLI framework with Click
- ✅ Configuration management system
- ✅ API client for Abacus.ai integration
- ✅ Logging and error handling
- ✅ Project files (README, requirements, setup.py, etc.)
- ✅ Basic commands for setup verification
- ✅ Git repository initialization

📦 Deliverables

1. Project Structure

```
solo-git/
├── sologit/                 # Main package
│   ├── __init__.py
│   ├── cli/                  # Command-line interface
│   │   ├── main.py          # Main CLI entry point
│   │   ├── commands.py      # Command implementations (placeholder)
│   │   └── config_commands.py # Configuration commands
│   ├── config/               # Configuration management
│   │   ├── manager.py       # Config loading/saving
│   │   └── templates.py     # Config templates
│   ├── api/                  # API clients
│   │   └── client.py        # Abacus.ai client
│   ├── utils/                # Utilities
│   │   └── logger.py        # Logging infrastructure
│   ├── core/                 # Core functionality (Phase 1+)
│   └── engines/              # Git/Test/AI engines (Phase 1+)
├── tests/                   # Test suite
├── docs/                    # Documentation
├── examples/                # Usage examples
├── scripts/                 # Utility scripts
├── README.md                # Project documentation
├── requirements.txt         # Python dependencies
├── setup.py                 # Package installation
├── .gitignore               # Git ignore rules
├── LICENSE                  # MIT License
└── pyproject.toml          # Build configuration
```

2. CLI Commands

Core Commands

- `evogitctl hello` - Test command to verify installation
- `evogitctl version` - Show version and API status
- `evogitctl --help` - Show all available commands

Configuration Commands

- `evogitctl config init` - Create default config file
- `evogitctl config setup` - Interactive API credential setup
- `evogitctl config show` - Display current configuration
- `evogitctl config test` - Test API connection (requires credentials)
- `evogitctl config path` - Show config file location
- `evogitctl config env-template` - Generate .env template

Placeholder Commands (Phase 1+)

- `evogitctl repo` - Repository management
- `evogitctl pad` - Workpad operations
- `evogitctl test` - Test execution
- `evogitctl pair` - AI pairing session

3. Configuration System

Features:
- YAML-based configuration at `~/.sologit/config.yaml`
- Environment variable overrides
- Multi-tier model configuration (planning, coding, fast)
- Budget controls and cost tracking
- Test configuration
- Workflow settings

Example Configuration:

```yaml
abacus:
  endpoint: https://api.abacus.ai/v1
  api_key: YOUR_API_KEY_HERE

models:
  planning_model: gpt-4o
  coding_model: deepseek-coder-33b
  fast_model: llama-3.1-8b-instruct

budget:
  daily_usd_cap: 10.0
  alert_threshold: 0.8
  track_by_model: true

promote_on_green: true
rollback_on_ci_red: true
```

4. API Client

Features:
- OpenAI-compatible chat interface
- Streaming support (for future use)
- Error handling and logging
- Connection testing

Supported Models (via Abacus.ai RouteLLM):
- Planning: GPT-4, Claude 3.5 Sonnet, Llama 3.3 70B
- Coding: DeepSeek-Coder 33B, CodeLlama 70B
- Fast: Llama 3.1 8B, Gemma 2 9B

5. Logging System

Features:
- Colored console output
- Configurable log levels (INFO/DEBUG)
- File logging support
- Structured logging with context

🧪 Verification

Installation Test

```
$ cd solo-git
$ pip install -e .
$ evogitctl hello
🏁 Solo Git is ready!

Solo Git - where tests are the review and trunk is king.
```

Version Check

```
$ evogitctl version
Solo Git (evogitctl) version 0.1.0
Python 3.11.6 (main, Sep 16 2025, 12:40:29) [GCC 12.2.0]
Abacus.ai API: ✗ not configured
```

Configuration

```
$ evogitctl config init
✅ Created configuration file at /home/ubuntu/.sologit/config.yaml

$ evogitctl config show
📋 Solo Git Configuration

🔐 Abacus.ai API:
  Endpoint:  https://api.abacus.ai/v1
  API Key:   YOUR_API...HERE (use --secrets to show)

🤖 Models:
  Planning:  gpt-4o
  Coding:    deepseek-coder-33b
  Fast:      llama-3.1-8b-instruct
...
```

📍 Current Status

✅ Completed

1. Project structure created
2. CLI framework operational
3. Configuration system working
4. API client implemented
5. Logging infrastructure ready
6. All project files created
7. Git repository initialized
8. Package installable via pip

🔧 Ready for Development

- Clear structure for Phase 1 implementation

- Placeholder command groups for future features
- Extensible architecture for Git, Test, and AI engines

📋 Next Steps: Phase 1

Phase 1: Core Git Engine & Workpad System
Duration: Days 3-5 (October 18-20)

Key Tasks

1. **Git Engine Implementation**
   - Repository initialization (from zip/git URL)
   - Workpad creation and management
   - Checkpoint system
   - Fast-forward merge operations

2. **Test Orchestration Foundation**
   - Docker sandbox integration
   - Test configuration parsing
   - Parallel test execution
   - Result collection and reporting

3. **MCP Tools** (if pursuing MCP architecture)
   - Tool: `repo.init`
   - Tool: `pad.create`
   - Tool: `pad.applyPatch`
   - Tool: `pad.promote`
   - Tool: `test.run`

Files to Create

- `sologit/engines/git_engine.py` - Git operations wrapper
- `sologit/engines/test_orchestrator.py` - Test execution
- `sologit/engines/patch_engine.py` - Diff application
- `sologit/core/workpad.py` - Workpad management
- `sologit/core/repository.py` - Repository abstraction
- CLI commands for repo and pad operations

Dependencies to Add

- `gitpython>=3.1.40` - Git operations
- `docker>=7.0.0` - Container management

💡 Design Decisions

1. Python + Click for CLI

Rationale: Fast development, excellent CLI building experience, wide Python ecosystem.

2. YAML Configuration

Rationale: Human-readable, supports complex nested structures, standard in DevOps tools.

3. Abacus.ai RouteLLM API

Rationale: Single endpoint for multiple models, no local hosting, pure cloud simplicity.

4. Modular Architecture

Rationale: Separation of concerns enables parallel development of Git, Test, and AI engines.

5. Git as Foundation

Rationale: Leverage Git's proven integrity and time-machine capabilities rather than reimplementing.

🎯 Phase 0 Success Criteria

All criteria met:
- ✅ Clean project structure created
- ✅ CLI framework operational and testable
- ✅ Configuration system implemented
- ✅ API client ready for use
- ✅ Logging and error handling in place
- ✅ Package installable and importable
- ✅ Git repository initialized
- ✅ Documentation complete

🚀 Getting Started (For Developers)

1. Install Solo Git

```
cd /home/ubuntu/code_artifacts/solo-git
pip install -e .
```

2. Configure API Credentials

```
# Interactive setup
evogitctl config setup

# Or manually edit config
evogitctl config init
nano ~/.sologit/config.yaml
```

3. Verify Installation

```
evogitctl hello
evogitctl version
evogitctl config show
```

4. Test API Connection (requires credentials)

```
evogitctl config test
```

📚 Resources

- **Game Plan**: See `~/solo_git_game_plan.md` for full roadmap
- **Vision**: See `/home/ubuntu/Uploads/sologit.txt` for philosophy
- **README**: See `README.md` for user-facing documentation
- **Config Template**: Run `evogitctl config init` to generate

🎉 Conclusion

Phase 0 is **COMPLETE**! The foundation is solid, the architecture is clean, and the project is ready for Phase 1 development.

Ready for: Git Engine implementation, Test Orchestration, and Workpad system.

Next session: Start Phase 1 - Core Git Engine & Workpad System

---

"The foundation is laid. Now we build the engines that make Solo Git fly."