Vigthoria CLI User Handbook

Vigthoria Labs

January 2026

Contents

Vigthoria CLI User Handbook

AI-Powered Terminal Coding Assistant

Version 1.0.0

Table of Contents

Introduction

What is Vigthoria CLI?

Vigthoria CLI is a powerful terminal-based coding assistant that brings the full capabilities of Vigthoria's AI models to your command line. Whether you're working locally, over SSH, or in a headless environment, Vigthoria CLI provides intelligent code assistance without leaving your terminal.

Key Features

Feature	Description
Interactive Chat	Natural language coding assistance with context awareness
? Agent Mode	Claude Code-like autonomous file editing and command execution
Session Management	Save, resume, and manage conversation sessions
File Editing	AI-powered code modifications with diff preview
Code Generation	Generate complete code from descriptions
Code Explanation	Understand complex code with detailed explanations
Bug Fixing	Automatic issue detection and resolution
Code Review	Quality analysis with actionable suggestions
Multi-Model Support	Access to all Vigthoria AI models based on subscription
Project Context	Understands your project structure and dependencies
? Local Mode	Works offline with Ollama for local development

Who Should Use This?

- **Backend Developers** - Working primarily in terminal environments
- **DevOps Engineers** - Scripting and automation tasks
- **Remote Workers** - SSH sessions to remote servers
- **Power Users** - Prefer keyboard-driven workflows
- **CI/CD Integration** - Automated code generation and review

Requirements

Component	Minimum	Recommended
Node.js	18.0.0	20.x LTS
npm	8.x	10.x
OS	Linux, macOS, Windows	Linux/macOS
Internet	Required	Stable connection
Terminal	Any	iTerm2, Warp, Kitty

Installation

Quick Install (Recommended)

```
curl -fsSL https://cli.vigthoria.io/install.sh | bash
```

This script will: 1. Check Node.js version 2. Install Vigthoria CLI globally 3. Set up shell completions 4. Create command shortcuts

npm Global Install

```
npm install -g vigthoria-cli
```

Manual Installation

```
# Clone from Vigthoria Market
git clone https://market.vigthoria.io/vigthoria/vigthoria-cli.git
cd vigthoria-cli

# Or from Vigthoria Community
git clone https://community.vigthoria.io/repos/vigthoria-cli.git
cd vigthoria-cli

# Install dependencies
npm install

# Build
npm run build

# Link globally
npm link
```

Verify Installation

```
vigthoria --version
# Output: 1.0.0

vigthoria --help
# Shows all available commands
```

Update

```
npm update -g vigthoria-cli
```

Quick Start Guide

Step 1: Login

```
vigthoria login
```

Choose your login method: - **Email & Password** - Enter your Vigthoria account credentials - **API Token** - Use a token from your Vigthoria Coder dashboard - **Browser Login** - OAuth flow (opens browser)

Step 2: Check Status

```
vigthoria status
```

This shows: - Account information - Subscription plan - Available models - API connectivity

Step 3: Start Coding

```
# Start interactive chat
vigthoria chat

# Or use the short alias
vig c
```

Step 4: Your First AI Conversation

```
you > Create a function to validate email addresses in TypeScript

vigthoria >
Here's a robust email validation function:

```typescript
function validateEmail(email: string): boolean {
  const emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
  return emailRegex.test(email);
}

// Enhanced version with detailed validation
interface EmailValidationResult {
  valid: boolean;
  reason?: string;
}

function validateEmailDetailed(email: string): EmailValidationResult {
  if (!email) {
```
```

```
    return { valid: false, reason: 'Email is required' };
  }

  if (email.length > 254) {
    return { valid: false, reason: 'Email too long' };
  }

  const regex = /^[a-zA-Z0-9.!#$%&'*+/=?^_`{|}~-]+@[a-zA-Z0-9](?:[a-zA-Z0-9-]{0,61}[a-zA-Z0-9])

  if (!regex.test(email)) {
    return { valid: false, reason: 'Invalid email format' };
  }

  return { valid: true };
}

---

## Authentication

### Login Methods

#### Email & Password

```bash
vigthoria login
# Select: Email & Password
# Enter your email and password
```
```

API Token Get your token from: `https://coder.vigthoria.io/settings/api`

```
# Interactive
vigthoria login
# Select: API Token
# Paste your token

# Or directly
vigthoria login --token YOUR_API_TOKEN
```

Environment Variable For CI/CD or automated environments:

```
export VIGTHORIA_TOKEN="your-api-token"
vigthoria status  # Will use the environment token
```

Check Authentication Status

```
vigthoria status
```

Output:

```
??? Account Status ???

Account:
  Email: user@example.com
  User ID: usr_abc123

Subscription:
  Plan: PRO
  Status: Active
  Expires: 2026-12-31 (365 days)

Available Models:
  o vigthoria-fast - Quick responses (1.1B)
  o vigthoria-mini - Balanced (3.8B)
  o vigthoria-code - Code specialist (8B)
  o vigthoria-creative - Creative (9B)

API Status:
  Server: Online
  Endpoint: https://coder.vigthoria.io
```

Logout

```
vigthoria logout
```

Interactive Chat Mode

The chat mode is the primary way to interact with Vigthoria AI. It maintains conversation context and understands your project.

Starting Chat

```
vigthoria chat
# or
vig c

# With specific model
vigthoria chat --model pro

# With project context
vigthoria chat --project /path/to/project
```

Chat Commands

While in chat mode, use these special commands:

Command	Description
`/file <path>`	Add a file's content to the conversation
`/edit <path>`	Start editing a specific file
`/diff`	Show pending changes
`/apply`	Apply pending changes to files
`/model <name>`	Switch AI model
`/clear`	Clear conversation history
`/help`	Show available commands
`/exit` or `/quit`	Exit chat mode

Example Chat Session

```
????????????????????????????????????????????????????????????
?   VIGTHORIA CLI - AI-Powered Coding Assistant             ?
????????????????????????????????????????????????????????????

+-----------------------------------------------------------+
| Vigthoria Chat                                            |
| Model: vigthoria-code                                     |
| Plan: PRO                                                 |
| Project: /home/user/my-project                            |
+-----------------------------------------------------------+

Type your message or /help for commands. Press Ctrl+C to exit.

you > /file src/api.ts
? Added src/api.ts to context (245 lines)

you > Add rate limiting to the authentication endpoint

vigthoria >
I'll add rate limiting to your authentication endpoint. Based on the code in
src/api.ts, here's what I recommend:

[... detailed response with code ...]

you > /edit src/api.ts
What changes would you like to make?
> Add the rate limiting code you just showed

[Diff preview]
- export async function login(req, res) {
+ import rateLimit from 'express-rate-limit';
+
+ const authLimiter = rateLimit({
+   windowMs: 15 * 60 * 1000,
+   max: 5
```

```
+ });
+
+ export async function login(req, res) {

Apply changes? (Y/n) y
? Changes applied to src/api.ts
```

Context Management

The chat remembers your conversation and any files you've added. To start fresh:

```
you > /clear
? Conversation cleared
```

? Agent Mode

Agent mode is Vigthoria CLI's most powerful feature - similar to **Claude Code**. It enables the AI to autonomously:

- ? **Read files** - Examine any file in your project
- ? **Write files** - Create or modify files
- ? **Search** - Grep patterns and glob files
- ? **Execute** - Run shell commands
- ? **Browse** - List directories
- ? **Git** - Run git commands

Starting Agent Mode

```
# Direct agent command
vigthoria agent

# Or with shortcut
vig a

# Or enable in chat
vigthoria chat --agent
vigthoria chat -a
```

Agent Options

Option	Description	Default
`-m, --model <model>`	AI model to use	`balanced`
`-p, --project <path>`	Project context path	Current directory
`-l, --local`	Use local Ollama	`false`
`--auto-approve`	Auto-approve all actions	`false`
`-r, --resume`	Resume last session	`false`

Available Tools

The AI can use these tools when in agent mode:

Tool	Description	Example Use
`read_file`	Read file contents with line range	“Read lines 1-50 of index.ts”
`write_file`	Create or overwrite a file	“Create a new config.ts file”
`edit_file`	Search & replace in file	“Change the function name”
`bash`	Execute shell commands	“Run npm install express”
`grep`	Search pattern in files	“Find all TODO comments”
`list_dir`	List directory contents	“Show files in src/”
`glob`	Find files by pattern	"Find all *.test.ts files"
`git`	Run git commands	“git status”

Permission System

By default, agent mode **asks for permission** before: - Writing or editing files - Running shell commands - Making destructive changes

```
? The AI wants to use write_file
   File: src/utils/helper.ts
   Content: [Preview shown]

Allow this action? [y/N] y
? write_file: Success
```

Auto-Approve Mode

? WARNING: DANGEROUS - Only use in trusted environments!

Auto-approve mode allows all AI actions without confirmation:

```
vigthoria agent --auto-approve
```

Or toggle within a session:

```
you > /approve
? Auto-approve: ON - All actions will execute automatically
```

Example Agent Session

```
????????????????????????????????????????????????????????????
?   VIGTHORIA CLI - AI-Powered Coding Assistant             ?
?   Version 1.0.0                                          ?
????????????????????????????????????????????????????????????

[Agent Mode] Model: vigthoria-code | Auto-approve: OFF

you > Create an Express server with TypeScript and add user authentication

vigthoria > I'll create an Express server with authentication for you. Let me:
```

```
1. Check your current project structure
2. Create the necessary files
3. Set up dependencies

? Executing: list_dir
   Path: .
? list_dir: package.json, src/, tsconfig.json

? Executing: read_file
   File: package.json, Lines: 1-30
? read_file: Success (25 lines)

? Executing: write_file
   File: src/server.ts
   Creating Express server with authentication...
Allow? [y/N] y
? write_file: Success

? Executing: write_file
   File: src/middleware/auth.ts
Allow? [y/N] y
? write_file: Success

? Executing: bash
   Command: npm install express jsonwebtoken bcryptjs @types/express @types/jsonwebtoken
Allow? [y/N] y
? bash: Success

I've created your Express server with JWT authentication!

Files created:
- src/server.ts - Main server file
- src/middleware/auth.ts - Authentication middleware
- src/routes/auth.ts - Login/register routes

Run `npm run dev` to start the server.
```

Agent vs Chat Mode Comparison

Feature	Chat Mode	Agent Mode
Conversation	?	?
File Context	Manual (`/file`)	Automatic
Read Files	?	?
Write Files	Via `/edit`	Direct
Run Commands	?	?
Autonomous Actions	?	?

Feature	Chat Mode	Agent Mode
Best For	Q&A, discussion	Implementation

Session Management

Sessions allow you to **save and resume** conversations, preserving context across multiple work sessions.

How Sessions Work

- Each chat/agent session has a unique ID
- Sessions are saved in `~/.vigthoria/sessions/`
- Sessions are project-specific (based on working directory)
- Auto-saved when you exit with `/exit`

Viewing Sessions

```
you > /sessions

??? Saved Sessions ???
m1abc123-xyz [agent] (current)
  /home/user/my-project - 1/13/2026, 8:30:00 PM
m1def456-uvw
  /home/user/other-project - 1/12/2026, 3:15:00 PM
m1ghi789-rst [agent]
  /home/user/my-project - 1/11/2026, 2:00:00 PM
```

Resuming a Session

```
# Resume the last session for current project
vigthoria chat --resume
vigthoria chat -r

# Or in agent mode
vigthoria agent --resume
```

Saving Sessions

Sessions are saved automatically on `/exit`. Force save manually:

```
you > /save
? Session saved: m1abc123-xyz
```

Starting a New Session

```
you > /new
? Previous session saved
? New session: m1jkl012-mno
```

Viewing Conversation History

```
you > /history

??? Conversation History ???
1. you: How do I create a REST API...
2. vigthoria: Here's how to create a REST API...
3. you: Can you add authentication...
4. vigthoria: I'll add JWT authentication...
```

Compacting Long Sessions

When conversations get long, compress old messages:

```
you > /compact
? Compacted 12 messages into summary
```

This summarizes older messages to reduce context size while preserving important information.

Session File Location

```
~/.vigthoria/sessions/
+-- m1abc123-xyz.json
+-- m1def456-uvw.json
+-- m1ghi789-rst.json
```

File Operations

Edit a File

```
vigthoria edit <file> [options]
```

Options: - `-i, --instruction <text>` - Editing instruction - `-m, --model <model>` - AI model to use

Examples:

```
# Interactive editing
vigthoria edit src/utils.ts

# With instruction
vigthoria edit src/api.ts -i "Add error handling to all async functions"

# With specific model
vigthoria edit src/complex.ts -m pro
```

Edit Workflow

1. File content is loaded
2. You describe the changes
3. AI generates the modified code

4. Diff is shown for review
5. You approve or reject changes
6. Backup is created automatically

Fix Code Issues

`vigthoria fix <file> [options]`

Options: - `-t, --type <type>` - Fix type: `bugs`, `style`, `security`, `performance` - `--apply` - Auto-apply fixes without confirmation

Examples:

```
# Find and fix bugs
vigthoria fix src/index.ts -t bugs

# Fix security issues with auto-apply
vigthoria fix src/auth.ts -t security --apply

# Fix performance issues
vigthoria fix src/heavy-computation.ts -t performance
```

Explain Code

`vigthoria explain <file> [options]`

Options: - `-l, --lines <range>` - Line range (e.g., `1-50`) - `-d, --detail <level>` - `brief`, `normal`, `detailed`

Examples:

```
# Explain entire file
vigthoria explain src/algorithm.ts

# Explain specific lines
vigthoria explain src/utils.ts -l 50-100

# Detailed explanation
vigthoria explain src/complex.ts -d detailed
```

Code Generation

Generate Code from Description

`vigthoria generate <description> [options]`

Options: - `-l, --language <lang>` - Target language (default: typescript) - `-o, --output <file>` - Output file path - `-m, --model <model>` - AI model to use

Supported Languages

- TypeScript / JavaScript

- Python
- Rust
- Go
- Java
- C# / C++
- Ruby
- PHP
- Swift
- Kotlin

Examples

```
# Generate TypeScript code
vigthoria generate "REST API endpoint for user authentication"

# Generate Python with output file
vigthoria generate "Database migration script for PostgreSQL" -l python -o migrate.py

# Generate with specific model
vigthoria generate "High-performance caching layer" -l rust -m pro
```

Generated Code Quality

Vigthoria generates production-ready code with: - Proper type definitions - Error handling - Input validation - Documentation comments - Modern best practices

---

Code Review & Fixes

Code Review

`vigthoria review <file> [options]`

Options: - `-f, --format <format>` - `text`, `json`, `markdown`

Review Output

```
??? Reviewing: src/api.ts ???
Language: typescript | Lines: 245

Quality Score: 78/100
??????????????????????????????

??? Issues (4) ???

? [security] Line 45: Potential SQL injection vulnerability
? [performance] Line 112: Unnecessary re-render in loop
? [style] Line 67: Magic number should be a named constant
i [info] Line 189: Consider using optional chaining
```

```
??? Suggestions ???

1. Add input validation to the createUser function
2. Consider implementing rate limiting on public endpoints
3. Use environment variables for configuration values
4. Add JSDoc comments to exported functions
```

Export Review to File

```
# Markdown report
vigthoria review src/app.ts -f markdown > code-review.md

# JSON for CI/CD
vigthoria review src/app.ts -f json > review-results.json
```

Configuration

Interactive Configuration

```
vigthoria config
```

This opens an interactive menu to configure: - Default AI model - Color theme - Auto-apply fixes - Show diffs - Max tokens

Set Individual Values

```
vigthoria config --set model=vigthoria-code
vigthoria config --set theme=dark
vigthoria config --set autoApply=true
vigthoria config --set maxTokens=8192
```

Get Values

```
vigthoria config --get model
# Output: vigthoria-code
```

List All Settings

```
vigthoria config --list
```

Reset to Defaults

```
vigthoria config --reset
```

Initialize Project

Create a project-specific configuration:

```
cd /path/to/project
vigthoria init
```

This creates `.vigthoria.json` with project settings.

Configuration File Location

- **Global**: `~/.config/vigthoria-cli/config.json`
- **Project**: `./.vigthoria.json`

Project settings override global settings.

Available AI Models

Model	Size	Tokens	Best For	Plans
`vigthoria-fast`	1.1B	4K	Quick answers, simple tasks	Free, Pro, Ultra
`vigthoria-mini`	3.8B	8K	Balanced performance	Free, Pro, Ultra
`vigthoria-code`	8B	16K	Code generation, debugging	Pro, Ultra
`vigthoria-creative`	9B	16K	Creative writing, documentation	Pro, Ultra
`vigthoria-pro`	32B	32K	Complex tasks, architecture	Ultra
`vigthoria-ultra`	70B	64K	Maximum capability	Ultra

Model Selection

```
# In chat
/model code

# Command line
vigthoria chat --model pro
vigthoria generate "..." --model ultra
```

Model Recommendations

Task	Recommended Model
Quick code snippets	`fast`
General coding	`code`
Documentation	`creative`
Architecture design	`pro`
Complex debugging	`ultra`

? Local Mode with Vigthoria Inference

Local mode allows you to use Vigthoria CLI with the **native Vigthoria LLMs** via Vigthoria Inference Server. This gives you access to the same powerful models available in the cloud, running locally on your machine.

Prerequisites

Install Vigthoria Inference from Vigthoria Market:

```
# Clone from Vigthoria Market
git clone https://market.vigthoria.io/vigthoria/vigthoria-inference.git
cd vigthoria-inference

# Or from Vigthoria Community
git clone https://community.vigthoria.io/repos/vigthoria-inference.git
cd vigthoria-inference

# Install dependencies
pip install -r requirements.txt

# Start Vigthoria Inference Server (Port 8010)
python3 vigthoria_inference_server_v2.py
```

Available Native Vigthoria Models

Vigthoria Inference provides access to all native Vigthoria LLMs:

```
# Mini (0.6B) - Ultra-fast responses
vigthoria-mini-0.6b

# Fast (1.7B) - Quick streaming
vigthoria-fast-1.7b

# Balanced (4B) - General purpose
vigthoria-balanced-4b

# Code (8B) - Code specialist
vigthoria-code-v2-8b

# Creative (9B) - Creative writing
vigthoria-creative-9b-v4

# Music (4B) - Music production
vigthoria-music-master-4b
```

Using Local Mode

```
# Start chat with Vigthoria Inference
vigthoria chat --local
vigthoria chat -l

# Agent mode with local inference
vigthoria agent --local

# With specific Vigthoria model
```

```
vigthoria chat --local --model code
vigthoria chat -l -m balanced
```

Local Model Mapping

When using `--local`, Vigthoria CLI connects to Vigthoria Inference (Port 8010):

Model Alias	Vigthoria Model	Size
`mini`	vigthoria-mini-0.6b	0.6B
`fast`	vigthoria-fast-1.7b	1.7B
`balanced`	vigthoria-balanced-4b	4B
`code`	vigthoria-code-v2-8b	8B
`creative`	vigthoria-creative-9b-v4	9B
`music`	vigthoria-music-master-4b	4B

Advantages of Local Mode

Benefit	Description
Privacy	Code never leaves your machine
Native Models	Full power of Vigthoria LLMs
Offline	Works without internet
No Auth	No login required
Speed	Low latency for local inference
Free	No API costs

System Requirements

Model	VRAM Required	Recommended GPU
Mini (0.6B)	2 GB	Any GPU
Fast (1.7B)	4 GB	RTX 3060+
Balanced (4B)	8 GB	RTX 3070+
Code (8B)	16 GB	RTX 4080+
Creative (9B)	20 GB	RTX 4090/A6000

Example Local Session

```
$ vigthoria chat -l -m code

????????????????????????????????????????????????????????????
?   VIGTHORIA CLI - AI-Powered Coding Assistant              ?
?   Version 1.0.0                                           ?
????????????????????????????????????????????????????????????

[Local Mode] Model: vigthoria-code-v2-8b | Inference: localhost:8010
```

```
you > Write a Python function to reverse a string

vigthoria > Here's a Python function to reverse a string:

def reverse_string(s: str) -> str:
    """Reverse the input string."""
    return s[::-1]

# Alternative using reversed()
def reverse_string_alt(s: str) -> str:
    return ''.join(reversed(s))
```

Integration with Vigthoria Coder

Shared Authentication

Vigthoria CLI uses the same authentication as: - coder.vigthoria.io (Web IDE) - vsuser.vigthoria.io (VS Code IDE) - extension.vigthoria.io (VS Code Extension)

One account works everywhere.

Accessing Cloud Projects

Your projects from Vigthoria Coder are accessible:

```
# List cloud projects (coming soon)
vigthoria projects list

# Clone a cloud project (coming soon)
vigthoria projects clone <project-id>
```

Syncing with Web IDE

Changes made via CLI can be synced with your cloud workspace:

```
# Push local changes (coming soon)
vigthoria sync push

# Pull cloud changes (coming soon)
vigthoria sync pull
```

Subscription Benefits

Your subscription plan determines: - Available AI models - Token limits per request - Priority processing - Advanced features

---

Command Reference

Core Commands

Command	Alias	Description
`vigthoria chat`	`vig c`	Start interactive chat
`vigthoria agent`	`vig a`	Start agent mode (Claude Code-like)
`vigthoria edit <file>`	`vig e`	Edit file with AI
`vigthoria generate <desc>`	`vig g`	Generate code
`vigthoria explain <file>`	`vig x`	Explain code
`vigthoria fix <file>`	`vig f`	Fix code issues
`vigthoria review <file>`	`vig r`	Review code quality

Chat/Agent Options

Option	Description
`-m, --model <model>`	AI model to use
`-p, --project <path>`	Project context path
`-a, --agent`	Enable agent mode (chat only)
`-r, --resume`	Resume last session
`-l, --local`	Use local Ollama (no auth)
`--auto-approve`	Auto-approve agent actions

In-Session Commands

Command	Description
`/help`	Show all commands
`/file <path>`	Add file to context
`/edit <path>`	Start editing a file
`/diff`	Show pending changes
`/apply`	Apply pending changes
`/model <name>`	Switch AI model
`/agent`	Toggle agent mode
`/approve`	Toggle auto-approve
`/clear`	Clear conversation
`/compact`	Compress old messages
`/sessions`	List saved sessions
`/history`	Show conversation history
`/save`	Save current session
`/new`	Start new session
`/exit`	Exit (auto-saves)

Auth Commands

Command	Description
`vigthoria login`	Authenticate with Vigthoria
`vigthoria logout`	Clear authentication
`vigthoria status`	Show account status

Config Commands

Command	Description
`vigthoria config`	Interactive configuration
`vigthoria config --list`	Show all settings
`vigthoria config --set key=value`	Set a value
`vigthoria config --get key`	Get a value
`vigthoria config --reset`	Reset to defaults
`vigthoria init`	Initialize project config

Global Options

Option	Description
`-V, --version`	Show version number
`-h, --help`	Show help

Workflows & Examples

Workflow 1: Feature Development with Agent Mode

```
# 1. Start agent mode
cd ~/projects/my-app
vigthoria agent

# 2. Let AI implement the feature
you > Create a user authentication system with JWT, including:
- Login and register endpoints
- Password hashing with bcrypt
- JWT token generation and validation
- Protected route middleware

# 3. Agent autonomously:
# - Reads existing code
# - Creates new files
# - Installs dependencies
# - Sets up the feature
```

Workflow 2: Quick Coding Session (Local)

```
# Start local mode (no internet needed)
vigthoria chat --local -m code

you > Write a Python script to batch resize images

# Get instant help without cloud dependency
```

Workflow 3: Resume Yesterday's Work

```
# Resume where you left off
vigthoria chat --resume

# Your previous conversation context is restored
you > Continue implementing the caching layer we discussed

# AI remembers the context
```

Workflow 4: Code Review Before Commit

```
# Create a pre-commit hook
#!/bin/bash
# .git/hooks/pre-commit

for file in $(git diff --cached --name-only --diff-filter=ACM | grep '\.ts$'); do
  # Review each file
  result=$(vigthoria review "$file" -f json)
  score=$(echo "$result" | jq '.score')

  if [ "$score" -lt 70 ]; then
    echo "Code review failed for $file (score: $score)"
    exit 1
  fi
done
```

Workflow 5: Documentation Generation

```
# Generate documentation for a file
vigthoria chat --model creative

you > /file src/core/engine.ts
? Added file to context

you > Generate comprehensive JSDoc documentation for all exported functions

# Apply the documentation
you > /edit src/core/engine.ts
```

Workflow 6: Debugging Session

```
# Start debugging session
vigthoria chat --model code

you > /file src/problematic.ts
you > /file src/tests/problematic.test.ts

you > The tests are failing with "undefined is not a function". Help me debug.

# AI analyzes both files and provides debugging steps
```

Workflow 7: Project Scaffolding with Agent

```
vigthoria agent --auto-approve

you > Create a new REST API project with:
- TypeScript + Express
- PostgreSQL with Prisma ORM
- JWT authentication
- Docker configuration
- GitHub Actions CI/CD
- Jest testing setup

# Agent creates entire project structure automatically
```

Workflow 8: Batch Processing

```
# Fix style issues in all TypeScript files
find src -name "*.ts" -exec vigthoria fix {} -t style --apply \;

# Generate review reports for a directory
for file in src/**/*.ts; do
  vigthoria review "$file" -f markdown >> reviews/$(basename "$file" .ts).md
done
```

Troubleshooting

Common Issues

"Not authenticated" Error

```
# Solution: Login again
vigthoria login

# Or use local mode without authentication
vigthoria chat --local
```

"Subscription expired" Warning Your subscription needs renewal. Visit: https://coder.vigthoria.io/subscriptio

"API connection failed" Error

```
# Check API status
vigthoria status

# Try local mode as fallback
vigthoria chat --local

# Or try alternative endpoint
vigthoria config --set apiUrl=https://backup.coder.vigthoria.io
```

"Model not available" Error The requested model isn't included in your subscription plan.

```
# Check available models
vigthoria status

# Use an available model
vigthoria chat --model mini
```

Ollama Connection Failed (Local Mode)

```
# Make sure Ollama is running
ollama serve

# Check if Ollama is accessible
curl http://localhost:11434/api/tags

# Pull a model if none available
ollama pull phi3:mini
```

Session Not Saving

```
# Sessions auto-save on /exit
# Force save manually:
/save

# Check session storage
ls ~/.vigthoria/sessions/
```

Agent Actions Not Working

```
# Check if agent mode is enabled
/agent

# Verify auto-approve status
/approve
```

Shell Completion Not Working

```
# Bash
source ~/.bash_completion.d/vigthoria

# Zsh - Add to ~/.zshrc
fpath=(~/.zsh/completion $fpath)
autoload -Uz compinit && compinit
```

Debug Mode

For detailed error information:

```
DEBUG=vigthoria* vigthoria chat
```

Reset Everything

```
# Clear config and auth
vigthoria logout
vigthoria config --reset

# Clear sessions
rm -rf ~/.vigthoria/sessions/

# Reinstall
npm uninstall -g vigthoria-cli
npm install -g vigthoria-cli
```

Report Issues

- GitHub: https://github.com/vigthoria/vigthoria-cli/issues
- Discord: https://discord.gg/vigthoria
- Email: support@vigthoria.io

FAQ

General

Q: Is Vigthoria CLI free? A: Basic features are free. Advanced models require a Pro or Ultra subscription. Local mode with Ollama is completely free.

Q: Does it work offline? A: Yes! Use `--local` mode with Ollama installed. Cloud features require internet.

Q: Is my code sent to the cloud? A: In cloud mode, code is sent to Vigthoria servers for AI processing. In local mode (`--local`), everything stays on your machine.

Q: How is this different from ChatGPT? A: Vigthoria CLI is specifically designed for coding: - Project context awareness - Agent mode for autonomous coding - Integrated file editing - Code-optimized models - Terminal-native interface

Technical

Q: What models can I use? A: Depends on your subscription. Check with `vigthoria status`. In local mode, use Ollama models.

Q: Can I use it in CI/CD? A: Yes, use the `VIGTHORIA_TOKEN` environment variable for authentication.

Q: Does it support Windows? A: Yes, via Windows Terminal, PowerShell, or WSL.

Q: How do I update?

```
npm update -g @vigthoria/cli
```

Account

Q: Can I use my Vigthoria Coder account? A: Yes, same account works for CLI, Web IDE, and VS Code extension.

Q: How do I upgrade my subscription? A: Visit https://coder.vigthoria.io/subscription

---

Getting Help

- **Documentation**: https://docs.vigthoria.io/cli
- **Discord Community**: https://discord.gg/vigthoria
- **GitHub Issues**: https://github.com/vigthoria/vigthoria-cli/issues
- **Email Support**: support@vigthoria.io (Pro/Ultra subscribers)

Quick Reference Card

```
????????????????????????????????????????????????????????????????
?                    VIGTHORIA CLI QUICK REFERENCE                ?
????????????????????????????????????????????????????????????????
? BASIC COMMANDS                                                  ?
?   vigthoria chat              Interactive AI chat               ?
?   vigthoria agent             Autonomous agent mode             ?
?   vigthoria chat --local      Local mode (no auth needed)       ?
?   vigthoria chat --resume     Resume last session               ?
????????????????????????????????????????????????????????????????
? CODE COMMANDS                                                   ?
?   vigthoria edit <file>       Edit file with AI                 ?
?   vigthoria generate "desc"   Generate code from description    ?
?   vigthoria explain <file>    Explain code                      ?
?   vigthoria fix <file>        Fix bugs/security/style           ?
?   vigthoria review <file>     Review code quality               ?
????????????????????????????????????????????????????????????????
? IN-SESSION COMMANDS                                             ?
?   /file <path>    Add file      /agent        Toggle agent      ?
?   /edit <path>    Edit file     /approve      Toggle auto-approve ?
```

```
?   /model <name>   Switch model  /sessions     List sessions       ?
?   /clear          Clear chat    /save         Save session        ?
?   /compact        Compress      /history      Show history        ?
?   /help           Show help     /exit         Exit (saves)        ?
??????????????????????????????????????????????????????????????????
? SHORTCUTS                                                         ?
?   vig c = chat    vig a = agent    vig e = edit    vig g = gen   ?
?   vig x = explain vig f = fix      vig r = review                ?
??????????????????????????????????????????????????????????????????
```

Happy Coding with Vigthoria CLI! ?

Built with ? by Vigthoria Labs

Last Updated: January 2026