

8gent Code

The Infinite Gentleman — Autonomous agentic coding powered by local LLMs.

Never hit usage caps again. Run locally via Ollama with BMAD method planning and 40%+ token savings.

License **MIT**

Built with **Bun**

Powered by **Ollama**

version **0.2.0**

 Follow @jamesspalding



What is 8gent?

8gent is an autonomous coding agent that runs entirely on your machine using local LLMs via Ollama. It combines the BMAD method (Breakthrough Method of Agile AI-driven Development) with AST-first code navigation for efficient, intelligent code generation.

Key features:

- 🏠 **100% Local** — No API costs, no rate limits, no data leaving your machine
- 🧠 **BMAD Planning** — Structured task decomposition and execution

- 🎯 **AST-First** — Symbol-level code retrieval (97% token savings)
 - 🛠️ **MCP Support** — Connect external tools via Model Context Protocol
 - 🌐 **Web Search** — Search and fetch web content
 - 🖼️ **Multimodal** — Read images and PDFs
 - 📓 **Notebooks** — Edit Jupyter notebooks
 - 🐛 **LSP Integration** — Code intelligence via Language Server Protocol
 - ⚡ **Parallel Execution** — Run tools concurrently
 - 🧑‍🤖 **Multi-Agent** — Orchestrate subagents for complex tasks
 - 🛡️ **Permissions** — Fine-grained command control
 - ⚙️ **Hooks** — Custom automation triggers
-

Quick Start

Prerequisites

1. **Install Ollama:** <https://ollama.ai>
2. **Pull a model:**

```
ollama pull glm4:latest # or qwen2.5, llama3, mistral, etc.
```

Install

```
# Clone and install
git clone https://github.com/PodJamz/8gent-code.git
cd 8gent-code
bun install

# Create global symlink (run from any folder)
mkdir -p ~/.local/bin
ln -sf "$(pwd)/bin/8gent-cli.sh" ~/.local/bin/8gent

# Add to PATH (if not already)
echo 'export PATH="$HOME/.local/bin:$PATH"' >> ~/.zshrc
source ~/.zshrc
```

Run

```
# Start interactive agent
8gent

# Or run a task directly
8gent "Create a React component with TypeScript and tests"
```

Slash Commands

Command	Description
<code>/model</code>	Show current model
<code>/model <name></code>	Switch to a different model
<code>/models</code>	List available Ollama models
<code>/plan <task></code>	Create a plan without executing
<code>/status</code>	Show model, working dir, history length
<code>/permissions</code>	Show permission config
<code>/hooks</code>	List registered hooks
<code>/skills</code>	List available skills
<code>/tasks</code>	Show task board
<code>/help</code>	Show all commands
<code>/clear</code>	Clear conversation history
<code>/exit</code>	Exit the REPL


Features

BMAD Method Planning

8gent uses structured planning for complex tasks:

User: Build a Next.js site with auth and dark mode

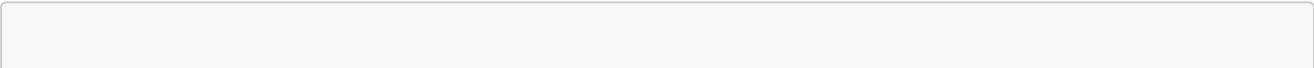
8gent:

 PLAN

1. Initialize Next.js project
2. Set up authentication (NextAuth)
3. Create theme provider (dark/light)
4. Build UI components
5. Add tests and verify

AST-First Code Navigation

Instead of reading entire files, 8gent extracts symbols:



```
# Traditional: Read whole file (2,119 tokens)
cat src/parser.ts

# 8gent: Get just what you need (61 tokens)
8gent outline src/parser.ts
8gent symbol src/parser.ts::buildSymbolId
```

Result: 97% token savings.

MCP Integration

Connect external tools via Model Context Protocol:

```
{
  "servers": {
    "github": {
      "command": "npx",
      "args": ["-y", "@modelcontextprotocol/server-github"]
    }
  }
}
```

Web Search & Fetch

Search the web and extract content:

```
User: Find the latest React 19 features

8gent: *searches web, fetches docs, summarizes*
```

Multi-Agent Orchestration

For complex tasks, 8gent spawns subagents:

```
// Parallel execution of independent tasks
await Promise.all([
  subagent.run("Write unit tests"),
  subagent.run("Update documentation"),
  subagent.run("Generate types")
]);
```

Permission System

Control what commands can run:

```
{
  "allowedPatterns": ["npm *", "git *", "bun *"],
  "deniedPatterns": ["rm -rf /", "sudo rm -rf"],
  "autoApprove": false
}
```

Hooks

Automate workflows with custom hooks:

```
{
  "hooks": [{
    "type": "onComplete",
    "command": "say -v Ava 'Task completed'"
  }]
}
```

Token Savings

Real benchmarks from 8gent's codebase:

Metric	Traditional	AST-First	Savings
Average file read	1,027 tokens	546 tokens	46.8%
Symbol retrieval	2,119 tokens	61 tokens	97.1%
10K operations	\$3,084	\$1,640	\$1,444 saved

Run `8gent benchmark` on your codebase.

Architecture



↓
Evidence Collection & Validation
↓
Completion Report

Core Packages

Package	Purpose
<code>packages/agent</code>	Main agent loop and REPL
<code>packages/ast-index</code>	TypeScript AST parsing
<code>packages/mcp</code>	MCP client implementation
<code>packages/lsp</code>	LSP client for code intelligence
<code>packages/orchestration</code>	Multi-agent coordination
<code>packages/planning</code>	Proactive planning engine
<code>packages/validation</code>	Evidence collection
<code>packages/reporting</code>	Completion reports
<code>packages/permissions</code>	Command permission system
<code>packages/hooks</code>	Automation hooks
<code>packages/skills</code>	Skill framework
<code>packages/tools</code>	Web, PDF, image, notebook tools
<code>packages/personality</code>	The Infinite Gentleman voice
<code>apps/tui</code>	Terminal UI (Ink/React)

Project Structure

```
8gent-code/
├── bin/
│   └── 8gent-cli.sh      # Global CLI entry point
├── apps/
│   └── tui/              # Terminal UI (Ink/React)
├── packages/
│   ├── agent/           # Main agent
│   ├── ast-index/       # AST parsing
│   ├── hooks/           # Hook system
│   ├── lsp/             # LSP client
│   ├── mcp/             # MCP client
│   ├── orchestration/   # Multi-agent
│   └── permissions/     # Permissions
```

```
|— personality/      # Brand voice
|— planning/        # BMAD planner
|— planner/         # Task decomposition
|— reporting/       # Completion reports
|— skills/          # Skill framework
|— tasks/           # Task management
|— tools/           # Web, PDF, image tools
|— toolshed/        # Tool registry
|— types/           # Shared types
|— validation/      # Evidence collection
|— workflow/        # Workflow execution
|— docs/            # Documentation
|— scripts/         # Benchmarks and demos
```

Commands

```
8gent          # Interactive mode
8gent "<task>"  # Run a task directly
8gent outline <file> # Get symbol outline
8gent symbol <id>  # Get symbol source code
8gent search <query> # Search for symbols
8gent benchmark   # Run efficiency benchmarks
8gent demo        # Show token savings demo
```

The Name

8gent combines two ideas:

- **8** → infinity (∞ rotated)
- **gent** → gentleman / agent

An **infinite gentleman**: a disciplined system that grows without increasing prompt size.

Contributing

1. Fork the repo
2. Create your feature branch (`git checkout -b feature/amazing`)
3. Run benchmarks to verify savings (`bun run benchmark`)
4. Commit your changes
5. Push to the branch
6. Open a Pull Request

See [CONTRIBUTING.md](#) for details.

License

MIT © James Spalding

The Infinite Gentleman. Always at your service.

★ [Star on GitHub](#)