Typical installation of Agentecflow

Install HomeBrew

to install npm required for Claude Code - or can download from https://nodejs.org/en/download https://brew.sh

Install npm

Install npm using the command in the terminal (https://phoenixnap.com/kb/install-npm-mac)

> brew install node

Install Claude Code

Install Claude Code https://claude.com/product/claude-code

Install Git tools

Install GitKraken or similar

Clone the ai-engineer repo

Clone https://github.com/Appmilla/ai-engineer

Install the Agentecflow

cd to the ai-engineer/installer/scripts

run the installation script

> ./install.sh

You should see this output (with relevant paths)

```
richwoollcott@macos scripts % ./install.sh


╔════════════════════════════════════════════════════════╗
║         Agentic Flow Installation System               ║
║         Version: 2.0.0                  ║
```

```
╚═════════════════════════════════════════════════════════╝

ℹ Installing Agentic Flow to /Users/richwoollcott/.agenticflow

ℹ Checking prerequisites...
✓ Node.js found: v24.9.0
✓ Python found: Python 3.9.6
✓ All required prerequisites met
⚠ Found existing installations: .claude
ℹ Creating backup of .claude at /Users/richwoollcott/.claude.backup.2025
✓ Backup created: /Users/richwoollcott/.claude.backup.20250928_121752
ℹ Creating complete directory structure...
✓ Complete directory structure created
ℹ Installing global files...
✓ Installed methodology instructions
✓ Installed project templates
✓ Installed commands
✓ Installed documentation
✓ Installed initialization script
✓ Global files installed
ℹ Installing global AI agents...
✓ Installed core global agents
✓ Installed dotnet-microservice stack agents
✓ Installed fullstack stack agents
✓ Installed maui stack agents
✓ Installed python stack agents
✓ Installed react stack agents
✓ Installed typescript-api stack agents
✓ Installed 29 total agents (      10 global +       19 stack-specific)
  Global agents:
    - bdd-generator
    - build-validator
    - code-reviewer
    - database-specialist
    - devops-specialist
    - requirements-analyst
    - security-specialist
    - task-manager
    - test-orchestrator
    - test-verifier
ℹ Creating CLI commands...
✓ Created agentic-init command
✓ Created CLI commands (agenticflow, agentic-init, af, ai)
ℹ Setting up shell integration...
ℹ Detected zsh shell
✓ Shell integration added to /Users/richwoollcott/.zshrc
ℹ Please restart your shell or run: source /Users/richwoollcott/.zshrc
ℹ Creating global configuration...
```

```
✓ Global configuration created
ℹ Installing shell completions...
✓ Shell completions installed
ℹ Setting up version management...
✓ Version management configured
ℹ Setting up cache directories...
✓ Cache directories created

════════════════════════════════════════════════════════
✅ Agentic Flow installation complete!
════════════════════════════════════════════════════════

Installation Summary:
  📁 Home Directory: /Users/richwoollcott/.agenticflow
  🔧 Configuration: /Users/richwoollcott/.config/agenticflow
  📦 Version: 2.0.0

Installed Components:
  🤖 AI Agents:       10
  📋 Templates:        7
  ⚡ Commands:        17

Available Commands:
  • agentic-init [template]  - Initialize a project
  • agenticflow init         - Alternative initialization
  • agenticflow doctor       - Check system health
  • af                       - Short for agenticflow
  • ai                       - Short for agentic-init

Available Templates:
  • default - Language-agnostic
  • dotnet-microservice - .NET microservice with FastEndpoints
  • fullstack - React + Python
  • maui - .NET MAUI mobile app
  • python - Python with FastAPI
  • react - React with TypeScript
  • typescript-api - NestJS TypeScript backend API

⚠ Next Steps:
  1. Restart your shell or run: source ~/.bashrc (or ~/.zshrc)
  2. Navigate to your project directory
  3. Run: agentic-init dotnet-microservice

📚 Documentation: /Users/richwoollcott/.agenticflow/docs/
❓ Check health: agenticflow doctor
richwoollcott@macos scripts %
 Session Restarted
Last login: Sun Sep 28 11:55:38 on ttys000
```

```
/Users/richwoollcott/.agenticflow/completions/agenticflow.bash:30: comman
/Users/richwoollcott/.agenticflow/completions/agenticflow.bash:31: comman
/Users/richwoollcott/.agenticflow/completions/agenticflow.bash:32: comman
/Users/richwoollcott/.agenticflow/completions/agenticflow.bash:33: comman
richwoollcott@macos Desktop % agenticflow --help
Agentic Flow - AI-Powered Software Engineering Lifecycle System

Usage: agenticflow <command> [options]

Commands:
  init [template]     Initialize Agentic Flow in current directory
  doctor              Check system health and configuration
  version             Show version information
  help                Show this help message

Examples:
  agenticflow init                   # Interactive initialization
  agenticflow init react             # Initialize with React template
  agenticflow init dotnet-microservice # Initialize with .NET microservic
  agenticflow doctor                 # Check installation health
richwoollcott@macos Desktop % agentic-init --help
Agentic Flow Project Initialization

Usage: agentic-init [template]

Templates:
  default             - Language-agnostic template
  react               - React with TypeScript
  python              - Python with FastAPI
  maui                - .NET MAUI mobile app
  dotnet-microservice - .NET microservice with FastEndpoints
  fullstack           - React + Python
  typescript-api      - NestJS TypeScript backend API

Examples:
  agentic-init                   # Interactive setup
  agentic-init react             # Initialize with React template
  agentic-init dotnet-microservice # Initialize with .NET microservice
```

Restart the session (terminal)

1. Activate the installation: source ~/.zshrc

or restart your terminal

1. Verify installation health: agenticflow doctor

2. Test CLI commands: agenticflow --help agentic-init --help
3. Test project initialization: mkdir test-project && cd test-project agenticflow init react

   # or

   agentic-init typescript-api

Install Visual Studio Code

https://code.visualstudio.com/docs/setup/mac#_install-vs-code-on-macos

Open the Extensions Tab and install the Claude Code extension and restart

Create an example project

Create a new directory and change to it

```
mkdir test-api
cd test-api
```

Run the init command for the project type, in this example a Typescript Api

```
agentic-init typescript-api
```

You should see output similar to this:

```
richwoollcott@macos test_api % agentic-init typescript-api


╔════════════════════════════════════════════════════════╗
║         Agentic Flow Initialization                   ║
║         Template: typescript-api                           ║
╚════════════════════════════════════════════════════════╝


✓ Found Agentic Flow installation at /Users/richwoollcott/.agenticflow
ℹ Detected project type: unknown
✓ Created test directories
ℹ Created src directory for new project
✓ Smart project structure created
✓ Copied project context file for typescript-api
✓ Copied global AI agents
✓ Copied typescript-api stack agents
✓ Copied template files
✓ Linked Agentic Flow commands
ℹ Creating project configuration...
```

```
✓ Created project configuration
ℹ Creating initial documentation...
✓ Created initial documentation
ℹ Adding TypeScript API-specific configuration...

════════════════════════════════════════════════════════
✅ Agentic Flow successfully initialized!
════════════════════════════════════════════════════════

Project Structure:
  .claude/       - Agentic Flow configuration (single location)
  docs/          - All project documentation (single location)
  tasks/         - Task management (backlog → in_progress → completed)
  epics/         - Epic management (active, archived)
  features/      - Feature management (active, archived)
  portfolio/     - Portfolio metrics and reports
  tests/         - Test suites
  src/           - Source code

Technology-Specific Setup:

For TypeScript API Projects:
1. Install dependencies (if not already done):
   npm install

2. Install testing packages:
   npm install -D jest @types/jest ts-jest
   npm install -D supertest @types/supertest

3. Install recommended packages:
   npm install class-validator class-transformer


Agentecflow Workflow:

Stage 1: Requirements & Planning
   /gather-requirements - Interactive requirements session
   /formalize-ears     - Convert to EARS notation
   /epic-create        - Create epic with PM tool integration

Stage 2: Feature & Task Definition
   /feature-create     - Create feature with epic linkage
   /generate-bdd       - Create test scenarios
   /task-create        - Create implementation tasks

Stage 3: Engineering & Implementation
   /task-work          - Implement with automatic testing
   /task-status        - Monitor task progress
```

```
Stage 4: Deployment & QA
   /task-complete       - Complete task with validation
   /hierarchy-view      - View project hierarchy
   /portfolio-dashboard - Executive overview

📚 Documentation: docs/
📋 Templates: .claude/templates/
⚙️  Configuration: .claude/settings.json

Important:
• Single .claude directory at project root
• Single docs directory for all documentation
• Requirements flow: draft → approved → implemented
```