DEVOPS INTERMEDIATE 3 DAYS

Go for DevOps Engineers Bootcamp

Build a strong foundation in Go for DevOps and SRE automation, from operational fundamentals to high-performance CLI tools. Learn how to build reliable diagnostics, Kubernetes automation, concurrent checks, and portable internal tooling with production-ready packaging and operational best practices.

Why GoLang for DevOps?

Faster DevOps tooling

Build CLIs, automation tools, and operators with strong performance.

Improved automation reliability

Type safety and concurrency support reduce scripting errors.

Stronger platform engineering

Go enables custom controllers and integrations for Kubernetes ecosystems.

Better team capability

DevOps teams can create internal tools instead of waiting on engineering backlogs.

Reduced dependency risks

Go binaries are easy to distribute and run with minimal runtime overhead.

What you'll be able to do

- Understand Go fundamentals required for DevOps tooling and automation.
- Write Go programs to automate operational tasks such as system checks, log parsing, and API monitoring.
- Build production-grade CLI tools with subcommands, flags, clean error handling, and exit codes.
- Work with files, JSON, YAML, and environment variables for configuration-driven automation.
- Call REST APIs reliably with timeouts, retries, and structured outputs.
- Integrate Go automation with DevOps tools such as kubectl and docker CLI.
- Build concurrent automation using goroutines for faster diagnostics and checks.
- Package and distribute tools using cross-compilation, versioning, and release best practices.
- Deliver a complete DevOps CLI tool as a capstone project.

Who this is for

- DevOps Engineers
- SRE Engineers
- Platform Engineers
- Cloud Engineers
- Linux Administrators
- Developers building internal automation tools

Curriculum

01 Go Fundamentals for DevOps Automation

- Why Go is popular for DevOps tooling (static binary, speed, portability)
- Go basics (types, structs, functions, methods)
- Error handling patterns (error returns, wrapping concept)
- Working with packages and modules (go mod basics)
- **Hands-on:** Lab: Build a simple Go tool to print system info and return proper exit codes

02 Automating Ops Tasks (System Checks and Log Parsing)

- Running system checks (CPU/memory/disk basics via Go libraries)
- Parsing logs (regex basics, filtering, aggregation patterns)
- Writing results in structured format (JSON output patterns)
- Building reusable utility functions for automation scripts
- **Hands-on:** Lab: Create a log parser tool that detects errors and outputs a summary report

03 Building Production-Grade CLI Tools (Flags, Subcommands, Exit Codes)

- CLI design principles (clear commands, predictable outputs)
- Subcommands and flags (config, verbose, output formats)
- Clean error messages and consistent exit codes
- Help text, usage patterns, and command validation
- **Hands-on:** Lab: Build a CLI with subcommands (check, parse, monitor) and proper exit behavior

04 Files, Config, and Automation Inputs (JSON, YAML, ENV)

- Reading and writing files safely (paths, permissions, errors)
- Parsing JSON and YAML configuration files
- Environment variables for 12-factor style configuration
- Merging config sources (defaults + env + file + flags)
- **Hands-on:** Lab: Implement config-driven automation using YAML + env overrides

05 Reliable REST API Monitoring (Timeouts, Retries, Structured Output)

- Calling REST APIs using net/http and best practices
- Timeouts, retries, exponential backoff concepts
- Parsing responses and validating status codes
- Structured outputs for automation pipelines (JSON reports)
- **Hands-on:** Lab: Build an API monitoring tool that checks health endpoints and outputs readiness summary

06 Integrating with DevOps Tools (kubectl and docker CLI)

- Executing shell commands from Go safely (os/exec patterns)
- Calling kubectl for cluster checks (pods, nodes, events basics)
- Calling docker CLI for container diagnostics (ps, logs, inspect)
- Capturing outputs and turning them into structured reports
- **Hands-on:** Lab: Build a diagnostics CLI that runs kubectl and docker commands and summarizes results

07 Concurrency for Faster Automation (Goroutines and Channels)

- Goroutines for parallel checks (multiple endpoints/nodes)
- Channels for collecting results and controlling flow
- Context cancellation and timeouts for long-running checks
- Avoiding common concurrency issues (leaks, races concepts)
- **Hands-on:** Lab: Run concurrent health checks across multiple services and aggregate results

08 Packaging and Distribution (Cross-Compile, Versioning, Releases)

- Building static binaries and cross-compiling for multiple OS/architectures
- Versioning strategy (semver, build metadata)
- Release best practices (checksums, changelog notes concept)
- Distributing tools internally (GitHub/GitLab releases concept)
- **Hands-on:** Lab: Package the CLI for Linux/macOS/Windows with version flags and release artifacts

09 Capstone Project (DevOps Automation CLI Tool)

- Capstone goal: Deliver a complete DevOps CLI tool ready for real usage
- Subcommands for system checks, API monitoring, and diagnostics
- Config-driven behavior using YAML/JSON + env + flags
- Concurrent execution for speed and reliability
- Packaged release artifacts with versioning and documentation
- **Hands-on:** Capstone Lab: Build, package, and demo the CLI tool with sample outputs and runbook notes