DevSecOps Scan CLI Documentation

1. Project Overview

- A Go-based CLI tool for automated security scanning.
- Scans **code**, **Docker images**, and **hardcoded secrets**.
- Generates **SBOMs** for compliance.
- Supports **JSON output**, Markdown tables, and CI/CD integration.

---

2. Features

- **Gosec Scan:** Static analysis for Go code.
- **Docker Scan:** Scan one or multiple Docker images for CVEs.
- **Secrets Scan:** Detect hardcoded passwords, tokens, API keys.
- **SBOM Generation:** Generate software bill of materials with Syft.
- **Unified Reporting:** JSON + Markdown console table.
- **CI/CD Friendly:** Exit codes based on `--fail-on-critical`.

---

3. Installation

Requirements

- Go 1.25+
- Syft (for SBOM)

- Docker (for Docker image scanning)

Steps

```
git clone https://github.com/yourusername/devsecops-scan.git
cd devsecops-scan
go build -o devsecops-scan ./cmd/devsecops-scan
```

---

4. Usage

CLI Flags

Flag	Description
`--gosec`	Run Gosec static analysis
`--docker`	Comma-separated Docker images to scan
`--path`	Path to scan (default `.`)
`--json`	Output unified JSON results to file
`--sbom`	Generate SBOM JSON file
`--fail-on-critical`	Exit code 1 if issues ≥ min-severity
`--min-severity`	Minimum severity to fail pipeline (HIGH, MEDIUM, LOW)

Example

```
go run main.go \
  --gosec \
  --docker=myapp:latest,redis:7.0 \
  --path=. \
  --json=results.json \
  --sbom=sbom.json \
  --fail-on-critical
```

---

5. Module Structure

```
devsecops-scan/
│
├── cmd/
│   └── devsecops-scan/
│       └── main.go         # CLI entry
├── internal/
│   ├── code/               # Gosec & secrets
│   │   └── scan.go
│   ├── docker/             # Docker image scanning
│   │   └── scan.go
│   ├── sbom/               # SBOM generation (Syft)
│   │   └── sbom.go
│   └── utils/              # Logging, ScanResult struct, Markdown
│       └── logger.go
├── go.mod
└── go.sum
```

---

6. Example Output

Console Markdown Table

Severity	Scan Type	Target	Line	Package	Details
MEDIUM	GOSEC	./internal/code/scan.go	172		Potential file inclusion via variable
HIGH	DOCKER	myapp:latest		openssl	CVE-2023-xxxx detected
HIGH	SECRET	./config/config.yaml	15		Hardcoded secret: apikey

JSON Output (`results.json`)


```
[
  {
    "scan_type": "GOSEC",
    "target": "./internal/code/scan.go",
    "line": 172,
    "severity": "MEDIUM",
    "details": "Potential file inclusion via variable"
  },
  {
    "scan_type": "DOCKER",
    "target": "myapp:latest",
```

```
    "package": "openssl",
    "severity": "HIGH",
    "details": "CVE-2023-xxxx detected"
  },
  {
    "scan_type": "SECRET",
    "target": "./config/config.yaml",
    "line": 15,
    "severity": "HIGH",
    "details": "Hardcoded secret: apikey"
  }
]
```

---

7. CI/CD Integration Example (GitHub Actions)

```
jobs:
  security_scan:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v3
      - name: Build CLI
        run: go build -o devsecops-scan ./cmd/devsecops-scan
      - name: Run Security Scan
        run: |
          ./devsecops-scan \
            --gosec \
            --docker=myapp:latest,redis:7.0 \
            --path=. \
            --json=results.json \
            --sbom=sbom.json \
            --fail-on-critical
      - name: Upload SBOM
        uses: actions/upload-artifact@v3
        with:
          name: sbom
          path: sbom.json
```