Go Supervisor (GSV) - Project Digest

▯ Core Information

1. **GitHub Repo**: github.com/kolkov/gosv
2. **Language**: Go 1.22+
3. **Status**: Alpha (functional core)
4. **Documentation**: English only (code comments, docs, issues)
5. **Key Features**:
   - Process management with auto-restart
   - YAML configuration
   - TUI interface (tview)
   - Cross-platform (Windows/Linux)
   - Graceful shutdown
   - Hot config reload

▯ Project Plan

Stage	Status	Next Steps
Core Supervisor	▯ Complete	Stabilize APIs
TUI Interface	▯ Complete	Add process control
HTTP API	▯ Not Started	Design REST/gRPC endpoints
WASM Plugins	▯ Not Started	Define plugin interface
Cluster Mode	▯ Not Started	Research Raft consensus
Monitoring	▯ Not Started	Add Prometheus metrics

▯ Project Structure

```
gosv/
├── cmd/
│   └── gosv/           # Main CLI entrypoint
│       └── main.go
├── internal/
│   ├── config/         # YAML config loader
│   │   └── config.go
│   ├── process/        # Process management
│   │   └── manager.go
│   └── supervisor/     # Core logic
│       └── supervisor.go
├── gosv.yaml           # Sample config
├── go.mod
└── go.sum
```

▯ Key Dependencies

1. **TUI**: github.com/rivo/tview
2. **Terminal**: github.com/gdamore/tcell
3. **Colors**: github.com/fatih/color
4. **YAML**: gopkg.in/yaml.v3

▯ Development Notes

1. **Coding Standards**:
    - English comments only
    - `go fmt` enforced
    - Semantic versioning
2. **Build/Run**:

    ```
    # Build
    go build -o gosv.exe ./cmd/gosv
    ```

Run (normal mode)

./gosv -c gosv.yaml

Run (TUI mode)

./gosv -c gosv.yaml -tui

```

### Known Issues

1. Windows process signaling limitations
2. Config reload requires process restart
3. TUI logs panel not implemented
4. No PID file support

### Immediate Next Steps

1. Implement HTTP API (REST)
2. Add Prometheus metrics endpoint
3. Create basic auth for API
4. Write unit tests (70% coverage target)
5. Setup GitHub Actions CI/CD

---

## Project Manifesto

We're building **gsv** as a modern, cloud-native process supervisor that combines:

- Simplicity of classic supervisors
- Scalability of container orchestrators
- Extensibility through WASM plugins

**Design Principles**:

1. **One Binary**: Zero dependencies deployment
2. **DevOps Friendly**: Metrics, APIs, cloud integration
3. **Secure By Default**: Minimal attack surface
4. **Batteries Included**: Built-in useful features

Let's continue building! Open New Terminal Session
```