Learning Go

1.Project Overview

Project Name: Concurrent Web Scraper Engine

Project Goal: To engineer a high concurrency service monitoring engine that utilizes Go's CSP (Communicating Sequential Processes) model to replace traditional sequential polling methods.

Repository: https://github.com/K6EDWIN/AI-Learning-Go-programming

Current Sprint: Completion & Deployment

Status: 🟢 Finished

Outcome: Successfully engineered a high-performance concurrent service monitor that replaces traditional sequential polling with the Go CSP model.

2.Strategic Pillars

Pillar	Focus	Relevance
Concurrency	Goroutines, Channels, & sync package.	Eliminating I/O wait times to scale network requests.
Defensive Design	Error handling and Type safety.	Preventing memory leaks and ensuring security.

Performance	Memory allocation and Pointer optimization.	Maximizing throughput with minimal memory footprint.

3.Engineering Log (Database View)

Date	Module	Concept	Reflection
2026-02-11	Environment	`go mod init`	Initialized the module system. Fundamental for managing the 2026 Go ecosystem.
2026-02-15	The Communication Pipeline	`http.Get` + Channels	Implemented a concurrent status checker. Channels are the pipes for Gopher communication.
2026-02-18	Synchronization	`WaitGroup`	Learning to synchronize multiple concurrent scrapers effectively.

4. Observations

1. The Verbosity is a Guardrail: In Go, checking `if err != nil` isn't just a chore; it's a security audit. It forces you to handle failure at the source rather than letting it crash the system later.
2. Composition over Inheritance: Since Go has no classes, I'm using Structs for data and Interfaces for behaviour.
3. The Feedback Loop: The compilation speed is a game-changer. It allows for an Iterative Design flow that feels closer to scripting but with the power of a compiled language.

5. Final Directory

AI-Learning-Go-programming/
├── bin/ >Compiled executables
├── cmd/
│ └── app.go >The Site Checker entry point
├── documentation/ >Research logs & Notion backups
├── package/ >Reusable internal logic

```
├── go.mod              > Module identity
└── README.md
```

6. Getting Started

1. **Initialize:** `go mod init learning-go`
2. **Execute:** `go run cmd/app.go`
3. **Deploy:** `go build -o bin/sentinel.exe cmd/app.go`