**Go programming courses by for building highly scalable backend systems and**

**microservices with Go ecosystem**

**Course Duration: 3 Days**

**Level: Beginner to Advanced**

**Prerequisites:** Nil (Prior Go skill is not required) Learning **Objectives:**

* Master on Go programming language and its core fundamentals.
* Learn in-depth on Go functions, including functional programming.
* Write highly maintainable and extensible systems with Go's type system.
* Write high-performance, concurrent systems using Go’s concurrency primitives and various concurrency patterns.
* Build production ready web apps and RESTful APIs in Go.
* Testing Go applications.

**Course Outline:**

|  |  |
| --- | --- |
| **Module No.** | **Module Name and Contents** |
| 1. | **Introduction to Go**   * Introduction to Go * The design philosophy of Go * Go Ecosystem * Setting up Go workspace |
| 2. | **Go Language Fundamentals** |
|  | * Core language fundamentals * Functions * Arrays, Slices and Maps * Defer, Panic and Recover * Error handling |
| 3. | **Go Package Ecoystem**   * A deep dive into package ecosystem in Go * Writing packages * Go Tools * Using Go standard library packages * Using third-party packages |
| 4. | **User-Defined Type System**   * A deep dive into Go’s type system * Introduction to Structs and Interfaces * Adding Behaviors to Structs * Value Receivers and Pointer Receivers * Using composition pattern for building data model for Go apps * Using interface for writing idiomatic Go code with Clean Architecture |
| 5. | **Concurrency Programming**   * Concurrency in Go * Goroutines * Channels * Unbuffered Channels * Buffered Channels * Channel Select * Advanced Concurrency patterns |
| 6. | **Mutexes**   * Preventing data race conditions with Mutexes |
| 7. | **HTTP Programming & RESTful APIs**   * A deep dive into Go’s http package * ServeMux and Handler * Creating custom handlers * Extending Go’s http package by using third-party packages * Routing using Gorilla Mux * Writing HTTP middleware * Building real-world RESTful APIs * Securing APIs using JWT |
| 8. | **Go Modules and Project Athens**   * Dependency management in Go * Introduction to Go Modules * Athens: A Go module datastore and proxy |
| 9. | **Testing Go Applications**   * Writing unit tests * Writing Benchmark tests * Testing HTTP applications * Test-Driven Development (TDD) and Behavior-Driven Development |

|  |  |
| --- | --- |
|  | (BDD)   * Writing BDD-styled testing * A deep dive into Ginkgo BDD test framework * Mocking Go interfaces * Mocking with GoMock - A mock framework for Go * Integrating GoMock with Ginkgo BDD test framework |

**Day Wise Training Plan for Professional Go**

|  |  |  |
| --- | --- | --- |
| **Days** | **Modules** | **Highlight of the Topic** |
| Day 1 | Module 1, 2, 3 | Core language fundaments |
| Day 2 | Module 4, 5, 6 | User-defined type system and concurrency |
| Day 3 | Module 7, 8, 9 | HTTP programming and testing |

Note: “Professional Go” can be conducted in two days if the attendees are already familiar on basics of Go programming language.

**Microservices in Go**

**Course Duration: 2 Days Level: Advanced**

**Prerequisites:** Complete the course Professional Go Learning **Objectives:**

* Learn to build massively scalable distributed systems with cloud native and

microservices based architectures.

* Learn event-driven architectures.
* Write Go applications with SOLID principles and Clean architectures.
* Build highly scalable distributed systems and microservices with gRPC, NATS and Go kit.

**Course Outline:**

|  |  |
| --- | --- |
| **Module No.** | **Module Name and Contents** |
| 1. | **A Primer on Microservices and Cloud Native**   * Introduction to Cloud Native |

|  |  |
| --- | --- |
|  | * Introduction to Microservices based distributed systems * Pros and Cons of Microservices * Practical challenges with Microservices |
| 2. | **Decomposition patterns for Microservices**   * Introduction to Domain-Driven Design (DDD) * Building blocks of Domain-Driven Design * Decomposing Microservices with Bounded Context pattern * Building Microservices with Domain-Driven Design Aggregate and Domain Events |
| 3. | **Building high performance APIs using gRPC and Protocol Buffers**   * Introduction to gRPC – A communication patterns for Microservices * Introduction to Protocol Buffers * Building APIs with gRPC and Protocol Buffers * Writing Interceptors in gRPC |
| 4. | **Highly Scalable Cloud Native Messaging for Distributed Systems and Microservices with NATS**   * Introduction to NATS and NATS related messaging patterns * Introduction to NATS Streaming Server * Building distributed systems with NATS |
| 5. | **Event-driven Architetcures**   * Real-world challenges of Microservices with managing transactions and querying data * How to solve the challnges of Microservices with decentralized data * Introduction to event-driven architetcures * Event Sourcing and CQRS * Building Event-Driven Microservices with NATS |
| 6. | **Building Microservices with Go kit**   * Introduction to Go kit: A toolkit for building microservices in Go * Go kit architecture: Services, Endpoints, Transports * Building microservices with Go kit * Writing Go kit midlewares |

**Day Wise Training Plan for Microservices in Go**

|  |  |  |
| --- | --- | --- |
| **Days** | **Modules** | **Highlight of the Topic** |
| Day 1 | Module 1, 2, 3 | An introduction to cloud-native distributed systems architetcures |
| Day 2 | Module 4, 5, 6 | Building distributed systems and Microservices in Go |