Go Developer Lab Exercises Roadmap

Basic Level Lab Exercises

1. **Setup and Hello World**
   o Install Go and configure the environment
   o Write, compile, and run a "Hello, World!" program
2. **Variables and Data Types**
   o Declare variables of different types (int, string, bool)
   o Use constants and iota enumerations
3. **Control Structures**
   o Implement `if`, `else if`, `else` conditions
   o Use `for` loops and `switch` statements
4. **Functions**
   o Define simple functions
   o Use multiple return values and named returns
5. **Arrays, Slices, and Maps**
   o Create and manipulate arrays and slices
   o Use maps for key-value storage
6. **Structs and Methods**
   o Define structs and initialize them
   o Add methods to structs
7. **Basic Error Handling**
   o Return and check errors
   o Use `defer` for cleanup
8. **Basic File I/O**
   o Read from and write to files
9. **Basic CLI Application**
   o Parse command-line arguments

2️⃣Intermediate Level Lab Exercises

1. **Interfaces and Polymorphism**
   o Define interfaces and implement them in structs
   o Use empty interfaces and type assertions
2. **Concurrency Basics**
   o Use Goroutines and channels
   o Implement worker pools

3. **Package Management**
   o Create and use custom packages
   o Manage dependencies with Go Modules
4. **JSON and XML Handling**
   o Marshal and unmarshal JSON and XML data
5. **HTTP Client and Server**
   o Build a simple REST API server with `net/http`
   o Make HTTP requests and handle responses
6. **Testing and Benchmarking**
   o Write unit tests using `testing` package
   o Benchmark functions
7. **Error Wrapping and Custom Errors**
   o Create custom error types
   o Wrap and unwrap errors
8. **Context Package**
   o Use `context` for cancellation and deadlines in Goroutines
9. **Logging**
   o Implement structured logging

3️⃣ Advanced Level Lab Exercises

1. **Advanced Concurrency Patterns**
   o Implement select statement with multiple channels
   o Use sync package: Mutex, WaitGroup, Once
2. **Reflection**
   o Use the `reflect` package for dynamic type inspection and manipulation
3. **Building CLI Tools with Cobra/Viper**
   o Create advanced CLI apps with command hierarchy and configuration management
4. **Database Integration**
   o Connect to SQL (PostgreSQL, MySQL) using `database/sql`
   o Use ORM libraries like GORM
5. **Middleware and Routing**
   o Build custom middleware for HTTP servers
   o Use advanced routers like Gorilla Mux or Echo
6. **Profiling and Performance Optimization**
   o Use `pprof` to analyze CPU and memory usage
   o Optimize slow code paths
7. **GRPC Services**

- o Define protobuf schemas
- o Implement gRPC client and server

8. **Security Practices**
   - o Implement JWT authentication
   - o Use TLS for secure communication

9. **Microservices Architecture**
   - o Design microservices using Go
   - o Service discovery and load balancing

10. **Deploying Go Applications**
   - o Cross-compilation for different OS/architectures
   - o Containerizing with Docker and orchestration with Kubernetes