Track Golang

# 

# Level 0 – Instalasi

*Syarat:*

Mengerti OS laptop yang digunakan

*Referensi:*

<https://go.dev/doc/tutorial/getting-started#install>

<https://dasarpemrogramangolang.novalagung.com/2-instalasi-golang.html>

<https://www.google.com/search?q=tutorial+instalasi+golang>

<https://www.youtube.com/results?search_query=tutorial+instalasi+golang>

Estimasi waktu (dalam jam):

1 - 2 jam (30 menit membaca + 30 menit instalasi).

# Tugas Level 0

Bisa menjalankan command dari command prompt/terminal:

go version

# 

# 

# Level 1 – Dokumentasi

## Muqadimah

*Syarat:*

Bahasa Inggris

*Referensi:*

<https://go.dev/doc/>

*Estimasi waktu*:

2 - 4 jam (2 jam membaca + 30 menit tugas)

*Deskripsi:*

Membaca dokumentasi, tidak harus dibaca semua di awal, kecuali why & getting started.

Prakteknya akan lebih sering ketika implementasi di golang, jika bingung bagaimana cari lah dulu di dokumentasi sebelum googling/stackoverflow dengan jawaban random.

## Hello World

*Referensi:*

<https://go.dev/doc/tutorial/getting-started#code>

## Membaca Cara Coding yang Efektif di Golang

*Referensi:*

<https://go.dev/doc/effective_go>

*Deskripsi:*

Sumber utama ketika bingung cara implementasi coding di golang, tidak harus selesai di modul ini, tetapi bisa menjadi pegangan untuk berikutnya.

## Membaca API SDK & Source Code

*Referensi:*

<https://pkg.go.dev/>

*Deskripsi:*

Memahami cara menggunakan function yang disediakan, beserta opsi-nya .

Ada kalanya ketika membaca API SDK tetap tidak paham, maka disarankan untuk membuka source-nya. Bisa dibuka dari dokumentasi API SDK .

## Library Bawaan (Standard library)

*Referensi:*

<https://pkg.go.dev/std>

## Pelengkap IDE

IDE yang digunakan bebas, bisa goland, vscode atau sublime. IDE vscode atau sublime biasanya memerlukan extension agar dapat mensupport golang, salah satu yang umum diantaranya adalah <https://pkg.go.dev/golang.org/x/tools/gopls>

# Tugas Level 1

1. Bagaimanakah dependency management dalam golang ?
2. Jelaskan kegunaan function fmt.Sprintln apa bedanya dengan fmt.Println, beri contoh code, copas output nya
3. Jelaskan kegunaan function fmt.Errorf apa bedanya dengan errors.New, beri contoh code, copas outputnya

# 

# Level 2 – Dasar Pemrograman Standar

*Syarat:*

Menyelesaikan tour go <https://go.dev/tour/welcome>

*Estimasi waktu*:

16 - 30 jam (4 - 8 jam menyelesaikan tour go, 6 - 8 jam tugas)

## package, import, type, condition, loop

*Referensi:*

<https://go.dev/tour/basics>

## Pointer & Struktur

*Referensi:*

<https://go.dev/tour/moretypes>

## Go Module

*Referensi:*

<https://go.dev/doc/tutorial/create-module>

## Hal Mendasar lain yang perlu di ketahui

* Log
* Error Handling
* nil
* method vs function
* interface
* type assertions

# Tugas Level 2

1. Buatkan HSI NIK generator:

func generateNIK(gender, tahun, jumlah\_yang\_digenerate) []string {

return []string{}

}

generateNIK(‘ikhwan’, 2019, 10)

hanya menggunakan SDK dasar golang

* 1. ARN192-00375:
     1. N: ikhwan
     2. 19: tahun 2019
     3. 2: pembuatan semester akhir, bulan 6-12
     4. 0375: padding 5 digit urutan

1. Sama dengan no 1 tetapi bisa melanjutkan NIK yg sebelumnya, gunakan kembali function yg sudah dibuat di no 1, jika angkatan nil maka, error

func generateNIKLanjutan(‘ARN192-24038’,10) []string{ return []string{}}

func generateNIKLanjutan(nil,10) []string{ return []string{}} → ERROR

1. Setelah generate beberapa angkatan, buatlah function untuk pengelompokan halaqah, admin angkatan sesudahnya adalah ikhwan atau akhwat dengan nip paling kecil, taruh anggota setelah mentor nya.

Contoh:

[‘ARN152-00321’,‘ARN152-00322’,‘ART152-00121’,‘ARN162-00011’,‘ARN162-00012’,‘ARN172-00211’,‘ARN172-00212’]

Menjadi

[‘ARN152-00321’,‘ARN162-00011’,‘ARN172-00211’,‘ARN172-00212’,‘ARN162-00012,‘ARN152-00322’,‘ART152-00121’]

# 

# 

# Level 3 – Koneksi Database

*Syarat*:  
Menguasai RDBMS SQL query standar, bisa CRUD ke database

*Estimasi waktu*:

16 - 32 jam

## DB Standard Library (tetap butuh driver)

<https://pkg.go.dev/database>

<https://go.dev/doc/tutorial/database-access>

## Driver (MySQL)

github.com/go-sql-driver/mysql

## Spesifik database

<https://github.com/jackc/pgx>

## Extensions

<https://github.com/jmoiron/sqlx>

## ORM

<https://gorm.io/index.html>

## Non RDBMS

Ada kalanya kita akan memerlukan non RDBMS, misal kita butuh yang cuma key value based, atau document based, atau write only dan struktur diabaikan, atau datanya sudah mulai besar dan butuh multi-tenancy.

## Catatan

ORM dan raw-SQL memiliki kelebihan dan kekurangan masing-masing, berikut beberapa contoh

* + Menggunakan ORM jika sudah operasi query yang batch dalam jumlah banyak akan kena di performa tetapi jika simpel masih bisa digunakan
  + Menggunakan ORM untuk skala kecil dan memerlukan time to production cepet akan cocok, kecuali sudah punya code generator sendiri untuk raw-SQL
  + Menggunakan raw-SQL akan lebih mudah untuk debugging karena bisa langsung disimulasikan di SQL client

# Tugas Level 3

1. Buatkan 2 tabel dengan struktur 1 to many
   1. Table item: id, name, status, amount
   2. Table item\_detail: id, item\_id, name
2. Buatkan CRUD untuk 2 tabel tersebut, kerjakan dengan sqlx atau gorm
3. Buatkan transaction untuk meng-insert item\_detail dan meng-update item.status dan item.amount, kerjakan dengan sqlx atau gorm
4. Buatkan batch insert untuk no 3, kerjakan dengan sqlx atau gorm

# 

# 

# 

# Level 4 – Messaging Protocol

Untuk selain REST dikenalkan saja, karena di HSI blom dipake

## HTTP REST

<https://pkg.go.dev/net/http>

Untuk mempermudah penggunaan library standard biasanya menggunakan library pembantu, misal untuk mengambil path variable atau ambil payload dsb.

<https://github.com/go-chi/chi> (router)

<https://gofiber.io/> (framework)

## gRPC

<https://grpc.io/docs/languages/go>

## Socket

<https://pkg.go.dev/golang.org/x/net/internal/socket>

## Publish Subscribe

2 tipe: smart subscriber (pull) dan smart publisher (push)

Contoh library yang sudah support ke banyak publish subscribe <https://watermill.io/> bisa dilihat ke dalam source code nya untuk contoh-contoh implementasi pubsub. Tetapi perlu diperhatikan akan lebih baik jika langsung menggunakan library yang sesuai tujuan.

# Tugas Level 4

1. Buatkan REST API untuk CRUD tabel pada tugas level 3
2. Buatkan client untuk menembak service lain dengan protokol REST API

# 

# Level 5 – Web Framework

## Jenis-jenis Web Framework

*Deskripsi:*

Menjelaskan berbagai macam web framework yang biasa digunakan dan jenisnya, apakah dia full framework ataukah micro framework.

* Chi
* Fiber
* Gin
* Echo
* Revel
* Beego

## Fiber

Referensi:

<https://gofiber.io>

### Routing

### Middleware

# Tugas Level 5

1. Buatkan CRUD menggunakan framework Fiber
2. Tambahkan middleware Authorization untuk CRUD tersebut

# 

# Level 6 – Unit Testing

Syarat:

<https://dinda-dinho.blogspot.com/2014/11/perbedaan-unit-test-integration-test.html?m=1>

## Unit Test

<https://go.dev/doc/tutorial/add-a-test>

### Table Driven Test Case

<https://dev.to/boncheff/table-driven-unit-tests-in-go-407b>

### Mocking dan Stubing(fake object)

Karena unit test ada pada level fungsi maka diperlukan mock/stub agar tidak perlu menghidupkan aplikasi atau koneksi I/O ke tempat lain

## Benchmarking

https://pkg.go.dev/testing#hdr-Benchmarks

# Tugas Level 6

1. Buatkan unit test dari fungsi sebelumnya NIK generator
2. Modifikasi generateNIKLanjutan agar dpt mengambil NIK awal dari function lain
3. Buatkan stub dari tugas no 2, dan simulasikan beberapa test case

# 

# 

# Level 7 – Go sebagai worker

Syarat:

Paham cron/scheduler & multi threading

## Go-Routine (concurrency di golang)

<https://go.dev/tour/concurrency/1>

## Scheduler

2 tipe: internal scheduler & external scheduler

# Tugas Level 7

* 1. Ganti perhitungan amount pada table tugas 4 dengan scheduler
  2. Ganti perhitungan amount pada table tugas 4 dengan goroutine

# 

# 

# Level 8 – Project

## Abstraksi/Struktur code

Tidak ada patokan khusus, implementasi di tiap organisasi bisa berbeda-beda, biasanya tergantung sebesar apa aplikasi. Standar utama tetap go.dev perhatikan cara penamaan variable, package dsb.

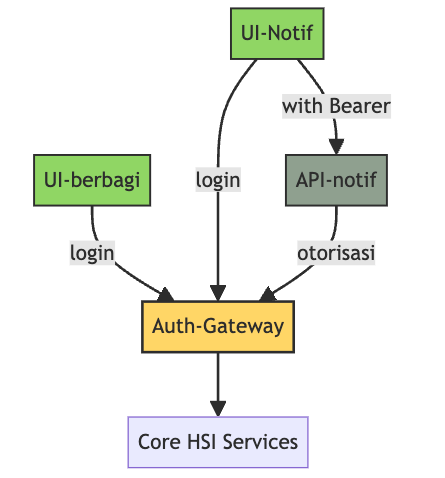
<https://blog.logrocket.com/flat-structure-vs-layered-architecture-structuring-your-go-app>

<https://go-docs.waresix.com/build-rest-api-framework/clean-architecture>

<https://dev.to/bagashiz/building-restful-api-with-hexagonal-architecture-in-go-1mij>

# Tugas Level 8

Studi kasus: Aplikasi autentikasi dan otorisasi

[[1]](#footnote-0)

1. **flowchart** TD

   FE1[UI-berbagi] **-->** |login| AAA[Auth-Gateway]

   FE2[UI-Notif] **-->** |login| AAA

   FE2[UI-Notif] **-->** |with Bearer| BE1

   BE1[API-notif] **-->** | otorisasi | AAA[Auth-Gateway]

   AAA[Auth-Gateway] **-->** BE2[Core HSI Services]

   classDef clientfe fill:#9d6,stroke:#333,stroke-width:1.5px**;**

   classDef clientbe fill:#9a9,stroke:#333,stroke-width:1.5px**;**

   classDef service fill:#fd6,stroke:#333,stroke-width:2px**;**

   class FE1,FE2,FE3 clientfe

   class BE1 clientbe

   class AAA service [↑](#footnote-ref-0)