Swaggo

Introduction

Setelah kita belajar membuat REST API, maka langkah selanjutnya adalah kita akan membuat dokumentasi dasri REST API kita dengan menggunakan library Swagger.

KENAPA SWAGGER ?

Swagger UI memberikan cara yang nyaman bagi consumer untuk menjelajahi API.

Selain itu, API berkembang dari waktu ke waktu dan dokumentasi harus mencerminkan perubahan yang sesuai (Jumlah bug yang muncul karena komunikasi perubahan API yang tidak tepat (atau tidak ada) terlalu tinggi!).

Dengan Swagger, memperbarui / memelihara dokumentasi API sangat mudah - pengembang hanya perlu menambahkan / mengubah anotasi dalam kode, dan perubahannya digabungkan saat dokumen API dibuat berikutnya.

Swaggo dan go-swagger adalah dua framework paling populer yang tersedia untuk menghasilkan dokumen.

Set Up

Install library yang kita perlukan dengan menjalankan perintah berikut :

```
go get -u github.com/swaggo/swag/cmd/swag
go get -u github.com/swaggo/swag/http-swagger
go get -u github.com/alecthomas/template
```

Adding Code Annotation

Kita Bagi seluruh proses dokumentasi API menjadi 3 Langkah :

1. Tambahkan Anotasi Kode

Tambahkan Deskripsi Kode untuk :

A. General API.

```
// @title Orders API
// @version 1.0
// @description This is a sample serice for managing orders
// @termsOfService http://swagger.io/terms/
// @contact.name API Support
// @contact.email soberkoder@swagger.io
// @license.name Apache 2.0
// @license.url http://www.apache.org/licenses/LICENSE-2.0.html
// @host localhost:8080
// @BasePath /
func main() {
        router := mux.NewRouter()
        ...
        log.Fatal(http.ListenAndServe(":8080", router))

}
```

Adding Code Annotation

B. Sekarang, kita telah menambahkan dokumentasi project-level.

Mari kita tambahkan dokumentasi per Endpoint.

Sebagai referensi, saya coba menggunakan [Method : GET] => getOrders API

```
// GetOrders godoc
// @Summary Get details of all orders
// @Description Get details of all orders
// @Tags orders
// @Accept  json
// @Produce  json
// @Success 200 {array} Order
// @Router /orders [get]
func getOrders(w http.ResponseWriter, r *http.Request) {
    w.Header().Set("Content-Type", "application/json")
    json.NewEncoder(w).Encode(orders)
}
```

Adding Code Annotation

Contoh [Method:POST] :

```
// CreateOrder godoc
// @Summary Create a new order
// @Description Create a new order with the input paylod
// @Tags orders
// @Accept  json
// @Produce  json
// @Param order body Order true "Create order"
// @Success 200 {object} Order
// @Router /orders [post]
func createOrder(w http.ResponseWriter, r *http.Request) {
        w.Header().Set("Content-Type", "application/json")
        var order Order
        json.NewDecoder(r.Body).Decode(&order)
        prevOrderID++
        order.OrderID = strconv.Itoa(prevOrderID)
        orders = append(orders, order)
        json.NewEncoder(w).Encode(order)
}
```

Adding Code Annotation

Isi Request Body dijelaskan oleh anotasi **@param**, dengan syntax :

```
@Param [param_name] [param_type] [data_type] [required/mandatory] [description]
```

Adding Code Annotation

Contoh : Values dari Model Attributes

```
// Order represents the model for an order
type Order struct {
        OrderID      string    `json:"orderId" example:"1"`
        CustomerName string    `json:"customerName" example:"Leo Messi"`
        OrderedAt    time.Time `json:"orderedAt" example:"2019-11-09T21:21:46+00:00"`
        Items        []Item    `json:"items"`
}

// Item represents the model for an item in the order
type Item struct {
        ItemID      string `json:"itemId" example:"A1B2C3"`
        Description string `json:"description" example:"A random description"`
        Quantity    int    `json:"quantity" example:"1"`
}
```

Adding Code Annotation

Contoh CreateOrder API :

```
{
  "customerName": "Leo Messi",
  "items": [
    {
      "description": "A random description",
      "itemId": "A1B2C3",
      "quantity": 1
    }
  ],
  "orderId": "1",
  "orderedAt": "2019-11-09T21:21:46+00:00"
}
```

Generate Swagger JSON

```
# In your project dir (~/GOPATH/src/swaggo-orders-api normally)
swag init -g orders.go
```

```
 019/12/02 19:19:43 Generate swagger docs....
2019/12/02 19:19:43 Generate general API Info, search dir:./
2019/12/02 19:19:43 create docs.go at  docs/docs.go
2019/12/02 19:19:43 create swagger.json at  docs/swagger.json
2019/12/02 19:19:43 create swagger.yaml at  docs/swagger.yaml
```

Serving Swagger UI

```
import (
        "encoding/json"
        "log"
        "net/http"
        "strconv"
        "time"

        _ "swaggo-orders-api/docs" // docs is generated by Swag CLI, you have to import it.

        httpSwagger "github.com/swaggo/http-swagger"

        "github.com/gorilla/mux"
)
```

Serving Swagger UI

Sebagai tambahan untuk menspesifikasikan routes pada semua API, kita akan mendefine route di main method menggunakan Pathprefix method.

```go
// @title Orders API
// @version 1.0
// @description This is a sample service for managing orders
// @termsOfService http://swagger.io/terms/
// @contact.name API Support
// @contact.email soberkoder@gmail.com
// @license.name Apache 2.0
// @license.url http://www.apache.org/licenses/LICENSE-2.0.html
// @host localhost:8080
// @BasePath /
func main() {
        router := mux.NewRouter()
        // Create
        router.HandleFunc("/orders", createOrder).Methods("POST")
        // Read
        router.HandleFunc("/orders/{orderId}", getOrder).Methods("GET")
        // Read-all
        router.HandleFunc("/orders", getOrders).Methods("GET")
        // Update
        router.HandleFunc("/orders/{orderId}", updateOrder).Methods("PUT")
        // Delete
        router.HandleFunc("/orders/{orderId}", deleteOrder).Methods("DELETE")

        // Swagger
        router.PathPrefix("/swagger").Handler(httpSwagger.WrapHandler)
        log.Fatal(http.ListenAndServe(":8080", router))

}
```

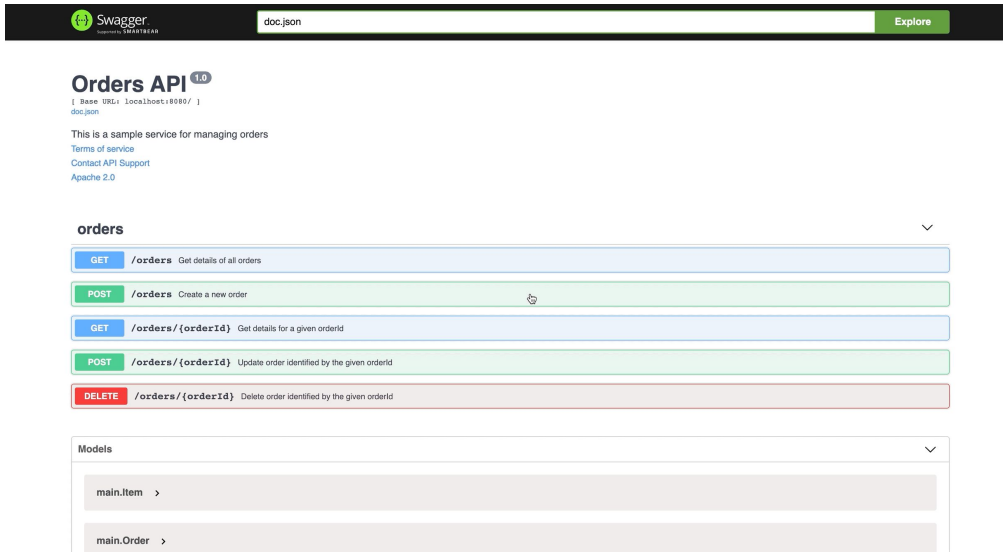
Serving Swagger UI

Setelah Selesai semua , eksekusi dengan jalankan perintah berikut :

```
go run orders.go
```

Serving Swagger UI

Hasil pada browser



Updating Swagger Docs

Dokumentasi API akan selalu Update, maka setiap update kita harus menjalankan :

```
run:
    swag init -g orders.go
    go run orders.go
```

```
swaggo-orders-api$ make run
swag init -g orders.go
2020/06/20 08:12:24 Generate swagger docs....
2020/06/20 08:12:24 Generate general API Info, search dir:./
2020/06/20 08:12:24 Generating main.Order
2020/06/20 08:12:24 Generating main.Item
2020/06/20 08:12:24 create docs.go at docs/docs.go
2020/06/20 08:12:24 create swagger.json at docs/swagger.json
2020/06/20 08:12:24 create swagger.yaml at docs/swagger.yaml
go run orders.go
```