Golang APIRest

Tags Golang

De primeira mão, foi criado a função

```
func HandleRequest() {
    http.HandleFunc("/",nomeFuncao)
    http.ListenAndServe(":8000", nil)
}
```

> → Faça de exercicio, algo bem parecido ao padrao MVC do Golang Aplicação Web do zero.
>
> → Faça um tipo personalidades e declare ela em main.go mocando algumas personalidades.

Quando for usar uma struct que va utilizar json, na declaração da struct,

... Nome string `json:"nome"` ...

🦍**Gorilla Mux: Melhor manipulação de rotas.**

Instalação: procurar no site do goc.

Para usar:

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

func HandleRequest() {
    r := mux.NewRouter()
    r.http.HandleFunc("/api/personalidade/{id}",nomeFunca
o).Methods("Get")
    r.http.ListenAndServe(":8000", r)
}
```

```
----
func RetornaUmaPersonalidade(w http.ResponseWriter, r *htt
p.Request) {
    vars:= mux.Vars(r)
    id := vars["id"]

    for _, personalidade.Id := range models.Personalidades
{
        if strconv.Itoa(peprsonalidade.Id) == id{
            json.NewEncoder(w).Encode(personalidade)
        }
    }
}
```

Modulo JSON:

`json.NewEncoder(w).Encode(models.Personalidades)`

→ Vai encodar um novo Json, e encodar as personalidades.

> Até aqui, deve retornar apenas uma das personalidades baseada no ID. e uma que retorne tudo!

💡 Lembrando: Para mocar os dados:

Crie uma variavel do tipo da struct do model. Exemplo: dentro de personalidade tem o tipo Personalidade, com isso crio uma variavel Personalidades []Personalidade