Федеральное государственное бюджетное образовательное учреждение высшего образования

«Московский государственный технический университет имени Н.Э. Баумана»
(национальный исследовательский университет)
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ

КАФЕДРА КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)

О т ч е т

по лабораторной работе №6

Название лабораторной работы: Основы Back-End разработки на Golang

Дисциплина: Языки интернет-программирования

Студент гр. ИУ6-33Б _____ Цыганчук П. В.
(Подпись, дата) (И.О. Фамилия)

Преподаватель _____ В.Д. Шульман
(Подпись, дата) (И.О. Фамилия)

Москва, 2024

Введение

Цель : Изучение основ сетевого взаимодействия и серверной разработки с использованием языка Golang.

Задание

1) Написать веб сервер, который по пути /get отдает текст "Hello, web!".

Порт должен быть :8080.

2) Написать веб-сервер который по пути /api/user приветствует пользователя:

Принимает и парсит параметр *name* и делает ответ *"Hello,<name>!"*

Пример: /api/user?name=Golang

Ответ: Hello,Golang!

Порт :9000

3) Написать веб сервер (порт :3333) - счетчик который будет обрабатывать GET (/count) и POST (/count) запросы:

GET: возвращает счетчик

POST: увеличивает ваш счетчик на значение (с ключом "count") которое вы получаете из формы, но если пришло НЕ число то нужно ответить клиенту: "это не число" со статусом http.StatusBadRequest (400).

Ход работы:

Задание 1

Используя пакет *net/http* запустим сервер на порту 8080 и с помощью функции *http.HandleFunc* зарегистрируем обработчик.

Ниже представлен листинг:

```
package main
// некоторые импорты нужны для проверки
import (
"fmt"
"io"
"net/http"
"os"
"time"
)
```

```
func handler(w http.ResponseWriter, r *http.Request){
   w.Write([]byte("Hello,web!"))
}
func main() {
   http.HandleFunc("/get",handler)
   err := http.ListenAndServe(":8080",nil)
   if err != nil {
      panic(err)
   }
}
```

Задание 2

С помощью *Query()* можно получить строку запроса и с помощью *.Get()* и *Has()* получить параметр *name*.

Ниже представлен листинг:

```
package main
// некоторые импорты нужны для проверки
import (
"fmt"
"io"
"net/http" // пакет для поддержки HTTP протокола
"os"
"time"
)

func handler(w http.ResponseWriter, r *http.Request){
   name:=r.URL.Query().Get("name")
   fmt.Fprintf(w, "Hello,%s!", name)
}
func main() {
   http.HandleFunc("/api/user",handler)
```

```
    err := http.ListenAndServe(":9000",nil)
    if err != nil {
        panic(err)
    }
}
```

Задание 3

Для выполнения данного задания в теле обработчика сначала определяется, какой пришел запрос, а далее в зависимости от запроса возвращает глобальную переменную *counter* или увеличивает ее на переданное значение.

Ниже представлен листинг

```
package main
// некоторые импорты нужны для проверки
import (
"fmt"
"io"
"log"
"net/http"
"net/url"
"os"
"time"
    "strconv" // вдруг понадобиться вам ;)
)
var counter = 0
func handler(w http.ResponseWriter, r *http.Request) {
    if r.Method == "POST" {
        a, err := strconv.Atoi(r.FormValue("count"))
        if err != nil {
            log.Println(err)
            w.WriteHeader(http.StatusBadRequest)
            w.Write([]byte("это не число"))
```

```
            return
        }
        counter += a
        w.Write([]byte("OK!"))
        return
    } else if r.Method == "GET" {
            w.Write([]byte(strconv.Itoa(counter)))
            return
    }
    w.Write([]byte("Разрешен только метод POST и GET!"))
}
func main() {
    http.HandleFunc("/count", handler)
    err := http.ListenAndServe(":3333", nil)
    if err != nil {
        fmt.Println("Ошибка запуска сервера:", err)
    }
}
```

Вывод:

Изучены основы сетевого взаимодействия и серверной разработки с использованием языка программирования Golang.