



**Министерство науки и высшего образования Российской Федерации  
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)**

**НАПРАВЛЕНИЕ ПОДГОТОВКИ 09.04.01 ИУ6-32Б**

## **ОТЧЕТ**

**по лабораторной работе № 9**

**Название: Back-End разработка с  
использованием фреймворка Echo**

**Дисциплина: Языки интернет-программирования**

Студент

ИУ6-32Б

Юдаков.Г.Д.

\_\_\_\_\_  
(Группа)

\_\_\_\_\_  
(Подпись, дата)

\_\_\_\_\_  
(И.О. Фамилия)

Преподаватель

Шульман В.Д.

\_\_\_\_\_  
(Подпись, дата)

\_\_\_\_\_  
(И.О. Фамилия)

*Цель работы: получение первичных навыков  
использования веб-фрейворков в Backend-  
разработке на Golang*

## Задание 1

Напишите веб сервер, который по пути /get отдает текст "Hello, web!".

Порт должен быть :8080.

Рисунок 1

```

1  package main
2
3  import (
4      "net/http"
5
6      "github.com/labstack/echo/v4"
7      "github.com/labstack/echo/v4/middleware"
8  )
9
10 var count int = 0
11
12 func main() {
13     e := echo.New()
14
15     e.Use(middleware.Logger())
16     e.Use(middleware.Recover())
17
18     e.GET("/hello", getHandler)
19
20     e.Logger.Fatal(e.Start(":2000"))
21 }
22
23 func getHandler(c echo.Context) error {
24     return c.String(http.StatusOK, "Hello, Web!")
25 }
26

```

Рисунок 2

На рисунке 2 показан мой результат.

Есть функция helloHandler, обрабатывающая запрос на “get” которая возвращает “Hello, web!”/

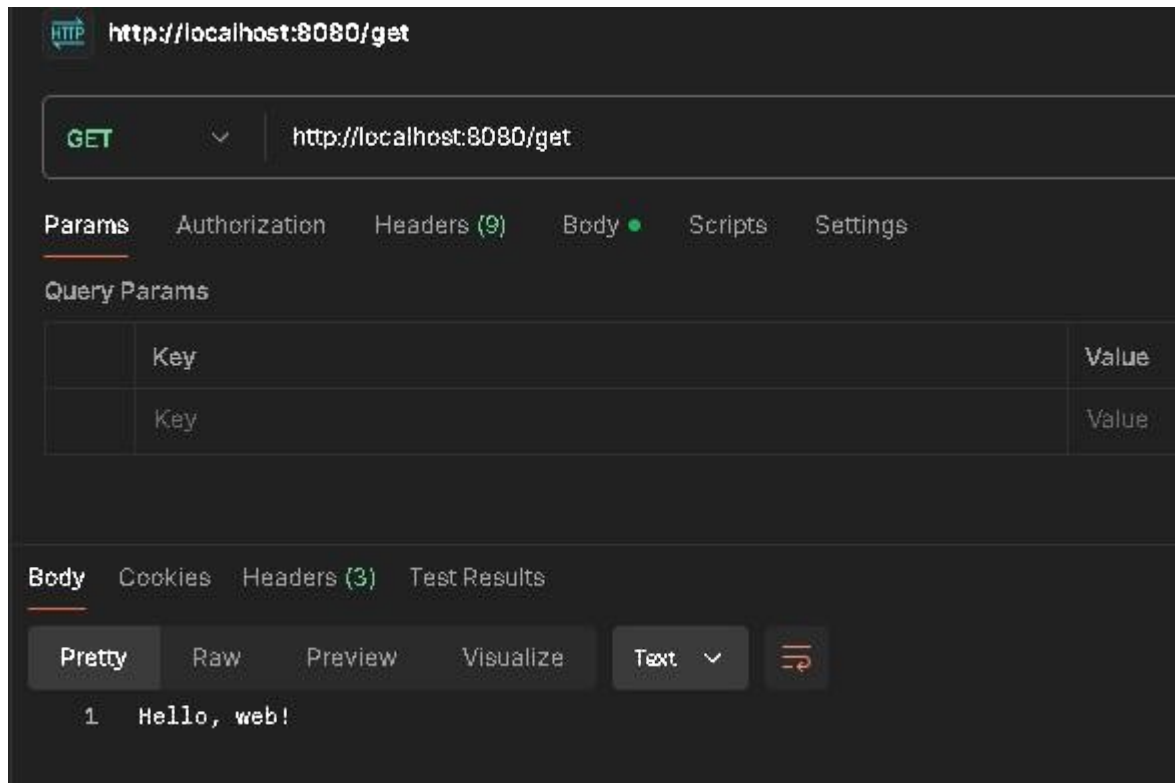


Рисунок 3

На рисунке 3 показан пример вывода.

## Задание 2

Напишите 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!`

Рисунок 4

Как и в предыдущем задании мы обрабатываем запрос `“api/user”` но теперь в запросе присутствует параметр `name` который задается `api/user?name=Stepan`

```
1  package main
2
3  import (
4      "net/http"
5
6      "github.com/labstack/echo/v4"
7      "github.com/labstack/echo/v4/middleware"
8  )
9
10 func main() {
11     e := echo.New()
12
13     e.Use(middleware.Logger())
14     e.Use(middleware.Recover())
15
16     e.GET("/api/user", getHandler)
17
18     e.Logger.Fatal(e.Start(":8000"))
19 }
20
21 func getHandler(c echo.Context) error {
22     name := c.QueryParam("name")
23     if name == "" {
24         name = "Stranger"
25     }
26     return c.String(http.StatusOK, "Hello, "+name+"!")
27 }
```

Рисунок 5

На рис.5 показан мой результат.

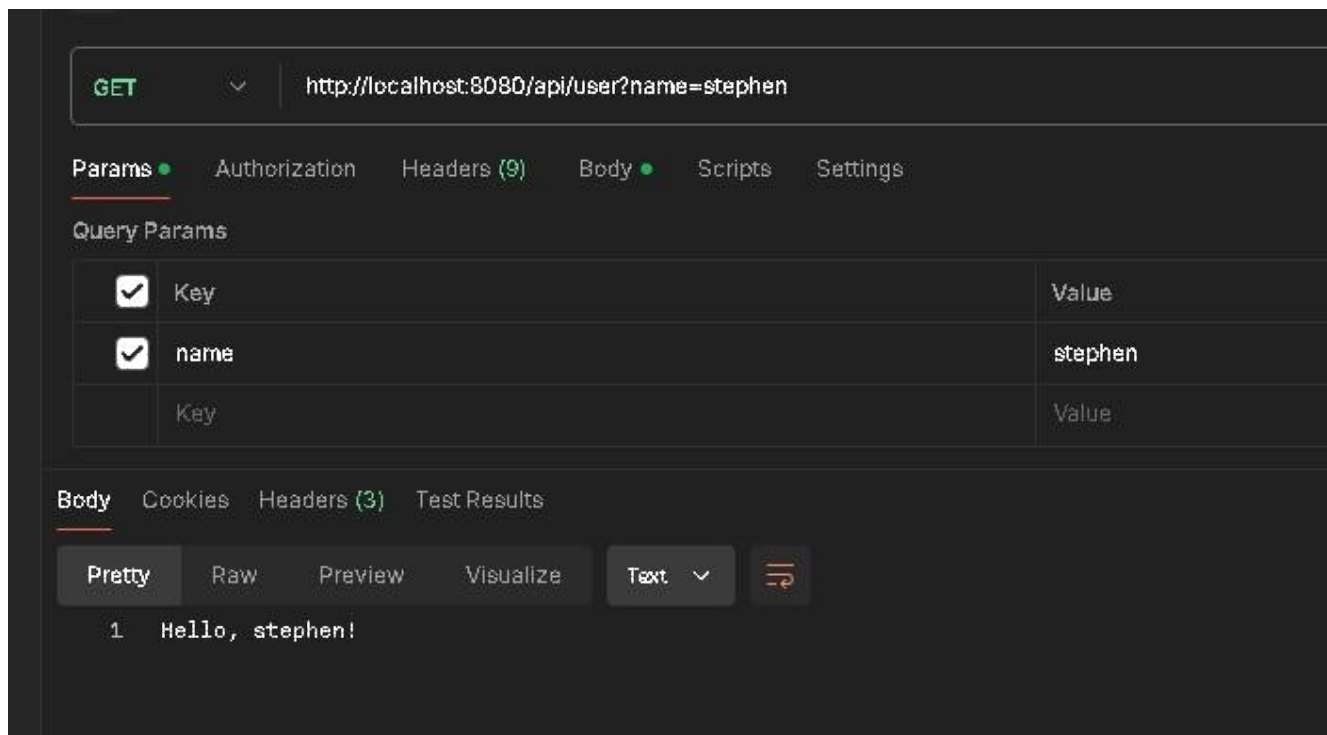


Рисунок 6 (Пример вывода)

## Задание 3

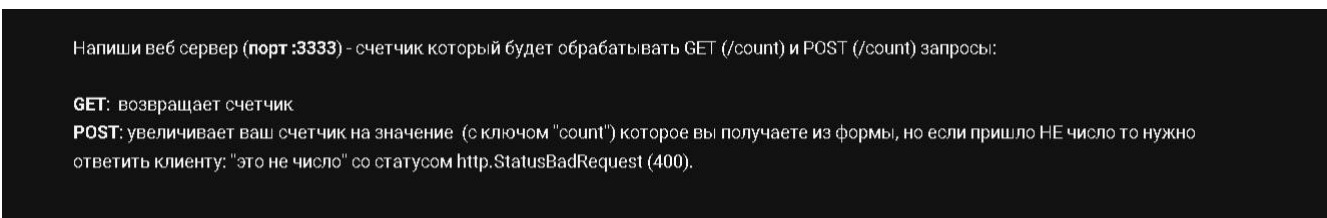


Рисунок 7

Для выполнения задания нам придется создать 2 функции обработчики, для POST и GET метода.

```

18     e.GET("/count", getHandler)
19     e.POST("/count", postHandler)
20
21     e.Logger.Fatal(e.Start(":1323"))
22 }
23
24 type Counter struct {
25     Count int `json:"count"`
26 }
27
28 func getHandler(c echo.Context) error {
29     return c.JSON(http.StatusOK, map[string]int{"count": count})
30 }
31
32 func postHandler(c echo.Context) error {
33     var counter Counter
34     if err := c.Bind(&counter); err != nil {
35         return err
36     }
37
38     count += counter.Count
39
40     return c.JSON(http.StatusOK, counter)
41 }
42

```

Рисунок 8 (Функция для POST запроса)

В ней мы сначала обрабатываем тип запроса и в зависимости от этого выбираем функцию-обработчик.

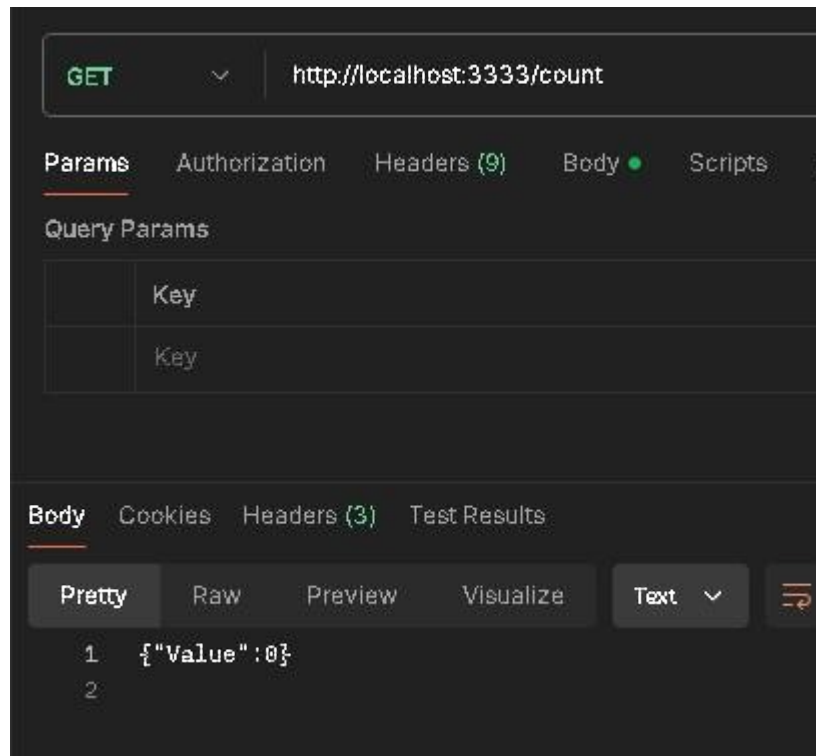


Рисунок 12 (GET запрос)

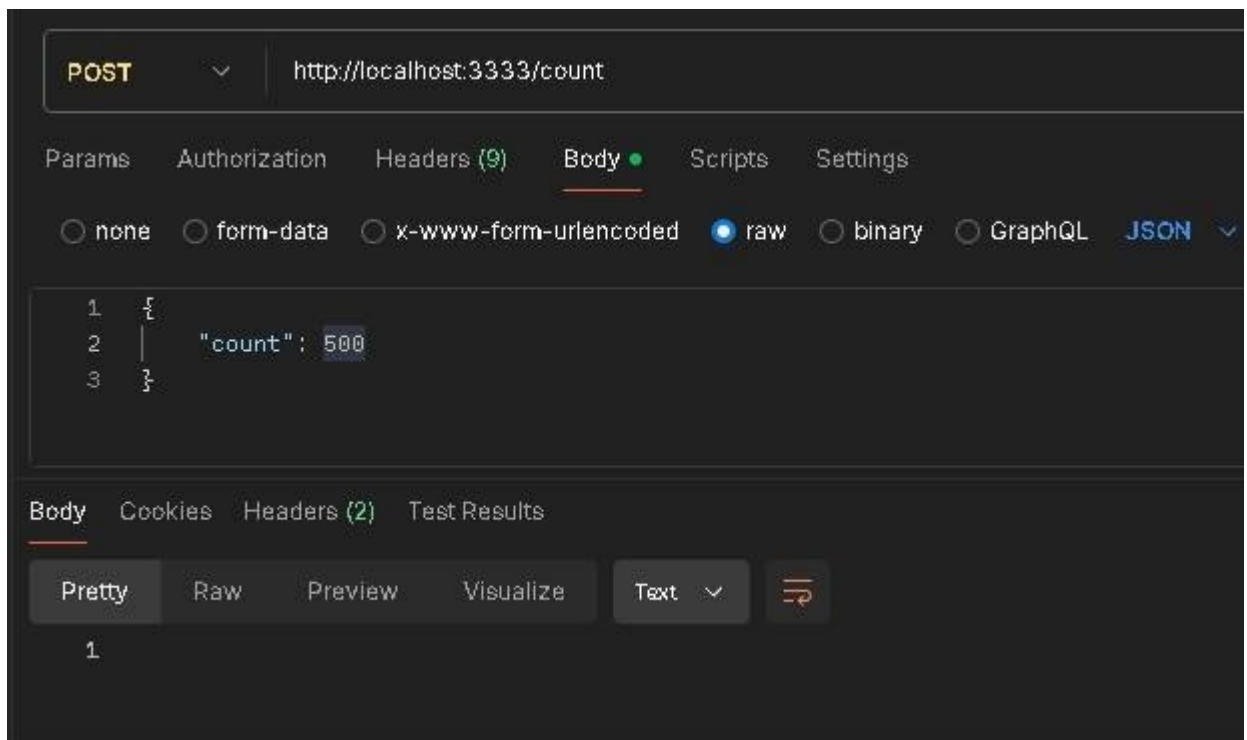


Рисунок 13 (POST запрос)



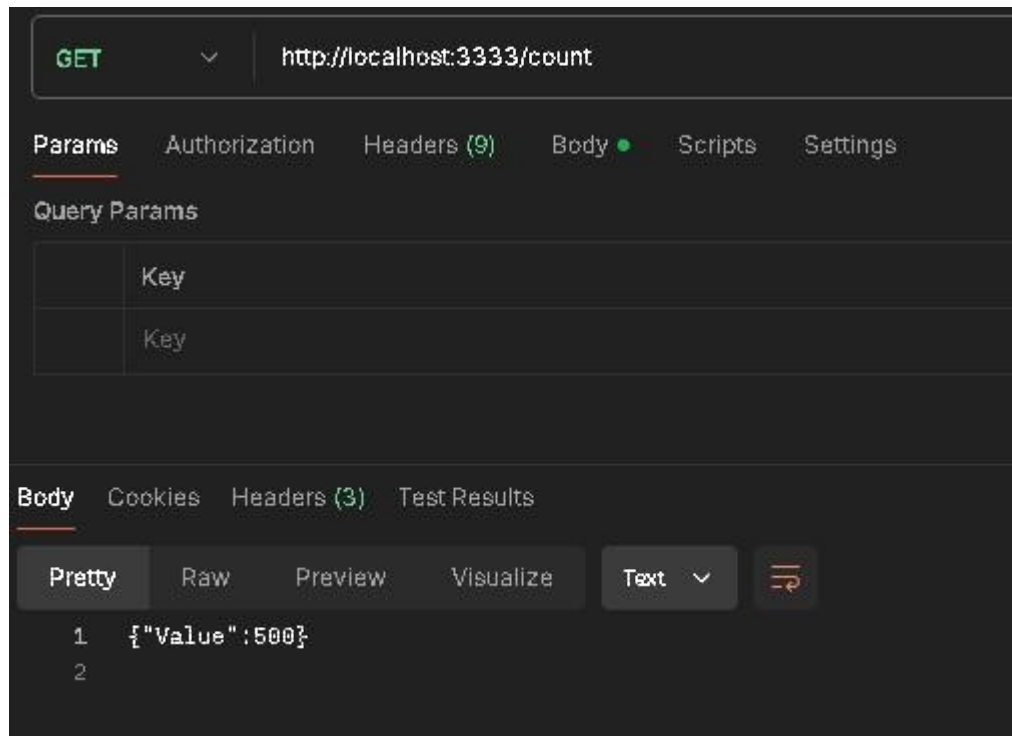


Рисунок 14 (GET запрос)

```
Last login: Tue Nov 19 09:39:34 on ttys008
swiftdeveloper@MacBook-Pro-Stephen web-9 % make lint
/bin/sh: line 0: [: missing `]'
### INSTALL GOLANGCI_LINT ###
[ -f /Users/swiftdeveloper/Desktop/everyting/labs/web-9/bin/golangci-lint ] || curl -sSfL https://raw.githubusercontent.com/golangci/golangci-lint/master/install.sh | sh -s -- -b /Users/swiftdeveloper/Desktop/everyting/labs/web-9/bin v1.59.1
golangci/golangci-lint info checking GitHub for tag 'v1.59.1'
golangci/golangci-lint info found version: 1.59.1 for v1.59.1/darwin/amd64
golangci/golangci-lint info installed /Users/swiftdeveloper/Desktop/everyting/labs/web-9/bin/golangci-lint
### RUN GOALNGCI-LINT ###
/Users/swiftdeveloper/Desktop/everyting/labs/web-9/bin/golangci-lint run ./... --config=../golangci.yaml
```

Рисунок 15(проверка кода с помощью команды make lint)

Заключение: Я научился создавать веб-сервер с помощью фреймворка Echo и обрабатывать GET и POST запросы, при этом получая параметры различными способами.

Москва, 2024