



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 09.03.01 Информатика и вычислительная техника

**О Т Ч Е Т**

**по лабораторной работе № 6**

**Название:** Основы Back-End разработки на Golang

**Дисциплина:** Языки интернет программирования

Студент

ИУ6-32Б

(Группа)

(Подпись, дата)

Д.М. Айдиев

(И.О. Фамилия)

Преподаватель

(Подпись, дата)

В.Д. Шульман

(И.О. Фамилия)

Москва, 2024

**Цель работы:** научиться разрабатывать бэкенд на golang

**Задание:** создать три бэкенда на golang

## Ход работы

Ознакомились с теорией (рис 1)

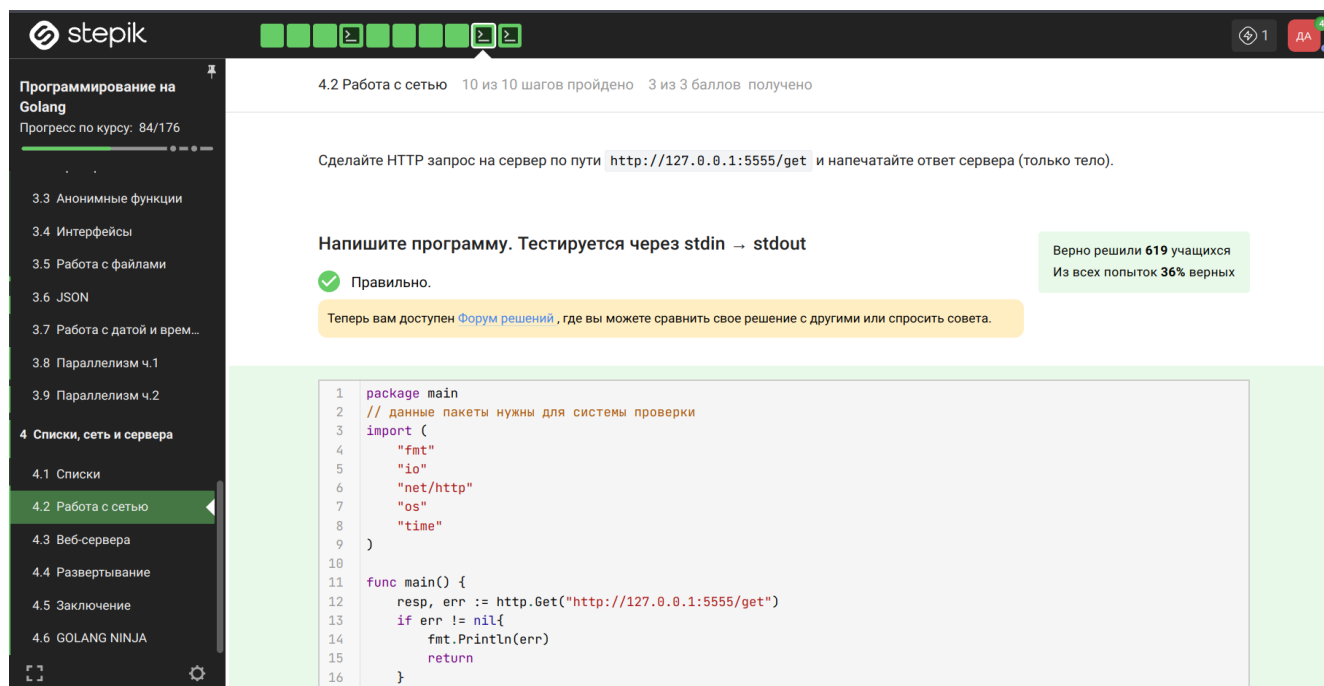


рис 1

Решили задание 1 (рис 2) и проверили через Postman (рис 3).

```
projects > 1_hello > -∞ main.go > 📦 handler
1  package main
2
3  // некоторые импорты нужны для проверки
4  import (
5      "fmt"
6      "net/http"
7  )
8  📌
9  func handler(w http.ResponseWriter, r *http.Request) {
10     w.Write([]byte("Hello, web!"))
11 }
12
13 func main() {
14     http.HandleFunc("/get", handler)
15     err := http.ListenAndServe(":8080", nil)
16     if err != nil {
17         fmt.Println(err)
18     }
19 }
20
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

o (base) jabrail@Jabrlone:~/Labs/Lab6/web-6/projects/1\_hello\$ go run main.go

рис 2

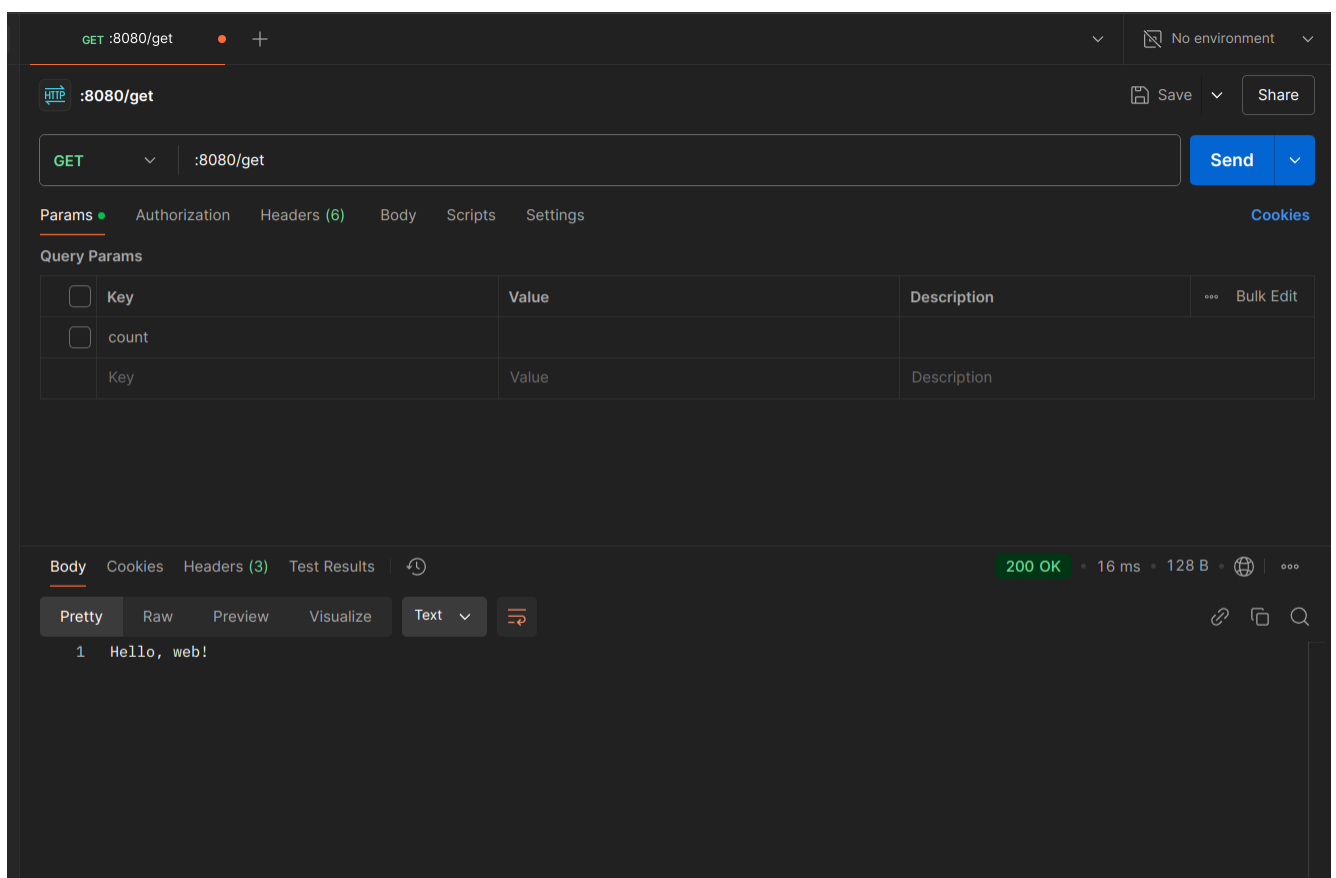


рис 3

Решили задание 2 (рис 3) и проверили (рис 4)

```
1 package main
2
3 // некоторые импорты нужны для проверки
4 import (
5     "fmt"
6     "net/http" // пакет для поддержки HTTP протокола
7 )
8
9 func handler(w http.ResponseWriter, r *http.Request) {
10     w.Write([]byte(fmt.Sprintf("Hello,%s!", r.URL.Query().Get("name"))))
11 }
12
13 func main() {
14
15     http.HandleFunc("/api/user", handler)
16     err := http.ListenAndServe(":9000", nil)
17     if err != nil {
18         fmt.Println(err)
19         return
20     }
21 }
22
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
⊗ (base) jabrail@Jabrailone:~/Labs/Lab6/web-6/projects/2_query$ go run main.go
^Csignal: interrupt
○ (base) jabrail@Jabrailone:~/Labs/Lab6/web-6/projects/2_query$
```

рис 3

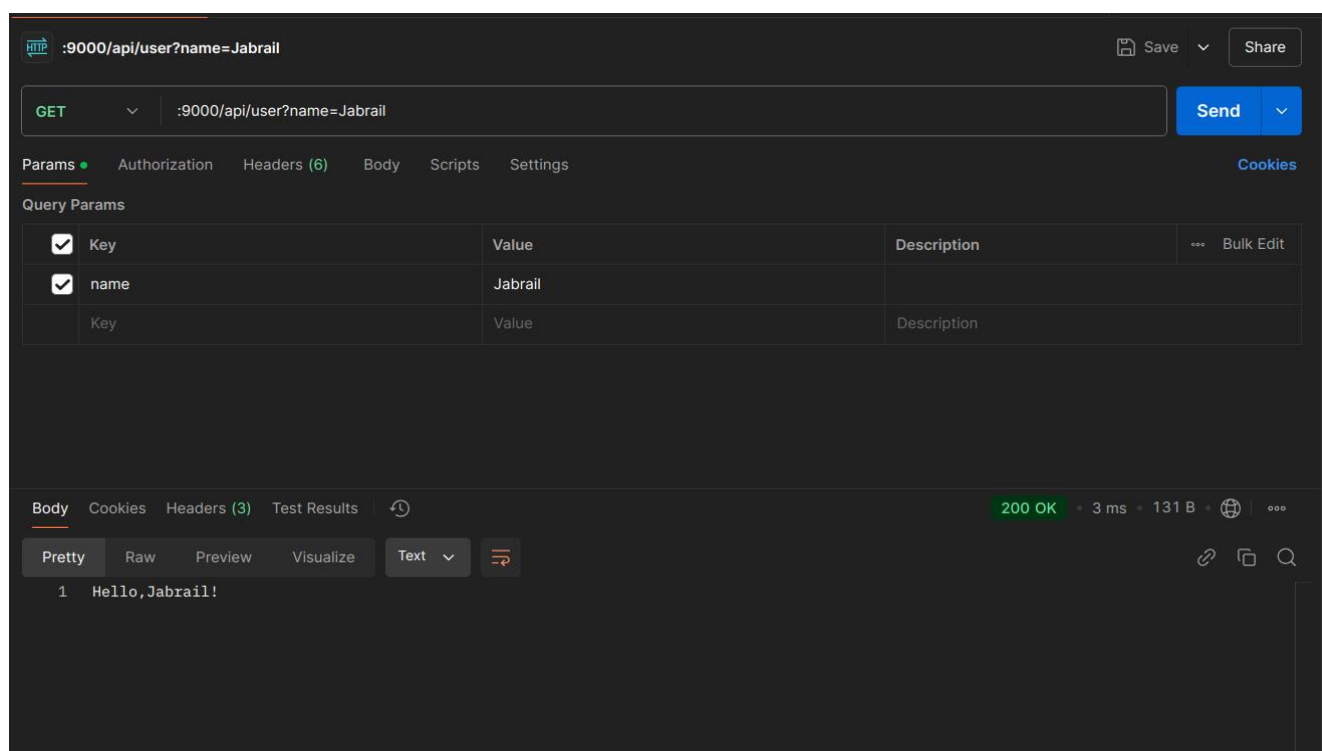


рис 4

Сделали задание 3 (рис 5) и проверили (рис 6, 7)

```

1 package main
2
3 // некоторые импорты нужны для проверки
4 import (
5     "fmt"
6     "net/http"
7     "strconv"
8 )
9
10 var count int
11
12 func handler(w http.ResponseWriter, r *http.Request) {
13     switch r.Method {
14     case http.MethodGet:
15         w.Write([]byte(strconv.Itoa(count)))
16     case http.MethodPost:
17         r.ParseForm()
18         data := r.Form.Get("count")
19         datanum, err := strconv.Atoi(data)
20         if err != nil {
21             w.WriteHeader(400)
22             w.Write([]byte("это не число"))
23             return
24         }
25         count += datanum
26     }
27 }
28
29 func main() {
30     http.HandleFunc("/count", handler)
31     err := http.ListenAndServe(":3333", nil)
32     if err != nil {
33         fmt.Println(err)
34         return
35     }
36 }
37

```

рис 5

PUT :3333/count?count=11

Send

Params Authorization Headers (7) Body Scripts Settings Cookies

Query Params

<input checked="" type="checkbox"/>	Key	Value	Description	...	Bulk Edit
<input checked="" type="checkbox"/>	count	11			
	Key	Value	Description		

Body Cookies Headers (2) Test Results

200 OK • 4 ms • 75 B •

Pretty Raw Preview Visualize Text

1

рис 6

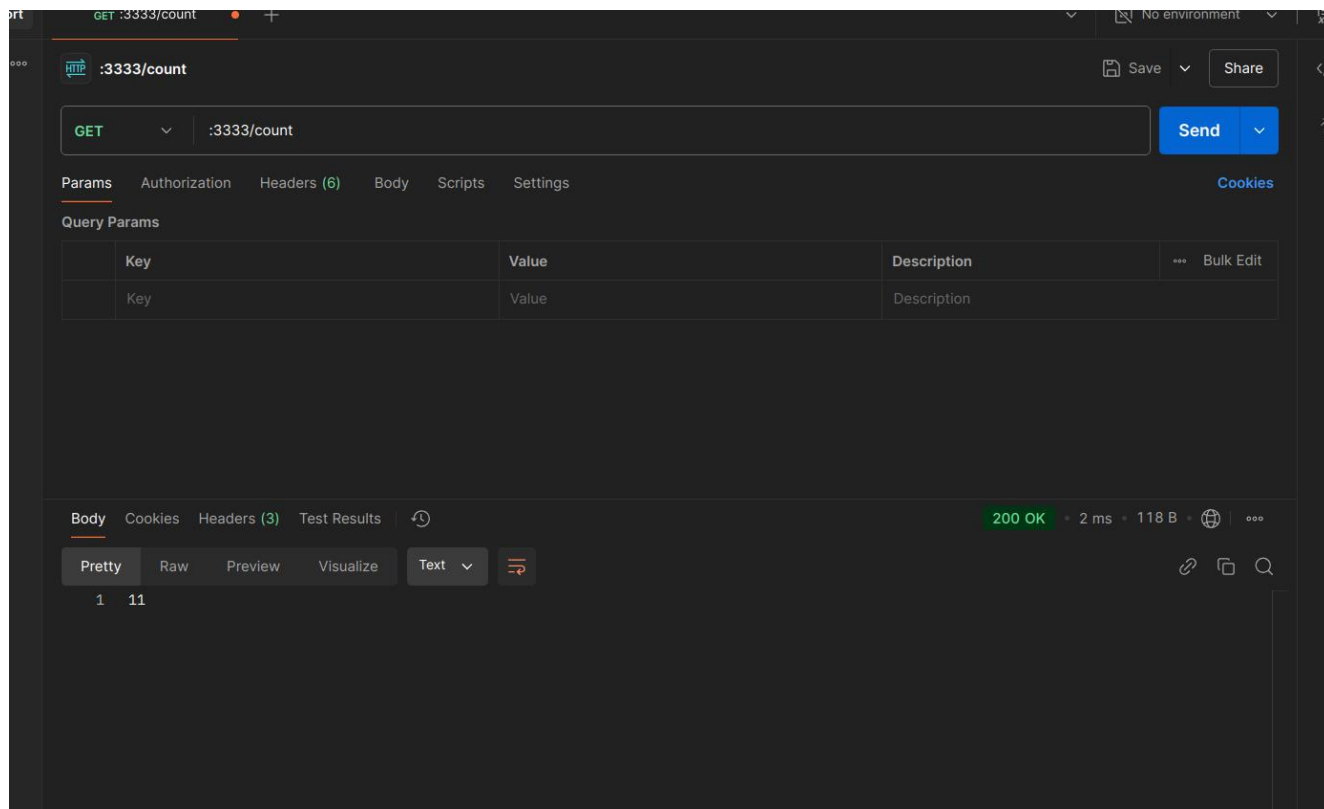


рис 7

**Заключение:** обучились основам создания бэкенда на golang

### **Список использованных источников**

<https://github.com/ValeryBMSTU/web-core/tree/master>