



**«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. ИУ6-32Б \_\_\_\_\_ **А.Д. Шахов**  
(Подпись, дата) (И.О. Фамилия)

Преподаватель \_\_\_\_\_  
(Подпись, дата) (И.О. Фамилия)

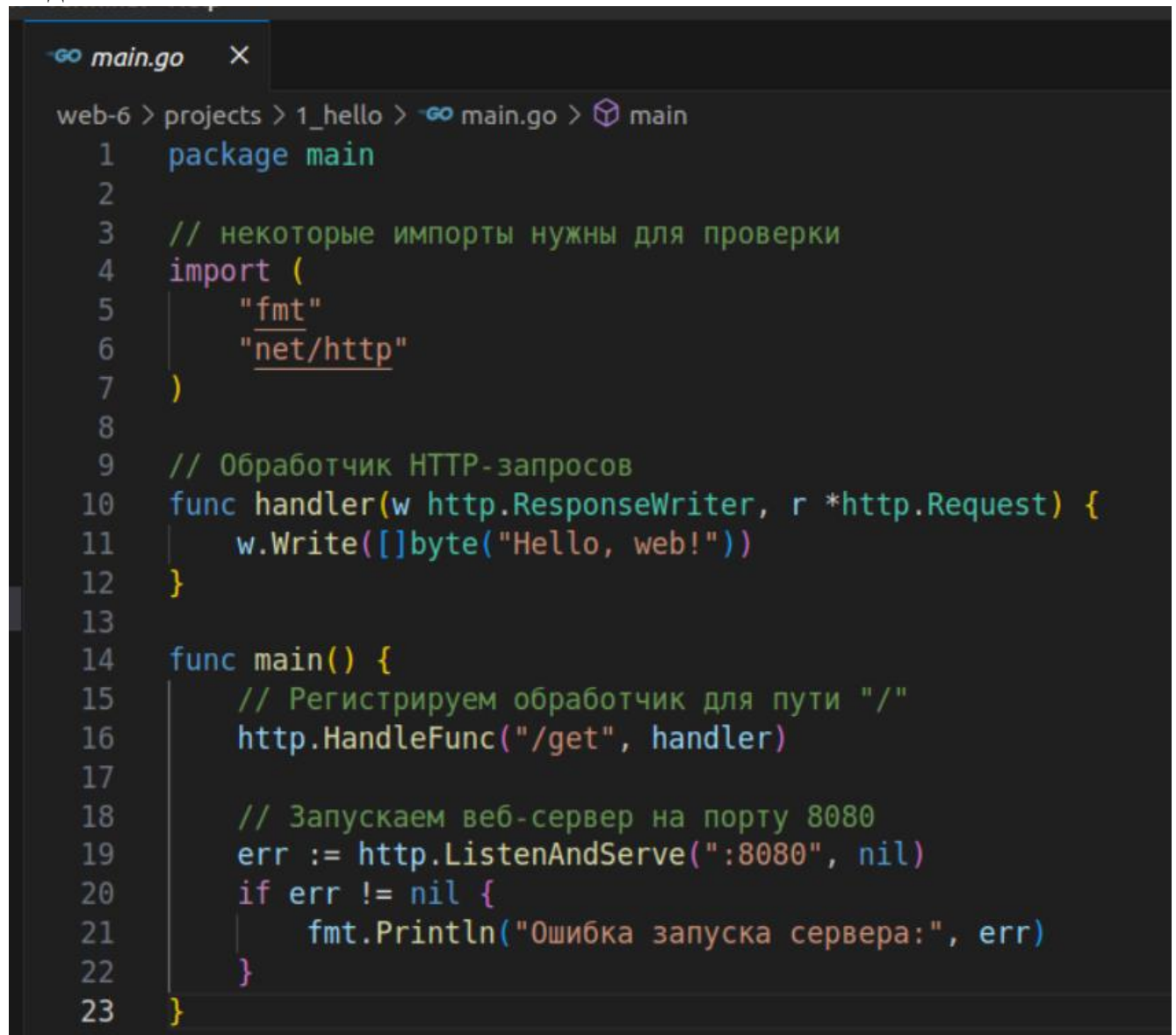
Москва, 2024

Цель работы — изучение основ сетевого взаимодействия и серверной разработки с использованием языка Golang.

В рамках данной лабораторной работы предлагается продолжить изучение Golang и познакомиться с набором стандартных библиотек, используемых для организации сетевого взаимодействия и разработки серверных приложений.

### Задание 1\_hello

Код:



```
main.go ×
web-6 > projects > 1_hello > main.go > main
1  package main
2
3  // некоторые импорты нужны для проверки
4  import (
5      "fmt"
6      "net/http"
7  )
8
9  // Обработчик HTTP-запросов
10 func handler(w http.ResponseWriter, r *http.Request) {
11     w.Write([]byte("Hello, web!"))
12 }
13
14 func main() {
15     // Регистрируем обработчик для пути "/"
16     http.HandleFunc("/", handler)
17
18     // Запускаем веб-сервер на порту 8080
19     err := http.ListenAndServe(":8080", nil)
20     if err != nil {
21         fmt.Println("Ошибка запуска сервера:", err)
22     }
23 }
```

Запускаем в браузере (рис. 1)

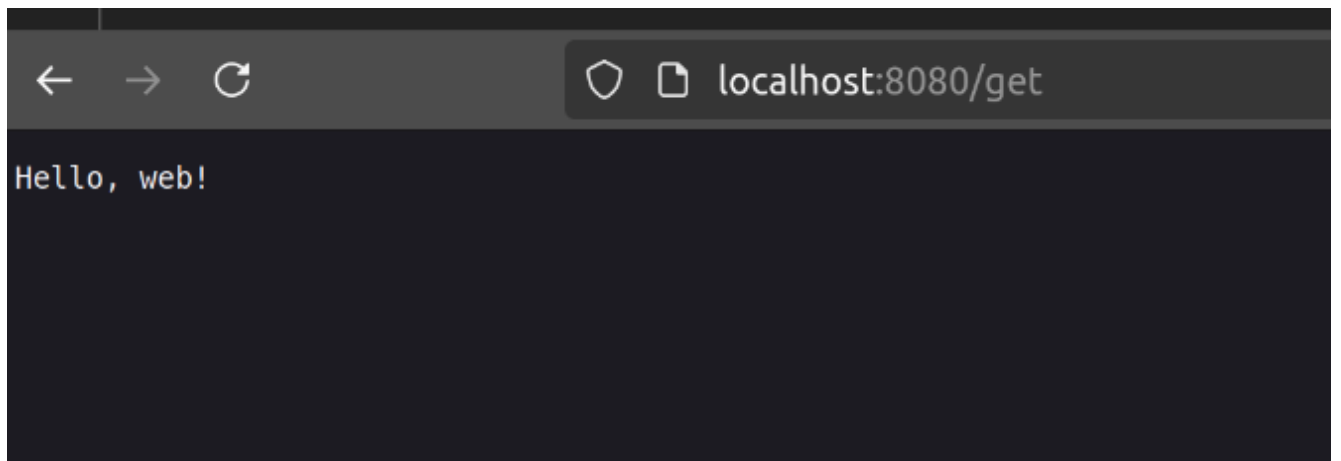


Рис. 1

## Задание 2\_query

Код:

```
package main

import (
    "fmt"
    "net/http"
)

// Обработчик HTTP-запросов
func handler(w http.ResponseWriter, r *http.Request) {
    w.Write([]byte("Hello " + r.URL.Query().Get("name") + "!"))
}

func main() {
    // Регистрируем обработчик для пути "/"
    http.HandleFunc("/api/user", handler)

    // Запускаем веб-сервер на порту 8080
    fmt.Println("starting server...")
    err := http.ListenAndServe(":9000", nil)
    if err != nil {
        fmt.Println("Ошибка запуска сервера:", err)
    }
}
```

Запускаем в Браузере (рис. 3)

Рис. 3

И проверяем что при поиске в браузере по url

<http://localhost:9000/api/user?name=polink> у нас все работает (рис. 4)

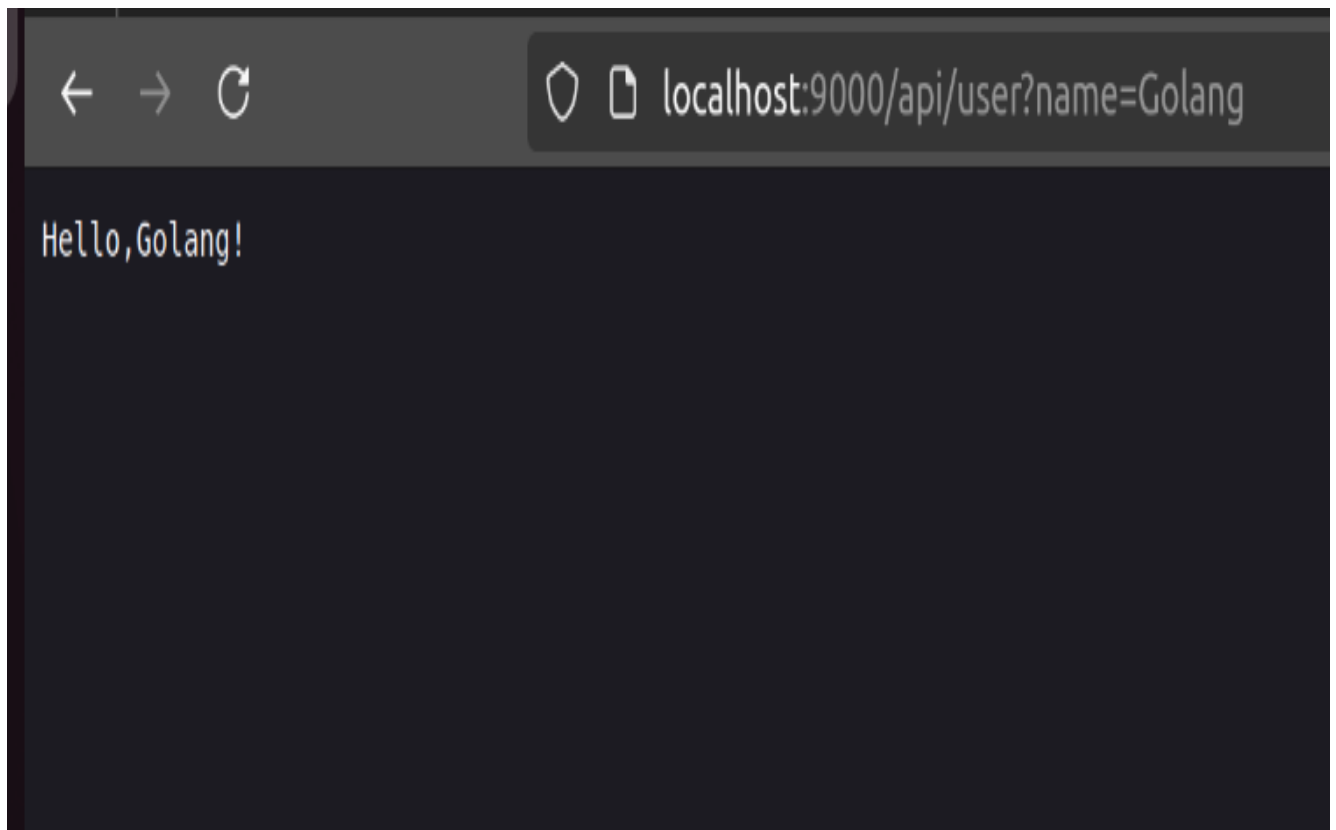


Рис. 4

Задание 3\_count

Код:

```
package main

import (
    "fmt"
    "net/http"
    "strconv"
)

var count int

// Обработчик HTTP-запросов
func handler(w http.ResponseWriter, r *http.Request) {
    if r.Method == "POST" {
        r.ParseForm()
        number_str := r.Form.Get("count")
        number, err := strconv.Atoi(number_str)
        if err != nil {
            w.WriteHeader(http.StatusBadRequest)
            w.Write([]byte("это не число"))
            return
        }
        count = count + number
    } else if r.Method == "GET" {
        w.Write([]byte(strconv.Itoa(count)))
    }
}

func main() {
    // Регистрируем обработчик для пути "/"
    http.HandleFunc("/count", handler)
```

```
// Запускаем веб-сервер на порту 8080
fmt.Println("starting server...")
err := http.ListenAndServe(":3333", nil)
if err != nil {
    fmt.Println("Ошибка запуска сервера:", err)
}
```

Запускаем в Postman (рис. 5–6)

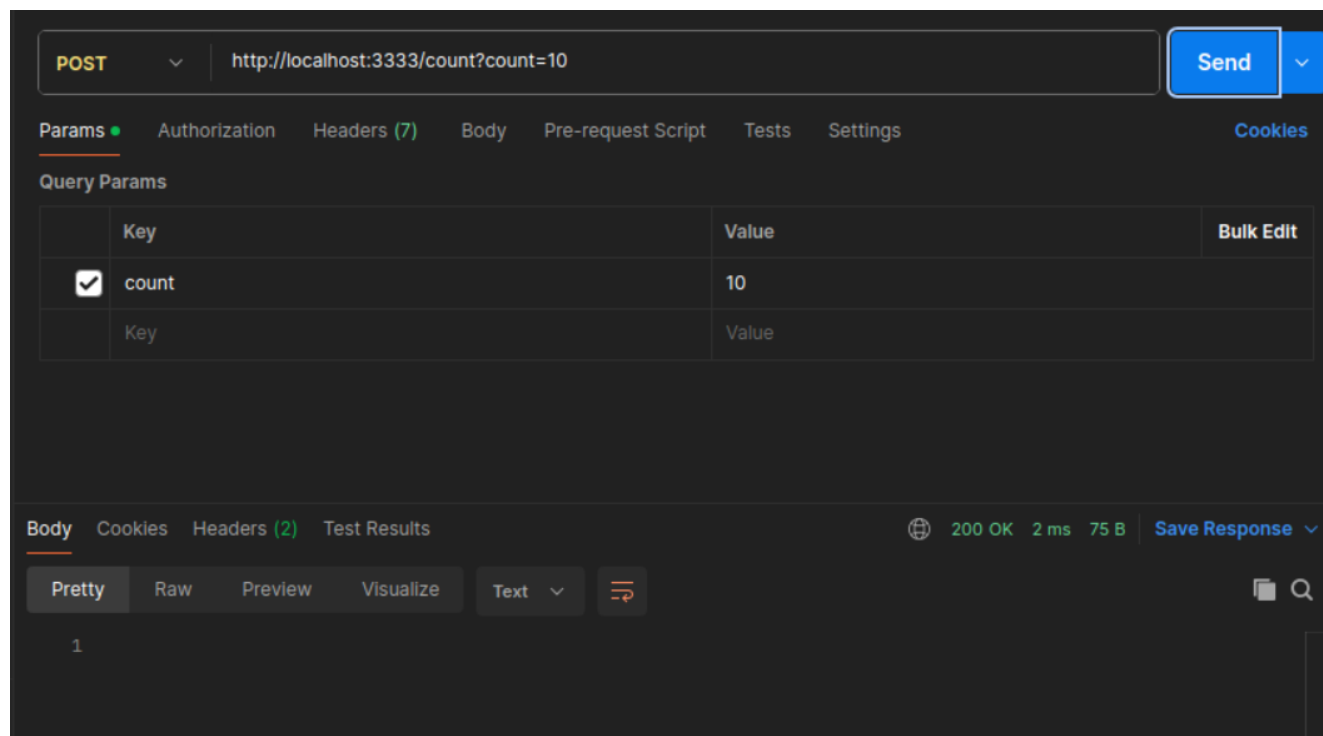


Рис. 5

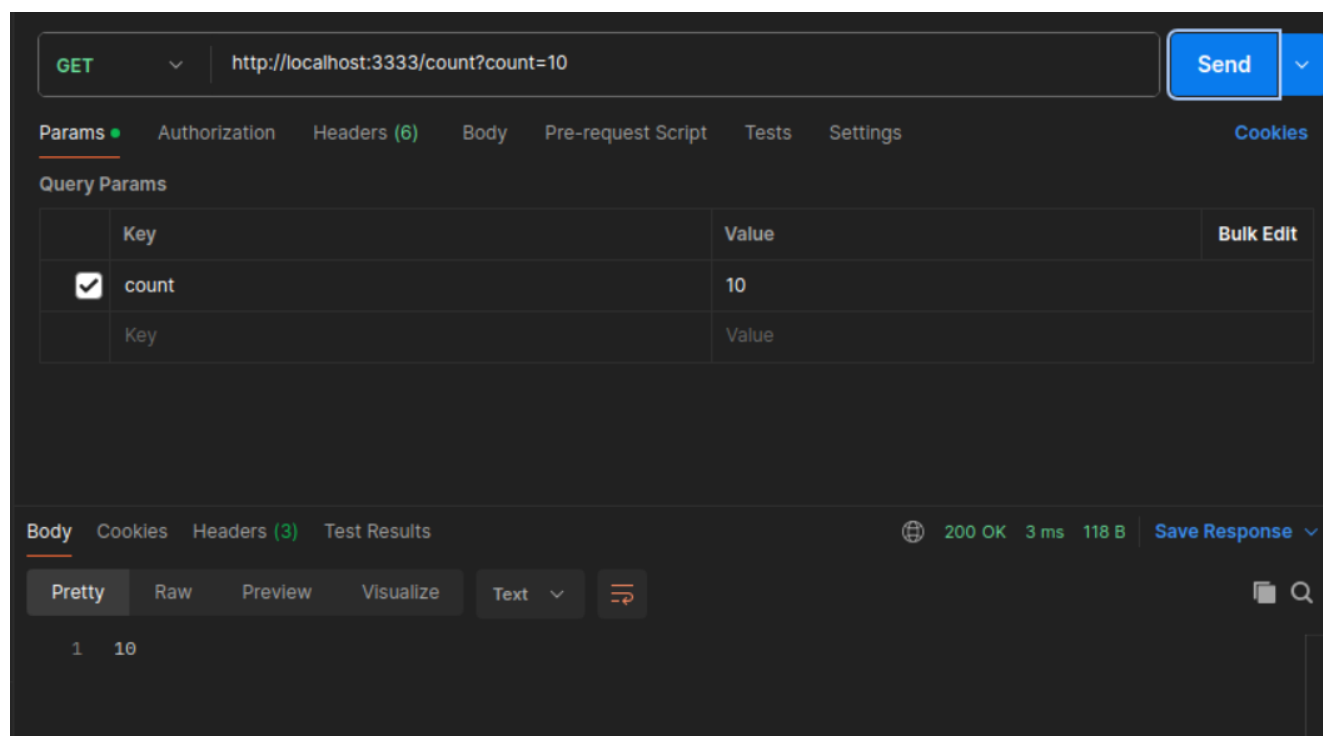


Рис. 6