|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | **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)**

**Отчет**

**по лабораторной работе № 6**

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

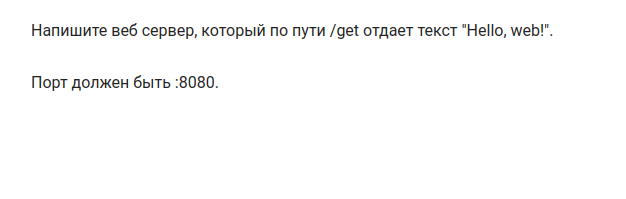
**Дисциплина: Языки интернет программирования**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ6-32Б |  |  | Кирикович М.А |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | И.О. Фамилия |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2024

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

**Часть 1**

Рисунок 1: Задание

1 **package** main

2

3 *// здесь надо написать код*

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

16 err := http.ListenAndServe(":8083", **nil**)

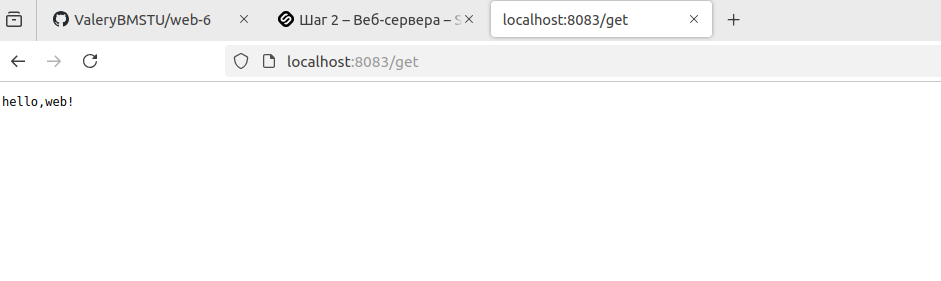
17 **if** err != **nil** {

18 fmt.Println("ошибка запуска сервера")

19 }

20 }

21

Рисунок 2: Результат

**Часть 2**

package main

import (

"fmt"

"net/http"

)

func handler(w http.ResponseWriter, r \*http.Request) {

inputname := r.URL.Query().Get("name")

w.Write([]byte("Hello," + inputname))

}

func main() {

http.HandleFunc("/api/user", handler)

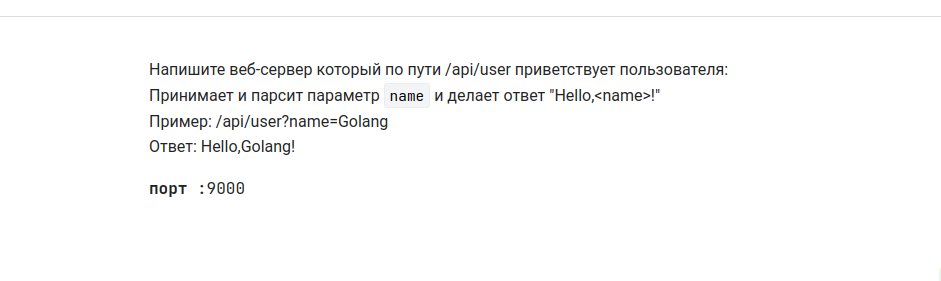
err := http.ListenAndServe(":9000", nil)

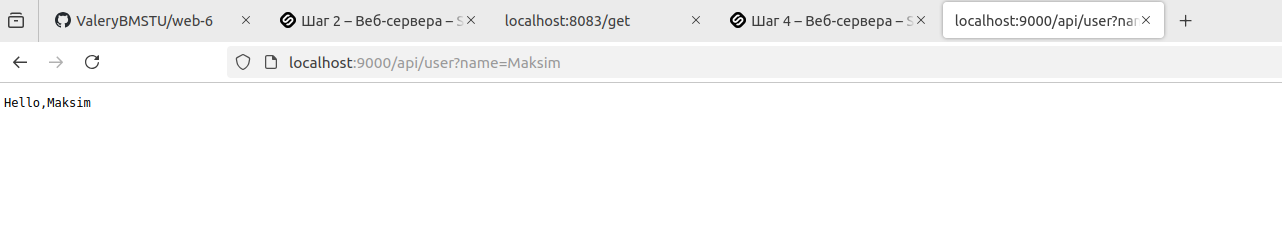
if err != nil {

fmt.Println("ошибка запуска сервера")

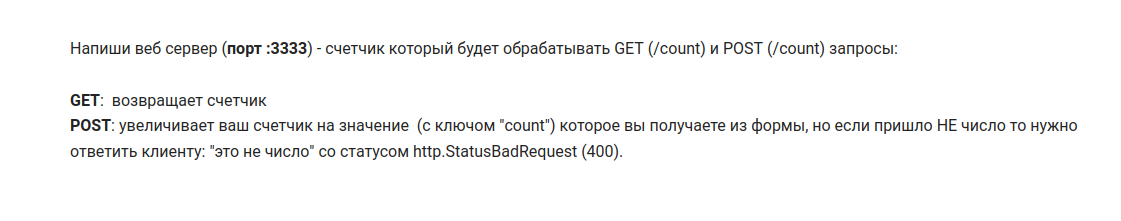
}

}

Рисунок 3: Задание 2

Рисунок 4: Результат тестирования

**Часть 3**

Рисунок 5: задание 3

1 **package** main

2

3 **import** (

4 "encoding/json"

5 "fmt"

6 "net/http"

7 )

8

9 **var** counter int = 0

10

11 **type** kol **struct** {

12 Count int `json:"count"`

13 }

14

15 **func** handler(w http.ResponseWriter, r \*http.Request) {

16 **switch** r.Method {

17 **case** "GET":

18 w.Header().Set("Content-Type", "application/json")

19 json.NewEncoder(w).Encode(**map**[string]int{"count": counter})

20

21 **case** "POST":

22 **var** c kol

23 err := json.NewDecoder(r.Body).Decode(&c)

24 **if** err != **nil** {

25 http.Error(w, err.Error(), http.StatusBadRequest)

26 }

27 fmt.Println(c.Count)

28 counter += c.Count

29 **default**:

30 http.Error(w, "неправильный запрос", 405)

31 }

32 }

33 **func** main() {

34 http.HandleFunc("/count", handler)

35

36 err := http.ListenAndServe(":3333", **nil**)

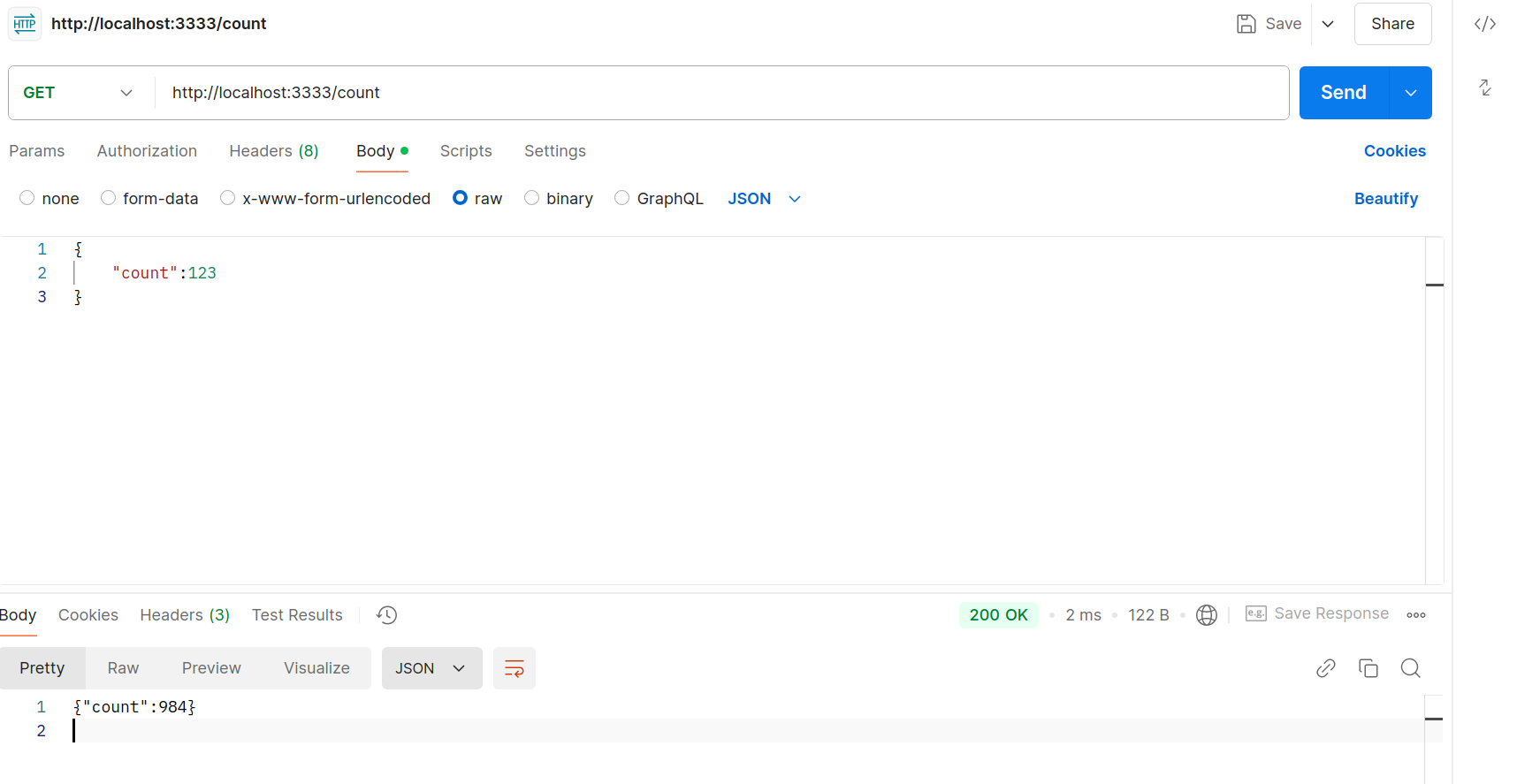
37 **if** err != **nil** {

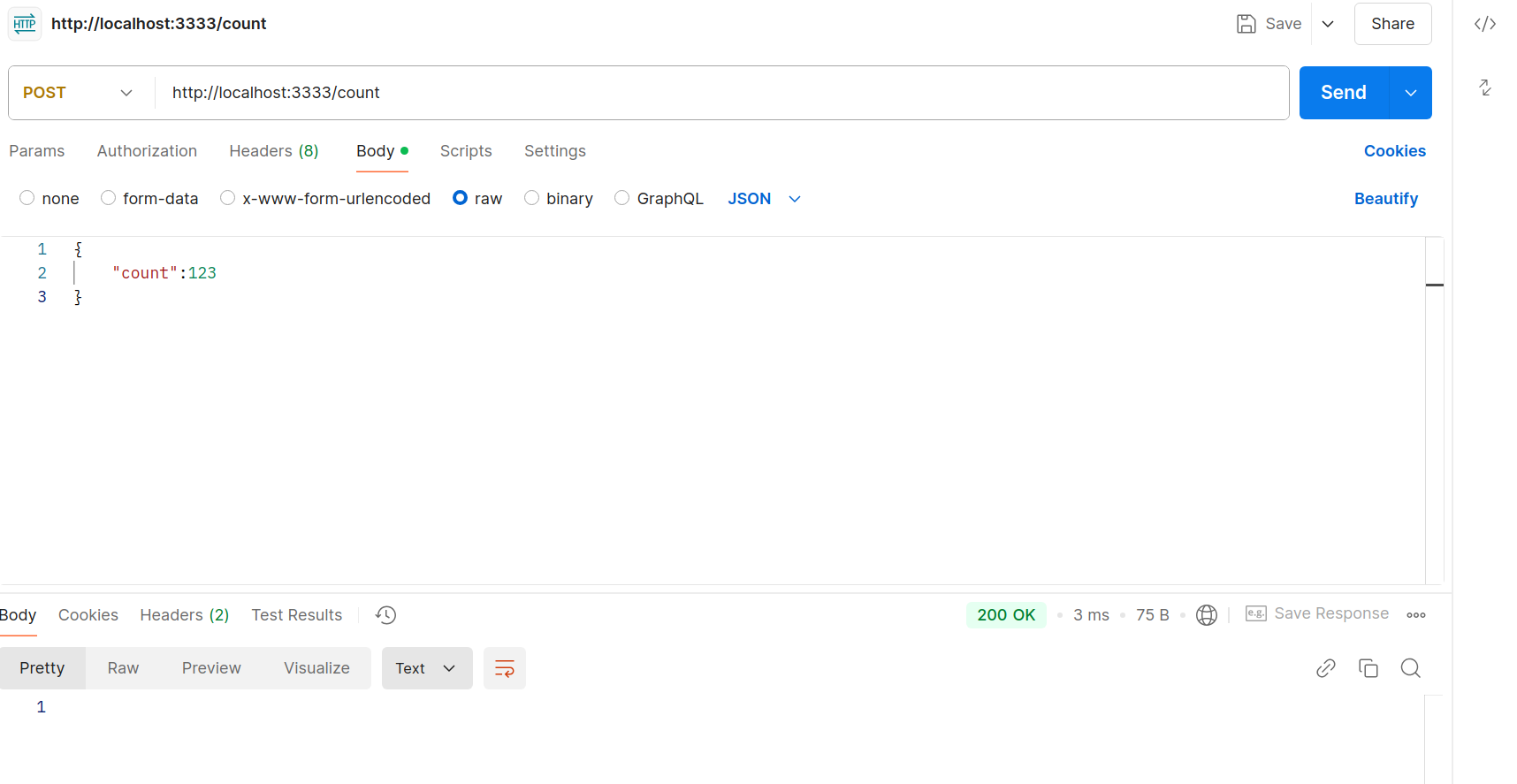
38 fmt.Println("Ошибка запуска сервера")

39 }

40 }

41

Рисунок 6: пример GET запроса

Рисунок 7: Пример POST запроса