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-Eng разработки на Golang

Дисциплина: Основы WEB-разработки.

Студент ИУ6-33Б _____ Е.В. Гредягина
(Группа) (Подпись, дата) (И.О. Фамилия)

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

Москва, 2024

1. **Цель работы**: изучение основ сетевого взаимодействия и серверной разработки с использованием языка Golang.

2. **Задание**:

- Задание 1 Напишите веб сервер, который по пути /get отдает текст "Hello, web!".

  Порт должен быть :8080.

- 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!

  порт :9000

- Задание 3 Напиши 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).

3. **Ход работы**:

1. Ознакомились с разделом "4. Списки, сеть и сервера" курса https://stepik.org/course/54403/info
2. Сделали форк данного репозитория в GitHub, склонировали получившуюся копию локально, создали от мастера ветку дев и переключитесь на нее
3. Выполнили задания.

   Коды сохранили в папке projects в подпапке с соответствующим названием задания. Скрины выполнения представлены ниже:

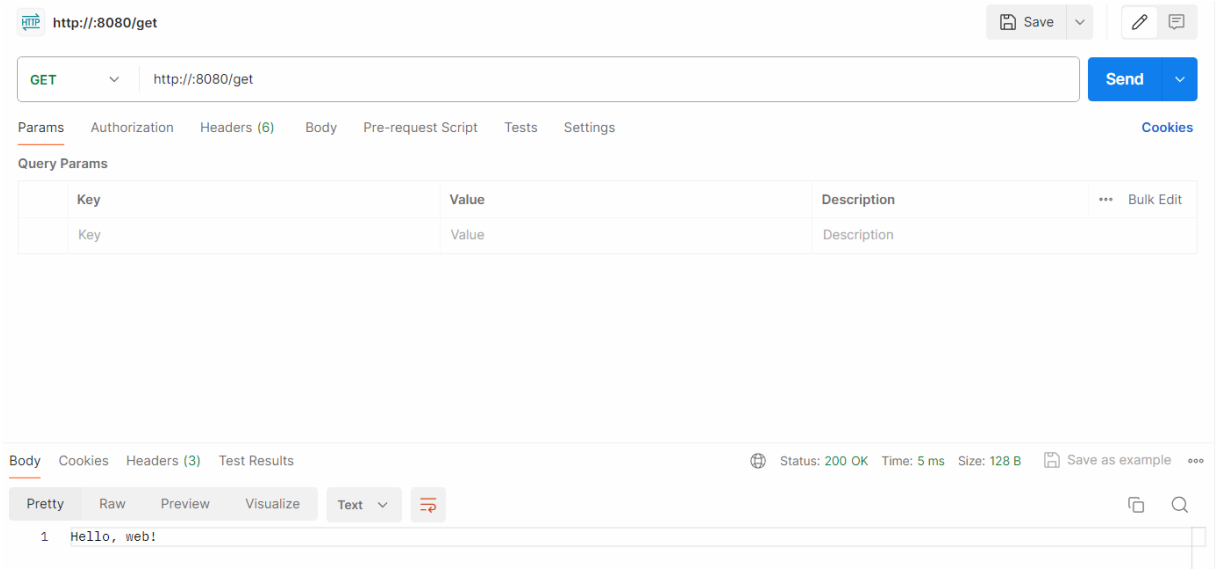

Рис. 1. Задача 1

Код:

```go
package main

import (
    "fmt"
    "net/http"
)

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 {
        fmt.Println(err)
    }
}
```

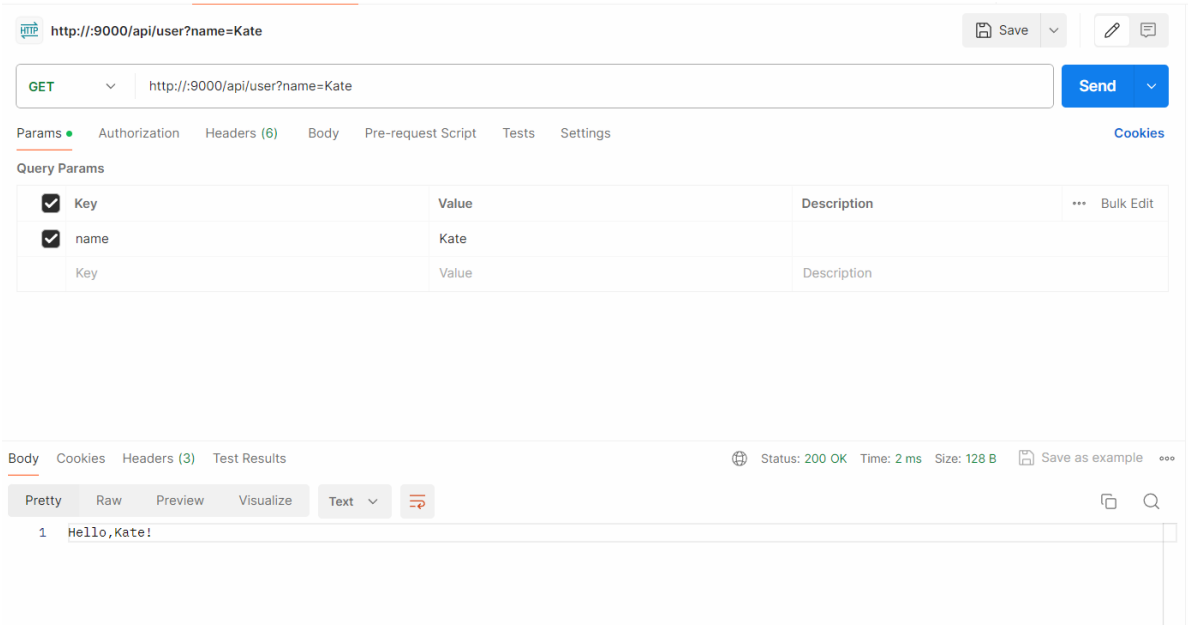

Рис. 2. Задача 2

Код:

```go
package main

import (
    "fmt"
    "net/http"
)

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

func main() {
    http.HandleFunc("/api/user", handler)
    err := http.ListenAndServe(":9000", nil)
    if err != nil {
        fmt.Println(err)
    }
}
```

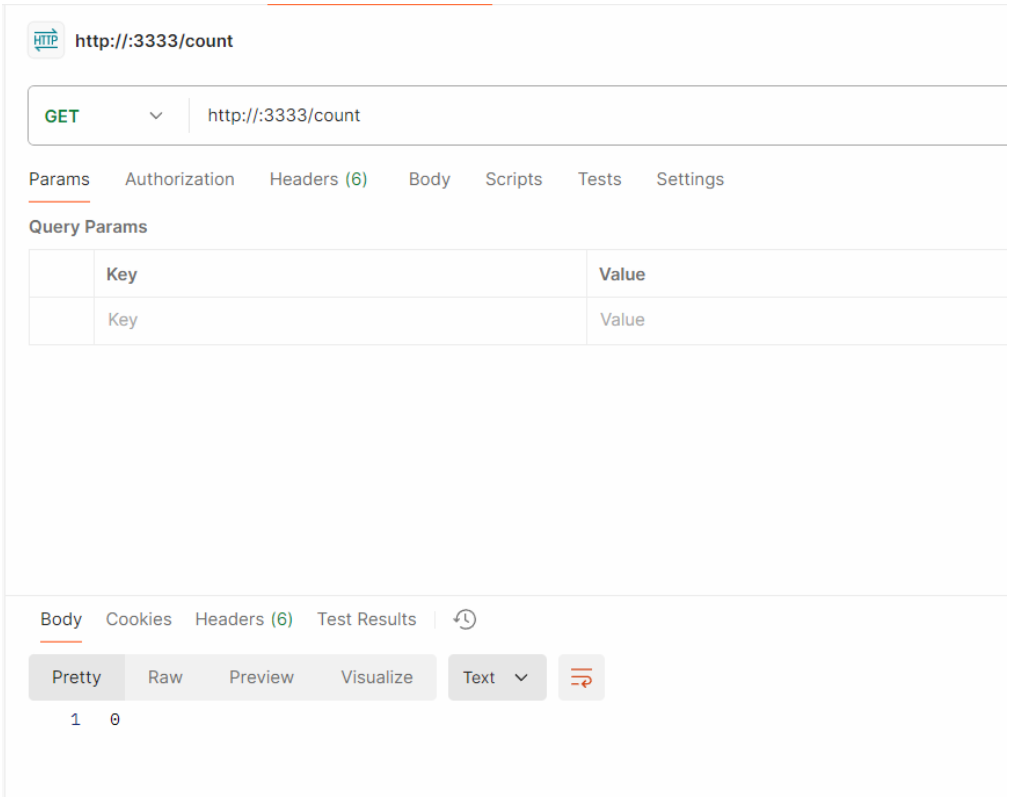

Рис. 3. Задача 3

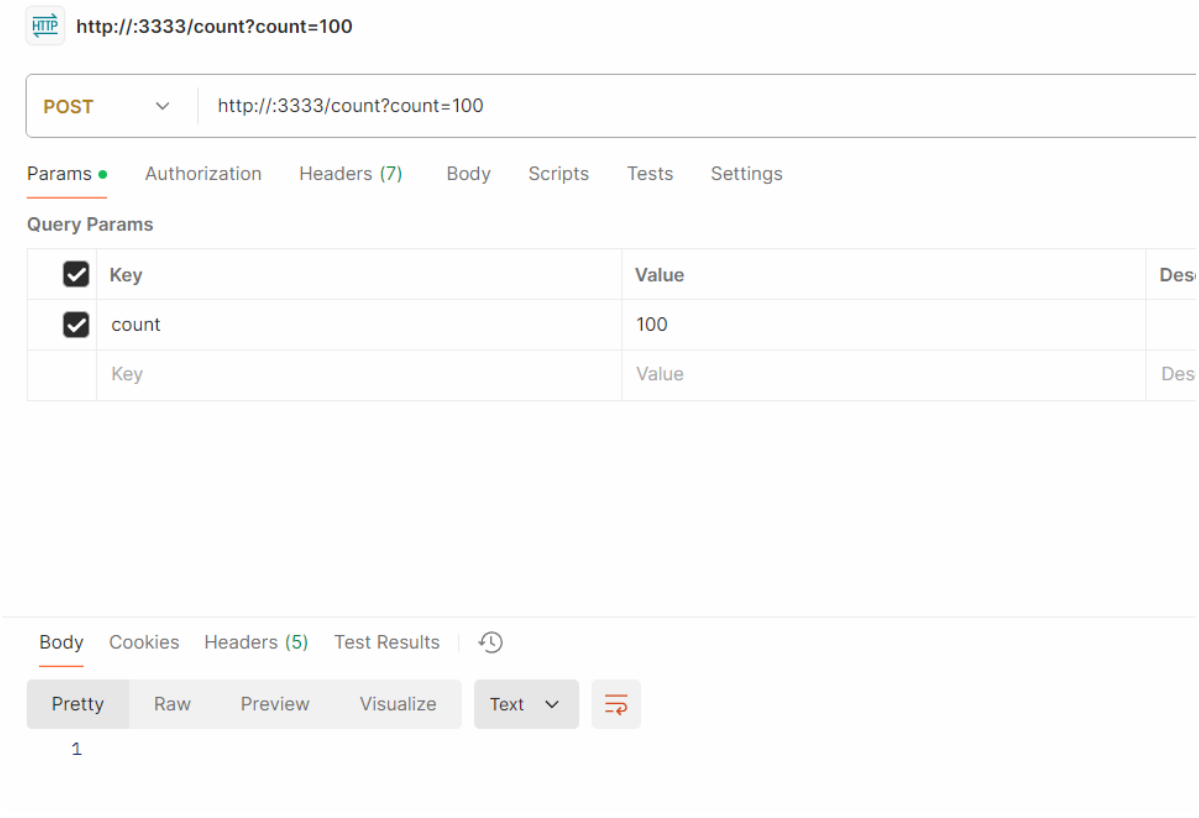

Рис. 4. Задача 3

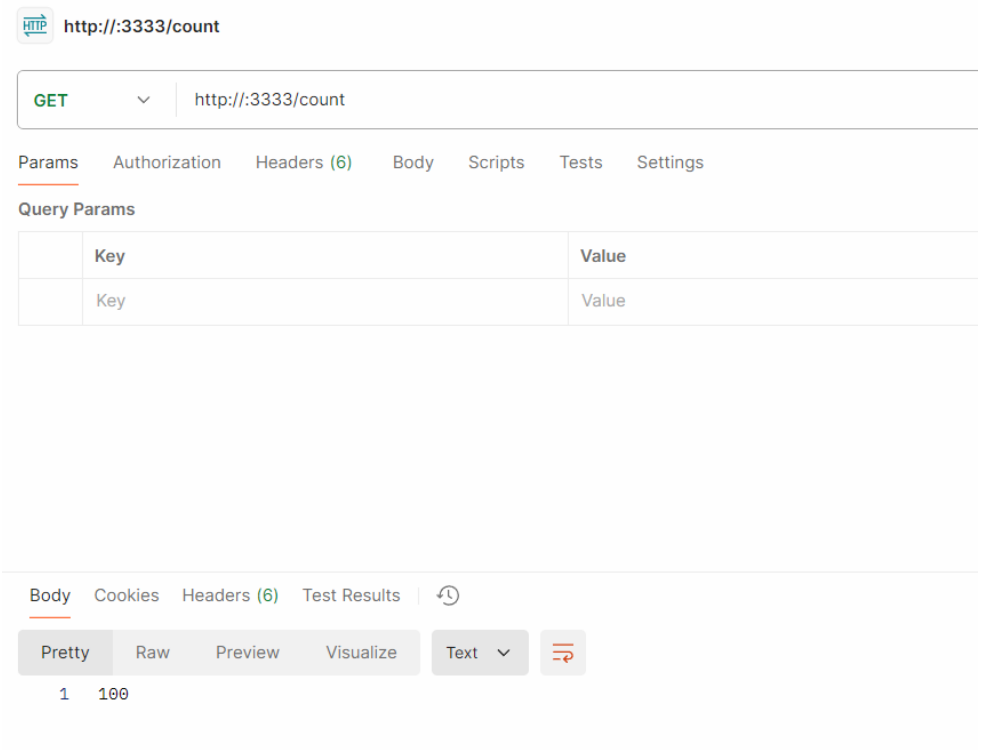

Рис. 5. Задача 3

Код:

```
package main

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

func main() {
    counter := 0
    http.HandleFunc("/count", func(w http.ResponseWriter, r *http.Request) {
        w.Header().Set("Access-Control-Allow-Origin", "*")
        w.Header().Set("Access-Control-Allow-Methods", "GET, OPTIONS")
        w.Header().Set("Access-Control-Allow-Headers", "Content-Type")
        if r.Method == http.MethodGet {
            w.Write([]byte(strconv.Itoa(counter)))
        }
        if r.Method == http.MethodPost {
            r.ParseForm()
            numberString := r.Form.Get("count")
            a, err := strconv.Atoi(numberString)
            if err != nil {
                w.WriteHeader(http.StatusBadRequest)
                fmt.Fprintln(w, "это не число")
                return
            }
            counter += a
        }
        if r.Method != http.MethodPost && r.Method != http.MethodGet {
            http.Error(w, "method is not allowed", http.StatusMethodNotAllowed)
        }
    })
    http.ListenAndServe(":3333", nil)
}
```

4. Сделали отчёт и поместите его в директорию docs
5. Зафиксировали изменения, сделали коммит и отправили полученное состояние ветки дев в удаленный репозиторий GitHub
6. Через интерфейс GitHub создали Pull Request dev --> master

4. **Заключение**: в ходе лабораторной работы я изучила основы асинхронного программирования с использованием языка Golang.

5. **Список используемых источников**:

https://stepik.org/course/54403/info (курс на Степике)