|  |  |
| --- | --- |
| 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)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.03.01 Информатика и вычислительная техника**

**Отчет**

**по рубежному контролю № 1**

**Название:** Разработка WEB-сервера на Golang

**Дисциплина:** Алгоритмизация и программирование

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ6-32Б |  | 22.10.2024 | Л.И. Заушников |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  | 22.10.2024 | В.Д. Шульман |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2024

**Цель работы:** необходимо реализовать простой сервер на Golang, принимающий клиентские запросы по протоколу HTTP.

**Ход работы**

## Билет №7. Среднее из трёх

Необходимо написать веб-севрер на GO, решающий задачу "Среднее из трёх". Сервер должен запускаться по адресу `127.0.0.1:8081`.

У севрера должна быть ручка (handler) `POST /middle`. Эта ручка ожидает, что через JSON будет передано 3 параметра типа int: `a`, `b` и `c`.

При обработке http-запроса должно возвращаться то число, которое больше одного, но меньше другого.

В качестве ответа сервер должен возвращать JSON с единственным полем `result`.

Примерм запроса (curl):

```

curl --header "Content-Type: application/json" --request POST --data '{"a":5,"b":11,"c":10}' http://127.0.0.1:8081/middle

```

Пример ответа:

```

{"result":10}

```

Автор: Шульман Виталий Дмитриевич

Был написан код для решения задачи билета. Код представлен ниже

package main

import (

"encoding/json"

"fmt"

"net/http"

)

type Request struct {

A \*int `json:"a"`

B \*int `json:"b"`

C \*int `json:"c"`

}

type Response struct {

Result int `json:"result"`

}

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

if r.Method != http.MethodPost {

http.Error(w, "Неподдерживаемый метод", http.StatusMethodNotAllowed)

return

}

var req Request

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

if err != nil {

http.Error(w, "Неверный формат запроса, проверьте его и повторите попытку", http.StatusBadRequest)

return

}

if req.A == nil {

http.Error(w, "Первое число потеряно", http.StatusBadRequest)

return

}

if req.B == nil {

http.Error(w, "Второе число потеряно", http.StatusBadRequest)

return

}

if req.C == nil {

http.Error(w, "Третье число потеряно", http.StatusBadRequest)

return

}

if \*req.A == \*req.B || \*req.B == \*req.C || \*req.C == \*req.A {

http.Error(w, "Необходимо предоставить разные числа", http.StatusBadRequest)

return

}

max := \*req.A

min := \*req.A

if \*req.B > max {

max = \*req.B

}

if \*req.C > max {

max = \*req.C

}

if \*req.B < min {

min = \*req.B

}

if \*req.C < min {

min = \*req.C

}

var middle int

if \*req.A != max && \*req.A != min {

middle = \*req.A

} else if \*req.B != max && \*req.B != min {

middle = \*req.B

} else {

middle = \*req.C

}

response := Response{Result: middle}

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

json.NewEncoder(w).Encode(response)

}

func main() {

http.HandleFunc("/middle", middleHandler)

fmt.Println("Сервер начал работать на http://127.0.0.1:8081...")

err := http.ListenAndServe("127.0.0.1:8081", nil)

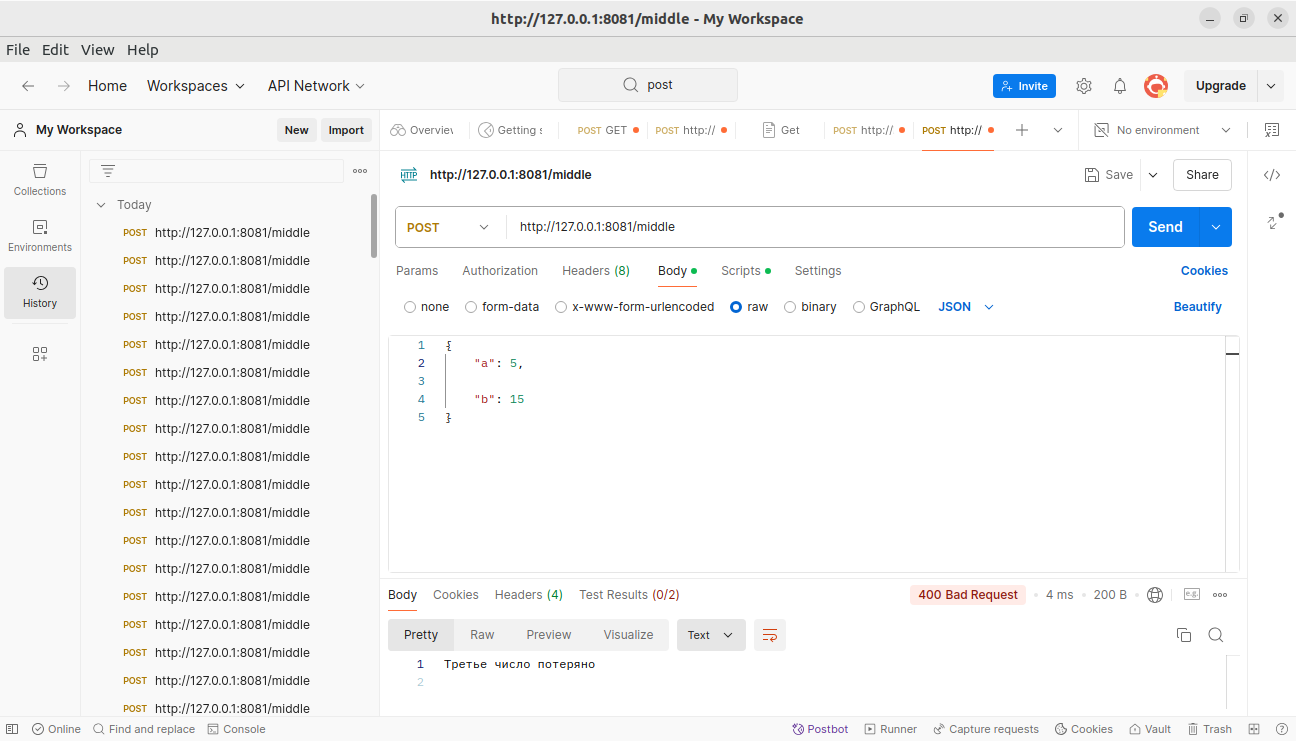
if err != nil {

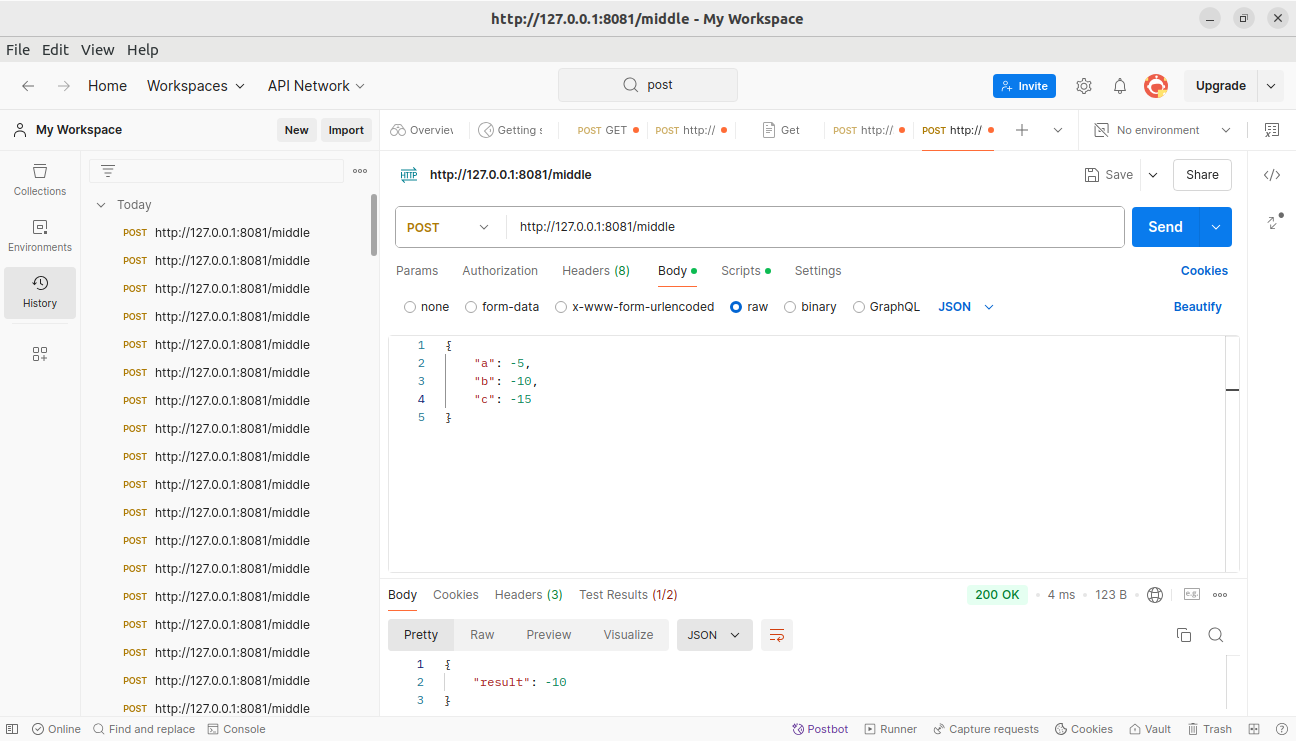
fmt.Println("Ошибка запуска сервера:", err)

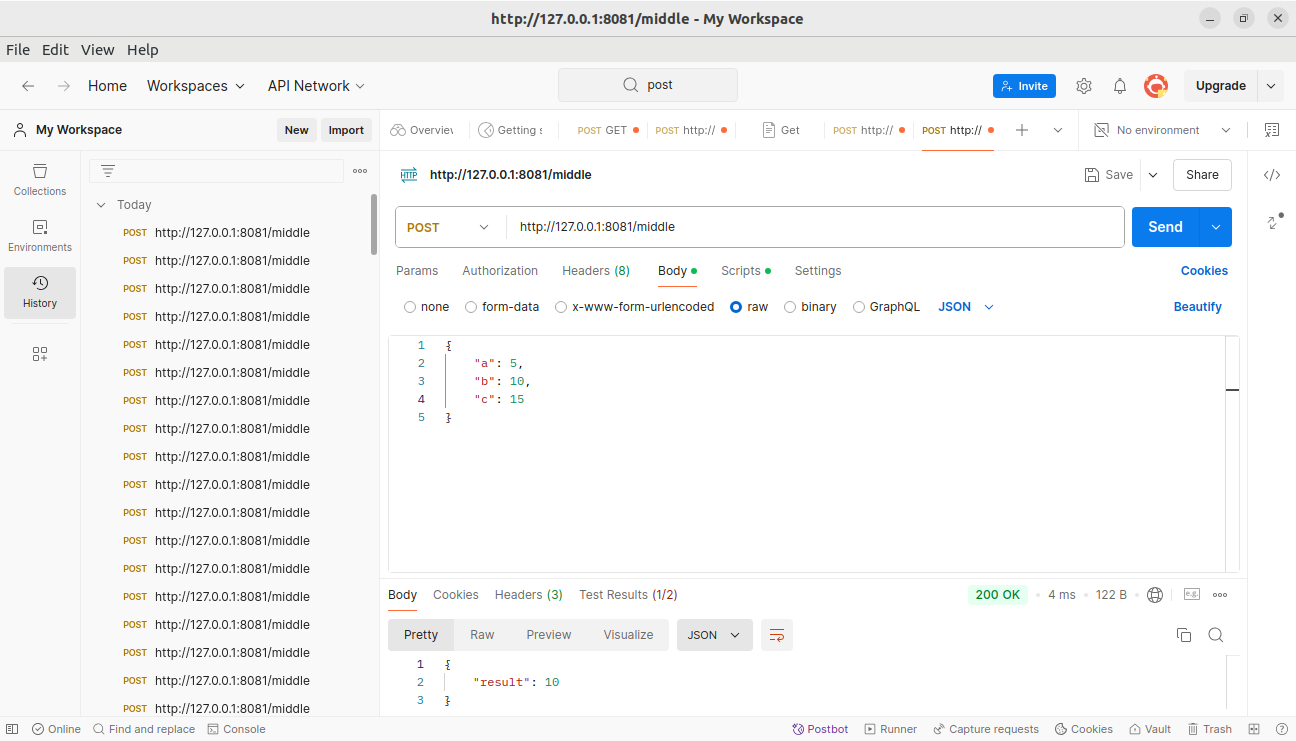
}

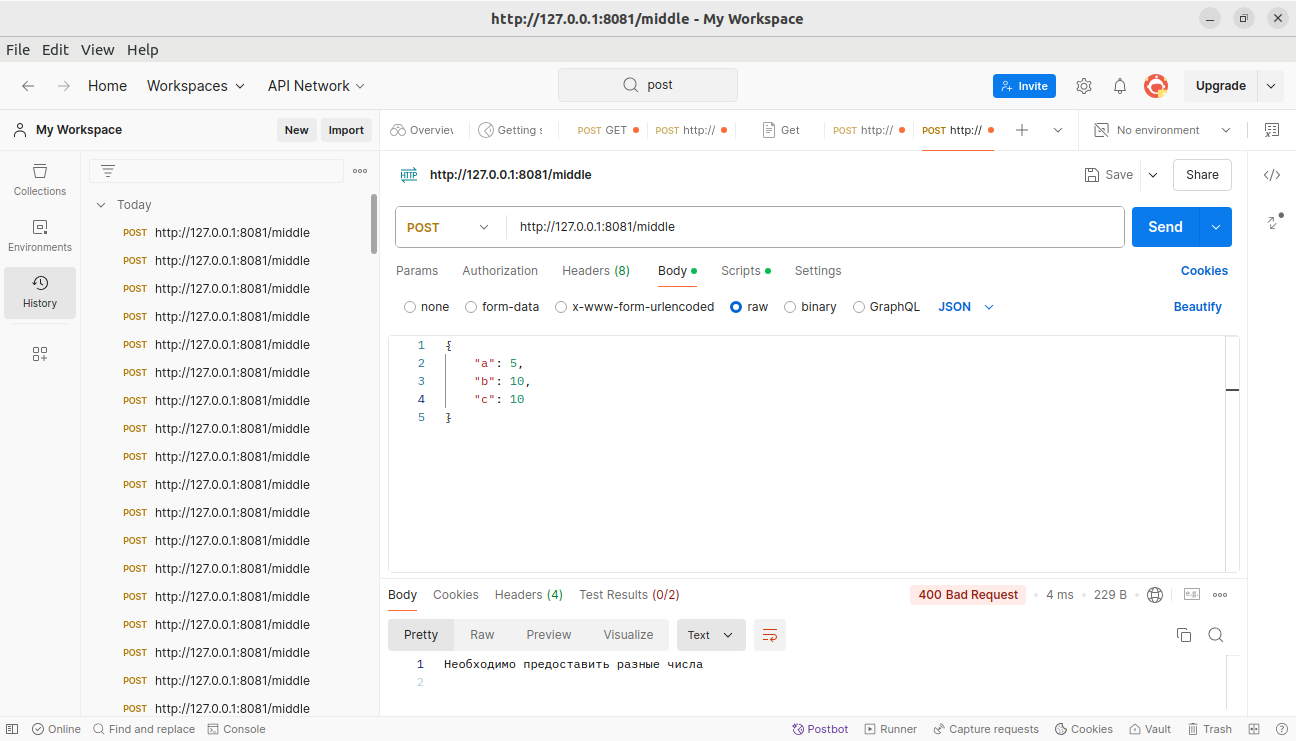
}

На рисунках 1-4 представлены результаты тестирования программы.

Рисунок 1 - Тестирование программы для решения задачи

Рисунок 2 - Тестирование программы для решения задачи

Рисунок 3 - Тестирование программы для решения задачи

Рисунок 4 - Тестирование программы для решения задачи

Заключение

В ходе решения задачи были отработаны навыки написания простейших веб-серверов.