

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное
учреждение высшего образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра автоматизированных систем управления (АСУ)

Лабораторная работа №3 дисциплине «Разработка веб-приложений для
научных и прикладных задач»

Отчет по лабораторной работе по дисциплине «Разработка веб-сервисов для
научных и прикладных задач» Вариант 8

Студент гр. 433-3

Дудник С.С.

15 июня 2024 г.

Руководитель

доцент каф. АСУ, канд.

техн. наук _____А.

Я. Суханов

16 июня 2024 г.

Оглавление

ВВЕДЕНИЕ.....	3
ХОД РАБОТЫ	4
ВЫВОД	9

ВВЕДЕНИЕ

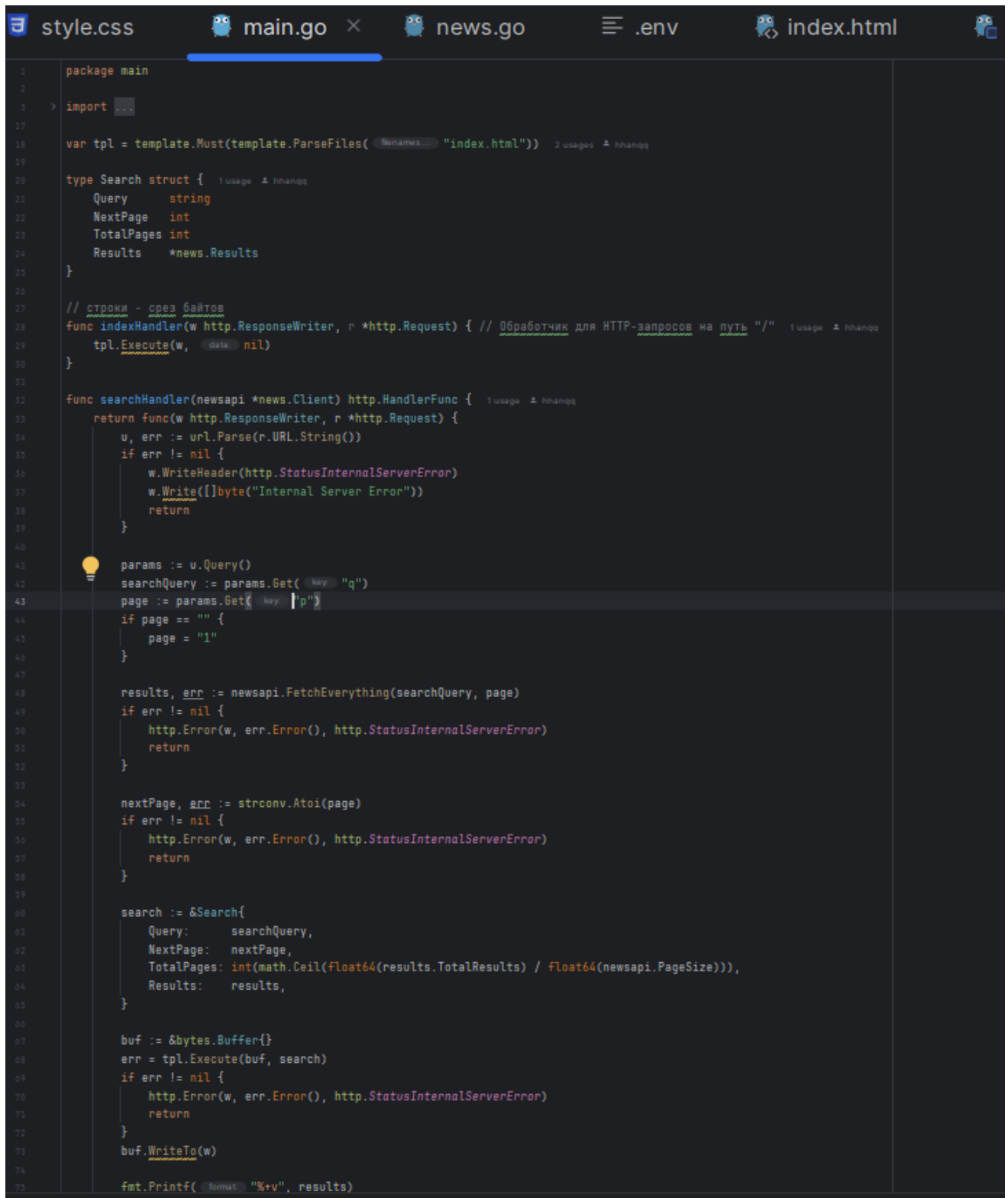
Цель работы является изучение возможностей языка Python, возможностей по обработке данных, создание веб-приложения с использованием фреймворка FastAPI, системы контроля версий GIT и систем непрерывной интеграции Travis или GitHub Action, а так же деплой на какую-либо PaaS платформу, например, Heroku или Render. Перед выполнением задания лабораторной работы необходимо ознакомиться с теоретическим материалом.

Задачи:

1. Сделать документированный веб-приложение, используя фреймворк fast-api;
2. Использовать систему контроля GIT;
2. Выполнить индивидуальное задание.

ХОД РАБОТЫ

Первая программа с использованием фреймворка net/http. Код программы с содержанием индивидуального задания файла main.go представлен на рисунке 2.1.



```
1 package main
2
3 > import
17
18 var tpl = template.Must(template.ParseFiles("index.html")) 1 usage 1 hhanqq
19
20 type Search struct { 1 usage 1 hhanqq
21     Query      string
22     NextPage    int
23     TotalPages  int
24     Results     *news.Results
25 }
26
27 // строки - срез байтов
28 func indexHandler(w http.ResponseWriter, r *http.Request) { // Обработчик для HTTP-запросов на путь "/" 1 usage 1 hhanqq
29     tpl.Execute(w, nil)
30 }
31
32 func searchHandler(newsapi *news.Client) http.HandlerFunc { 1 usage 1 hhanqq
33     return func(w http.ResponseWriter, r *http.Request) {
34         u, err := url.Parse(r.URL.String())
35         if err != nil {
36             w.WriteHeader(http.StatusInternalServerError)
37             w.Write([]byte("Internal Server Error"))
38             return
39         }
40
41         params := u.Query()
42         searchQuery := params.Get("q")
43         page := params.Get("p")
44         if page == "" {
45             page = "1"
46         }
47
48         results, err := newsapi.FetchEverything(searchQuery, page)
49         if err != nil {
50             http.Error(w, err.Error(), http.StatusInternalServerError)
51             return
52         }
53
54         nextPage, err := strconv.Atoi(page)
55         if err != nil {
56             http.Error(w, err.Error(), http.StatusInternalServerError)
57             return
58         }
59
60         search := &Search{
61             Query:      searchQuery,
62             NextPage:    nextPage,
63             TotalPages:  int(math.Ceil(float64(results.TotalResults) / float64(newsapi.PageSize))),
64             Results:     results,
65         }
66
67         buf := &bytes.Buffer{}
68         err = tpl.Execute(buf, search)
69         if err != nil {
70             http.Error(w, err.Error(), http.StatusInternalServerError)
71             return
72         }
73         buf.WriteTo(w)
74
75         fmt.Printf("Format: \"%v\", results)
```

Рисунок 2.1- Содержание файла main.go

Содержание файла index.html, style.css и news.go представлено на рисунках 2.2, 2.3 и 2.4.



```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <meta http-equiv="X-UA-Compatible" content="ie=edge">
7   <title>news</title>
8   <link rel="stylesheet" href="/assets/style.css">
9   <style>
10    .rotate-180 {
11      transform: rotate(180deg);
12      transition: transform 0.5s ease-in-out;
13    }
14  </style>
15 </head>
16 <body>
17 <main>
18 <header>
19   <a class="logo" href="/">News Demo</a>
20   <form action="/search" method="GET">
21     <input
22       autofocus
23       class="search-input"
24       value="{{ .Query }}"
25       placeholder="Поиск"
26       type="search"
27       name="q"
28     />
29   </form>
30   <a
31     href="https://github.com/hhanqq/newsapi.go"
32     class="button github-button"
33   >Посмотреть проект на GitHub</a>
34 </header>
35 <section class="container">
36   <ul class="search-results">
37     {{ range .Results.Articles }}
38     <li class="news-article">
39       <div>
40         <a target="_blank" rel="noopener" href="{{ .URL }}">
41           <h3 class="title">{{ .Title }}</h3>
42         </a>
43         <p class="description">{{ .Description }}</p>
44         <div class="metadata">
45           <p class="source">{{ .Source.Name }}</p>
46           <time class="published-date">{{ .PublishedAt }}</time>
47         </div>
48       </div>
49       
50       <button onclick="rotateImage(this)">Rotate 180°</button>
51     </li>
52     {{ end }}
53   </ul>
54 </section>
55 </main>
56 <script>
57   function rotateImage(button) {
58     const img = button.previousElementSibling;
59     img.classList.toggle('rotate-180');
60   }
61 </script>
62 </body>
```

Рисунок 2.2 – Содержание файла index.html

```
style.css x main.go news.go .env index.html go.mod
18 :root {
19   --black: #000;
20 }
21 body {
22   font-family: -apple-system, BlinkMacSystemFont, 'Segoe UI', Roboto, Oxygen-Sans, Ubuntu, Cantarell, 'Helvetica Neue', sans-serif;
23 }
24
25 a {
26   text-decoration: none;
27   color: #333;
28 }
29
30 a:hover {
31   text-decoration: underline;
32 }
33
34 a.button {
35   border: 2px solid #004400;
36   color: var(--dark-green);
37   border-radius: 4px;
38   padding: 6px 24px;
39   font-size: 14px;
40   font-weight: 400;
41 }
42
43 a.button:hover {
44   text-decoration: none;
45   background-color: var(--dark-green);
46   color: var(--white );
47 }
48
49 header {
50   width: 100%;
51   height: 50px;
52   position: fixed;
53   top: 0;
54   left: 0;
55   right: 0;
56   display: flex;
57   justify-content: space-between;
58   background-color: var(--sky-blue);
59   padding: 5px 10px;
60   align-items: center;
61 }
62
63 .logo {
64   color: #000000;
65 }
66
67 form {
68   height: calc(100% - 10px);
69 }
70
71 .search-input {
72   width: 500px;
73   height: 100%;
74   border-radius: 4px;
75   border-color: transparent;
76   background-color: var(--white);
77   color: var(--black);
78   font-size: 16px;
```

Рисунок 2.3 – Содержание файла style.css

```

1 package main
2
3 import (
4     "encoding/json"
5     "fmt"
6     "io/ioutil"
7     "net/http"
8     "net/url"
9     "time"
10 )
11
12 type Client struct {
13     http *http.Client
14     key   string
15     PageSize int
16 }
17
18 func (c *Client) FetchEverything(query, page string) (*Results, error) {
19     endpoint := fmt.Sprintf("https://newsapi.org/v2/everything?q=%s&pageSize=%d&page=%d&apiKey=%s&sortBy=publishedAt&language=en", url.QueryEscape(query), c.PageSize, page, c.key)
20     resp, err := c.http.Get(endpoint)
21     if err != nil {
22         return nil, err
23     }
24
25     defer resp.Body.Close()
26
27     body, err := ioutil.ReadAll(resp.Body)
28     if err != nil {
29         return nil, err
30     }
31
32     if resp.StatusCode != http.StatusOK {
33         return nil, fmt.Errorf(string(body))
34     }
35
36     res := &Results{}
37     return res, json.Unmarshal(body, res)
38 }
39
40 type Article struct {
41     Source struct {
42         ID interface{} `json:"id"`
43         Name string       `json:"name"`
44     } `json:"source"`
45     Author string `json:"author"`
46     Title string `json:"title"`
47     URL string `json:"url"`
48     Description string `json:"description"`
49     URLToImage string `json:"urlToImage"`
50     PublishedAt time.Time `json:"publishedAt"`
51     Content string `json:"content"`
52 }
53
54 type Results struct {
55     Status string `json:"status"`
56     TotalResults int `json:"totalResults"`
57     Articles []Article `json:"articles"`
58 }
59
60 func NewClient(httpClient *http.Client, key string, pageSize int) *Client {
61     if pageSize > 100 {
62         pageSize = 100
63     }
64 }
65
66 Client.FetchEverything(query string, page string) (*Results, error)

```

Рисунок 2.4 – Содержание файла news.go

Результат выполнения задания по созданию веб-приложения представлен на рисунке 2.5.

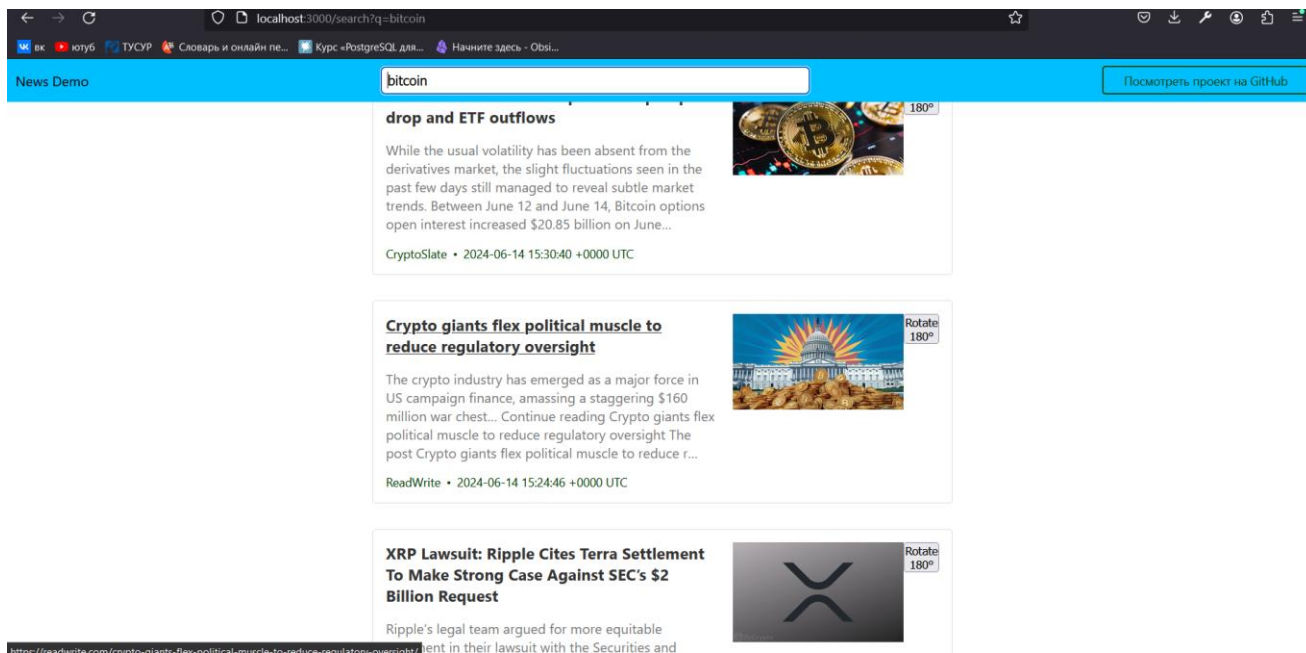


Рисунок 2.5 – Результат работы кода программы

Результат выполнения индивидуального задания виден на рисунке 2.6.

Crypto giants flex political muscle to reduce regulatory oversight

The crypto industry has emerged as a major force in US campaign finance, amassing a staggering \$160 million war chest... Continue reading Crypto giants flex political muscle to reduce regulatory oversight The post Crypto giants flex political muscle to reduce r...

ReadWrite • 2024-06-14 15:24:46 +0000 UTC




Рисунок 2.6 – Повёрнутое изображение

ВЫВОД

В ходе лабораторной работы мы изучили возможности языка GoLang, возможности фреймворка net/http. Освоили инструменты для создания веб-сервиса. Получили практические навыки по работе с фреймворками и веб-приложениями.