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 «Теоретическая информатика и компьютерные технологии»

Лабораторная работа № 0

по курсу «Компьютерные сети»

«Разработка Web-ориентированного клиент-серверного приложения получения и представления данных из RSS-канала»

Вариант 8

Студент группы ИУ9-32Б Павлов И. П.

Преподаватель Посевин Д. П.

Москва 2023

1 Цель работы

Рассматривается задача разработки web-сервера на языке GO на основе пакета net/http и разработки приложения на языке GO, реализующего синтаксический разбор XML файла формата RSS.

2 Условие

Необходимо разработать web-сервер, который выполняет соединение с удаленным (удаленными) серверами RSS-новостей и возвращает результаты обработки данных в структурированном виде (страница гипертекста) web-клиенту, в нашем случае в браузер по вариантам.

3 Реализация main.go

```
package main

import (
        "fmt"
        "github.com/mmcdole/gofeed"
        "html/template"
        "log"
        "net/http"
        "time"
)

const feedURL = "https://www.aviaport.ru/digest/press-releases/rss/"

type Article struct {
        URL         string
        Title       string
        Description template.HTML
        PublishedAt time.Time
}

func NewsRouterHandler(w http.ResponseWriter, r *http.Request) {
        if r.URL.Path != "/news/" {
                http.Error(w, "404 Not Found", http.StatusNotFound)
                return
        }

        parser := gofeed.NewParser()
        feed, err := parser.ParseURL(feedURL)
        if err != nil {
                panic(err)
        }

        articles := make([]Article, 0)
```

```
	for _, v := range feed.Items {
		articles = append(articles, Article{
			URL:         v.Link,
			Title:       v.Title,
			Description: template.HTML(v.Description),
			PublishedAt: *v.PublishedParsed,
		})
	}

	tl, err := template.ParseFiles("static/news.html")
	if err != nil {
		panic(err)
	}
	err = tl.Execute(w, articles)
	if err != nil {
		panic(err)
	}
}

func ContactRouterHandler(w http.ResponseWriter, r *http.Request) {
	if r.URL.Path != "/contacts/" {
		http.Error(w, "404 Not Found", http.StatusNotFound)
		return
	}
	http.ServeFile(w, r, "./static/contacts.html")
}

func InfoRouterHandler(w http.ResponseWriter, r *http.Request) {
	if r.URL.Path != "/info/" {
		http.Error(w, "404 Not Found", http.StatusNotFound)
		return
	}
	http.ServeFile(w, r, "./static/info.html")
}

func HomeRouterHandler(w http.ResponseWriter, r *http.Request) {
	if r.URL.Path != "/" {
		http.Error(w, "404 Not Found", http.StatusNotFound)
		return
	}

	switch r.Method {
	case "GET":
		http.ServeFile(w, r, "./static/index.html")
	case "POST":
		if err := r.ParseForm(); err != nil {
			http.Error(w, "400 Bad request", http.StatusBadRequest)
			return
		}
		fmt.Println(r.Form)
		fmt.Println("path", r.URL.Path)
		for k, v := range r.Form {
			fmt.Println("key:", k)
			fmt.Println("val:", v)
		}
		fmt.Fprintf(w, "Data sent!")
	}
}
```

```go
func main() {
        http.HandleFunc("/", HomeRouterHandler)
        http.HandleFunc("/info/", InfoRouterHandler)
        http.HandleFunc("/contacts/", ContactRouterHandler)
        http.HandleFunc("/news/", NewsRouterHandler)
        fs := http.FileServer(http.Dir("assets"))
        http.Handle("/assets/", http.StripPrefix("/assets/", fs))
        if err := http.ListenAndServe("localhost:9000", nil); err != nil {
                log.Fatalf("ListenAndServe: %v", err)
        }
}
```

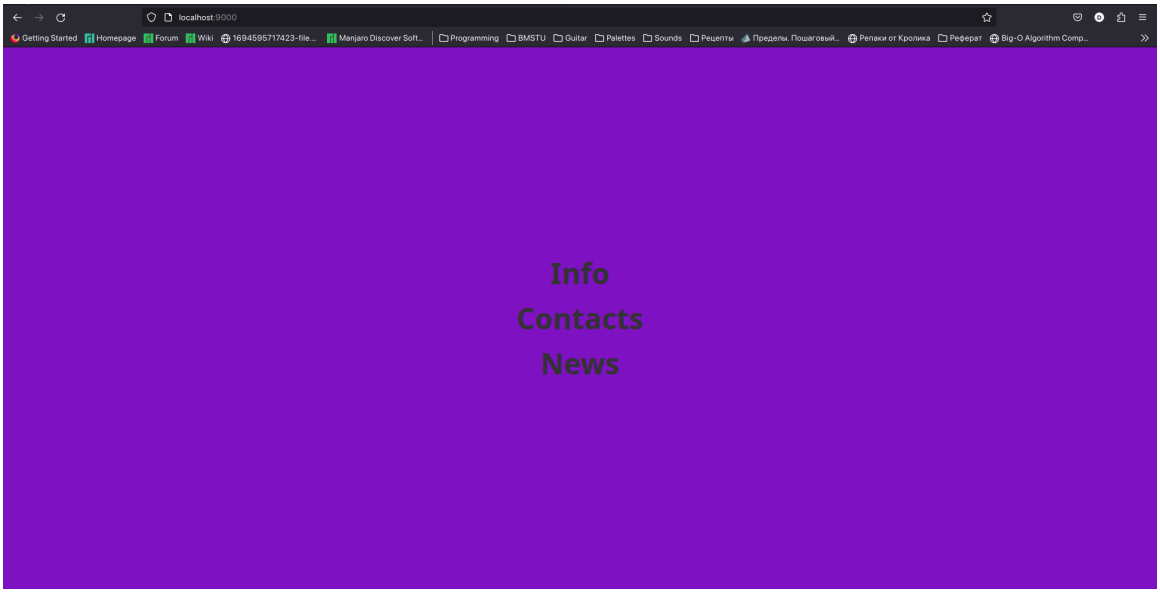

Рис. 1: Главная страница

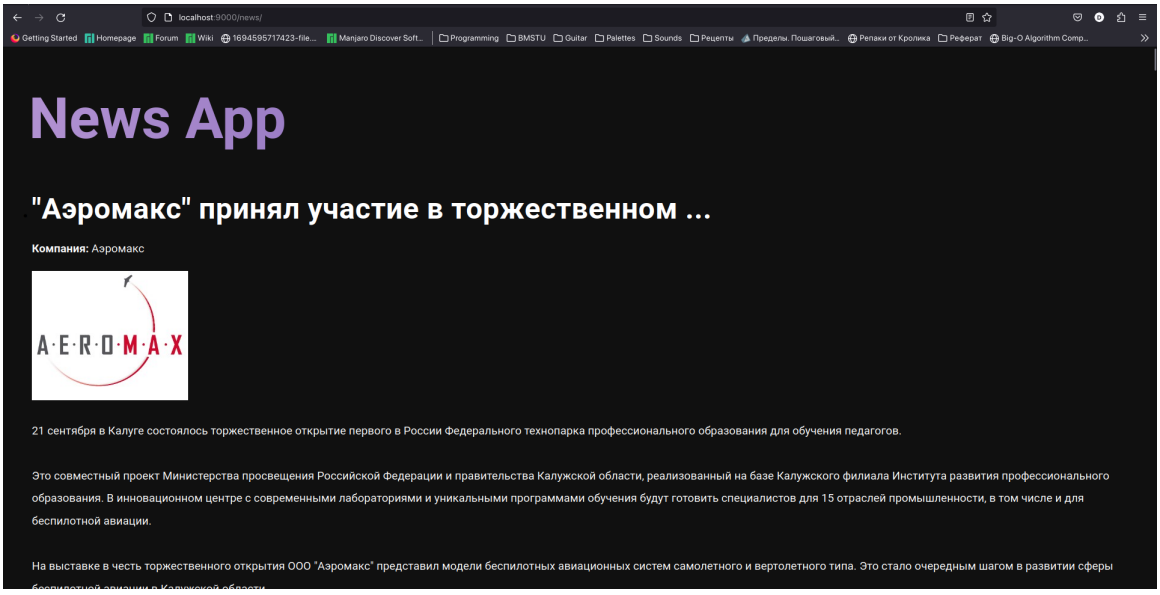

Рис. 2: Новостная страница

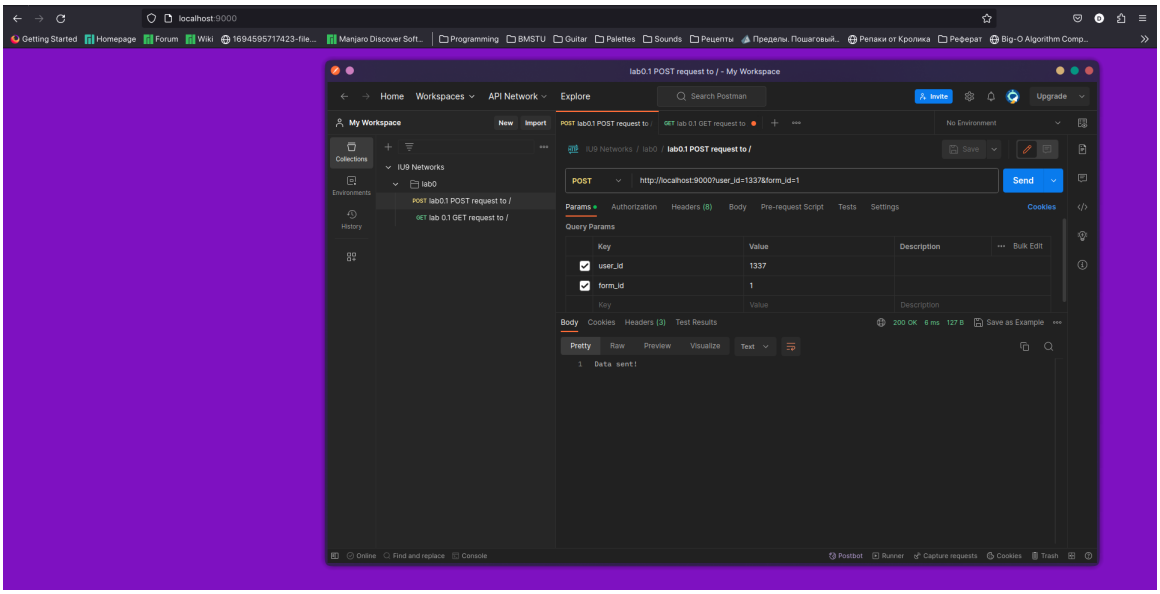

Рис. 3: POST-запрос на сервер, получен ответ "Data sent!"