Московский государственный технический университет им. Н.Э. Баумана

Факультет «Информатика и системы управления»
Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Парадигмы и конструкции языков программирования»

Отчет по домашнему заданию «Разработка вебсайта на языке программирования Go»

Выполнил:
студент группы ИУ5-35Б
Ходырев Роман
Подпись и дата:

Проверил:
преподаватель каф. ИУ5
Гапанюк Юрий Евгеньевич
Подпись и дата:

Постановка задачи

1. Написать сайт для хранения обзоров фильмов, используя язык программирования Go
2. Использовать для хранения обзоров СУБД MySQL
3. Реализовать технологию динамических страниц
4. Добавить возможность создания нового обзора

Текст программы

main.go (бэкенд сайта, взаимодейтсвие с БД MySQL)

```go
package main

import (
    "database/sql"
    "fmt"
    "html/template"
    "io"
    "net/http"
    "os"
    "strings"

    _ "github.com/go-sql-driver/mysql"
    "github.com/gorilla/mux"
)

type Film struct {
    Id    uint16 `json:"id"`
    Name  string `json:"name"`
    Plot  string `json:"plot"`
    Photo string `json:"photo"`
}

func home_page(w http.ResponseWriter, r *http.Request) {
    tmp, err := template.ParseFiles("templates/home_page.html")
    if err != nil {
        panic(err)
    }

    db, err := sql.Open("mysql", "root:root@tcp(127.0.0.1:3306)/film_reviews")
    if err != nil {
        panic(err)
    }
    defer db.Close()

    dbfilms, err := db.Query("SELECT id, name, plot, photo FROM films")
    if err != nil {
        panic(err)
    }
```

```
    Films := []Film{}
    for dbfilms.Next() {
        var film Film
        err = dbfilms.Scan(&film.Id, &film.Name, &film.Plot, &film.Photo)
        if err != nil {
            panic(err)
        }
        Films = append(Films, film)
    }

    tmp.ExecuteTemplate(w, "home_page", Films)
    http.ListenAndServe("localhost:8000", nil)
}

func autor_page(w http.ResponseWriter, r *http.Request) {
    fmt.Println("The autor of this website")
}

func new_review_page(w http.ResponseWriter, r *http.Request) {
    tmp, err := template.ParseFiles("templates/new_review.html")
    if err != nil {
        panic(err)
    }

    tmp.ExecuteTemplate(w, "new_review", nil)
}

func save_new_review(w http.ResponseWriter, r *http.Request) {
    r.ParseMultipartForm(10 << 20)
    name := r.FormValue("filmName")
    plot := r.FormValue("plot")
    file, _, err := r.FormFile("posterLink")

    cname := strings.ReplaceAll(name, " ", "_") //Убираем из названия пробелы
    //Далее обработка полученного в поле постер файла
    if err != nil {
        panic(err)
    }
    defer file.Close()

    //Создаем новый файл в папке для сохранения
    newFile, err := os.Create("./images/film/" + cname + ".jpg")
    if err != nil {
        panic(err)
    }

    //Копируем содержимое первого файла в новый файл
    io.Copy(newFile, file)

    //Далее добавляем в БД запись
    db, err := sql.Open("mysql", "root:root@tcp(127.0.0.1:3306)/film_reviews")
    if err != nil {
        panic(err)
    }
```

```
    defer db.Close()

    insert, err := db.Query(fmt.Sprintf("INSERT INTO films (name, plot, photo) VALUES
('%s', '%s', '%s')", name, plot, "images/film/"+cname+".jpg"))
    if err != nil {
        panic(err)
    }
    defer insert.Close()

    http.Redirect(w, r, "/", http.StatusSeeOther)
}

func show_review(w http.ResponseWriter, r *http.Request) {
    tmp, err := template.ParseFiles("templates/review.html")
    if err != nil {
        panic(err)
    }

    vars := mux.Vars(r) // id фильма который мы хотим увидеть

    db, err := sql.Open("mysql", "root:root@tcp(127.0.0.1:3306)/film_reviews")
    if err != nil {
        panic(err)
    }
    defer db.Close()

    res, err := db.Query(fmt.Sprintf("SELECT * FROM films WHERE id = '%s'", vars["id"]))
    if err != nil {
        panic(err)
    }

    var showFilm Film //Фильм который будем показывать
    for res.Next() {
        var film Film
        err := res.Scan(&film.Id, &film.Name, &film.Plot, &film.Photo)
        if err != nil {
            panic(err)
        }

        showFilm = film //Нашли нужный фильм
    }

    tmp.ExecuteTemplate(w, "review", showFilm)
}

func handleFunc() {
    http.Handle("/images/film/", http.StripPrefix("/images/film/",
http.FileServer(http.Dir("./images/film"))))
    http.Handle("/static/", http.StripPrefix("/static/",
http.FileServer(http.Dir("./static/"))))

    rtr := mux.NewRouter()
    rtr.HandleFunc("/", home_page).Methods("GET")
    rtr.HandleFunc("/new_review/", new_review_page).Methods("GET")
```

```
    rtr.HandleFunc("/save_new_review/", save_new_review).Methods("POST")
    rtr.HandleFunc("/autor_page/", autor_page).Methods("GET")
    rtr.HandleFunc("/review/{id:[0-9]+}", show_review).Methods("GET")

    http.Handle("/", rtr)
    http.ListenAndServe("localhost:8000", nil)
}

func main() {
    handleFunc()
}
```

home_page.html (html-шаблон главной страницы)

```
{{ define "home_page"}}

<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <title>Обозреватель - Обзоры фильмов</title>
    <link rel="stylesheet" href="static/style.css">
</head>

<body>
    <header>
        <h1>Обзоры фильмов</h1>
        <a href="/new_review/">Добавить обзор</a>
    </header>

    <div class="movies">
        {{range $index, $film := .}}
            <div class="movie">
                <a href="/review/{{.Id}}">
                    <img src="{{.Photo}}" alt="{{.Name}}">
                </a>
                <h3>{{.Name}}</h3>
            </div>
        {{end}}
    </div>

    <footer>
        <p>&copy; 2023 Обзоры фильмов</p>
    </footer>
    </body>

</html>
{{ end }}
```

new_review.html (страница для добавления нового обзора)

```
{{ define "new_review" }}

<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <title>Новый обзор</title>
    <link rel="stylesheet" href="../static/style.css">
</head>
<body>
    <header>
        <h1>Обзоры фильмов</h1>
    </header>
    <main>
        <h1>Новый обзор</h1>
        <form action="/save_new_review/" method="post", enctype="multipart/form-data">
            <label for="filmName">Название фильма:</label><br>
            <input type="text" id="filmName" name="filmName" required><br><br>

            <label for="plot">Краткое описание сюжета:</label><br>
            <textarea id="plot" name="plot" rows="4" cols="50"
required></textarea><br><br>

            <label for="posterLink">Ссылка на постер фильма:</label><br>
            <input type="file" id="posterLink" name="posterLink" required><br><br>

            <input type="submit" value="Добавить обзор">
        </form>
    </main>

    <footer>
        <p>&copy; 2023 Обзоры фильмов</p>
    </footer>
</body>

</html>

{{ end }}
```

review.html (динамическая страница с обзором фильма)

```
{{ define "review"}}

<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <title>{{.Name}}</title>
    <link rel="stylesheet" href="../static/style.css">
    <style>
        body {
```

```
            font-family: Arial, sans-serif;
            margin: 0;
            padding: 0;
            background-color: #f4f4f4;
        }
    </style>
</head>

<body>
    <header>
        <h1>Обзоры фильмов</h1>
        <a href="/">На главную</a>
    </header>

    <div class="film-container">
        <div class="film-details">
            <h1>{{.Name}}</h1>
            <p>{{.Plot}}</p>
        </div>
        <div class="film-poster">
            <img src="../{{.Photo}}" alt="{{.Name}}">
        </div>
    </div>

    <footer>
        <p>&copy; 2023 Обзоры фильмов</p>
    </footer>
</body>

</html>

{{ end }}
```

style.css (CSS-стили)

```
body {
    font-family: Arial, sans-serif;
    margin: 0;
    padding: 0;
    background-color: #f4f4f4;
}

header {
    background-color: #333;
    color: #fff;
    padding: 20px 0;
    text-align: center;
}

h1 {
    margin: 0;
    margin-bottom: 20px;
}
```

```css
.movies {
    display: flex;
    flex-wrap: wrap;
    justify-content: space-around;
    padding: 20px;
    gap: 20px;
}

.movie {
    text-align: center;
    width: calc(25% - 40px); /* 4 элемента в ряд, вычтем отступы */
    margin-bottom: 20px;
    border-radius: 5px;
    box-shadow: 0 0 5px rgba(0, 0, 0, 0.1); /* Добавим небольшую тень */
}

.movie h3 {
    font-size: 1.2em; /* Увеличим размер шрифта для названий */
    margin-top: 0; /* Уберем верхний отступ */
    text-decoration: none; /* Убираем стандартное оформление ссылки */
    display: block; /* Делаем названия блочными элементами */
    padding: 10px; /* Добавим отступ вокруг текста */
    background-color: lightgrey; /* Установим цвет фона */
    border-radius: 5px; /* Закруглим углы */
    cursor: default; /* Отключим курсор для текста */
}

.movie img {
    width: 100%;
    height: 90%; /* Задай высоту изображений, чтобы они были одного размера */
    object-fit: cover; /* Обрежь изображения до заданных размеров */
    border-radius: 5px 5px 0 0; /* Округлим углы изображения */
}


.movie a:hover {
    opacity: 0.7; /* При наведении на ссылку уменьшим немного прозрачность */
}


footer {
    background-color: #333;
    color: #fff;
    text-align: center;
    padding: 10px 0;
}

main {
    max-width: 600px;
    margin: 20px auto;
    background-color: #f9f9f9;
    padding: 20px;
    border-radius: 10px;
```

```
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
}

form {
    display: flex;
    flex-direction: column;
}

label {
    font-weight: bold;
    margin-bottom: 5px;
    color: #333;
}

input[type="text"],
textarea {
    width: 100%;
    padding: 10px;
    margin-bottom: 15px;
    border-radius: 5px;
    border: 1px solid #ccc;
}

input[type="submit"] {
    padding: 10px;
    background-color: #333;
    color: #fff;
    border: none;
    border-radius: 5px;
    cursor: pointer;
    transition: background-color 0.3s ease;
}

input[type="submit"]:hover {
    background-color: #555;
}

header a {
    position: absolute;
    top: 20px;
    right: 20px;
    display: inline-block;
    padding: 8px 16px;
    background-color: #333;
    color: #fff;
    text-decoration: none;
    border-radius: 4px;
    transition: background-color 0.3s ease;
}

header a:hover {
    background-color: #555;
}
```

```css
.film-container {
    display: flex;
    justify-content: space-between;
    padding: 20px;
}

.film-details {
    flex: 1;
    padding-right: 20px;
    max-width: 70%;
    text-align: justify; /* Выравниваем текст по ширине */
}

.film-details h1 {
    font-size: 36px;
    text-transform: uppercase;
    margin-bottom: 10px;
    text-align: center; /* Центрируем название */
}

.film-details p {
    font-size: 20px;
    line-height: 1.6;
}

.film-poster {
    max-width: 30%; /* Сокращаем постер до 30% */
}

.film-poster img {
    max-width: 100%;
    height: auto;
}
```

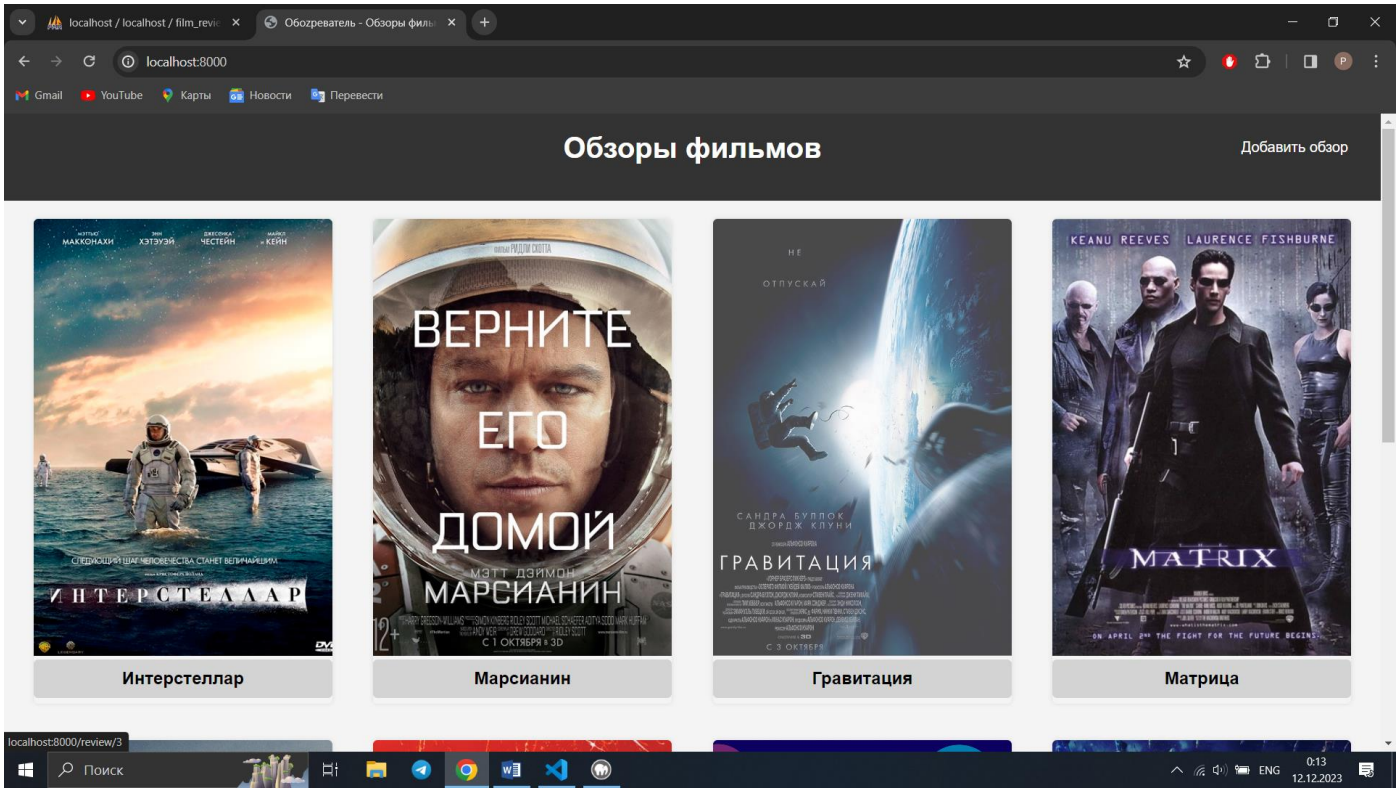
Таблица films в БД MySQL

+ Options

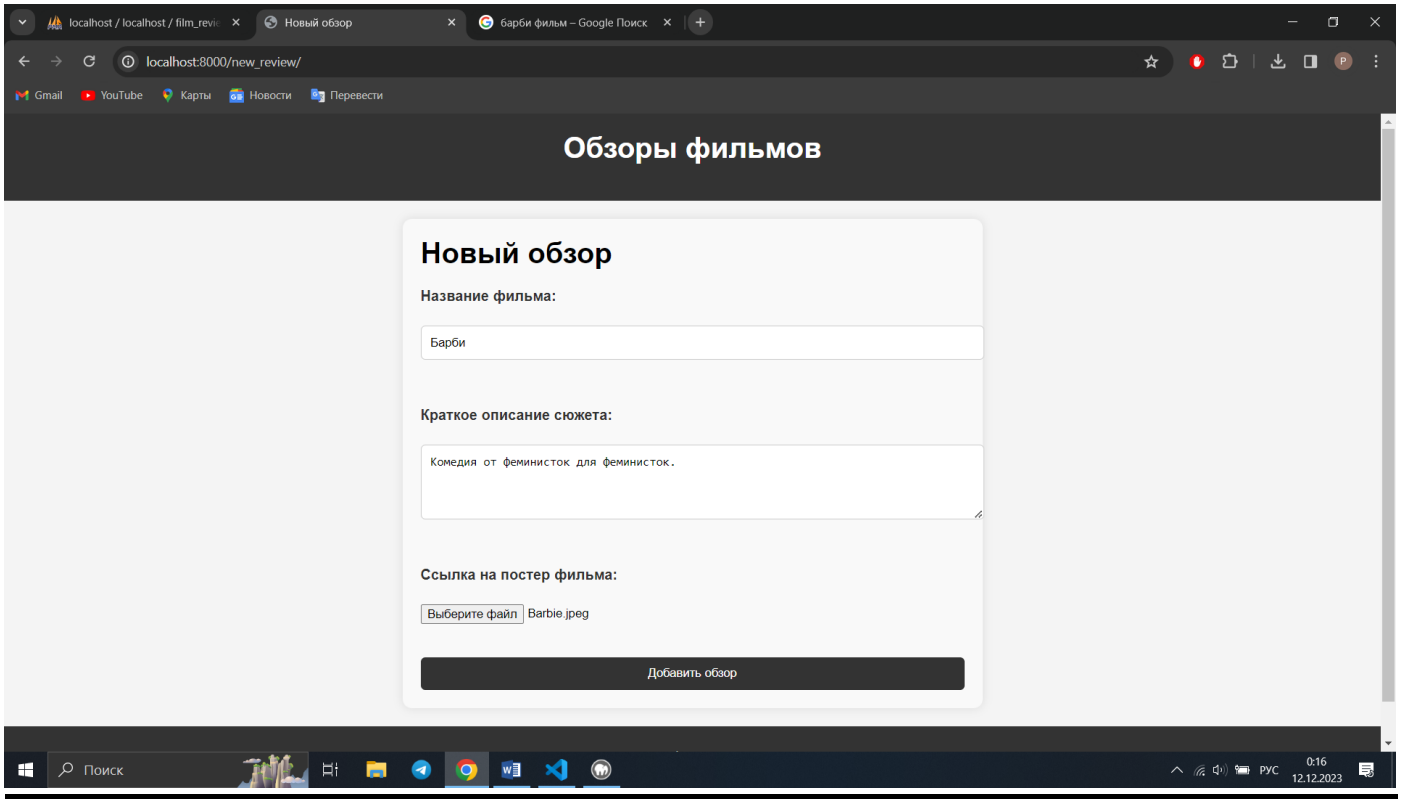
	id	name	plot	photo
Edit Copy Delete	1	Интерстеллар	"Интерстеллар" - это захватывающее кинематографиче...	images/film/Интерстеллар.jpg
Edit Copy Delete	2	Марсианин	"Марсианин" — путеводная звезда среди космических ...	images/film/Марсианин.jpg
Edit Copy Delete	3	Гравитация	"Гравитация" — это кинематографическое чудо, котор...	images/film/Гравитация.jpg
Edit Copy Delete	4	Матрица	"Матрица" – величественный фильм, кульминация гени...	images/film/Матрица.jpg
Edit Copy Delete	5	Прибытие	"Прибытие" - это шедевр, который, словно музыкальн...	images/film/Прибытие.jpg
Edit Copy Delete	17	Криминальное чтиво	"Путь праведника недостижим, так как извилисты пут...	images/film/Криминальное_чтиво.jpg
Edit Copy Delete	16	Головоломка	Задумывались ли вы, откуда берутся эмоции? "Голово...	images/film/Головоломка.jpg
Edit Copy Delete	18	Вратарь Галактики	Едва ли остались те, кто, видя в новостных строках...	images/film/Вратарь_Галактики.jpg

Анализ результатов

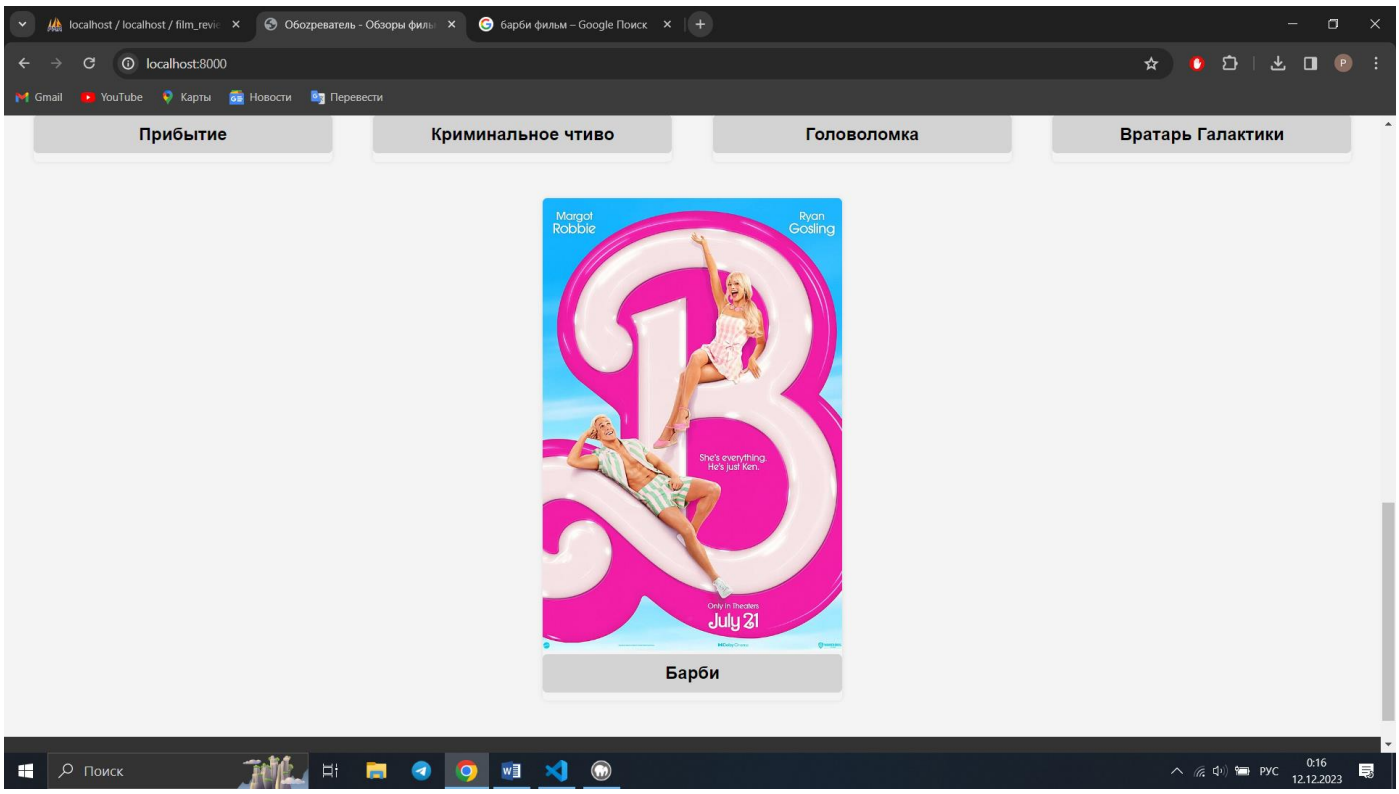
1) Главная страница



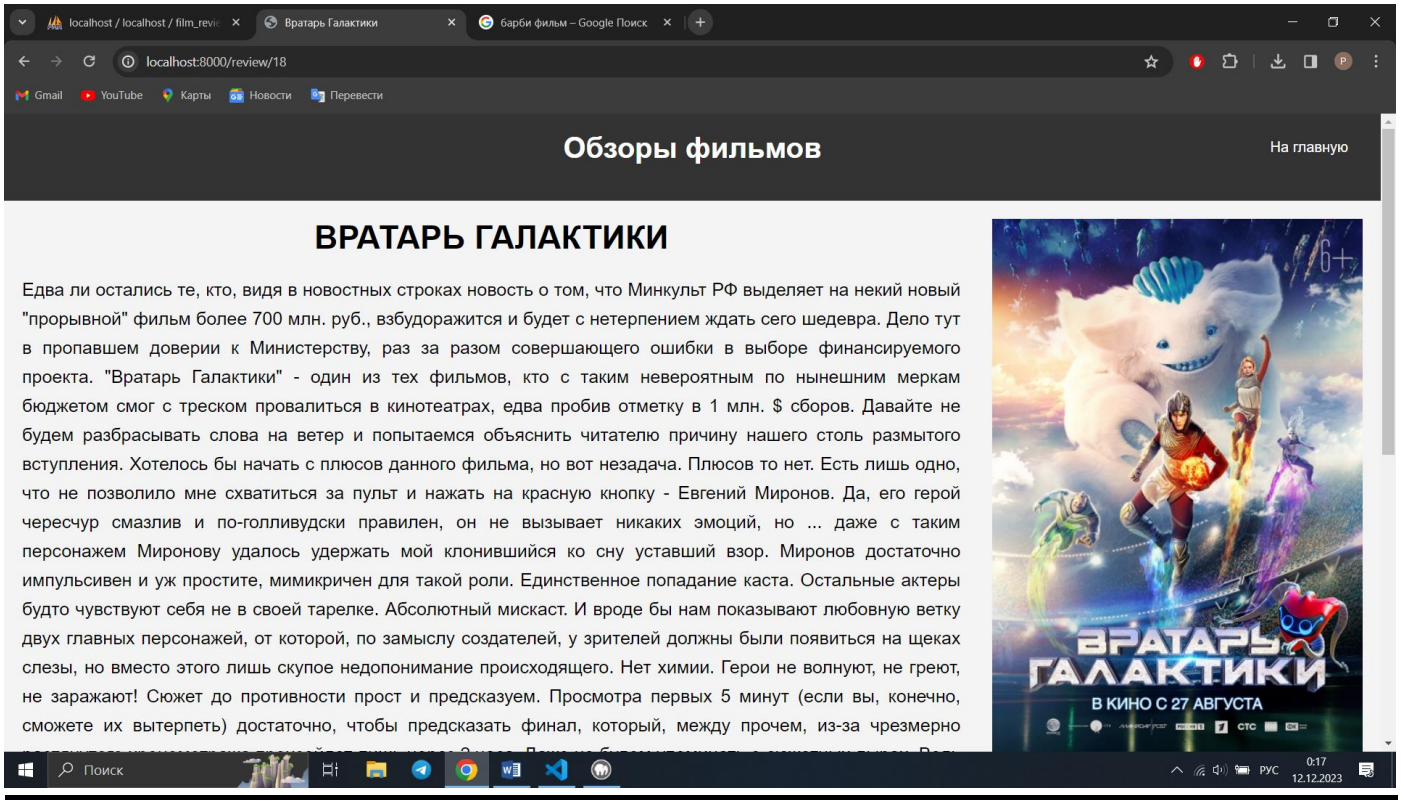
2) Добавление нового обзора



3) Фильм добавился на главную страницу



4) Возможность чтения обзора



Вывод

Я изучил основные возможности языка программирования Go и создал с помощью него веб сервис, который будет полезен мне в будущем. Кроме того, я поработал с СУБД MySQL с помощью инструмента phpMyAdmin.