

Национальный исследовательский университет «МЭИ»

---

Институт информационных и вычислительных технологий

Кафедра вычислительных машин, систем и сетей

Курсовая работа  
Разработка веб-приложения

Выполнил

Студент Кукушкин Евгений Андреевич

---

Группа А-12-20

---

Дата

---

Принял

Преподаватель Мишин Алексей Алексеевич

---

Оценка

---

Дата

---

Москва, 2023

Введение	3
Основная часть	4
1. Постановка задачи	4
2. Стек технологий	6
3. Разработка	7
3.1. Разработка базы данных	7
3.2. Разработка интерфейса пользователя	10
3.3. Разработка интерфейса администратора	11
4. Ограничения для пользователя и администратора	14
4.1. Запрет перехода на личную страницу	14
4.2. Корректность get-запросов	14
4.3. Проверка при авторизации	15
Заключение	17
Список литературы	18
Приложение	19
Приложение А	19
Приложение Б	24
Приложение В	25

## **Введение**

Сегодня невозможно представить мир без интернета. Заходя в поисковую систему, мы встречаем обилие всевозможных сайтов. Все они представляют пользователям информацию. Это может быть новости, учебники, форумы, социальные сети и т.д. Чтобы создать сайт, необходимо создать веб-сервер, который будет являться «посредником» между пользователем и данными сайта. Пользователь взаимодействует с интерфейсом, отправляя различные запросы, со стороны сервера запрос обрабатывается, и отправляется ответ.

## **Основная часть**

### **1. Постановка задачи**

В рамках курсовой работы необходимо разработать веб-приложение, которое будет специализировано на предоставлении пользователям новостной информации. При проектировании приложения необходимо разработать удобный пользовательский интерфейс, а также интерфейс для администратора и его функционал.

Для пользователя и администратора будут разработаны:

1. Главная страница сайта, где будет представлена новостная информация, которая может быть отсортированная по типу новостей (Европа, Культура, Образование, Россия, Спорт), а также вход на личную страницу пользователя
2. Страница пользователя, интерфейс которого будет схож с главной страницей с добавлением функционала для опубликования новых новостей
3. Страница администратора, в которой он может просматривать новости, и решать, опубликовывать их на сайт или нет

Чтобы хранить данные о пользователях и новостях необходимо разработать базу данных, в которой будут присутствовать:

#### **1. Информация о пользователе:**

- 1) Логин
- 2) Пароль (зашифрованный)
- 3) Почта
- 4) Рейтинг на сайте

#### **2. Информация о новостях:**

- 1) Автор новости
- 2) Тип новости (Европа, Культура, Образование, Россия, Спорт)
- 3) Изображение новости (необязательно)

4) Заголовок (краткая информация, которая будет отображаться на сайте)

5) Текстовый файл (полное описание новости)

6) Дата публикации (год - месяц - день)

7) Подтверждение публикации (будет ли новость отображаться на сайте: принято, отказ, в рассмотрении)

## 2. Стек технологий

Для создания веб-сервера используют связку программ - **стек технологий**, включающая в себя веб-сервер, язык программирования, базу данных и операционную систему. В качестве такой связки будут взяты:

1. Apache HTTP Server - веб-сервер
2. PHP - язык программирования для разработки веб-приложений
3. PostgreSQL - база данных
4. MacOS - операционная система

Подробное установка и проверка веб-сервера представлены в приложении

А.

### 3. Разработка

В ходе проектирования были разработаны:

1. База данных
2. Интерфейс пользователя
3. Интерфейс администратора

#### 3.1. Разработка базы данных

База данных представляет собой совокупность таблиц, в которых хранится информация, и связей между ними:

1. Таблица пользователей (*users*):

Столбцы таблицы *users* и их ограничения/модификации представлены в табл. 1.

Таблица 1

Описание столбцов таблицы пользователей

Название	Тип	Ограничения/Модификации
Id	Целочисленный	Автоинкремент, не может быть пустым
Login	Строка	Количество символов [10, 30], не может быть пустым
Email	Строка	Количество символов [10, 30], не может быть пустым, не может повторяться у разных записей
Password	Строка	Количество символом не ограничено, не может быть пустым
Raiting	Целочисленный	Не может быть пустым, по умолчанию присваивается значение 100

В работе приложения приходится часто искать пользователя по почте, например, при входе в аккаунт. В связи с этим был создан индекс по столбцу *email* для увеличения производительности.

2. Таблица новостей (*news*):

Прежде чем создавать итоговую таблицу *news*, необходимо ее нормализовать. Для этого были созданы дополнительные таблицы:

### 1) Таблица типов новостей (*type\_news*):

Столбцы таблицы *type\_news* и их ограничения/модификации представлены в табл. 2.

Таблица 2

Описание столбцов таблицы типов новостей

Название	Тип	Ограничения/Модификации
Id	Целочисленный	Автоинкремент, не может быть пустым
Type	Строка	Количество символов до 20, не может быть пустым

Так как количество типов новостей на сайте имеет фиксированное значение, то сразу создадим необходимые данные. Сведения о данных таблицы *type\_news* представлены на рис. 1.

	id [PK] integer	type character varying (20)
1	1	Европа
2	2	Россия
3	3	Спорт
4	4	Культура
5	5	Образование

Рис. 1. Данные таблицы *type\_news*

### 2) Таблица подтверждения (*confirms*):

Столбцы таблицы *confirms* и их ограничения/модификации представлены в табл. 3.

Таблица 3

Описание столбцов таблицы подтверждения

Название	Тип	Ограничения/Модификации
Id	Целочисленный	Автоинкремент, не может быть пустым
Type	Строка	Количество символов до 20, не может быть пустым

Так как количество подтверждений на сайте имеет фиксированное значение, то сразу создадим необходимые данные. Сведения о данных таблицы *confirms* представлены на рис. 2.

Теперь можем создать таблицу новостей. Столбцы таблицы *news* и их ограничения/модификации представлены в табл. 4.



	id [PK] integer	type character varying (20)
1	1	принято
2	2	отказ
3	3	в рассмотрении

Рис. 2. Данные таблицы confirms

Таблица 4

#### Описание столбцов таблицы новостей

Название	Тип	Ограничения/Модификации
Id	Целочисленный	Автоинкремент, не может быть пустым
Id_author	Целочисленный	Внешний ключ, связанный с таблицей <i>users</i>
Id_type_new	Целочисленный	Внешний ключ, связанный с таблицей <i>type_news</i>
Image	Строка	Количество символов до 30, может быть пустым
Title	Строка	Количество символов [5, 100], не может быть пустым
text_file	Строка	Количество символов до 30, не может быть пустым
Data	Дата	Не может быть пустым, по умолчанию присваивается значение текущей даты
id_confirm	Целочисленный	Внешний ключ, связанный с таблицей <i>confirms</i>

В работе приложения приходится часто искать новость по подтверждению и дате. В связи с этим были созданы индексы по столбцу *id\_confirm* и *data* для увеличения производительности.

При отказе от публикации новости администратором необходимо удалять эти данные. Так был разработан триггер, который при присвоении столбцу *id\_confirm* значение 2 (отказ), будет автоматически удалять эти данные из таблицы *news*.

Создание всех таблиц, индексов и триггера представлены в приложении Б.

### 3.2. Разработка интерфейса пользователя

Интерфейс пользователя представляет собой главную страницу и личную страницу, представленную на рис. 3.

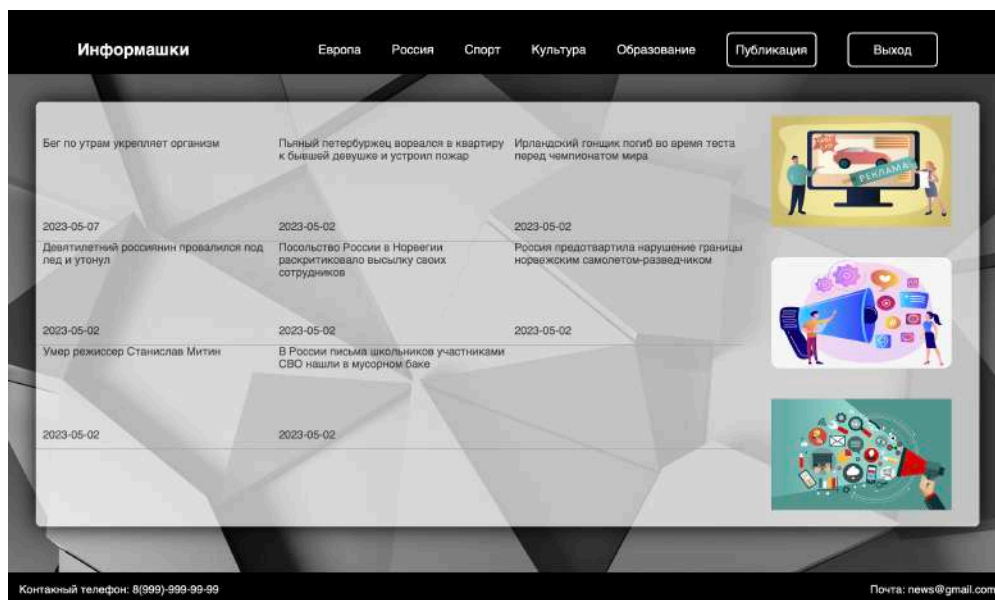


Рис. 3. Личная страница

Интерфейс на рис.3, 5 разбит на 3 части:

1. Заголовок
2. Содержимое
3. Контактная информация

В заголовке представлена навигация по сайту, а также пользовательский функционал. При нажатии на кнопку «Публикация» всплывает окно, в котором пользователю представлено вести информацию, которую он бы хотел опубликовать. Всплывающее окно представлено на рис. 4.

При нажатии на кнопку «Выход» пользователя переносят на главную страницу, представленную на рис. 5.

При нажатии на кнопку «Вход» всплывает окно для входа в аккаунт или регистрация. Всплывающее окно представлено на рис. 6.



Рис. 4. Всплывающее окно публикации

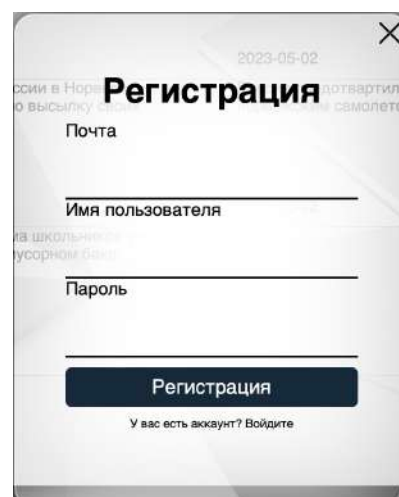
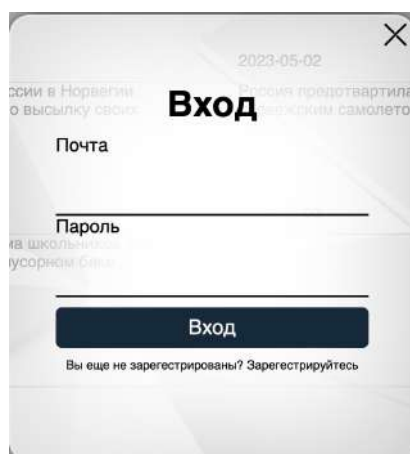


Рис. 6. Всплывающее окно входа/регистрации

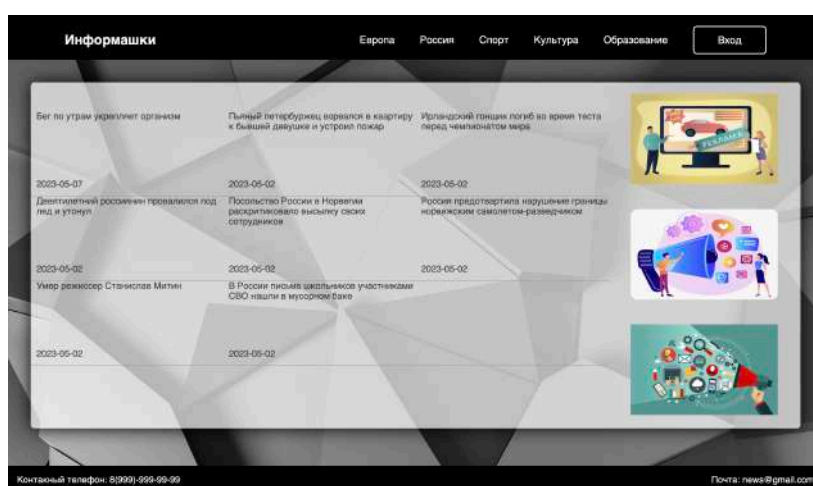


Рис. 5. Главная страница

В содержании располагаются ссылки на новости. При нажатии на текст пользователю раскрывается полная информация содержимого, представленного на рис. 7. В качестве полного описания пользователю представляется фотография (при наличии), заголовок и основной текст выбранной новости.

### 3.3. Разработка интерфейса администратора

Страница администратора представлена на рис. 8.

Интерфейс администратора разбит на 3 секции:

1. Список новостей в рассмотрении
2. Поле для просмотра новости

### 3. Функционал администратора (кнопки: Разрешить, Отклонить, Выйти)

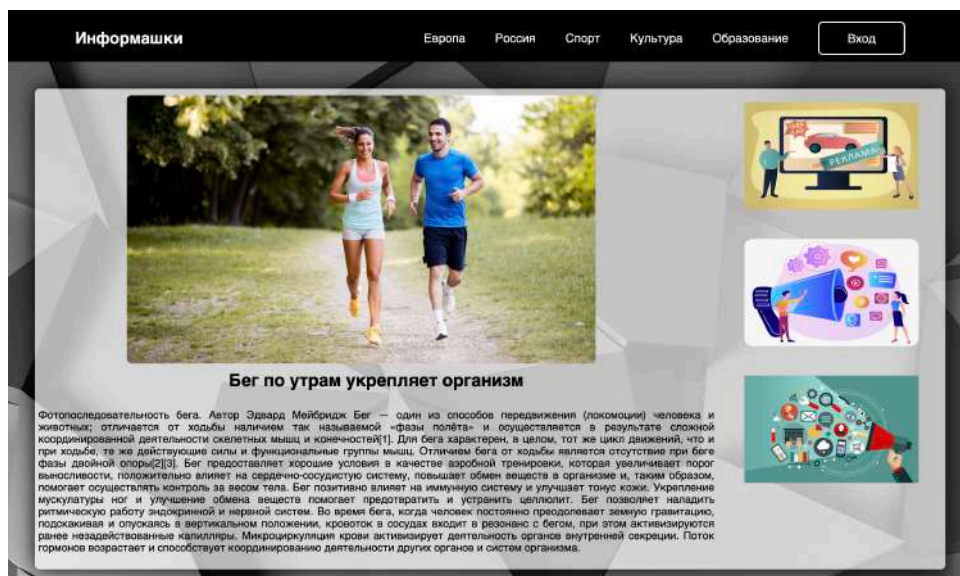


Рис. 7. Полное описание новости

При нажатии на любую новость в поле просмотра появляется полная информация о новости (содержимое, тип, автор). Администратор, руководствуясь своими соображениями, нажимает на кнопки «Разрешить» или «Отклонить». Если была нажата кнопка «Разрешить», то в базе данных этой

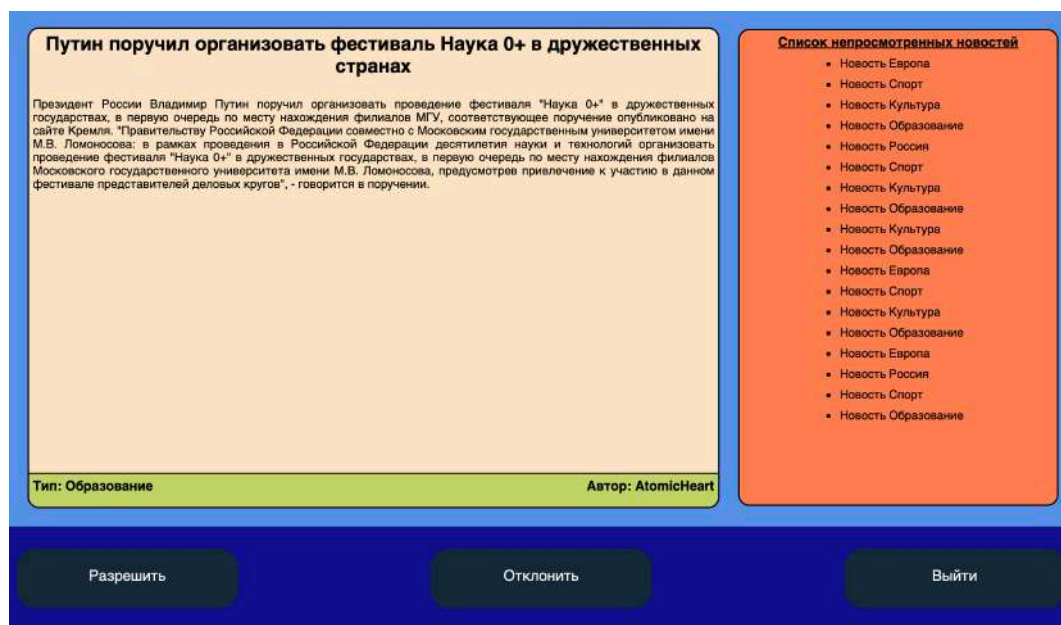


Рис. 8. Страница администратора

новости устанавливается столбец *id\_confirm* с 3 (в рассмотрении) на 1 (принято). В последствии, эту новость увидят все пользователи на страницах

сайта. Если была нажата кнопка «Отклонить», то записи об этой новости удаляются из базы данных, также связанные с ней файлы удаляются с сервера. При разрешении или отклонении у пользователя, отправивший новость, увеличивается или уменьшается рейтинг соответственно.

При нажатии на кнопку «Выход» переходим на страницу входа администратора, показанный на рис. 9.

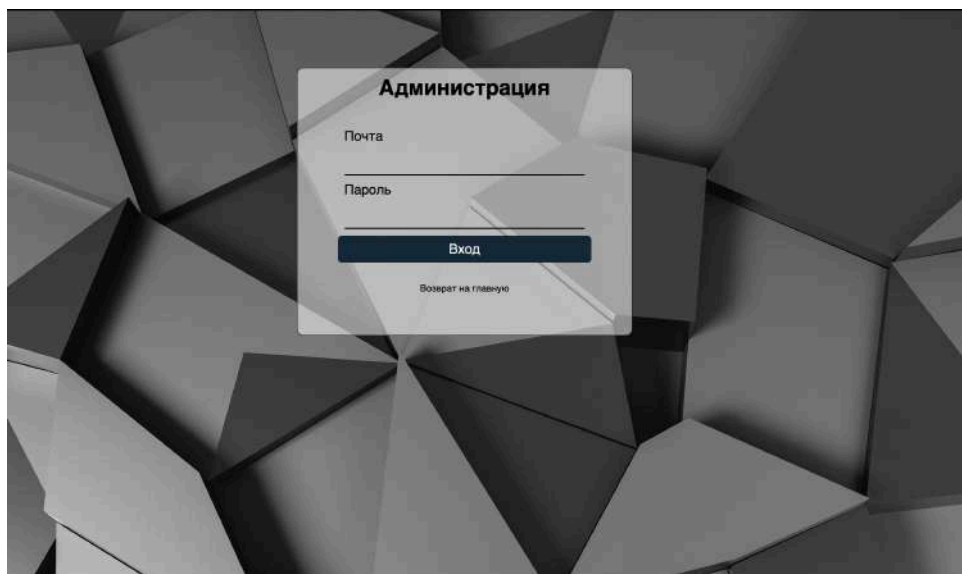


Рис. 9. Вход администратора

Все файлы стилей, функций и разметок представлены в приложении В.

## **4. Ограничения для пользователя и администратора**

### **4.1. Запрет перехода на личную страницу**

Рассмотрим такую ситуацию, первый пользователь авторизировался, сделал свои дела и вышел с аккаунта. В истории браузера у него сохранилась ссылка на его страницу. В какой-то момент другой пользователь как-то решил этой ссылкой воспользоваться и успешно зашел по ней на чужой аккаунт, так как никакой проверки на сайте нет. Так, в целях безопасности пользователя был разработан алгоритм проверки, находится ли сейчас пользователь на странице или же нет.

Алгоритм заключается в том, что создается сессия для вошедшего пользователя, а именно, переменная `$_SESSION['user']`, в которой будет храниться некая информация о пользователе. Эта информация хранится на сервере, так что ее просто так невозможно найти.

Рассмотрим случаи, когда эта переменная есть. Если пользователь захочет вернуться на главную страницу рис.5, при этом сделав это, не нажав кнопку выйти, то его обратно вернет на его страницу рис.3. Аналогично происходит со страницами, связанные с администратором.

Другой случай, когда переменной нет. Тогда если пользователь воспользуется ссылкой на чью-либо страницу, его перекинет обратно на главную. Тоже самое касается страниц для администратора.

Все те же разъяснения реализуются и для безопасности страниц администратора. При этом будет фигурировать переменная `$_SESSION['admin']`.

Удаление этих переменных происходит после того, как пользователь или администратор нажали на кнопку выйти.

### **4.2. Корректность get-запросов**

В данной работе часто приходится иметь дело с передачей какой-либо информации, которая отражает сущность тех или иных данных, например, вывод на экран только новостей про образование. Так, информация, которая передается через get-запрос видна в адресе страницы, является не безопасной. Поэтому на коррекцию данных в таком запросе происходит проверка:

1. Есть ли необходимые данные для отображения информации (необходимые переменные)
2. Корректны ли данные с точки зрения их типа (число, строка)
3. Можно ли по этим данным найти информацию из базы данных (делается запрос в базу и проверяется наличие данных)

При не соблюдении одного из перечисленных пунктов будет выдаваться значения по умолчанию или же пустая страница без данных. Пустая страница представлена на рис. 10.

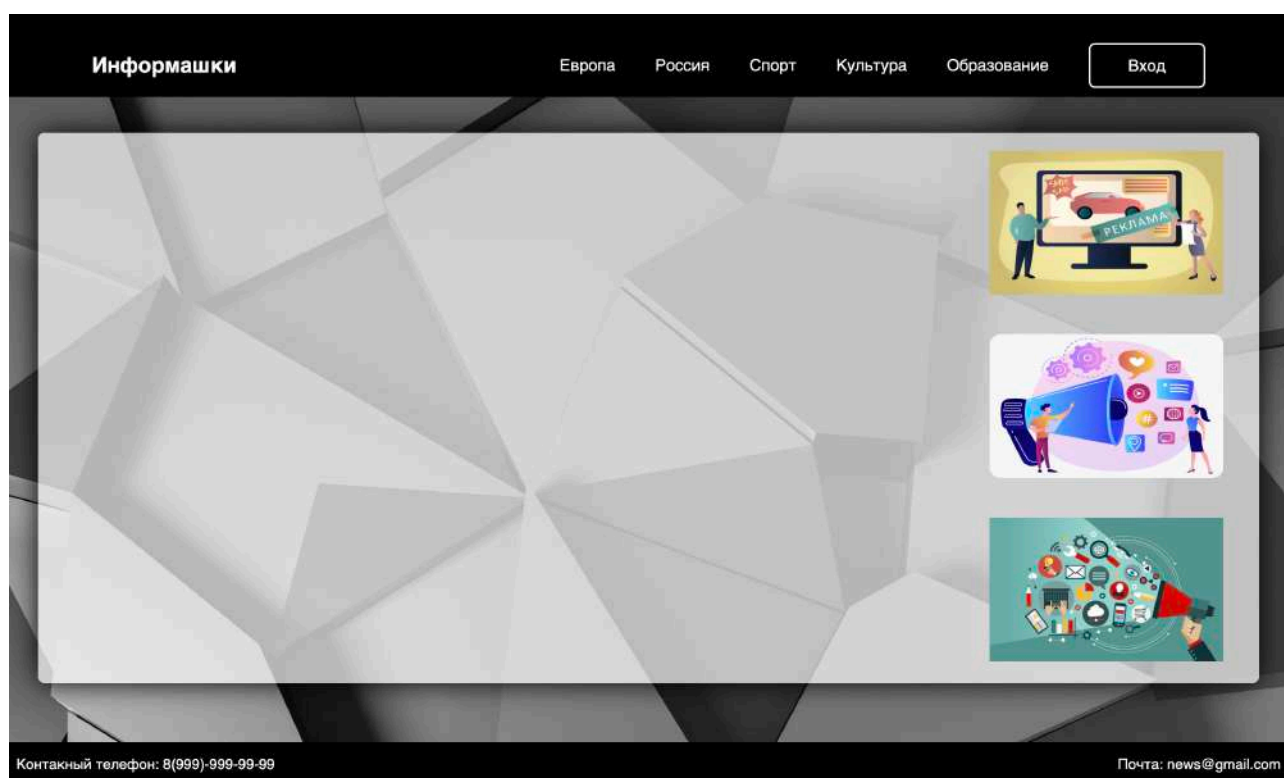


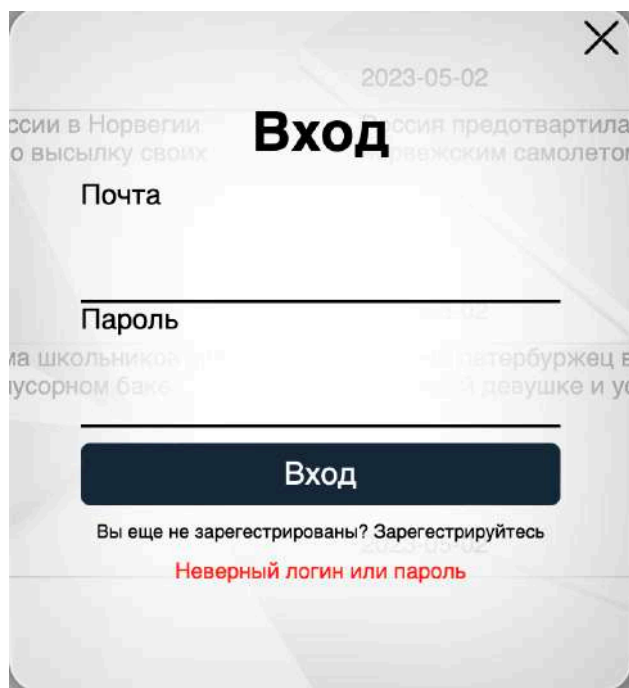
Рис. 10. Пустая страница (указан некорректный get-запрос)

### 4.3. Проверка при авторизации

Первая защита личных данных - указание логина и пароля. При заходе на личную страницу пользователь указывает свои данные или создает себе новую страницу рис. 6. В целях безопасности все пароли, находящиеся в базе данных, зашифрованы. При нажатии на кнопку войти проверяется правильность логина и пароля. При отсутствии информации в базе данных выводится предупреждения о некорректных данных рис. 11(а). При регистрации



проверяется почта, если она уже используется, то выводится предупреждение об этом рис. 11(б).



2023-05-02

## Вход

Почта

Пароль

**Вход**

Вы еще не зарегистрированы? Зарегистрируйтесь

Неверный логин или пароль

а)



2023-05-02

## Регистрация

Почта

Имя пользователя

Пароль

**Регистрация**

У вас есть аккаунт? Войдите

Под этой почтой уже авторизовались

б)

Рис. 11. Некорректные данные



## **Заключение**

Был разработан и протестирован новостной ресурс «Информашки». В качестве дальнейшей доработки функционала за пользователем закреплён его рейтинг, благодаря которому можно будет блокировать страницы недобросовестных пользователей.

## **Список литературы**

1. <https://metanit.com/php/tutorial/>
2. <https://webrazrab.ru/html/>
3. <http://htmlbook.ru/html/>
4. <https://developer.mozilla.org/ru/docs/Learn/HTML>
5. <https://www.php.net/>
6. tz\_web\_2023.pdf

## Приложение

### Приложение А

С версии macOS Monterey из операционной системы был удален PHP, который устанавливался по умолчанию в предыдущих версиях. Также во встроенном сервере в настройке файла конфигурации может не доставать некоторых пакетов. В связи с этим установка веб-сервера будет осуществляться через пакетный менеджер **Homebrew**. Для начала необходимо его установить, вводим в терминале команду установки:

```
/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
```

Чтобы убедиться, что пакет Homebrew установлен, а также его версию, используем следующую команду:

```
brew --version
```

```
[evgeniy@MacBook-Air-Evgenij ~ % brew --version  
Homebrew 4.0.5  
Homebrew/homebrew-core (git revision a96802675e1; last commit 2023-03-09)
```

Теперь мы можем устанавливать необходимые пакеты для нашего веб-сервера. Для установки PHP и Apache воспользуемся следующими командами, соответственно:

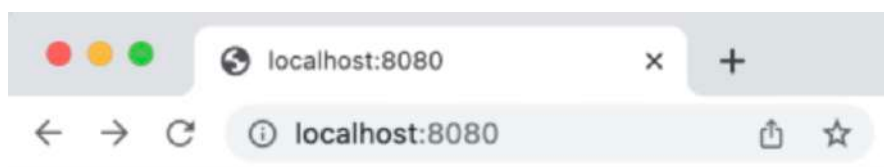
```
brew install php
```

```
brew install httpd
```

После установки Apache необходимо проверить его работоспособность. Запуск сервера осуществляется командой:

```
brew services start httpd
```

При удачном запуске в браузере при переходе на адрес <http://localhost:8080> увидим надпись:



## It works!

Также стоит упомянуть некоторые другие команды для сервера, а именно, остановка работы и перезагрузка сервера, соответственно:

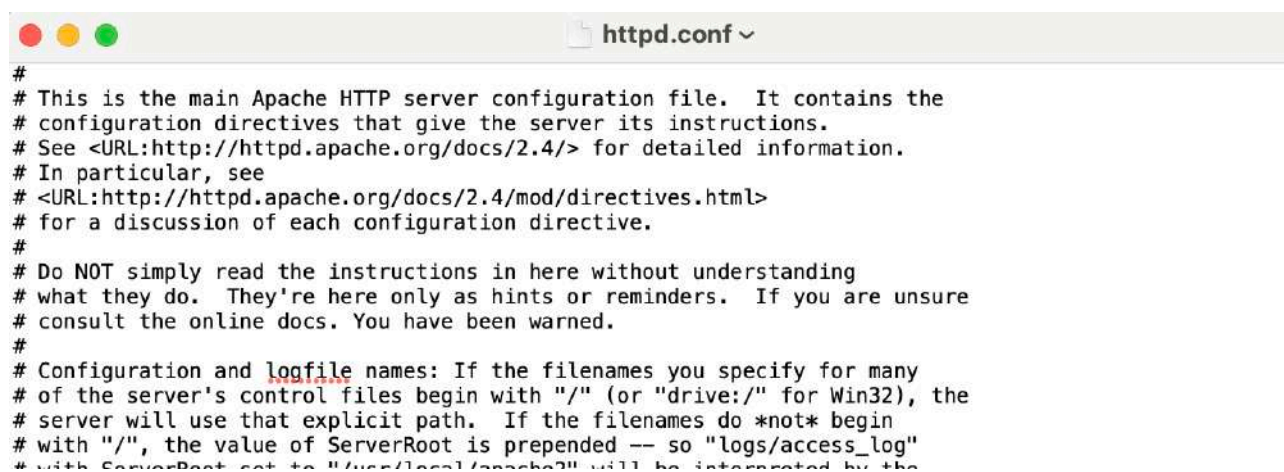
```
brew services stop httpd
```

```
brew services restart httpd
```

### 1.1. Настройка Apache

Настройка сервера осуществляется изменением файла конфигурации. Откроем его для дальнейшего изменения следующей командой:

```
open -e /usr/local/etc/httpd/httpd.conf
```



Теперь, пролистывая файл, будем изменять соответствующие строчки. Изменим порт, по которому можно будет обращаться к веб-серверу:

```
Listen 8080 —> Listen 80
```

Дальше настроим для удобства использования директорию для файлов веб-сервера. Создадим эту директорию под названием **localhost** и изменим файл конфигурации:

*DocumentRoot «/usr/local/var/www" —> DocumentRoot «/Users/evgeniy/localhost"*

*<Directory «/usr/local/var/www"> —> <Directory «/Users/evgeniy/localhost">*

Установим имя сервера на **localhost**:

*#ServerName www.example.com:8080 —> ServerName localhost*

Также изменим пути к файлам, в которые заносятся сведения об ошибках и посещения сайта, соответственно:

*ErrorLog "/usr/local/var/log/httpd/error\_log" —> ErrorLog «/Users/evgeniy/localhost/error.log"*

*CustomLog "/usr/local/var/log/httpd/access\_log" common —> CustomLog "/Users/evgeniy/localhost/access\_log" common*

## 1.2. Настройка PHP

Теперь необходимо связать Apache с PHP. Для начала узнаем версию PHP следующей командной:

*php -v*

```
[evgeniy@MacBook-Air-Evgenij ~ % php -v
PHP 8.2.3 (cli) (built: Feb 15 2023 00:34:20) (NTS)
Copyright (c) The PHP Group
Zend Engine v4.2.3, Copyright (c) Zend Technologies
with Zend OPcache v8.2.3, Copyright (c), by Zend Technologies
```

Узнав версию PHP, продолжим изменять наш файл конфигурации. Добавим модуль **php@8.2**. Вставим после строки **#LoadModule rewrite\_module lib/httpd/modules/mod\_rewrite.so** следующее:

*LoadModule php\_module /usr/local/opt/php@8.2/lib/httpd/modules/libphp.so*

Добавим поддержку для файлов с расширением **.php** и **.phps**. Найдем блок **<IfModule mime\_module>** и вставим внутри него следующее:

*AddType application/x-httpd-php .php*

*AddType application/x-httpd-php-source .phps*

Дальше изменим обращение к главной странице веб-сайта, а именно, найдем блок `<IfModule dir_module>` и допишем строку:

*DirectoryIndex index.html —> DirectoryIndex index.html index.php*

Теперь при обращении к главной странице веб-сервера можно использовать как файл с расширением `.html`, так и `.php` с названием `index`.

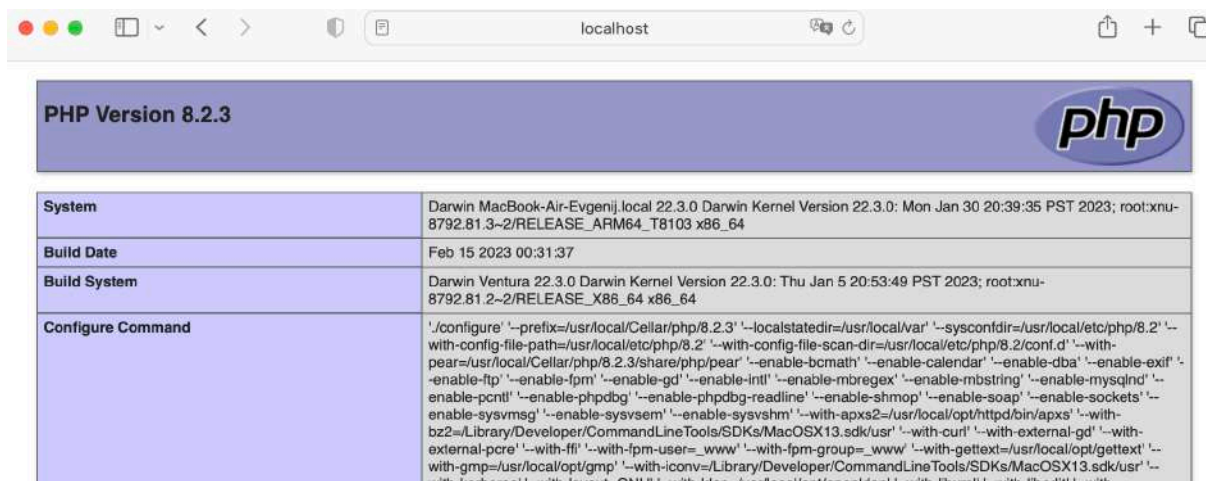
После надо убедиться, что PHP подключен к серверу правильно. Для этого создадим файл `index.php` в нашей директории `~/localhost` и напишем небольшой скрипт:

```
index.php ×
Users > evgeniy > localhost > index.php
1  <?php
2  phpinfo();
3  ?>
```

Перезагрузим сервер уже известной нам командой:

*brew services restart httpd*

На экране появится информационная справка о версии PHP:



Также стоит упомянуть о том, что в дальнейшем может понадобится изменять конфигурацию php. Для этого надо открыть файл `php.ini`:

*open -e /usr/local/etc/php/8.2/php.ini*

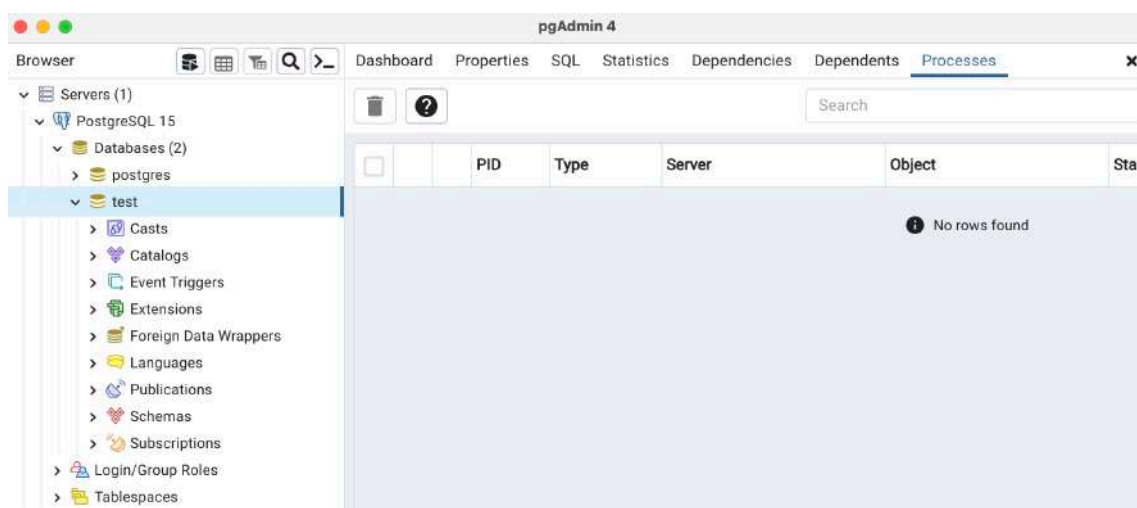
```
[PHP]

;::::::::::::::::::;
; About php.ini    ;
;::::::::::::::::::;
; PHP's initialization file, generally called php.ini, is responsible for
; configuring many of the aspects of PHP's behavior.

; PHP attempts to find and load this configuration from a number of locations.
; The following is a summary of its search order:
; 1. SAPI module specific location.
; 2. The PHPRC environment variable.
; 3. A number of predefined registry keys on Windows
```

### 1.3. Установка PostgreSQL

Для управления базой данных(БД) будем использовать *pgAdmin 4*:



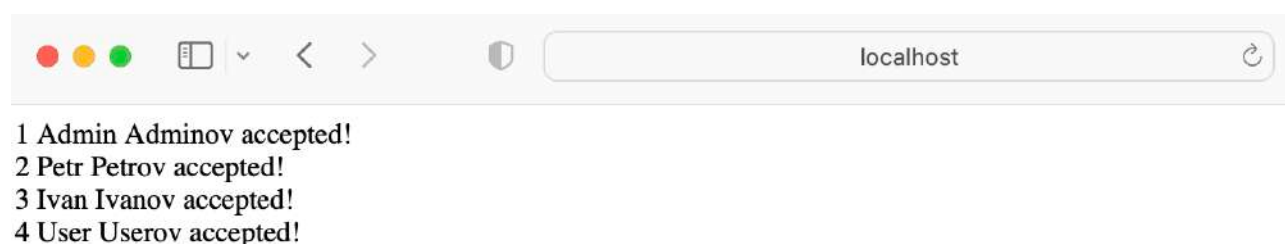
Проверим подключение БД к нашему веб-серверу. Для этого создадим тестовую БД *test* и создадим простую таблицу *hello* и вставим в нее данные:

id [PK] integer	first_name character varying (255)	last_name character varying (255)
1	Admin	Adminov
2	Petr	Petrov
3	Ivan	Ivanov
4	User	Userov

Дальше создадим файл *test.php* и напишем скрипт:

```
test.php x
Users > evgeniy > localhost > test.php
1  <?php
2  $connect_string = "host=localhost port=5433 dbname=test user=postgres password=1065";
3  $dbconn = pg_connect($connect_string);
4  $result = pg_query($dbconn, "SELECT * FROM hello");
5  while ($row = pg_fetch_row($result))
6  {
7      foreach($row as $col)
8      {
9          echo "$col ";
10     }
11     echo "accepted!<br>";
12 }
13 ?>
```

При обращении по адресу <http://localhost/test.php> увидим вывод значений таблицы *hello*



## Приложение Б

1. Таблица пользователей

```
CREATE TABLE users (
    id SERIAL,
    login VARCHAR(30) NOT NULL CHECK (char_length(login) >= 10),
    email VARCHAR(30) NOT NULL CHECK (char_length(email) >= 10),
    password TEXT NOT NULL,
    rating INT NOT NULL DEFAULT 100,
    CONSTRAINT users_id_pkey PRIMARY KEY(id),
    CONSTRAINT users_email_unique UNIQUE(email)
)
```

Индекс по почте

```
CREATE INDEX ON users (email)
```

2. Таблица типов новостей

```
CREATE TABLE type_news (
    id SERIAL,
    type VARCHAR(20) NOT NULL,
    CONSTRAINT type_news_id_pkey PRIMARY KEY(id)
)
```

Вставка типов новостей (неизменна)

```
INSERT INTO type_news (type) VALUES ('Европа'), ('Россия'), ('Спорт'), ('Культура'), ('Образование')
```

3. Таблица соглашений



```
CREATE TABLE confirms (
    id SERIAL,
    type VARCHAR(20) NOT NULL,
    CONSTRAINT confirms_id_pkey PRIMARY KEY(id)
)
```

Вставка типов соглашений (неизменна)

```
INSERT INTO confirms (type) VALUES ('принято'), ('отказ'), ('в рассмотрении')
```

4. Таблица новостей

```
CREATE TABLE news (
    id SERIAL,
    id_author INT NOT NULL,
    id_type_new INT NOT NULL,
    image VARCHAR(30),
    title VARCHAR(100) NOT NULL CHECK (char_length(title) >= 5),
    text_file VARCHAR(30) NOT NULL,
    data DATE DEFAULT current_date,
    id_confirm INT NOT NULL DEFAULT 3,
    CONSTRAINT news_id_pkey PRIMARY KEY(id),
    CONSTRAINT news_author_fkey FOREIGN KEY (id_author)
    REFERENCES users (id),
    CONSTRAINT news_type_new_fkey FOREIGN KEY (id_type_new)
    REFERENCES type_news (id),
    CONSTRAINT news_confirm_fkey FOREIGN KEY (id_confirm)
    REFERENCES confirms (id)
)
```

Индекс по соглашению

```
CREATE INDEX ON news (id_confirm)
```

Индекс по соглашению и дате

```
CREATE INDEX ON news (id_confirm, data DESC)
```

Функция для триггера

```
CREATE FUNCTION remove_news() RETURNS TRIGGER
AS $$
BEGIN
    IF (NEW.id_confirm = 2) THEN
        DELETE FROM news WHERE id = NEW.id;
    END IF;
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;
```

Триггер для удаления непринятой новости

```
CREATE TRIGGER check_reject
AFTER UPDATE
ON news
FOR EACH ROW
WHEN (NEW.id_confirm <> OLD.id_confirm)
EXECUTE PROCEDURE remove_news();
```

## Приложение В

### Файлы .css

header.css

```
.section__header{
    position: relative;
    flex: 0 0 auto;
    width: 100%;
    height: 80px;
```

```

padding: 20px 100px;
top: 0;
left: 0;
background-color: black;
}
.header{
background-color: transparent;
display: flex;
justify-content: space-between;
align-items: center;
z-index: 100;
min-width: 835px;
}

.header__logo{
color: white;
user-select: none;
}

.header__logo a{
position: relative;
color: white;
text-decoration: none;
font-weight: 800;
}

/*настройка меню*/
.header__navigation .navig{
position: relative;
font-size: 18px;
color: white;
text-decoration: none;
font-weight: 500;
margin-left: 40px;
}

.header__navigation .navig::after{
content: "";
position: absolute;
left: 0;
bottom: -6px;
width: 100%;
height: 2px;
background: white;
border-radius: 7px;
transform: scale(0);
transition: 0.5s;
}

.header__navigation .navig:hover::after{
transform: scale(1);
}

.header__navigation .header__btn-login{
position: relative;
width: 130px;
height: 50px;
background: transparent;
border: 2px solid white;
outline: none;
border-radius: 7px;
color: white;
cursor: pointer;
font-size: 18px;
text-align: center;
justify-content: center;

```

```

    font-weight: 500;
    margin-left: 40px;
    transition: 0.5s;
}

.header__navigation .header__btn-login a{
    position: relative;
    width: 100px;
    height: 50px;
    color: inherit;
    text-decoration: none;
    font-weight: 500;
    padding: 12px 35px;
}

.header__navigation .header__btn-login:hover,
.header__navigation .header__btn-login .exit:hover{
    color: black;
    background: white;
}

/*настройка модального окна регистрации/входа*/
.wrapper{
    position: fixed;
    width: 400px;
    height: 440px;
    top: 30%;
    left: 35%;
    background: transparent;
    border: 2px solid rgba(255, 255, 255, 0.509);
    border-radius: 20px;
    overflow: hidden;
    box-shadow: 0 0 30px rgba(0, 0, 0, 1);
    display: flex;
    justify-content: center;
    align-items: center;
    transform: scale(0);
    transition: 0.5s;
    z-index: 10;
}

.wrapper.active-btn{
    transform: scale(1);
}

/*задний фон модального окна*/
.wrapper::before{
    background-color: rgba(255, 255, 255, 1);
    content: "";
    position: absolute;
    filter: blur(65px);
    width: 100%;
    height: 100%;
    z-index: -1;
}

/*крестик*/
.wrapper .icon-close{
    position: absolute;
    top: 0;
    right: 0;
    width: 35px;
    height: 35px;
    border-bottom-left-radius: 12px;
    background-image:
    url(../image/crest.svg);

```

```

}

.wrapper .icon-close button{
  position: absolute;
  width: 35px;
  height: 35px;
  border: none;
  background-color: transparent;
  outline: none;
  cursor: pointer;
  z-index: 10;
}

.wrapper .register{
  width: 100%;
  padding: 50px;
}

/*смотрим что нам надо скрываем/раскрываем окно*/
.wrapper .register{
  position: absolute;
  transform: translateX(400px);
  display: none;
}

.wrapper.active .login{
  transition: none;
  transform: translateX(-400px);
  display: none;
}

.wrapper.active{
  height: 500px;
}

.wrapper.active .register{
  transition: none;
  transform: translateX(0);
  display: block;
}

/*дизайн модального окна*/
.form-box h2{
  position: relative;
  top: -10px;
  text-align: center;
  font-size: 35px;
}

.input-box{
  position: relative;
  width: 100%;
  height: 70px;
  border-bottom: 2px solid black;
  padding: 15px 0;
  margin: 10px 0;
}

.input-box label{
  position: absolute;
  top: 0%;
  left: 0px;
  transform: translateY(-50%);
  font-size: 18px;
  font-weight: 500;
  pointer-events: none;
}

```

```

    transition: 0.5s;
}

/* .input-box input:focus~label,
.input-box input:valid~label{
    top: -5px;
}*/

.input-box input{
    width: 100%;
    height: 100%;
    background: transparent;
    border: none;
    outline: none;
    font-size: 16px;
    font-weight: 600;
}

.btn{
    width: 100%;
    height: 40px;
    border: none;
    outline: none;
    border-radius: 6px;
    background-color: #162938;
    cursor: pointer;
    color: white;
    font-size: 20px;
    font-weight: 500;
    transition: 0.5s;
}

.btn:hover{
    color: aqua;
}

.login-register{
    font-size: 12px;
    text-align: center;
    margin: 10px;
}

.login-register p a{
    color: black;
    text-decoration: none;
}

.login-register a:hover{
    text-decoration: underline;
}

.publish{
    top: 10%;
    height: 600px;
}

/* Настройка select*/
.select-css {
    display: block;
    font-size: 16px;
    font-family: sans-serif;
    font-weight: 700;
    color: #444;
    line-height: 1.3;
    padding: .6em 1.4em .5em .8em; width: 100%;

```

```

max-width: 100%;
box-sizing: border-box;
margin: 0;
border: 1px solid #aaa;
box-shadow: 0 1px 0 1px rgba(0,0,0,.04);
border-radius: .5em;
appearance: none;
background-color: #fff;
}
.select-css::-ms-expand { display: none; }
.select-css:hover { border-color: #888; }
.select-css:focus { border-color: #aaa; }
box-shadow: 0 0 1px 3px rgba(59, 153, 252, .7);
color: #222;
outline: none;
}
.select-css option { font-weight: normal; }
*[dir="rtl"] .select-css, :root:lang(ar) .select-css, :root:lang(iw) .select-css {
background-position: left .7em top 50%, 0 0;
padding: .6em .8em .5em 1.4em;
}

```

### config.css

/\*настройка фона\*/

```

*{
margin: 0;
padding: 0;
box-sizing: border-box;
font-family: 'Poppins', sans-serif;
}

```

```

html, body{
height: 100%;
min-width: 1310px;
box-sizing: border-box;
}

```

```

body{
background-color: black;
}

```

```

.layout{
display: flex;
flex-direction: column;
min-width: 935px;
min-height: 100%;
justify-content: center;
align-items: center;

background: url('../image/back-img.jpg') no-repeat;
background-size: cover;
background-position: center;
}

```

```

.error_registr, .error_log, .error_publish{
color: red;
font-size: 14px;
text-align: center;
}

```

```

.error_registr.invisible, .error_log.invisible, .error_publish.invisible{
display: none;
}

```

### Main.css

```

.section__main{
position: relative;

```

```

    flex: 1 0 auto;
    width: 100%;
    padding: 30px;
    text-align: center;
    font-size: 16px;
    color: white;
}

.main-page__section{
    display: flex;
    margin: 10px;
    justify-content: space-between;
    /*flex-wrap: wrap;*/
    background-color: rgba(255, 255, 255, 0.7);
    box-shadow: 0 0 30px black;
    color: black;
    height: 100%;
    border-radius: 7px;
}

.main-container{
    width: 75%;
    height: 100%;
    margin: 0;
    display: flex;
    flex-direction: column;
    justify-content: space-between;
}

.sidebar{
    width: 25%;
    margin: 0;
}

.topnews__row{
    display: flex;
    justify-content: space-between;
    width: 100%;
}

.main-container .topnews__row:first-child{
    margin-top: 5%;
}

.card{
    position: relative;
    width: 100%;
    border-bottom: 1px solid rgb(162, 160, 160);
    align-content: center;
    align-items: center;
    text-align: left;
    padding-left: 10px;
}

.card.big{
    height: 300px;
}

.cards-mini{
    display: flex;
    width: 100%;
}

.card.mini{
    height: 150px;
}

```

```

.card-mini, .card-big{
    position: relative;
    width: 100%;
    text-decoration: none;
    color: rgba(0, 0, 0, 0.685);
    transition: 0.5s;
}

.card-mini:hover, .card-big:hover{
    color: aqua;
    text-decoration: underline;
}

.card-big{
    display: flex;
    flex-direction: column;
    width: 250px;
}

.card-big__image{
    height: 180px;
    width: 250px;
}

.card-big__title{
    height: 50px;
    overflow-y: hidden; /*remove */
}

.card-mini__title{
    height: 100px;
    overflow-y: hidden /*remove */
}

.time_new{ /* replace to .data_new*/
    position: relative;
    bottom: -20px;
    text-align: left;
}

.black{
    background-color: black;
    color: white;
}

.siderbar__image{
    padding: 20px;
    height: 200px;
    width: 300px;
}

.new_page, .image, .new_header{
    margin: 5px;
}

.new_page{
    text-align: justify;
}

.new_header{
    font-size: 26px;
    text-align: center;
}

```



```

.image{
  margin-top: 10px;
  position: relative;
  left: 13%;
  border-radius: 7px;
}
footer.css
.section__footer{
  position: relative;
  flex: 0 0 auto;
  bottom: 0;
  height: 50px;
  width: 100%;
  background-color: black;
  font-size: 16px;
  color: white;
}

.footer__info{
  display: flex;
  justify-content: space-between;
  text-align: center;
  justify-items: center;
}

.footer p{
  padding: 16px;
}

.footer a{
  text-decoration: none;
  color: transparent;
}

admin.css
*{
  margin: 0;
  padding: 0;
  box-sizing: border-box;
  font-family: 'Poppins', sans-serif;
}

html, body{
  height: 100%;
  min-width: 1440px;
  box-sizing: border-box;
}

.section{
  width: 100%;
  min-height: 100%;
  min-width: 1440px;
}

.contents{
  position: relative;
  display: flex;
  border: none;
  background-color: #5892e9;
  height: 700px;
}

.new_info{
  position: relative;
  width: 950px;
  height: 650px;
}

```

```

margin: 25px;
border: 2px solid black;
border-radius: 15px;
overflow: hidden;
}

.full_new{
position: relative;
top: 0;
height: 600px;
width: 100%;
overflow: scroll;
background-color: bisque;
border-top-right-radius: 11 px;
border-top-left-radius: 11px;
}

.info{
display: flex;
text-align: center;
height: 60px;
border-top: 2px solid rgba(0, 0, 0, 0.771);
justify-content: space-between;
background-color: rgb(195, 213, 102);
color: black;
font-size: 18px;
font-weight: 600;
}

.info div{
position: relative;
margin: 5px;
justify-content: space-between;
}

.spisok{
position: relative;
top: 27px;
height: 645px;
margin-right: 20px;
width: 440px;
border: 2px solid black;
border-radius: 15px;
background-color: coral;
overflow: scroll;
}

.spisok p{
font-size: 18px;
font-weight: 600;
text-decoration: underline;
text-align: center;
}

.spisok p, .spisok li{
padding: 5px 2px;
}

.spisok ul{
position: relative;
left: 134px;
width: 200px;
}

.spisok li {

```

```

    list-style-type:square;
    text-align: left;
}

.spisok li a{
    text-decoration: none;
    color: black;
    transition: 0.5s;
}

.spisok li a:hover{
    color: rgba(187, 255, 0, 0.581);
    font-weight: 600;
    text-decoration: underline;
}

.action{
    position: relative;
    height: 150px;
    background-color: rgb(19, 15, 145);
    display: flex;
    justify-content: space-between;
    text-align: center;
}

.action button, .action a{
    position: relative;
    margin: 10px;
    top: 20px;
    width: 300px;
    height: 80px;
    border: none;
    outline: none;
    border-radius: 25px;
    background-color: #162938;
    /* background-color: blueviolet; */
    cursor: pointer;
    color: white;
    font-size: 20px;
    font-weight: 500;
    transition: 0.5s;
}

.action a{
    text-decoration: none;
    text-align: center;
}

.action a p{
    position: relative;
    top: 25px;
}

.action button:hover, .action a:hover{
    color: aqua;
}

.new_page, .image, .new_header{
    margin: 5px;
}

.new_page{
    text-align: justify;
}

```

```

.new_header{
  font-size: 26px;
  text-align: center;
}

.image{
  margin-top: 10px;
  position: relative;
  left: 13%;
  border-radius: 7px;
}

adminLog.css
*{
  margin: 0;
  padding: 0;
  box-sizing: border-box;
  font-family: 'Poppins', sans-serif;
}
html, body{
  height: 100%;
  width: 100%;
  box-sizing: border-box;
  background-color: black;
}

.layout{
  min-height: 100%;
  background: url('../image/back-img.jpg') no-repeat;
  background-size: cover;
  background-position: center;
}

.wrapper{
  position: fixed;
  top: 10%;
  left: 30%;
  height: 400px;
  width: 500px;
  min-width: 400px;
  min-height: 400px;
  justify-content: center;
  text-align: center;
  border: 1px solid #162938;
  border-radius: 10px;
  background-color: rgba(255, 255, 255, 0.538);
}

.wrapper h1{
  position: relative;
  padding: 10px;
  height: 70px;
}

.wrapper form{
  position: relative;
  left: 10%;
  width: 80%;
  height: 250px;
  padding: 10px;
}

.error, .return{
  position: relative;
  height: 20px;
}

.error p{

```

```

    text-align: center;
    color: red;
}
.return a{
    text-decoration: none;
    font-size: 14px;
    color: black;
    transition: 0.3s;
}

.return a:hover{
    color: aqua;
    text-decoration: underline;
}

.input-box{
    position: relative;
    height: 70px;
    border-bottom: 2px solid black;
    padding: 10 px 0px;
    margin: 10px 10px;
}
.input-box label{
    position: absolute;
    top: 0%;
    left: 0px;
    font-size: 20px;
    font-weight: 500;
    pointer-events: none;
    transition: 0.5s;
}

.input-box input{
    width: 100%;
    height: 100%;
    background: transparent;
    border: none;
    outline: none;
    font-size: 16px;
    font-weight: 600;
}

.btn{
    width: 100%;
    height: 40px;
    border: none;
    outline: none;
    border-radius: 6px;
    background-color: #162938;
    cursor: pointer;
    color: white;
    font-size: 20px;
    font-weight: 500;
    transition: 0.5s;
}

.btn:hover{
    color: aqua;
}

```

## Файлы .php

### admin.php

```

<!-- Интерфейс админа -->
<?php

```

```

session_start();
if (!isset($_SESSION["admin"])){
    header("Location: adminLog.php");
}
require_once "instructAd.php";
$id = isset($_GET["id"]) ? $_GET["id"] : null;
$type = isset($_GET["type"]) ? $_GET["type"] : null;
$ath = isset($_GET["ath"]) ? $_GET["ath"] : null;
?>
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <link rel="stylesheet" href="css/admin.css">
    <title>Админ</title>
</head>
<body>
    <div class="section">
        <div class="contents">
            <div class="new_info">
                <?php
                    printNewShow($dbconn, $id, $type, $ath);
                ?>
            </div>
            <nav class="spisok">
                <p>Список непросмотренных новостей</p>
                <?php
                    printNewsAdmin($dbconn);
                ?>
            </nav>
        </div>
        <div class="action">
            <a href="updateNew.php?id=<?php echo $id?>&conf=1"><p>Разрешить</p></a>
            <a href="updateNew.php?id=<?php echo $id?>&conf=2"><p>Отклонить</p></a>
            <a href="exitAdmin.php"><p>Выйти</p></a>
        </div>
    </div>
</body>
</html>

```

#### adminCheck.php

```

<?php
    session_start();
    $admin_pass = password_hash(1234567890, PASSWORD_DEFAULT);
    $admin_email = "news@gmail.com";
    if (isset($_SESSION["user"])){
        header("Location: profile.php");
    }
    if (isset($_SESSION["admin"])){
        header("Location: admin.php");
    }
    //если пользователь случайно зашел в этот файл,
    //то возвращают на основную страницу
    if (!isset($_POST["email"])){
        header("Location: adminLog.php");
    }
    else{
        $email = htmlspecialchars($_POST["email"]);
        $password = htmlspecialchars($_POST["password"]);
        if ($email == $admin_email && password_verify($password, $admin_pass)){
            $_SESSION["admin"] = [
                "login" => $password,
                "email" => $email
            ];
            header("Location: admin.php");
        }
    }
}

```

```

    else{
        $_SESSION["error_admin"] = "Неверный логин или пароль";
        header("Location: adminLog.php");
    }
}

?>

```

### adminLog.php

```

<!-- Заход админа пароль -->
<?php
    session_start();
    if (isset($_SESSION["user"])){
        header("Location: profile.php");
    }
    if (isset($_SESSION["admin"])){
        header("Location: admin.php");
    }
?>
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <title>Вход администрации сайта</title>
    <link rel="stylesheet" href="css/adminlog.css">
</head>
<body>
    <div class="layout">
        <div class="wrapper">
            <h1>Администрация</h1>
            <form action="adminCheck.php" method="POST">
                <div class="input-box">
                    <input name="email"
                        type="email" minlength="10" maxlength="30"
                        required>
                    <label>Почта</label>
                </div>
                <div class="input-box">
                    <input name="password" type="password" minlength="10" maxlength="15" required>
                    <label>Пароль</label>
                </div>
                <button type="submit" class="btn">Вход</button>
            </form>
            <div class="return">
                <a href="index.php">
                    <p>Возврат на главную</p>
                </a>
            </div>
            <div class="error">
                <p>
                    <?php
                        if (isset($_SESSION["error_admin"])){
                            echo $_SESSION["error_admin"];
                        }
                        unset($_SESSION["error_admin"]);
                    ?>
                </p>
            </div>
        </div>
    </div>
</body>
</html>

```

### connectedBd.php

```

<?php
    $connect_string = "host=localhost port=5433 dbname=test user=postgres password=1065";

```

```

$dbconn = pg_connect($connect_string);
if (!$dbconn) {
    exit(header('Location: error404/'));
    die("Error connect to DataBase");
}
?>

```

#### exit.php

```

<?php
    session_start();

    if (isset($_SESSION["user"])){
        unset($_SESSION["user"]);
    }
    header("Location: index.php");

?>

```

#### exitAdmin.php

```

<?php
    session_start();

    if (isset($_SESSION["admin"])){
        unset($_SESSION["admin"]);
    }
    header("Location: adminLog.php");

?>

```

#### index.php

```

<?php
    session_start();
    if (isset($_SESSION["user"])){
        header("Location: profile.php");
    }
    require_once "selectBD.php";

?>
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <link rel="stylesheet" href="css/header.css">
    <link rel="stylesheet" href="css/config.css">
    <link rel="stylesheet" href="css/main.css">
    <link rel="stylesheet" href="css/footer.css">
    <title>Новости</title>
</head>

<body>
    <div class="layout">
        <div class="section__header">
            <header class="header">
                <h2 class="header__logo"><a href="index.php">Информашки</a></h2>
                <nav class="header__navigation">
                    <a class="navig" href="index.php?type=Европа">Европа</a>
                    <a class="navig" href="index.php?type=Россия">Россия</a>
                    <a class="navig" href="index.php?type=Спорт">Спорт</a>
                    <a class="navig" href="index.php?type=Культура">Культура</a>
                    <a class="navig" href="index.php?type=Образование">Образование</a>
                    <button class="header__btn-login">Вход</button>
                </nav>
            </header>

            <div class="wrapper">
                <span class="icon-close"><button></button></span>
                <div class="form-box login">
                    <h2>Вход</h2>

```



```

<form action="login.php" method="POST">
  <div class="input-box">
    <input name="email"
      type="email" minlength="10" maxlength="30"
      required>
    <label>Почта</label>
  </div>
  <div class="input-box">
    <input name="password" type="password" minlength="10" maxlength="15" required>
    <label>Пароль</label>
  </div>
  <button type="submit" class="btn">Вход</button>
  <div class="login-register">
    <p>Вы еще не зарегистрированы? <a href="#" class="register-link">Зарегистрируйтесь</a></p>
  </div>
  <p class="error_log">
    <?php
      if (isset($_SESSION["error_msg_log"])) {
        echo $_SESSION["error_msg_log"];
      }
    ?>
  </p>
</form>
</div>
<div class="form-box register">
  <h2>Регистрация</h2>
  <form action="registration.php" method="POST">
    <div class="input-box">
      <input name="email"
        type="email" minlength="10" maxlength="30"
        required>
      <label>Почта</label>
    </div>
    <div class="input-box">
      <input name="login"
        type="text" minlength="10" maxlength="20"
        required>
      <label>Имя пользователя</label>
    </div>
    <div class="input-box">
      <input name="password" type="password" minlength="10" maxlength="15" required>
      <label>Пароль</label>
    </div>
    <button type="submit" class="btn">Регистрация</button>
    <div class="login-register">
      <p>У вас есть аккаунт? <a href="#" class="login-link">Войдите</a></p>
      <p class="error_registr">
        <?php
          if (isset($_SESSION["error_msg_reg"])) {
            echo $_SESSION["error_msg_reg"];
          }
        ?>
      </p>
    </div>
  </form>
</div>
</div>
</div>
<div class="section__main">
  <div class="main-page__section">
    <div class="main-container">
      <?php
        if (isset($_GET['type'])) {
          if (isset($_GET['id'])) {
            //Вывод всей инфы новости то есть текст + фото)))

```

```

        printOnlyOneNew($dbconn, (int)$_GET['id']);
    }
    else{
        echo printAllNews($dbconn, false, $_GET['type']);
    }
}
else{
    echo printAllNews($dbconn, false);
}
?>
</div>
<div class="sidebar">
    
    
    
</div>
</div>

<div class="section__footer">
    <footer class="footer">
        <div class="footer__info">
            <p>Контактный телефон: 8(999)-999-99-99</p>
            <a href="adminLog.php">админ</a>
            <p>Почта: news@gmail.com</p>
            <!-- Ссылка на переход к входу админа -->
        </div>
    </footer>
</div>
</div>
<?php
if (isset($_SESSION["error_msg_reg"])) {
    echo '<script src="script/registr_error.js"></script>';
}
unset($_SESSION["error_msg_reg"]);
?>
<?php
if (isset($_SESSION["error_msg_log"])) {
    echo '<script src="script/log_error.js"></script>';
}
unset($_SESSION["error_msg_log"]);
?>
<script src="script/login-registration.js"></script>
</body>

</html>

```

#### instructAd.php

```

<?php
require_once "connectBD.php";

//создание списка новостей еще не проверенных
function printNewsAdmin($db){
    $query = "SELECT a.id, b.type, c.login FROM news as a JOIN type_news as b ON
        a.id_type_new = b.id JOIN users as c ON
        c.id = a.id_author WHERE id_confirm=3 ORDER BY id ASC";
    $res = pg_query($db, $query);
    $cnt_news = pg_num_rows($res);
    if ($cnt_news > 0){
        echo "<ul>";
        while ($row = pg_fetch_array($res, NULL, PGSQL_ASSOC)){
            printNewAdmin($i, $row);
        }
        echo "</u>";
    }
}
}

```

```

//создание элемента списка с ссылкой get-запроса, чтобы при нажатии выводил на экран эту новость
function printNewAdmin(&$i, &$new){
    $id = $new["id"];
    $type = $new["type"];
    $login = $new["login"];
    echo "<li><a href='admin.php?type=$type&id=$id&ath=$login'>Новость $type</a></li>";
}

//вывод отдельной новости, если пользователь нажал на нее
function printOnlyOneNew($db, $id){
    $query = "SELECT image, title, data, text_file FROM news WHERE id = '$id'";
    $res = pg_query($db, $query);
    $new = pg_fetch_array($res);
    if ($new != false){
        //изображение
        if (($new['image'] != null)){
            echo "<img class='image' src='image/' . $new['image'] . \"' alt='альтернативный текст' width='700'
height='400'>";
        }
        $file = file_get_contents("filestxt/" . $new['text_file']);
        echo "<p class='new_header'><b>" . $new['title'] . "</b></p><br>";
        echo "<p class='new_page'>" . htmlspecialchars($file) . "</p><br>";
    }
}

//показ новости и ее дополнительные поля
function printNewShow($db, $id, $type, $ath){
    if (isset($id))
    {
        echo "<div class='full_new'>";
        printOnlyOneNew($db, $id);
        echo "
</div>
<div class='info'>
    <div>
        <span>Тип: </span>
        <span>
            $type
        </span>
    </div>
    <div>
        <span>Автор: </span>
        <span>
            $ath
        </span>
    </div>
</div>";
    }
}
?>

```

#### login.php

```

<?php
session_start();
require_once "connectBD.php";

if (isset($_SESSION["user"])){
    header("Location: profile.php");
}

if (!isset($_POST["email"])){
    header("Location: index.php");
}
else{
    $email = htmlspecialchars($_POST["email"]);
}

```

```

$password = htmlspecialchars($_POST["password"]);
$query = "SELECT email, password FROM users WHERE email = '$email'";
$check = pg_query($dbconn, $query);
$check_log = pg_fetch_row($check);
if ($check_log[0] == $email && password_verify($password, $check_log[1])){
    $_SESSION["user"] = [
        "login" => $password,
        "email" => $email
    ];
    header("Location: profile.php");
}
else{
    $_SESSION["error_msg_log"] = "Неверный логин или пароль";
    header("Location: index.php");
}
}
}

```

?>

### profile.php

```

<?php
    session_start();
    if (!isset($_SESSION["user"])){
        header("Location: index.php");
    }
    require_once "selectBD.php";
?>
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="css/header.css">
    <link rel="stylesheet" href="css/config.css">
    <link rel="stylesheet" href="css/main.css">
    <link rel="stylesheet" href="css/footer.css">
    <title>Новости</title>
</head>

<body>
    <div class="layout">
        <div class="section__header">
            <header class="header">
                <h2 class="header__logo"><a href="profile.php">Информашки</a></h2>
                <nav class="header__navigation">
                    <a class="navig" href="profile.php?type=Европа">Европа</a>
                    <a class="navig" href="profile.php?type=Россия">Россия</a>
                    <a class="navig" href="profile.php?type=Спорт">Спорт</a>
                    <a class="navig" href="profile.php?type=Культура">Культура</a>
                    <a class="navig" href="profile.php?type=Образование">Образование</a>
                    <button class="header__btn-login btn-publish">Публикация</button>
                    <button class="header__btn-login"><a href="exit.php">Выход</a></button>
                </nav>
            </header>

            <div class="wrapper publish">
                <span class="icon-close"><button></button></span>
                <div class="form-box">
                    <h2>Публикация</h2>
                    <form action="publish.php" enctype="multipart/form-data" method="POST">
                        <div class="input-box">
                            <select class="select-css" name="type_new">
                                <option value="Россия" selected>Россия</option>
                                <option value="Европа">Европа</option>
                                <option value="Образование">Образование</option>

```

```

        <option value="Культура">Культура</option>
        <option value="Спорт">Спорт</option>
    </select>
    <label>Тип новости</label>
</div>
<div class="input-box">
    <input name="header"
        type="text" minlength="10" maxlength="100"
        required>
    <label>Заголовок</label>
</div>
<div class="input-box">
    <input class="file_style" name="new" type="file" accept=".txt" required>
    <label>Файл публикации (.txt)</label>
</div>
<div class="input-box">
    <input class="file_style" name="photo" type="file" accept="image/png, image/jpeg">
    <label>Картинка (.png, .jpeg)</label>
</div>
<button type="submit" class="btn">Опубликовать</button>
<p class="error_publish">
    <?php
        if (isset($_SESSION["error_msg_publish"])) {
            echo $_SESSION["error_msg_publish"];
        }
    ?>
</p>
</form>
</div>
</div>
</div>

<div class="section__main">
    <div class="main-page__section">
        <div class="main-container">
            <?php
                if (isset($_GET['type'])) {
                    if (isset($_GET['id'])) {
                        //Вывод всей инфы новости то есть текст + фото)))
                        printOnlyOneNew($dbconn, (int)$_GET['id']);
                    }
                    else {
                        echo printAllNews($dbconn, true, $_GET['type']);
                    }
                }
                else {
                    echo printAllNews($dbconn, true);
                }
            ?>
        </div>
        <div class="sidebar">
            
            
            
        </div>
    </div>
</div>

<div class="section__footer">
    <footer class="footer">
        <div class="footer__info">
            <p>Контактный телефон: 8(999)-999-99-99</p>
            <p>Почта: news@gmail.com</p>
        </div>
    </footer>
</div>

```

```

</div>
<?php
    if (isset($_SESSION["error_msg_publish"])) {
        echo '<script src="script/publish_error.js"></script>';
    }
    unset($_SESSION["error_msg_publish"]);
?>
<script src="script/publish.js"></script>
</body>

</html>

```

## publish.php

```

<?php
    session_start();
    require_once "connectBD.php";

    //для корректного названия текстового файла на сервере
    function printType($type){
        switch ($type) {
            case "Россия":
                return "russia";
                break;
            case "Культура":
                return "culture";
                break;
            case "Образование":
                return "education";
                break;
            case "Спорт":
                return "sport";
                break;
            case "Европа":
                return "europa";
                break;
        }
    }

    //проверка на авторизованного пользователя
    if (isset($_SESSION["user"])){
        //проверка на удачную загрузку файла
        if (isset($_FILES["new"]) && $_FILES["new"]["type"] == "text/plain" && $_FILES["new"]["error"] ==
        UPLOAD_ERR_OK){

            $header = $_POST["header"];
            $type = $_POST["type_new"];
            $photo = null;
            $en_type = printType($type);
            $email = $_SESSION["user"]["email"];
            $uploaddir = '/Users/evgeniy/localhost/kr2023/';

            //проверка на удачную загрузку фото
            if (isset($_FILES["photo"]) && ($_FILES["photo"]["type"] == "image/png" || $_FILES["photo"]["type"] ==
            "image/jpeg")
                && $_FILES["photo"]["error"] == UPLOAD_ERR_OK)
            {
                //махинации с названием файла и его созданием на сервере
                $type_photo = mb_substr($_FILES["photo"]["type"], 6);
                $photo = $en_type . time() . "." . $type_photo;
                $uploadfile = $uploaddir . "image/" . $photo;
                move_uploaded_file($_FILES["photo"]["tmp_name"], $uploadfile);
            }

            //махинации с названием файла и его созданием на сервере
            $new = $en_type . time() . ".txt";
            $uploadfile = $uploaddir . "filestxt/" . $new;

```

```

move_uploaded_file($_FILES["new"]["tmp_name"], $uploadfile);

$query = "SELECT id FROM type_news WHERE type = '$type'";
$type_id = pg_query($dbconn, $query);
$type_id = pg_fetch_array($type_id);
$type_id = $type_id['id'];
$query = "SELECT id FROM users WHERE email = '$email'";
$id = pg_query($dbconn, $query);
$id = pg_fetch_array($id);
$id = $id['id'];

$query = "INSERT INTO news (id_author, id_type_new, image, title, text_file) VALUES('$id', '$type_id',
'$photo', '$header', '$new')";
pg_query($dbconn, $query);

}
else{
    $_SESSION["error_msg_publish"] = "Данные не приняты";
}
header("Location: profile.php");
}
else{
    header("Location: index.php");
}

?>

```

#### registration.php

```

<?php
session_start();
require_once "connectBD.php";

if (isset($_SESSION["user"])){
    header("Location: profile.php");
}
if (!isset($_POST["login"])){
    header("Location: index.php");
}
else{
    //проверка почты на авторизованность
    $email = htmlspecialchars($_POST["email"]);
    $query = "SELECT email FROM users WHERE email = '$email'";
    $check = pg_query($dbconn, $query);
    $check_email = pg_fetch_row($check);
    //если уже есть почта, то соответственно говорим это пользователю
    if ($check_email){
        $_SESSION["error_msg_reg"] = "Под этой почтой уже авторизовались";
        header("Location: index.php");
    }
    else{
        //все ок, заносим данные в БД и переадресовываем его на его личную страничку
        $login = htmlspecialchars($_POST["login"]);
        $password = password_hash(htmlspecialchars($_POST["password"]), PASSWORD_DEFAULT);
        $query = "INSERT INTO users (login, email, password) VALUES('$login', '$email', '$password')";
        pg_query($dbconn, $query);
        //создаем информацию о пользователе для блокировки
        //для блокировки переходов между страницами
        $_SESSION["user"] = [
            "login" => $login,
            "email" => $email
        ];
        header("Location: profile.php");
    }
}

?>

```

## selectBD.php

```
<?php
require_once "connectBD.php";
//Вывод мини-новости на экран
function printStrNew(&$news, $isuser, $id, $cnt_news){
    $data = "";
    $title = "";
    if ($id < $cnt_news){
        $type = $news[$id]['type'];
        $data = $news[$id]['data'];
        $title = $news[$id]['title'];
        $id = $news[$id]['id'];
        //здесь прописываем ссылку при нажатие(неавторизированный/авторизированный пользователь)
        if (!$isuser){
            $id = "index.php?type=$type&id=$id";
        }
        else{
            $id = "profile.php?type=$type&id=$id";
        }
    }
    else{
        $id = "#";
    }
    return
        "<div class='card mini'>
        <a href=
        '$id'
        class='card-mini'>
            <div class='card-mini__title'>
                <span>
                    "
                    $title
                    "
                </span>
            </div>
            <div class='time_new'>
                "
                $data
                "
            </div>
        </a>
        </div>";
    }

//Вывод на экран строки из 3 новостей
function printStrRowNews(&$news, $isuser, $ids, $cnt_news){
    $start = "<div class='topnews_row'>
        <div class='cards-mini'> ";
    $end = " </div></div>";

    for ($i = 0; $i < 3; $i++){
        $start .= printStrNew($news, $isuser, $ids, $cnt_news);
        $ids++;
    }
    return $start . $end;
}

//Вывод всех новостей на экран
function printAllNews(&$db, $isuser, $type=NULL){
    //в зависимости нажал ли пользователь на отдельный тип новостей(get запрос)
    if (isset($type)){
        $query = "SELECT a.id, image, title, data, type
        FROM news a JOIN type_news b ON a.id_type_new = b.id
        WHERE type = '$type' AND id_confirm = 1 ORDER BY data DESC";
```



```

    }
    else{
        $query = "SELECT a.id, image, title, data, type
        FROM news a JOIN type_news b ON a.id_type_new = b.id
        WHERE id_confirm = 1 ORDER BY data DESC";
    }
    $res = pg_query($db, $query);
    //возвращает все найденные строки
    $news = pg_fetch_all($res);

    $len = count($news);
    $current = 0;
    $rows = ceil($len / 3);
    $result = "";
    for ($i = 0; $i < $rows; $i++){
        $result .= printStrRowNews($news, $isuser, $current, $len);
        $current += 3;
    }
    return $result;
}

//вывод отдельной новости,если пользователь нажал на нее
function printOnlyOneNew($db, $id){
    $query = "SELECT image, title, data, text_file FROM news WHERE id = '$id'";
    $res = pg_query($db, $query);
    $new = pg_fetch_array($res); //получаем массив строк
    if ($new != false){
        //проверяем есть ли изображение
        if (($new['image'] != null)){
            echo "<img class='image' src='image/' . $new['image'] . \"' alt='альтернативный текст' width='700'
height='400'>";
        }
        $file = file_get_contents("filestxt/" . $new['text_file']);
        echo "<p class='new_header'><b>\" . $new['title'] . \"</b></p><br>";
        echo "<p class='new_page'>\" . htmlspecialchars($file) . '</p><br>";
    }
}
?>

```

#### updateNew.php

```

<?php
session_start();
require_once "connectBD.php";

function removeNew($db, $confirm, $id){
    $query = "SELECT text_file, image FROM news WHERE id='$id'";
    $res = pg_query($db, $query);
    $res = pg_fetch_array($res);
    $textfile = $res["text_file"];
    $image = $res["image"];
    $uploadDirText = '/Users/evgeniy/localhost/kr2023/filestxt/';
    $uploadDirImage = '/Users/evgeniy/localhost/kr2023/image/';
    unlink($uploadDirText . $textfile);
    if (isset($image)){
        unlink($uploadDirImage . $image);
    }
    $query = "UPDATE users as a SET rating=rating - 1 WHERE a.id=
(SELECT id_author FROM news as b WHERE b.id='$id')";
    $res1 = pg_query($db, $query);
    $query = "UPDATE news SET id_confirm='$confirm' WHERE id='$id'";
    $res = pg_query($db, $query);
}

function successNew($db, $confirm, $id){
    $query = "UPDATE news SET id_confirm='$confirm' WHERE id='$id'";
    $res = pg_query($db, $query);
}

```

```

$query = "UPDATE users as a SET raiting=raiting + 1 WHERE a.id=
(SELECT id_author FROM news as b WHERE b.id='$id')";
$res1 = pg_query($db, $query);
}

if (isset($_SESSION["admin"])){
    $id = $_GET["id"];
    $conf = $_GET["conf"];
    if (isset($id) && is_numeric($id) && isset($conf) && is_numeric($conf)){
        $conf = (int)$conf;
        $id = (int)$id;
        $query = "SELECT id FROM news WHERE id='$id'";
        $res = pg_query($dbconn, $query);
        $res = pg_fetch_array($res); //получаем массив строк
        if ($res != false){
            if ($conf == 1){
                successNew($dbconn, $conf, $id);
            }
            else{
                if ($conf == 2){
                    removeNew($dbconn, $conf, $id);
                }
            }
        }
    }
}
header("Location: admin.php");
}
else{
    if (isset($_SESSION["user"]))
    {
        header("Location: profile.php");
    }
    else{
        header("Location: index.php");
    }
}
}
?>

```

## Файлы js

### log\_error.js

```

function error_login(){
    const wrapper2= document.querySelector('.wrapper');
    wrapper2.classList.add('active-btn');
}

var isRealiseLog = true
if (isRealiseLog)
{
    error_login();
    isRealiseLog = false
}

```

### login\_registration.js

```

const wrapper= document.querySelector('.wrapper');
const loginLink = document.querySelector('.login-link');
const registerLink = document.querySelector('.register-link');
const btnLogin = document.querySelector('.header__btn-login');
const btnClose = document.querySelector('.icon-close');
const p_error_registr = document.querySelector('.error_registr');
const p_error_log = document.querySelector('.error_log');

```

```

registerLink.addEventListener("click", ()=>{
  wrapper.classList.add('active');
  p_error_log.classList.add('invisible')
});

```

```

loginLink.addEventListener("click", ()=>
{
  wrapper.classList.remove('active');
  p_error_registr.classList.add('invisible')
});

```

```

btnLogin.addEventListener("click", ()=>
{
  wrapper.classList.add('active-btn');
});

```

```

btnClose.addEventListener("click", ()=>
{
  wrapper.classList.remove('active-btn');
  p_error_registr.classList.add('invisible')
  p_error_log.classList.add('invisible')
});

```

#### **publish\_error.js**

```

function error_publish(){
  const wrapper2= document.querySelector('.wrapper');
  wrapper2.classList.add('active-btn');
}

```

```

var isRealisePublish = true
if (isRealisePublish)
{
  error_publish();
  isRealisePublish = false
}

```

#### **publish.js**

```

const wrapper= document.querySelector('.wrapper');
const btnPublish = document.querySelector('.btn-publish');
const btnClose = document.querySelector('.icon-close');
const p_error_publish = document.querySelector('.error_publish');

```

```

btnPublish.addEventListener("click", ()=>
{
  wrapper.classList.add('active-btn');
});

```

```

btnClose.addEventListener("click", ()=>
{
  wrapper.classList.remove('active-btn');
  p_error_publish.classList.add('invisible')
});

```

#### **registr\_error.js**

```

function error_registr(){
  const wrapper1= document.querySelector('.wrapper');
  wrapper1.classList.add('active-btn');
  wrapper1.classList.add('active');
}

```

```

var isRealise = true
if (isRealise)
{
  error_registr();
}

```

```
    isRealise = false  
}
```