

ГУАП

КАФЕДРА № 14

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ
ПРЕПОДАВАТЕЛЬ

старший преподаватель
должность, уч. степень, звание

подпись, дата

О. М. Шарапова
инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ

«Создание сайта с БД (клиентская часть)»

по курсу: ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННЫХ СИСТЕМ С
ПРИМЕНЕНИЕМ WEB-ТЕХНОЛОГИЙ

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. № 1042

подпись, дата

Н.В.Корзун
инициалы, фамилия

Санкт-Петербург 2024

1. Постановка задачи

Создание сайта, содержащего следующие структуры:

- a. Форма авторизации (регистрации) на страницах должна появляться при нажатии на кнопку или ссылку;
- b. Фильтрация должна осуществляться по выбору через ЭУ (выпадающий список, список, переключатели, флажки);
- c. Реализация корзины М:М;
- d. Личный кабинет. Редактирование информации. Просмотр заказов, удаление старых заказов;
- e. Добавить сервисную страницу: ГАЛЕРЕЯ
- f. Создание страницы КОНТАКТЫ с обратной связью (применением почтового сервера).

2. Описание предметной области

Ветеринарная клиника ориентирована на один вид питомцев - кошек— пользователи могут зарегистрировать своего питомца(создать карточку питомца), также они могут записать питомца на доступное время. Ветеринары(их рабочее время назначает админ – 3 роль для сущности пользователи) способны обрабатывать полученные заявки и писать характеристики питомцам.

В то время как не владельцев необходимо вручную заносить в систему (что обосновывается здравым смыслом), владельцы питомцев могут зарегистрироваться в системе через форму регистрации.

Сущности предметной области необходимо вручную добавлять через СУБД, хотя для некоторых добавлений созданы страницы, но они не протестированы

3. Схема БД

Схема БД представлена на рисунке 1.

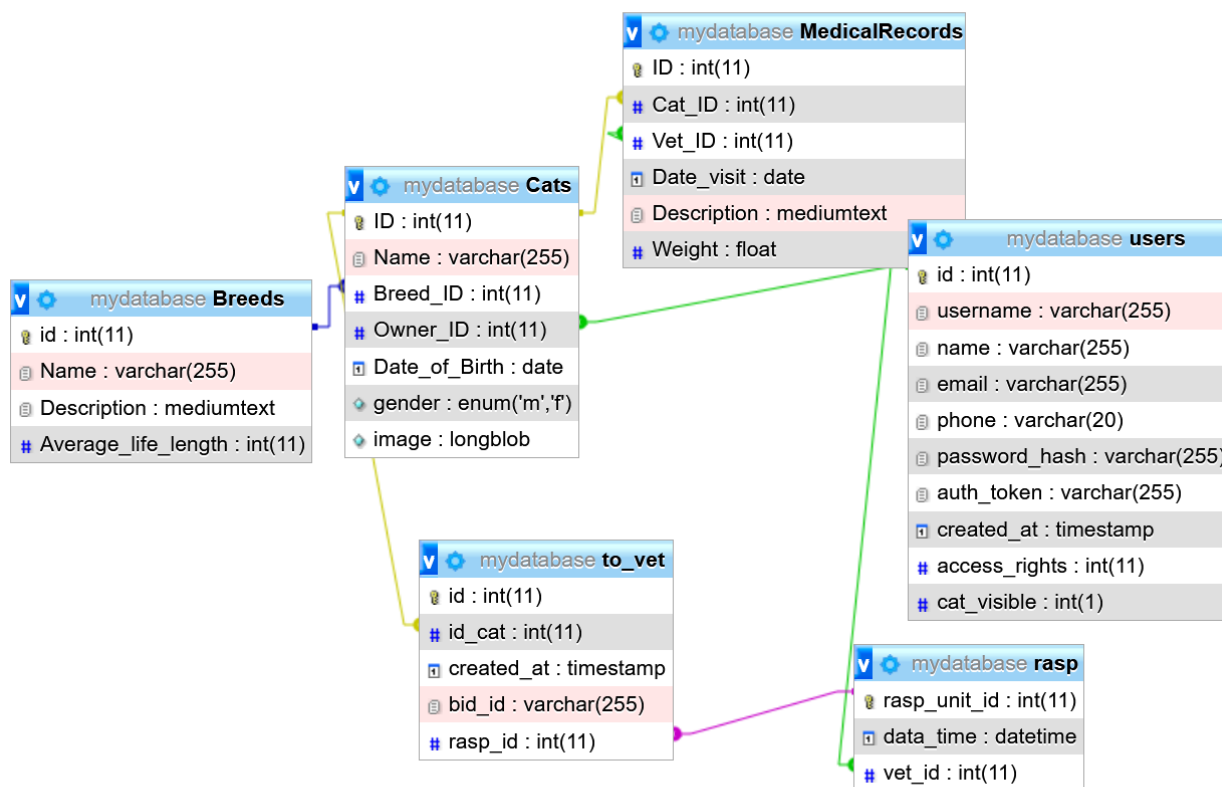


Рисунок 1 – Схема БД

4. Структура сайта

Основной целью выполнения лабораторных работ было не просто создание сайта, а получение навыков web-разработки: языков PHP, JS, HTML, CSS. Исходя из этого структура сайта была выбрана как одностраничный сайт, с асинхронной подгрузкой данных(страниц). Этот подход, очевидно, не самый лучший, но он был выбран для усложнения разработки и более подробного изучения возможностей стека web-технологий.

Сайт состоит из следующих страниц:

- Информационная (рисунок 2)
- Страница профиля пользователя (рисунки 3-4)
- Страница регистрации (рисунок 5)
- Страница логина (рисунок 6)
- Страница успеха регистрации (рисунок 7)
- Страница списка питомцев (рисунок 8-9)
- Страница добавления питомцев (рисунок 10)
- Страница записи к врачу (рисунок 11)
- Страница детальной информации о заявках (рисунок 12)
- Галерея питомцев, которых разрешили демонстрировать владельцы (рисунок 13)
- Страница КОНТАКТЫ (рисунок 14)

Наверху каждой страницы имеется навигационная панель, с помощью которой происходит основная навигация по сайту. Кроме того, при клике на логотип сайта, пользователя перенаправит на информационную страницу.

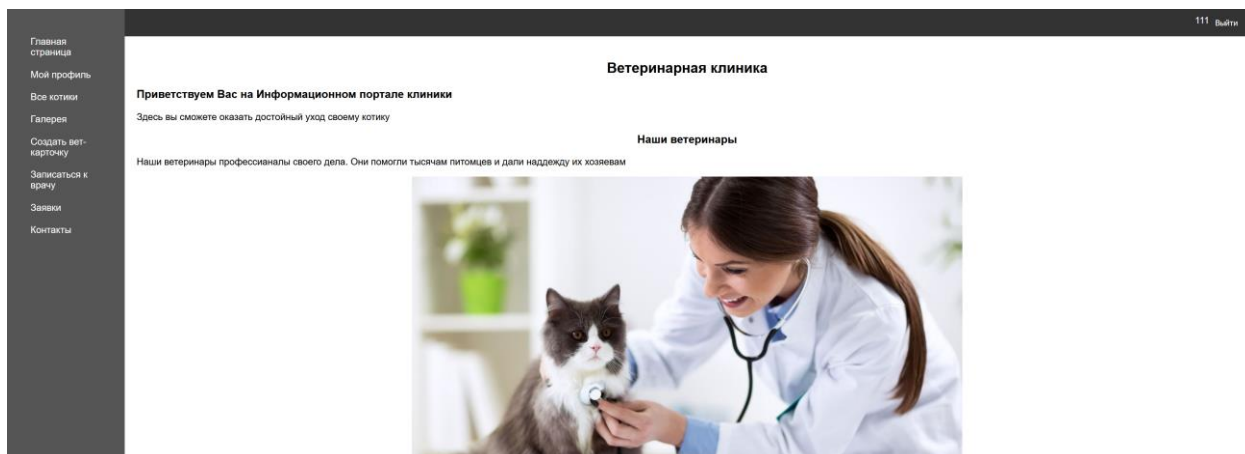


Рисунок 2 – Информационная страница сайта

Профиль пользователя

ID: 1

Имя пользователя:

Имя:

Email:

Телефон:

Разрешить отображение ваших питомцев: ☒

Права доступа: Пользователь

Рисунок 3 – Профиль пользователя

Профиль пользователя

ID: 3

Имя пользователя:

Имя:

Email:

Телефон:

Разрешить отображение ваших питомцев: ☐

Права доступа: Врач

Рисунок 4 – Профиль ветеринара

Имя пользователя:

Email:

Пароль:




Рисунок 5 – Регистрация.

Рисунок 6 – Логин

111 Выйти

Рисунок 7 – Информация об успешной авторизации

Вася Пупкин - владелец




Кличка	Порода	Дата рождения	Дата визита к врачу	Описание	Вес	Image
Сол	Британская короткошерстная кошка	2011-11-30				
Мистер Сноу	Сиамская кошка	2018-04-15	2023-01-01	Первичный осмотр	5.5	
Белла	Рэгдолл	2020-06-18				
Кайф	Северная кошка	2024-03-05				
Вано	Сфинкс	2024-03-06	2024-03-05	прием у онколога	0.5	

Рисунок 8 – Форма сортировки питомцев

Вася Пупкин - владелец

Кличка	Порода	Дата рождения	Дата визита к врачу	Описание	Вес	Image
Белла	Рэгдолл	2020-06-18				
Вано	Сфинкс	2024-03-06	2024-03-05	прием у онколога	0.5	
Кайф	Северная кошка	2024-03-05				
Мистер Сноу	Сиамская кошка	2018-04-15	2023-01-01	Первичный осмотр	5.5	
Сол	Британская короткошерстная кошка	2011-11-30				

Рисунок 9 – Список питомцев с фильтрацией по кличке

Создать ветеринарную карточку для питомца

Кличка:

Порода:

Сиамская кошка

Дата рождения:

mm / dd / уууу

Пол:

Мужской

Изображение:

Browse...

 No file selected.

Сохранить

Рисунок 10 – страница заполнения вет-карточки

Вася Пупкин - питомцы

Кличка	Порода	Дата рождения	Изображение	Действие
Мистер Сноу	Сиамская кошка	2018-04-15		Записать кота к врачу
Белла	Рэгдолл	2020-06-18		Записать кота к врачу
Сол	Британская короткошерстная кошка	2011-11-30		Записать кота к врачу
Вано	Сфинкс	2024-03-06		Записать кота к врачу
Кайф	Северная кошка	2024-03-05		Записать кота к врачу

Заявка

Белла записан

Сол записан

Кайф записан

Оставить заявку

Рисунок 11 – страница записи к врачу

Заявки

Идентификатор заявки: d0c2670d626b5a57c7cee1b7780e44916dabd713

Вы записаны на: 2024-04-18 11:00:00

Кличка

Белла

Сол

Кайф

Идентификатор заявки: 26ae782003fa0604750e00e0f49088694538b8b9

Кличка

Белла

Выберите дату:

2024-04-18

 Выберите время:

10:00:00

10:00:00

12:00:00

13:00:00

14:00:00

15:00:00

16:00:00

17:00:00

Рисунок 12 – страница выбора времени и просмотра заявок

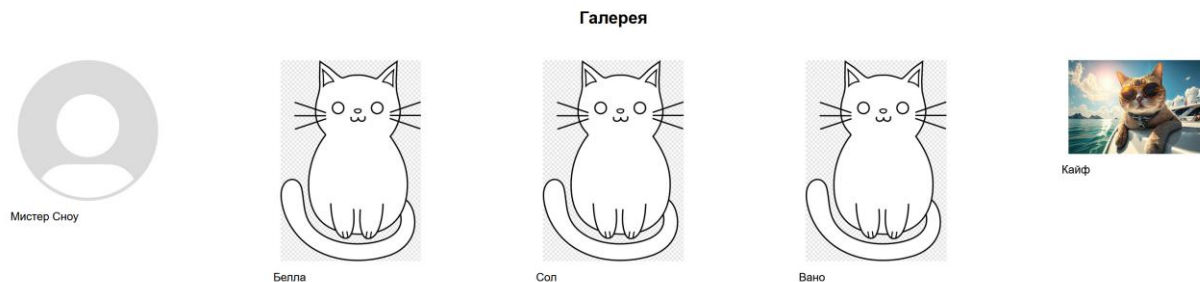


Рисунок 13 – страница-галерея

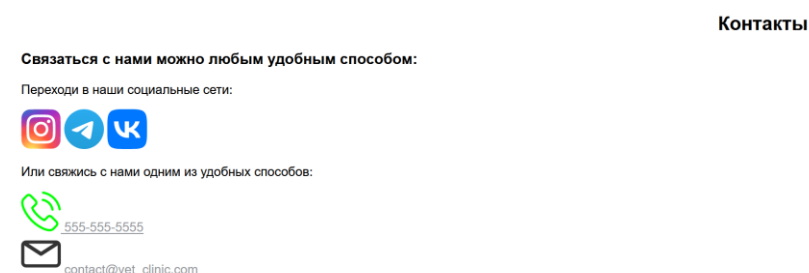


Рисунок 14 – Страница КОНТАКТЫ

5. Инструкция для пользователей

5.1 Инструкция для Гостей

Гость при первом посещении сайта попадает на информационную страницу. Гость может просматривать галерею и контакты. Гость может посетить соответствующие страницы посредством навигационной панели. На любой из страниц Гость может перейти на страницу регистрации/регистрации (при переходе на интересующую страницу на навигационной панели). После авторизации Гость становится пользователем или другими ролями.

Также Гость вправе связаться (переход на страницу КОНТАКТЫ).

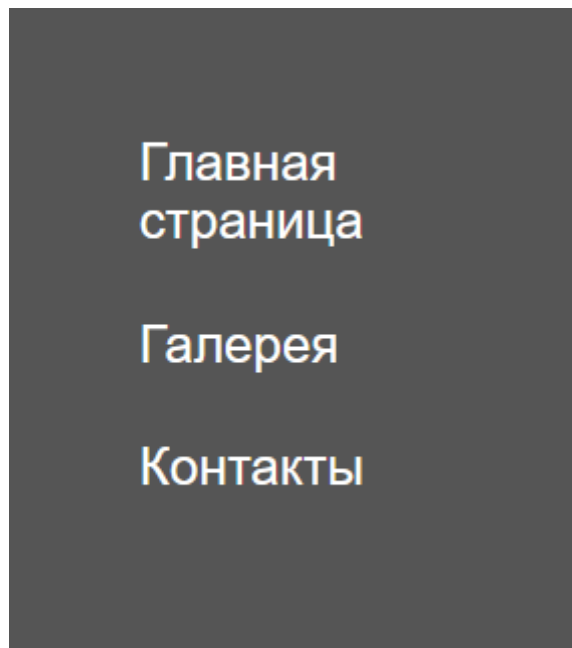


Рисунок 15 – Навигационная панель гостя

5.2 Инструкция для пользователей

После авторизации пользователей попадёт на информационную страницу. Оттуда он может перейти на страницу профиля или другие функциональные страницы.

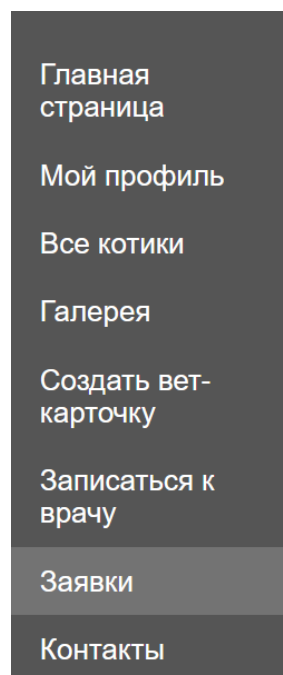


Рисунок 16 – Навигационная панель пользователей

На различных страницах представленных на рисунке 16, пользователь может выполнить указанные действия, такие как Занесение записей о новых питомцах(создание вет-карточек), Запись к врачу, просмотр и редактирование пользовательской информации.

5.3 Инструкция для Ветеринары

Пользовательский опыт ветеринара во многом похож на опыт Гостя. Единственное значимое отличие – наличие страницы заявки. При переходе на страницу заявки, ветеринар способен обрабатывать заявки.

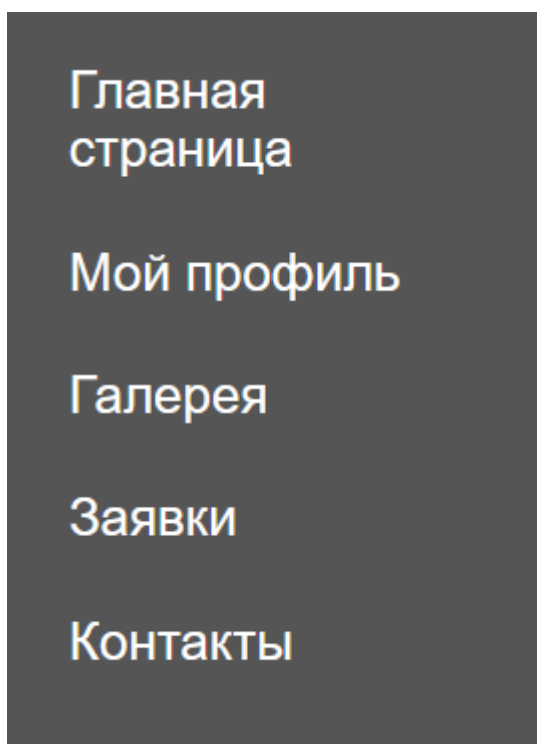


Рисунок 17 – Навигационная панель Ветеринара

Библиотекарь может просматривать заявки и создавать записи о питомцах.

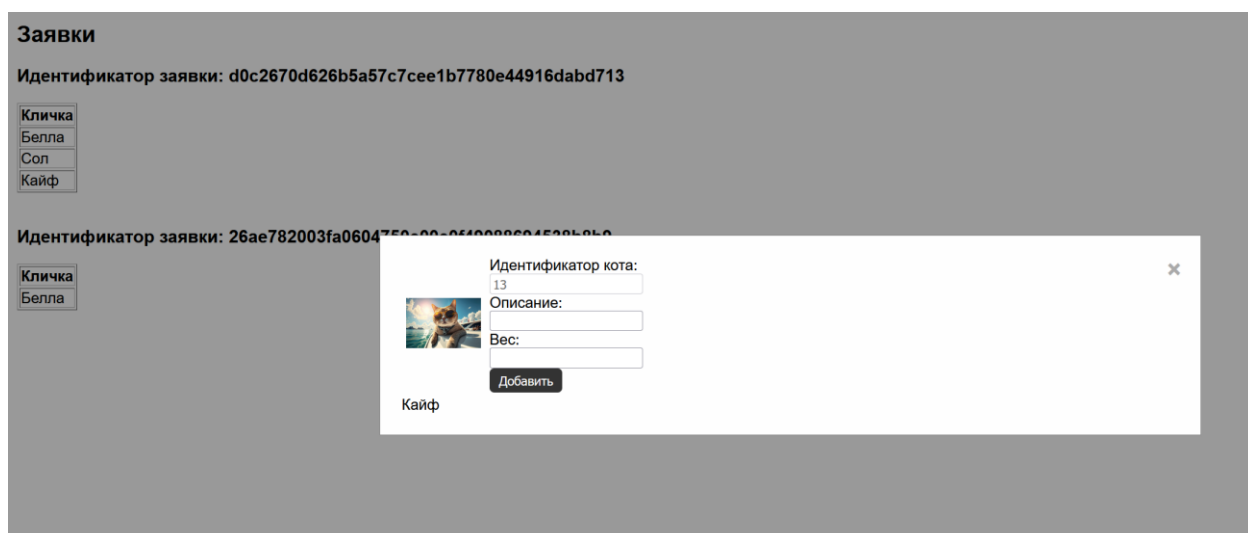
Скриншот интерфейса «Заявки». Вверху заголовок «Заявки». Ниже «Идентификатор заявки: d0c2670d626b5a57c7cee1b7780e44916dabd713». Далее список «Кличка» с элементами «Белла», «Сол», «Кайф». Ниже «Идентификатор заявки: 26ae782003fa0604750-00-06400000045000000». Далее список «Кличка» с элементом «Белла». В центре экрана открыто модальное окно с заголовком «Идентификатор кота: 13». В окне есть поля «Описание:» и «Вес:», кнопка «Добавить» и изображение котика. Внизу модального окна написано «Кайф».

Рисунок 18 – форма ввода информации

После нажатия на «ПОДТВЕРДИТЬ» будет отправлен запрос на сервер и внесены необходимые изменения в БД, а страница – перезагрузится. В изменения в БД входит:

5.4 Инструкция для Администратора

Роль администратора позволяет вносить рабочие часы для ветеринаров, просматривать общую информацию. Его функционал выполнен достаточно посредственно, т.к. предполагается что его задачи должны быть сформулированы более точно.

6. Программный код

Программный код предоставлен в Приложении А-В.

7. Выводы

В рамках выполнения лабораторной работы были изучены методы проектирования и реализации Веб-Сайтов. Изучена СУБД MySQL, получены базовые знания о веб-технологиях, используемых при создании сайта. Получен опыт в рамках основ вёрстки, изучена теория о построении user-friendly пользовательских интерфейсов.

Приложение А

Файл to_vet.php (интерфейс корзины)

```
<?php

if ($_SERVER['REQUEST_METHOD'] === 'POST') {

    $authToken = $_COOKIE['auth_token'] ?? null;

    if ($authToken) {
        include '../db_connect.php';

        $get_username = "SELECT * FROM users WHERE auth_token = :authToken";
        $stmt = $conn->prepare($get_username);
        $stmt->bindParam(':authToken', $authToken);
        $stmt->execute();

        $result = $stmt->fetch(PDO::FETCH_ASSOC);

        if ($result) {
            if ($result["access_rights"] == 2) {
                $sql = "SELECT Cats.ID as CatID, Cats.Name as CatName,
Breeds.Name as BreedName, Cats.Date_of_Birth, Cats.image FROM Cats JOIN Breeds ON
Cats.Breed_ID = Breeds.ID WHERE Owner_ID = :userId";
                $stmt = $conn->prepare($sql);
                $stmt->bindParam(':userId', $result['id']);
                $stmt->execute();
                $res = $stmt->fetchAll(PDO::FETCH_ASSOC);

                echo "
<style>
#bid {
    position: fixed;
    right: 0;
    top: 50px;
    width: 300px;
    border-left: 1px solid #ccc;
    padding: 10px;
    background-color: #f9f9f9;
}
```



```

        <td class='no-border' onclick='to_vet(\"\".
$row['CatID'].":".$row['CatName'].\"\\")'>Записать кота к врачу</td>
        </tr>";
    }
    echo "</table>";
}
else {
    echo "У вас пока нет котиков";
}
}

else {
    echo "Не достаточно прав на просмотр этой страницы";
}
} else {
    echo "Не действительный auth token";
}
}
} else {
    echo "Вы не вошли";
}
} else {
    // Если запрос не является POST-запросом, отправляем ошибку
    http_response_code(405);
    echo json_encode(['error' => 'Метод не поддерживается.']);
}
}

```

Файл tovet.js (добавление в корзину)

```

function sortTable(n)
{
    var table, rows, switching, i, x, y, shouldSwitch, dir, switchcount = 0;
    table = document.getElementById("to_vet-table-id");
    switching = true;
    dir = "asc";

    while (switching) {
        switching = false;
        rows = table.rows;
        for (i = 1; i < (rows.length - 1); i++) {
            shouldSwitch = false;
            x = rows[i].getElementsByTagName("TD")[n];
            y = rows[i + 1].getElementsByTagName("TD")[n];
            if (dir == "asc") {
                if (x.innerHTML.toLowerCase() > y.innerHTML.toLowerCase()) {
                    shouldSwitch = true;
                    break;
                }
            } else if (dir == "desc") {
                if (x.innerHTML.toLowerCase() < y.innerHTML.toLowerCase()) {

```

```

        shouldSwitch = true;
        break;
    }
}
}
if (shouldSwitch) {
    rows[i].parentNode.insertBefore(rows[i + 1], rows[i]);
    switching = true;
    switchcount ++;
} else {
    if (switchcount == 0 && dir == "asc") {
        dir = "desc";
        switching = true;
    }
}
}
}

function send_bid()
{
    let data = {};
    for (let key in sessionStorage) {
        if (sessionStorage.hasOwnProperty(key)) {
            data[key.split(":")[0]] = sessionStorage.getItem(key);
        }
    }

    // Преобразуем объект в JSON
    let jsonData = JSON.stringify(data);

    const xhr = new XMLHttpRequest();
    xhr.open('POST', "/send_bid", true);
    // Установка заголовка Content-Type для отправки JSON
    xhr.setRequestHeader('Content-Type', 'application/json;charset=UTF-8');
    xhr.onload = function() {
        if (this.status >= 200 && this.status < 400)
        {
            let response = JSON.parse(this.response);
            if (response.status === "success")
            {
                sessionStorage.clear();
                let bidItems = document.getElementById('bidItems');

                bidItems.innerHTML = "Заявка отправлена. Уточните удобное время в
заявках";
            }
        }
    }
    else

```

```

        {
            console.error('Ошибка при загрузке страницы');
        }
    };
    xhr.onerror = function() {
        console.error('Ошибка при загрузке страницы');
    };
    // Отправка данных в теле запроса
    xhr.send(jsonData);
}

```

```

function to_vet(cat_id)
{
    //console.log(cat_id);
    let el = sessionStorage.getItem(cat_id);
    if(el)
    {
        let vvv= Number(el)+1;
        if(vvv > 1)
            vvv = 1;
        sessionStorage.setItem(cat_id, vvv);
    }
    else
        sessionStorage.setItem(cat_id, 1);

    bid_draw();
}

```

```

function decrease_cat_bids(key)
{
    let el = sessionStorage.getItem(key);
    if(el == 1)

        sessionStorage.removeItem(key);
    else
        sessionStorage.setItem(key, Number(el)-1);

    bid_draw();
}

```



```

function bid_draw()
{
    let bidItems = document.getElementById('bidItems');
    bidItems.innerHTML = "";
    for (let key in sessionStorage)
    {
        // Проверяем, что ключ действительно принадлежит sessionStorage,
        // так как в JavaScript ключи объекта могут быть перечислены из прототипа
        if (sessionStorage.hasOwnProperty(key))
        {
            // Получаем значение по ключу
            let value = sessionStorage.getItem(key);

            const listItem = document.createElement('li');

            listItem.onclick = () => {decrease_cat_bids(key)};
            let name = key.split(":")[1];
            listItem.textContent = `${name} записан`;
            bidItems.appendChild(listItem);

            //console.log(key + ": " + value);
        }
    }
}

if (document.getElementById('bidItems')) {
    // Если элемент уже присутствует, запускаем функцию
    //console.log(111);
    bid_draw();
} else {
    // Если элемент еще не появился, начинаем наблюдение за изменениями в DOM
    let observer = new MutationObserver(function(mutations) {
        mutations.forEach(function(mutation) {
            if (mutation.type === 'childList') {
                let el = document.getElementById('bidItems');
                if (el) {
                    // Если элемент появился, запускаем функцию
                    //console.log(222);
                    bid_draw();
                    // Отключаем наблюдение, чтобы функция не запускалась
                    повторно
                    observer.disconnect();
                }
            }
        })
    })
}

```

```

    });
});

// Начинаем наблюдение за изменениями в DOM
observer.observe(document.body, { childList: true, subtree: true });
}

```

Файл to_bid.php (подтверждение в БД)

```

<?php

session_start();
// Проверяем, что запрос был отправлен методом POST
if ($_SERVER['REQUEST_METHOD'] === 'POST')
{
    $authToken = $_COOKIE['auth_token'] ?? null;
    $ss = $_SESSION["auth_token"] ?? 1;

    if ($ss == $authToken)
    {

        // Получаем данные из тела запроса
        $json = file_get_contents('php://input');

        // Преобразуем JSON в массив PHP
        $data = json_decode($json, true);

        // Проверяем, что данные были успешно преобразованы
        if (is_array($data))
        {
            // Здесь вы можете обработать данные, например, сохранить их в базе
            данных

            include '../db_connect.php';
            $temp = bin2hex(random_bytes(20));
            foreach ($data as $key => $value)
            {
                for ($i=0; $i < intval($value); $i++)
                {

                    $sql = "INSERT INTO to_vet (id_cat, bid_id) VALUES (:id,
:bid_num)";

                    $stmt = $conn->prepare($sql);
                    $stmt->bindParam(':id', $key);

                    $stmt->bindParam(':bid_num', $temp);

                    $stmt->execute();

```

```
    }

    }

    echo json_encode(['status' =>"success", 'message' => 'Данные успешно
получены и обработаны']);
    } else {
        // Если данные не могут быть преобразованы в массив, возвращаем
ошибку
        http_response_code(400);
        echo json_encode(['status' =>"error", 'message' => 'Неверный формат
данных']);
    }
}
else
{
    // Если запрос не был отправлен методом POST, возвращаем ошибку
    http_response_code(405);
    echo json_encode(['status' =>"error", 'message' => 'Метод не
поддерживается']);
}
```

Приложение Б

Файл all_cats.js (выборка, фильтрация, упорядочивание)

```
function sortTable(n)
{
    var table, rows, switching, i, x, y, shouldSwitch, dir, switchcount = 0;
    table = document.getElementById("allCats-table-id");
    switching = true;
    dir = "asc";

    while (switching) {
        switching = false;
        rows = table.rows;
        for (i = 1; i < (rows.length - 1); i++) {
            shouldSwitch = false;
            x = rows[i].getElementsByTagName("TD")[n];
            y = rows[i + 1].getElementsByTagName("TD")[n];
            if (dir == "asc") {
                if (x.innerHTML.toLowerCase() > y.innerHTML.toLowerCase()) {
                    shouldSwitch = true;
                    break;
                }
            } else if (dir == "desc") {
                if (x.innerHTML.toLowerCase() < y.innerHTML.toLowerCase()) {
                    shouldSwitch = true;
                    break;
                }
            }
        }
        if (shouldSwitch) {
            rows[i].parentNode.insertBefore(rows[i + 1], rows[i]);
            switching = true;
            switchcount++;
        } else {
            if (switchcount == 0 && dir == "asc") {
                dir = "desc";
                switching = true;
            }
        }
    }
}
```

Приложение С

Файл register_process.php (обработка формы регистрации)

```
<?php
session_start();
function generate_authtoken($length = 32)
{
    // Генерируем случайные байты
    $random_bytes = random_bytes($length);
    // Преобразуем байты в шестнадцатеричную строку
    $token = bin2hex($random_bytes);
    return $token;
}

if ($_SERVER['REQUEST_METHOD'] === 'POST') {

    include '../db_connect.php';
    $data = $_POST['text'];

    $data = json_decode($data, true);
    // Получение данных из POST-запроса

    $check_username = "SELECT * FROM users WHERE username = :username";
    $stmt = $conn->prepare($check_username);
    $stmt->execute([':username' => $data['username']]);

    if ($stmt->rowCount() > 0)
    {
        echo json_encode(array("status" => "error", "message" => "Username already exists."));
        exit;
    }

    $auth_token = generate_authtoken();

    setcookie('auth_token', $auth_token, [
        'expires' => time() + 3600, // Установка времени истечения срока действия cookie
        'path' => '/', // Доступ к cookie на всем сайте
        'secure' => true, // Только через HTTPS
        'httponly' => true, // Только для HTTP, недоступно для JavaScript
    ]);
}
```

```

        'samesite' => 'Strict', // Защита от CSRF-атак
    ]);

    $sql = "INSERT INTO users (email, username, password_hash, auth_token) VALUES
    (:email, :username, :password_hash, :auth_token)";
    $stmt = $conn->prepare($sql);
    $stmt->execute([
        ':email' => $data['email'],
        ':username' => $data['username'],
        ':password_hash' => $data['password'],
        ':auth_token' => $auth_token
    ]);

    if ($stmt->rowCount() > 0)
    {
        include '../add_session_data.php';

        include "../mail_features/send_email.php";

        send_email($data['email'], "Welcome", "Спасибо что зарегистрировались на на
        нашем портале.", 'from@example.com');

        echo json_encode(array("status" => "success", "message" => "New record
        created successfully"));
    }
    else
    {
        echo json_encode(array("status" => "error", "message" => "Error: " . $sql .
        "<br>" . $conn->errorInfo()[2]));
    }
}

```

Файл logout.php (выход из профиля)

```
<?php
```

```

if ($_SERVER['REQUEST_METHOD'] === 'GET')
{

    include '../db_connect.php';

    if (isset($_COOKIE['auth_token']))
    {
        $auth_token = $_COOKIE['auth_token'];

        // Подготовка SQL запроса
        $stmt = $conn->prepare("UPDATE users SET auth_token = NULL WHERE
auth_token = :auth_token");

        // Привязка параметров
        $stmt->bindParam(':auth_token', $auth_token);

        // Выполнение запроса
        $stmt->execute();

        session_destroy();

    }
}

```

Файл login.php (api входа в профиль)

```

<?php
session_start();

function generate_auth_token($length = 32)
{
    // Генерируем случайные байты
    $random_bytes = random_bytes($length);
    // Преобразуем байты в шестнадцатеричную строку
    $token = bin2hex($random_bytes);
    return $token;
}

if ($_SERVER['REQUEST_METHOD'] === 'POST')
{

    include '../db_connect.php';

```

```

$data = $_POST['text'];

$data = json_decode($data, true);

// Проверка на кооректность данных
if (empty($data['username']) || empty($data['password']))
{
    echo json_encode(array("status" => "error", "message" => "All fields are
required."));
    exit;
}

$sql = "SELECT * FROM users WHERE username = :username";
$stmt = $conn->prepare($sql);
$stmt->execute([':username' => $data['username']]);
$user = $stmt->fetch(PDO::FETCH_ASSOC);

if ($user)
{
    $auth_token = generate_authtoken();

    if ($data['password'] == $user["password_hash"])
    {
        setcookie('auth_token', $auth_token, [
            'expires' => time() + 3600, // Установка времени истечения срока
            действия cookie
            'path' => '/', // Доступ к cookie на всем сайте
            'secure' => true, // Только через HTTPS
            'httponly' => true, // Только для HTTP, недоступно для JavaScript
            'samesite' => 'Strict', // Защита от CSRF-атак
        ]);

        $sql = "UPDATE users SET auth_token = :auth_token WHERE username =
:username";
        $stmt = $conn->prepare($sql);
        $stmt->execute([
            ':auth_token' => $auth_token,
            ':username' => $user["username"]
        ]);

        if ($stmt->rowCount() > 0)
        {
            include '../add_session_data.php';
        }
    }
}

```



```
        echo json_encode(array("status" => "success", "message" => "login
success"));

    }
    else
    {
        echo json_encode(array("status" => "error", "message" => "Error:
" . $conn->errorInfo()[2]));
    }

}
else
{
    echo json_encode(array("status" => "error", "message" => "invalid
password"));

}

}
else
{
    echo json_encode(array("status" => "error", "message" => "Username not
found."));
}
}
```