

Южно-Уральский
государственный
университет

Национальный
исследовательский
университет

КРОК

Аутентификация и авторизация. Работа с сессиями.

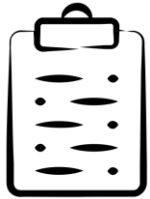
КРОК

Челябинск, ул. Карла Маркса, д. 38


Смирнов Анатолий
Технический менеджер


Антонов Сергей
Тимлид группы разработчиков,
старший инженер-разработчик

Кузнецов Сергей
Инженер-разработчик



- Аутентификация — процедура проверки подлинности, например проверка подлинности пользователя путем сравнения введенного им пароля с паролем, сохраненным в базе данных.
- Авторизация — предоставление определенному лицу или группе лиц прав на выполнение определенных действий.

 Аутентификация
Проверка подлинности (логин/пароль)

 Авторизация
Проверка доступа (роли)



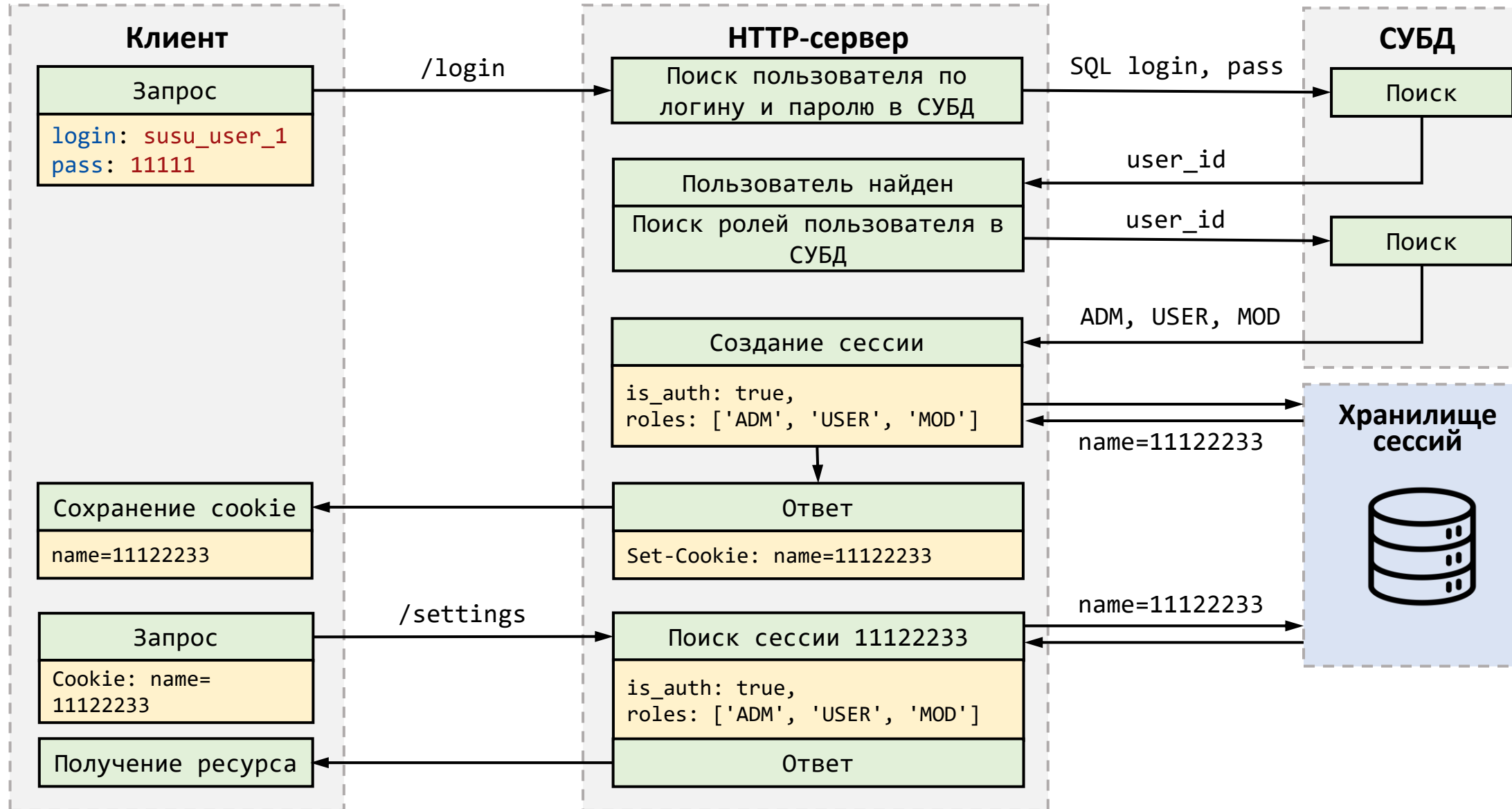
- Сессия — это диалоговое состояние между клиентом и сервером, включающее информацию о предыдущих запросах клиента и ответах сервера.
- Для передачи данного UID наиболее популярными являются такие способы, как использование пользовательских cookies, скрытые поля формы и передача непосредственно в адресе запроса.



- Куки (англ. cookies) — информация о пользователе, которую сервер передаёт браузеру при посещении сайта
- Эти данные позволяют идентифицировать посетителя сайта без повторной авторизации
- Если сервер хочет установить cookie для клиента, он указывает их в заголовке Set-Cookie
- Если клиент хочет передать на сервер он указывает их в заголовке Cookie

Сайт использует cookie. Вы можете отказаться от использования cookie, изменив настройки в браузере. Используя сайт, вы соглашаетесь на обработку персональных данных на условиях Политики.





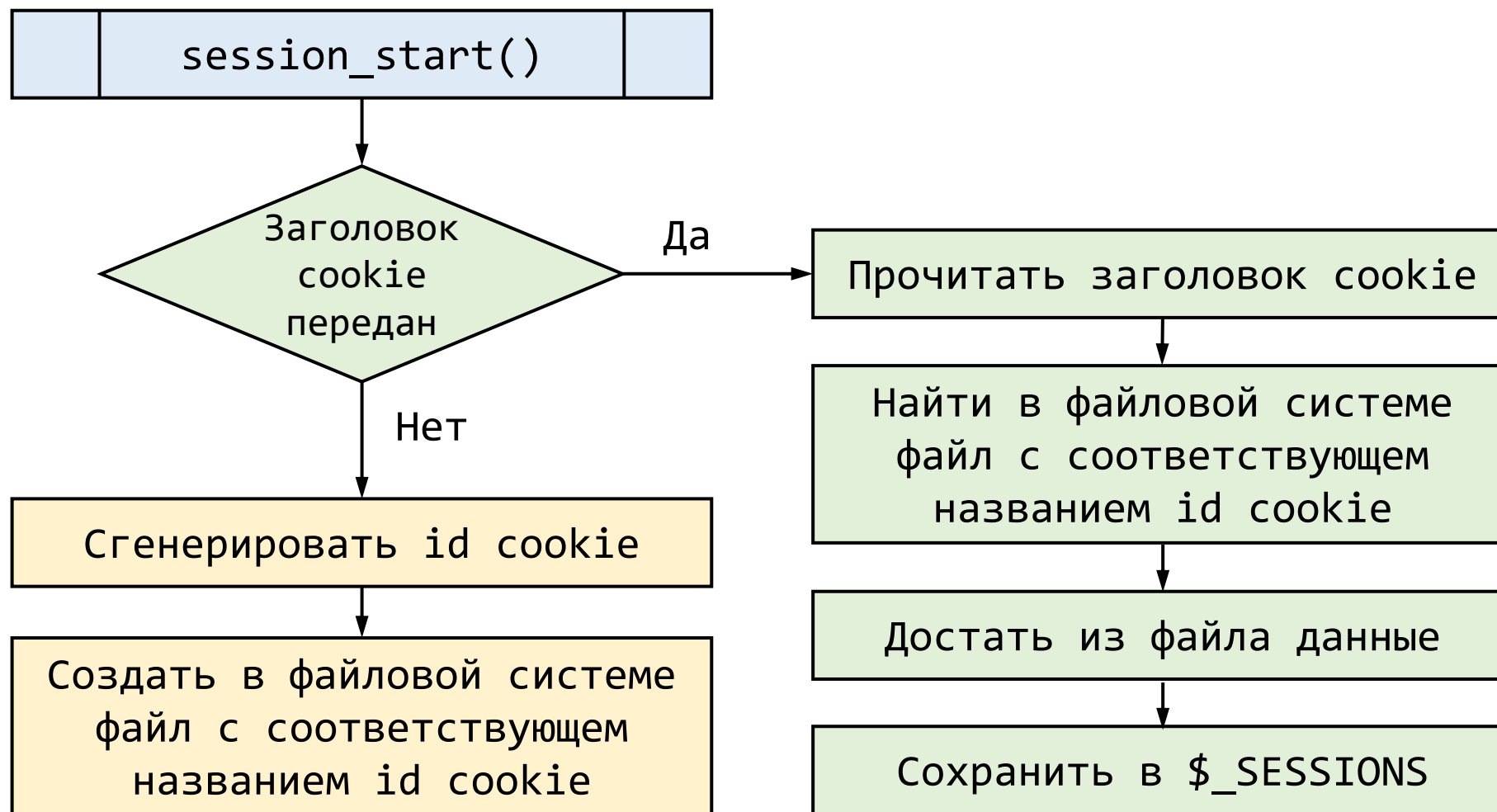


- По умолчанию PHP использует внутренний обработчик files для сохранения сессий, который установлен в INI-переменной `session.save_handler`. Этот обработчик сохраняет данные на сервере в директории `session.save_path`.
- Для начала работы с сессиями используется функция `session_start()`.
- Всю информацию о сессиях можно получить в глобальном массиве `$_SESSION`.
- Для очистки сессии используется `session_destroy()`.

```
httpd.conf x php.ini x
1348
1349 [Session]
1350 ; Handler used to store/retrieve data.
1351 ; https://php.net/session.save-handler
1352 session.save_handler = files
1353
```



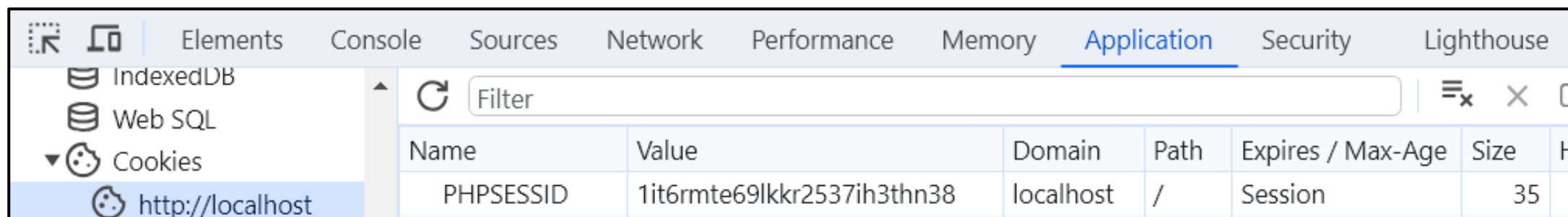
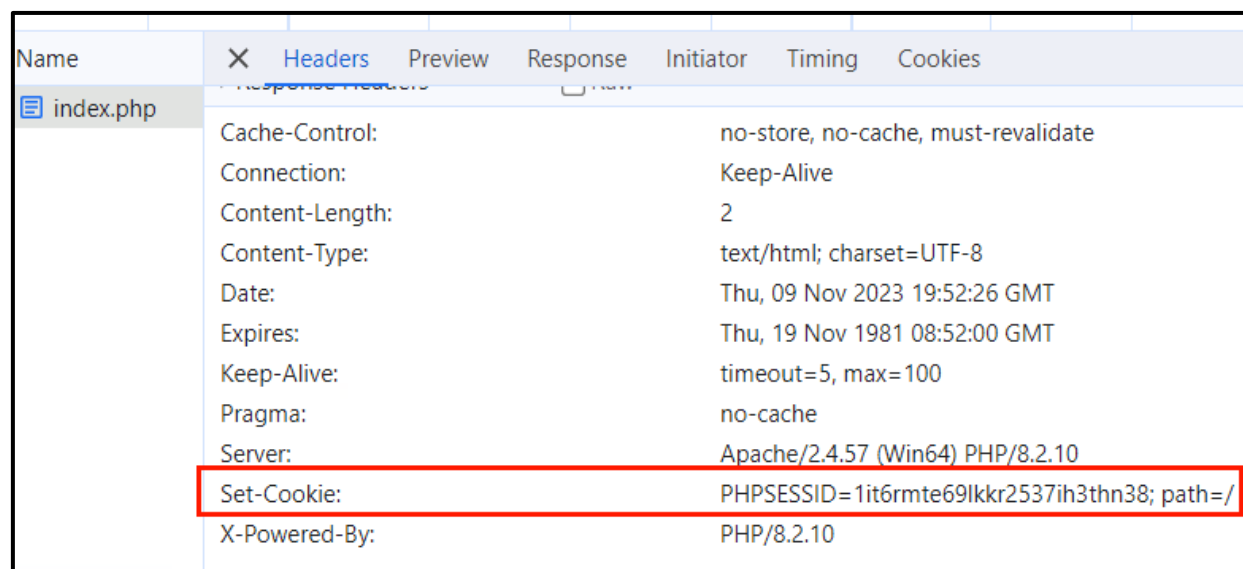
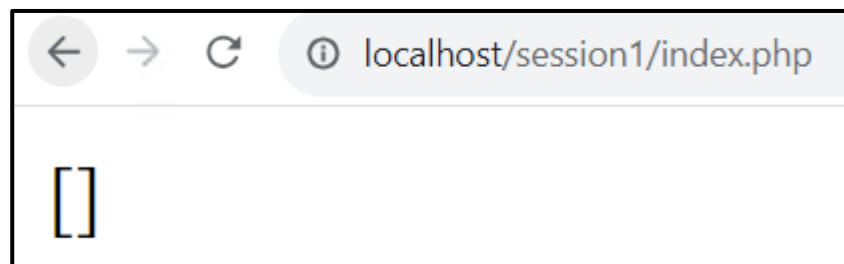
- Алгоритм работы `session_start()` по умолчанию:





- При обращении к ресурсу `/index.php` произошла установка сессии.
- Объект сессии не содержит данных.

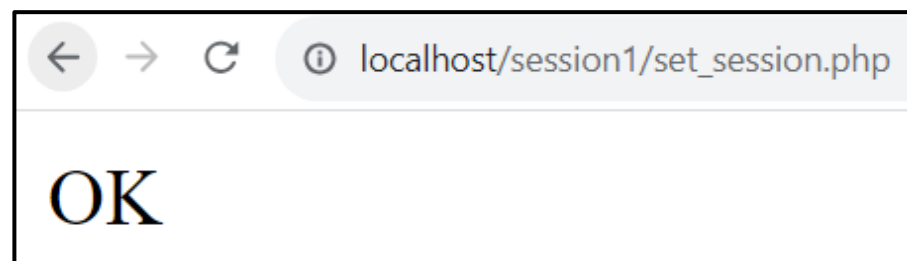
```
<?php
session_start();
echo json_encode($_SESSION,
JSON_UNESCAPED_UNICODE);
?>
```





- При обращении к ресурсу `/set_session.php` произошло сохранение данных в сессию.

```
<?php
session_start();
$_SESSION['IS_AUTH'] = true;
$_SESSION['name'] = 'Информация';
$_SESSION['roles'] = ['AMD', 'MOD'];
echo 'OK';
?>
```



The screenshot shows the Chrome DevTools Application tab. On the left, the 'Cookies' section is expanded for the domain 'http://localhost'. The main table displays the following data:

Name	Value	Domain	Path	Expires / Max-Age	Size	Http
PHPSESSID	1it6rmte69lkk2537ih3thn38	localhost	/	Session	35	



- Снова заходим на /index.php.
- Информация в сессии сохранилась.

```
<?php
    session_start();
    echo json_encode($_SESSION,
        JSON_UNESCAPED_UNICODE);
?>
```

Name	✕ Headers	Preview	Response	Initiator	Timing	Cookies
index.p...	Accept-Language:			ru-RU,ru;q=0.9,en-US;q=0.8,en;q=0.7		
	Cache-Control:			max-age=0		
	Connection:			keep-alive		
	Cookie:			PHPSESSID=1it6rmte69lkr2537ih3thn38		

← → ↻ ⓘ localhost/session1/index.php

```
{"IS_AUTH":true,"name":"Информация","roles":["AMD","MOD"]}
```



- Сессии по умолчанию хранятся в файловой системе.
- Можно открыть блокнотом.

Name	Value	Domain	Path	Expires / Max-Age	Size	Http
PHPSESSID	<u>1it6rmte69lkk2537ih3thn38</u>	localhost	/	Session	35	

Имя	Дата изменения	Тип	Размер
WinSAT	09.11.2023 13:14	Папка с файлами	
chrome_installer.log	09.11.2023 10:57	Текстовый докум...	121 КБ
<u>sess_1it6rmte69lkk2537ih3thn38</u>	09.11.2023 23:02	Файл	1 КБ
sess_7s6j7as4f5n62801oipbcbjufg	09.11.2023 17:21	Файл	0 КБ

```
IS_AUTH|b:1;name|s:20:"Информация";roles|a:2:{i:0;s:3:"AMD";i:1;s:3:"MOD";}
```



- Создадим таблицу пользователей

```
CREATE TABLE public.users
(
    id serial NOT NULL,
    name character varying(40) NOT NULL,
    login character varying(10) NOT NULL,
    pass character varying(20) NOT NULL,
    PRIMARY KEY (id),
    CONSTRAINT users_login_uk UNIQUE (login),
    CONSTRAINT users_pass_len CHECK (length(pass) >= 5) NOT VALID,
    CONSTRAINT users_login_len CHECK (length(login) >= 3) NOT VALID
);
```

id [PK] integer	name character varying (40)	login character varying (10)	pass character varying (20)
1	Иван	susu1	11111
2	Ольга	susu2	11111
3	Алексей	susu3	11111



- Создадим главную страницу /index.php.
- Если пользователь не аутентифицирован, то перенаправлять на страницу ввода логина и пароля.

```
<?php
    session_start();
    if (!array_key_exists('IS_AUTH', $_SESSION)) {
        header('Location: authorization.php');
        die();
    }
?>

<header>
    <?php
        echo $_SESSION['name'] . ' ' . $_SESSION['login'];
    ?>
    <form action="logout.php">
        <input type="submit" value="Выйти">
    </form>
</header>
```



- Если пользователь аутентифицирован, то перенаправлять на главную страницу.
- При нажатии на «Ок» идет запрос к /login.php.

```
<?php
    session_start();
    if (array_key_exists('IS_AUTH', $_SESSION)) {
        header('Location: index.php');
        die();
    }
?>
<meta charset="utf-8"/>
<form action="login.php" method="POST">
    Login:<br><input name="login"><br>
    Password:<br><input name="password" type="password"><br>
    <input type="submit" value="Ok">
</form>
```

localhost/session_demo/authorization.php

Логин:

Пароль:

Ок



- Ищем в БД пользователя по логину и паролю.
- Если найден – сохраняем данные в сессию и перенаправляем на главную.
- Если не найден – выдаем сообщение о неверных данных.

```
<?php
session_start();
if (array_key_exists('IS_AUTH', $_SESSION)) {
    header('Location: index.php');
    die();
}

$dsn = "pgsql:host=localhost;port=5432;dbname=postgres";
$pdo = new PDO($dsn, 'postgres', '1');

$sql = 'select name, id from users
       where login = :login and pass = :password';
$stmt = $pdo->prepare($sql);
$stmt->bindParam(':login', $_POST['login']);
$stmt->bindParam(':password', $_POST['password']);
$stmt->execute();
$row = $stmt->fetch(PDO::FETCH_ASSOC);
```

```
if ($row) {
    $_SESSION['IS_AUTH'] = true;
    $_SESSION['name'] = $row['name'];
    $_SESSION['user_id'] = $row['id'];
    $_SESSION['login'] = $_POST['login'];
    header('Location: index.php');
    die();
} else {
    echo 'Неверные логин или пароль!<br/>';
    echo '<form action="authorization.php">
        <input type="submit"
            value="Вернуться">
    </form>';
}
?>
```



- Неверные логин или пароль.

← → ↻ ⓘ localhost/session_demo/authorization.php

Логин:

Пароль:

Ок

← → ↻ ⓘ localhost/session_demo/login.php

Неверные логин или пароль!

Вернуться



- Если логин и пароль верные – сохранить данные в сессию и перенаправить на главную страницу.

← → ↻ ⓘ localhost/session_demo/authorization.php?

Логин:

Пароль:

Ок

← → ↻ ⓘ localhost/session_demo/index.php

Иван susu1

Выйти

nf x php.ini x sess_29up7f553shbn3ba67c4m5jhjm x

```
IS_AUTH|b:1;name|s:8:"Иван";user_id|i:1;login|s:5:"susu1";
```



- При нажатии на «Выйти» идет сброс сессии и перенаправление на страницу ввода логина и пароля.

```
<?php
    session_start();
    if (array_key_exists('IS_AUTH', $_SESSION)) {
        session_destroy();
        header('Location: authorization.php');
        die();
    }
?>
```

← → ↻ ⓘ localhost/session_demo/index.php

Иван susu1

ВЫЙТИ

← → ↻ ⓘ localhost/session_demo/authorization.php

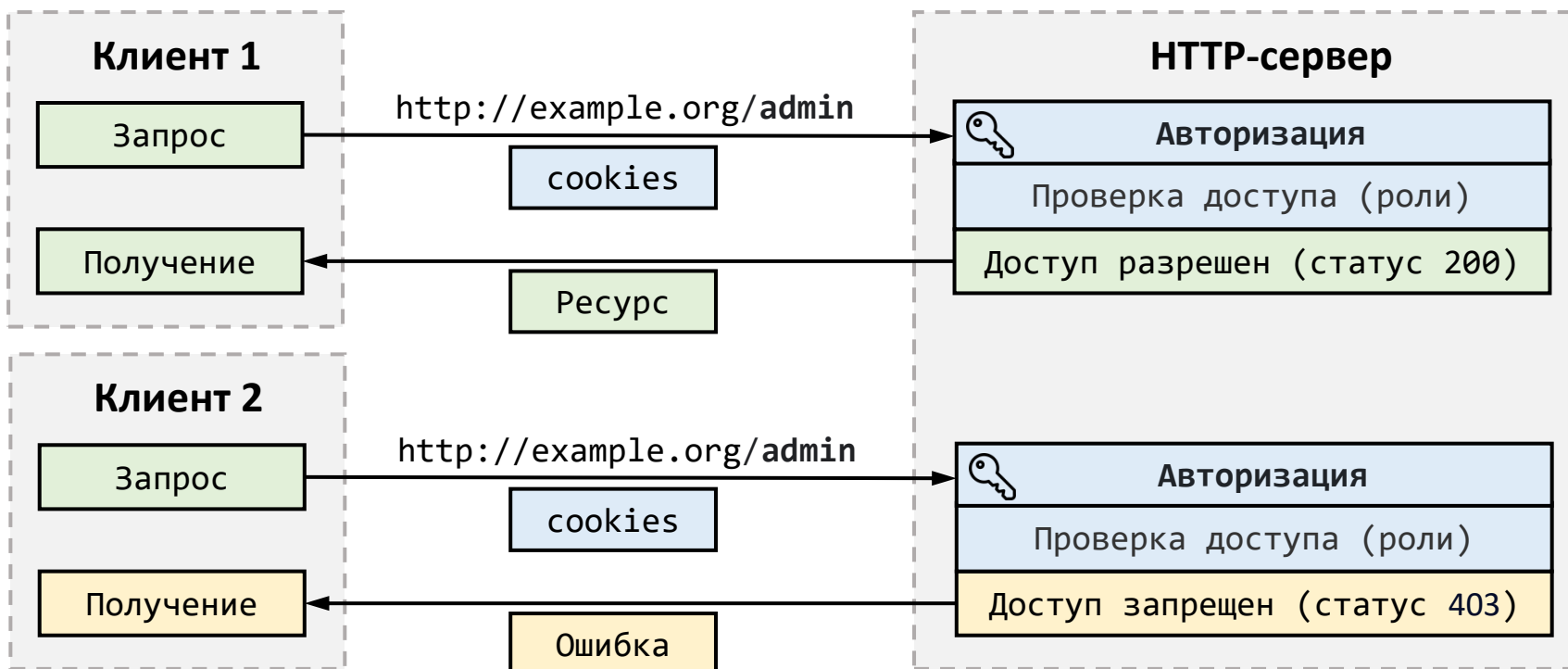
Логин:

Пароль:

Ок

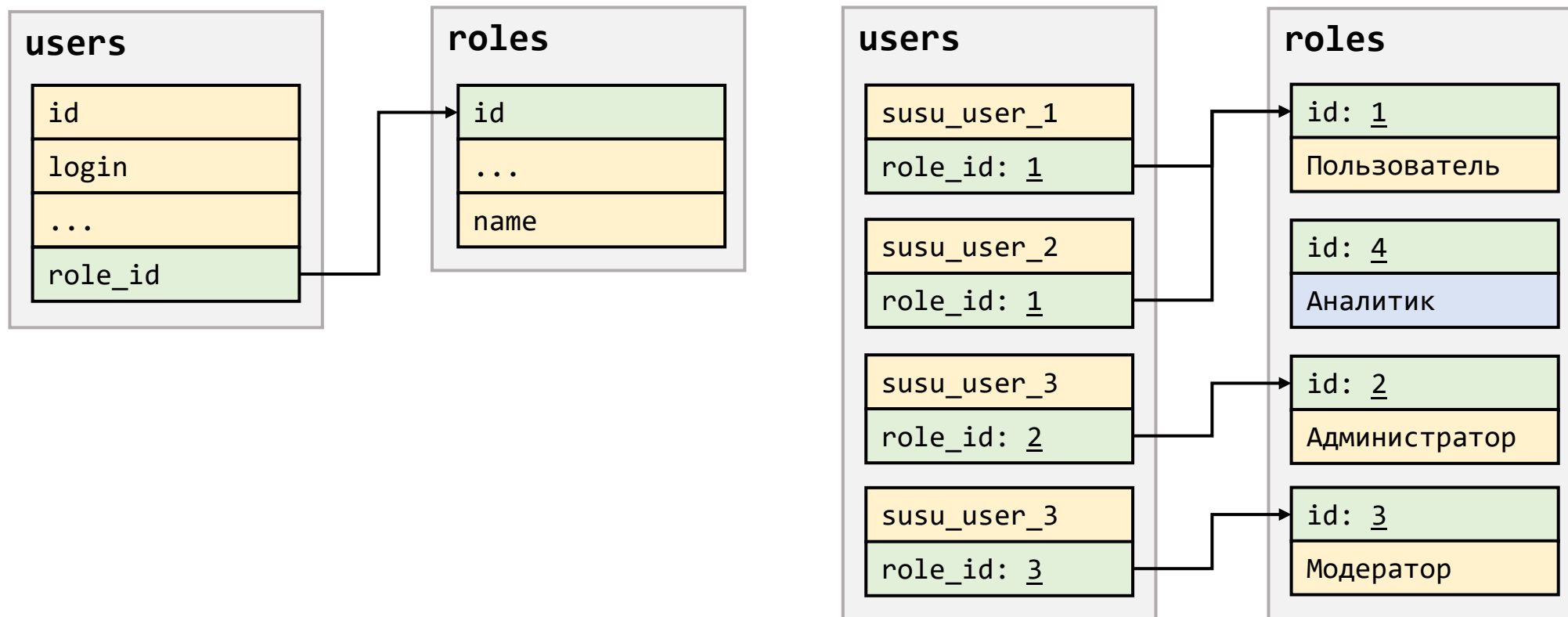


- Один из способов авторизации - управление доступом на основе ролей.
- Авторизация по ролям позволяет разграничить доступ к ресурсам приложения в зависимости от роли, к которой принадлежит пользователь.



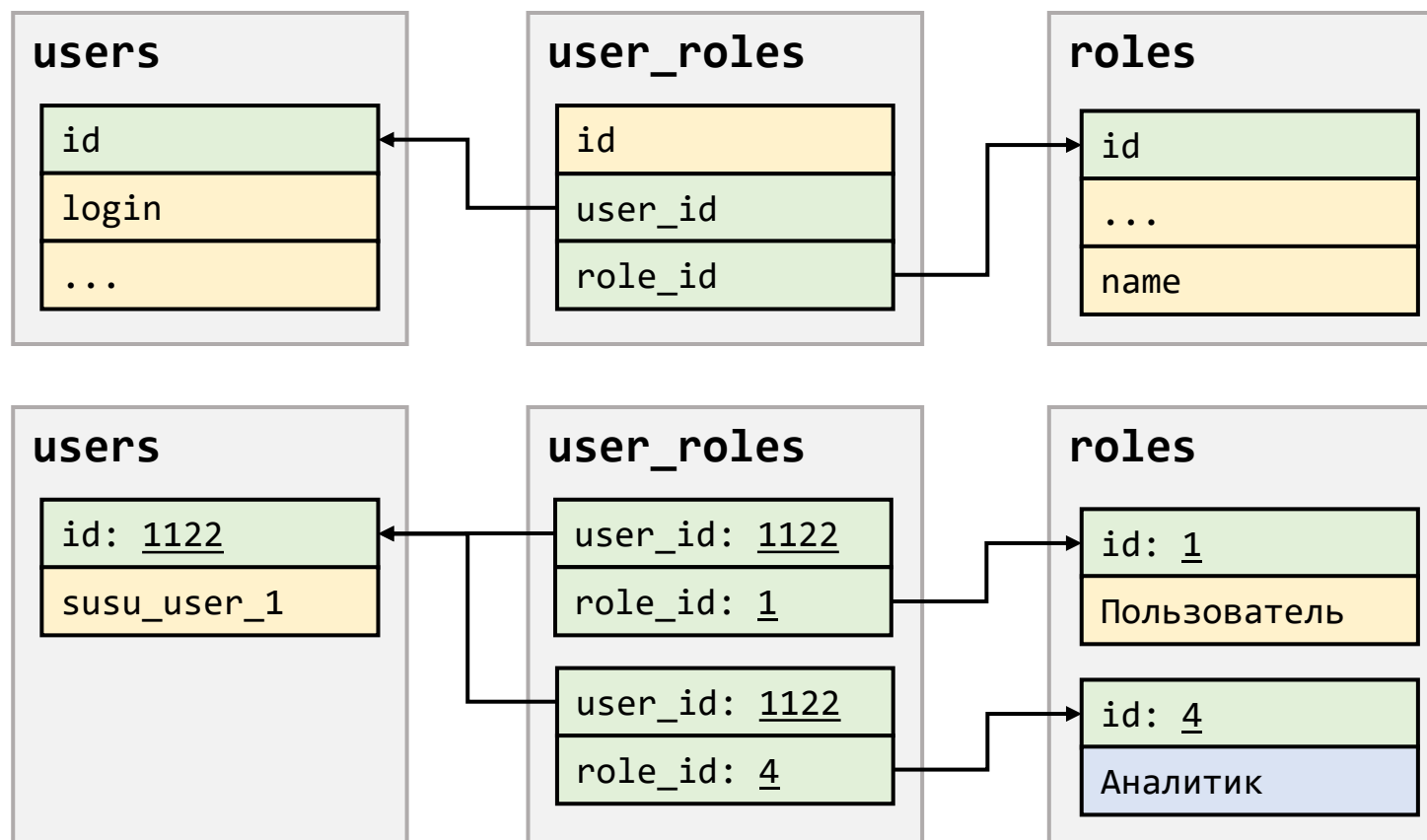


- Простейший вариант реализации: 2 таблицы
- Главный минус – нет возможности добавить несколько ролей одному пользователю (отсутствие гибкости)





- Следующий этап: 3 таблицы
- Главный плюс – возможность добавить несколько ролей для одного пользователя



Пример авторизации. Таблица «Роли»

```
CREATE TABLE public.roles
(  
    id serial NOT NULL,  
    code character varying(20) NOT NULL,  
    name character varying(40) NOT NULL,  
    PRIMARY KEY (id),  
    CONSTRAINT roles_code_uk UNIQUE (code),  
    CONSTRAINT roles_name_uk UNIQUE (name)  
);
```

id [PK] integer	code character varying (20)	name character varying (40)
1	ADM	Администратор
2	MOD	Модератор
3	AN	Аналитик
4	US	Пользователь

```
CREATE TABLE public.user_roles
(
    id serial NOT NULL,
    "user" integer NOT NULL,
    role integer NOT NULL,
    PRIMARY KEY (id),
    CONSTRAINT user_roles_user_fk FOREIGN KEY ("user")
        REFERENCES public.users (id),
    CONSTRAINT user_roles_role_fk FOREIGN KEY (role)
        REFERENCES public.roles (id)
);
```

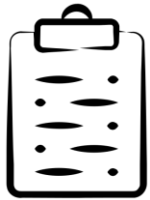
id [PK] integer	user integer	role integer
5	3	3
4	3	2
3	2	1
2	1	2
1	1	4

```
select r.name "Имя роли",  
       count(ur.user) "Кол-во пользователей"  
  from roles r  
     left join user_roles ur on ur.role = r.id  
 group by r.name
```

Имя роли	Кол-во пользователей
character varying (40)	bigint
Аналитик	1
Пользователь	1
Администратор	1
Модератор	2

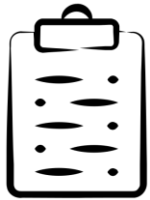
```
select u.login "Логин",  
       u.name "Имя пользователя",  
       string_agg(r.name, ', ') "Роли"  
  from users u  
     left join user_roles ur on ur.user = u.id  
     left join roles r on r.id = ur.role  
 group by u.login, u.name  
 order by u.login
```

Логин	Имя пользоват	Роли
character var	character varyin	text
susu1	Иван	Модератор, Пользователь
susu2	Ольга	Администратор
susu3	Алексей	Модератор, Аналитик



- Добавляем к аутентификации сохранение списка ролей в сессию.

```
if ($row) {  
    include('find_roles.php');  
    $_SESSION['roles'] = get_roles($row['id']);  
  
    $_SESSION['IS_AUTH'] = true;  
    $_SESSION['name'] = $row['name'];  
    $_SESSION['user_id'] = $row['id'];  
    $_SESSION['login'] = $_POST['login'];  
    header('Location: index.php');
```



- Файл `find_roles.php` достаёт роли из БД и возвращает массив этих ролей.

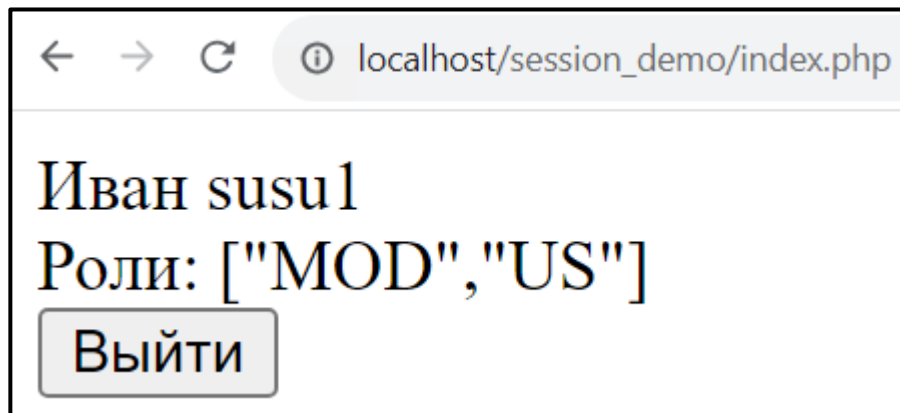
```
<?php
function get_roles($user_id) {
    $dsn = "pgsql:host=localhost;port=5432;dbname=postgres;";
    $pdo = new PDO($dsn, 'postgres', '1');

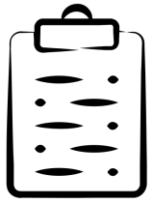
    $sql = 'select r.code
            from user_roles ur
            join roles r on r.id = ur.role
            where ur.user = :user_id';
    $stmt = $pdo->prepare($sql);
    $stmt->bindParam(':user_id', $user_id);
    $stmt->execute();
    return $stmt->fetchAll(PDO::FETCH_COLUMN, 0);
}
?>
```



- Добавляем вывод списка ролей на главную страницу:

```
<header>
  <?php
    echo $_SESSION['name'] . ' ' . $_SESSION['login'];
    echo '<br/>Роли: ';
    echo json_encode($_SESSION['roles']);
  ?>
  <form action="logout.php">
    <input type="submit" value="Выйти">
  </form>
</header>
```



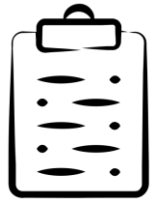


- Создаем страницу /admin.php:

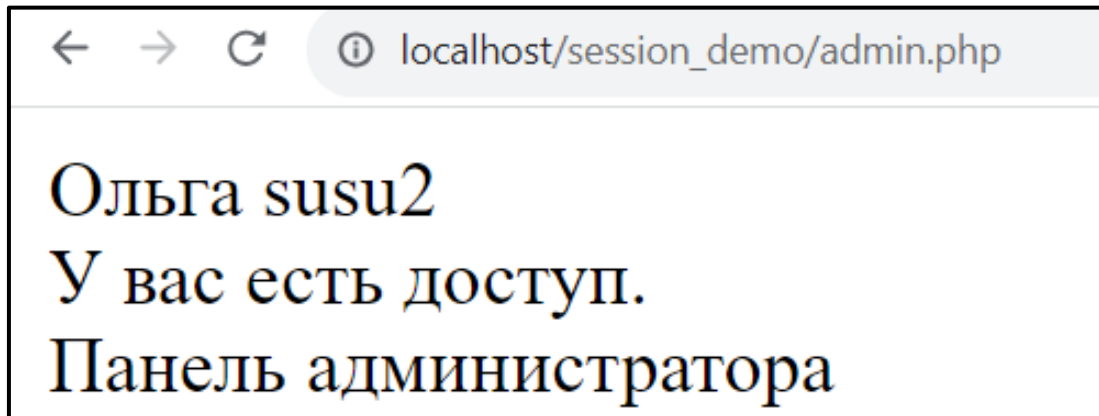
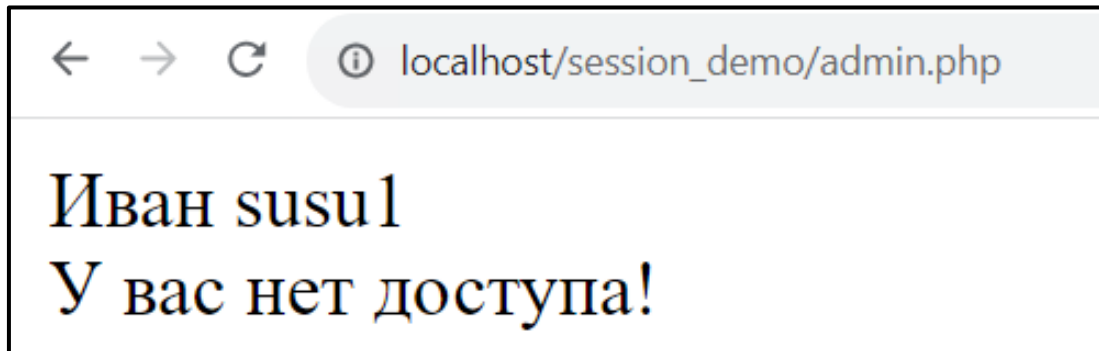
```
<?php
    session_start();
    if (!array_key_exists('IS_AUTH', $_SESSION) ||
        !array_key_exists('roles', $_SESSION)) {
        header('Location: authorization.php');
        die();
    }

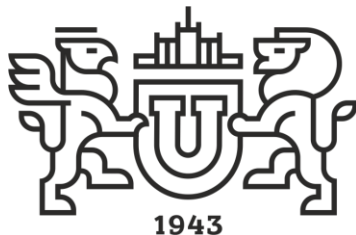
    echo $_SESSION['name'] . ' ' . $_SESSION['login'] . '<br/>';

    if (in_array('ADM', $_SESSION['roles'])) {
        echo 'У вас есть доступ.';
    } else {
        echo 'У вас нет доступа!';
        die();
    }
?>
<div>Панель администратора</div>
```



- Страница доступна только тем, у кого есть роль администратора.





Южно-Уральский
государственный
университет

Национальный
исследовательский
университет

КРОК

Спасибо за внимание!



КРОК

Челябинск, ул. Карла Маркса, д. 38