Оглавление

[**1. Формулировка задания 2**](#_gjdgxs)

[**2. Предметная область 3**](#_30j0zll)

[2.1 Описание предметной области и вводимых ограничений 3](#_1fob9te)

[2.2 Проектирование базы данных 4](#_2et92p0)

[2.3. Функциональная модель 6](#_tyjcwt)

[2.4. Алгоритмическая модель 7](#_4d34og8)

[**3. Реализация 8**](#_2s8eyo1)

[3.1. Логическое проектирование базы данных 8](#_17dp8vu)

[**3.2. Физическое построение базы данных 11**](#_26in1rg)

[1. Создание таблиц (users, access\_rights, status\_ap): 13](#_lnxbz9)

[2.Заполнение таблиц начальными данными: 13](#_35nkun2)

[3.3 Описание Структуры Проекта и используемых технологий 15](#_44sinio)

[3.4. Соответствие web-приложения требованиям 18](#_2jxsxqh)

[3.5. Соответствие системы требованиям 25](#_z337ya)

[**4. Демонстрация 33**](#)

[4.1. Демонстрация функционала работы неавторизированного пользователя 33](#_1y810tw)

[4.2. Демонстрация функционала авторизированного пользователя 35](#_wrqysmolz7er)

[4.3. Демонстрация функционала сотрудника 39](#_2xcytpi)

[4.4. Демонстрация функционала проверяющего отелей 41](#_36q70o2dgvbo)

# Формулировка задания

В соответствии с условием выполнения курсовой работы, полученным от

преподавателя, было необходимо создать прототип web-приложения турфирмы.

**Типовой сценарий работы системы:**

На домашней странице **незарегистрированный пользователь** видит список отелей, которые может предложить турфирма для выбора, а также регистрацию и аутоинфекцию для получения доступа.

**Зарегистрированный пользователь** видит ту же информацию, но у него так же пишется номер его телефона и статус доступа

**Сотрудник** может оформить заявку на тур.

На основе предоставленных от преподавателя указаний, Web-приложение должно соответствовать следующим требованиям и включать:

1. Набор динамических страниц PHP с использованием технологий HTML, хотя бы одна страница должна включать CSS , JavaScript, AJAX;
2. Несколько таблиц в MySql (в одной или более БД), кроме MySql можно дополнительно использовать другую СУБД (необходимо предварительно согласовать);
3. Home-страницу с несколькими пунктами меню (можно обычные ссылки) и авторизацией;
4. Два типа пользователей с разными правами (например, администратор, который может записывать в БД, и неавторизованный пользователь, который только считывает информацию);
5. Страницу (страницы) с вводом (изменением, добавлением) данных как минимум в 2 связанные таблицы БД одновременно (предполагается, что данные связаны, например, в одну таблицу записываются данные о персоне, а в связанную – список контактных телефонов);
6. Вывод данных по разным запросам (возможно, один запрос будет представлен графиком), как минимум один запрос должен быть по нескольким «связанным» таблицам БД; организация «постраничного» вывода таблиц – по желанию.
7. Асинхронный запрос (AJAX) хотя бы к одной таблице БД.

# 2. Предметная область

## **2.1 Описание предметной области и вводимых ограничений**

При исследовании предметной области были изучены сайты различных туристических фирм. На основе полученной информации были выделены четыре вида основных пользователей web – приложения:

1. Неавторизованный пользователь

Данный пользователь имеет минимальный набор прав доступа к системе таких как: просмотр главной страницы сайта, регистрация пользователя на сайте.

1. Авторизованный пользователь (клиент)

В дополнение к возможностям неавторизованного пользователя, данный тип пользователей может: смотреть статус и номер заявки.

1. Сотрудник (работник турфирмы)

Данный тип пользователя может получать информацию обо всех заявлениях, поданных клиентами, изменять статус заявки на «в обработке» с последующим подтверждением и сделать заявку.

## 

## **2.2 Проектирование базы данных**

В разработанной системе будет представлено три вида пользователей - клиент, сотрудник и проверяющий отеля. Они будут взаимодействовать следующим образом: **клиент** приходит в турфирму, **сотрудник** делает заявку,  **проверяющий отеля** проверяет в стороннем приложении наличие отеля и подтверждает или же отменяет заявку. Таким образом, главные сущности - **пользователь, заявка**.

В таблице **пользователь** будет храниться информация обо всех пользователях системы, в том числе и о сотрудниках с проверяющими отелей. Значит необходима следующая сущность – **статус доступа** – пользователь, сотрудник, проверяющий.

Турфирма дает пользователю смотреть номер и статус своей заявки, значит следующая сущность -  **заявка**. В ней будет находиться пожелания для тура (например: чтобы была двуспальная кровать, а не две односпальной), дата поездки и выбор направления - поездка будет в горы или на море. Так же в заявке будут находиться дополнительные услуги: нужно ли оформлять визу, включается ли трансфер до отеля или человек сам будет добираться и авиаперелет. Все данные пункты можно будет добавлять или нет.

При обработке заявки сотрудники могут менять статусы на:

* новая
* обработка
* отклонена
* подтверждена
* оплачена
* изменена
* отменена

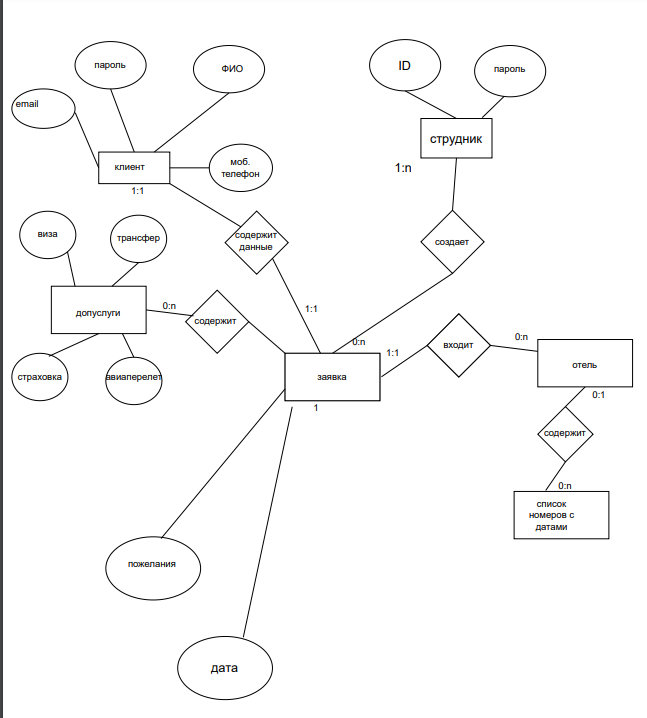


Рис. 2.1. - Концептуальная модель базы данных в нотации Питера Чена

Концептуальная модель базы данных (ER диаграмма) представлена на (Рис. 2.1). Все сущности, связи между ними и их количество можно увидеть на (Рис. 2.1). Одну заявку может создать только один пользователь, но у одного пользователя может быть множество заявок. У каждого пользователя есть только один статус доступа. У одной заявки, может быть, только тур его может не быть вообще. У каждой заявки может быть несколько статусов, записи о которых заносятся в форму изменения статуса заявки. Также в форму изменения статуса заявки всегда заносится информация о том, кто именно, что именно и когда изменял в статусах данного заявления.

## **2.3. Функциональная модель**

Основной функционал, предоставляемый системой, описан на Use Case диаграмме (Рис. 2.2).

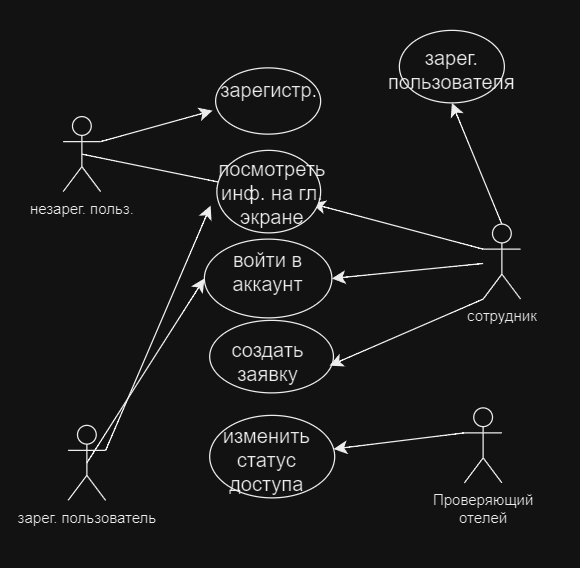


Рис. 2.2 – Функциональная диаграмма (Use Case)

## 

## 

## **2.4. Алгоритмическая модель**

Основной алгоритм обработки заявлений представлен на схеме на рис. 2.3:

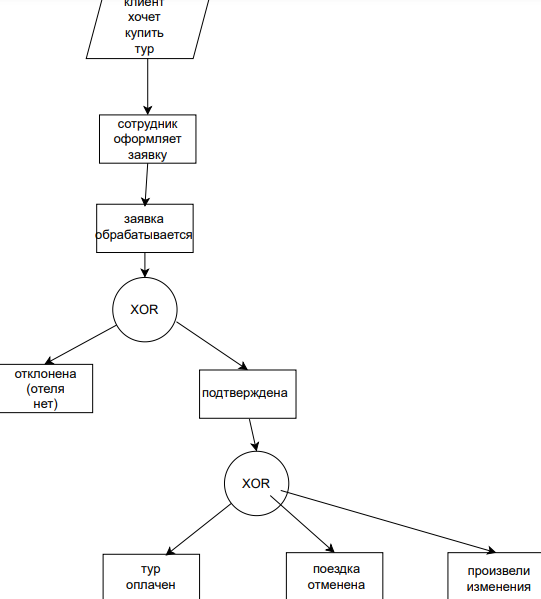


Рис. 2.3 – Алгоритм обработки заявки

# 3. Реализация

## **3.1. Логическое проектирование базы данных**

База данных **tour** состоит из следующих таблиц:

Для таблицы **users:**

| **Атрибут** | **Расшифровка** |
| --- | --- |
| user\_id | айди юзера |
| acs\_righrs\_id | айди статуса доступа |
| user\_name | фио пользователя |
| User\_mail | почта пользователя |
| number\_tel | номер телефона пользователя |
| pasword | пароль |

Для таблицы **status\_ap**:

| **Атрибут** | **Расшифровка** |
| --- | --- |
| status\_id | fqlb идентификатор статуса заявки |
| status\_aplication | Наименование статуса заявки |

Для таблицы **access\_rights**:

| **Атрибут** | **Расшифровка** |
| --- | --- |
| acs\_rights\_id | айди идентификатор индекса, в котором хранятся права доступа пользователя (пользователь, сотрудник, эксперт, администратор) |
| access\_right | Наименование прав доступа пользователя |

Для таблицы **hotel:**

| **Атрибут** | **Расшифровка** |
| --- | --- |
| hotel\_id | айди отеля |
| hotel\_name | Название отеля |
| 1\_room | однокомнатные апартаменты |
| 2\_room | двухкомнатные апартаменты |
| country | страна поездки |

Для таблицы **adress:**

| **Атрибут** | **Расшифровка** |
| --- | --- |
| id\_adress | айди адреса |
| city | город клиента |
| street | улица проживания |
| home | дом |

Для таблицы **aplication:**

| **Атрибут** | **Расшифровка** |
| --- | --- |
| aplication\_id | айди заявки |
| user\_id | айди для отображения заявки в личном кабинете |
| status\_id | айди для индитификации заявки |
| wishes | пожелания клиента |
| date\_trip | дата поездки |
| fly\_see | поезка на море |
| trip\_to\_the\_cave | поездка в гору |
| visa | виза |
| transfer | трансфер до отеля из аэропорта |
| air\_travel | авиаперелет |
| editing | изменения |
| aplication\_user | статус блокировки |

## 

## **3.2. Физическое построение базы данных**

В качестве СУБД для реализации разработанной базы данных был выбран MySQL. Исходя из приведённой ранее ER модели была построена следующая физическая модель (Рис. 3.2.1).

На приведенном (Рис. 3.2.1) типы данных обозначаются следующими значениями (в силу особенностей DBeaver — средства, используемого для визуализации физической модели):

* 123» - целое число ( integer )
* «abc» - строка ( varchar )
* Синий круг - дата ( timestamp )

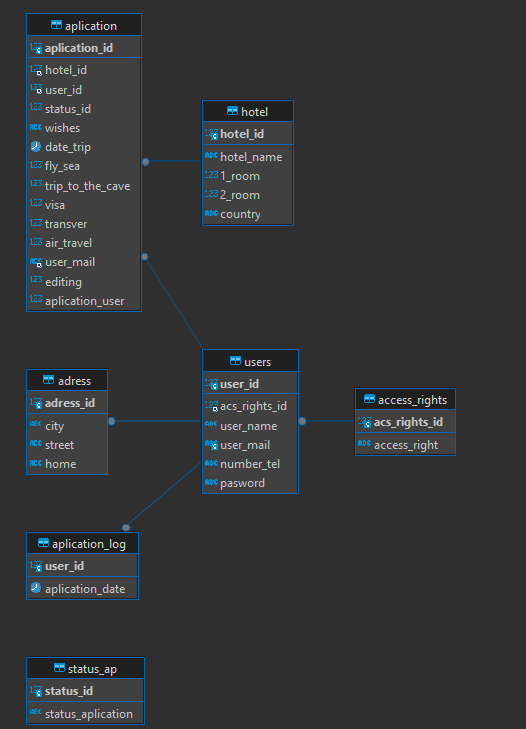


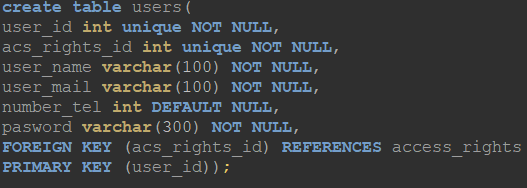
рис. 3.2.1 – физическая модель базы данных

Полный скрипт создания базы данных и скрипт для заполнения базы данных представлен в SQL файле creat\_bd\_tour.php.

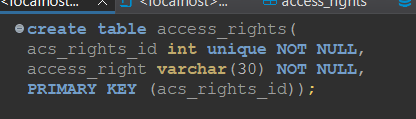
Ниже приведены примеры создания таблиц с данными о пользователях, статусах пользователей и статусах заявок и их заполнения:

### Создание таблиц (users, access\_rights, status\_ap):

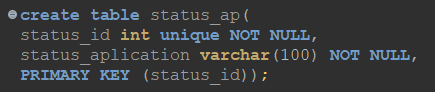
Создание таблицы (**users**):



Создание таблицы (**access\_rights**):

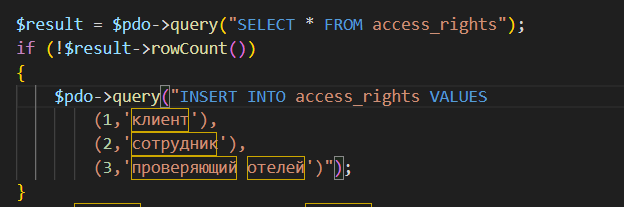


Создание таблицы (**status\_ap**):

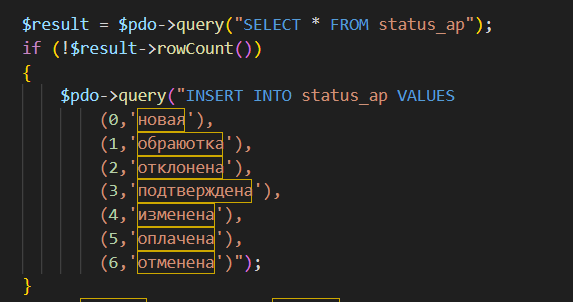


### 2.Заполнение таблиц начальными данными:

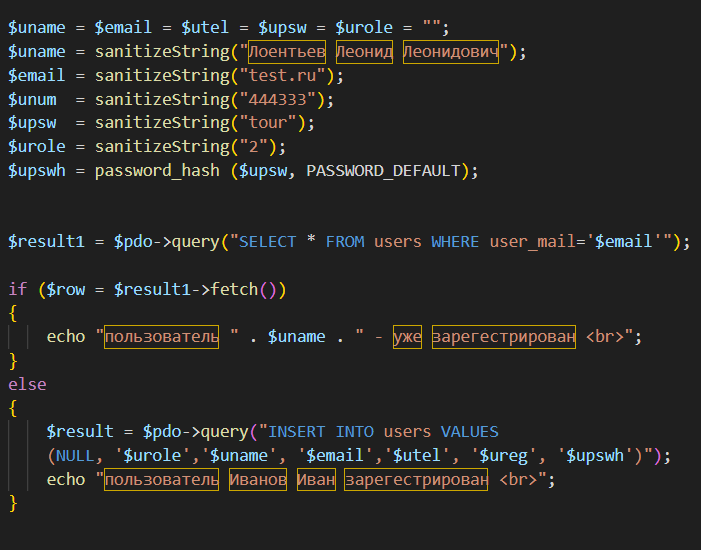
Заполнение таблицы (**access\_rights**):



Заполнение таблицы (**status\_ap**):



Заполнение таблицы (**users**):

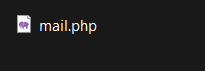


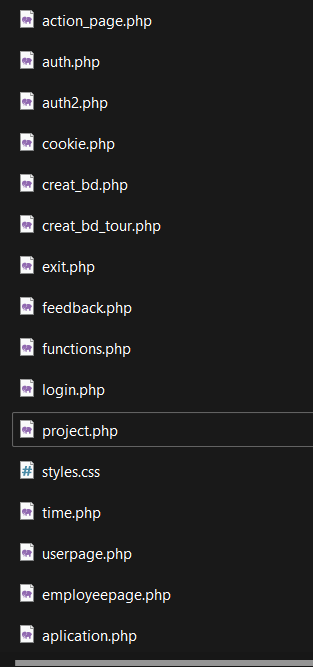
## 

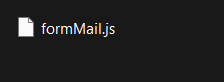
## **3.3 Описание Структуры Проекта и используемых технологий**

В качестве сервера, на котором работает созданное web-приложение, был выбран **Wampserver.** Для написания backend-части приложения был выбран язык **PHP.** Для работы с базой данных **MySQL.** Для статических элементов использовались **HTML5, CSS3, JavaScript**.

В главной директории проекта лежат php-файлы, отображающие страницы, которые видит пользователь, а также файлы, реализующие backend приложения. (Рис. 3.3.1, Рис. 3.3.2, Рис. 3.3.3, Рис 3.3.4, Рис. 3.3.5)

  
 Рис. 3.3.1

  
Рис. 3.3.3

  
Рис. 3.3.2

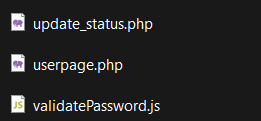
  

Рис 3.3.5

**mail.php** – в файле содержится код, который отправляет запрос на сервер(почту), если пользователь делает запрос для обратной связи

**action\_page.php** - в файле содержится код, который содержит регистрацию пользователя

**auth.php** - в файле содержится код, который содержит вход зарегистрированного пользователя

**auth2.php** - в файле содержится код, который содержит вход сотрудника

**auth3.php -** в файле содержится код, который содержит вход проверяющего отелей

**cookie.php** - в файле содержится код, который содержит функцию с работой cookie

**creat\_bd.php** - в файле содержится код, который содержит создание базы данных

**creat\_bd\_tour.php** - в файле содержится код, который содержит создание атрибутов для базы данных

**exit.php** - в файле содержится код, который содержит выход из учетной записи

**feedback.php** - в файле содержится код, который содержит форму с обратной связью

**funktions.php** - в файле содержится код, который содержит функции, которые были использованы дополнительно

**login.php** - в файле содержится код, который содержит подключение к базе данных

**project.php** - в файле содержится код, который содержит основной файл с домашней страницей

**styles.css** - в файле содержится код, который содержит стили, использованные для оформления динамических страниц

**time.php -** в файле содержится код, который содержит выход из сессии через некоторое время

**userpage.php** - в файле содержится код, который отображает страницу с правами доступа зарегистрированного пользователя

**emploeepage.php** - в файле содержится код, который отображает страницу с правами доступа сотрудника турфирмы

**aplication.php** - в файле содержится код, который отображает заполнение заявки

**formMail.js** - в файле содержится код, который содержит обработку выведенного сообщения для обратной связи

**change\_status2.php -** в файле содержится код, который содержит блокировку таблицы

**validatePassword.js** - в файле содержится код, который содержит проверку пароля, для избежания взлома

**inspectorpage.php -** в файле содержится код, который содержит функционал проверяющего отелей

## **3.4. Соответствие web-приложения требованиям**

1. Был реализован набор динамических страниц PHP с использованием технологий HTML и CSS. Созданы страницы с использованием JavaScript, AJAX (log02.php) (Рис. 3.4.1)

Рис. 3.4.1

2. Была создана база данных tour (см. файлы creat\_bd\_tour.php, create\_bd.php) (См. Рис. 3.2.1).

3. Была реализована home-страница с несколькими пунктами меню (project.php) (Рис. 3.4.2), страница с авторизацией (auth.php) (Рис. 3.4.3) и регистрацией (actipn\_page.php) (Рис. 3.4.4)

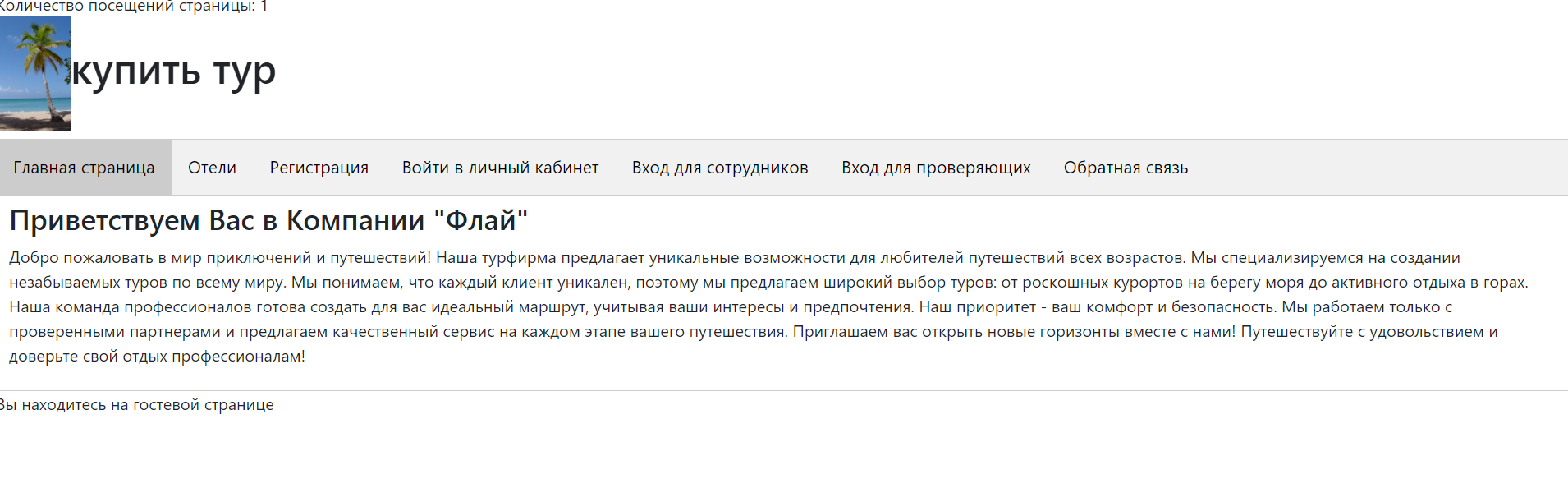


Рис. 3.4.2

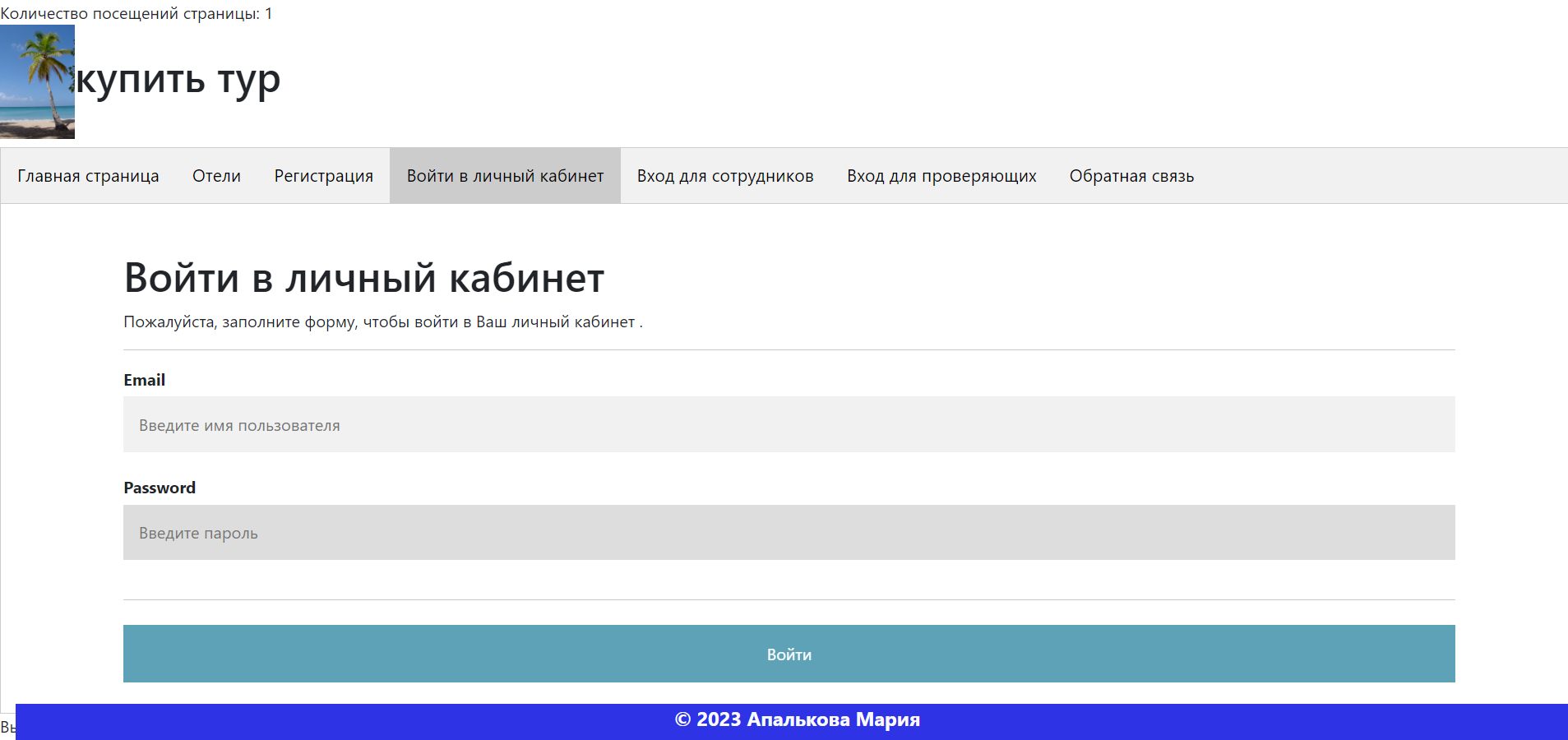


Рис 3.4.3

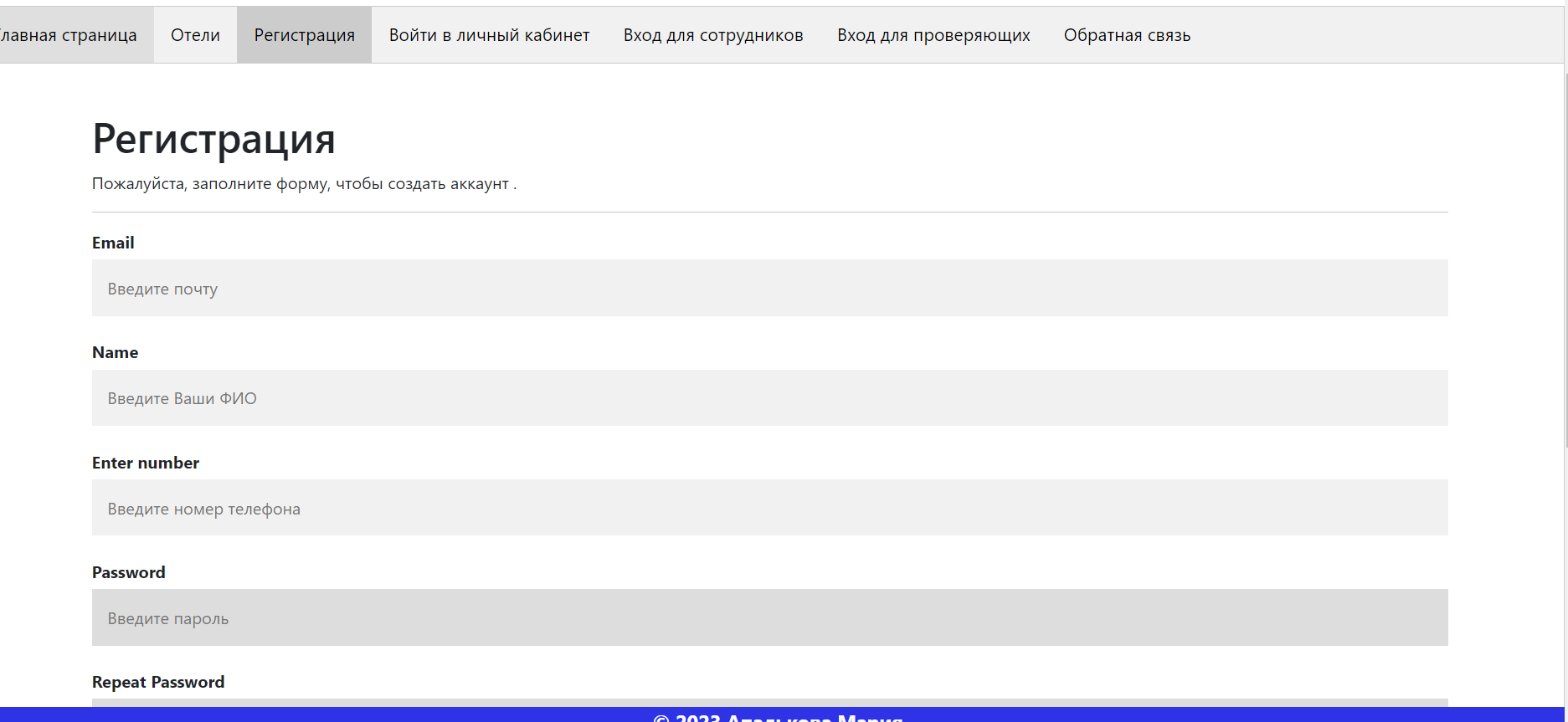


Рис 3.4.4

4. Было создано 3 типа пользователей, обладающих разными правами:

● **Неавторизованный пользователь**, имеющий возможность:

○ Просмотреть главную страницу системы

○ Зарегистрироваться

●  **Авторизованный пользователь**, имеющий возможность:

○ Просмотреть главную страницу системы

○ Посмотреть свой статус доступа и номер телефона

●  **Сотрудник**, имеющий возможность:

○  Создавать заявки

○  Регистрировать нового пользователя

● **Проверяющий отелей**, имеющий возможность:

○ Изменять статусы заявок

5. Был реализован ввод данных в 2 таблицы одновременно на страницы регистрации, когда данные пользователя попадают в **user**, а его адрес регистрации в **address**.

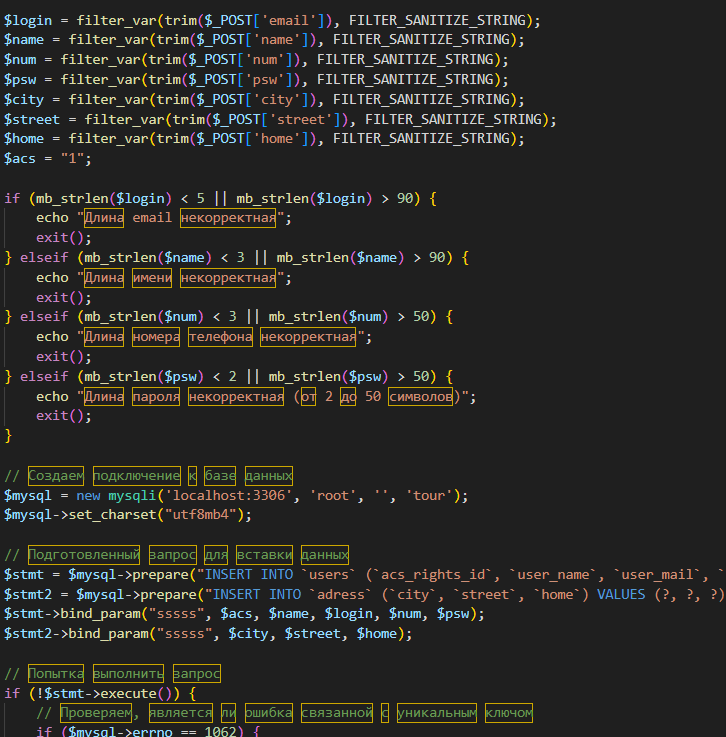


Рис 3.4.5. - заполнение двух таблиц базы данных одновременно

6. Был реализован вывод данных из нескольким связанных таблиц. На странице **userpage.php** пользователь может посмотреть номер телефона, который хранится в таблице **user** и статус своего доступа, который хранится в таблице **acs\_rights** (Рис. 3.4.6, 3.4.7)

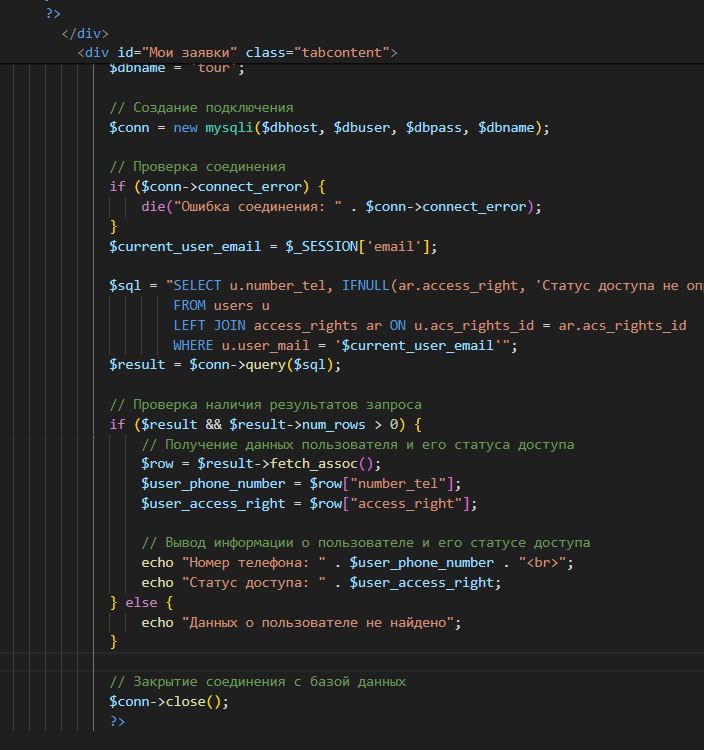


Рис 3.4.6 - Код для получения информации из двух таблиц

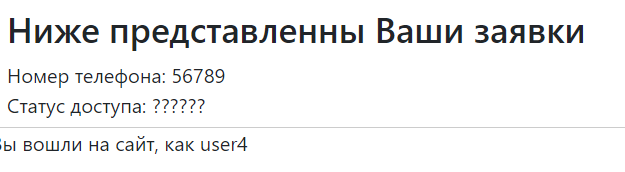


Рис 3.4.7. - Вывод номера телефона пользователя

7. Был реализован асинхронный запрос к таблицам базы данных. Реализация данного метода хранится в файле mail.php (Рис. 3.4.8). и userpage.php (Рис 3.4.9)

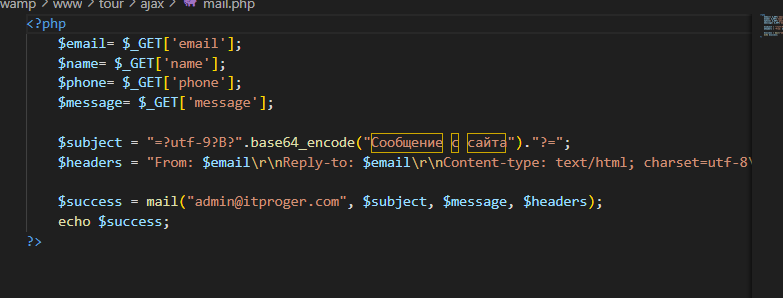
 

Рис. 3.4.8 - Код асинхронного запроса обратной связи

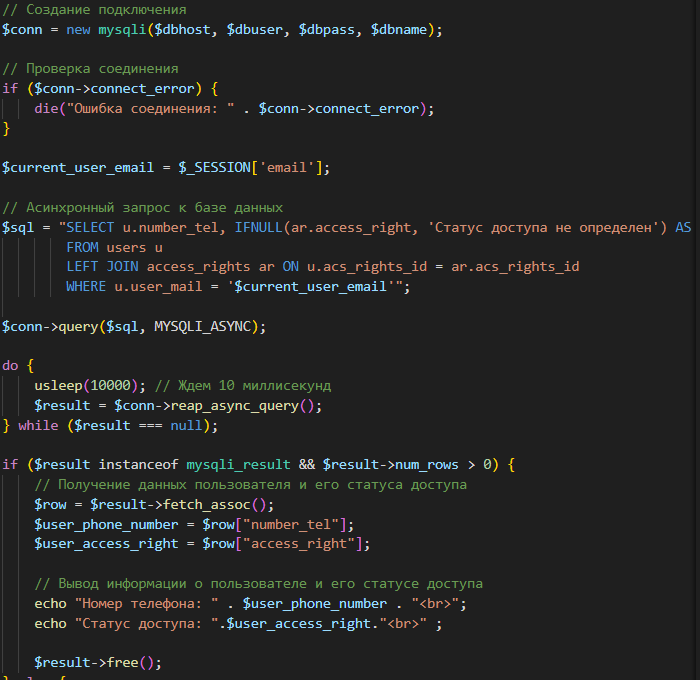


Рис 3.4.9 - Код асинхронного запроса вывода информации о пользователе

Также была реализована отправка сообщений – обратная связь с отправкой сообщений на сервер (Рис 3.4.9)



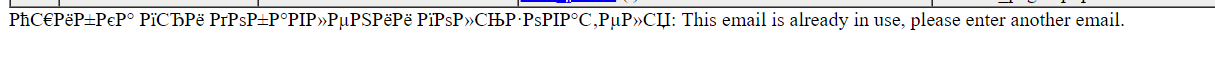
Рис 3.4.9 - Форма обратной связи

## 

## **3.5. Соответствие системы требованиям**

1. Скрипты для создания и заполнения базы данных находятся в файлах creat\_bd.php, creat\_bd\_tour.php (Примеры создания и заполнения таблиц приведены в пункте **3.2)**
2. Был реализован триггер, который перед тем, как зарегистрировать пользователя проверяет отсутствие зарегистрированного пользователя с такой почтой.

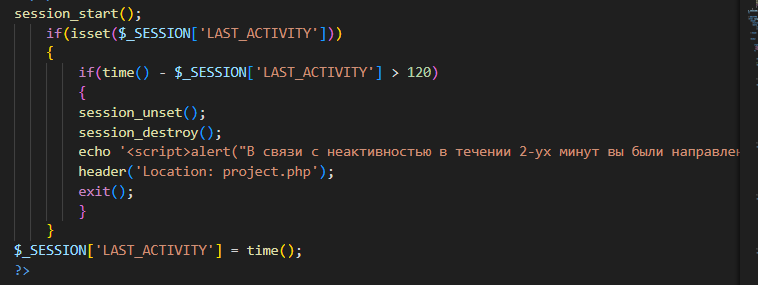
3.5.1. - триггер



3.5.2 - результат работы триггера

3. Код для создания сайта разбит на модули php. Их описание представлено в пункте **3.3.**

4. Была организована сессия продолжительностью 2 минуты для авторизованного пользователя. После 2 минут бездействия работа продолжается, как с неавторизованным пользователем. Если работа была приостановлена на странице, которая предназначается только для авторизированных пользователей, то пользователя перенаправляет на главную страницу. Код в файле project.php. (Рис. 3.5.3)



3.5.3. - Код, описывающий максимальное время работы

По истечении 2-минутной сессии, система сообщает пользователю о необходимости повторной авторизации.

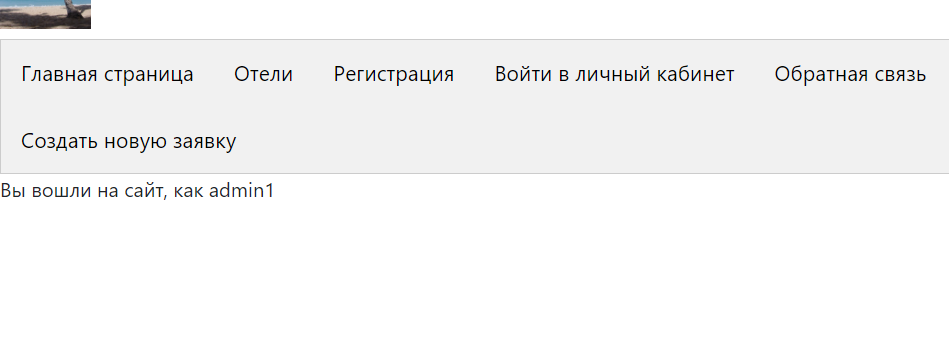
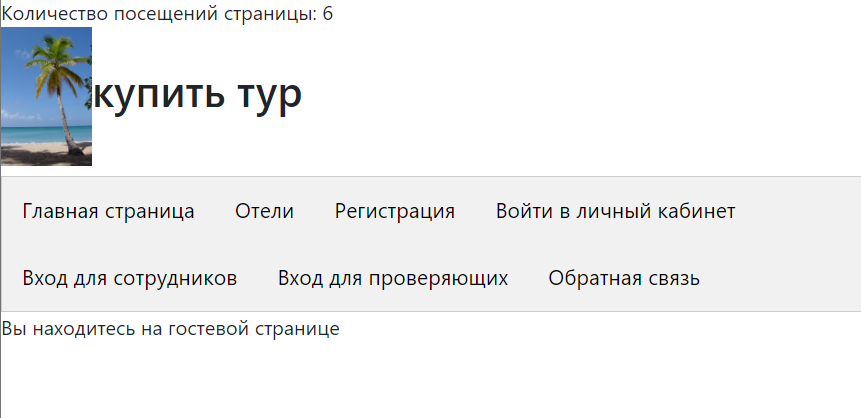


Рис 3.5.4. - Начало сессии



3.5.5. - Конец сессии

5. Сookies используются в файле cookies.php – т. е. реализован механизм подсчета посещений страницы с использованием time.php (Рис. 3.5.6, 3.5.7)

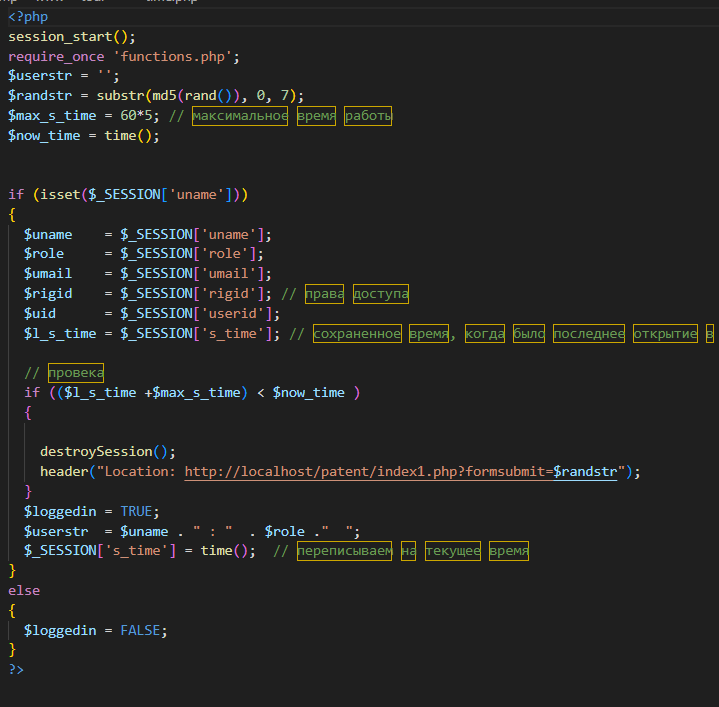


Рис. 3.5.6 – Код cookies

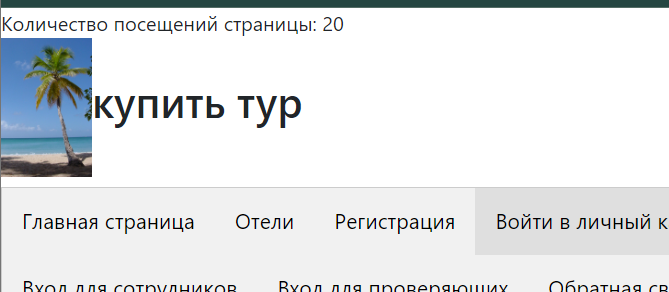


Рис. 3.5.7 – Сообщение о кол-ве посещений страницы

6. Были использованы методы GET и POST

В рамках асинхронного запроса происходит использование метода Get ( Рис. 3.4.8). Так же при выводе номера телефона пользователя и его статуса доступа (рис 3.4.9)

POST реализуется в виде форм регистрации

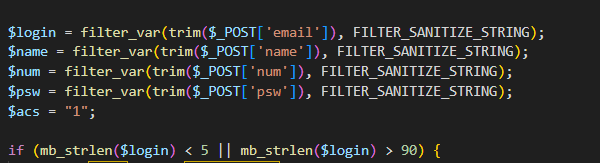


Рис. 3.5.8 – Пример использования метода POST

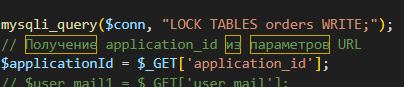


Рис 3.5.9 - Пример использования метода GET

7. Приложение способно обрабатывать ошибки пользователя если пользователь при регистрации вводит недопустимые по длине значения имени или номера телефона, программа выводит ошибку. Так же ошибка выводится, если не хватает каких-либо данных при регистрации

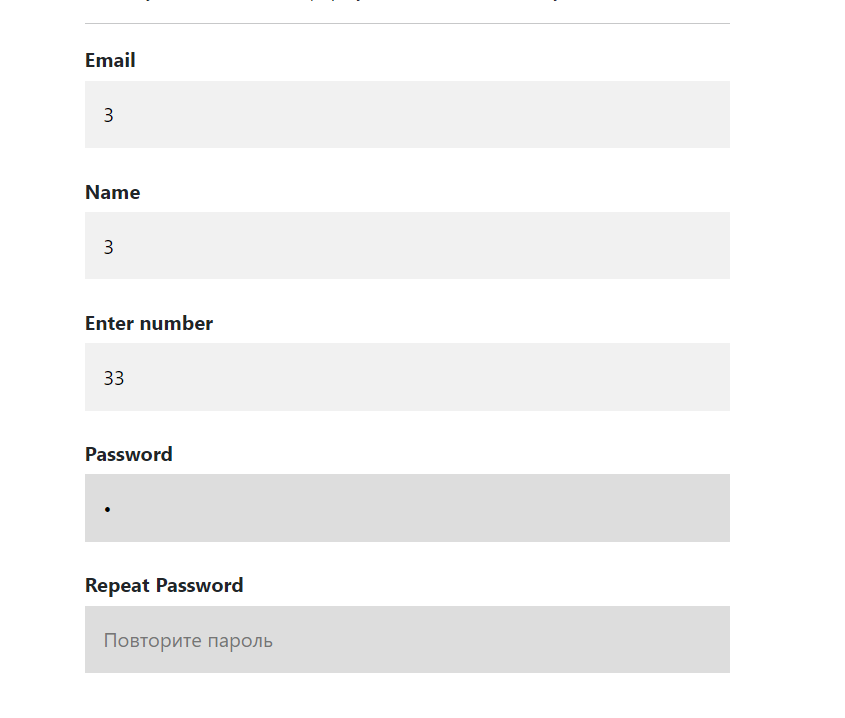


Рис 3.5.10 - Пример неправильно введенных данных

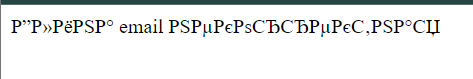


Рис. 3.5.11 – Предупреждение о некорректности введенного электронного адреса

Есть обязательные поля, не заполнив которые регистрация не проходит

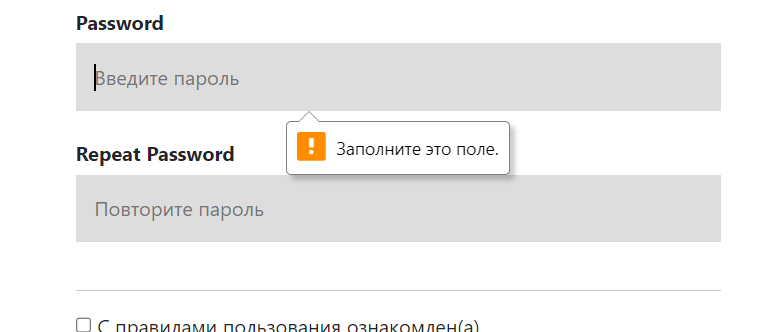


Рис 3.5.12 - Пример не заполнения нужных строк

8. При реализации проекта был разработан собственный интерфейс, выполненный в черно-белой гамме, с акцентом на подсветку важных. Они оформлены в зеленом цвете для удобства работы (Рис. 3.5.13 и Рис 3.5.14).Также, был проведен опрос и демонстрация интерфейса друзьям, которые, сообщили, что интерфейс удобен в использовании.

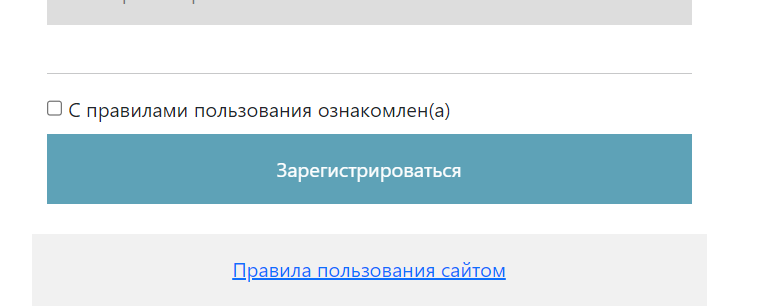


Рис. 3.5.13 – Демонстрация цветовой гаммы для кнопок

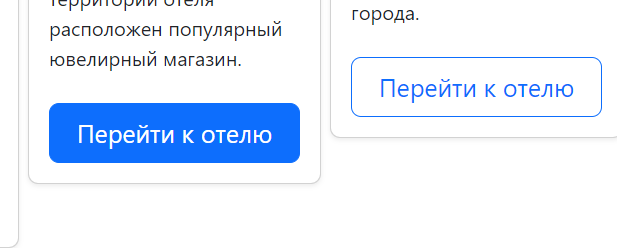


Рис. 3.5.14 – Демонстрация цветовой гаммы для кнопок

9. Для всех страниц сайта применяются стили для аккуратного и красивого оформления. Все используемые стили представлены в файле c CSS корневого каталога. Файл styles.css. (Рис. 3.5.15)

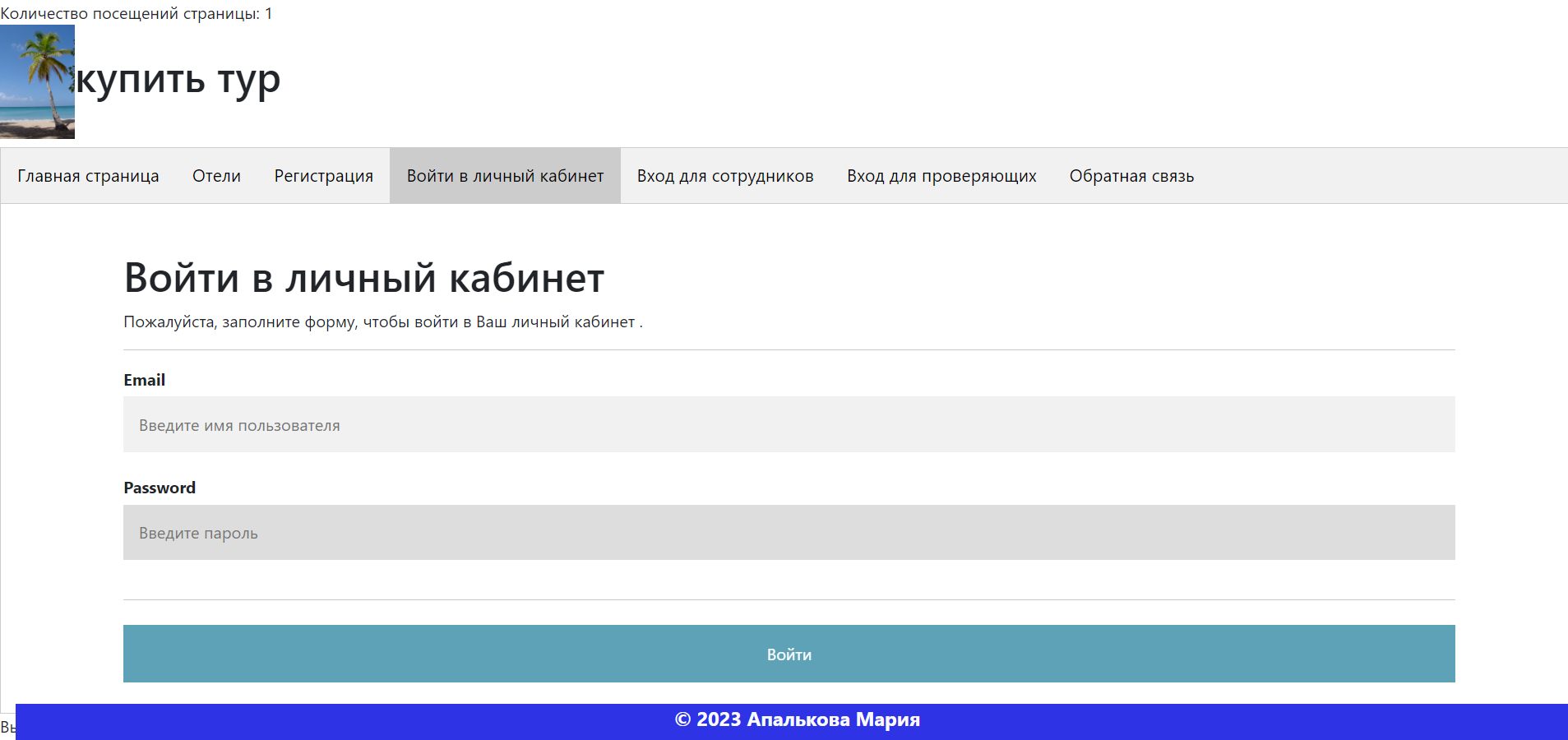


Рис. 3.5.15 – Демонстрация оформления

10. Был реализован случай одновременного запроса двух пользователей. Код данного случая представлен в файле change\_status.php. В нашем случае есть некая переменная aplication\_status, которая равняется 0,если еще никто не пытался изменить статус данной заявки. статус становится равным 0, когда первый, кто зашел на страницу нажимает “Обновить”. Всем последующим пользователям приходит ошибка, что статус занят. После завершения транзакции все будет происходить заново

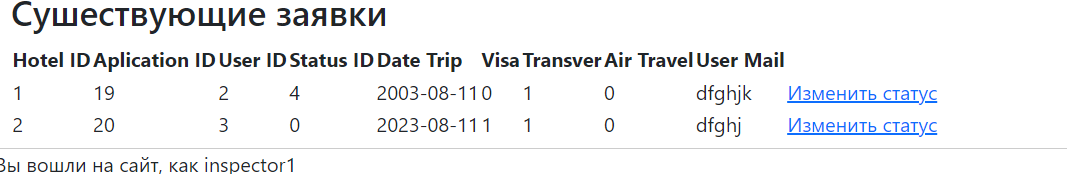


Рис 3.5.16- Изначальный статус заявки



3.5.17. - оба пользователя зашли в меню изменить статус

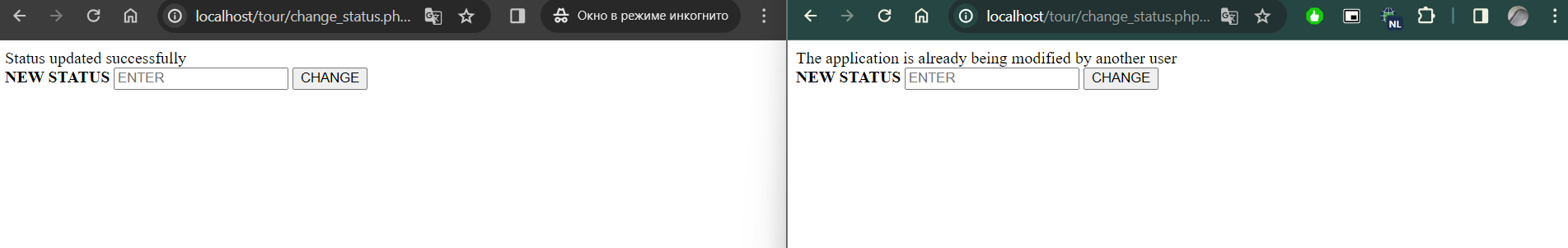


Рис 3.5.18 - Пользователь правее первым нажал изменить статус. второму заблокирован доступ для изменения

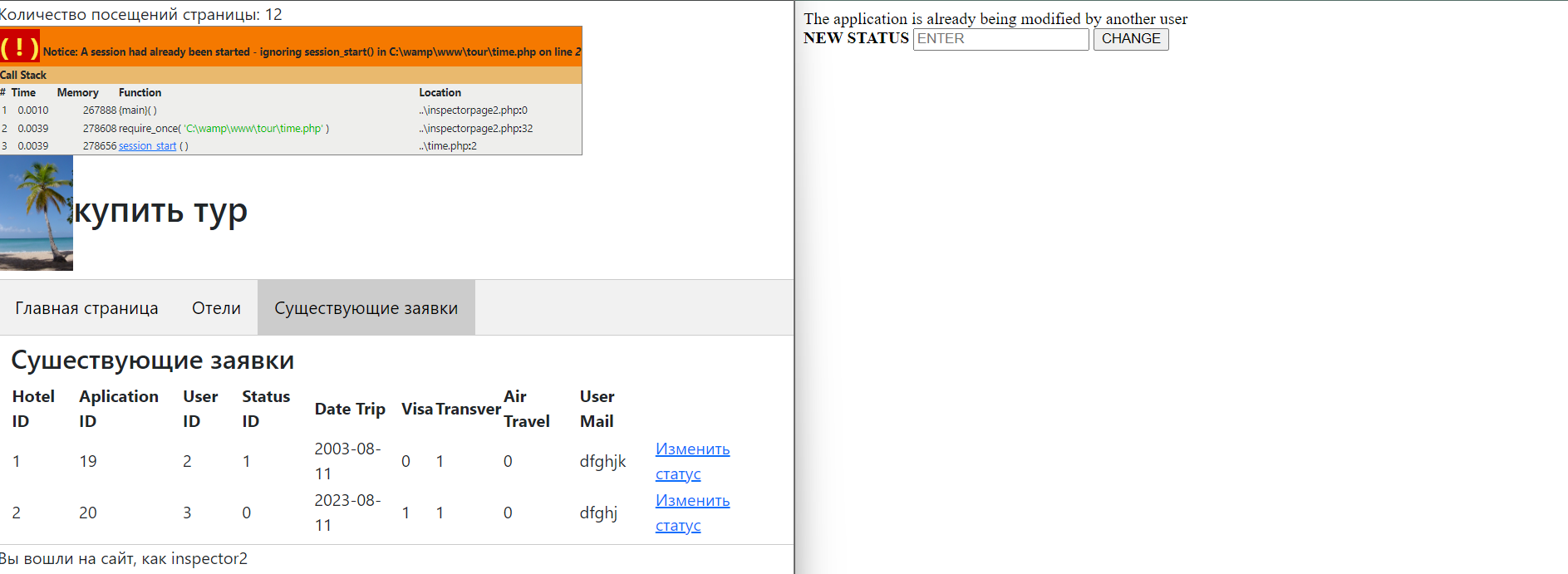
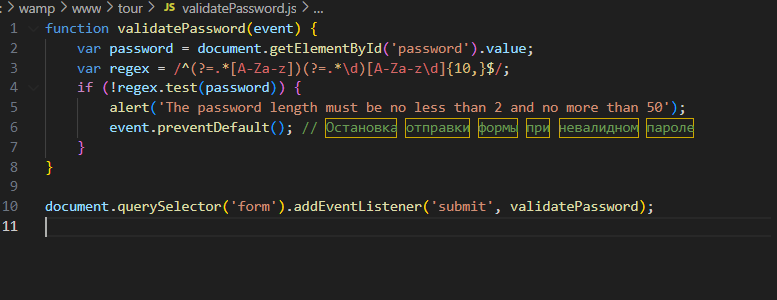


Рис 3.5.18 - Конечный статус после изменения

11.Была реализована защита от изменения кода страницы на стороне клиента, за счет дублирующей проверки на стороне сервера (на этапе регистрации). На этапе регистрации пользователя идет проверка на валидность данных, заводящихся в форму.

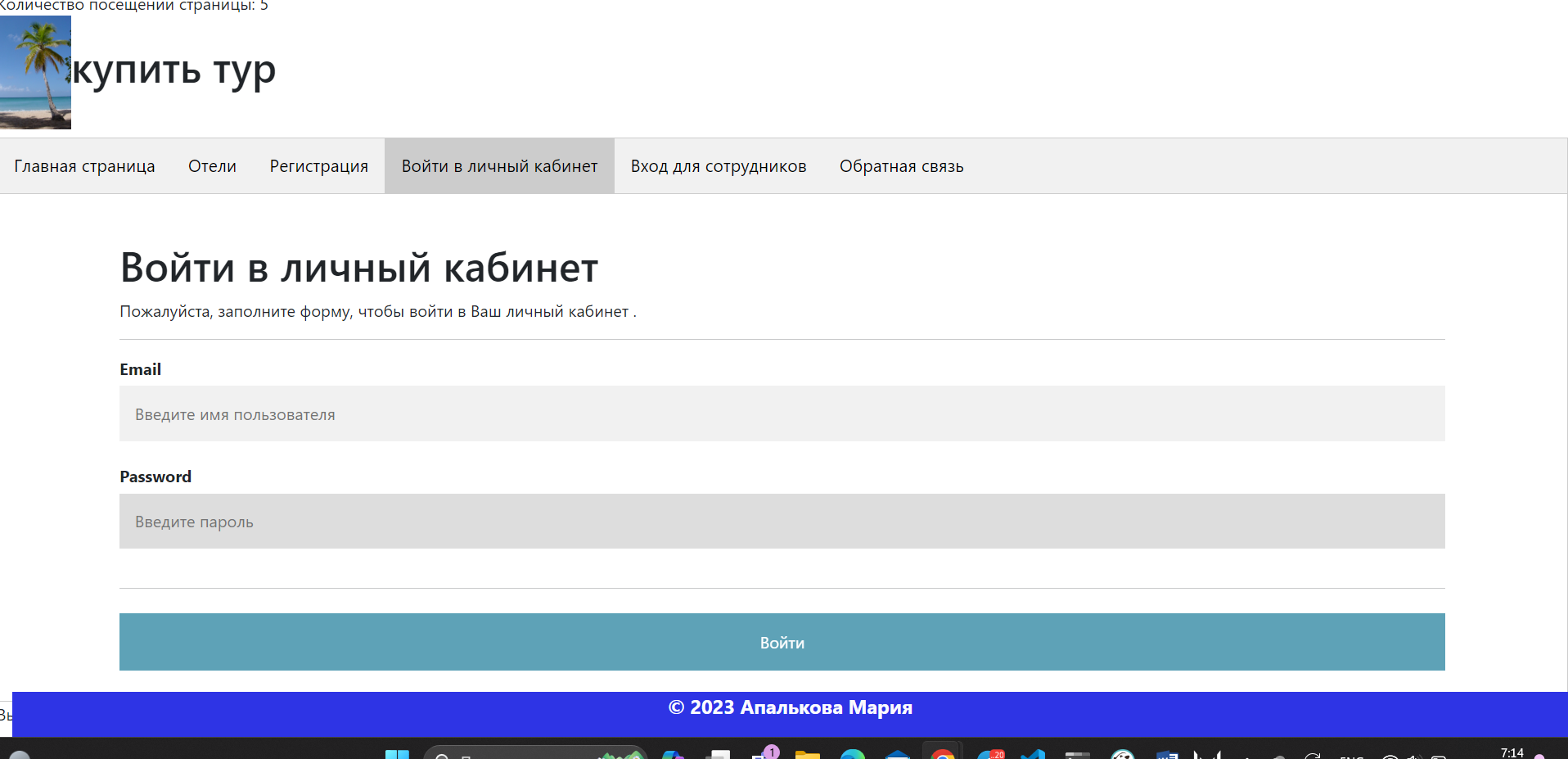
Но при этом на стороне сервера проверяется, что пароль введен в нужном формате, эта проверка происходит в файлах validatePassword\_worker.js, validatePassword.js

Рис. 3.5.19 – Дублирующая проверка пароля

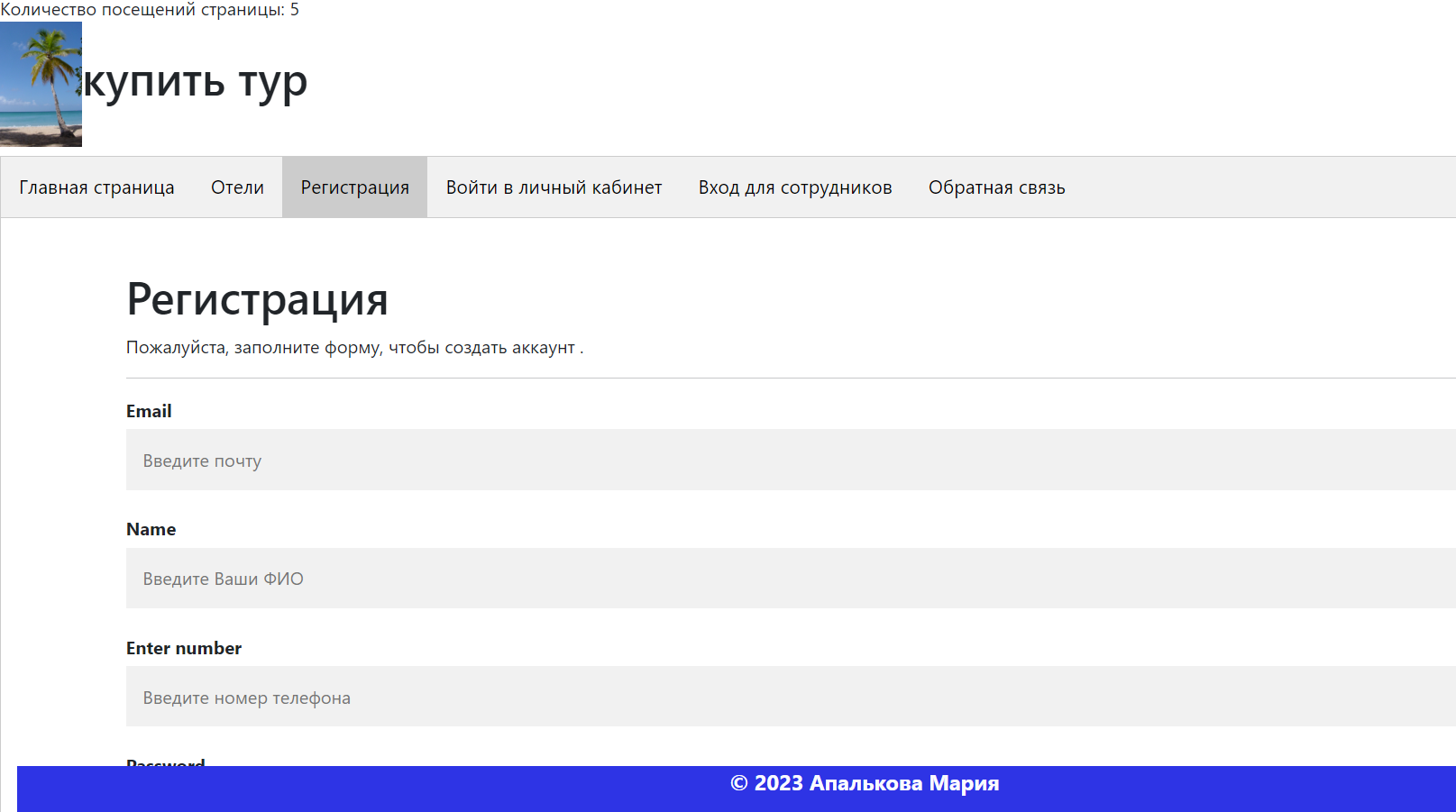
# 4. Демонстрация

## **4.1. Демонстрация функционала работы неавторизированного пользователя**

При входе незарегистрированный пользователь в системе видит главную страницу сайта. На ней можно посмотреть список отелей (Рис 4.1.4), которые предлагает турфирма, можно зарегистрироваться (рис 4.1.2) и войти в личный кабинет (это может сделать пользователь или сотрудник) (Рис 4.1.1)



4.1.1 - Вход в личный кабинет



4.1.2. - Форма регистрации

Если ввести почту, которая уже есть в системе, выведется ошибка, говорящая об этом пользователю

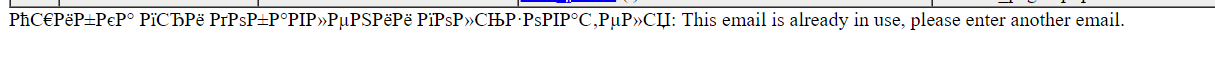


Рис 4.1.3 - Ошибка при неверном вводе почты(такая уже существует)

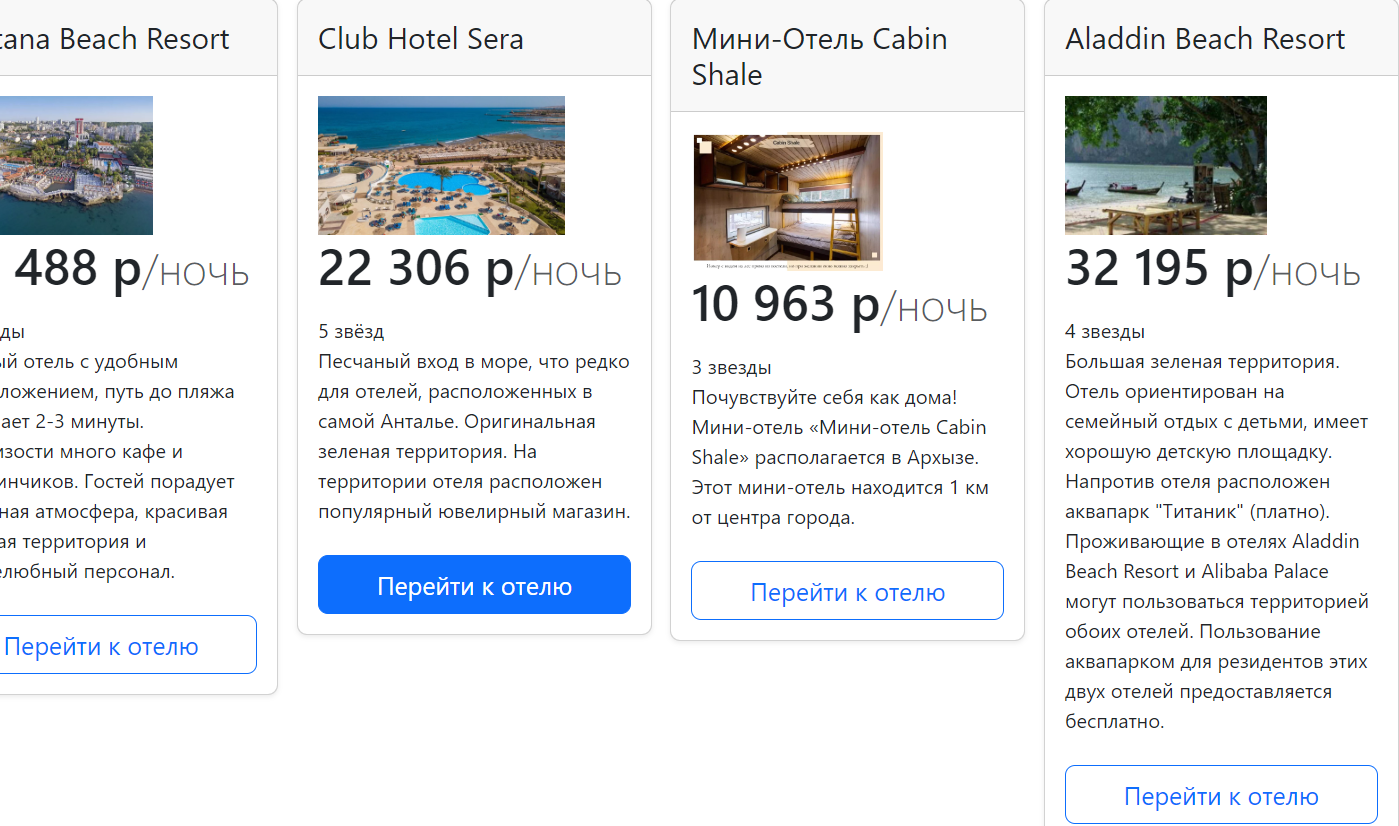


Рис 4.1.4 - Список предлагаемых отелей

Также пользователю показывается, сколько раз за один сеанс он посещал данную страницу (т. е. обновил сайт)

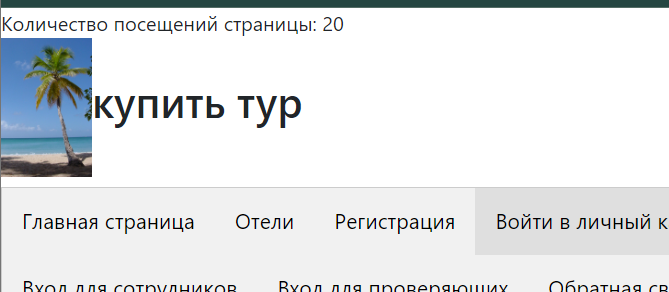


Рис. 4.1.5 – Сообщение о кол-ве посещений пользователей страницы

Если пользователь при регистрации вводит недопустимые по длине значения имени или номера телефона, программа сообщает об ошибке.( Рис 4.1.6, Рис 4.1.7)

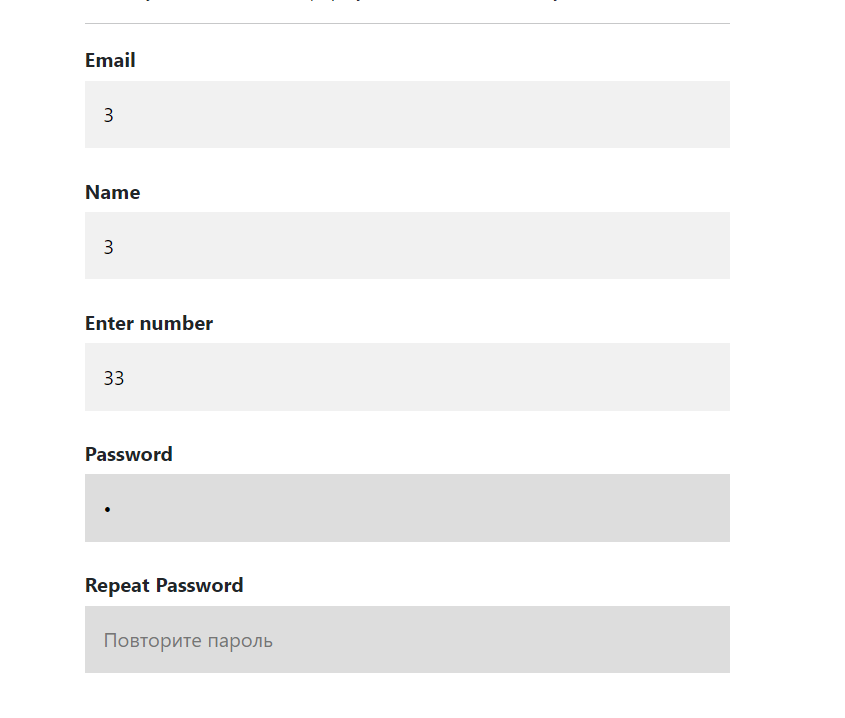


Рис 4.1.6 - Пример неправильно введенных данных

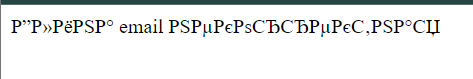


Рис. 4.1.7. – Предупреждение о некорректности введенного электронного адреса

Есть обязательные поля, не заполнив которые регистрация не будет осуществлена.

Также есть ошибка, которая выводится если не хватает каких-то данных при регистрации (Рис 4.1.8)

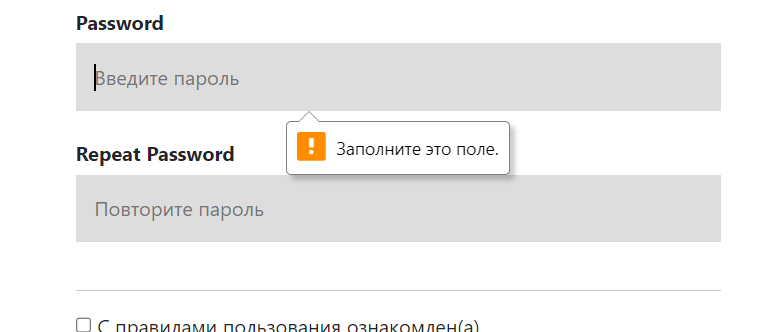


Рис 4.1.8 - Пример не заполнения нужных строк

Была реализована защита от изменения кода страницы на стороне клиента, за счет дублирующей проверки на стороне сервера (на этапе регистрации). На этапе регистрации пользователя идет проверка на валидность данных, заводящихся в форму.

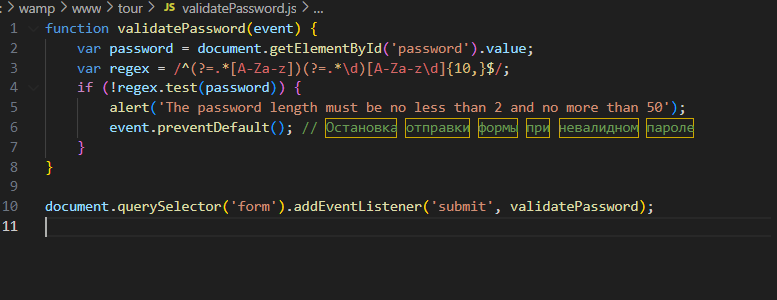


Рис. 3.5.19 – Дублирующая проверка пароля

## **4.2. Демонстрация функционала авторизированного пользователя**

При регистрации и последующем входе в личный кабинет зарегистрированный пользователь (покупатель) видит оповещение, о том что он вошел именно под своим логином (почтой) (Рис 4.2.1). Также зарегистрированный пользователь видит главную страницу сайта. Именно на ней можно просмотреть весь список отелей (Рис 4.2.2),



Рис 4.2.1 - Вход от имени пользователя

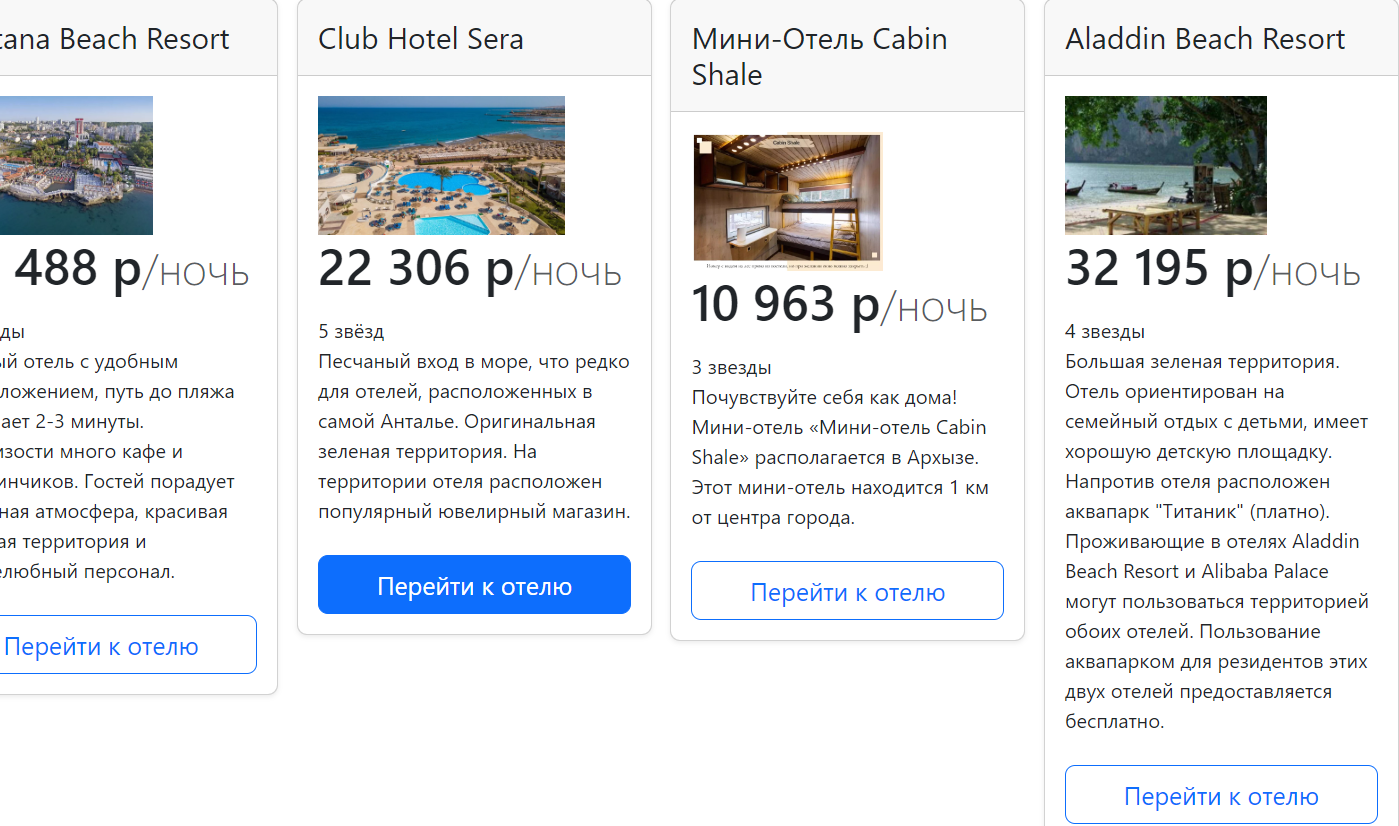


Рис 4.2.2 - Список предлагаемых отелей

Также пользователю имеет возможность просмотреть, сколько раз за один сеанс он посещал данную страницу (обновлял сайт).

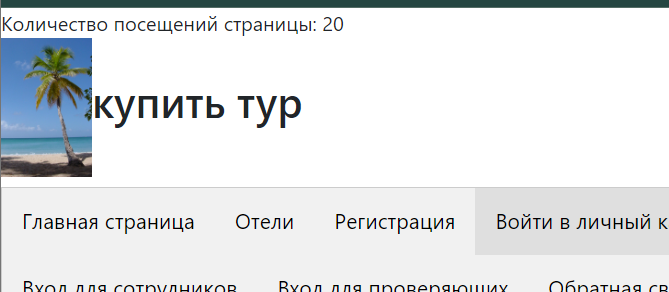


Рис. 4.2.3 – Сообщение о кол-ве посещений страницы

Во вкладке «Ваши заявки» пользователь может увидеть свой номер телефона

(Рис. 4.2.4-5)

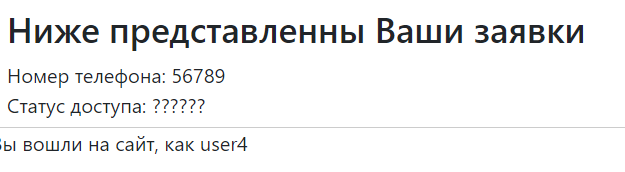
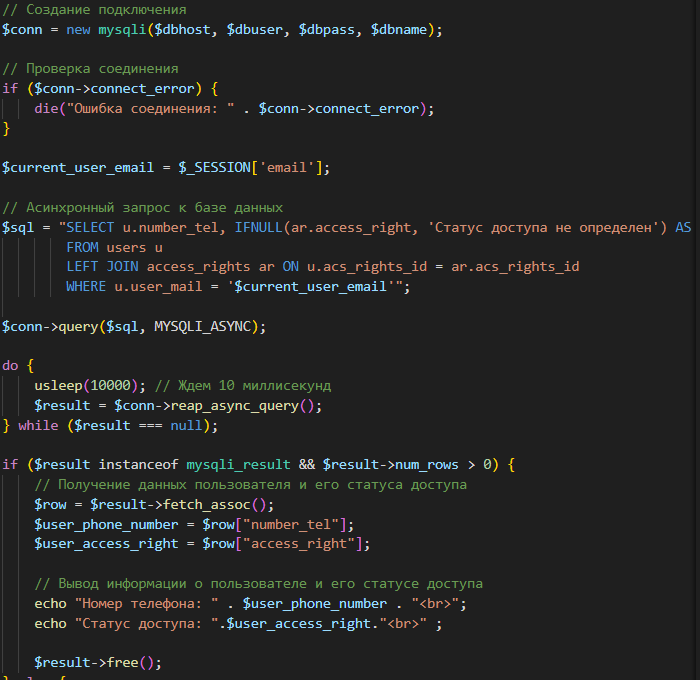


Рис 4.2.4 - Вывод номера телефона пользователя(ассинхронный запрос)



Рис 4.2.5 - Проверка выведенных данных

Рис 4.2.6 - Асинхронный запрос

Был реализован одновременный ввод информации в две таблицы (users , adress)(Рис 4.2.7., Рис 4.2.8)

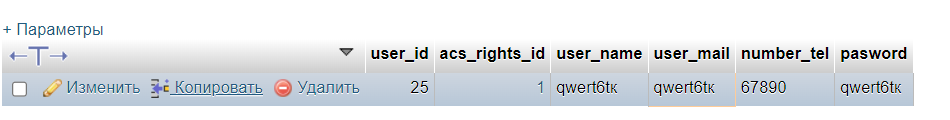
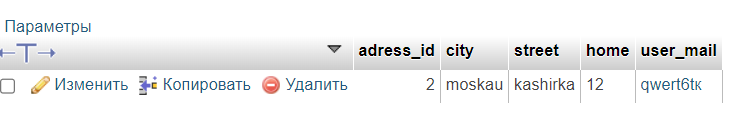


Рис 4.2.7. -ввод в таблицу юзеров



4.2.8 - Ввод в таблицу адреса

## **4.3. Демонстрация функционала сотрудника**

При входе в личный кабинет сотрудник видит оповещение, что он вошел именно под своей почтой (Рис 4.2.1)

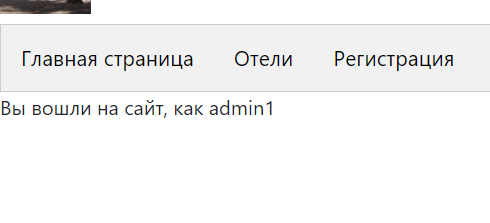


Рис 4.3.1 -Вход от имени пользователя

При входе сотруднику видит главную страницу, на которой имеется возможность посмотреть список отелей (Рис 4.3.2).

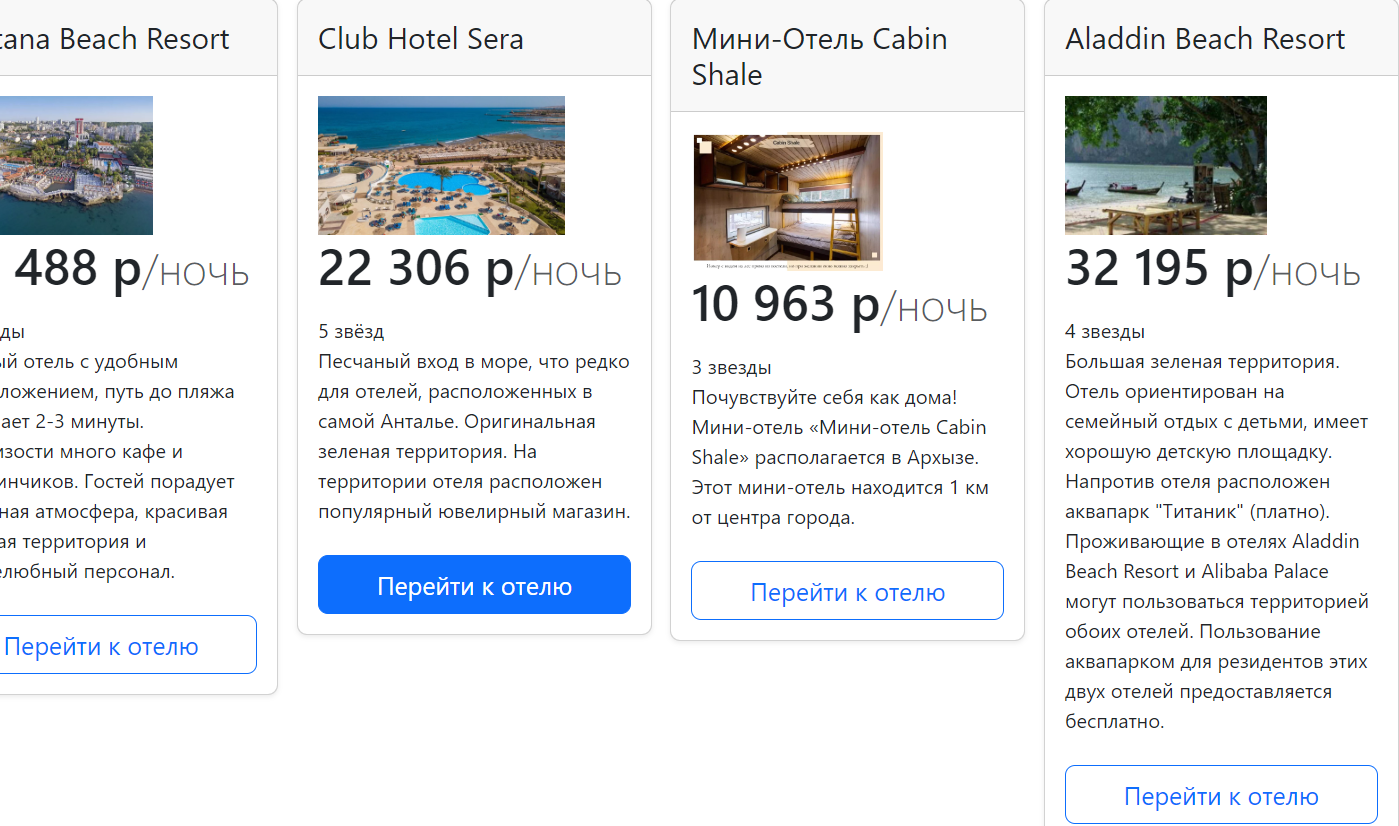


Рис 4.3.2 - список предлагаемых отелей

Так же пользователю видно, сколько раз за один сеанс он посетил данную страницу ( обновил сайт)

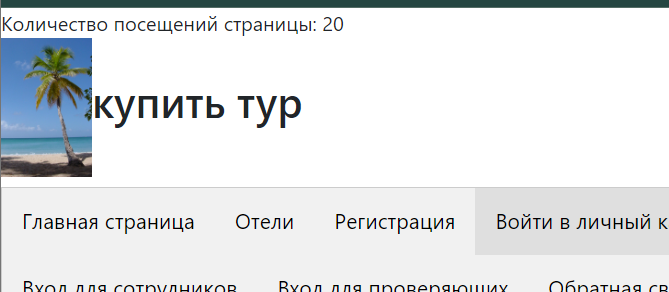


Рис. 4.3.3 – Сообщение о кол-ве посещений страницы

Сотрудник турфирмы обладает правами доступа, которые позволяют ему создавать новые заявки (Рис 4.3.4–5)

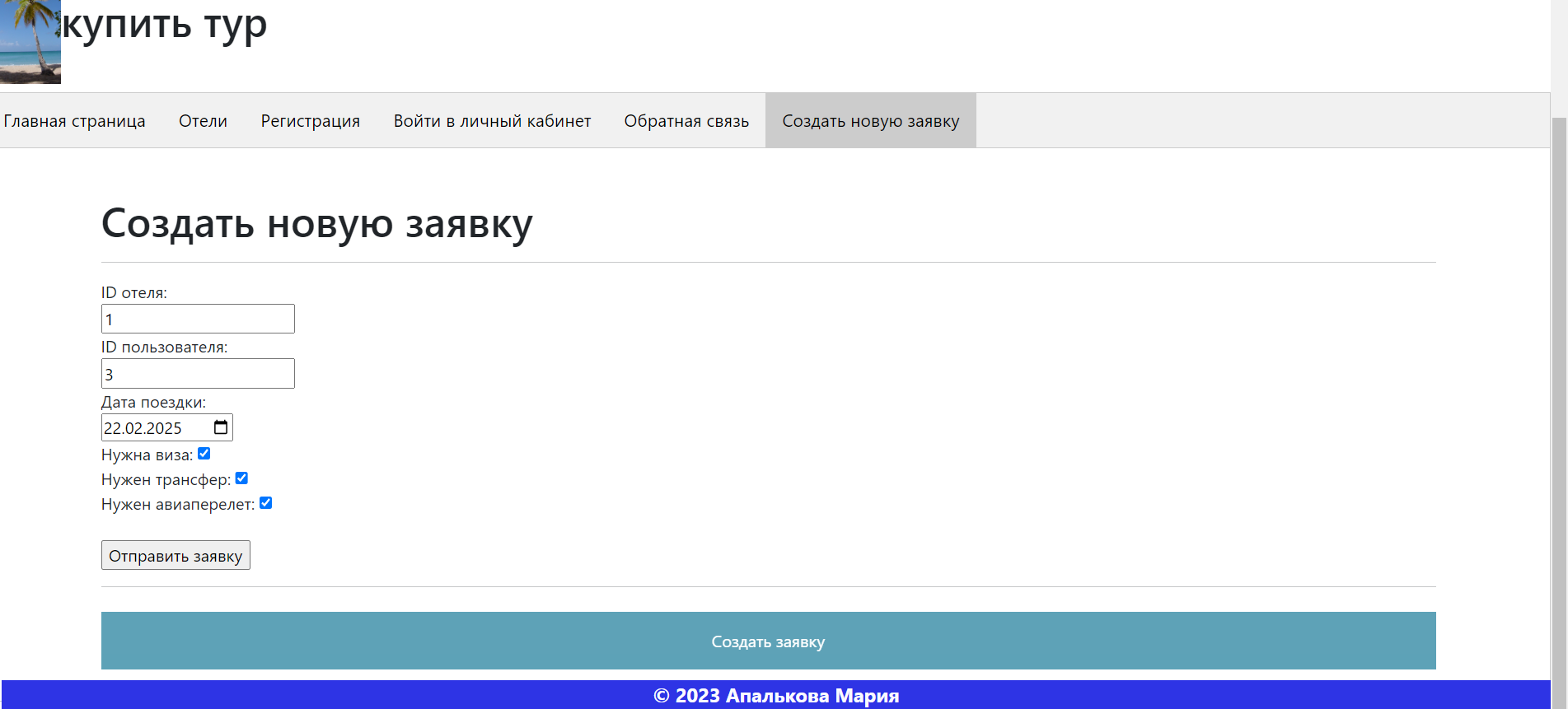


Рис 4.3.4 - Заполнение новой заявки



Рис 4.3.5 - Результату заполнения формы

Также была реализована функция регистрации нового клиента (Рис. 3.4.6)

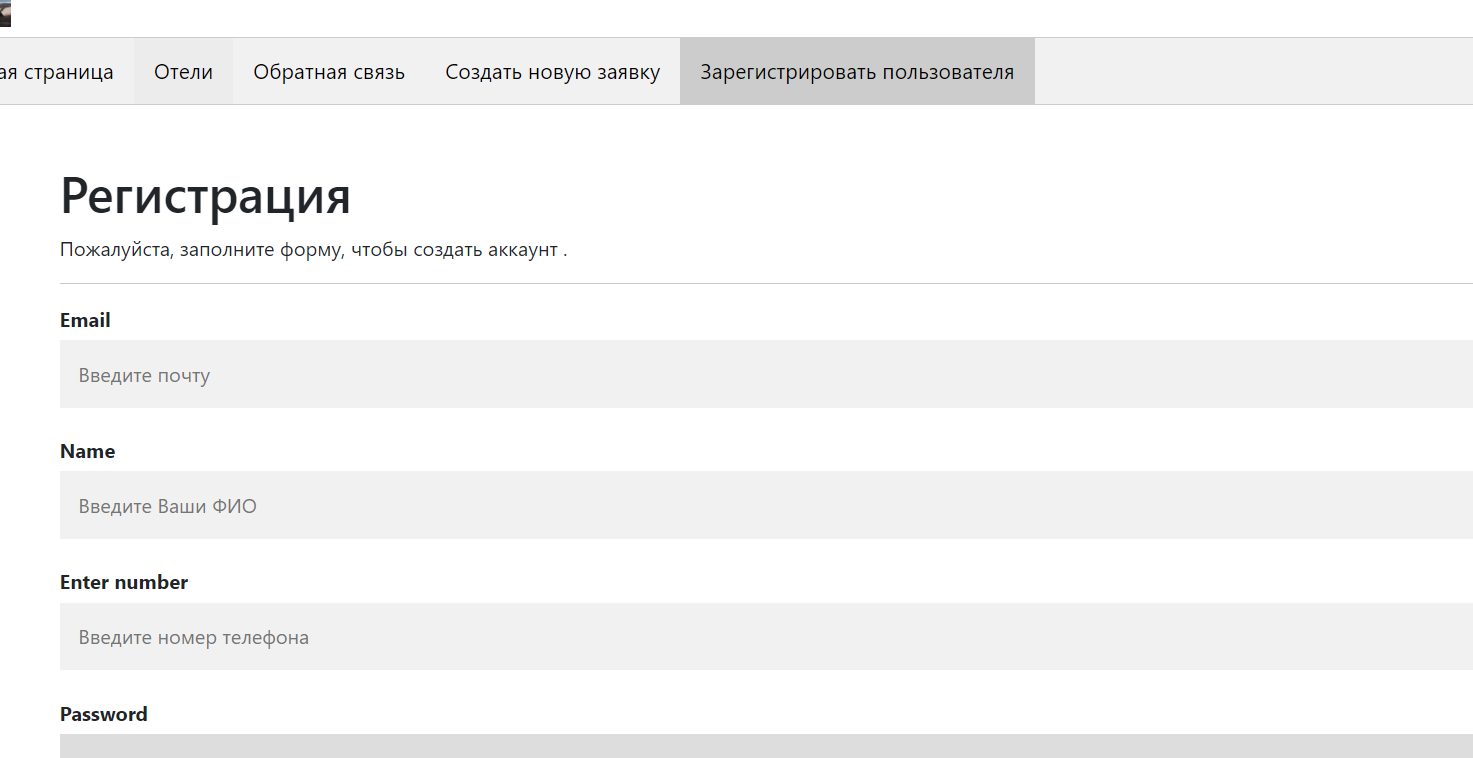


Рис 3.4.6. - регистрация сотрудником нового пользователя

Был реализован вывод из айди, почты и имени из таблицы users

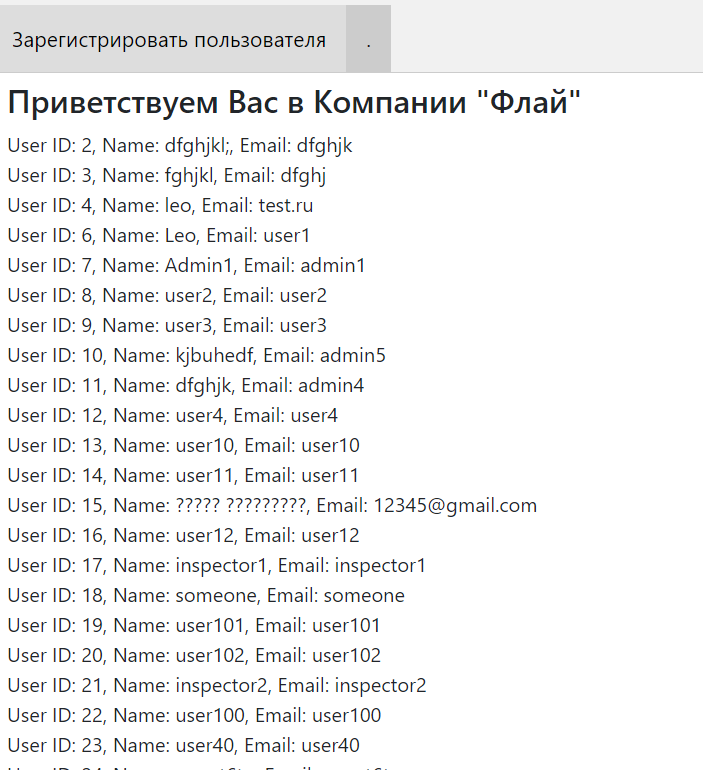


Рис 3.4.7. - вывод пользователей

## 4.4. **Демонстрация функционала проверяющего отелей**

При входе в личный кабинет сотрудник видит оповещение, что он вошел именно под своим именем(Рис 4.4.1)

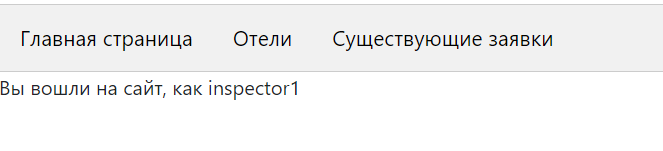
****

Рис 4.4.1 -Вход от имени пользователя

При входе сотруднику видит главную страницу, на которой имеется возможность посмотреть список отелей (Рис 4.3.2).

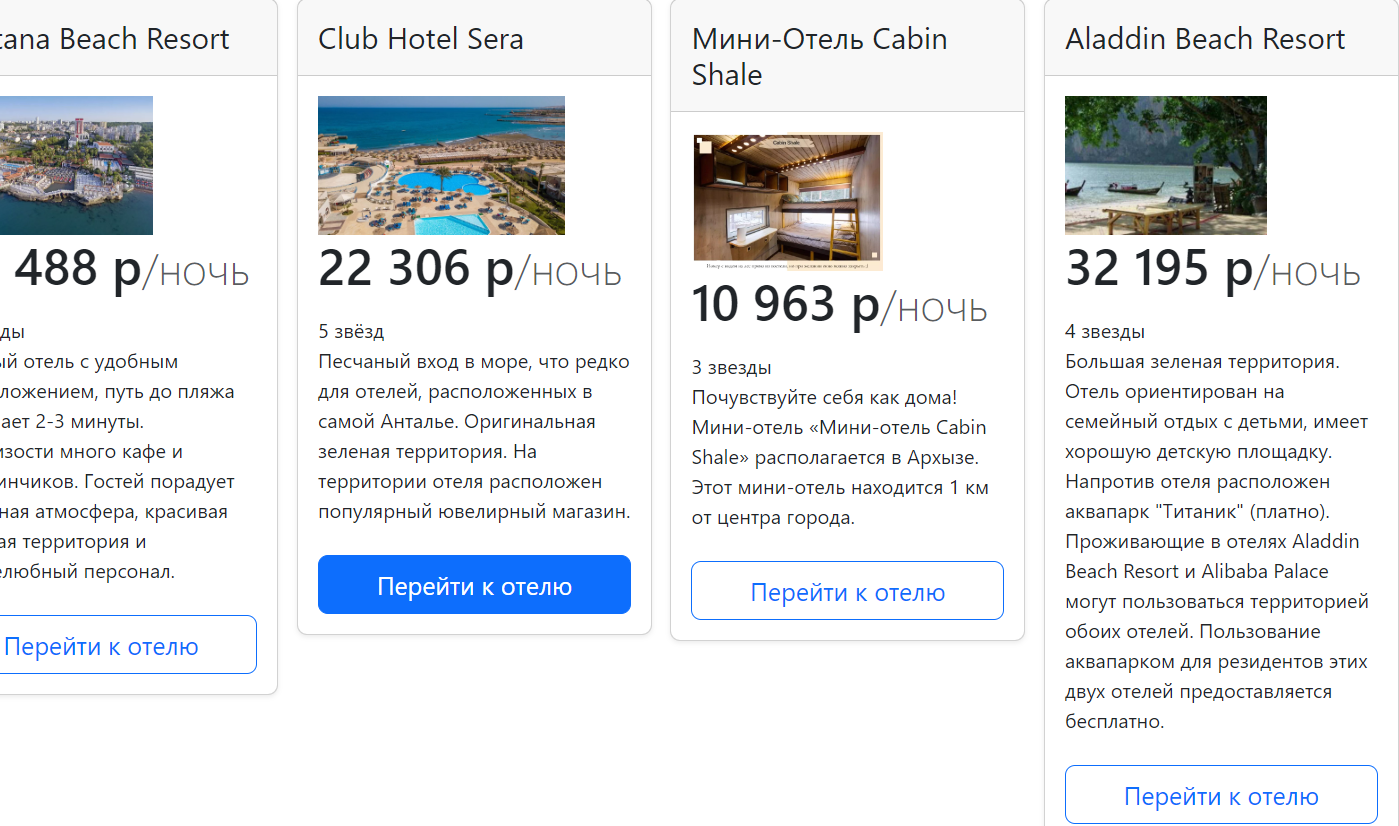


Рис 4.4.2 - список предлагаемых отелей

Так же пользователю видно, сколько раз за один сеанс он посетил данную страницу ( обновил сайт)

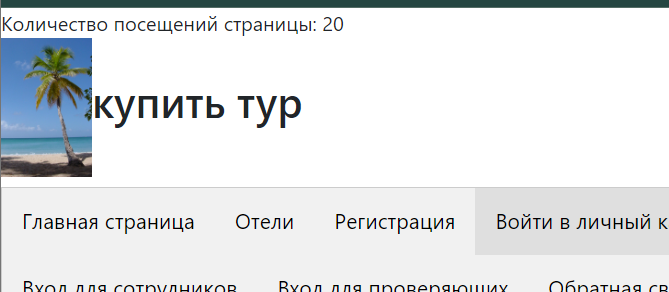


Рис. 4.4.3 – Сообщение о кол-ве посещений страницы

Данный тип доступа дает возможность посмотреть все заявки, которые есть в системе и при нажатии на кнопку “Изменить стаус” есть возможность для его изменения. в данной таблице дополнительно показывают почту узека, к которому прикреплена данная заявка (Рис 4.4.4, Рис 4.4.5, Рис 4.4.6)

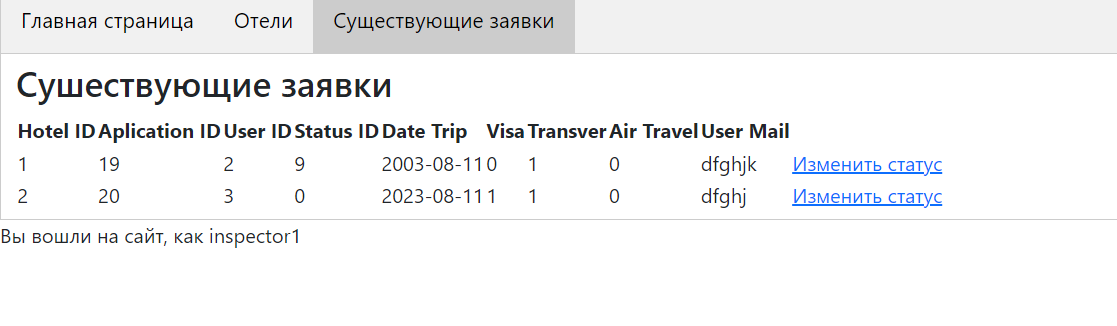


Рис 4.4.4 - Показ всех существующих заявок

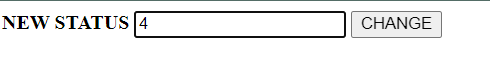


Рис 4.4.5 - запрос на изменение статуса

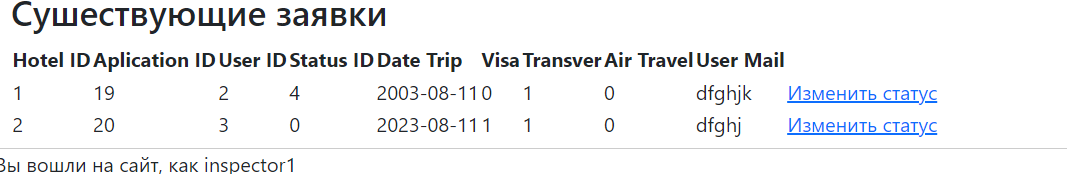


Рис 4.4.6 - результат выполнения запроса

Был реализован случай одновременного запроса двух пользователей.



Рис 4.4.7. - два пользователя одновременно хотят изменить статус

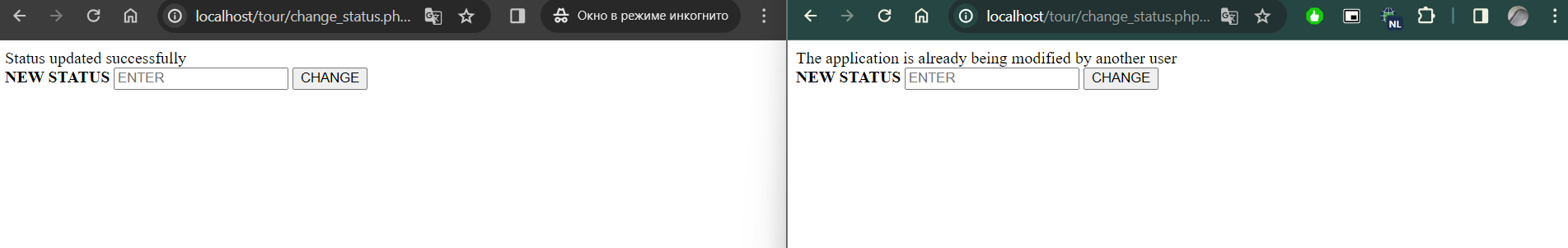


Рис 4.4.8. - Один пользователь изменяет статус, другому запрешено это сделать

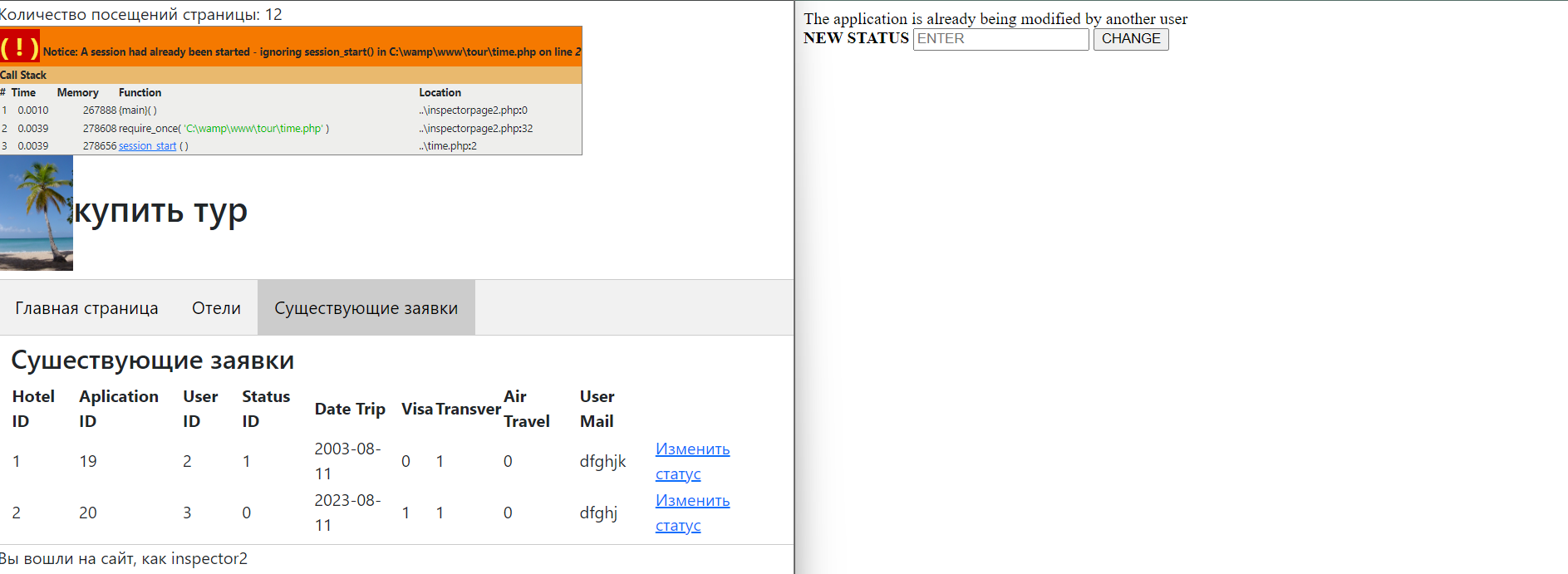


Рис 4.4.9. - результат изменения статуса