|  |  |
| --- | --- |
|

|  |
| --- |
| МИНОБРНАУКИ РОССИИФедеральное государственное бюджетное образовательное учреждениевысшего образования**«МИРЭА – Российский технологический университет»****РТУ МИРЭА** |

 |

Институт Информационных Технологий

Кафедра корпоративных информационных систем (КИС)

**ОТЧЕТ**

по Лабораторной работе №1

по дисциплине

«Разработка программного обеспечения для корпоративных информационных систем»

Выполнил студент группы ИКБО-08-18 Валяев Д.А.

Принял ассистент Масленников В.В.

Выполнено «\_\_\_» \_\_\_\_\_\_\_\_\_2021 г.

Зачтено «\_\_\_» \_\_\_\_\_\_\_\_\_2021 г.

**Задание**

На основе веб-сервера и сервера СУБД создать сайт типа «гостевая книга» с аутентификацией пользователей. Гостевая книга должна предусматривать регистрацию и аутентификацию пользователей, возможность аутентифицированному пользователю оставить сообщение и отображать все оставленные сообщения всем пользователям в виде списка. Дополнительные функции (редактирование и удаление сообщений, блокировка пользователей и пр.) реализуются по желанию. В клиентской части приложения запрещено использовать JavaScript. Наличие функций безопасности (предотвращение SQL Injection, экранирование спецсимволов, шифрование паролей) не требуется и реализуется по желанию.

**Содержание работы**

1. Реализация основного функционала
	1. Регистрация пользователей
	2. Авторизация пользователей
	3. Добавление сообщений
	4. Получение сообщений, написанных другими пользователями
2. Реализация дополнительного функционала
	1. Хэширование паролей
	2. Блокировка пользователей
	3. Редактирование и удаление сообщений

**Выполнение работы**

1. **Основной функционал**

 Поскольку в лабораторной работе разрешено использовать дополнительное ПО, я добавил поддержку Python в xampp.

Для начала была создана страница, на которой можно зарегистрироваться, авторизоваться, а также просмотреть сообщения пользователей. Авторизация и регистрация производятся через одну форму, но в случае с авторизацией ввод электронный почты не требуется. На страницах была добавлена стилизация к тегу <div> чтобы добавить отступ между элементами.



Рис 1.1 – Главная страница



Рис 1.2 – Код главной страницы

Процесс авторизации/регистрации происходит следующим образом:

* Со страницы отправляется POST запрос на authorization.py;
* Из базы данных выбирается пользователь с введенным логином;
* Если пользователь найден, то проверяется введенный пароль;
* Если пароль подошел, пользователь увидит логин, под которым авторизован в верхней части страницы (рис 1.3), а также на сайте появятся Cookies (рис 1.4);
* Если пароль не подошел будет выдано сообщение «Wrong password»;
* В случае, когда пользователя нет в базе данных, он будет зарегистрирован.



Рис 1.3 – страница авторизованного пользователя



Рис 1.4 – Cookie сайта

 

Рис 1.5 – Обработчик авторизации/регистрации

На рисунке 1.5 можно увидеть код обработчика авторизации и регистрации, под пунктом 1 мы получаем данные, отправленные в POST запросе, в пункте 2 происходит проверка если пользователя нет в базе данных, то он добавляется, пункт 3 выполняется, если пользователь уже был зарегистрирован и пароль не прошел проверку, иначе вызывается пункт 4, который устанавливает cookie и перенаправляет на главную страницу.

После авторизации пользователь может добавить свое сообщение, чтобы это сделать необходимо ввести текст в поле ввода и нажать кнопку «Add message»



Рис 1.6 – пример добавления сообщения

1. **Дополнительный функционал**

Кроме возможности авторизации и добавления записи были реализованы дополнительные функции, такие как:

* 1. Хранение паролей не в открытом виде с использование библиотеки brypt, при регистрации пароль хешируется с использованием соли (рис 1.8), а при авторизации введенные пароль сравнивается с хешем из базы данных (рис 1.9).



Рис 1.7 – формат хранения пароля в базе данных



Рис 1.8 – хеширование пароля



Рис 1.9 – сравнение введенного пароля с хешем из базы данных, где password – введенный пароль, а db\_data[0] хеш, прочитанный из базы данных

* 1. Блокировка пользователей – если пользователь администратор, тогда у него на главной странице будет кнопка «Ban» рядом с сообщениями других пользователей, при нажатии на кнопку, вызывается ban.py, где проверяется пользователь и если он является администратором, то будет выведено сообщение «User name been banned». Как видит страницу обычный пользователь изображено на рисунках 1.3 и 1.4, а так видит страницу администратор



Рис 1.10 – как видит страницу администратор



Рис 1.11 – ban.py

На рисунке 1.11 можно видеть:

1. – Получение данных пользователя, чтобы проверить является ли он администратором;
2. – Получение имени пользователя, которого необходимо заблокировать;
3. – Подключение к базе данных;
4. – Получение хеша пароля администратора;
5. – Проверка пароля администратора и выдача блокировки.
	1. Также есть возможность удалять и редактировать свои сообщения, при выводе сообщений на страницу происходит проверка, если пользователь и автор сообщения одинаковые, то к сообщению добавляются 2 кнопки «Delete» и «Edit»



Рис 1.12 – добавление кнопок к сообщениям

 1 – подключение к базе данных;

 2 – выполнение запроса для получения всех сообщений, отсортированных по id;

 3 – проверка является ли пользователь автором сообщения, если да, добавляется форма для POST запроса с кнопками «Delete» и «Edit»;

 4 – если пользователь является администратором, то у него появляется кнопка «Ban»;

 5 – отображение записи для других пользователей.

 При нажатии на кнопку «Edit» пользователя перенаправляет на страницу редактирования, где уже предзаполнено поле текстом исходного сообщения.



Рис 1.13 – редактирование сообщения



Рис 1.14 – код страницы обновления сообщения

 После изменения текста необходимо нажать кнопку «Update», после чего произойдет проверка пароля пользователя, редактирование сообщения и перенаправление на главную страницу.



Рис 1.15 – отредактированное сообщение



Рис 1.16 – код редактирования сообщения

 Как и в прошлых случаях сначала мы получаем данные владельца записи и сравниваем хеш пароля владельца с паролем текущего пользователя, если пароль подошел, обновляем запись:

1 – Получение данных текущего пользователя;

2 – Получение текста сообщения и id из тела запроса для обновления;

3 – Подключение к базе данных;

4 – Получение пароля владельца сообщения;

5 – Происходит обновление сообщения с id, который был передан в тебе запроса;

6 – Перенаправление пользователя на главную страницу.

Нажатие кнопки «Delete» вызывает delete.py, код схож с тем, что был в обновлении записи, единственное обличие – это запрос базе данных



Рис 1.17 – запрос для удаления сообщения



Рис 1.18 – таблица с сообщениями



Рис 1.19 – таблица пользователей

**Вывод**

 В данной лабораторной работе были приобретены навыки работы с СУБД, созданием сайта гостевой книги, работы с веб-сервером Apache, а также работы с html