МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Ижевский государственный технический университет имени М. Т. Калашникова»

Факультет «Информатика и вычислительная техника»

Кафедра «Программное обеспечение»

Работа защищена с оценкой

«\_\_\_\_»

Дата \_\_\_\_\_\_\_\_\_\_

Подпись \_\_\_\_\_\_\_/\_\_\_\_\_\_\_

ОТЧЕТ ПО ПРЕДДИПЛОМНОЙ ПРАКТИКЕ

«Система управления учебным процессом. Серверная часть»

Выполнил:

Студент группы Б08-191-2 М.Л. Поздеев

Руководитель:

к.т.н., доцент кафедры ПО И.О. Архипов

Рецензия:

степень достижения поставленной цели работы\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

полнота разработки темы\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

уровень самостоятельности работы обучающегося\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

недостатки работы\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

СОДЕРЖАНИЕ

[2. РАЗРАБОТКА ЗАДАЧ ПО СЕРВЕРНОЙ ЧАСТИ СИСТЕМЫ УПРАВЛЕНИЯ УЧЕБНЫМ ПРОЦЕССОМ 3](#_Toc40817875)

[2.1. Разработка сервиса авторизации 3](#_Toc40817876)

[2.1.1. Описание постановки задачи 3](#_Toc40817877)

[2.1.2. Описание алгоритма регистрации пользователей 4](#_Toc40817878)

2. РАЗРАБОТКА ЗАДАЧ ПО СЕРВЕРНОЙ ЧАСТИ СИСТЕМЫ УПРАВЛЕНИЯ УЧЕБНЫМ ПРОЦЕССОМ

2.1. Разработка сервиса авторизации

2.1.1. Описание постановки задачи

2.1.1.1. Характеристика задачи

Задача – разработать сервис авторизации.

Сервер авторизации разрабатывается с целью обеспечения безопасности данных, хранимых в системе. Для реализации возможности масштабирования системы в будущем на большое количество университетов – было решено разработать собственный сервер авторизации, который будет предоставлять доступ мобильным приложениям пользователей к API системы.

Разработка сервера авторизации связана с разработкой сервера API системы. С помощью проверки доступа к методам API будет проверяться работоспособность сервера авторизации.

Разработка задачи будет завершена при успешном прохождении тестирования сервера.

2.1.1.2. Входная информация

Перечень входных данных при регистрации пользователей:

1. номер зачетной книжки (в случае, если пользователь – студент);
2. идентификатор преподавателя в системе (в случае, если пользователь – преподаватель);
3. уникальный логин пользователя;
4. пароль.

Перечень входных данных при аутентификации и авторизации пользователей:

1. логин пользователя, созданный при регистрации;
2. пароль пользователя, созданный при регистрации.

Перечень входных данных при валидации токена доступа:

1. токен доступа.

Перечень входных данных при обновлении токена доступа:

1. токен восстановления доступа;
2. старый токен доступа.

Вся вышеперечисленная информация передается на сервер с помощью http запросов.

2.1.1.3. Выходная информация

Перечень выходных данных при регистрации пользователей:

1. сообщение об успешной или о неудачной регистрации.

Перечень выходных данных при аутентификации и авторизации пользователей:

1. токен доступа;
2. токен восстановления доступа;
3. идентификатор пользователя в системе.

Перечень выходных данных при валидации токена доступа:

1. сообщение об успешной или неудачной валидации.

Перечень выходных данных при обновлении токена доступа:

1. новый токен доступа при успешном обновлении токена;
2. новый токен восстановления доступа при успешном обновлении токена;
3. идентификатор пользователя в системе при успешном обновлении токена;
4. ошибка обновления токена при не правильных входных данных.

Вся вышеперечисленная информация выдается сервером в ответ на http запросы.

2.1.2. Описание алгоритма регистрации пользователей

2.1.2.1. Назначение и характеристика

Регистрация необходима для создания пользователя в системе и его сопоставления с сущностью студента или преподавателя в базе данных.

Алгоритм должен сохранять данные пользователей в базе данных, при этом проверив существование этого пользователя как студента или преподавателя в системе. Пароли пользователей должны записываться в базу в виде хеша таким образом, чтобы одинаковые пароли имели разное представление в базе данных. Полученный хеш должен быть устойчивым к атакам перебора паролей.

2.1.2.2. Используемая информация

Для алгоритма регистрации используются:

1. номер зачетной книжки студента или идентификатор преподавателя в системе;
2. уникальный логин пользователя;
3. пароль пользователя.

2.1.2.3. Результаты решения

В результате выполнения алгоритма формируется следующая информация:

1. оповещение клиентского приложения об успешной или неудачной регистрации;
2. сохранение регистрационных данных пользователя (логин, пароль в виде хеша, идентификатор пользователя) в базе данных.

2.1.2.4. Алгоритм решения

Схема алгоритма:



Описание алгоритма:

1. сервер авторизации получает регистрационные данные от клиентского приложения;
2. сервер авторизации сопоставляет полученный номер зачетной книжки или идентификатор преподавателя с данными в базе;
3. если такого студента или преподавателя не существует, то сервер посылает на клиентское приложение ответ с сообщением об ошибке регистрации и алгоритм завершается;
4. если такой студент или преподаватель существует в университете, то алгоритм выполняется дальше;
5. сервер авторизации генерирует «соль» для хеширования пароля – случайный текст, который добавится к введенному паролю для обеспечения уникальности хешей при введении пользователями одинаковых паролей;
6. сервер генерирует хеш пароля и добавлением «соли»;
7. сервер сохраняет регистрационные данные в базу данных пользователей системы;
8. сервер посылает на клиент ответ с сообщением об успешной регистрации и алгоритм завершается.

2.1.3. Описание алгоритма аутентификации и авторизации пользователей

2.1.3.1. Назначение и характеристика

Аутентификация необходима для подтверждения идентичности пользователя. Авторизация необходима для предоставления пользователю прав доступа на необходимые ресурсы.

Алгоритм должен сверять данные пользователя (логин и пароль), пришедшие с клиентского приложения с данными в базе данных, и если все указано верно, то алгоритм должен предоставить доступ пользователю клиентского приложения с помощью токена доступа к необходимым для него ресурсам. Время действия токена должно быть ограничено, для обеспечения безопасности. Алгоритм также должен предоставить метод обновления токена доступа с помощью токена восстановления доступа, для того чтобы токен доступа можно было восстановить без введения логина и пароля пользователя. В добавок к токенам, алгоритм должен отправлять на клиентское приложение идентификатор пользователя в системе.

2.1.3.2. Используемая информация

При первой аутентификации используются:

1. логин пользователя;
2. пароль пользователя.

При наличии токена доступа для аутентификации и авторизации используется этот токен доступа.

При обновлении токена доступа используются:

1. токен доступа;
2. токен восстановления доступа.

2.1.3.3. Результаты решения

В результате выполнения алгоритма при первой аутентификации и при обновлении токена доступа формируется следующая информация:

1. токен доступа;
2. токен восстановления доступа;
3. идентификатор пользователя в системе.

Токен восстановления доступа сохраняется в базе данных пользователей. Все сформированные данные отправляются на клиентское приложение.

При наличии актуального токена – пользователю предоставляется доступ к необходимым ресурсам.

2.1.3.4. Алгоритм решения

Описание алгоритма при первой аутентификации пользователя:

1. сервер получает данные для аутентификации с клиентского приложения (логин и пароль);
2. сервер проверяет введенные данные;
3. если данные введены неверно, то на клиентское приложение возвращается ошибка;
4. если данные введены верно, то сервер генерирует токен доступа и токен восстановления доступа;
5. сервер сохраняет токен восстановления доступа в базе данных;
6. сервер отправляет на клиент токен доступа, токен восстановления доступа и идентификатор пользователя в системе.

Описание алгоритма при наличии токена доступа:

1. сервер получает токен доступа;
2. сервер проверяет валидность полученного токена доступа;
3. если токен не валиден, то на клиентское приложение отправляется ошибка авторизации с информацией, показывающей, почему токен не валиден (если время действия токена истекло);
4. если токен валиден, то сервер предоставляет пользователю доступ к запрашиваемому ресурсу.

Описание алгоритма при восстановлении токена доступа:

1. сервер получает токен доступа и токен восстановления доступа;
2. сервер проверяет валидность токена доступа (опуская проверку времени действия токена);
3. если токен доступа не валиден, то сервер отправляет на клиентское приложение ошибку с сообщением о неправильном токене доступа;
4. сервер проверяет валидность токена восстановления доступа;
5. если токен восстановления доступа не валиден, то сервер отправляет на клиентское приложение ошибку с сообщением о неправильном токене восстановления доступа;
6. если полученные токены валидны, то сервер генерирует токен доступа и токен восстановления доступа;
7. сервер сохраняет токен восстановления доступа в базе данных;
8. сервер отправляет на клиент токен доступа, токен восстановления доступа и идентификатор пользователя в системе.

2.2. Разработка сервиса ресурсов

2.2.1. Описание постановки задачи

2.2.1.1. Характеристика задачи

Задача – разработка сервиса ресурсов.

2.2.1.2. Входная информация

2.2.1.3. Выходная информация

2.2.2. Описание алгоритма предоставления данных для приложения преподавателя

2.2.2.1. Назначение и характеристика

2.2.2.2. Используемая информация

2.2.2.3. Результаты решения

2.2.2.4. Алгоритм решения

2.2.3. Описание алгоритма предоставления данных для приложения студента

2.2.2.1. Назначение и характеристика

2.2.2.2. Используемая информация

2.2.2.3. Результаты решения

2.2.2.4. Алгоритм решения

2.2.3. Описание контрольного примера

2.2.3.1. Назначение

2.2.3.2. Исходные данные

2.2.3.4. Результаты расчета

2.2.3.5. Результаты испытания программы