## Лабораторная работа №2.

На основе лабораторной работы №1 реализовать механизм авторизации и регистрации пользователей через API.

**Начиная с этой лабораторной работы, для всех последующих вводятся дополнительные обязательные требования:**

1. **Для реализации функциональных возможностей согласно требованиям запрещается использовать программные библиотеки, не входящие в состав первоначально установленного фреймворка;**
2. **Использовать программные библиотеки из состава фреймворка разрешается;**
3. **Если необходимый функционал нельзя реализовать, ограничиваясь возможностями фреймворка, такой функционал реализуется самостоятельно;**
4. **Код, использующий сторонние библиотеки не будет принят при защите лабораторной работы, если вы не сможете его объяснить.**

Требования к выполнению работы.

1. Создать от главной ветки новую с наименованием «lb2».
2. Переключиться на новую ветку.
3. Опубликовать новую ветку.
4. Создать 7 маршрутов.
5. Создать 2 класса запросов для авторизации и регистрации пользователей.
6. Создать 3 DTO класса ресурсов для авторизации, регистрации и получения пользователей.
7. В созданные классы запросов добавить метод возвращающий экземпляр соответствующего ресурса.
8. Создать класс контроллер с методами для созданных ранее роутов.
9. Методы авторизации и регистрации получают в качестве входного параметра экземпляры соответствующих классов запросов.
10. Метод регистрации в случае успеха возвращает экземпляр ресурса созданного пользователя со статусом 201. В противном случае возвращается сообщение об ошибке с описанием проблемы.
11. Метод авторизации в случае успеха возвращает токен доступа и токен обновления со статусом 200. В противном случае возвращается сообщение об ошибке с описанием проблемы. Токен обновления реализуется по желанию.
12. Метод получения информации об авторизованном пользователе в случае успеха возвращает экземпляр ресурса авторизованного пользователя через DTO.
13. Метод разлогирования отзывает используемый токен доступа и выданный токен обновления, запрещая их дальнейшее использование.
14. Метод получение списка авторизованных токенов в случае успеха возвращает массив с списком токенов. Требования к оформлению списка токенов не предъявляется. Метод возвращает только токены авторизованного пользователя.
15. Метод отзыва всех действующих токенов доступа в случае успеха возвращает сообщение со статусом 200. Метод отзывает только токены авторизованного пользователя.
16. При передаче токена обновления в refresh метод, в случае, если он уже был использован, система сообщает об этом пользователю, и отзывает все его действующие токены. Если токен обновления не вызывает проблем, то пользователь получает новую пару токенов. Использованный токен обновления утрачивает силу.
17. Количество активных токенов у пользователя не может превышать количество, заданное переменной в файле среды окружения.
18. Время жизни токенов задается через конфигурацию или переменные в файле среды окружения.
19. Обработка маршрутов выполняется через методы созданного контроллера.
20. Обеспечить защиту токена на случай перехвата. Требование к способу реализации не предъявляется. Требование не является обязательным к выполнению.
21. Обеспечить пользователю возможность изменять пароль.
22. Изменение пароля пользователя осуществляется с подтверждением действующего пароля.
23. Обеспечить использование токена обновления.
24. **Токеном является** строка, сгенерированная исключительно для авторизующегося пользователя. Это **не случайно сгенерированная строка**. Токен передает пользователю как минимум время жизни, и идентификатор пользователя.
25. Токены **нельзя** явно хранить в базе данных.
26. Все методы кроме login/register выполняются только авторизованными пользователями. Подтверждение авторизации осуществляется передачей в каждом запросе токена доступа.
27. Токен обновления передается на сервер, только при вызове метода refresh.
28. Зафиксировать изменения и опубликовать их в ветку «lb2».

Требования к роутам:

|  |  |  |  |
| --- | --- | --- | --- |
| Операция | URL | Метод | Открытость |
| Авторизация | api/auth/login | POST | Да |
| Регистрация | api/auth/register | POST | Только неавторизованные |
| Получение информации об авторизованном пользователе | api/auth/me | GET | Только авторизованные |
| Разлогирование | api/auth/out | POST | Только авторизованные |
| Получение списка авторизованных токенов пользователя | api/auth/tokens | GET | Только авторизованные |
| Разлогирование всех действующих токенов пользователя | api/auth/out\_all | POST | Только авторизованные |

Требования к классам запросов:  
LoginRequest:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Param | Description | Type | Require | Validate | Comment |
| username | Имя пользователя | String | True | 1. Содержит только буквы латинского алфавита.  2. Начинается с большой буквы.  3. Минимальная длина 7 символов |  |
| password | Пароль пользователя | String | True | 1. Минимальная длина 8 символов.  2. Содержит не менее 1 цифры.  3. Содержит не менее 1 символа.  4. Содержит не менее чем по 1 символу в верхнем и нижнем регистре. |  |

RegisterRequest:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Param | Description | Type | Require | Validate | Comment |
| username | Имя пользователя | String | True | 1. Содержит только буквы латинского алфавита.  2. Начинается с большой буквы.  3. Минимальная длина 7 символов.  4. Свойство уникальное в таблице пользователей (Без учета регистра). |  |
| email | Email пользователя | String | True | 1. Email корректный.  2. Email уникален в таблице пользователей. |  |
| password | Пароль пользователя | String | True | 1. Минимальная длина 8 символов.  2. Содержит не менее 1 цифры.  3. Содержит не менее 1 символа.  4. Содержит не менее чем по 1 символу в верхнем и нижнем регистре. |  |
| c\_password | Подтверждение пароля | String | True | Соответствует полю password |  |
| birthday | Дата рождения | Date | True | Передается в формате: 2000-12-31  Возраст пользователя на момент регистрации должен быть не менее 14 лет. |  |

Критерии приемки:

|  |  |  |  |
| --- | --- | --- | --- |
| № | Критерий приемки | Обязательность | Баллы за выполнение критерия |
| 1 | Все требования выполнены. Все требуемые методы реализованы согласно заданию, работают корректно и без ошибок. | Да | 30 |
| 2 | Код чистый, типизированный и оптимизированный. Файлы не содержат закомментированные участки кода. Наименования используемых методов и переменных интуитивно понятны. Все используемые методы дополнены комментариями. Выполняется принцип SOLID. Файлы не содержат не используемых методов и переменных. | Да | 10 |
| 3 | Механизм генерации, выдачи, валидации и отзыва токенов доступа реализован самостоятельно. | Да | 15 |
| 4 | Используемые маршруты сгруппированы по префиксу | Да | 2 |
| 5 | Реализована проверка контроля доступа к методам | Да | 6 |
| 6 | Реализована проверка количества активных токенов доступа | Нет | 6 |
| 7 | Все методы вызываются соответствующими методами запросов | Да | 4 |
| 8 | Все методы возвращают корректные статусы | Нет | 2 |
| 9 | Реализован механизм обновления токена доступа | Нет | 8 |
| 10 | Токены защищены от перехвата | Нет | 6 |
| 10 | Данные из методов возвращаются пользователям в формате JSON. | Да | 4 |
| 11 | Код покрыт тестами. | Нет | 8 |
| 12 | Выполнение всех требований в работе с git | Нет | 4 |
| Всего | | | 105 |
| Необходимо для сдачи лабораторной | | | 70 |

Штрафы.

|  |  |  |
| --- | --- | --- |
| № | Штраф | Баллы за нарушение |
| 1 | Метод не реализован. | 5 |
| 2 | Метод работает не корректно или не работает. | 3 |
| 3 | Метод вызывается http запросом неверного типа. | 1 |
| 4 | Код грязный. | 2 |
| 5 | Нарушена или не указана типизация переменных или методов (обсуждаемо). | 1 |
| 6 | Код не оптимизирован (обсуждаемо). | 1 |
| 7 | Изменяемые содержат закомментированные участки кода. | 2 |
| 8 | Наименование используемых методов и переменных не интуитивно понятны (Обсуждаемо). | 1 |
| 9 | Нарушение требований CodeStyle. | 1 |
| 10 | Нарушение требований SOLID. | 2 |
| 11 | Неправильная работа с git. | 2 |
| 12 | Файлы содержат не используемые методы или переменные. | 1 |
| 13 | Реализованные методы не содержат комментариев. | 1 |

Штрафы назначаются за каждое обнаруженное нарушение.

Например, если код содержит 2 метода без комментария, то штраф будет в двойном размере.