Описание API проекта WEB\_project.

Сергей Жолудев

2024

Оглавление

[1. Основная информация 2](#_Toc172487810)

[1.1 История изменений документа 2](#_Toc172487811)

[2. Sigin. Процесс аутентификации в системе. 3](#_Toc172487812)

[2.1 Запрос/Ответ 3](#_Toc172487813)

[2.2 Входные параметры 3](#_Toc172487814)

[2.3 Выходные параметры: 4](#_Toc172487815)

[2.4 Проверки 4](#_Toc172487816)

[2.5 Описание интеграции 4](#_Toc172487817)

[3. SigUP. Процесс регистрации в системе. 5](#_Toc172487818)

[3.1 Запрос/ответ. 5](#_Toc172487819)

[3.2 Входные параметры. 6](#_Toc172487820)

[3.3 Выходные параметры 7](#_Toc172487821)

[3.4 Проверки 7](#_Toc172487822)

[3.5 Описание интеграции 7](#_Toc172487823)

[4. Events. Главная страница с перечнем всех мероприятий 8](#_Toc172487824)

[4.1 Запрос/ответ. 8](#_Toc172487825)

[4.2 Входные параметры 8](#_Toc172487826)

[4.3 Выходные параметры 8](#_Toc172487827)

[4.4 Проверки 8](#_Toc172487828)

[4.5 Описание интеграции 9](#_Toc172487829)

# 1. Основная информация

|  |  |
| --- | --- |
| **Автор** | Сергей Жолудев |
| **Задача** | Обеспечить аутентификацию пользователя в системе. |
| **Бизнес постановка** | Пользователь при входе в систему вводит пароль и логин, в ответ получает сообщение об успешной аутентификации или об отказе входа в систему. |
| **Связанные документы** | Ссылка на Jira должна быть здесь.  Сам проект: <https://github.com/Burchenkov/web_project>  Коды HTTP сообщений:  https://developer.mozilla.org/ru/docs/Web/HTTP/Status |

## 1.1 История изменений документа

|  |  |  |  |
| --- | --- | --- | --- |
| **Версия** | **Дата** | **Автор** | **Описание изменений** |
| 1.0 | 14.07.2024 | Сергей Жолудев | Первая версия документа |
| 1.1 | 21.07.2024 | Сергей Жолудев | Добавлено описание процедуры регистрации signUp, получение списка мероприятий Events |
| 1.2 | 25.07.2024 | Сергей Жолудев | Изменены скрипты для проверки, теперь используется HTTPie |

# 2. Sigin. Процесс аутентификации в системе.

На WEB-сайте пользователь вводит пароль и логин, данные передаются на Backend. Базовый код проверяет есть ли пользователь в системе, правильный ли пароль указан. Если результат проверки успешен, пользователь входит в систему. Если введён неправильный логин – пользователь получает сообщение: «Нет такого пользователя, сначала зарегистрируйтесь.» Если введён неправильный логин, сообщение: «Неправильный пароль».

## 2.1 Запрос/Ответ

Пример запроса и ответа для добавления информации о животном на сайте

|  |  |
| --- | --- |
| **Запрос** | **Ответ** |
| http POST :8000/signin login: “MasterOfTheUniverse” password : "exapleqwrety1234" | Response code: 200, 400  Response body:  For 200: “OK”  For 400: “HTTP\_400\_BAD\_REQUEST”  body of message:  “Status: bad email”  Or  “The password is incorrect”. |

## 2.2 Входные параметры

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Параметр** | **Тип данных** | **Обязательность** | **Описание** | **Варианты значений** |
| 1 | Login | Str | Да | Логин пользователя | MasterOfTheUniverse |
| 2 | Passwrod | Str | Да | Пароль для входа в систему. | SXM8V6WLR |

## 2.3 Выходные параметры:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Параметр** | **Тип данных** | **Обязательность** | **Описание** | **Варианты значений** |
| 1 | HTTP response | Str | Да | Ответ системы об успешности аутентфификации | “200” |
| 2 | HTTP response | Str | Да | Ответ системы об неуспешности аутентфификации, с уточнением что конкретно не так. | "400” |

## 2.4 Проверки

|  |  |
| --- | --- |
| **Параметр** | **Проверка** |
| email | * Логин, в нашем случае аутентифицируемся с помощью email, должен быть в формате [xxxxxxx@xxx.xxx](mailto:xxxxxxx@xxx.xxx). * Логин не должен превыщать N символов. * Пароль не должен превышать N симолов. |

## 2.5 Описание интеграции

Изображение выглядит как текст, снимок экрана, диаграмма, число

Автоматически созданное описание

# 3. SigUP. Процесс регистрации в системе.

Если пользователь не зарегистрирован в системе, он нажимает кнопку «Зарегистрироваться». Перед ним появляется модальное окно, куда он вводит данные:

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Данные через POST запрос направляются на сервер backend. Происходит проверка, есть ли такой login, есть ли такой email. Если есть – возвращается сообщение об ошибки с информационным сообщением. В случае уникальности предоставленных данных, добавляем данным аккаунт(экземплят класс User в БД).

## 3.1 Запрос/ответ.

Успешный:

|  |  |
| --- | --- |
| **Запрос** | **Ответ** |
| http POST :8000/signup/ email="someone@example.com" login="MegaUser" role="user" password= "123456" | Response code: 201 “HTTP\_201\_CREATED” |

Не успешный:

|  |  |
| --- | --- |
| **Запрос** | **Ответ** |
| http POST :8000/signup/ email="someone@example.com" login="MegaUser" role="user" password= "123456" | Response code: 400 Bad Request  Response body:  ‘{  “message” : “login is already exist”  }’  Or  ‘{  “message” : “email is already exist”  }’ |

## 3.2 Входные параметры.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Параметр** | **Тип данных** | **Обязательность** | **Описание** | **Варианты значений** |
| 1 | email | str | да | Email пользователя | some@example.com |
| 2 | login | str | да | login пользователя | MasterOfTheUniverse |
| 3 | role | str | да | роль пользователя в системе. От Frontend всегда user. | user |
| 7 | password | str | да | пароль пользователя | SXM8V6WLR |

## 3.3 Выходные параметры

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Параметр** | **Тип данных** | **Обязательность** | **Описание** | **Варианты значений** |
| 1 | HTTP response | Str | Да | Ответ системы об успешности регистрации пользователя | HTTP\_201\_CREATED |
| 3 | HTTP response | Str | Да | Ответ системы об неуспешности регистрации | 400 |
| 4 | Body of response | Str | Да | Ответ системы, что конкретно не так логин или email уже существуют | ‘{  “message” : “login is already exist”  }’ |

## 3.4 Проверки

При регистрации со стороны backend должна выполняться проверка на уникальность email и login. Делаем запрос в БД, если предоставленные данные не уникальны – возвращаем HTTP\_400\_BAD\_REQUEST, в теле сообщения уточнение, что именно неправильно.

## 3.5 Описание интеграции

Изображение выглядит как текст, снимок экрана, диаграмма, Параллельный

Автоматически созданное описание

# 4. Events. Главная страница с перечнем всех мероприятий

При входе пользователь получает список всех мероприятий.

## 4.1 Запрос/ответ.

|  |  |
| --- | --- |
| **Запрос** | **Ответ** |
| http :8000/events/ | Response code: HTTP\_200\_OK  Response body:  ‘{  "pk":3,  "title":"Box fight",  "category":"Sport",  "date":"2024/01/01 19:00:00",  "location":"Moscow",  "description":"this is a fight of the century"  }’ |

## 4.2 Входные параметры

Входных параметров нет, просто отправляем запрос GET на получение списка Events.

## 4.3 Выходные параметры

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Параметр** | **Тип данных** | **Обязательность** | **Описание** | **Варианты значений** |
| 1 | HTTP response | Str | Да | Ответ системы об успешности | 200 |
| 2 | Body of response | JSON | Да | Тело ответа системы | ‘{  "pk":3,  "title":"Box fight",  "category":"Sport","date":"2024-07-30T19:00:00Z",  "location":"Moscow",  "description":"this is a fight of the century"  }’ |

## 4.4 Проверки

## 4.5 Описание интеграции

Изображение выглядит как текст, снимок экрана, диаграмма, Параллельный

Автоматически созданное описание