

EventUp

REST API Documentation

Author(s):
Cory Lewis
Vincent Santos

Updated: 5/6/19

Endpoints(s)

1. User Routes

- 1.1. users/Login - POST
- 1.2. users/Register - PUT
- 1.3. users/ - GET
- 1.4. users/RSVP - POST
- 1.5. users/RSVP - DELETE
- 1.6. users/RSVP/:EventId(\\d+) - GET
- 1.7. users/posts - POST

2. Event Routes

- 2.1. events/ - GET
- 2.2. events/ - POST
- 2.3. events/:id - DELETE
- 2.4. events/:id - GET
- 2.5. events/filter/:id - GET

3. Message Routes (1 to 1 relationship)

- 3.1. messages/send - POST
- 3.2. messages/:EventId - GET

4. Map Routes **[Not Implemented on Back-End]**

- 4.1. maps/log - POST
- 4.2. maps/log - GET

4.3. maps/log/:Limit - GET

5. Category

5.1. categories/ - GET

6. Location

6.1. locations/ - GET

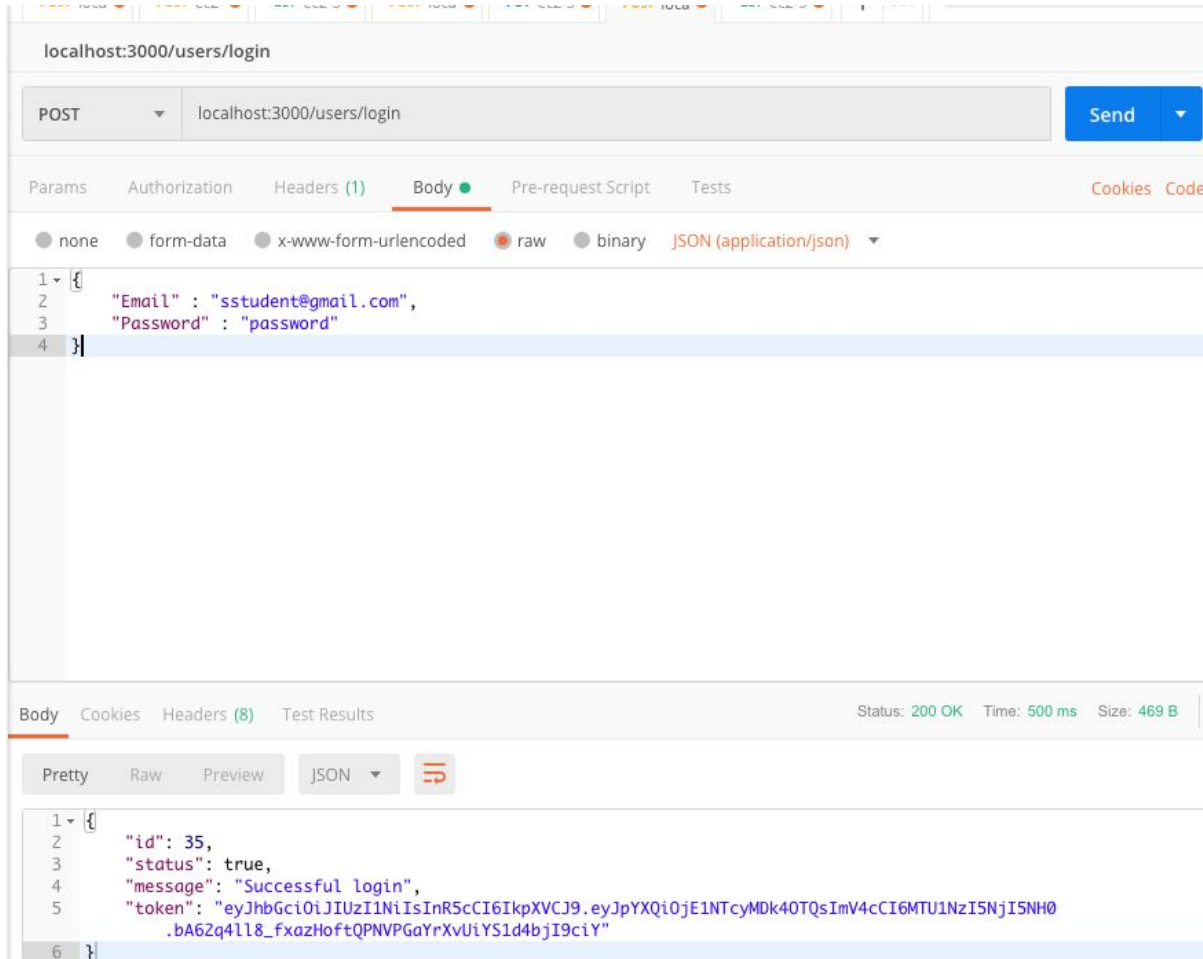
1.1 /users/login

- Description: Login a user that has already registered. You receive a token and that is used to gain access to other routes
- Request:
 - endpoint: /users/login
 - method="POST"
 - Body
 - Raw -> application/json

```
{  
    "Email" : "sstudent@gmail.com",  
    "Password" : "password"  
}
```
- Response:
 - *(int)* **id**: id of user in db
 - *(bool)* **status**: successful login (*true == yes, false == no*)
 - *(string)* **message**: information about your response
 - *(string)* **token**: token used for header in requests

```
{  
    "id": #,  
    "status": true/false,  
    "message": "Successful Login",  
    "token": "HASHED_TOKEN_USED_FOR_LOGIN"  
}
```

- Sample Screenshot:



1.2 /users/register

- Description: registers a user into the db and receive a token.
- Request:
 -
 - endpoint: /users/register
 - method="PUT"
 - Body
 - Raw -> application/json

```
{  
    "FirstName": "Generic",  
    "LastName": "User",  
    "Email": "gu@gmail.com",  
    "Password": "password"  
}
```
- Response:
 - *(int)* **id**: id of user in db
 - *(bool)* **status**: successful (*true == yes, false == no*)
 - *(string)* **message**: information about your response
 - *(string)* **token**: token used for header in requests

```
{  
    "id": #,  
    "status": true/false,  
    "message": "Successful login",  
    "token": "HASHED_TOKEN_USED_FOR_LOGIN"  
}
```

- Sample Screenshot:

The screenshot displays a REST client interface with the following details:

- URL:** localhost:3000/users/register
- Method:** PUT
- Body:** A JSON object with the following fields:

```
{  "FirstName": "Generic",  "LastName": "User",  "Email": "gu@gmail.com",  "Password": "password"}
```
- Response:** A 200 OK status with the following JSON body:

```
{  "status": true,  "id": 43,  "message": "Registered Successfully",  "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6NDMsIm1hdCI6MTU1NzIxMTE2MiwiZXhwIjoxNTU3Mjk3NTYyYyQ.KU9rbS_6dfcCvGzvdLAQayTlsf8AVxIkX_TYQPem6o0"}
```
- Response Metadata:** Status: 200 OK, Time: 131 ms, Size: 487 B

1.3 /users/getUsers

- Description: This route returns all users in db (*for testing*)
- Request:

- Description: Returns all users in db.
- endpoint: /users/
- method="GET"
- Headers

Key	Value
Authorization	"USER_TOKEN"
Content-Type	application/json

- Body
 - none
- Response:
 - (*bool*) **status**: successful (*true == yes, false == no*)
 - (*string*) **message**: information about your response
 - (*int*) **UserCount**: amount of users queried
 - (*Object Array*) **users**: array of users

```
{  
  "status": true/false,  
  "UserCount": #,  
  "users": [  
    {  
      "id": #,  
      "FirstName": "...",  
      "LastName": "...",  
      "Email": "...@domain.com",  
      "Password": "...",  
      "Image": URL_PATH
```


}
...
}

Sample screenshot:

The screenshot shows a REST client interface with the following details:

- URL:** `ec2-54-183-219-162.us-west-1.compute.amazonaws.com:3000/events/`
- Method:** GET
- Headers:**
 - Authorization: `eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6ND...`
 - Content-Type: `application/json`
- Status:** 200 OK, Time: 57 ms, Size: 10.56 KB
- Response Body (JSON):**

```
1 {  
2   "status": true,  
3   "data": [  
4     {  
5       "id": 1,  
6       "Name": "Soccer",  
7       "Description": "Some bois coming to play soccer",  
8       "AgeRestriction": 0,  
9       "UserId": 1,  
10      "CategoryId": 1,  
11      "LocationId": 1,  
12      "Image": "ec2-54-183-219-162.us-west-1.compute.amazonaws.com:3000/uploads/2019-05-04T01-42-20.Z5Z2RaveEvent.jpg",  
13      "StartDate": "2019-05-20T00:00:00.000Z",  
14      "StartTime": "12:00:00",  
15      "EndTime": "16:00:00",  
16      "CategoryName": "Arts",  
17      "LocationName": "J Paul Leonard Library"  
18    },  
19    {  
20      "id": 2,  
21      "Name": "Study Group",  
22      "Description": "Oh shit we have an exam",  
23      "AgeRestriction": 0,  
24      "UserId": 3,  
25      "CategoryId": 2,  
26    }  
27  ]  
28 }
```

1.4 /users/RSVP

- Description: When a user rsvp's for an event we add that record into the db

- Request:

- endpoint: /users/RSVP
- method="POST"
- Headers

Key	Value
Authorization	"USER_TOKEN"
Content-Type	application/json

- Body

- Raw -> application/json

```
{  
    "UserId" : #,  
    "EventId" : #  
}
```

- Response:

- (bool) **status**: successful (*true == yes, false == no*)
- (string) **message**: information about your response

```
{  
    "status": true/false,  
    "message": "...",  
}
```

1.5 /users/RSVP

- Description: Removes an RSVP record from RSVP table in db.
- Request:

- endpoint: /users/RSVP
- method="DELETE"
- Headers

Key	Value
Authorization	"USER_TOKEN"
Content-Type	application/json

- Body
 - Raw -> application/json

```
{  
    "UserId" : #,  
    "EventId" : #  
}
```

- Response:
 - (bool) **status**: successful (*true == yes, false == no*)
 - (string) **message**: information about your response

```
{  
    "status": true/false,  
    "message": "...",  
}
```

1.6 /users/RSVP/:UserId(\\d+)

- Description: Retrieves all events a specific user has RSVP'd for.

- Request:

- endpoint: /users/RSVP/:UserId(\\d+)
- method="GET"
- Headers

Key	Value
Authorization	"USER_TOKEN"
Content-Type	application/json

- Body
 - none

- Response:

- (*bool*) **status**: successful (*true* == yes, *false* == no)
- (*string*) **message**: information about your response
- (Object array) **data**: array of events user has RSVP'd for

```
{
  "status": true/false,
  "message": "...",
  "data": [
    {
      //event contents
    }
  ]
}
```

1.7 /users/posts

- Description: Return all posts related to a specific user id.
- Request:

- endpoint: /users/posts
- method="POST"
- Headers

Key	Value
Authorization	"USER_TOKEN"
Content-Type	application/json

- Body
 - Raw -> application/json

```
{
    "UserId" : #,
}
```

- Response:
 - (bool) **status**: successful (*true == yes, false == no*)
 - (string) **message**: information about your response
 - (object array) **data** : array of events

```
{
  "status": true/false,
  "message": "...",
  "data": [
    {
      //event contents
    }
  ]
}
```

2.1 events/

- Description: get all events from db

- Request:

- endpoint: /events/
- method="GET"
- Headers

Key	Value
Authorization	"USER_TOKEN"
Content-Type	application/json

- Body
none

- Response:

- (bool) **status**: successful (*true == yes, false == no*)
- (string) **message**: information about your response
- (object array) **data** : array of events

```
{  
  "status": true/false,  
  "message": "...",  
  "data": [  
    {  
      //event details  
    }  
  ]  
}
```

2.2 events/

- Description: Post an event. This event is inserted into database

- Request:

- endpoint: /events/
- method="POST"
- Headers

Key	Value
Authorization	"USER_TOKEN"
Content-Type	application/json

- Body

- Raw -> application/json

```
{ "Name" : "...",  
  "Description" : "...",  
  "AgeRestriction" : #,  
  "UserId" : #,  
  "CategoryId" : #,  
  "LocationId" : #,  
  "Image" : RAW,  
  "StartDate" : date,  
  "StartTime" : time,  
  "EndTime" : time }
```

- Response:

- (bool) **status**: successful insertion(*true == yes, false == no*)
 - (string) **message**: information about your response
 - (int) **EventId**: id of event inserted into database
- ```
{ "status": true/false,
 "message": "response info",
 "EventId": # }
```

## 2.3 events/:id

- Description: delete an event from db.

- Request:

- endpoint: /events/:id
  - method="DELETE"
  - Headers

| Key           | Value            |
|---------------|------------------|
| Authorization | "USER_TOKEN"     |
| Content-Type  | application/json |

- Body

- Raw -> application/json

```
{
 "id" : #
}
```

- Response:

- (bool) **status**: successful insertion(*true == yes, false == no*)
  - (string) **message**: information about your response

```
{
 "status": true/false,
 "message": "response info"
}
```



## 2.4 events/:id

- Description: get a specific event by id

- Request:

- endpoint: /events/:id
  - method="GET"
  - Headers

| Key           | Value            |
|---------------|------------------|
| Authorization | "USER_TOKEN"     |
| Content-Type  | application/json |

- Body (none)

- Response:

- (bool) **status**: successful insertion(*true == yes, false == no*)
  - (string) **message**: information about your response
  - (object) **event**: event details

```
{ "event" : [
 { "status": true/false,
 "message": "response info",
 "Name" : "...",
 "Description" : "...",
 "AgeRestriction" : #,
 "UserId" : #,
 "CategoryId" : #,
 "LocationId" : #,
 "Image" : RAW,
 "LocationName": "...",
 "CategoryName": "...",
 "Longitude": #,
 "Latitude": #,
 "StartDate" : date,
 "StartTime" : time,
```

```
 "EndTime" : time }],
 "status": true/false,
 "message": "..."
}
```

## 2.5 events/filter/:id

- Description: get all events by category id

- Request:

- endpoint: /events/filter/:id
  - method="GET"
  - Headers

| Key           | Value            |
|---------------|------------------|
| Authorization | "USER_TOKEN"     |
| Content-Type  | application/json |

- Body (none)

- Response:

- (*bool*) **status**: successful insertion(*true == yes, false == no*)
  - (*object array*) **data**: array of events and their info  
{ "data" : [{event details}, {event details}, {event details}],  
"status": true/false,  
}

## 3.1 messages/send

- Description: insert record in Message table that keeps track of messages between user and event

- Request:

- endpoint: /messages/send
- method="POST"
- Headers

| Key           | Value            |
|---------------|------------------|
| Authorization | "USER_TOKEN"     |
| Content-Type  | application/json |

- Body

- Raw -> application/json

```
{
 "SenderId" : #,
 "ReceiverEventId": #,
 "Message": #
}
```

- Response:

- (bool) **status**: successful insertion(*true == yes, false == no*)
- (string) **message**: information about your response

```
{
 "status": true/false,
 "message": "response info"
}
```

## 3.1 messages/:EventId

- Description: get all messages for a specific event.

- Request:

- endpoint: /messages/:EventId
  - method="GET"
  - Headers

| Key           | Value            |
|---------------|------------------|
| Authorization | "USER_TOKEN"     |
| Content-Type  | application/json |

- Body (none)

- Response:

- (*bool*) **status**: successful insertion(*true == yes, false == no*)
  - (*string*) **message**: information about your response
  - (object array) **data**: array of messages

```
{
 "status": true/false,
 "message": "response info",
 "data": [{
 "Message": "...",
 "TimeStamp": time
 }]
}
```

## 4.1 maps/log

- Description: log location and internet speed for wifiMap
- Request:

- endpoint: /maps/log
- method="POST"
- Headers

| Key           | Value            |
|---------------|------------------|
| Authorization | "USER_TOKEN"     |
| Content-Type  | application/json |

- Body

```
{
 "Long": #,
 "Lat": #,
 "ResponseTime": #
}
```

- Response:
    - (*bool*) **status**: successful insertion(*true* == yes, *false* == no)
    - (*string*) **message**: information about your response
- ```
{  
    "status": true/false,  
    "message": "response info",  
}
```

4.2 maps/log

- Description: get all logs for WifiMap

- Request:

- endpoint: /maps/log
 - method="GET"
 - Headers

Key	Value
Authorization	"USER_TOKEN"
Content-Type	application/json

- Body (none)

- Response:

- *(bool)* **status**: successful insertion(*true == yes, false == no*)
 - *(string)* **message**: information about your response
 - *(object array)* **data**: array of logs

```
{  
  "status": true/false,  
  "message": "response info",  
  "data": [{  
    "Long": #,  
    "Lat": #,  
    "ResponseTime": #  
  }]  
}
```

4.2 maps/log/:Limit

- Description: get limited amount of logs for WifiMap

- Request:

- endpoint: /maps/log/:Limit

- :Limit (*int*)

- method="GET"

- Headers

Key	Value
Authorization	"USER_TOKEN"
Content-Type	application/json

- Body (none)

- Response:

- (*bool*) **status**: successful insertion(*true == yes, false == no*)

- (*string*) **message**: information about your response

- (*object array*) **data**: array of logs with a max size of :Limit

```
{  
  "status": true/false,  
  "message": "response info",  
  "data": [{  
    "Long": #,  
    "Lat": #,  
    "ResponseTime": #  
  }]  
}
```


5.1 categories/

- Description: get all categories
- Request:
 - endpoint: /categories/
 - method="GET"
 - Headers (none)
 - Body (none)
- Response:
 - (*bool*) **status**: successful insertion(*true == yes, false == no*)
 - (*string*) **message**: information about your response
 - (object array) data: array of categories

```
{  
  "status": true/false,  
  "message": "response info",  
  "data": [{  
    "id": #,  
    "Name": "..."  
  }]  
}
```

6.1 categories/

- Description: get all locations
- Request:
 - endpoint: /locations/
 - method="GET"
 - Headers (none)
 - Body (none)
- Response:
 - (*bool*) **status**: successful insertion(*true* == *yes*, *false* == *no*)
 - (*string*) **message**: information about your response
 - (object array) **data**: array of locations

```
{  
  "status": true/false,  
  "message": "response info",  
  "data": [{  
    "id": #,  
    "Name": "...",  
    "Longitude": #,  
    "Latitude": #  
  }]  
}
```