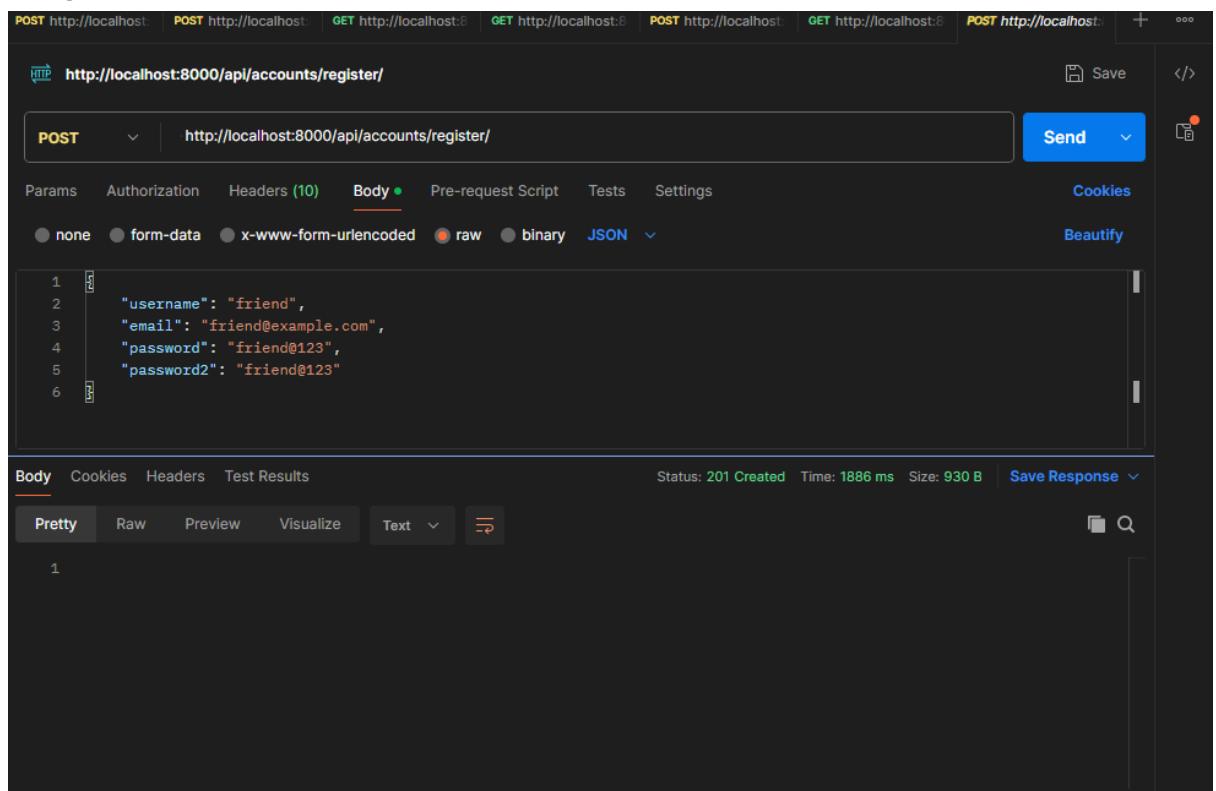


This documentation provides a comprehensive guide for testing the Event Management System API using Postman, featuring complete request configurations, expected responses, and authentication flows to validate all endpoints including user registration, event creation, RSVP management, and review functionality.

## 1. Register



The screenshot shows the Postman interface with a POST request to `http://localhost:8000/api/accounts/register/`. The request body is set to `JSON` and contains the following JSON payload:

```
1
2   "username": "friend",
3   "email": "friend@example.com",
4   "password": "friend@123",
5   "password2": "friend@123"
```

The response status is `201 Created`, time `1886 ms`, and size `930 B`.

## 2. Login

The screenshot shows the Postman application interface. At the top, the URL `http://localhost:8000/api/accounts/login/` is entered. The method dropdown shows "POST". Below the URL, the "Body" tab is selected, showing the JSON payload:

```
1
2     "username": "friend",
3     "password": "friend@123"
```

Below the body, the response status is shown as 200 OK with a response time of 4.03s and a size of 913B. The response body is displayed in pretty JSON format:

```
1
2     "message": "Login successful",
3     "user": {
4         "id": 4,
5         "username": "friend",
6         "email": "friend@example.com"
7     },
8     "tokens": {
9         "refresh": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
          eyJ0b2tlb190eXBlijoicVmcmVzaCisImV4cIC6MTc2NTU0NDI3MCwiWF0IjoxNzY1NDU30DcwLCJqdGkiOiIyMjY3NmI0ZTzlMmY0MW
          U5OTMyNTI2NDc1ZWYwYzc2NiIsInVzZXJfaWQiOii0In0.mFbhaXKiqyRjkbx7U8cXfZBnC5uo2LSJ54t89U21IFo",
10        "access": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
          eyJ0b2tlb190eXBlijoicYWNjZXNzIiwiZXhwIjoxNzY1NDYxNDcwLCJpYXQiOjE3NjU0NTc4NzEsImp0aSI6IjdkMDlkMWNlNWVlYjQzYw
          M4ZmZjODM4ZDZmNGJk0tg2IiwiidXNlc19pZC16IjQifQ.bRa4I5P4WHDbMqUy7y00xyYmhMPIw4qzYTIVrynujvg"
```

### 3. Profile

The screenshot shows the Postman application interface. At the top, there are several tabs representing different requests: POST http://localhost:8000/api/accounts/login, POST http://localhost:8000/api/accounts/logout, GET http://localhost:8000/api/accounts/profile, POST http://localhost:8000/api/accounts/register, GET http://localhost:8000/api/accounts/me, and GET http://localhost:8000/api/accounts/me. The current request being viewed is a GET request to http://localhost:8000/api/accounts/profile.

The main interface shows the following details:

- Method:** GET
- URL:** http://localhost:8000/api/accounts/profile/
- Headers:** (7)
  - Authorization:** Bearer eyJhbGciOiJIUzI1NlslslnR5cCl6lkpXVCJ9.eyJ0b2tlbl90eXBllj... (checkbox checked)
  - Key:** Value
- Body:** Cookies Headers (10) Test Results
- Status:** 200 OK Time: 213 ms Size: 455 B Save Response
- Pretty:** Raw Preview Visualize JSON
- Response Body (JSON):**

```
1   {
2     "id": 4,
3     "user": {
4       "id": 4,
5       "username": "friend",
6       "email": "friend@example.com"
7     },
8     "full_name": "",
9     "bio": "",
10    "location": "",
11    "profile_picture": null
12  }
```

## 4. Put /Profile:

The screenshot shows the Postman interface with a PUT request to `http://localhost:8000/api/accounts/profile/`. The Headers tab is selected, showing two checked items: `Authorization` with the value `Bearer eyJhbGciOiJIUzI1NilsInR5cCl6IkpxVCJ9eyJ0b2tlbl90eXBilj...` and `Content-Type` with the value `application/json`. The Body tab is selected, showing a JSON response with the following data:

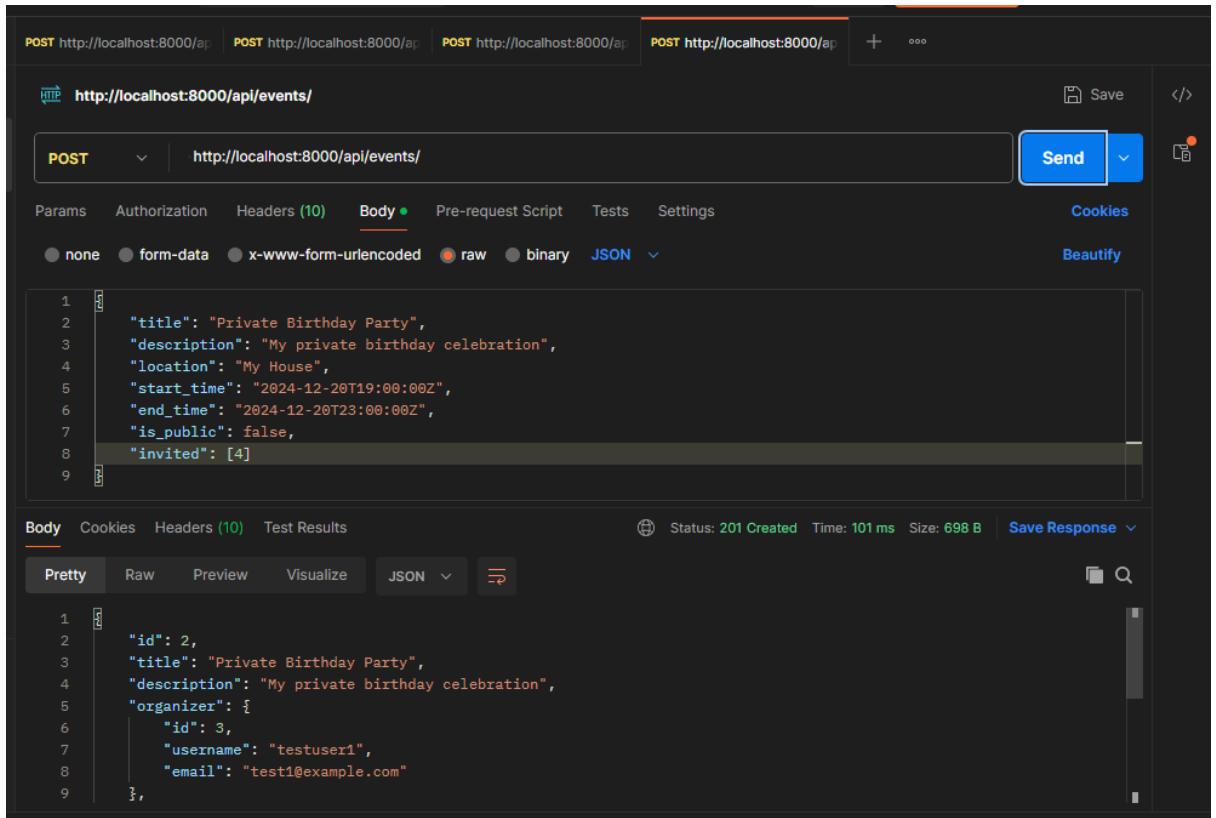
```
1 "id": 3,
2 "user": {
3     "id": 3,
4     "username": "testuser1",
5     "email": "test1@example.com"
6 },
7 "full_name": "Nima Shah",
8 "bio": "Software Developer",
9 "location": "Ahmedabad",
10 "profile_picture": null
11
12
```

## 5. Post event

The screenshot shows the Postman interface with a POST request to `http://localhost:8000/api/events/`. The Headers tab is selected, showing `Content-Type: application/json`. The Body tab is selected, showing a JSON body with the following data:

```
1 {
2     "title": "Tech Conference 2024",
3     "description": "Annual technology conference",
4     "location": "Gujarat Convention Center",
5     "start_time": "2024-12-15T09:00:00Z",
6     "end_time": "2024-12-15T18:00:00Z",
7     "is_public": true
8 }
```

## 6. Post Private event



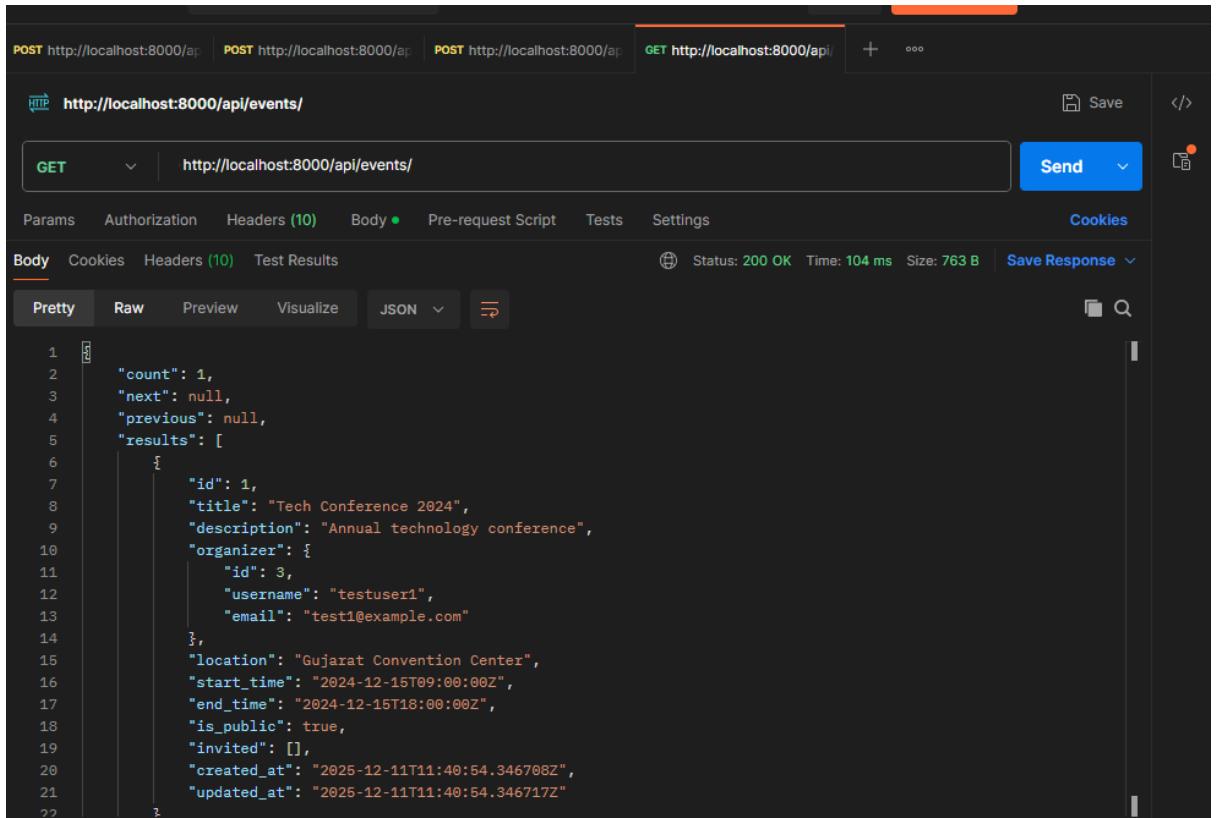
The screenshot shows a Postman interface with a POST request to `http://localhost:8000/api/events/`. The request body is a JSON object representing a private event:

```
1 "title": "Private Birthday Party",
2 "description": "My private birthday celebration",
3 "location": "My House",
4 "start_time": "2024-12-20T19:00:00Z",
5 "end_time": "2024-12-20T23:00:00Z",
6 "is_public": false,
7 "invited": [4]
```

The response status is 201 Created, and the response body is:

```
1 {
2   "id": 2,
3   "title": "Private Birthday Party",
4   "description": "My private birthday celebration",
5   "organizer": {
6     "id": 3,
7     "username": "testuser1",
8     "email": "test1@example.com"
9   },
10 }
```

## 7. Get public events



The screenshot shows a Postman interface with a GET request to `http://localhost:8000/api/events/`. The response status is 200 OK, and the response body is:

```
1 {
2   "count": 1,
3   "next": null,
4   "previous": null,
5   "results": [
6     {
7       "id": 1,
8       "title": "Tech Conference 2024",
9       "description": "Annual technology conference",
10      "organizer": {
11        "id": 3,
12        "username": "testuser1",
13        "email": "test1@example.com"
14      },
15      "location": "Gujarat Convention Center",
16      "start_time": "2024-12-15T09:00:00Z",
17      "end_time": "2024-12-15T18:00:00Z",
18      "is_public": true,
19      "invited": [],
20      "created_at": "2025-12-11T11:40:54.346708Z",
21      "updated_at": "2025-12-11T11:40:54.346717Z"
22    }
23  ]
```

## 8. Search:

The screenshot shows the Postman interface with a search query for events. The URL is `http://localhost:8000/api/events/?search=conference`. The 'Params' tab has a key 'search' with the value 'conference'. The response status is 200 OK, time 183 ms, size 763 B. The JSON response is:

```
1  {
2      "count": 1,
3      "next": null,
4      "previous": null,
5      "results": [
6          {
7              "id": 1,
8              "title": "Tech Conference 2024",
9              "description": "Annual technology conference",
10             "organizer": {
11                 "id": 3,
12                 "username": "testuser1",
13                 "email": "test1@example.com"
14             },
15             "location": "Gujarat Convention Center",
16             "start_time": "2024-12-15T09:00:00Z",
17             "end_time": "2024-12-15T18:00:00Z",
18             "is_public": true,
19         }
20     ]
21 }
```

## 9. Get event by id

The screenshot shows the Postman interface with a request for an event by ID. The URL is `http://localhost:8000/api/events/1`. The 'Body' tab is selected with 'raw' selected. The response status is 200 OK, time 193 ms, size 725 B. The JSON response is:

```
1  {
2      "id": 1,
3      "title": "Tech Conference 2024",
4      "description": "Annual technology conference",
5      "organizer": {
6          "id": 3,
7          "username": "testuser1",
8          "email": "test1@example.com"
9      },
10     "location": "Gujarat Convention Center",
11     "start_time": "2024-12-15T09:00:00Z",
12     "end_time": "2024-12-15T18:00:00Z",
13     "is_public": true,
14     "invited": [],
15     "created_at": "2025-12-11T11:40:54.346708Z",
16     "updated_at": "2025-12-11T11:40:54.346717Z"
17 }
```

## 10. Update event

The screenshot shows the Postman interface with a PATCH request to `http://localhost:8000/api/events/1/`. The request body contains the following JSON:

```
1
2   "title": "Tech Conference 2024 - Updated",
3   "location": "Updated Location"
4
```

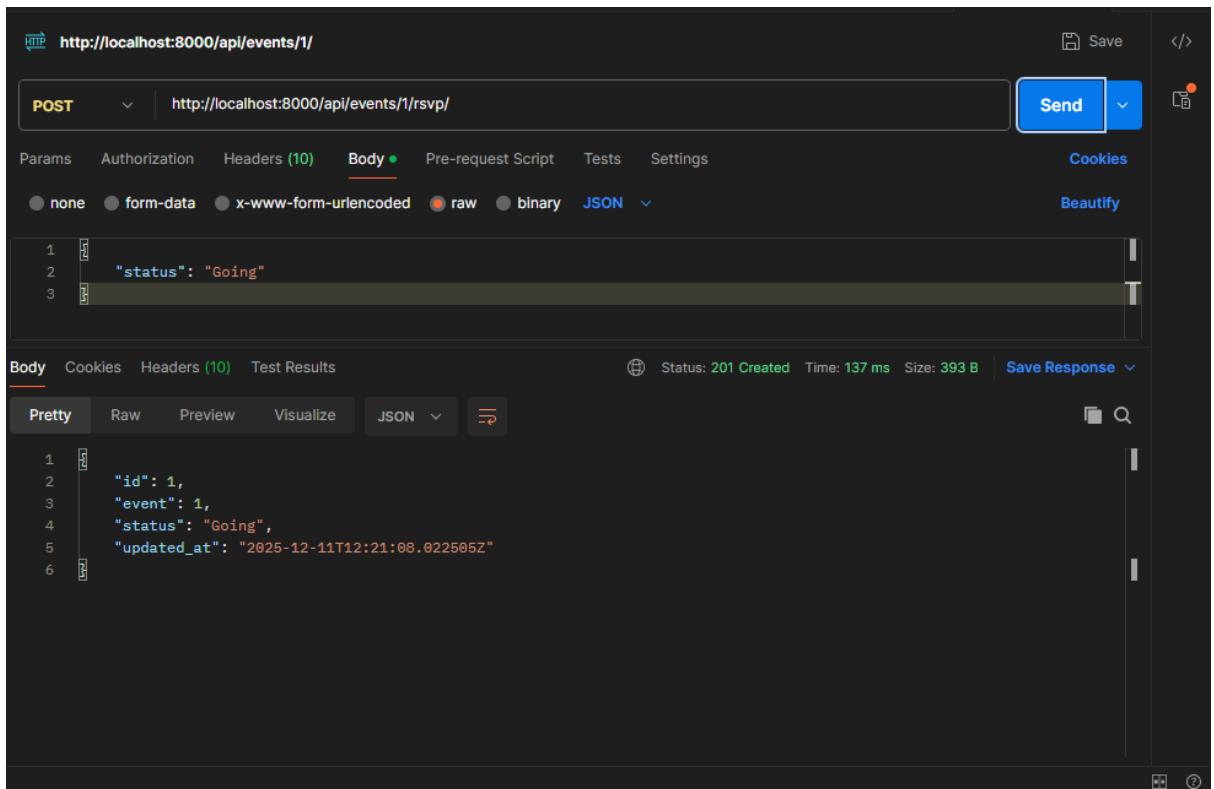
The response status is 200 OK, with a response body showing the updated event details:

```
1
2   "id": 1,
3   "title": "Tech Conference 2024 - Updated",
4   "description": "Annual technology conference",
5   "organizer": {
6     "id": 3,
7     "username": "testuser1",
8     "email": "test1@example.com"
9   },
10  "location": "Updated Location",
11  "start_time": "2024-12-15T09:00:00Z",
12  "end_time": "2024-12-15T18:00:00Z",
13  "is_public": true,
14  "invited": []
```

## 11. Delete event(only organizer)

The screenshot shows the Postman interface with a DELETE request to `http://localhost:8000/api/events/1/`. The response status is 204 No Content.

## 12. Post RSVP (only organizer)



HTTP <http://localhost:8000/api/events/1/>

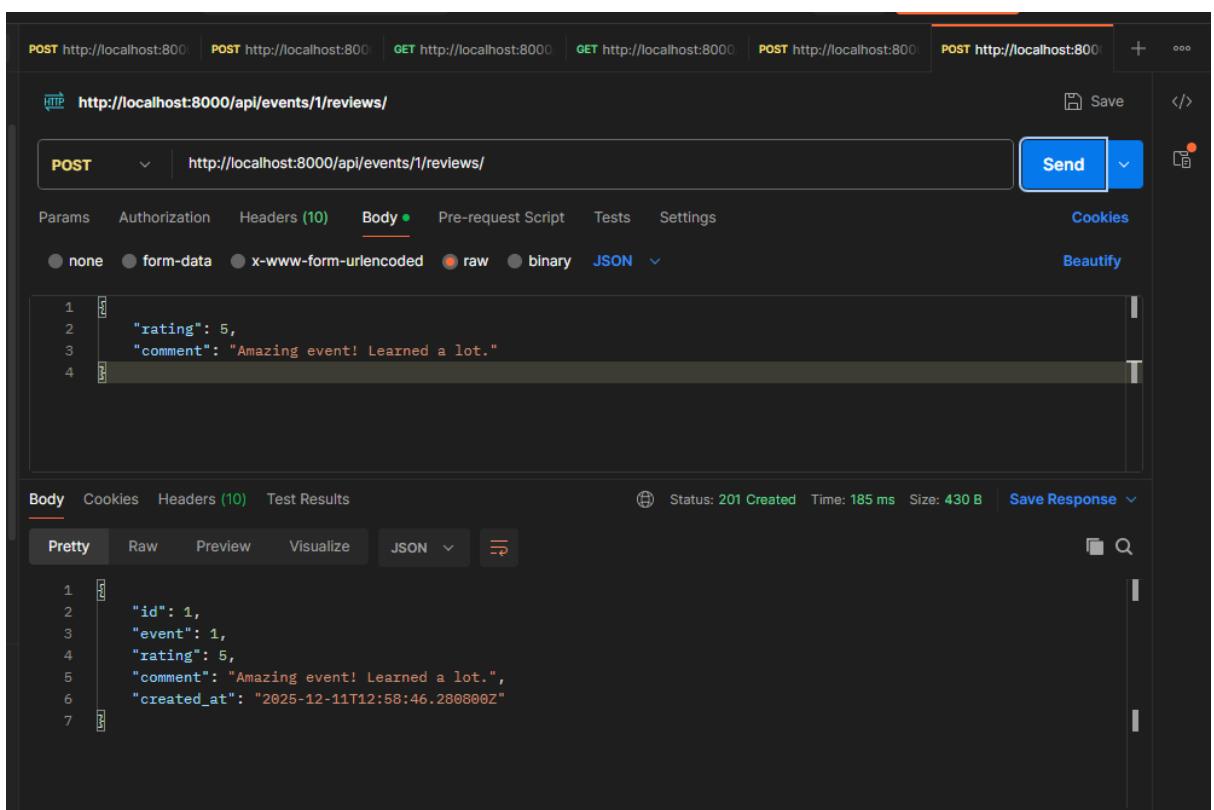
POST <http://localhost:8000/api/events/1/rsvp/>

Body `status: "Going"`

Status: 201 Created Time: 137 ms Size: 393 B

```
1 "id": 1,
2 "event": 1,
3 "status": "Going",
4 "updated_at": "2025-12-11T12:21:08.022505Z"
```

## 13. Post Review



HTTP <http://localhost:8000/api/events/1/reviews/>

POST <http://localhost:8000/api/events/1/reviews/>

Body `rating: 5, comment: "Amazing event! Learned a lot."`

Status: 201 Created Time: 185 ms Size: 430 B

```
1 "id": 1,
2 "event": 1,
3 "rating": 5,
4 "comment": "Amazing event! Learned a lot.",
5 "created_at": "2025-12-11T12:58:46.280800Z"
```