

# API testing

## API: GET /events

Pentru a verifica corectitudinea API-ului de obtinerea a tuturor evenimentelor(request-ul de obtinere a evenimentelor existente intoarce ceea ce ne asteptam) in cazul de fata (we already know we have 12 events 😊) ) putem verifica ca raspunsul contine 12 evenimente, iar status code-ul este 200.

```
GET http://localhost:3000/events/

{
  "id": "1",
  "topics": "sports",
  "thumbnail": "/img/t-1.jpeg",
  "url": "index.html",
  "overrideURL": "",
  "linkType": "",
  "title": "Georgetown at Butler: Georgetown at Butler",
  "summary": "Georgetown at Butler: Georgetown's D'Vaunte Smith-Rivers squares off with Butler's Kellen Dunham in a battle of senior guards. The Hoyas took two of three meetings with the Bulldogs last season."
},
{
  "id": "2",
  "topics": "fashion",
  "thumbnail": "/img/t-2.jpeg",
  "url": "index.html",
  "overrideURL": "",
  "linkType": "",
  "title": "2016 Olympic Trials",
  "summary": "Qualifying time standards for the 2016 Olympic Trials were announced Thursday, September 10, at 2 p.m. EDT, via a live webcast from the United States Aquatic Sports Convention in Jacksonville, Florida. It can be viewed via usa swimming.org/trials."
},
{
  "id": "3",
  "topics": "business",
  "thumbnail": "/img/t-3.jpeg",
  "url": "index.html",
  "overrideURL": "",
  "linkType": "",
  "title": "Business 3",
  "summary": "Lorem ipsum dolor sit amet"
},
{
  "id": "4",
  "topics": "technology",
  "thumbnail": "/img/t-4.jpeg",
  "url": "index.html",
  "overrideURL": "",
  "linkType": "",
  "title": "Tech 1",
  "summary": "Ut enim ad minim veniam"
},
{
  "id": "5",
  "topics": "transport",
  "thumbnail": "/img/tr-1.jpeg",
  "url": "index.html",
  "overrideURL": "",
  "linkType": "",
  "title": "Autonomous Cars Silicon Valley 2016",
  "summary": "The Newest Tools, Technologies, and Techniques required for the Pursuit of the Autonomous Passenger Vehicle."
},
{
  "id": "6",
  "topics": "sports",
  "thumbnail": "/img/t-6.jpeg",
  "url": "index.html",
  "overrideURL": "",
  "linkType": "",
  "title": "Base Jumping",
  "summary": "Is very extreme activity that includes a parachute (can be used both parachute and wingsuit) to jump from fixed objects, with unopened parachute like skydiving. The acronym \"B.A.S.E.\" stands for: Building, Antenna, Span, Earth - four categories of objects from which B.A.S.E. jumper can jump."
},
{
  "id": "7",
  "topics": "business",
  "thumbnail": "/img/t-7.jpeg",
  "url": "index.html",
  "overrideURL": "",
  "linkType": "",
  "title": "Business 2",
  "summary": "Lorem ipsum dolor sit amet"
},
{
  "id": "8",
  "topics": "technology",
  "thumbnail": "/img/t-8.jpeg",
  "url": "index.html",
  "overrideURL": "",
  "linkType": "",
  "title": "Tech 2",
  "summary": "Lorem ipsum dolor sit amet"
},
{
  "id": "9",
  "topics": "fashion",
  "thumbnail": "/img/t-9.jpeg",
  "url": "index.html",
  "overrideURL": "",
  "linkType": "",
  "title": "Fashion Week 2016",
  "summary": "Fashion Week 2016 is coming 'round the bend!"
},
{
  "id": "10",
  "topics": "technology",
  "thumbnail": "/img/t-10.jpeg",
  "url": "index.html",
  "overrideURL": "",
  "linkType": "",
  "title": "Tech 3",
  "summary": "Lorem ipsum dolor sit amet"
},
{
  "id": "11",
  "topics": "transport",
  "thumbnail": "/img/tr-2.jpeg",
  "url": "index.html",
  "overrideURL": "",
  "linkType": "",
  "title": "Transport 3",
  "summary": "Lorem ipsum dolor sit amet"
},
{
  "id": "12",
  "topics": "sports",
  "thumbnail": "/img/t-12.jpeg",
  "url": "index.html",
  "overrideURL": "",
  "linkType": "",
  "title": "Sports 12",
  "summary": "Lorem ipsum dolor sit amet"
}
```

Dupa cum putem observa, API-ul functioneaza corect, deoarece rezultatul intors este cel asteptat (12 evenimente), iar structura fiecarui eveniment este cea din descrierea oferita de developer.

### API: GET /events/:id

Pentru a valida faptul ca acest API functioneaza corect, trebuie sa testam cele doua cazuri posibile:

1. id-ul furnizat identifica un eveniment existent in “baza de date”
2. id-ul furnizat este invalid.

Pentru primul caz utilizam un id (folosim id-ul unui eveniment din cele 12; avem acces la id-uri, deoarece am considerat ca API-ul precedent functioneaza corect).

GET http://localhost:3000/events/5

Params Authorization Headers (7) Body Pre-request Script Tests Settings

Query Params

KEY	VALUE
Key	Value

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": "5",
3   "topics": "",
4   "thumbnail": "/img/tr-3.jpeg",
5   "url": "index.html",
6   "overrideURL": "",
7   "linkType": "",
8   "title": "Transport 1",
9   "summary": "Lorem ipsum dolor sit amet"
10 }
```

Pentru cazul in care id-ul este invalid, raspunsul ar trebui sa reflecte acest lucru. Let's test and see the result.

GET http://localhost:3000/events/50

Params Authorization Headers (7) Body Pre-request Script Tests Settings

Query Params

KEY	VALUE
Key	Value

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "error": "Id not found"
3 }
```

## API: POST /events

În cazul acestui API avem 5 cazuri:

1. încercăm să creăm un eveniment cu toate datele necesare
2. încercăm să creăm un eveniment cu toate datele, mai puțin câmpul id.
3. încercăm să creăm un eveniment cu valori invalide pentru anumite câmpuri.
4. încercăm să creăm un eveniment cu unele câmpuri lipsă (diferite de id)
5. încercăm să creăm un eveniment cu un id deja existent (care identifică o resursă existentă).

Pentru primul caz flow-ul de testare este următorul:

- facem request-ul de crearea a unui nou eveniment

The screenshot displays a REST client interface for a POST request to `http://localhost:3000/events/`. The 'Body' tab is selected, showing a JSON payload. Below the request, the 'Test Results' tab is active, displaying the same JSON response in a 'Pretty' format.

```
POST http://localhost:3000/events/

{
  "id": "50",
  "topics": "bla bla",
  "thumbnail": "/img/s-4.jpeg",
  "url": "index.html",
  "overrideURL": "",
  "linkType": "",
  "title": "Created by Gigel",
  "summary": "Gigel este Gigel"
}
```

Test Results (JSON):

```
{
  "id": "50",
  "topics": "bla bla",
  "thumbnail": "/img/s-4.jpeg",
  "url": "index.html",
  "overrideURL": "",
  "linkType": "",
  "title": "Created by Gigel",
  "summary": "Gigel este Gigel"
}
```

- verificam daca noul eveniment exista in “baza de date” folosind API-ul de obtinere a unui eveniment identificat prin id-ul sau.

GET http://localhost:3000/events/50

Params Authorization Headers (7) Body Pre-request Script Tests Settings

Query Params

KEY	VALUE
Key	Value

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize JSON

```

1 {
2   "id": "50",
3   "topics": "bla bla",
4   "thumbnail": "/img/s-4.jpeg",
5   "url": "index.html",
6   "overrideURL": "",
7   "linkType": "",
8   "title": "Created by Gigel",
9   "summary": "Gigel este Gigel"
10 }
```

Pentru cel de-al doilea caz urmam aceeasi pasi ca mai sus (doar ca de data aceasta nu vom mai furniza id-ul):

POST http://localhost:3000/events/

```

1 {
2   "topics": "another bla bla",
3   "thumbnail": "/img/f-1.jpeg",
4   "url": "index.html",
5   "overrideURL": "",
6   "linkType": "",
7   "title": "Created by Frone",
8   "summary": "Frone nu este Gigel"
9 }
```

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize JSON

```

1 {
2   "topics": "another bla bla",
3   "thumbnail": "/img/f-1.jpeg",
4   "url": "index.html",
5   "overrideURL": "",
6   "linkType": "",
7   "title": "Created by Frone",
8   "summary": "Frone nu este Gigel",
9   "id": "13"
10 }
```

GET http://localhost:3000/events/13

Params Authorization Headers (7) Body Pre-request Script

Query Params

KEY	VALUE
Key	Value

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize JSON

```

1 {
2   "topics": "another bla bla",
3   "thumbnail": "/img/f-1.jpeg",
4   "url": "index.html",
5   "overrideURL": "",
6   "linkType": "",
7   "title": "Created by Frone",
8   "summary": "Frone nu este Gigel",
9   "id": "13"
10 }
```

Dupa cum putem observa, aplicatia creeaza un nou eveniment chiar daca nu am furnizat un id.

Pentru cazul al treilea, vom incerca sa creem un eveniment care contine o valoare invalida pentru campul 'thumbnail'. De asemenea, vom urma pasii de la celelalte doua cazuri:

The screenshot displays a REST client interface with two panels. The left panel shows a POST request to `http://localhost:3000/events/` with a JSON body containing an invalid thumbnail URL: `"/img/invalid"`. The right panel shows the response, which is a 200 OK status with a JSON body containing the created event details, including an `"id": "14"`.

```
POST http://localhost:3000/events/
{
  "topics": "another bla bla",
  "thumbnail": "/img/invalid",
  "url": "index.html",
  "overrideURL": "",
  "linkType": "",
  "title": "Created by Invalid",
  "summary": "Invalid este Invali :))"
}
```

```
GET http://localhost:3000/events/14
{
  "topics": "another bla bla",
  "thumbnail": "/img/invalid",
  "url": "index.html",
  "overrideURL": "",
  "linkType": "",
  "title": "Created by Invalid",
  "summary": "Invalid este Invali :))",
  "id": "14"
}
```

Se poate observa ca evenimentul este creat cu succes, ceea ce inseamna ca aplicatia nu face validarea datelor, deci va trebui sa semnalam un bug 😊).

Pentru cel de-al patrulea caz urmam aceeasi pasi, dar nu vom furniza campul 'url' cand vom face request-ul de creare:

The screenshot displays a REST client interface with two panels. The left panel shows a POST request to `http://localhost:3000/events/` with a JSON body that omits the 'url' field. The right panel shows the response, which is a 200 OK status with a JSON body containing the created event details, including an `"id": "13"`.

```
POST http://localhost:3000/events/
{
  "topics": "another bla bla",
  "thumbnail": "/img/b-1.jpeg",
  "overrideURL": "",
  "linkType": "",
  "title": "Created by Ion",
  "summary": "No URL provided"
}
```

```
GET http://localhost:3000/events/13
{
  "topics": "another bla bla",
  "thumbnail": "/img/b-1.jpeg",
  "overrideURL": "",
  "linkType": "",
  "title": "Created by Ion",
  "summary": "No URL provided",
  "id": "13"
}
```

Din nou putem observa ca aplicatia creeaza un nou eveniment, chiar daca unul dintre campuri lipseste; probabil acel camp este optional sau mai dagraaba aplicatia nu face validarea datelor (eu pariez pe aceasta optiune 😊) ).

Pentru ultimul caz, urmam aceeasi pasi ca mai sus, la care adaugam pasul de verificare a resursei identificata prin id-ul ales:

GET ▼ http://localhost:3000/events/11

Params Authorization Headers (7) Body Pre-request Script Tests

Query Params

KEY	VALUE
Key	Value

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize JSON ▼ ≡

```
1 {
2   "id": "11",
3   "topics": "fashion",
4   "thumbnail": "/img/f-1.jpeg",
5   "url": "index.html",
6   "overrideURL": "",
7   "linkType": "",
8   "title": "Fashion Week 2016",
9   "summary": "Fashion Week 2016 is coming 'round the bend!"
10 }
```

POST ▼ http://localhost:3000/events/

```
1 {
2   "id": "11",
3   "topics": "another bla bla",
4   "thumbnail": "/img/invalid",
5   "url": "index.html",
6   "overrideURL": "",
7   "linkType": "",
8   "title": "Created by Invalid",
9   "summary": "Id already exists"
10 }
```

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize JSON ▼ ≡

```
1 {
2   "id": "11",
3   "topics": "another bla bla",
4   "thumbnail": "/img/invalid",
5   "url": "index.html",
6   "overrideURL": "",
7   "linkType": "",
8   "title": "Created by Invalid",
9   "summary": "Id already exists"
10 }
```

GET ▼ http://localhost:3000/events/11

Params Authorization Headers (7) Body Pre-request Script Tests Settings

Query Params

KEY	VALUE
Key	Value

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize JSON ▼ ≡

```
1 {
2   "id": "11",
3   "topics": "fashion",
4   "thumbnail": "/img/f-1.jpeg",
5   "url": "index.html",
6   "overrideURL": "",
7   "linkType": "",
8   "title": "Fashion Week 2016",
9   "summary": "Fashion Week 2016 is coming 'round the bend!"
10 }
```

GET ▼ http://localhost:3000/events/

Pretty Raw Preview Visualize JSON ▼ ≡

```
139 {
140   "summary": "Frone nu este Gigel",
141   "id": "13"
142 },
143 {
144   "topics": "another bla bla",
145   "thumbnail": "/img/invalid",
146   "url": "index.html",
147   "overrideURL": "",
148   "linkType": "",
149   "title": "Created by Invalid",
150   "summary": "Invalid este invalid :))",
151   "id": "14"
152 },
153 {
154   "topics": "another bla bla",
155   "thumbnail": "/img/invalid",
156   "overrideURL": "",
157   "linkType": "",
158   "title": "Created by Invalid",
159   "summary": "Invalid este invalid :))",
160   "id": "16"
161 },
162 {
163   "id": "11",
164   "topics": "another bla bla",
165   "thumbnail": "/img/invalid",
166   "url": "index.html",
167   "overrideURL": "",
168   "linkType": "",
169   "title": "Created by Invalid",
170   "summary": "Id already exists"
171 }
```

Dupa cum se poate observa, din nou, aplicatia creeaza un nou eveniment cu un id care deja identifica o resursa. Din nou putem afirma ca aplicatia nu face validarea datelor.

P.S: eu am presupus ca acel camp id este un identificator unic pentru o resursa.

## API: PUT /events/:id

În cazul acestui API, va trebui să verificăm următoarele două cazuri de bază (cazurile cu date lipsă/invalide le-am testat în cadrul API-ului de creare a unei noi resurse și am observat că aplicația nu face validarea datelor):

1. actualizarea unei resurse identificată printr-un id valid.
2. actualizarea unei resurse inexistente în “bază de date” (utilizăm un id invalid)

Pentru primul caz vom face request-ul de update, iar mai apoi vom verifica dacă resursa a fost actualizată corespunzător:

The first screenshot shows a PUT request to `http://localhost:3000/events/11` with a JSON body containing event details. The second screenshot shows the same PUT request with the body expanded. The third screenshot shows a GET request to `http://localhost:3000/events/11` returning the updated event data.

```
PUT http://localhost:3000/events/11
{
  "id": "11",
  "topics": "fashion",
  "thumbnail": "/img/f-1.jpeg",
  "url": "index.html",
  "overrideURL": "updated overrideURL",
  "linkType": "updated linkType",
  "title": "Updated by Florinel",
  "summary": "Lorem ipsum dolor sit amet -> c\u00e2eva care stie latine, please?"
}
```

```
GET http://localhost:3000/events/11
{
  "id": "11",
  "topics": "update topics",
  "thumbnail": "/img/f-1.jpeg",
  "url": "index.html",
  "overrideURL": "updated overrideURL",
  "linkType": "updated linkType",
  "title": "Updated by Florinel",
  "summary": "Lorem ipsum dolor sit amet -> c\u00e2eva care stie latine, please?"
}
```

Pentru cel de-al doilea caz, vom urma același flow, dar de data aceasta ne așteptăm ca server-ul să ne întoarcă o eroare deoarece event-ul nu există.

The first screenshot shows a PUT request to `http://localhost:3000/events/110` with a JSON body. The second screenshot shows the GET request to `http://localhost:3000/events/110` returning an error response.

```
PUT http://localhost:3000/events/110
{
  "id": "110",
  "topics": "update topics",
  "thumbnail": "/img/f-1.jpeg",
  "url": "index.html",
  "overrideURL": "updated overrideURL",
  "linkType": "updated linkType",
  "title": "Updated by Florinel",
  "summary": "Lorem ipsum dolor sit amet -> c\u00e2eva care stie latine, please?"
}
```

```
GET http://localhost:3000/events/110
{
  "error": "Id not found"
}
```

După cum putem observa, în acest caz, server-ul ne raportează cu un mesaj de eroare (care reflectă faptul că resursa nu există în “bază de date”).

## API: DELETE /events/:id

De asemenea, pentru acest API, va trebui sa testam comportamentul sau pentru cele doua cazuri tratate in caul API-ului de update, si anume:

1. stergerea unui eveniment existent in baza de date -> id valid
2. stergerea unui eveniment inexistent in baza de date -> id invalid

Pentru primul caz, vom face requeus-tul de stergere a unei resurse existente, iar mai apoi vom verifica ca respectiva resursa nu se mai gaseste in “baza de date”.

The first screenshot shows a GET request to `http://localhost:3000/events/5`. The response body is a JSON object with the following structure:

```
1 {
2   "id": "5",
3   "topics": "",
4   "thumbnail": "/img/tr-3.jpg",
5   "url": "index.html",
6   "overrideURL": "",
7   "linkType": "",
8   "title": "Transport 1",
9   "summary": "Lorem ipsum dolor sit amet"
10 }
```

The second screenshot shows a DELETE request to `http://localhost:3000/events/5`. The response body is a JSON object with the following structure:

```
1 {
2   "id": "5",
3   "topics": "",
4   "thumbnail": "/img/tr-3.jpg",
5   "url": "index.html",
6   "overrideURL": "",
7   "linkType": "",
8   "title": "Transport 1",
9   "summary": "Lorem ipsum dolor sit amet"
10 }
```

The third screenshot shows the same GET request, but the response body is a JSON object with the following structure:

```
1 {
2   "error": "Id not found"
3 }
```

Aceasi pasi ii vom urma si pentru cazul al doilea (evenimentul nu exista in “baza de date”).

The first screenshot shows a GET request to `http://localhost:3000/events/509`. The response body is a JSON object with the following structure:

```
1 {
2   "error": "Id not found"
3 }
```

The second screenshot shows a DELETE request to `http://localhost:3000/events/509`. The response body is a JSON object with the following structure:

```
1 {
2   "error": "Id not found"
3 }
```

Dupa cum putem observa, in acest caz, server-ul ne raspunde cu un mesaj de eroare sugestiv. Totusi, in cazul acesta, server-ul ar fi trebuit sa raspunda cu un mesaj de genul: “Event was deleted successfully”.