

AWS Lambda Function

Michele Dentella - 1063244

Get_Watch_Next_by_Idx: dati

Talk.js: oggetto nodeJs dove “mappare” il contenuto ricevuto dalla query su mongoDB

campi importanti:

- `_id` : String
- `watch_next_ids` : [String]
array di stringhe

```
1 const mongoose = require('mongoose');
2
3 const talk_schema = new mongoose.Schema({
4   _id:String,
5   title: String,
6   url: String,
7   watch_next_ids:[String]
8 }, { collection: 'tedz_data' });
9
10 module.exports = mongoose.model('talk', talk_schema);
```

~
"TCM-lab/laboratorio-TCM/compito3/Talk.js" 10 lines --10%--

1,1

All

Get_Watch_Next_by_Idx: handler

essenziale lambda function, basata sulla versione presentata a lezione.

- controllo che il campo non sia nullo
- query per id ricevuto dalla GET restituisco solo l'array dei watch_next

```
1 const connect_to_db = require('./db');
2
3 // GET BY TALK HANDLER
4
5 const talk = require('./Talk');
6
7 module.exports.get_next = (event, context, callback) => {
8   context.callbackWaitsForEmptyEventLoop = false;
9   console.log('Received event:', JSON.stringify(event, null, 2));
10  let body = {}
11  if (event.body) {
12    body = JSON.parse(event.body)
13  }
14  // set default
15  if(!body.id) {
16    callback(null, {
17      statusCode: 500,
18      headers: { 'Content-Type': 'text/plain' },
19      body: 'Could not fetch the talks. Watch_next_ids is null.'
20    })
21  }
22
23  connect_to_db().then(() => {
24    console.log('=> get_all talks');
25    talk.find({_id: body.id}, {_id: 0, watch_next_ids:1})
26      .then(talks => {
27        callback(null, {
28          statusCode: 200,
29          body: JSON.stringify(talks)
30        })
31      })
32    )
33    .catch(err =>
34      callback(null, {
35        statusCode: err.statusCode || 500,
36        headers: { 'Content-Type': 'text/plain' },
37        body: 'Could not fetch the talks.'
38      })
39    );
40  });
41 };
42
```

-b/laboratorio-TCM/compito3/Get_Watch_Next_by_Idx.js" 42 lines --2%-- 1,1 All

Get_Watch_Next_by_Idx: risultati

GET fatta all'API gateway pubblico

chiedo un id

ricevo un elenco di watch_next

The screenshot displays a REST client interface with a GET request to the URL `https://n3o0v1o640.execute-api.us-east-1.amazonaws.com/default/Get_Watch_Next_by_Idx`. The 'Body' tab is selected, showing a JSON object with an 'id' field. Below the request, the 'Body' tab of the response is shown, displaying a JSON array of 'watch_next_ids'.

```
GET https://n3o0v1o640.execute-api.us-east-1.amazonaws.com/default/Get_Watch_Next_by_Idx
```

Params Authorization Headers (8) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   "id": "8d2005ec35280deb6a438dc87b225f89"
3 }
```

Body Cookies Headers (7) Test Results

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "watch_next_ids": [
4       "fe35edd737282ab3a325f2387cf1b50b",
5       "8576654442b6633b1dc0eb48a989172a",
6       "5134ae81a27c94354173f38e84289ad5",
7       "5bd34fcc55d9e1267f605fa0c060d54e",
8       "d9896b41b372ec60cdd3c662e57caad3",
9       "078766d6cc461cf71d45dc268b66db95"
10    ]
11  }
12 ]
```

caso particolare: watch_next vuoto

chiedo l'id di un talk che non ha watch_next

FILTER {"watch_next_ids" : {\$exists : false}}

QUERY RESULTS 1-1 OF 1

> **_id**: "f3f90dad17fdc8b8cdf959618c356eb4"
title: "Year In Ideas 2015"
details: "The Year In Ideas 2015."
posted: "Posted Nov 2015"
url: "https://www.ted.com/talks/year_in_ideas_2015"
num_views: "0"
> **tags**: Array

```
1 {  
2   "id": "f3f90dad17fdc8b8cdf959618c356eb4"  
3 }
```

dy Cookies Headers (7) Test Results

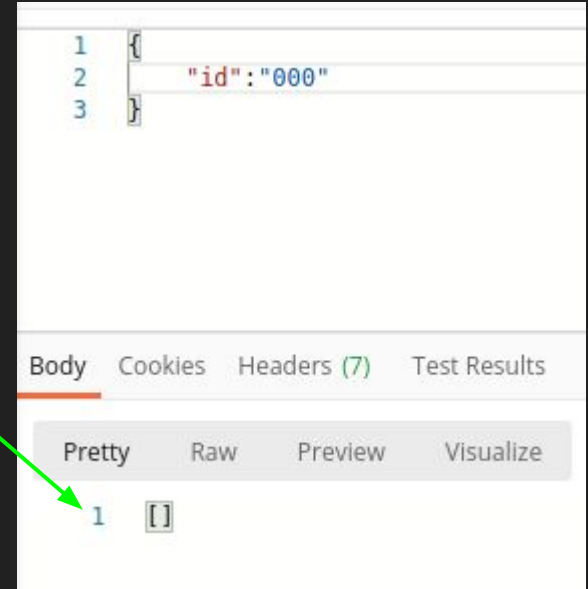
Pretty Raw Preview Visualize JSON ▼

```
1 [  
2   {  
3     "watch_next_ids": []  
4   }  
5 ]
```

risposta corretta: campo watch_next_ids vuoto

caso particolare: id non valido

riesco a distinguere il caso con id non valido da quello con watch_next vuoto perché la risposta è diversa



esperienza utente/possibili evoluzioni

l'utente finale non utilizzerà direttamente questa funzione.

A seconda di come è pensata l'app finale si possono fare leggere modifiche per ottenere direttamente l'URL del video o il titolo o altre informazioni utili per consigliare l'utente nella visione di altri contenuti

criticità tecniche

- difficoltà nel fare debug
- nomi simili tra dati/variabili su servizi diversi