TEDx TOk

Progetto parte 3:
Lambda

Mazzoleni Gabriele – 1079514
Masinari Gabriele - 1079692

Funzione MyTedx: Get_Watch_Next_By_Idx

Utilizzo

Questa Lambda Function consente di visualizzare tutti i video correlati, dato un ID iniziale di un video, visualizzando i seguenti parametri:

- Related_video_ids, ossia l'ID del video correlato.
- Related_video_title, ossia il titolo del video correlato.
- Related_presentedBy, ossia il presentatore del video.

Esperienza utente

Questo permette al nostro utente di avere una chiara comprensione della lista dei video correlati e di scegliere al meglio il video desiderato.

Codice della Lambda Function e API

```
connect_to_db().then(() => {
  console.log('=> get_all talks');
  talk.find({ "_id": body.id }).select('Related_videos -_id')
      .skip((body.doc_per_page * body.page) - body.doc_per_page)
      .limit(body.doc_per_page)
      .then(talks => {
       if (!talks|| talks.lenght===0) {
          throw new Error('Talk not found');
      }

        let relatedVideos = talks[0].Related_videos;

        // Remove _id from each related video
       relatedVideos = relatedVideos.map(video => {
         const { _id, ...rest } = video.toObject();
         return rest;
       });

       callback(null, {
          statusCode: 200,
          headers: { 'Content-Type': 'application/json' },
          body: JSON.stringify(relatedVideos)
       })
      }
      )
```

Questa Lambda function può essere testata con il seguente link API:

https://ee9l9w52bk.execute-api.us-east-1.amazonaws.com/default/Get_Watch_Next_by_Idx

Risultato della funzione: Get_Watch_Next_By_Idx

Parametro in INPUT:

```
{
    "id":"526880"
}
```

OUTPUT testato tramite Postman:


```
[
    {
        "related_video_ids": "109914",
        "related_video_title": "Whatever happened to the hole in the ozone layer?"
        "related_presentedBy": "Stephanie Honchell Smith"
    },
    {
        "related_video_ids": "76541",
        "related_video_title": "What's in the air you breathe?",
        "related_presentedBy": "Amy Hrdina and Jesse Kroll"
    },
    {
        "related_video_ids": "100294",
        "related_video_title": "Why plague doctors wore beaked masks",
        "related_presentedBy": " TED-Ed"
    }
]
```

Lambda Function TedTok: Get_Talk_List_By_Tags

Utilizzo

Questa Lambda Function, dato un array di tag, restituisce un JSON con tutti i talk corrispondenti a quei tag, ordinati in modo decrescente e senza duplicati.

Esperienza utente

Questo permette al nostro utente, una volta selezionati i tag di suo interesse, di visualizzarli tramite il sistema.

Codice della funzione: Get_Talk_List_By_Tags

```
const mongoose = require('mongoose');

const talk_schema = new mongoose.Schema({
    _id: String,
    speakers: String,
    title: String,
    url: String,
    duration: String,
    publishedAt: Date,
}, { collection: 'TedTok_data' });

module.exports = mongoose.model('talk', talk_schema);
```

Questo è il modello utilizzato per la visualizzazione dei dati (file Talk.js), il codice del gestore della funzione (file handler.js) è mostrato alla diapositiva successiva.

La lambda function, è disponibile attraverso questo link API: https://kz0253u01m.execute-api.us-east-1.amazonaws.com/default/Get_Talk_List_By_Tags

Codice della funzione: Get_Talk_List_By_Tags

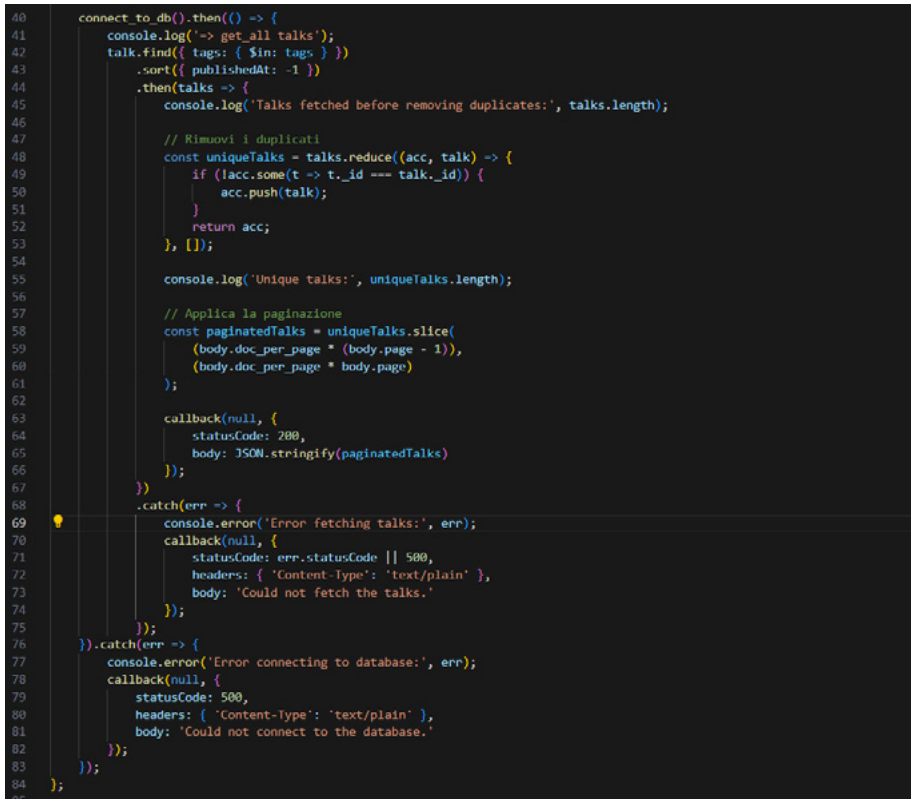
```
const connect_to_db = require('./db');
const talk = require('./Talk');

module.exports.get_by_tag = (event, context, callback) => {
    context.callbackWaitsForEmptyEventLoop = false;
    console.log('Received event:', JSON.stringify(event, null, 2));

    let body = {};
    if (event.body) {
        try {
            body = JSON.parse(event.body);
        } catch (error) {
            console.error('Error parsing body:', error);
            callback(null, {
                statusCode: 400,
                headers: { 'Content-Type': 'text/plain' },
                body: 'Invalid JSON body'
            });
            return;
        }
    }

    const tags = body.tags;
    if (!tags || !Array.isArray(tags) || tags.length === 0) {
        callback(null, {
            statusCode: 400,
            headers: { 'Content-Type': 'text/plain' },
            body: 'Tags are null or they are not an array.'
        });
        return;
    }

    if (!body.doc_per_page) {
        body.doc_per_page = 10;
    }
    if (!body.page) {
        body.page = 1;
    }
```



```
    connect_to_db().then(() => {
        console.log('=> get_all talks');
        talk.find({ tags: { $in: tags } })
            .sort({ publishedAt: -1 })
            .then(talks => {
                console.log('Talks fetched before removing duplicates:', talks.length);

                // Rimuovi i duplicati
                const uniqueTalks = talks.reduce((acc, talk) => {
                    if (!acc.some(t => t._id === talk._id)) {
                        acc.push(talk);
                    }
                    return acc;
                }, []);

                console.log('Unique talks:', uniqueTalks.length);

                // Applica la paginazione
                const paginatedTalks = uniqueTalks.slice(
                    (body.doc_per_page * (body.page - 1)),
                    (body.doc_per_page * body.page)
                );

                callback(null, {
                    statusCode: 200,
                    body: JSON.stringify(paginatedTalks)
                });
            })
            .catch(err => {
                console.error('Error fetching talks:', err);
                callback(null, {
                    statusCode: err.statusCode || 500,
                    headers: { 'Content-Type': 'text/plain' },
                    body: 'Could not fetch the talks.'
                });
            });
    }).catch(err => {
        console.error('Error connecting to database:', err);
        callback(null, {
            statusCode: 500,
            headers: { 'Content-Type': 'text/plain' },
            body: 'Could not connect to the database.'
        });
    });
};
```

Risultato della funzione: Get_Talk_List_By_Tags

Parametro in INPUT:

```
{
    "tags":["wildlife"]
}
```

OUTPUT testato tramite Postman:


```
[
  {
    "_id": "524079",
    "slug": "rebecca_mcmackin_let_your_garden_grow_wild",
    "speakers": "Rebecca McMackin",
    "title": "Let your garden grow wild",
    "url": "https://www.ted.com/talks/rebecca_mcmackin_let_your_garden_grow_wild",
    "description": "Many gardeners work hard to maintain clean, tidy environments ... which is the exact opposite of what wildlife wants,
    "duration": "742",
    "publishedAt": "2024-03-27T14:46:24.000Z",
    "tags": [
      "climate change",
      "environment",
      "animals",
      "nature",
      "biodiversity",
      "trees",
      "ecology",
      "plants",
      "gardening",
      "wildlife"
    ],
    "Related_videos": [
      {
        "related_video_ids": "125757",
        "related_video_title": "How poop turns into forests",
        "related_presentedBy": "Ludmila Rattis"
      },
```

Criticità tecniche

- Non vengono effettuati controlli sugli input, ossia che l'utente può inserire ciò che vuole senza vincoli.
- Potrebbero verificarsi tempi di computazione elevati riguardo all'ordinamento e alla rimozione dei duplicati

Possibili evoluzioni

- Dare maggiore dettaglio agli errori mostrando all'utente l'errore, se esistente.
- Assicurarsi che tutti gli input siano vincolati

TEDx TOk

Trello board

GitHub