GeoTEDx

Matteo Cesari (MAT. 1073570)

Davide Girolamo (MAT. 1073645)

GitHub: https://github.com/TurnTheBait/GeoTEDx

Lambda functions

Abbiamo realizzato due lambda functions:

- **Get_Watch_Next**: recupera il talk successivo da guardare dopo un talk specifico, basandosi sul numero di visualizzazioni, e restituisce le informazioni su tale talk.
- **Get_Talks_By_Location**: che, una volta che l'utente ha inserito la sua location, restituisce una serie di talks attinenti alla posizione indicata.

Get_Watch_Next

La funzione Lambda Get_Watch_Next riceve una richiesta contenente l'ID di un talk. La funzione cerca il talk corrispondente all'ID e quindi trova il talk successivo nella lista "watch_next" del talk corrente.

```
module.exports.getwatchnext = (event, context, callback) => {
    context.callbackWaitsForEmptyEventLoop = false;
    console.log('Received event:', JSON.stringify(event, null, 2));
    let body = {}
    if (event.body) {
        body = JSON.parse(event.body)
    }
    // set default
    if(!body.idx) {
        callback(null, {
                    statusCode: 500,
                    headers: { 'Content-Type': 'text/plain' },
                    body: 'Could not fetch the talks. idx is null.'
        })
    }

    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({_id: body.idx})
            .then(talks => {
                    talk.find({_id: talks[0].watch_next_s})
                    .sort('num_views')
                    .then(t => {
                        callback(null, {
                        statusCode: 200,
                        body: JSON.stringify({id: t[0]._id, title: t[0].title, details: t[0].details, url: t[0].url, num_views: t[0].num_views})
                    })
                })
            })
```

Per fare ciò, la funzione si connette al database e recupera il talk con l'ID fornito. Poi, utilizzando l'ID del talk corrente, cerca il talk successivo nella lista "watch_next".

Una volta trovato il talk successivo, vengono restituite alcune informazioni su di esso, come l'ID, il titolo, i dettagli, l'URL e il numero di visualizzazioni.

Get_Talks_By_Location

La funzione Lambda Get_Talk_By_Location accetta una richiesta HTTP con un parametro di posizione e restituisce un talk casuale associato a quella posizione.

Per prima cosa abbiamo creato delle location prefissate.

Fatto ciò abbiamo inserito i tag corrispondenti a ciascuna location.

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

  const { location } = JSON.parse(event.body);

  let tag = ['general'];

  if (location.toLowerCase() === 'university') {
    tag.push('education');
    tag.push('science');
    tag.push('technology');
    tag.push('personal growth');
    tag.push('innovation');
  } else if (location.toLowerCase() === 'work') {
    tag.push('business');
    tag.push('economics');
    tag.push('comunication');
    tag.push('leadership');
    tag.push('relationships');
    tag.push('society');
    tag.push('goals');
  } else if (location.toLowerCase() === 'home') {
    tag.push('culture');
    tag.push('family');
    tag.push('health');
    tag.push('parenting');
    tag.push('music');
  } else if(location.toLowerCase() ==='gym'){
    tag.push('motivation');
    tag.push('sports');
    tag.push('race');
    tag.push('music');
    tag.push('human body');
  } else if (location.toLowerCase() === 'park') {
    tag.push('nature');
    tag.push('biology');
    tag.push('climate change');
    tag.push('ecology');
    tag.push('environment');
    tag.push('sustainability');
  } else if (location.toLowerCase() === 'bed') {
    tag.push('memory');
    tag.push('sleep');
    tag.push('mental health');
    tag.push('brain');
    tag.push('storytelling');
  }
```

Get_Talks_By_Location

La funzione ottiene la posizione dalla richiesta HTTP e determina il tag corrispondente in base alla posizione. Successivamente, chiama il metodo "get_by_tag" passando il tag come parametro.

```
module.exports.get_by_tag = async (tag, callback) => {
  try {
    await connect_to_db();
    console.log('=> get_all talks');
    const uniqueTalks = [];
    const usedTags = new Set();

    for (const t of tag) {
      if (!usedTags.has(t)) {
        const talksForTag = await talk.aggregate([
          { $match: { tags: t } },
          { $sample: { size: 1 } },
        ]).exec();

        if (talksForTag.length > 0) {
          uniqueTalks.push(talksForTag[0]);
          usedTags.add(t);
        }
      }
    }
```

Il metodo "get_by_tag" effettua una query al database per ottenere un talk casuale associato al tag specificato. Infine, il talk casuale viene restituito come risposta nella callback della funzione Lambda.

Output

Per mostrare il funzionamento della Lambda function **Get_Talk_By_Location** abbiamo testato lo script su **Postman**.

Dando come input la location in formato JSON

```
{
    "location":"university"
}
```

Si ottiene un documento contente diversi talk estratti in modo casuale tra quelli aventi tag associato alla location indicata


```
{
    "_id": "270800e3be49a5548f236dc930919053",
    "main_speaker": "Fabio Pacucci",
    "title": "Newton's three-body problem explained",
    "details": "In 2009, researchers ran a simple experiment. They took everything we know about our solar system and calculated where every planet would be up to 5 billion years in the
        future. They ran over 2,000 simulations, and the astonishing variety in results revealed  that our solar system may be much less stable than it seems. Fabio Pacucci explores the
        n-body problem and the motion of gravitating objects. [Directed by Hype CG, narrated by Addison Anderson, music by Gabriel Maia].",
    "posted": "Posted Aug 2020",
    "url": "https://www.ted.com/talks/fabio_pacucci_newton_s_three_body_problem_explained",
    "num_views": "551,391",
    "duration": "5:09",
    "tags": [
        "education",
        "math",
        "physics",
        "astronomy",
        "universe",
        "talks",
        "science",
        "TED",
        "TED-Ed",
        "solar system",
        "Planets",
        "animation",
        "space"
    ],
    "watch_next": [
        "d46ce1af590233424313803da902361a",
        "80debe7a42c065ef82811c49ee5a56b7",
        "9f7b1654e792011b7e1c6f4288520226",
        "b90ae75a3da0fdf59912d5188fdd82fb",
        "58eb0fd9baae5f2b8f0e1aa089814efb",
        "657da08d854d99c0328e90b3306a9848",
        "19b65f184dd2bbf8862de0a054b1e747"
```


Sopra viene riportato un estratto dell'output contente un talk associato alla location "university" con tutti i vari dettagli.

Criticità e sviluppi futuri

- Location fornita dall’utente non correttamente riconosciuta dall’applicazione. In futuro si potrebbe incrementare la quantità di location per migliorare la precisione dei suggerimenti.
- Indirizzamento verso video non coerenti ai propri interessi e/o alla propria posizione a causa di tag errati.