TED IN FILA

Giulia Migliorati
Luca Frangiamore
Michael Marzella

Lambda Watch_Next

Lambda Watch_Next esamina il body della richiesta contenente **TitleNow**.

Recupera poi in modo casuale un elemento **dell'array watch-next**, presente nel database precedentemente elaborato con **AWS Glue**

```
try {
  console.log("mi sto connettendo")
  await connect_to_db();
  console.log("connesso")

  const TalkDetail =  (await talk.findOne({title:body.titleNow}));
  console.log(TalkDetail.watch_next.length)

  const totalTalks =await TalkDetail.watch_next.length;
  const randomIndex = Math.floor(Math.random() * totalTalks);
  console.log("numero massimo next:"+totalTalks+",indice randomico:"+randomIndex)

  callback(null, {
    statusCode: 200,
    body: JSON.stringify(TalkDetail.watch_next[randomIndex]),
  });
} catch (err) {
      callback(null, {
         statusCode: err.statusCode || 500,
         headers: { 'Content-Type': 'text/plain' },
         body: 'video non trovato',
      });
}
```

Lambda Range_Customers

```
if (body.orario === 'morning') {
  talk = require('./RangeCustomers_morning');
} else if (body.orario === 'afternoon') {
  talk = require('./RangeCustomers_afternoon');
}
```

Lambda Range_Customers esamina il body della richiesta contenente **Orario**.

In base al contenuto della richiesta va a collegarsi a due diverse **collections** del database create precedentemente tramite un **job di AWS Glue**. Recupera poi in modo casuale un video appartenente a esse

```
try {
  await connect_to_db();
  console.log('get talk ' + body.orario);
  const totalTalks = await talk.countDocuments();

  // Generate a random index within the range of total documents
  const randomIndex = Math.floor(Math.random() * totalTalks);

  // Retrieve a random talk using .find() and .skip()
  const randomTalk = await talk.find()
    .skip(randomIndex)
    .limit(1);

  callback(null, {
    statusCode: 200,
    body: JSON.stringify(randomTalk),
  });
} catch (err) {
  callback(null, {
    statusCode: err.statusCode || 500,
    headers: { 'Content-Type': 'text/plain' },
    body: 'video non trovato',
  });
```

Esperienza utente Watch_Next

- Il client può accedere ad un **video della lista watch-next** specificando all'interno della richiesta rest il **titolo del video appena finito**

```
{
  "titleNow": "It's OK to feel overwhelmed. Here's what to do next"
}
```

Esperienza utente Range_Customers

- Il client può accedere ad un **video filtrato in base alla fascia oraria** specificata nella richiesta rest

```
{
  "orario": "morning"
}
```

Criticità

- Divisione **schema mongoose** morning e afternoon
- Introduzione operatore **await** per ottenere il completamento del valore
- Introduzione funzione **random**

Possibili sviluppi

- Imporre dei controlli in modo da evitare che lo **stesso video si ripeta più volte** nel corso di poco tempo

- Mostrare all'utente una serie di possibili prossimi video da vedere, **dandogli la possibilità di sceglierne uno**