

TEDx-Perience



**Lambda
Functions**

Colombu Gabriele (1075574) - Odajiu Romeo (1073807)

Get_Watch_Next_by_Idx

La funzione "Get_Watch_Next_by_Idx" restituisce la lista dei dati ("**_id**", "**title**", "**url**") riguardanti i talk correlati ordinati per "**num_views**" associati al video avente "**_id**" passato come parametro.

```
connect_to_db().then(async () => {
  let next_talks_data_list = [];

  try{
    let talk = await talksModel.findOne({_id: body.id});
    let watch_next_ids = talk.watch_next_list;

    for(let i = 0; i < watch_next_ids.length; i++){
      let next_talk = await talksModel.findOne({_id: watch_next_ids[i]}, {title:1, url:1});

      next_talks_data_list.push(next_talk);
    }

    callback(null, {
      statusCode: 200,
      body: JSON.stringify(next_talks_data_list)
    })
  }
  catch(err){
    callback(null, {
      statusCode: err.statusCode || 500,
      headers: { 'Content-Type': 'text/plain' },
      body: 'Could not fetch the next talk data.'
    });
  }
});
```

Input:

- id: **_id** del talk di cui desideriamo visualizzare i correlati

```
{
  "id": "7e52bbc6379463aa0a6776d117b5fffd"
}
```

Output:

```
{
  {
    "_id": "c4b0ade4a4862ecfc44b19adb7db339d",
    "title": "Why can't we talk about periods?",
    "url": "https://www.ted.com/talks/jen_gunter_why_can_t_we_talk_about_periods"
  },
  {
    "_id": "a2a717eaffc4f504dc36dc908285207",
    "title": "How does the immune system work?",
    "url": "https://www.ted.com/talks/emma_bryce_how_does_the_immune_system_work"
  },
  {
    "_id": "3d09fa5a4a64a82554252244c9420355",
    "title": "6 tips for better sleep",
    "url": "https://www.ted.com/talks/matt_walker_6_tips_for_better_sleep"
  },
  null
}
```



Most_Liked



La funzione “**Most_Liked**” restituisce una lista dei video che hanno ricevuto più likes.

Funzionamento:

La funzione crea una lista contenente “**title**”, “**url**” e “**likes**” estratti dai primi **n** video (**n** = “**num_video**”) ordinati per “**likes**” in modo decrescente.

Abbiamo rinominato le voci dell’output in “*Title*”, “*URL*” e “*Likes*”.



```
connect_to_db().then(async () => {

  try{
    let most_liked_list = [];
    let talk_list = await talksModel.find({}, {_id:0, title: 1,url:1, likes:1}).sort(({likes:-1}).limit(body.num_video);

    talk_list.forEach((talk) =>{
      most_liked_list.push({
        Title: talk.title,
        URL: talk.url,
        Likes: talk.likes
      });
    });

    callback(null, {
      statusCode: 200,
      body: JSON.stringify(most_liked_list)
    })
  }
  catch(err){
    callback(null, {
      statusCode: err.statusCode || 500,
      headers: { 'Content-Type': 'text/plain' },
      body: 'Could not fetch the next talk data.'
    });
  }
});
```

Most_Liked - API Request

Input:

Abbiamo lasciato di default il valore 3. Altrimenti è possibile richiedere una profondità specificando "num_video".

Output:

Vengono visualizzati il titolo, l'url e il numero di like dei migliori video.



```
{
  "Title": "Do schools kill creativity?",
  "URL": "https://www.ted.com/talks/sir_ken_robinson_do_schools_kill_creativity",
  "Likes": 2100000
},
{
  "Title": "Your body language may shape who you are",
  "URL": "https://www.ted.com/talks/amy_cuddy_your_body_language_may_shape_who_you_are",
  "Likes": 1900000
},
{
  "Title": "Inside the mind of a master procrastinator",
  "URL": "https://www.ted.com/talks/tim_urban_inside_the_mind_of_a_master_procrastinator",
  "Likes": 1800000
}
```





Most_Viewed



La funzione **"Most_Viewed"** restituisce una lista dei video che hanno ricevuto più visualizzazioni.

Funzionamento:

La funzione crea una lista contenente **"title"**, **"url"** e **"avg_views"** estratti dai primi **n** video (**n** = **"num_video"**) ordinati per **"avg_views"** in modo decrescente.

Abbiamo rinominato le voci dell'output in **"Title"**, **"URL"** e **"num_views"**.



```
connect_to_db().then(async () => {

  try{
    let most_viewed_list = [];
    let talk_list = await talksModel.find({}, {_id:0, title: 1,url:1, num_views:1}).sort({avg_views:-1}).limit(body.num_video);

    talk_list.forEach((talk) =>{
      most_viewed_list.push({
        title: talk.title,
        URL: talk.url,
        num_views: talk.num_views

      });

    });

    callback(null, {
      statusCode: 200,
      body: JSON.stringify(most_viewed_list)
    })
  }
  catch(err){
    callback(null, {
      statusCode: err.statusCode || 500,
      headers: { 'Content-Type': 'text/plain' },
      body: 'Could not fetch the next talk data.'
    });
  }
});
```

Most_Viewed - API Request

Input:

Abbiamo lasciato di default il valore 3. Altrimenti è possibile richiedere una profondità specificando "num_video".

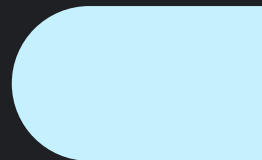
Output:

Vengono visualizzati il titolo, l'url e il numero di visualizzazioni dei migliori video.

```
{
  {
    "title": "Do schools kill creativity?",
    "URL": "https://www.ted.com/talks/sir_ken_robinson_do_schools_kill_creativity",
    "num_views": 70572226
  },
  {
    "title": "Your body language may shape who you are",
    "URL": "https://www.ted.com/talks/amy_cuddy_your_body_language_may_shape_who_you_are",
    "num_views": 61483714
  },
  {
    "title": "How great leaders inspire action",
    "URL": "https://www.ted.com/talks/simon_sinek_how_great_leaders_inspire_action",
    "num_views": 54865355
  }
}
```



Top_Speaker_by_Trending



La funzione “Top_speaker_by_Trending” restituisce una lista dei migliori speakers.



Funzionamento:

Creiamo due liste ordinate in modo decrescente: “talk_list_by_likes” e “talk_list_by_views” estraendo i primi n video (n=“num_video”) ed estraiamo il “main_speaker”.

Per ogni lista contiamo quante volte ogni speaker è presente, quindi filtrando per speaker, ritorniamo i nomi di coloro che compaiono più di una volta in ordine decrescente.



```
connect_to_db().then(async () => {  
  
  try{  
    let Top_Speaker_list_by_views = [];  
    let Top_Speaker_list_by_likes = [];  
    let talk_list_by_views = await talksModel.find({}, {id:0, title: 1, url:1, avg_views:1, main_speaker:1 }).sort({avg_views:-1}).limit(body.num_video);  
    let talk_list_by_likes = await talksModel.find({}, {id:0, title: 1, url:1, likes:1, main_speaker:1 }).sort({likes:-1}).limit(body.num_video);  
  
    talk_list_by_views.forEach((talk) =>{  
      Top_Speaker_list_by_views.push({  
        main_speaker: talk.main_speaker  
      });  
    });  
  
    talk_list_by_likes.forEach((talk) =>{  
      Top_Speaker_list_by_likes.push({  
        main_speaker: talk.main_speaker  
      });  
    });  
  
    let speakerCount = {};  
  
    Top_Speaker_list_by_views.forEach((speaker) => {  
      const mainSpeaker = speaker.main_speaker;  
      speakerCount[mainSpeaker] = (speakerCount[mainSpeaker] || 0) + 1;  
    });  
  
    Top_Speaker_list_by_likes.forEach((speaker) => {  
      const mainSpeaker = speaker.main_speaker;  
      speakerCount[mainSpeaker] = (speakerCount[mainSpeaker] || 0) + 1;  
    });  
  
    let Top_Speakers_by_Trending = Object.keys(speakerCount).filter((speaker) => {  
      return speakerCount[speaker] > 1;  
    });  
  
    callback(null, {  
      statusCode: 200,  
      body: JSON.stringify(Top_Speakers_by_Trending)  
    })  
  }  
  catch(err){  
    callback(null, {  
      statusCode: err.statusCode || 500,  
      headers: { 'Content-Type': 'text/plain' },  
      body: "Could not fetch the next talk data."  
    });  
  }  
});
```

Top_Speaker_by_Trending - API Request

Input:

Abbiamo modificato il valore di
"num_video" a 11.



Output:

Analizzando la top 11 di most
liked e viewed otteniamo la top
10 dei migliori speaker.

```
[  
  "Sir Ken Robinson",  
  "Amy Cuddy",  
  "Simon Sinek",  
  "Tim Urban",  
  "Brené Brown",  
  "Julian Treasure",  
  "Bill Gates",  
  "Sam Berns",  
  "Robert Waldinger",  
  "Cameron Russell"  
]
```


Funzionalità utente

Get_Watch_Next_by_Idx

Quando l'utente termina la visione di un video, gli verrà mostrata una lista di video correlati.

Most_Liked

Viene mostrata la lista dei video che hanno accumulato più like. L'utente può scegliere la profondità di ricerca.

Most_Viewed

Viene mostrata la lista dei video più visualizzati della piattaforma. L'utente può scegliere la profondità di ricerca.


Top_Speaker_by_Trending

L'utente può vedere una lista dei migliori speaker scelti per numero di like e visualizzazioni.



Criticità

- In base al criterio scelto per definire la lista dei **"Top_speaker_by_Trending"** si ottengono risultati differenti. Quello scelto da noi premia la consistenza.
- È stato necessario modificare il formato dei dati per garantire un corretto ordinamento.



Possibili Evoluzioni

- Definire un criterio di scelta dei top speaker migliore.
- Permettere all'utente di scegliere secondo quale criterio ordinare i video.
- Implementare nuove strategie per la ricerca dei video correlati, per esempio basate su tag e/o titoli pertinenti.
- Implementare la visualizzazione dei video relativi ai migliori speakers.



Link Utili:

TRELLO:

- <https://trello.com/invite/b/BKOVfpsB/ATTI5f54a246955ffbf71ce713d520e1ab97DDF7283/tedx-perience>

REPOSITORY GITHUB:

- <https://github.com/AndreaCremonesi4/TedX-Perience>