



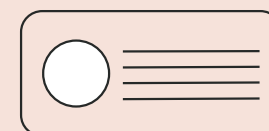
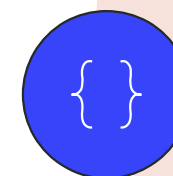
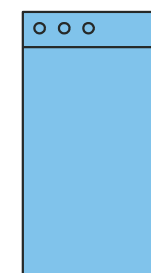
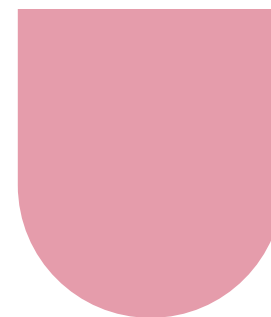
CommuniTEDx

Applicazione

Mobile

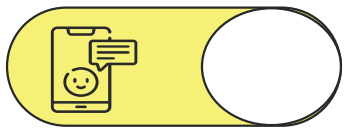
Progetto per il corso di
tecnologie Cloud e Mobile

Samuel Locatelli
Giorgio Tentori



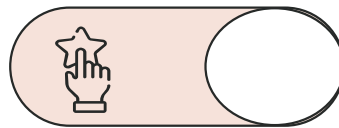
Descrizione e obiettivo di CommuniTEDx

CommuniTedx è un'applicazione di intrattenimento che ha come obiettivo principale quello di fornire agli utenti i contenuti a cui sono più interessati e creare una connessione con altri utenti con interessi comuni.



Community

Consente di conoscere e interagire con altri utenti della piattaforma tramite chat e collegamenti



Divulgazione

Favorisce la divulgazione scientifica e culturale attraverso la condivisione di video TEDx



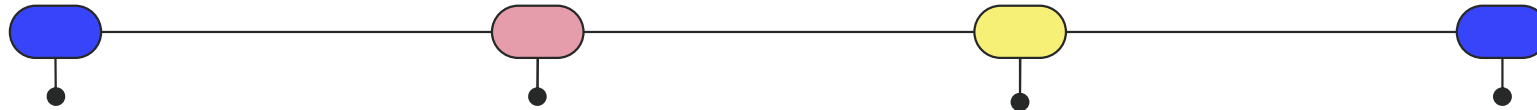
Intrattenimento

Permette un accrescimento delle conoscenze senza tralasciare l'aspetto divertimento !





Funzioni principali



Ricerca e filtraggio

Si possono cercare video, resi disponibili dalla piattaforma, in base al titolo, ad un particolare topic o al relatore

Suggerimenti e amicizie

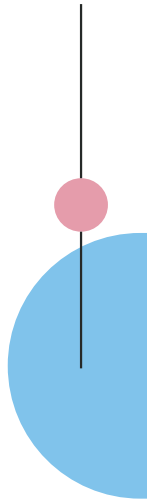
L'applicazione suggerisce determinati video in base alle proprie preferenze e consente di creare collegamenti con altri utenti

Salvataggio dei video TEDx

L'utente può creare, modificare ed eliminare playlist in cui è possibile salvare video TEDx a cui un utente è particolarmente interessato

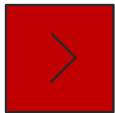
Chat e condivisione

Possibilità di condividere video con i propri collegamenti con l'ulteriore possibilità di iniziare una conversazione





Criticità



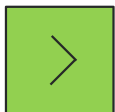
Possibilità di conversazione con tutti gli utenti o solo con i collegamenti?



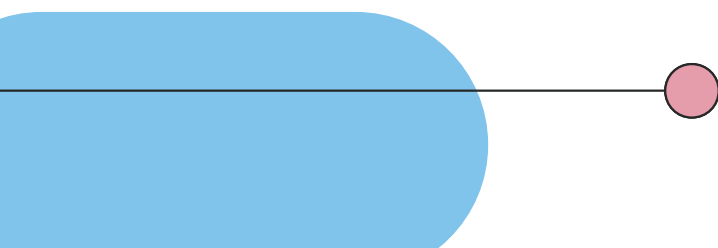
Quali sono i servizi a cui può accedere un utente non autenticato?



Possibilità di conversazione solamente con i propri collegamenti



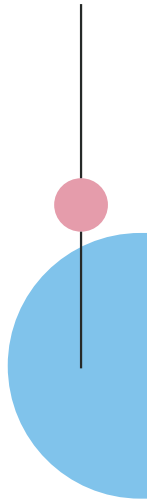
Accesso solamente a video più visti in quel periodo per gli utenti non autenticati





A chi è rivolto CommuniTEDx?

Questa applicazione è focalizzata su un pubblico con un forte interesse per la **scienza** e per la **tecnologia**, in particolare:

- Studenti delle scuole medie e superiori
 - Studenti universitari
 - Professori, docenti e insegnanti
 - Ricercatori e scienziati
 - Appassionati
- 

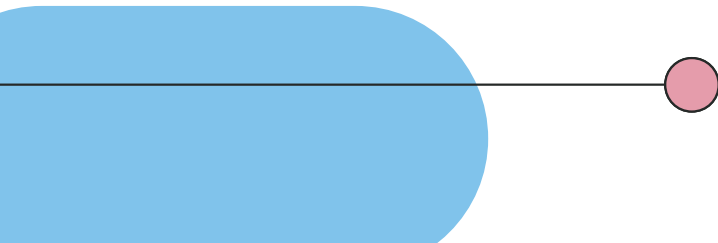


Note sull'app e sull'interfaccia grafica

L'interfaccia grafica dell'applicazione mobile viene realizzata tramite l'ausilio del framework flutter.

L'obiettivo è quello di rendere l'intera piattaforma *user-friendly*, in modo da permetterne l'utilizzo anche agli utenti meno esperti.

L'applicazione è costruita utilizzando tecnologie cloud, come i *tools* di Amazon AWS.





Approfondimento sul servizio di autenticazione alla piattaforma: Amazon Cognito

• •
• •
• •
• •
• •
• •
• •
• •

Per la funzione di autenticazione viene utilizzato il servizio Amazon Cognito. Esso si basa su due concetti principali:

- pool di utenti
- pool di identità

I pool di utenti servono per l'autenticazione (verifica dell'identità) mentre i pool di identità servono per l'autorizzazione (controllo degli accessi).





Puoi utilizzare i pool di identità per creare identità univoche per gli utenti e consentire loro l'accesso ad altri servizi AWS.

Gli utenti dell'app possono accedere tramite il pool di utenti o accedere in modalità federata, in particolare Amazon Cognito implementa lo standard *OAuth 2* che permette l'autenticazione tramite un gestore dell'identità digitale (IdP) di terze parti come Google o Facebook.

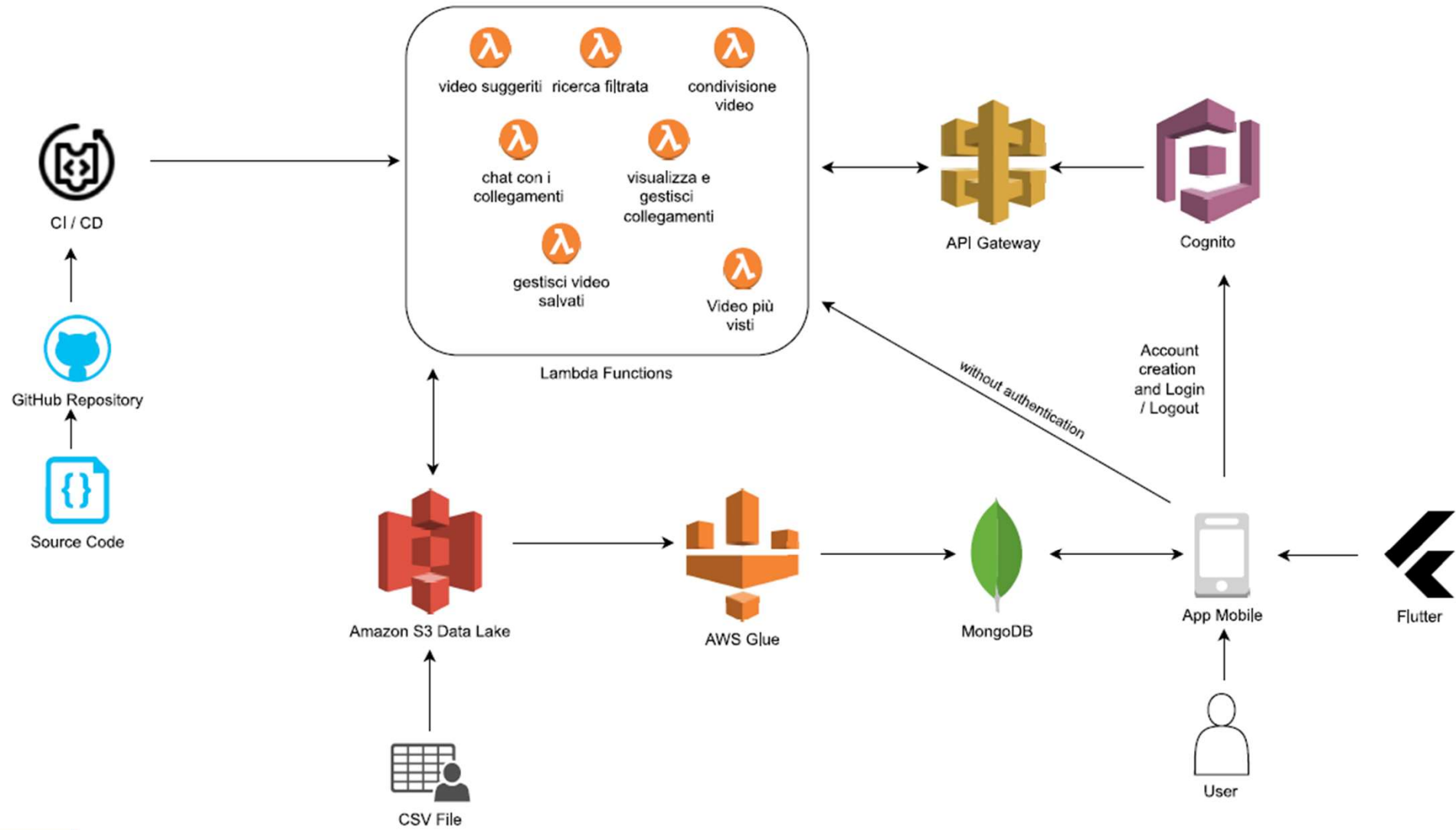
Queste funzionalità di Amazon Cognito sono utili perché permettono di:

- Progettare pagine web di iscrizione e accesso per l'app tramite un pool di utenza.
- Generare credenziali AWS temporanee per utenti non autenticati tramite un pool di identità.

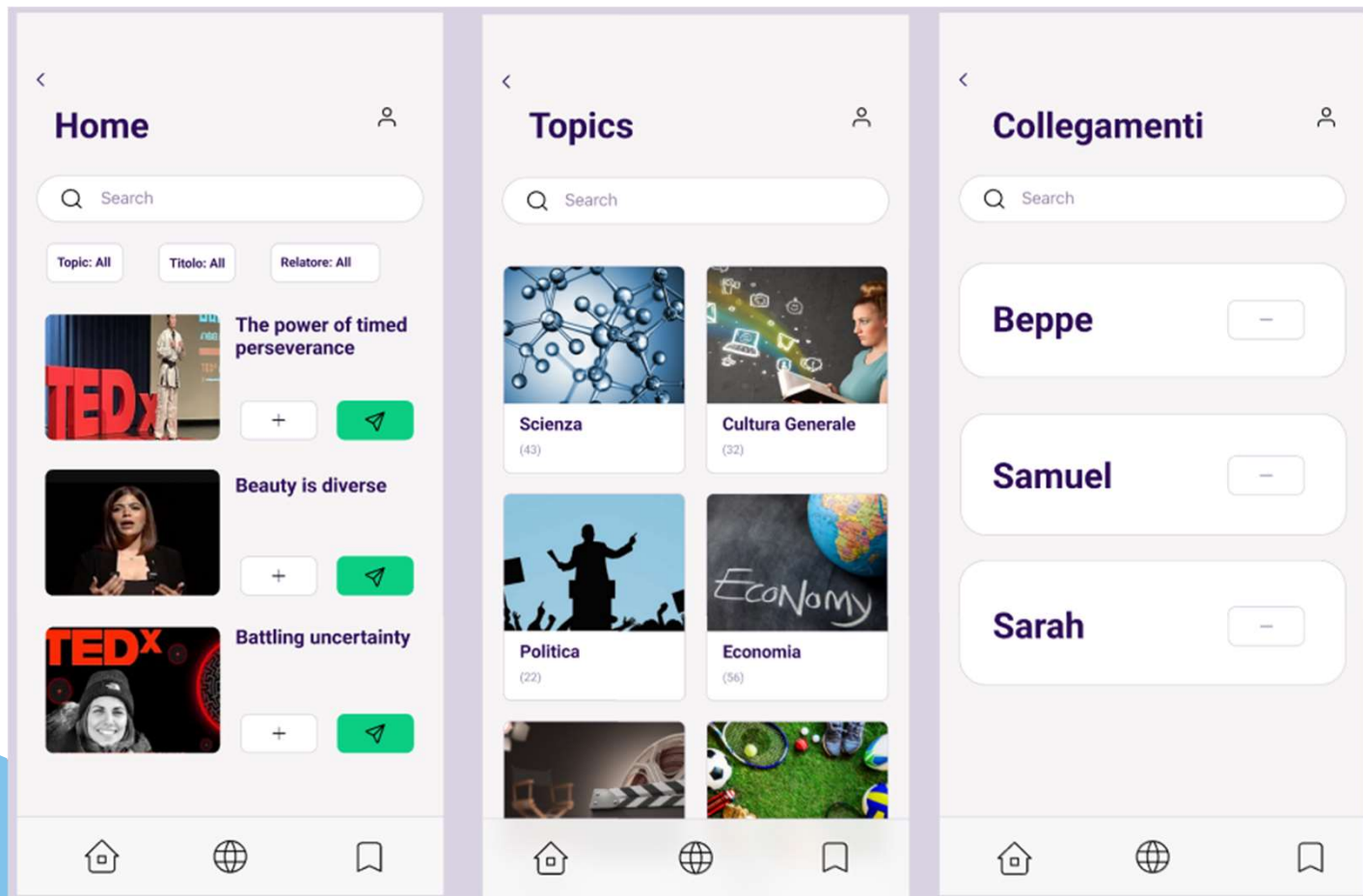
• •
• •
• •
• •
• •
• •
• •
• •



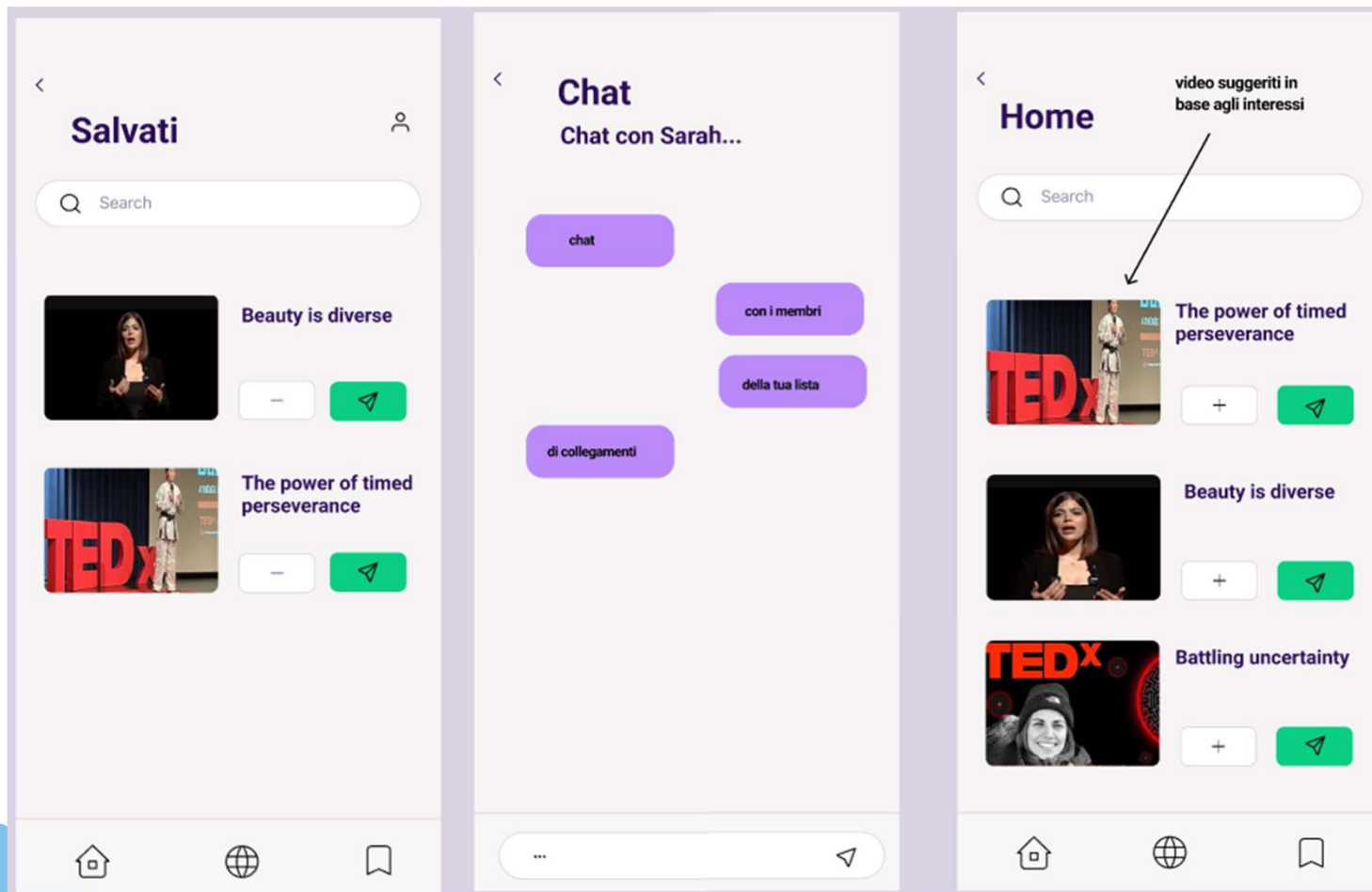
Architettura



Presentazione dell'interfaccia grafica



<https://www.figma.com/>



<https://www.figma.com/>

Trello Board

<https://trello.com/invite/b/0XzzqBXs/ATTI3aacf7431c519460e792bc42dcb6309d0EA2C9CA/communitedx>

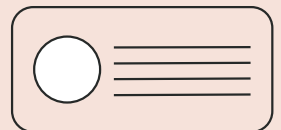
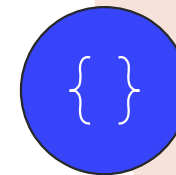
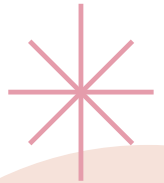
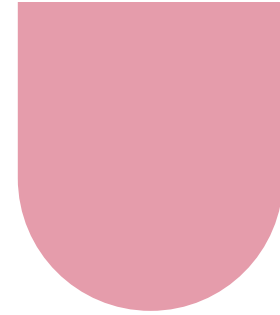
The screenshot shows a Trello board with five lanes. The first lane, titled 'Fatto', contains a list of completed tasks. The subsequent lanes, labeled 01 through 04, contain lists of tasks to be completed. Each lane has a header with a title and a three-dot menu icon. The tasks are listed in a vertical stack, with a '+ Aggiungi una scheda' button at the bottom of each lane.

Lane	Header	Tasks
Fatto	...	Riunione iniziale Descrizione del servizio Definire l'architettura Funzionalità del servizio Presentazione in Power Point Configurazione Git e GitHub con relativi comandi Punti critici del progetto Valore del servizio Fabbisogni e necessità del cliente Identificare i potenziali clienti ed eventuali stakeholder + Aggiungi una scheda
01	...	Tecnologie da utilizzare Disegnare l'interfaccia grafica ridefinire gli obiettivi dell'app approfondire i servizi utilizzati definire i dati che servono in aggiunta a Tedx + Aggiungi una scheda
02	...	Definire il modello dei dati su MongoDB Importare o caricare i dati su MongoDB Controllo qualità Caricare i dati su S3 + Aggiungi una scheda
03	...	Implementare la funzione cerca by video Design Lambda Functions API per ricercare i TED talk in base al topic Implementare la funzione cerca by tag (API) Login e logout Implementare la funzione playlist + Aggiungi una scheda
04	...	Definire piano di test Design App - Flutter Release App Implementare interfaccia Flutter + Aggiungi una scheda



PARTE 2

- 1) Aggiunta dei watch next videos
- 2) Gestione del dataset per rendere i dati conformi e coerenti con l'obiettivo dell'applicazione



Video suggeriti e Selezione dati

Nella seconda parte del progetto è stata implementata, in primo luogo, la possibilità di visualizzare i video associati ad un determinato Ted, quindi i video suggeriti in base al Ted che si sta guardando (watch next videos) e in secondo luogo la selezione dei dati presenti nel dataset in base ai criteri descritti nella prima parte del progetto e quindi video Ted relativi a **scienza** e **tecnologia** , scremando dall'intero insieme di dati solo quelli che servono per lo sviluppo del progetto.

A questo proposito viene inizializzato e implementato un job PySpark.
Il job è descritto in seguito.

Aggiunta del Related_videos Dataset

Il blocco di codice seguente dello script implementa la funzionalità che permette aggiungere al dataset iniziale, contenente i video Tedx, anche i watch next videos, cioè i video correlati e consigliati relativi ad un certo Ted video.

```
## READ WATCH_NEXT DATASET
watch_next_dataset_path = "s3://communitedx-2024-data/related_videos.csv"
watch_next_dataset = spark.read.option("header", "true").csv(watch_next_dataset_path)

# ADD WATCH_NEXT TO TEDX_DATASET
watch_next_dataset = watch_next_dataset.dropDuplicates()
watch_next_dataset_agg = watch_next_dataset.groupBy(col("id").alias("id_ref")).agg(collect_list("related_id").alias("id_next"), collect_list("title").alias("title_next"))
watch_next_dataset_agg.printSchema()

# AND JOIN WITH THE AGG TABLE
tedx_dataset_agg = tedx_dataset_agg.join(watch_next_dataset_agg, tedx_dataset_agg.id == watch_next_dataset_agg.id_ref, "left") \
    .drop("id_ref") \
    .select(col("id").alias("_id"), col("*")) \
    .drop("id") \
tedx_dataset_agg.printSchema()
```

Per ogni video Tedx si possono avere più video correlati e viene deciso di indicare per ogni documento in MongoDB due vettori: id dei video correlati e titolo dei video correlati.

Descrizione dello script

Lo script relativo al job PySpark presenta inizialmente la lettura del set di dati iniziale e il filtraggio dei record che contengono un id nullo.

```
#### READ INPUT FILES TO CREATE AN INPUT DATASET
tedx_dataset = spark.read \
    .option("header", "true") \
    .option("quote", "\"") \
    .option("escape", "\\") \
    .csv(tedx_dataset_path)

tedx_dataset.printSchema()

#### FILTER ITEMS WITH NULL POSTING KEY
count_items = tedx_dataset.count()
count_items_null = tedx_dataset.filter("id is not null").count()

print(f"Number of items from RAW DATA {count_items}")
print(f"Number of items from RAW DATA with NOT NULL KEY {count_items_null}")
```


Aggiunta Dettagli

Viene letto un altro dataset contenente i dettagli di ogni video come: descrizione e durata.

```
## READ THE DETAILS
details_dataset_path = "s3://communitedx-2024-data/details.csv"
details_dataset = spark.read \
    .option("header", "true") \
    .option("quote", "\"") \
    .option("escape", "\\") \
    .csv(details_dataset_path)

details_dataset = details_dataset.select(col("id").alias("id_ref"),
                                         col("description"),
                                         col("duration"),
                                         col("publishedAt"))
```

Il precedente dataset viene unito al precedente tramite 'Join'

```
# AND JOIN WITH THE MAIN TABLE
tedx_dataset_main = tedx_dataset.join(details_dataset, tedx_dataset.id == details_dataset.id_ref, "left") \
    .drop("id_ref")
```

Aggiunta Immagini

Si è deciso, come per i dettagli, di aggiungere i link delle immagini relativi ad un video Ted presi da un ulteriore dataset. Viene, in seguito aggiornato il dataset iniziale con quello delle immagini

```
## READ THE IMAGES
images_dataset_path = "s3://communitedx-2024-data/images.csv"
images_dataset = spark.read \
    .option("header", "true") \
    .option("quote", "\"") \
    .option("escape", "\\") \
    .csv(images_dataset_path)

images_dataset = images_dataset.select(col("id").alias("id_ref"),
                                       col("url").alias("image_url"))

# AND JOIN WITH THE MAIN TABLE
tedx_dataset_main = tedx_dataset_main.join(images_dataset, tedx_dataset_main.id == images_dataset.id_ref, "left") \
    .drop("id_ref") \

tedx_dataset_main.printSchema()
```

Aggiunta Immagini

Viene, infine, aggiornato il dataset finale con l'aggiunta di un vettore di tag identificativi relativi ad un video Tedx

```
## READ TAGS DATASET
tags_dataset_path = "s3://communitedx-2024-data/tags.csv"
tags_dataset = spark.read \
    .option("header","true") \
    .csv(tags_dataset_path)

# CREATE THE AGGREGATE MODEL, ADD TAGS TO TEDX_DATASET
tags_dataset_agg = tags_dataset.groupBy(col("id").alias("id_ref")).agg(collect_list("tag").alias("tags"))
tags_dataset_agg.printSchema()

# AND JOIN WITH THE MAIN TABLE
tedx_dataset_agg = tedx_dataset_main.join(tags_dataset_agg, tedx_dataset_main.id == tags_dataset_agg.id_ref, "left") \
    .drop("id_ref") \

tedx_dataset_agg.printSchema()
```

Gestione dei dati e filtraggio

Il dataset finale, a questo punto, deve essere aggiornato in modo da avere solamente i dati che sono interessanti al fine dell'obiettivo del progetto e quindi devono essere isolati e selezionati solo i dati relativi a **scienza** e **tecnologia**

Il seguente blocco di codice si occupa proprio di questo:

```
# FILTER TAG FROM THE DATASET
tedx_dataset_agg = tedx_dataset_agg.filter(array_contains(col("tags"),"technology") | array_contains(col("tags"),"science"))

tedx_dataset_agg.printSchema()
```

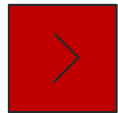
Vengono filtrati solamente i dati che interessano, controllando che tra i tag dei relativi video ci siano : ' science ' oppure ' technology '

Documento MongoDB

Dal dataset iniziale vengono esclusi i dati che non interessano al progetto.
Un documento si presenta in questo modo.

```
_id: "526880"  
slug: "george_zaidan_how_do_gas_masks_actually_work"  
speakers: "George Zaidan"  
title: "How do gas masks actually work?"  
url: "https://www.ted.com/talks/george_zaidan_how_do_gas_masks_actually_work"  
description: "You might think of gas masks as clunky military-looking devices. But i..."  
duration: "254"  
publishedAt: "2024-04-30T15:14:51Z"  
image_url: "https://talkstar-assets.s3.amazonaws.com/production/talks/talk_128547/..."  
▸ tags: Array (8)  
▸ id_next: Array (3)  
▸ title_next: Array (3)
```

Criticità



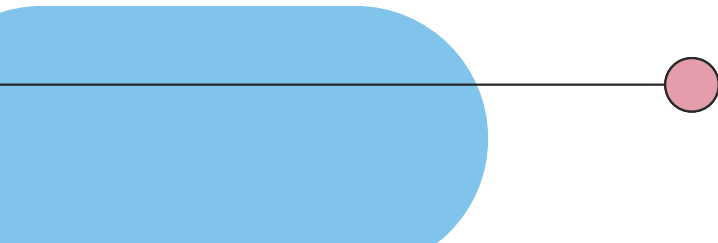
Tempistiche lente a causa dell'ambiente cloud



Iniziale difficoltà ad apprendere i concetti legati al linguaggio di programmazione



Difficoltà nel capire il comportamento di MongoDB con l'aggiornamento dei dati e quindi aumento costi di utilizzo





Sviluppi futuri

In un documento sono state inserite appositamente molte informazioni e dati, anche più di quelle che effettivamente servono. Questo perché si è pensato alle implementazioni future del progetto e alle eventuali evoluzioni che esso può avere.

