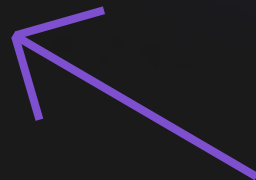




TEDx Idea Weaver

Da Catalogo di Talk a Rete di Conoscenza Interattiva



Romeo Odajiu - 1073807

Una costellazione di idee

L'universo TEDx è una vasta foresta di conoscenza, dove la ricerca tradizionale per parola chiave non è più sufficiente per orientarsi e scoprire connessioni profonde.

TEDx Idea Weaver non è un catalogo, ma una piattaforma di **meta-analisi** che trasforma l'intero dataset di talk in una mappa interattiva delle idee.

L'obiettivo è rivelare le connessioni nascoste tra argomenti, analizzare i **profili di specializzazione** dei relatori e abilitare una nuova forma di scoperta basata sull'esplorazione e sulla curiosità.





Obiettivi



01

Analisi Profonda

Andare oltre le metriche superficiali (views, likes) per analizzare le relazioni strutturali tra i contenuti.

02

Visualizzazione Intuitiva

Creare interfacce (come la "Costellazione delle Idee") che rendano queste complesse relazioni facili da esplorare.

03

Nuovi Percorsi di Scoperta

Permettere agli utenti di scoprire talk basandosi sul profilo analitico degli speaker o navigando "ponti" tematici tra argomenti diversi.






Stakeholders



Studenti e Ricercatori



Per trovare materiale interdisciplinare e connessioni non ovvie per le loro tesi e ricerche.

Lifelong Learners

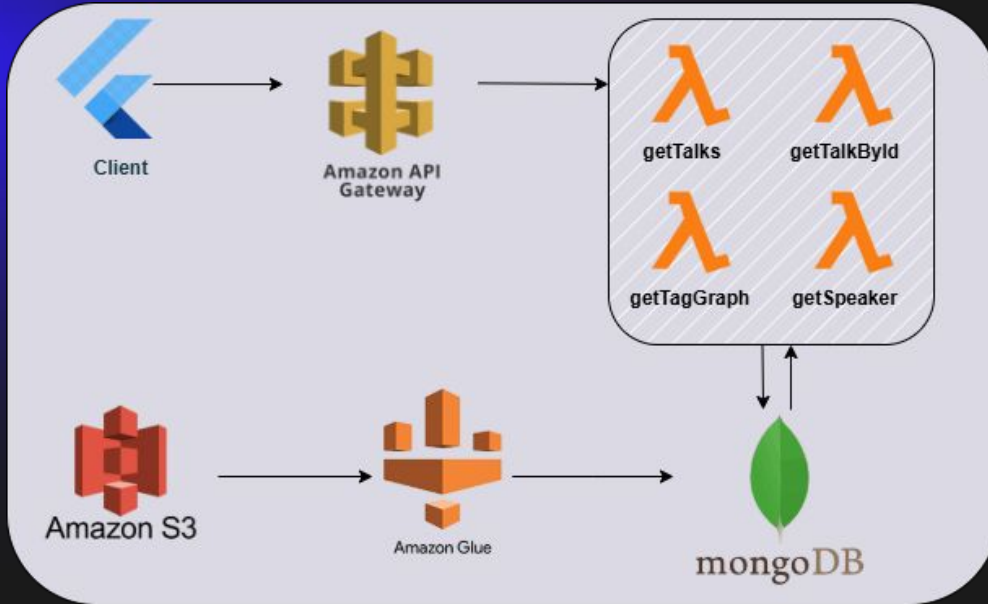
Per navigare la conoscenza in modo libero, seguendo percorsi dettati dalla propria curiosità intellettuale.

Content Creator e Organizzatori TEDx

Per analizzare quali argomenti sono più interconnessi, quali speaker sono più specializzati e quali sono i trend emergenti per pianificare eventi futuri.



Architettura



Flusso Dati (ETL - Offline):

- **1. S3:** I dati grezzi (CSV) vengono archiviati su Amazon S3.
- **2. Glue:** Uno script PySpark su AWS Glue processa e analizza i dati, calcolando weights e scores.
- **3. MongoDB:** I dati analitici vengono salvati su MongoDB, pronti per essere consultati.

Flusso Utente (API - Real-time):

- **1. Client (Flutter):** L'utente interagisce con l'app, inviando una richiesta.
- **2. API Gateway:** La richiesta viene ricevuta e instradata in modo sicuro.
- **3. Lambda:** Una funzione AWS Lambda dedicata esegue la logica e interroga il database.



ETL e Analisi con AWS Glue e Spark

Lo script PySpark eseguito da AWS Glue è il "cervello" analitico del progetto e opera in tre fasi

1. Unificazione Dati:

I cinque dataset CSV vengono letti da S3, puliti e uniti tramite una chiave comune per creare un documento completo e coerente per ogni talk, che andrà a popolare la collection talks.

```
# =====  
# 1. PREPARAZIONE DELLA COLLECTION 'talks'  
# =====  
  
images_agg = df_images.groupBy(JOIN_KEY).agg(collect_list(col("url")).alias("images"))  
tags_agg = df_tags.groupBy(JOIN_KEY).agg(collect_list(col("tag")).alias("tags"))  
videos_agg = df_related_videos.groupBy(JOIN_KEY).agg(  
    collect_list(  
        struct(  
            col("title").alias("video_title"),  
            col("duration").alias("video_duration"),  
            col("viewedCount").alias("video_views"),  
            col("presenterDisplayName").alias("video_speaker")  
        )  
    ).alias("related_videos")  
)  
  
talks_df = df_final_list \  
    .join(df_details, JOIN_KEY, "left") \  
    .join(images_agg, JOIN_KEY, "left") \  
    .join(videos_agg, JOIN_KEY, "left") \  
    .join(tags_agg, JOIN_KEY, "left") \  
    .withColumnRenamed(JOIN_KEY, "_id")  
  
talks_final_df = talks_df.select(  
    col("_id"), col("title"), col("presenterDisplayName").alias("speaker"),  
    col("publishedAt").alias("date"), col("duration"), col("url"),  
    col("description"), col("images"), col("related_videos"), col("tags")  
)
```



ETL e Analisi con AWS Glue e Spark



2. Analisi del Grafo:

Lo script analizza le co-occorrenze dei tag in tutti i talk. Per ogni coppia di tag che appare insieme, viene calcolato un **weight**, che rappresenta la forza della loro connessione tematica. Questi dati popolano la collection `tag_graph`.



```
# =====  
# 2. PREPARAZIONE DELLA COLLECTION 'tag_graph'  
# =====  
  
talk_tags_df = talks_final_df.select(col("_id"), col("tags")).dropna(subset=["tags"])  
  
combination_schema = ArrayType(StructType([  
    StructField("source", StringType(), False),  
    StructField("target", StringType(), False)  
]))  
  
@udf(combination_schema)  
def get_tag_combinations(tags):  
    if tags is None or len(tags) < 2: return []  
    sorted_tags = sorted(list(set(tags)))  
    return [{"source": combo[0], "target": combo[1]} for combo in combinations(sorted_tags, 2)]  
  
tag_pairs_df = talk_tags_df.withColumn("tag_pairs", get_tag_combinations(col("tags"))) \  
    .select(explode(col("tag_pairs")).alias("pair"))  
  
tag_graph_df_temp = tag_pairs_df.groupBy("pair.source", "pair.target") \  
    .agg(count(lit(1)).alias("weight"))  
  
tag_graph_df = tag_graph_df_temp.withColumn(  
    "_id",  
    concat_ws("-", col("source"), col("target"))  
).select("_id", "source", "target", "weight")
```





ETL e Analisi con AWS Glue e Spark

3. Profilazione Speaker:

I talk vengono raggruppati per relatore per calcolare lo **Specialization Score**, un indice che misura il grado di focalizzazione tematica di ogni speaker. Questi profili analitici popolano la collection `speaker_profiles`.

Lo **Specialization Score** indica se un relatore è uno **specialista** (score basso) o un **generalista** (score alto). È un valore tra 0 e 1.



```
# =====  
# 3. PREPARAZIONE DELLA COLLECTION 'speaker_profiles'  
# =====  
speaker_profiles_df = talks_final_df \  
    .filter(  
        col("speaker").isNotNull() &  
        (trim(col("speaker")) != "") &  
        (lower(trim(col("speaker")))) != "null"  
    ) \  
    .withColumn("tag_exploded", explode(col("tags"))) \  
    .groupBy("speaker") \  
    .agg(  
        collect_list(struct(col("_id"), col("title"))).alias("talks"),  
        collect_set("tag_exploded").alias("unique_tags_list"),  
        count("tag_exploded").alias("total_tags_count")  
    ) \  
    .withColumn("unique_tags_count", size(col("unique_tags_list"))) \  
    .withColumn(  
        "specialization_score",  
        round(  
            when(col("total_tags_count") > 0, col("unique_tags_count") / col("total_tags_count"))  
            .otherwise(0),  
            3  
        )  
    ) \  
    .withColumnRenamed("speaker", "_id")
```


Integrazione con MongoDB



Il database è organizzato in tre collection specializzate

talks:

Contiene i documenti completi di ogni talk, con tutti i dettagli, URL, immagini e tag associati.

tag_graph:

Archivia gli "archi" della nostra rete di idee. Ogni documento rappresenta un collegamento tra due tag e il suo peso.

speaker_profiles:

Contiene i profili analitici di ogni relatore, con la lista dei suoi talk e il suo punteggio di specializzazione.

```
{
  "_id": "567505",
  "title": "The true story of the iconic tagline \"Because I'm worth it.\" | The Fin_\"",
  "speaker": "Ben Proudfoot",
  "date": "2025-03-07T13:49:56Z",
  "duration": "1059",
  "url": "https://www.ted.com/talks/ben_proudfoot_the_true_story_of_the_iconic_t_\"",
  "description": "From two-time Oscar winner Ben Proudfoot comes THE FINAL COPY OF ILON _\"",
  "images": Array (3)
    0: "https://talkstar-assets.s3.amazonaws.com/production/talks/talk_147074/_\"",
    1: "https://talkstar-assets.s3.amazonaws.com/production/talks/talk_147074/_\"",
    2: "https://talkstar-assets.s3.amazonaws.com/production/talks/talk_147074/_\"",
  "related_videos": Array (3)
    0: Object
      video_title: "Why are women still taken less seriously than men?"
      video_duration: "769"
      video_views: "714134"
      video_speaker: "Mary Ann Sieghart"
    1: Object
    2: Object
  "tags": Array (8)
    0: "culture"
    1: "media"
    2: "social change"
    3: "marketing"
    4: "communication"
    5: "film"
    6: "women"
    7: "feminism"
```

```
{
  "_id": ObjectId('68768a421060154151dab5a1'),
  "source": "education",
  "target": "evolution",
  "weight": 54
}

{
  "_id": ObjectId('68768a421060154151dab5aa'),
  "source": "emotions",
  "target": "work-life balance",
  "weight": 24
}

{
  "_id": ObjectId('68768a421060154151dab5ae'),
  "source": "food",
  "target": "health",
  "weight": 83
}
```

```
{
  "_id": "Kristine McDivitt Tompkins",
  "talks": Array (27)
  "unique_tags_list": Array (13)
    0: "social change"
    1: "climate change"
    2: "South America"
    3: "animals"
    4: "philanthropy"
    5: "nature"
    6: "biodiversity"
    7: "natural resources"
    8: "Countdown"
    9: "environment"
    10: "conservation"
    11: "wildlife"
    12: "science"
  "total_tags_count": 27
  "unique_tags_count": 13
  "specialization_score": 0.481
}
```

API Serverless con Lambda

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

Handler

```
module.exports.getTalks = async (event) => {
  const queryParams = event.queryStringParameters || {};
  const page = parseInt(queryParams.page) || 1;
  const limit = parseInt(queryParams.limit) || 10;
  const tag = queryParams.tag;
```

```
  const query = tag ? { tags: tag } : {};
```

```
  try {
    await connect_to_db();

    const [talks, totalTalks] = await Promise.all([
      Talk.find(query)
        .sort({ date: -1 })
        .skip((page - 1) * limit)
        .limit(limit)
        .lean(),
      Talk.countDocuments(query)
    ]);
```

```
    const responsePayload = {
      totalTalks: totalTalks,
      currentPage: page,
      totalPages: Math.ceil(totalTalks / limit),
      data: talks
    };
  }
```

```
  return {
    statusCode: 200,
    headers: { "Content-Type": "application/json" },
    body: JSON.stringify(responsePayload, null, 2)
  };
}
```

```
} catch (err) {
  console.error("Errore durante il recupero dei talk:", err);
```

```
  return {
    statusCode: 500,
    headers: { "Content-Type": "application/json" },
    body: JSON.stringify({ message: 'Impossibile recuperare i talk.', error: err.message }, null, 2)
  };
}
```

Funzione getTalks -> GET /talks:

Restituisce una lista paginata di talk, con la possibilità di filtrare per tag. La risposta include metadati per la paginazione.

```
{
  "totalTalks": 7055,
  "currentPage": 1,
  "totalPages": 706,
  "data": [
    {
      "id": "567505",
      "title": "The true story of the iconic tagline \"Because I'm worth it.\" | The Final Copy of Ikon Spectr",
      "speaker": "Ben Proudfoot",
      "date": "2025-03-07T13:49:56Z",
      "duration": "1059",
      "url": "https://www.ted.com/talks/ben_proudfoot_the_true_story_of_the_iconic_tagline_because_i_m_worth_it_the_final_copy_of_ikon_spectr",
      "description": "From two-time Oscar winner Ben Proudfoot comes THE FINAL COPY OF IKON SPECTR, an intimate deathbed account of the unsung advertising genius who coined L'Oréal's iconic \"Because I'm Worth It\" slogan in 1973, creating a four-word feminist manifesto that, against all odds, changed advertising forever. Produced by Traverse32 and Breakwater Studios in partnership with L'Oréal Paris and McCann. (Contains mature language)",
      "images": [
        "https://talkstar-assets.s3.amazonaws.com/production/talks/talk_147074/d4647e7-5544-4b4e-a33c-144e62855770/ikon_spectr_16x9.jpg",
        "https://talkstar-assets.s3.amazonaws.com/production/talks/talk_147074/a3f95eb7-fd32-44d8-b0e6-0d5e69e78f5a/ikon_spectr_2x1.jpg",
        "https://talkstar-assets.s3.amazonaws.com/production/talks/talk_147074/a790f3be-8923-446e-8ee3-3313c1739494/ikon_spectr_4x3.jpg"
      ],
      "related_videos": [
        {
          "video_title": "Why are women still taken less seriously than men?",
          "video_duration": "769",
          "video_views": "714134",
          "video_speaker": "Mary Ann Sieghart"
        },
        {
          "tags": [
            "culture",
            "media",
            "social change",
            "marketing",
            "communication",
            "film",
            "women",
            "feminism"
          ]
        }
      ]
    }
  ]
}
```

Risposta

API Serverless con Lambda



Funzione getTalkById -> GET /talks/{id}:

Recupera un singolo talk utilizzando l'ID numerico fornito nel path dell'URL.

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

module.exports.getTalkById = async (event) => {
  await connect_to_db();
  const talkId = event.pathParameters.id;

  try {
    const talk = await Talk.findById(talkId);

    if (!talk) {
      return {
        statusCode: 404,
        body: JSON.stringify({ message: 'Talk non trovato.' }),
      };
    }

    return {
      statusCode: 200,
      headers: { "Content-Type": "application/json" },
      body: JSON.stringify(talk),
    };
  } catch (err) {
    console.error(`Errore durante il recupero del talk ${talkId}:`, err);
    return {
      statusCode: 500,
      body: JSON.stringify({ message: 'Impossibile recuperare il talk.', error: err.message }),
    };
  }
};
```

Handler

GET https://4407knyp00.execute-api.us-east-1.amazonaws.com/v1/talks/567505 Send

Params Authorization Headers (7) Body Scripts Settings Cookies

Body Cookies Headers (7) Test Results Visualize Save Response

{ } JSON Preview Visualize

Risposta

```
1 {
2   "_id": "567505",
3   "title": "The true story of the iconic tagline \"Because I'm worth it.\" | The Final Copy of Ilon Specht",
4   "speaker": "Ben Proudfoot",
5   "date": "2025-03-07T13:49:56Z",
6   "duration": "1059",
7   "url": "https://www.ted.com/talks/ben_proudfoot_the_true_story_of_the_iconic_tagline_because_i_m_worth_it_the_final_copy_of_ilon_specht",
8   "description": "From two-time Oscar winner Ben Proudfoot comes THE FINAL COPY OF ILON SPECHT, an intimate deathbed account of the unsung advertising genius who coined L'Oréal's iconic \"Because I'm Worth It\" slogan in 1973, creating a four-word feminist manifesto that, against all odds, changed advertising forever. Produced by Traverse32 and Breakwater Studios in partnership with L'Oréal Paris and McCann. (Contains mature language)",
9   "images": [
10     "https://talkstar-assets.s3.amazonaws.com/production/talks/talk_147074/db4647e7-5544-4b4e-a33c-144e62855770/Ilon_Specht_16x9.jpg",
11     "https://talkstar-assets.s3.amazonaws.com/production/talks/talk_147074/a3f95eb7-fd32-44d8-b0e6-0d5e69e78f5a/Ilon_Specht_2x1.jpg",
12     "https://talkstar-assets.s3.amazonaws.com/production/talks/talk_147074/a790f3be-8923-446e-8ee3-3313c1739494/Ilon_Specht_4x3.jpg"
13   ],
14   "related_videos": [
15     {
16       "video_title": "Why are women still taken less seriously than men?",
17       "video_duration": "769",
18       "video_views": "714134",
19       "video_speaker": "Mazy Ann Sieghart"
20     },
21     {
22       "video_title": "What if women built the world they want to see?",
23       "video_duration": "774",
```

API Serverless con Lambda



Funzione `getSpeakerProfile` -> `GET /speakers/{id}`:

Recupera il profilo analitico di un singolo speaker, decodificando il nome dall'ID nel path.

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

module.exports.getSpeakerProfile = async (event) => {
  await connect_to_db();

  const speakerId = decodeURIComponent(event.pathParameters.id);

  try {
    const profile = await SpeakerProfile.findById(speakerId);

    if (!profile) {
      return {
        statusCode: 404,
        body: JSON.stringify({ message: 'Profilo speaker non trovato.' }),
      };
    }

    return {
      statusCode: 200,
      headers: { 'Content-Type': 'application/json' },
      body: JSON.stringify(profile),
    };
  } catch (err) {
    console.error(`Errore durante il recupero del profilo di ${speakerId}:`, err);
    return {
      statusCode: 500,
      body: JSON.stringify({ message: 'Impossibile recuperare il profilo dello speaker.', error: err.message }),
    };
  }
};
```

Handler

GET `https://44o7knyp00.execute-api.us-east-1.amazonaws.com/v1/tasks/567506a%20E.%20Yar%20Khan` Send

Params Authorization Headers (7) Body Scripts Settings Cook

Body Cookies Headers (7) Test Results Preview Visualize

Risposta

200 OK · 190 ms · 1.04 KB Save Response

```
1 {
2   "_id": "Cara E. Yar Khan",
3   "talks": [
4     {
5       "_id": "311683",
6       "title": "The beautiful balance between courage and fear",
7       "unique_tags_list": [
8         "health",
9         "self",
10        "storytelling",
11        "fear",
12        "humanity",
13        "personal growth"
14      ],
15      "total_tags_count": 6,
16      "unique_tags_count": 6,
17      "specialization_score": 1
18    }
19  ]
20 }
```


API Serverless con Lambda



Funzione getTagGraph -> GET /graph/tags:

Restituisce l'intero set di dati che rappresenta le connessioni tra i tag, ovvero gli archi del grafo.

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

Handler

```
module.exports.getTagGraph = async (event) => {
  await connect_to_db();

  try {
    const graphData = await TagGraph.find({});

    return {
      statusCode: 200,
      headers: { "Content-Type": "application/json" },
      body: JSON.stringify(graphData),
    };
  } catch (err) {
    console.error("Errore durante il recupero del grafo dei tag:", err);
    return {
      statusCode: 500,
      body: JSON.stringify({ message: 'Impossibile recuperare il grafo dei tag.', error: err.message }),
    };
  }
};
```

```
{
  "_id": "68768a421060154151dab5aa",
  "source": "emotions",
  "target": "work-life balance",
  "weight": 24
},
{
  "_id": "68768a421060154151dab5ae",
  "source": "food",
  "target": "health",
  "weight": 83
},
{
  "_id": "68768a421060154151dab5b1",
  "source": "climate change",
  "target": "social change",
  "weight": 169
},
{
  "_id": "68768a421060154151dab5b3",
  "source": "social change",
  "target": "society",
  "weight": 480
}
```



Gestione della Configurazione e Codice Comune



La connessione al database è gestita da un modulo db.js standardizzato e replicato in ogni pacchetto di funzione.

L'interazione con le collection è resa sicura e strutturata tramite Modelli Mongoose dedicati (Talk.js, SpeakerProfile.js, TagGraph.js).

```
const mongoose = require('mongoose');
mongoose.Promise = global.Promise;
let isConnected;

const DB_URL = process.env.DB;
const DB_NAME = "tedx_weaver_db";

module.exports = connect_to_db = () => {
  if (isConnected) {
    console.log('=> using existing database connection');
    return Promise.resolve();
  }

  console.log('=> using new database connection');
  return mongoose.connect(DB_URL, {
    dbName: DB_NAME,
    useNewUrlParser: true,
    useUnifiedTopology: true
  }).then(db => {
    isConnected = db.connections[0].readyState;
  });
};
```

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

const talk_schema = new mongoose.Schema({
  _id: String,
  title: String,
  peaker: String,
  date: String,
  duration: String,
  url: String,
  description: String,
  images: Object,
  related_videos: Object,
  tags: Object
}, { collection: 'talks' });

module.exports = mongoose.model('talk', t
```

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

const graph_schema = new mongoose.Schema({
  _id: String,
  source: String,
  target: String,
  weight: Number
}, { collection: 'tag_graph' });

module.exports = mongoose.model('TagGraph', graph_schema);

const mongoose = require('mongoose');

const speker_schema = new mongoose.Schema({
  _id: String,
  talks: Object,
  unique_tags_list: Object,
  total_tags_count: Number,
  unique_tags_count: Number,
  specialization_score: Number
}, { collection: 'speaker_profiles' });

module.exports = mongoose.model('SpeakerProfile', speker_schema);
```




Funzionalità dell'Applicazione

Costellazione delle Idee: Una visualizzazione interattiva del grafo dei tag che permette all'utente di esplorare visivamente le connessioni tematiche tra migliaia di talk.

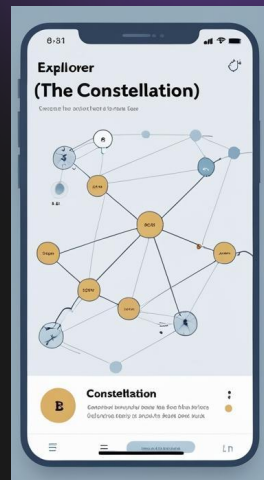
Profili Analitici degli Speaker: Pagine dedicate a ogni relatore che mostrano il loro portfolio completo di interventi e il loro "Specialization Score", per capire se si tratta di uno specialista verticale o di un pensatore poliedrico.

Scoperta Contestuale: La possibilità di navigare in modo fluido da un talk a un tag, da un tag a un altro argomento correlato, o da un talk al profilo del suo relatore, creando percorsi di scoperta personalizzati.

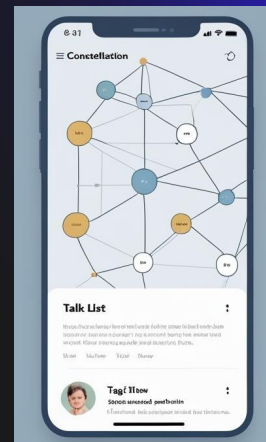
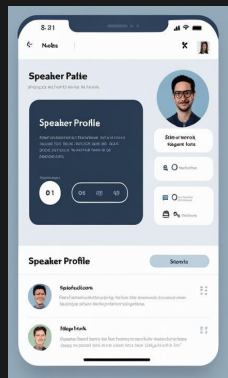


Wireframes e Schermate Principali

- **Explorer (La Costellazione):** La schermata principale mostra il grafo interattivo delle idee. L'utente può navigare (zoom, pan), cliccare su un "nodo" (un tag) per vedere le sue connessioni e visualizzare informazioni in una sidebar contestuale.
- **Profilo dello Speaker:** Una pagina dedicata che visualizza la biografia del relatore, il suo **Specialization Score** (magari con un indicatore grafico) e l'elenco completo dei suoi talk.



- **Lista dei Talk:** Selezionando un tag o un percorso, l'utente viene portato a una vista a lista che mostra tutti i talk pertinenti, con opzioni di filtro e ordinamento.





UX

L'obiettivo primario della UX è trasformare l'utente da semplice "spettatore" a "**esploratore della conoscenza**". L'intera esperienza è costruita per incoraggiare:

- **La Scoperta:** Invece di fornire solo risultati diretti a una ricerca, l'interfaccia stimola la curiosità, mostrando connessioni inaspettate e percorsi non lineari.

L'Approfondimento Contestuale: Ogni elemento (un talk, un tag, uno speaker) funge da "pivot" per esplorare la rete di informazioni correlate, creando un'esperienza di navigazione fluida e immersiva.



Criticità 1

Rischio di "Echo Chamber" Concettuale

Rischio: La visualizzazione del grafo, enfatizzando i collegamenti più forti (weight), potrebbe sovra-rappresentare le idee mainstream e oscurare le connessioni più uniche e innovative tra argomenti di nicchia.

Inconsistenza dei Dati di Input

Rischio: La presenza di dati errati nel campo speaker (frasi invece che nomi, modalità non uniforme nella compilazione del campo)

Duplicazione dei Dati nell'Analisi

Rischio: La lista dei talk per ogni relatore conteneva duplicati, un effetto collaterale indesiderato dell'analisi dei tag.



Criticità 2

Performance della Visualizzazione del Grafo

Il rendering di un grafo con migliaia di nodi e connessioni può essere estremamente pesante per il browser o il dispositivo mobile, causando lentezza e un'esperienza utente frustrante

Complessità del Layout Cross-Platform

Sebbene Flutter faciliti lo sviluppo multi-piattaforma, garantire un'esperienza utente ottimale e intuitiva su schermi molto diversi (dal mobile al web al desktop) è una sfida di design complessa.





Possibili Evoluzioni Future

Applicazione Cross-Platform con Flutter:

Sviluppo di un'applicazione client utilizzando il framework Flutter di Google. Questa scelta strategica permetterebbe di creare, a partire da un'unica codebase, un'esperienza utente nativa e performante sia per il mobile (iOS e Android) sia per il web.

Analisi Semantica con NLP:

Utilizzo di Natural Language Processing sui trascritti dei talk per creare connessioni ancora più profonde, basate non solo sui tag ma sul significato effettivo del testo.

Motore di Raccomandazione:

Integrazione con servizi di Machine Learning come AWS SageMaker per suggerire proattivamente talk o percorsi di scoperta personalizzati per l'utente.





Risorse del Progetto

- **Repository GitHub:**

https://github.com/RomeoOdajiuUnubg/Odajiu-TedX_IdeaWeaver/tree/main

- **Project Management Board:**

<https://trello.com/b/z5tcdWC4/tedx-idea-weaver>

