

UNIVERSITÀ DELLA CALABRIA

Dipartimento di ingegneria Informatica, Modellistica, Elettronica e
Sistemistica



CORSO DI LAUREA MAGISTRALE IN INGEGNERIA INFORMATICA

Progetto di Sistemi Distribuiti e Cloud Computing

Mattia Presta
239051

Anno Accademico 2023-2024

Indice

1	Introduzione	2
1.1	Vantaggi dell'applicazione	2
2	Analisi dei requisiti	3
2.1	Requisiti funzionali	3
2.2	Requisiti non funzionali	3
3	Progettazione	4
3.1	Google Cloud	4
3.2	Google Cloud SQL	5
3.2.1	Istanza Cloud SQL	5
3.3	Google Compute Engine (GCE) - VM	7
3.3.1	Creazione Istanza VM	7
3.3.2	Preparazione ambiente	8
3.3.3	Avvio Applicazione VM	8
4	Implementazione	9
4.1	Architettura WebApp	9
4.2	Caricamento File	10
4.3	Mail Processing	11
4.3.1	Word Frequency	11
4.3.2	NER - Named Entity Recognition	11
4.3.3	Sentiment Analysis - VADER	12
4.3.4	Topic Extraction - BERTopic	12
4.3.5	Summarization - BART	13
5	Dashboard	14
5.1	Widget	14
5.2	Graph Widget	16
6	View Page	17
6.1	Widget	19
7	Email Filters	21
8	Test e Risultati	22
9	Conclusioni	22

1 Introduzione

Oggi quasi tutte le persone possiedono uno o più account email, sia personale che per lavoro. Vengono raccolte centinaia di email al giorno per ogni persona, proveniente da centinaia di indirizzi riguardanti i più svariati ambiti, persone, pubblicità newsletter, giornali, social, figure professionali e tanto altro. I sistemi di gestione di email possiedono un filtraggio basato per mittente, oggetto della email e data, ma queste sono informazioni poco informative rispetto al contenuto della email, è possibile realizzare un'applicazione per estrarre più informazioni di qualità dalle email.

La gestione efficace delle informazioni contenute nelle email è una sfida crescente per molte aziende e individui. Le email spesso contengono una quantità significativa di dati non strutturati che, se analizzati correttamente, possono fornire preziose informazioni aziendali, sentimenti dei clienti, argomenti di tendenza e molto altro. Tuttavia, l'analisi manuale delle email è inefficiente e soggetta a errori. Per affrontare questa sfida, è stata sviluppata un'applicazione web basata su *Python* implementata con *Flask* che sfrutta le moderne tecniche di Natural Language Processing (NLP) e i modelli **Transformer** per automatizzare l'analisi delle email.

L'applicazione web permette agli utenti di caricare email, analizzarne il contenuto e ottenere informazioni dettagliate quali i termini più usati, i principali argomenti trattati, il sentiment, un riassunto e altre informazioni rilevanti. Questa applicazione sfrutta i servizi di Google Cloud, librerie open source per l'estrazione d'informazione e analisi del testo.

1.1 Vantaggi dell'applicazione

- **Risparmio di tempo:** automatizzando l'analisi delle email, gli utenti possono risparmiare tempo prezioso che può essere dedicato ad altre attività strategiche;
- **Miglioramento della Qualità dei Dati:** l'uso di modelli NLP avanzati garantisce una maggiore accuratezza nell'estrazione delle informazioni, migliorando la qualità dei dati raccolti.
- **Decisioni Informate:** Gli insights derivati dall'analisi delle email possono aiutare aziende ed individui a prendere decisioni più informate e strategiche.
- **Aumento della Produttività:** centralizzando e automatizzando l'analisi delle email, le aziende possono aumentare la produttività dei loro dipendenti.
- **Facilità di Utilizzo:** un'interfaccia utente intuitiva rende facile per gli utenti caricare email, effettuare ricerche e visualizzare i risultati dell'analisi.

2 Analisi dei requisiti

Una delle principali fasi per la realizzazione di un sistema distribuito è l'analisi dei requisiti perché fornisce una base solida per la successiva progettazione, sviluppo e test. Consiste nel definire e stabilire in modo efficace quelle che sono le esigenze, le aspettative e i vincoli del sistema da realizzare. Una corretta analisi dei requisiti permette di garantire che un progetto software soddisfi le esigenze dei suoi stakeholders e aiuta a evitare errori costosi. Gli stakeholder del sistema in questione non sono ben definiti in quanto si tratta di un'applicazione grossomodo generica, ma possono rientrare nella categoria:

- **Analisti di dati**
- **Partner commerciali**
- **Responsabili della sicurezza**
- **Ricercatori**

In questo contesto, quindi, sono stati definiti gli obiettivi del sistema: requisiti funzionali e requisiti non funzionali.

2.1 Requisiti funzionali

Per quanto riguarda i requisiti funzionali, ovvero le funzionalità che il sistema deve svolgere per soddisfare le esigenze degli utenti e raggiungere gli obiettivi prefissati, dalla traccia sono stati identificati i seguenti:

- **Sistema di Autenticazione:** l'utente ha bisogno di un account per analizzare i suoi file email;
- **Interfaccia web:** interfaccia per caricare i file analizzarli, visualizzarli e filtrarli;
- **Widget Dashboard:** la presenza di vari widget permette agli utenti di visualizzare informazioni specifiche o statistiche sulle email, anche tramite grafici;
- **Ricerca e raggruppamento:** dopo aver processato i file email, è possibile filtrare per mittente, topic, sentiment;

2.2 Requisiti non funzionali

A differenza dei requisiti funzionali, quelli non funzionali si concentrano sulle qualità globali del sistema. Sono fondamentali tanto quanto gli altri, perché essi contribuiscono a definire l'esperienza complessiva dell'utente

- **Scalabilità:** considerato che il progetto deve essere implementato come sistema distribuito, a maggior ragione bisogna tenere d'occhio questo requisito. Infatti, il sistema deve essere in grado di adattarsi a carichi di lavoro variabili;
- **Disponibilità:** il sistema deve essere disponibile e accessibile agli utenti per la maggior parte del tempo, evitando interruzioni o momenti di inattività;
- **Affidabilità:** si intende la capacità di un sistema di svolgere le sue funzioni in modo coerente e prevedibile nel corso del tempo, senza guasti o interruzioni. Il sistema deve essere tollerante ai guasti;
- **Usabilità:** è un aspetto critico, influenzato molto dall'interfaccia grafica, affinché si possa garantire un'esperienza positiva agli utenti, in particolare si riferisce alla facilità con cui gli utenti possono interagire con il sistema e sfruttare le sue funzionalità in modo intuitivo ed efficiente;
- **Compatibilità:** gli utenti devono avere la possibilità di accedere al sistema tramite i principali browser web e dispositivi più usati;

- **Prestazioni:** i tempi di risposta del sistema devono essere brevi anche quando si tratta di gestire un grande volume di dati;
- **Sicurezza:** bisogna prevedere un opportuno sistema di autenticazione e autorizzazioni adeguate.

3 Progettazione

Una volta individuati i requisiti, si è passati alla fase di progettazione dove, in funzione delle caratteristiche richieste, vengono definite le linee principali della struttura del sistema insieme alle scelte architetturali.

3.1 Google Cloud

Come richiesto dalla traccia la tecnologia di riferimento per lo sviluppo del progetto è Google Cloud, terzo fornitore di servizi cloud al mondo, offre risorse informatiche ospitate nei data center di Google in tutto il mondo gratuitamente o in base al consumo. Questa distribuzione delle risorse offre diversi vantaggi, tra cui la ridondanza in caso di errore e la riduzione della latenza posizionando le risorse più vicino ai client.

Google Cloud Platform è una piattaforma di cloud computing che offre una vasta gamma di strumenti e servizi per aiutare le aziende a innovare, gestire e scalare le loro applicazioni in modo efficiente e sicuro.

Tra i servizi principali vi sono:

- Servizi di elaborazioni quali:
 - **Google Compute Engine (GCE):** Macchine virtuali (VM) scalabili per eseguire carichi di lavoro in qualsiasi momento.
 - **Google Kubernetes Engine (GKE):** Servizio gestito per eseguire applicazioni containerizzate usando Kubernetes.
 - **App Engine:** Piattaforma come servizio (PaaS) per sviluppare e ospitare applicazioni web.
- Servizi di Storage e Database quali:
 - **Google Cloud Storage:** Servizio di storage scalabile e duraturo per oggetti.
 - **Google Cloud SQL:** Database relazionale gestito per MySQL, PostgreSQL e SQL Server.
 - **Google Cloud Spanner:** Database relazionale globale e scalabile.

Esistono diversi altri tipi di servizi riguardanti Big Data e Analytics, Machine Learning e AI ed anche servizi riguardanti la Sicurezza.

In particolare, i due servizi utili ai fini dello sviluppo del progetto sono stati: *Google Cloud SQL* e *Google Compute Engine* (VM).

3.2 Google Cloud SQL

Google Cloud SQL è un servizio di database relazionale completamente gestito che supporta MySQL, PostgreSQL e SQL Server. È progettato per semplificare la configurazione, la gestione e la scalabilità dei database relazionali nel cloud, eliminando la necessità di gestire hardware e software sottostante.

Utilizzato come parte di una strategia di backup e ripristino, grazie alla sua capacità di eseguire backup automatici e replica dei dati. Ottimo in termini di scalabilità verticale con possibilità di aumentare CPU, memoria e storage, e anche per scalabilità orizzontale con la replica di lettura per bilanciare il carico delle query.

Ai fini dello sviluppo del progetto è essenziale avere uno storage veloce ed efficace per caricare file contenenti email, e successivamente salvare i risultati delle analisi per effettuare query per la visualizzazione delle informazioni estratte.

3.2.1 Istanza Cloud SQL

Per creare un'istanza Cloud SQL su Google Cloud, basta accedere alla Console di Google Cloud e andare su "SQL". si clicca su "Create Instance" e si sceglie il motore di database desiderato.

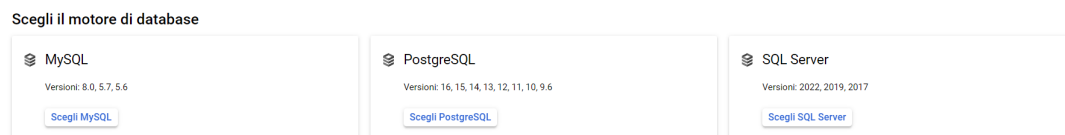


Figure 1: Istanze SQL

Esaminiamo i passi principali per la creazione dell'istanza

Riepilogo	
Versione Cloud SQL	Enterprise
Regione	europe-west2 (Londra)
Versione DB	MySQL 5.7
vCPU	1 vCPU
RAM	3,75 GB
Cache di dati	Disabilitata
Archiviazione	10 GB
Connessioni	IP pubblico
Backup	Automatico
Disponibilità	Zona singola
Recupero point-in-time	Abilitato
Velocità effettiva rete (MB/s)	250 di 250
Velocità effettiva disco (MB/s)	Lettura: 4,8 di 200,0 Scrittura: 4,8 di 200,0
IOPS	Lettura: 300 di 12.000 Scrittura: 300 di 10.000

Figure 2: Riepilogo Istanza SQL

- Piano di fatturazione: *Enterprise*
- Regione geografica di collocamento della macchina: (*europe-west2 (Londra)*)
- Utilizziamo un'istanza MySQL versione *5.7* per una maggiore compatibilità.
- Hardware della macchina: *1 vCPU*, *3,75 GB* Ram, *10 GB* SSD per archiviazione
- Connessione con un *IP pubblico*
- Selezioniamo il servizio di *Backup* e di *Recupero*

È importante specificare quale sarà l'IP della macchina che usufruirà dell'istanza, cioè l'IP della nostra istanza WepApp.



Figure 3: Reti Autorizzate

Possiamo passare alla creazione del nostro database che ospiterà i dati dell'applicazione.



Figure 4: Database

In conclusione ora abbiamo la nostra istanza SQL pronta per essere utilizzata, basterà effettuare una connessione lato back-end specificando *username, password, IP pubblico dell'istanza cloud e nome del database creato*. In questo modo:

```
app.config["SQLALCHEMY_DATABASE_URI"] = 'mysql+pymysql://root:****:@34.32.36.123/appdb'
```

Connessione Istanza Cloud SQL

3.3 Google Compute Engine (GCE) - VM

Google Compute Engine (GCE) è un servizio di cloud computing offerto da Google Cloud Platform che consente di creare e gestire macchine virtuali (VM) su infrastruttura di Google. Utilizzando GCE, gli sviluppatori possono sfruttare la scalabilità, l'affidabilità e le prestazioni elevate dell'infrastruttura globale di Google.

Le VM di GCE possono essere personalizzate in termini di CPU, memoria e spazio di archiviazione, permettendo di adattare le risorse alle esigenze specifiche delle applicazioni. Una delle principali applicazioni di GCE è l'hosting di web app, rendendole accessibili dall'esterno tramite indirizzi IP pubblici o bilanciatori di carico.

Inoltre, l'integrazione con altri servizi di Google Cloud, come Google Cloud SQL, facilita la gestione di database relazionali, consentendo di collegare facilmente le istanze VM alle istanze di database per un'architettura back-end robusta e scalabile.

Questo setup non solo ha migliorato la disponibilità e la sicurezza della tua applicazione, ma ha anche semplificato la scalabilità orizzontale, permettendoti di gestire picchi di traffico e di espandere le risorse on-demand. Grazie a queste caratteristiche, GCE rappresenta una soluzione potente per sviluppatori e aziende che desiderano implementare applicazioni cloud-native con alta disponibilità e performance.

3.3.1 Creazione Istanza VM

La creazione di un'istanza su Google Compute Engine (GCE) è un processo semplice che si può eseguire tramite la Console di Google Cloud. Dopo aver effettuato l'accesso, si seleziona "Compute Engine" e poi "VM instances". Si clicca su "Create Instance", configurando le risorse desiderate come CPU, memoria e sistema operativo.

Nome *
webapp

▼ GESTISCI TAG ED ETICHETTE

Regione *
europe-west10 (Berlino)
La regione è permanente

Zona *
Qualsiasi
Google sceglierà una zona per tuo conto, massimizzando l'ottenibilità della VM. La zona è permanente.

Tipo di macchina
Scegli un tipo di macchina con quantità predefinite di vCPU e memoria adatte alla maggior parte dei carichi di lavoro. In alternativa, puoi creare una macchina personalizzata per le esigenze specifiche del tuo carico di lavoro. [Scopri di più](#)

n2-standard-2 (2 vCPU, 1 core, 8 GB di memoria)

	vCPU	Memory
	2 (1 core)	8 GB

Disco di avvio

Nome	webapp
Tipo	Nuovo disco permanente bilanciato
Dimensioni	10 GB
Tipo di licenza	Gratuita
Immagine	Debian GNU/Linux 12 (bookworm)

Firewall

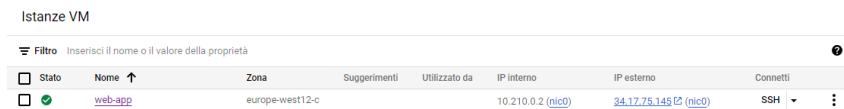
Aggiungi tag e regole firewall per consentire traffico di rete specifico da Internet

- ☒ Consenti traffico HTTP
- ☒ Consenti traffico HTTPS

Figure 5: Riepilogo Istanza VM

- Regione geografica di collocamento della macchina: *europe-west10 (Berlino)*
- Hardware della macchina: *2 vCPU*, *8 GB* Ram, *10 GB* SSD per archiviazione
- Regole Firewall: consenti traffico *HTTP*, *HTTPS*
- Immagine OS: *Debian GNU/Linux 12 (bookworm)*

Una volta creata l'istanza sarà visualizzata e pronta all'avvio.



Filtro <small>Inserisci il nome o il valore della proprietà</small>							
Stato	Nome ↑	Zona	Suggerimenti	Utilizzato da	IP interno	IP esterno	Connetti
<input checked="" type="checkbox"/>	web-app	europe-west12-c			10.210.0.2 (nic0)	34.17.75.145 (nic0)	SSH

Figure 6: Avvio Istanza VM

Da qui possiamo prendere l' *IP Pubblico* da inserire tra le reti autorizzate a connettersi all'istanza *Cloud SQL*.

3.3.2 Preparazione ambiente

Dopo aver creato l'istanza VM su Google Compute Engine, si accede ad essa tramite *SSH*. Prima di tutto, si aggiornano i pacchetti esistenti, si installa *Python*, *pip* e *git*. Carichiamo i file dell'applicazione tramite git clone.

```
mattiapresta15@ web-app:~$ git clone https://github.com/Mattia018/Email-Analysis.git
```

Codice Caricamento File git

Creiamo un ambiente virtuale *python venv*, e procediamo all'istallazione delle librerie necessarie contenute all'interno del file *requirements.txt*

```
mattiapresta15@ web-app:~$ python3 -m venv venv
mattiapresta15@ web-app:~$ python3 -m venv venv
mattiapresta15@ web-app:~$ source venv/bin/activate
mattiapresta15@ web-app:~$ pip install -r requirements.txt
```

Codice Preparazione Ambiente

3.3.3 Avvio Applicazione VM

Adesso la VM è pronta all'uso, l'applicazione sarà eseguibile con il comando:

```
mattiapresta15@ web-app:~$ python main.py
```

Codice Avvio Applicazione

4 Implementazione

Questa sezione riguarda l'effettiva realizzazione del sistema. Dopo avere introdotto gli obiettivi del progetto e il tipo di tecnologie utilizzate, di seguito sono riportate le varie fasi affrontate per portare a termine il progetto.

4.1 Architettura WebApp

Si è scelto di procedere alla realizzazione della web app tramite l'utilizzo di Python e libreria *Flask*, per lo storage è stato utilizzato *MySQL* tramite l'utilizzo della libreria *SqlAlchemy*.

Nello specifico:

- ***routes.py***: Questo file definisce le rotte web (endpoint) per l'applicazione. Gestisce le interazioni degli utenti e processa le richieste per analizzare le email.
 - **Caricamento e Analisi delle Email**: Gestione del caricamento dei file (email in formato CSV), elaborazione del contenuto delle email utilizzando modelli NLP e memorizzazione dei risultati dell'analisi nel database.
 - **Dashboard e Visualizzazione**: pagine web per la dashboard e la visualizzazione dei dati delle email analizzate.
- ***models.py***: Questo file definisce i modelli e schemi di dati utilizzati nell'applicazione, tramite SQLAlchemy per interagire con un database, memorizzare informazioni sugli utenti, il contenuto delle email e i risultati dell'analisi.
 - **Schema Utente**: Memorizza le informazioni relative agli utenti.
 - **Schema Email**: Memorizza i file caricati dagli utenti
 - **Schema MailAnalysisResult**: Memorizza i risultati delle varie analisi delle email
- ***features.py***: Questo file contiene le funzionalità principali per l'analisi delle email. Include varie funzioni per l'elaborazione del testo, l'analisi del sentiment, il riconoscimento delle entità nominate, la sintesi del testo e l'estrazione dei topic.
 - **Elaborazione del Testo**: Pulizia e pre-elaborazione del testo delle email.
 - **Analisi NLP**: Esecuzione dell'analisi del sentiment, riconoscimento delle entità nominate (NER), sintesi del testo e modellazione dei temi.
 - **Visualizzazione**: Creazione di visualizzazioni (ad esempio, grafici a barre) per mostrare i risultati dell'analisi.

4.2 Caricamento File

Tramite form viene chiesto all'utente di caricare un file CSV o txt contenente le mail nel formato: *From, Subject, Content*. Vengono effettuati dei controlli per l'autenticazione dell'utente, si verifica l'estensione del file corrispondente, si controlla se l'utente ha già caricato uno stesso file. Successivamente viene salvato su DB il file caricato dall'utente in formato binario TextBlob.

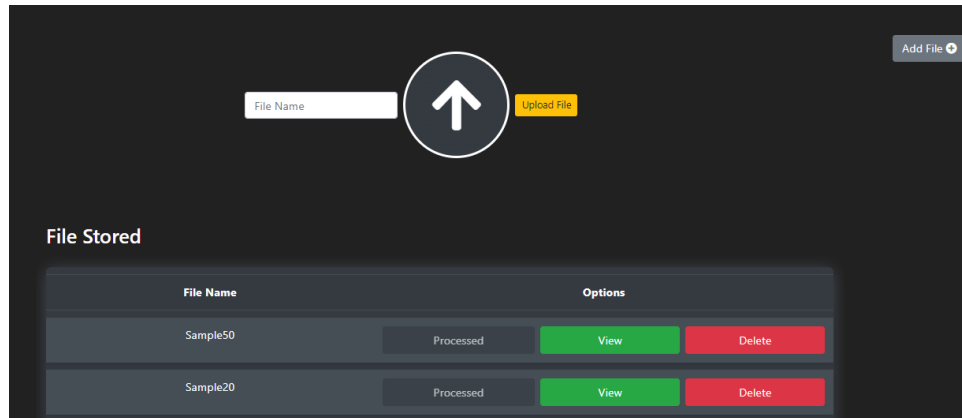


Figure 7: UI Caricamento File

```
form = UploadForm()
if request.method == "POST":
    if form.validate_on_submit():
        file_data = form.file.data.read()
        filename= Mails.query.filter_by(file_name=form.file_name.data).first()

        if filename is not None:
            if filename.file_name == form.file_name.data:
                flash(f"File name already exist, Please try a different File Name",
                    category='danger')
                return redirect(url_for('dashboard_page'))

            if file_data is None:
                flash(f"Please Upload a File", category='danger')
                return redirect(url_for('dashboard_page'))

            file_to_upload = Mails(file_name=form.file_name.data, file=file_data, owner=
current_user.id)
            db.session.add(file_to_upload)
            db.session.commit()
            flash(f"File Upload successfully", category='success')
            return redirect(url_for('dashboard_page'))
        else:
            flash(f"Only csv or txt file are allowed !", category='danger')
```

Codice Caricamento File

4.3 Mail Processing

Il passo successivo è l'estrazione delle informazioni all'interno del corpo della email.

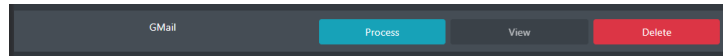


Figure 8: Analisi File

4.3.1 Word Frequency

Viene calcolata la frequenza delle parole escludendo stop-word e punteggiatura nel testo della email. Preso in input il file da processare viene utilizzato il modello NLP **SpaCy**, vengono estratti i token dal testo e tramite counter vengono selezionate le parole più frequenti.

```
doc = nlp(mail_content)
word_freq = Counter(token.text for token in doc if not token.is_stop
                    and not token.is_punct)
top_10_words = ' '.join([word for word, _ in word_freq.most_common(10)])
top_10_words = re.sub(r'^a-zA-Z0-9\s', '', top_10_words)
```

Codice Word Frequency

4.3.2 NER - Named Entity Recognition

Il riconoscimento delle entità nominate è il passo iniziale nel recupero delle informazioni. Questa tecnica è usata per estrarre informazioni importanti e utili da documenti di testo non strutturati. Riconoscimento di entità nominate **NER** ha l'obiettivo di localizzare e identificare le entità nominate presenti nel testo non strutturato, in categorie standard come nomi di persone, luoghi, organizzazioni, espressioni temporali, quantità, valori monetari, percentuali, numeri di telefono, email ecc. SpaCy è dotato di un'entità statistica estremamente veloce che assegna etichette a porzioni contigue di token.

```
def extract_ner(text):
    doc = nlp(text)
    named_entities_set = set()
    for ent in doc.ents:
        if ent.label_ not in ['PHONE', 'EMAIL']:
            named_entities_set.add(f'{ent.text} ({ent.label_})')

    email_pattern = r'\b[A-Za-z0-9._%+~]+@[A-Za-z0-9.-]+\.[A-Z|a-z]{2,}\b'
    emails = re.findall(email_pattern, text)
    for email in emails:
        named_entities_set.add(f'{email} (EMAIL)')

    phone_pattern = r'\b(\d{3}[-.\s]??\d{3}[-.\s]??\d{4}|\(\d{3}\)\s*\d{3}[-.\s]??\d{4}|\d{3}[-.\s]??\d{4})\b'
    phone_numbers = re.findall(phone_pattern, text)
    for number in phone_numbers:
        named_entities_set.add(f'{number} (PHONE)')

    named_entities_str = ' '.join(named_entities_set)

    return named_entities_str
```

Codice NER

4.3.3 Sentiment Analysis - VADER

VADER sta per *Valence Aware Dictionary e Sentiment Reasoner*. È uno strumento utilizzato per l'analisi del sentiment, riesce a distinguere se un pezzo di testo esprime emozioni positive, negative o neutre. VADER è particolarmente bravo a comprendere il sentiment nei testi dei social media, come tweet e commenti online, e si adatta molto bene anche al contenuto delle email. Funziona esaminando un elenco di parole con connotazioni positive o negative. VADER non si limita a dire positivo o negativo, assegna un punteggio compreso tra -1 (il più negativo) e +1 (il più positivo) per mostrare quanto è forte il sentimento. Tiene conto di caratteri come le maiuscole e la punteggiatura: VADER capisce che un punto esclamativo può rendere una parola positiva ancora più positiva, o che un "bravo" sarcastico in realtà significa il contrario.

```
def sentiment_vader(text, pos_threshold=0.55, neg_threshold=-0.05):
    analyzer = SentimentIntensityAnalyzer()
    result = analyzer.polarity_scores(text)

    compound_score = result['compound']
    if compound_score >= pos_threshold:
        main_sentiment = 'Positive'
    elif compound_score <= neg_threshold:
        main_sentiment = 'Negative'
    else:
        main_sentiment = 'Neutral'

    return main_sentiment
```

Codice Sentiment

4.3.4 Topic Extraction - BERTopic

Il topic modeling è una tecnica non supervisionata per estrarre argomenti da un insieme di documenti noto come corpus. LDA (Latent Dirichlet Allocation) è da tempo l'algoritmo più popolare per la modellazione degli argomenti da quando è stato introdotto. Queste tecniche vedono i documenti come una raccolta di argomenti latenti che a loro volta sono costituiti da parole. Si basano sulle co-occorrenze statistiche e sulle frequenze delle parole nel corpus dato. Con lo sviluppo di embeddings di parole basati su reti neurali e di architetture NLP avanzate come Transformers, l'approccio è stato rinnovato.

BERTopic si basa sugli embeddings di BERT pre-trained, da qui il nome. Ci sono tre passaggi per convertire i documenti in parole-topic. Innanzitutto, converte il testo in embeddings, quindi riduce la loro dimensionalità, li raggruppa e infine li converte in topic e parole. Nello specifico è stato usato il modello pre-trained **BERTopic_Wikipedia**, che riesce a riconoscere 2377 topic diversi ed è stato allenato su 1 Milione di documenti

```
topic_model = BERTopic.load("MaartenGr/BERTopic_Wikipedia")
def topic_Bert(text):
    topic_model.verbose = False
    topics, probs = topic_model.transform([text])
    topic = topics[0]
    topic_label = topic_model.topic_labels_[topic]

    return topic_label
```

Codice BERTopic

4.3.5 Summarization - BART

Il riepilogo del testo è un'attività di elaborazione del linguaggio naturale (NLP) che prevede la condensazione di un lungo documento di testo in una versione più breve e compatta pur conservando le informazioni e il significato più importanti. L'obiettivo è produrre un riassunto che rappresenti accuratamente il contenuto del testo originale in forma concisa.

Esistono diversi approcci al riepilogo del testo, inclusi metodi estrattivi che identificano ed estraggono frasi o frasi importanti dal testo e metodi astrattivi che generano nuovo testo in base al contenuto del testo originale. **BART** è un modello sviluppato da Facebook AI basato su transformer pre-addestrato attraverso un autoencoder di denoising, il che significa che è ottimo per ricostruire frasi pulite da quelle rumorose.

BART (*Bidirectional and Auto-Regressive Transformers*) è un modello che ha mostrato ottime prestazioni in varie attività di elaborazione del linguaggio naturale, incluso il riepilogo astrattivo del testo. Nello specifico è stato utilizzato **distilBART-xsum**, un modello fine-tuned sul dataset XSum.

```
model = BartForConditionalGeneration.from_pretrained("sshleifer/distilbart-xsum-12-1")

def summary_text_t5(text):

    tokenizer = AutoTokenizer.from_pretrained("sshleifer/distilbart-xsum-12-1")
    inputs = tokenizer([text], max_length=768, return_tensors="pt")
    summary_ids = model.generate(inputs["input_ids"], num_beams=3, min_length=10,
                                max_length=40, do_sample=False)
    result= tokenizer.batch_decode(summary_ids, skip_special_tokens=True,
                                   clean_up_tokenization_spaces=False)
    summary_text = ' '.join(result)

    return summary_text
```

Codice BERTopic

5 Dashboard

Sulla dashboard dedicata all'utente sono presenti tutti i file salvati e processati dall'utente, dei widget che restituiscono i risultati delle query riguardanti le principali informazioni estratte da tutte le email processate. È disponibile la possibilità di effettuare ricerche filtrate e visualizzare le analisi di ogni file caricato.

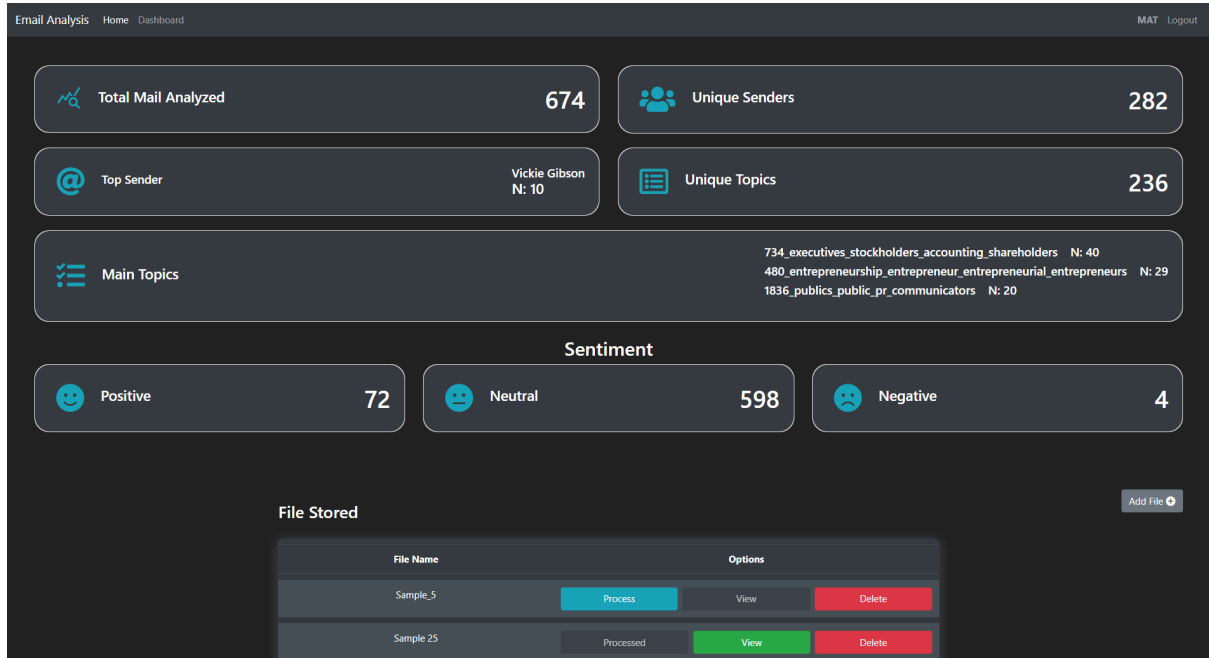


Figure 9: Dashboard

5.1 Widget

Prese tutte le email contenute nei file email relativi all'utente loggato, vengono eseguite delle query per visualizzare:

- **Total Mail Analyzed:** visualizza il numero complessivo di email processate dall'utente

```
total_emails_analyzed = MailAnalysisResult.query.  
    filter_by(user_id=current_user.id).count()
```

Total Mail Analyzed

- **Unique Senders:** visualizza il numero complessivo di mittenti unici

```
unique_senders_count = db.session.query(db.func.count(db.func.distinct(  
    MailAnalysisResult.mail_from))  
    ).join(Mails, MailAnalysisResult.file_id == Mails.id)  
    .filter(  
        Mails.owner == current_user.id  
    ).scalar()
```

Unique Senders

- **Top Sender:** visualizza il nome/email e il count del mittente che ha inviato il maggior numero di email

```
top_sender = db.session.query(
    MailAnalysisResult.mail_from, db.func.count(
        MailAnalysisResult.id).label('email_count')
    ).join(
        Mails, MailAnalysisResult.file_id == Mails.id
    ).filter(
        Mails.owner == current_user.id
    ).group_by(
        MailAnalysisResult.mail_from
    ).order_by(
        db.desc('email_count')
    ).all()
```

Top Senders

- **Unique Topics:** visualizza il numero di topics unici

```
unique_topics_count = db.session.query(
    db.func.count(db.func.distinct(MailAnalysisResult.mail_topic))
    ).join(Mails, MailAnalysisResult.file_id == Mails.id).filter(
        Mails.owner == current_user.id
    ).scalar()
```

Unique Topics

- **Main Topics:** visualizza le etichette e i count relativi ai 3 main topics estratti

```
main_topics = db.session.query(MailAnalysisResult.mail_topic,
    db.func.count(MailAnalysisResult.mail_topic)) \
    .filter_by(user_id=current_user.id) \
    .group_by(MailAnalysisResult.mail_topic) \
    .order_by(db.func.count(MailAnalysisResult.mail_topic).desc()) \
    .limit(3) \
    .all()
```

Main Topics

- **Sentiment:** visualizza il numero di email classificate con l'etichette *Positive*, *Neutral* e *Negative*

```
positive_sentiment_count = MailAnalysisResult.query.join(Mails).filter(
    Mails.owner == current_user.id,
    MailAnalysisResult.mail_Sent == 'positive'
).count()

neutral_sentiment_count = MailAnalysisResult.query.join(Mails).filter(
    Mails.owner == current_user.id,
    MailAnalysisResult.mail_Sent == 'neutral'
).count()

negative_sentiment_count = MailAnalysisResult.query.join(Mails).filter(
    Mails.owner == current_user.id,
    MailAnalysisResult.mail_Sent == 'negative'
).count()
```

Main Topics

5.2 Graph Widget

Sono stati implementati due grafici a barre per visualizzare la frequenza dei topics e il count di ogni sentiment all'interno delle email processate

- **Topics Frequency:**

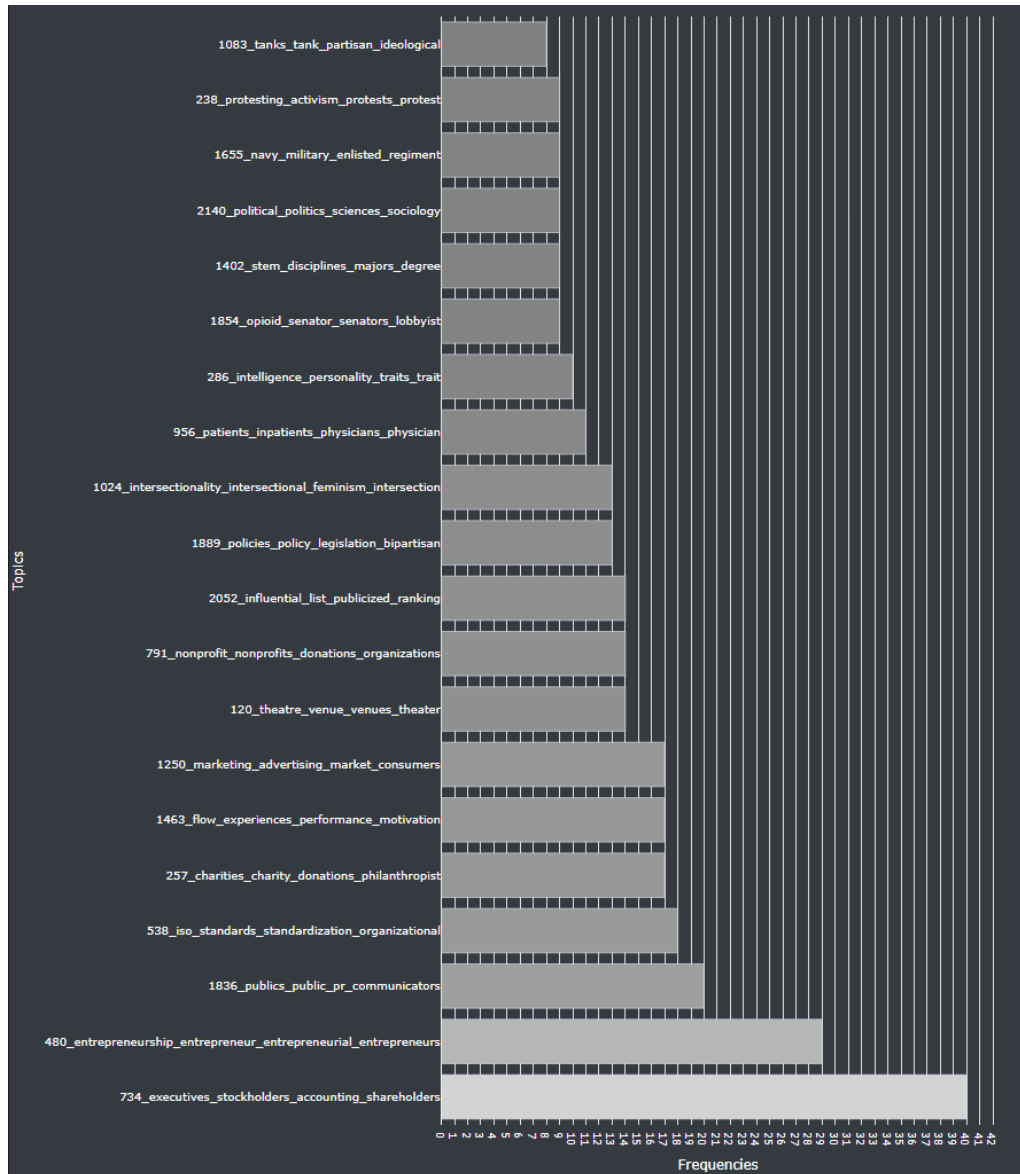


Figure 10: Topics Frequency

- **Sentiment Counts:**

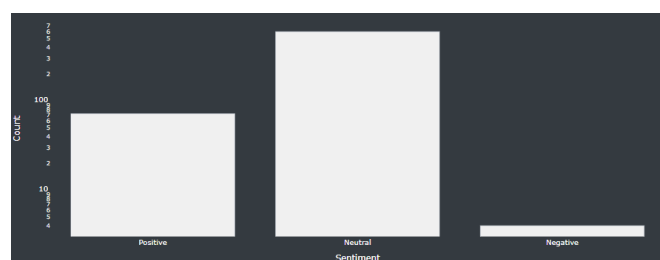


Figure 11: Sentiment Counts

6 View Page

In questa pagina web vengono visualizzati i risultati delle analisi effettuate su ogni file email caricato dall'utente. Naturalmente viene consentito solo all'utente possessore del file di accedere alla pagina.

L'utente ha la possibilità di visualizzare le informazioni più importanti estratte tramite le query e visualizzate con i widget.

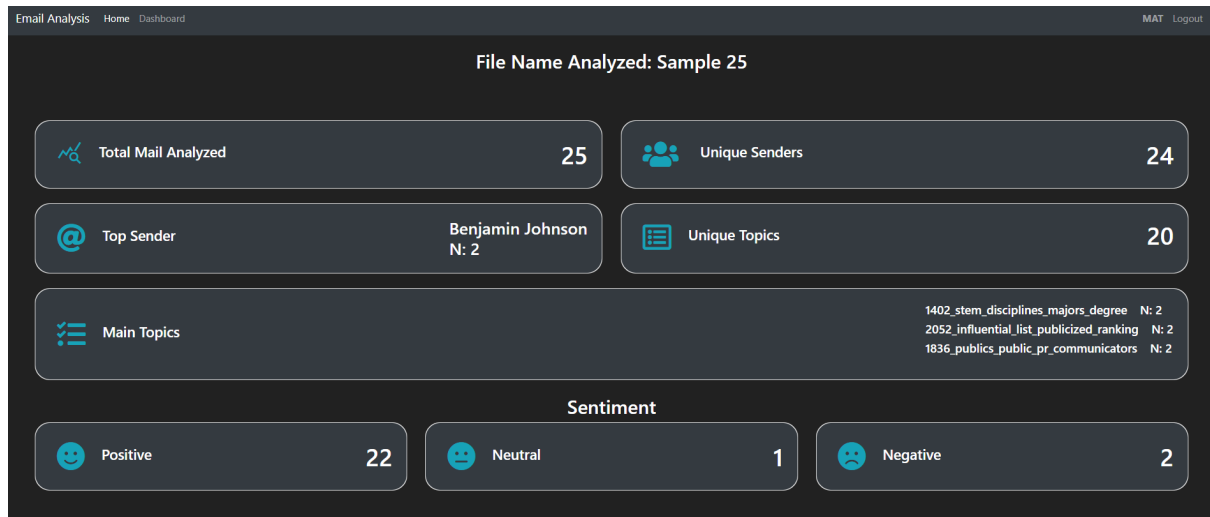


Figure 12: View Page

È presente una tabella per visualizzare tutte le informazioni estratte per ogni singola email presente nel file caricato.

Mail Content		
<p>Our cat, Whiskers, has gone missing. If anyone sees a grey tabby cat in the neighborhood, please email ella.parker@example.com or call (555) 123-9876. Thank you!</p> <p>Extend</p>		
Sender	Subject	
Ella Parker	Lost Cat	
Summary		
You can you find a cat in your neighbourhood in New York.		
NER		
ella.parker@example.com (EMAIL), 123-9876 (PHONE), 123 (CARDINAL), 555 (CARDINAL), Whiskers (ORG)		
Top 10 Words	Topic	Sentiment
cat Whiskers gone missing sees grey tabby neighborhood email ellaparkerexamplecom	332_tabby_cat_feline_cats	Neutral

Figure 13: View Page Result

Mail Content		
<p>I have two extra tickets for the Coldplay concert next Saturday at Madison Square Garden. If anyone is interested, please email me at rachel.green@example.com or call me at (555) 987-6543. First come, first served!</p> <p>Extend</p>		
Sender	Subject	
Rachel Green	Concert Tickets	
Summary		
I have a chance to get two extra tickets for a concert in New York.		
NER		
Madison Square Garden (FAC), rachel.green@example.com (EMAIL), 555 (CARDINAL), two (CARDINAL), first (ORDINAL), 987-6543 (CARDINAL), 987-6543 (PHONE), First (ORDINAL), Coldplay (NORP), next Saturday (DATE)		
Top 10 Words	Topic	Sentiment
extra tickets Coldplay concert Saturday Madison Square Garden interested email	120_theatre_venue_venues_theater	Positive

Figure 14: View Page Result

Mail Content		
<p>Hi, I'm sorry to inform you that I won't be able to keep our dinner reservation for tomorrow night. I've just been called in to work and won't be able to make it. I hope we can try again another time. Regards, Sofia.</p> <p>Extend</p>		
Sender	Subject	
Sofia Hernandez	Re: Dinner Reservation	
Summary		
A couple have had to cancel their dinner reservation for a night out on a date.		
NER		
tomorrow (DATE), Sofia (GPE)		
Top 10 Words	Topic	Sentiment
wo able Hi sorry inform dinner reservation tomorrow night called	1253_cookbook_cookbooks_recipes_chef	Neutral

Figure 15: View Page Result

6.1 Widget

Per ogni file email caricato dall'utente, vengono eseguite delle query per visualizzare:

- **Total Mail Analyzed:** visualizza il numero complessivo di email processate all'interno del file caricato dall'utente

```
total_emails_analyzed = MailAnalysisResult.query.filter_by(file_id=id).count()
```

Total Mail Analyzed

- **Unique Senders:** visualizza il numero complessivo di mittenti unici

```
unique_senders_count = db.session.query(db.func.count(db.func.distinct(
    MailAnalysisResult.mail_from))).filter_by(file_id=id).scalar()
```

Unique Senders

- **Top Sender:** visualizza il nome/email e il numero di count del mittente che ha inviato il maggior numero di email

```
top_sender = db.session.query(MailAnalysisResult.mail_from, db.func.count(
    MailAnalysisResult.id)).filter_by(file_id=id).group_by(MailAnalysisResult.
    mail_from).all()
```

Top Senders

- **Unique Topics:** visualizza il numero di topics unici

```
unique_topics_count = db.session.query(db.func.count(db.func.distinct(
    MailAnalysisResult.mail_topic))).filter_by(file_id=id).scalar()
```

Unique Topics

- **Main Topics:** visualizza le etichette e i count relativi ai 3 main topics estratti

```
main_topics = db.session.query(MailAnalysisResult.mail_topic, db.func.count(
    MailAnalysisResult.mail_topic)) \
    .filter_by(file_id=id) \
    .group_by(MailAnalysisResult.mail_topic) \
    .order_by(db.func.count(MailAnalysisResult.mail_topic).desc()) \
    .limit(3) \
    .all()
```

Main Topics

- **Sentiment:** visualizza il numero di email classificate con l'etichette *Positive*, *Neutral* e *Negative*

```
positive_sentiment_count = MailAnalysisResult.query.filter_by(file_id=id,
    mail_Sent='positive').count()

neutral_sentiment_count = MailAnalysisResult.query.filter_by(file_id=id,
    mail_Sent='neutral').count()

negative_sentiment_count = MailAnalysisResult.query.filter_by(file_id=id,
    mail_Sent='negative').count()
```

Sentiment

- **Topics Frequency:** visualizza il grafico a barre per rappresentare la frequenza dei top 10 topic delle email contenute nel file

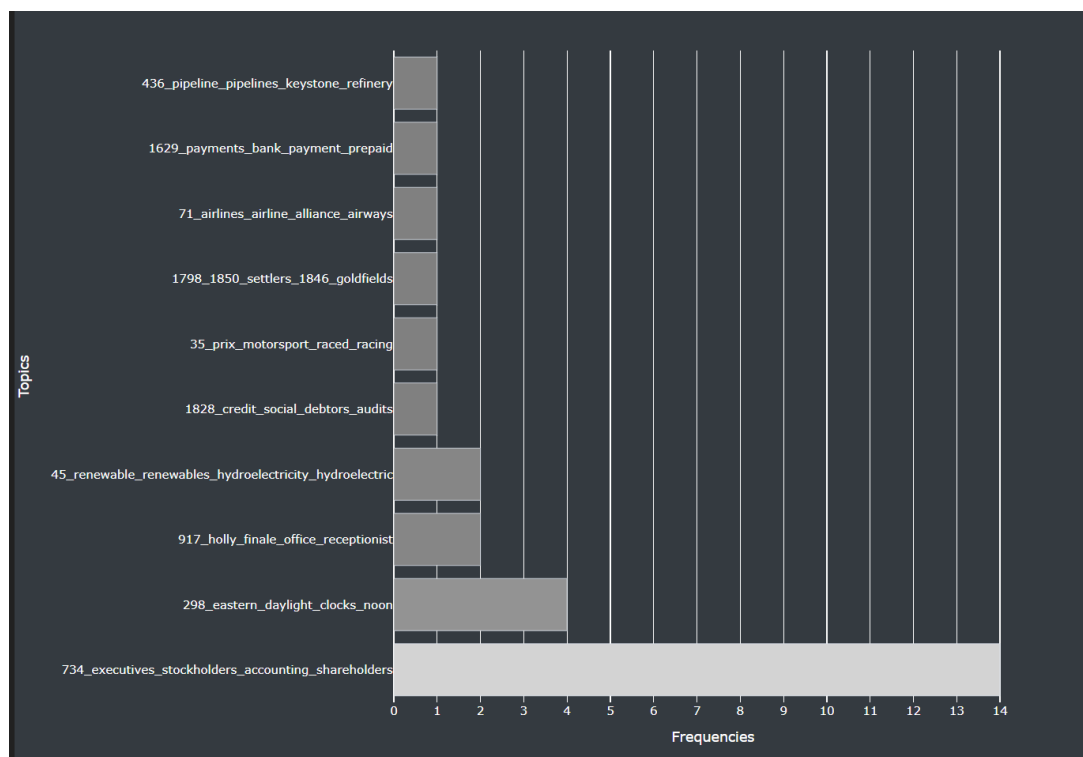


Figure 16: Topics Frequency

7 Email Filters

Come da obbiettivo è stata create la funzionalità per poter filtrare le email processate dall'utente. Sono stati creati due tipi di gestione di filtri:

- Email Filters per tutte le email processate dall'utente eseguibile all'interno della Dashboar
- Email Filters per ogni file email caricato eseguibile all'interno della View Page

L'utente ha possibilità di fare ricerche filtrate per:

- Topic
- Sentiment
- Mittente

È quindi possibile visualizzare solo le email processate, ricercate secondo un criterio di filtraggio scelto dall'utente tramite form.

The image shows a dark-themed filter form. It has three dropdown menus: 'Filter by Topic:' with 'All Topics' selected, 'Filter by Sentiment:' with 'All Sentiments' selected, and 'Filter by Sender:' with 'All Senders' selected. To the right of these is a yellow button labeled 'Filter'.

Figure 17: Filters Form

Il risultato della ricerca sarà visualizzato come:

The image shows the results of a search. At the top, the same filter form as in Figure 17 is shown, but with '120_theatre_venue_venues_the' selected for Topic and 'Positive' for Sentiment. Below the filters, it says '8 mail(s) found.' and 'Mails Processed'. The main content is a table with several sections:

Mail Content		
I have two extra tickets for the Coldplay concert next Saturday at Madison Square Garden. If anyone is interested, please email me at rachel.green@example.com or call me at (555) 987-6543. First come, first served!		
<small>Extend</small>		
Sender	Subject	
Rachel Green	Concert Tickets	
Summary		
I have a chance to get two extra tickets for a concert in New York.		
NER		
two (CARDINAL), first (ORDINAL), Madison Square Garden (FAC), Coldplay (NORP), 555 (CARDINAL), 987-6543 (PHONE), 987-6543 (CARDINAL), next Saturday (DATE), rachel.green@example.com (EMAIL), First (ORDINAL)		
Top 10 Words	Topic	Sentiment
extra tickets Coldplay concert Saturday Madison Square Garden interested email	120_theatre_venue_venues_theater	Positive

Figure 18: Filters Result

8 Test e Risultati

Nel contesto del progetto realizzato, è fondamentale valutare l'efficacia e l'affidabilità del sistema.

A tal fine è stata dedicata una fase di test approfondita per valutare le prestazioni in base agli obiettivi prefissati.

Sono state verificate che le funzionalità di processing, filtraggio, visualizzazione delle mail fossero implementate correttamente, inoltre è stata valutata la capacità del sistema di elaborare grandi quantità di dati in modo scalabile e senza compromettere le prestazioni complessive.

È stata fatta una valutazione rispetto ai risultati delle email processate, valutando la qualità delle informazioni estratte.

Per facilitare il lavoro di testing, sono stati utilizzati due dataset contenenti email con il formato richiesto.

- **The Enron Email Dataset** con 500,000 email di dipendenti aziendali
- **Email Dataset GPT** generando 500 email con topics e sentiment randomici

Presi in input questi dataset sono stati creati diversi file csv, sono stati processati dal sistema, sono stati valutati i risultati raccolti che si sono rilevati veritieri rispetto al contenuto delle email. È stato riscontrato il successo rispetto alla ricerca filtrata e la visualizzazione dei risultati.

Complessivamente, i test hanno confermato l'affidabilità del sistema e la sua capacità di soddisfare gli obiettivi prefissati

9 Conclusioni

L'applicazione web basata su tecniche di elaborazione del linguaggio naturale (NLP) e modelli Transformer rappresenta un avanzamento significativo nella gestione automatizzata delle email. Utilizzando una combinazione di strumenti potenti come il riconoscimento di entità nominate (NER), l'analisi del sentiment, l'estrazione di argomenti e la sintesi automatica, l'applicazione consente agli utenti di estrarre informazioni preziose e dettagliate dalle loro comunicazioni digitali. Questo non solo riduce il tempo necessario per analizzare manualmente le email, ma migliora anche la qualità dei dati estratti, facilitando decisioni più informate e strategiche.

L'architettura distribuita, supportata da servizi scalabili come Google Cloud SQL e Google Compute Engine, garantisce che l'applicazione possa gestire elevati volumi di dati con alta affidabilità e prestazioni ottimali. La scelta di utilizzare Flask per l'implementazione e le librerie open source per l'analisi del testo dimostra l'efficacia e la flessibilità delle moderne tecnologie cloud e di NLP.

Inoltre, la piattaforma offre un'interfaccia utente intuitiva, che rende facile per gli utenti caricare email, effettuare ricerche e visualizzare i risultati dell'analisi tramite widget e grafici interattivi. Questo aumenta la produttività, permettendo agli utenti di concentrarsi su attività più strategiche. La sicurezza e la compatibilità con i principali browser e dispositivi assicurano un'esperienza utente fluida e affidabile.

In sintesi, questo progetto dimostra come l'integrazione di tecnologie avanzate di NLP e infrastrutture cloud possa trasformare la gestione delle email, rendendola più efficiente e accurata. L'applicazione non solo soddisfa le esigenze attuali degli utenti, ma pone le basi per future innovazioni nel campo dell'automazione della gestione delle informazioni.