Università di Bologna
Corso di Laurea in Ingegneria e Scienze Informatiche

Sviluppo di un Chatbot Aziendale basato su Tecniche di AI per il Supporto al Cliente

Tesi di Laurea Triennale

Relatore:
Prof. Marco Antonio Boschetti

Correlatore:
Dott. Luca Foschi

Candidato:
Sajmir Buzi

Anno Accademico 2023/2024

Indice

1 Introduzione **3**
- 1.1 L'azienda Project . . . 4
- 1.2 Contesto e metodologia . . . 5
- 1.3 Obiettivo del progetto all'interno di Project . . . 5
- 1.4 Rilevanza del progetto e impatti attesi . . . 6

2 Metodologie per l'Implementazione **7**
- 2.1 Utilizzo di Whisper . . . 7
  - 2.1.1 Funzionamento di Whisper . . . 8
  - 2.1.2 Applicazione di Whisper nel progetto . . . 8
  - 2.1.3 Problemi riscontrati . . . 8
  - 2.1.4 Valutazione finale di Whisper . . . 9
- 2.2 Libreria NLTK . . . 9
  - 2.2.1 Funzionalità principali di NLTK . . . 9
- 2.3 Utilizzo di tecnologie NLP tramite Python . . . 10
  - 2.3.1 Addestramento e risultati del modello . . . 11
  - 2.3.2 Difficoltà incontrate . . . 11
- 2.4 Utilizzo di Voiceflow . . . 12
  - 2.4.1 Integrazione e flessibilità di Voiceflow . . . 12
  - 2.4.2 Vantaggi di Voiceflow . . . 12
  - 2.4.3 Valutazione complessiva . . . 12

3 Varie tecnologie utilizzate **13**
- 3.1 OpenAI . . . 13
  - 3.1.1 Cos'è OpenAI . . . 13
  - 3.1.2 Obiettivi e Missione . . . 13
  - 3.1.3 Modelli Principali di OpenAI . . . 14
  - 3.1.4 GPT-4: Il Modello di Ultima Generazione di OpenAI . . . 15
  - 3.1.5 Costi di OpenAI . . . 17
  - 3.1.6 Considerazioni sui costi . . . 18
- 3.2 Voiceflow . . . 19
  - 3.2.1 Cos'è Voiceflow . . . 19
  - 3.2.2 A cosa serve Voiceflow . . . 19
  - 3.2.3 Perchè viene usato Voiceflow . . . 19
  - 3.2.4 Costi di Voiceflow . . . 20
  - 3.2.5 Considerazioni sui Costi e Sulla Sostenibilità . . . 21

4 Soluzione Implementata **22**
4.1 Creazione chiave API su OpenAI 23
4.1.1 Funzione e Scopo di una Chiave API 23
4.1.2 Legame OpenAI e Voiceflow tramite API key 23
4.2 Creazione flusso conversionale su Voiceflow 24
4.2.1 Blocco Welcome 25
4.2.2 Blocco Modello AI 27
4.2.3 Blocco di ritorno "not-found" 28
4.3 Utilizzo delle API per Integrare i Modelli di OpenAI 29
4.4 Frontend con servizio web 30
4.5 Taratura Parametri 30
4.5.1 AI model 32
4.5.2 Temperature 32
4.5.3 Tokens 33
4.5.4 Chunk limit 34

5 Conclusioni **35**
5.1 Risultati ottenuti 36
5.2 Sviluppo Futuro 37

Capitolo 1

Introduzione

Il presente lavoro di tesi è stato sviluppato presso l'azienda Project SRL, con sede a Cervia, nel contesto di una collaborazione tra l'ambiente accademico e quello aziendale, finalizzata all'approfondimento di un tema di rilevante interesse per il settore dell'assistenza digitale. L'obiettivo principale di questa tesi è stato quello di studiare, progettare e implementare una soluzione innovativa per l'automazione dell'assistenza clienti tramite un chatbot aziendale, capace di interagire con gli utenti in modo efficiente e di rispondere a domande frequenti in maniera automatizzata. Durante il periodo di collaborazione con l'azienda, sotto la supervisione del tutor aziendale, il dott. Luca Foschi, è stato possibile comprendere le specifiche esigenze operative di Project SRL, nonché acquisire una visione chiara del contesto tecnologico e organizzativo in cui il progetto di tesi sarebbe stato implementato. Dopo una serie di riunioni esplorative con il dott. Foschi e con il titolare dell'azienda, si è deciso di concentrare il progetto su uno studio approfondito delle tecnologie di chatbot, con l'obiettivo di realizzare una versione beta di un sistema automatizzato di assistenza via testo, pensato per essere accessibile da qualsiasi dispositivo, sia fisso che mobile. Prima di entrare nel merito della progettazione tecnica e delle tecnologie selezionate, è utile presentare una panoramica dell'azienda Project SRL, delineando i suoi settori operativi e le principali aree di intervento.

1.1 L'azienda Project

Project SRL (vedi [1]), fondata nel 1992 e situata nella città di Cervia, si occupa della fornitura di soluzioni software e servizi di consulenza per enti pubblici e aziende private. Nonostante le dimensioni ridotte (con circa 30 dipendenti), l'azienda è riuscita a ritagliarsi uno spazio significativo nel settore delle tecnologie informatiche per la gestione dei servizi scolastici e delle pratiche amministrative.

All'interno di Project SRL sono presenti diverse aree operative che collaborano tra loro per garantire l'erogazione di un servizio di alta qualità. Di seguito vengono descritte le principali unità organizzative:

- **Assistenza Clienti:** questo reparto rappresenta una parte fondamentale dell'organico aziendale e si occupa di supportare i clienti attraverso diverse modalità di assistenza. Le figure professionali di questo settore includono consulenti esperti nella gestione finanziaria dei prodotti offerti dall'azienda, oltre a personale dedicato all'help desk, il quale fornisce supporto ai clienti tramite chat o assistenza telefonica. La disponibilità e la preparazione di questo team sono essenziali per mantenere elevata la soddisfazione del cliente e rispondere tempestivamente a problematiche o domande frequenti.

- **Servizio Web:** questa divisione è composta da sviluppatori e tecnici specializzati nella realizzazione e manutenzione dei software aziendali. Il reparto si occupa di tutte le attività legate allo sviluppo del prodotto, dalle fasi di programmazione e testing alla gestione delle funzionalità e al rilascio di aggiornamenti periodici. Grazie a questa unità, Project SRL è in grado di garantire una costante evoluzione delle proprie soluzioni, rispondendo alle necessità di un mercato in continuo mutamento.

- **Sistemisti:** il reparto sistemistico è responsabile delle infrastrutture tecnologiche dell'azienda. I suoi componenti si occupano delle connessioni di rete, garantendo la comunicazione tra i diversi sistemi informatici e la sicurezza dei dati aziendali. I sistemisti sono inoltre responsabili della gestione e manutenzione dei database aziendali, che possono essere configurati sia in rete che in locale, a seconda delle esigenze operative e delle specifiche normative di sicurezza.

Uno dei prodotti principali sviluppati da Project SRL è **eCivis**(vedi [2]), una piattaforma software completa e integrata per la gestione dei servizi scolastici. eCivis è registrato come servizio SaaS qualificato sul marketplace AgID, posizionandosi come una soluzione innovativa e conforme alle normative italiane per gli enti pubblici.

1.2 Contesto e metodologia

Questo progetto di tesi si inserisce nel contesto più ampio dell'automazione dei servizi di assistenza clienti, un tema di grande attualità e rilevanza sia per il settore pubblico che per quello privato. Negli ultimi anni, infatti, la crescente digitalizzazione dei servizi e l'aumento delle aspettative degli utenti per risposte rapide ed efficienti hanno stimolato l'interesse verso soluzioni tecnologiche avanzate come i chatbot, progettati per rispondere autonomamente a richieste di vario tipo, con notevoli benefici in termini di efficienza e riduzione dei costi operativi.

La metodologia seguita per lo sviluppo di questo progetto di tesi ha previsto diverse fasi: una prima fase di ricerca e analisi delle tecnologie disponibili per la creazione di chatbot, una fase comparativa per selezionare la soluzione tecnologica più adatta alle esigenze di Project SRL e, infine, una fase di implementazione e testing del prototipo. L'approccio adottato ha permesso di affrontare il progetto con una solida base teorica e di realizzare una soluzione personalizzata, rispondente alle specifiche necessità dell'azienda.

Nella fase iniziale, sono state esaminate diverse librerie e piattaforme di sviluppo open-source e linguaggi di programmazione orientati alla creazione di chatbot, con particolare attenzione a Python e alle librerie NLTK (Natural Language Toolkit) per l'elaborazione del linguaggio naturale. Sono stati inoltre studiati altri strumenti e risorse reperibili sul web, tra cui l'applicazione Whisper, al fine di integrare funzionalità di comprensione e risposta testuale. L'analisi ha incluso una valutazione di stabilità, flessibilità e capacità di adattamento ai sistemi di Project SRL.

Successivamente, una volta selezionate le tecnologie più appropriate, è stata avviata la fase di sviluppo, che ha incluso la progettazione dell'architettura del sistema, la definizione dei flussi di conversazione e la creazione di un'interfaccia user-friendly. Questo processo ha consentito di creare un prototipo funzionante, capace di rispondere in modo efficace alle esigenze del servizio clienti dell'azienda.

1.3 Obiettivo del progetto all'interno di Project

Vista la grande quantità di richieste e comunicazioni che giornalmente arrivano al servizio clienti, Project SRL ha avvertito l'esigenza di ottimizzare la gestione delle richieste ricorrenti attraverso una soluzione tecnologica innovativa. È in questo contesto che nasce l'idea di sviluppare un chatbot aziendale, pensato per alleggerire il carico di lavoro del personale di supporto e rispondere immediatamente alle domande frequenti degli utenti. Questo chatbot, integrato direttamente nel sito di eCivis dei comuni affiliati, è concepito per fornire risposte rapide e automatizzate ai quesiti più comuni, permettendo così agli utenti di ottenere informazioni in modo tempestivo e senza dover attendere l'intervento diretto di un operatore.

Il chatbot è stato progettato per gestire le richieste semplici e le domande frequenti, mentre le richieste più complesse e specifiche vengono ancora gestite dal team di esperti aziendali. Grazie a un accesso privilegiato come amministratori nel sistema eCivis, gli operatori possono effettuare le modifiche necessarie e rispondere a richieste di natura più tecnica o delicata.

1.4 Rilevanza del progetto e impatti attesi

Questo progetto, oltre a rispondere a una specifica esigenza operativa di Project SRL, mira a contribuire allo sviluppo delle tecniche di intelligenza artificiale applicate all'assistenza clienti. La realizzazione di un chatbot aziendale efficace rappresenta non solo un valore aggiunto per l'azienda, che potrà migliorare l'efficienza dei propri servizi, ma costituisce anche un esempio applicativo di come le tecnologie di automazione possano essere adattate alle specificità del settore pubblico e privato, aprendo la strada a ulteriori studi e sviluppi in questo ambito.

A livello operativo, l'introduzione del chatbot permetterà a Project SRL di ottimizzare i processi di assistenza clienti, con una conseguente riduzione dei tempi di risposta e un miglioramento dell'esperienza dell'utente finale. Questo approccio ha il potenziale di abbattere significativamente i costi associati al servizio clienti, consentendo all'azienda di dedicare il proprio personale alle richieste più complesse, che richiedono un livello di esperienza e competenza superiore. In ambito accademico, il progetto offre un'opportunità di studio concreta delle applicazioni pratiche dell'intelligenza artificiale e delle sfide tecnologiche legate all'implementazione di chatbot in un contesto reale, fornendo una base per future ricerche e sviluppi nel settore dell'assistenza digitale.

Capitolo 2

Metodologie per l'Implementazione

Come accennato nel Capitolo 1, l'implementazione del chatbot aziendale ha richiesto un lungo e complesso processo di studio e sperimentazione, rappresentando una delle fasi più impegnative del progetto. La scelta delle tecnologie da adottare, così come la definizione dell'architettura più adatta, ha comportato molteplici tentativi e una continua rivalutazione delle soluzioni.
Questo capitolo approfondirà i diversi studi condotti, le tecnologie considerate e le soluzioni finali adottate. È importante sottolineare che, oltre alle tecnologie che hanno contribuito direttamente alla realizzazione del chatbot, sono stati esplorati anche approcci che non hanno portato risultati positivi, ma che hanno comunque offerto importanti lezioni per una migliore comprensione del problema.
Questa fase di esplorazione ha avuto una doppia funzione: da un lato, ha permesso di individuare e superare le sfide tecniche legate alla creazione del chatbot; dall'altro, ha fornito una base teorica solida che ha consentito di comprendere le potenzialità e i limiti delle varie tecnologie di intelligenza artificiale e di elaborazione del linguaggio naturale.

2.1 Utilizzo di Whisper

Uno dei primi approcci esplorati per la creazione del chatbot è stato l'uso dell'applicativo Whisper, suggerito dal tutor aziendale come possibile soluzione iniziale. Whisper è un modello di trascrizione e traduzione del parlato in testo, sviluppato da OpenAI. Si tratta di una tecnologia avanzata che sfrutta l'architettura Transformer per l'elaborazione del linguaggio naturale e per la gestione del parlato, con l'obiettivo di convertire automaticamente l'audio in testo. Whisper è progettato per gestire in modo efficiente vari compiti di NLP (Natural Language Processing) e di elaborazione del parlato, rendendolo uno strumento interessante per lo sviluppo di chatbot.

2.1.1 Funzionamento di Whisper

Whisper (vedi [3]) è basato su un'architettura encoder-decoder che consente di mappare l'input audio direttamente su una sequenza di testo. Il processo inizia con la conversione dell'input audio in un "Log-mel spectrogram" , una rappresentazione visiva dell'audio che può essere elaborata dal modello. Questo spettrogramma viene quindi elaborato attraverso una serie di Encoder Blocks, che estraggono le caratteristiche salienti del segnale acustico, preservando al contempo la sequenzialità temporale attraverso un meccanismo chiamato Sinusoidal Positional Encoding. Successivamente, un meccanismo di Cross Attention permette all'encoder di trasferire le informazioni al decoder, che genera la sequenza di testo corrispondente all'audio trascritto.

Il decoder di Whisper utilizza un approccio di previsione del "next-token" (token successivo), il che significa che ogni parola della trascrizione viene generata in modo sequenziale, basandosi su ciò che è stato già elaborato. Questo processo garantisce una certa fluidità e coerenza nella trascrizione, rendendo il modello particolarmente efficace in contesti di trascrizione continua del parlato.

2.1.2 Applicazione di Whisper nel progetto

Inizialmente, l'idea era quella di sfruttare Whisper per trascrivere automaticamente le telefonate dei clienti. Le conversazioni, registrate in formato audio, sarebbero state convertite in testo tramite Whisper e successivamente analizzate per estrarre informazioni rilevanti e categorizzare le richieste.
Tale approccio sembrava promettente, soprattutto considerando la capacità del modello di gestire input audio di qualità variabile e il potenziale di adattamento a diversi contesti linguistici. In teoria, Whisper avrebbe potuto permettere l'elaborazione automatica di chiamate di assistenza clienti, consentendo di generare un report scritto di ciascuna interazione.

2.1.3 Problemi riscontrati

Nonostante le aspettative iniziali, durante l'implementazione sono emersi diversi problemi. In primo luogo, i risultati ottenuti con la lingua italiana sono stati nettamente inferiori rispetto a quelli in lingua inglese. Whisper, essendo ottimizzato per l'inglese, ha mostrato una serie di errori significativi nella trascrizione dell'italiano, specialmente nella gestione di accenti, dialetti o inflessioni linguistiche locali.
Un altro problema fondamentale riguardava l'incapacità di Whisper di distinguere tra i vari interlocutori all'interno di una conversazione telefonica. Questo rappresenta un aspetto cruciale, poiché nel contesto aziendale è necessario sapere se l'interlocutore sia un cliente o un operatore, al fine di garantire risposte coerenti e accurate. La mancanza di tale separazione vocale ha reso difficile l'automazione delle risposte e il controllo della qualità dell'interazione.
Infine, l'elaborazione delle chiamate si è rivelata estremamente dispendiosa dal punto di vista computazionale. La trascrizione di telefonate lunghe richiede un notevole consumo di risorse, rallentando l'intero processo e rendendo il sistema meno efficiente di quanto previsto. Questo, unitamente ai problemi di accuratezza con la lingua

italiana e alla mancata distinzione tra gli interlocutori, ha portato a riconsiderare l'uso di Whisper come soluzione primaria.

2.1.4 Valutazione finale di Whisper

Alla luce delle difficoltà riscontrate, Whisper è stato infine scartato come soluzione principale per il progetto. Sebbene abbia dimostrato potenzialità in contesti specifici, le limitazioni incontrate nell'implementazione pratica ne hanno precluso l'utilizzo. Tuttavia, si riconosce che Whisper potrebbe rappresentare una valida opzione in scenari futuri, soprattutto se combinato con altre tecnologie o con aggiornamenti che migliorino le sue prestazioni linguistiche e computazionali.

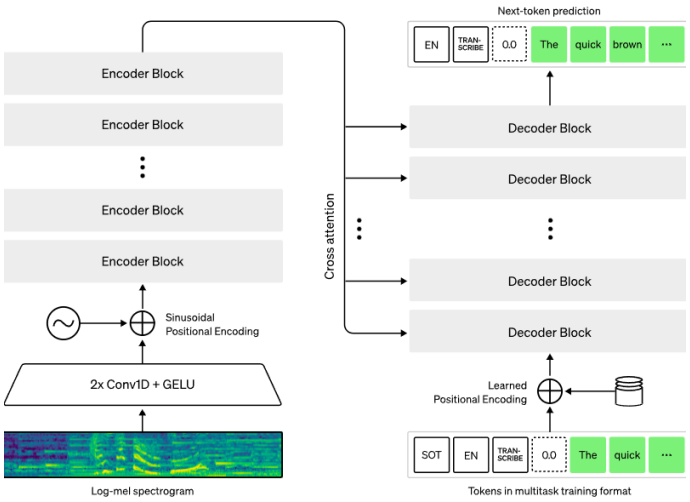

Figura 2.1: Rappresentazione visiva del funzionamento del modello Whisper

2.2 Libreria NLTK

Dopo l'esplorazione di Whisper, un altro approccio esplorato è stato l'uso della libreria NLTK (Natural Language Toolkit), una delle librerie più complete per l'elaborazione del linguaggio naturale in Python. NLTK è largamente utilizzata sia in ambito accademico sia industriale per compiti di text mining, linguistica computazionale e intelligenza artificiale. Nel contesto di questo progetto, si è rivelata uno strumento essenziale durante la fase di prototipazione del chatbot.

2.2.1 Funzionalità principali di NLTK

NLTK (vedi [4]) ha fornito un ampio set di strumenti per l'analisi e la manipolazione del linguaggio, consentendo di processare il testo in maniera strutturata ed efficiente. Le funzionalità principali utilizzate nel progetto includono:

- **Tokenizzazione**: La tokenizzazione è il processo che suddivide il testo in unità più piccole, come parole o frasi. Questo è stato essenziale per costruire

modelli di machine learning più accurati, basati su singole parole o insiemi di parole.

- **Rimozione delle Stop Words**: Le stop words sono parole molto comuni (come "il", "e", "di") che non aggiungono un particolare significato alle frasi e che, in molti casi, possono essere eliminate. Questo processo ha reso l'analisi più precisa, concentrandosi sulle parole più rilevanti.

- **Stemming e Lemmatizzazione**: Questi due processi riducono le parole alle loro forme base. Lo stemming rimuove i suffissi per ottenere una radice, mentre la lemmatizzazione porta le parole alla loro forma base corretta, tenendo conto del contesto grammaticale. La lemmatizzazione si è rivelata particolarmente utile per ottenere risultati accurati.

- **Part-of-Speech (POS) Tagging**: Questa funzione permette di etichettare ogni parola con la sua categoria grammaticale (sostantivo, verbo, aggettivo, ecc.), migliorando la comprensione della struttura delle frasi e delle relazioni sintattiche.

- **Named Entity Recognition (NER)**: La NER identifica e classifica entità nominate, come persone, organizzazioni, luoghi e date. Questo ha permesso di estrarre informazioni specifiche e dettagliate dai testi, migliorando la qualità delle risposte fornite dal chatbot.

- **Parsing e Analisi Sintattica**: La libreria offre strumenti per analizzare la struttura sintattica di una frase, facilitando la comprensione delle relazioni grammaticali e semantiche tra le parole.

2.3 Utilizzo di tecnologie NLP tramite Python

Dopo aver sperimentato le tecnologie offerte da NLTK, si è deciso di ampliare l'approccio utilizzando altre tecnologie NLP basate su Python, con l'obiettivo di migliorare il matching tra le domande dei clienti e le risposte del servizio clienti. Per raggiungere questo obiettivo, si è costruito un dataset di domande e risposte frequenti, che è stato successivamente utilizzato per addestrare un modello di machine learning. Questo modello doveva essere in grado di associare automaticamente le domande degli utenti alle risposte più appropriate, migliorando l'accuratezza del chatbot.
Per rappresentare le domande in un formato sono state utilizzate tecniche di text embedding come il TF-IDF (Term Frequency-Inverse Document Frequency), che converte il testo in vettori numerici. TF-IDF ha permesso di valutare l'importanza delle parole nel contesto del dataset, dando più peso alle parole che appaiono meno frequentemente e riducendo l'importanza delle parole più comuni, come "il", "e", "di". Questo ha aiutato il modello a identificare meglio le parole chiave rilevanti per ogni domanda.

2.3.1 Addestramento e risultati del modello

Il modello è stato addestrato utilizzando Support Vector Machine (SVM), una tecnica di machine learning supervisionato che classifica i dati cercando di separare le classi di domande in uno spazio vettoriale. SVM ha mostrato buoni risultati nella classificazione delle domande frequenti, ma ha avuto difficoltà con le domande che presentavano variazioni rispetto a quelle presenti nel dataset di addestramento.

Il sistema ha ottenuto un'accuratezza del 75%, un miglioramento rispetto all'utilizzo esclusivo di NLTK, ma mostrava ancora difficoltà nel gestire domande formulate in modo diverso da quelle presenti nel dataset. Per risolvere questo problema, sarebbe stato necessario ampliare ulteriormente il dataset con varianti delle domande e adottare modelli più avanzati come BERT o GPT, che possono comprendere meglio il contesto e le intenzioni dell'utente. Questi modelli avrebbero migliorato la capacità del chatbot di rispondere a domande più complesse e a conversazioni multi-turno, dove il contesto delle interazioni precedenti è fondamentale.

Nonostante i limiti, l'utilizzo delle tecnologie NLP tramite Python ha rappresentato un miglioramento significativo rispetto a NLTK, in particolare per quanto riguarda l'accuratezza delle risposte e la flessibilità del modello. Tuttavia, per ottenere risultati migliori e gestire un numero maggiore di variazioni nelle domande, sarebbe stato necessario investire in un dataset più ampio e in modelli di NLP più avanzati.

2.3.2 Difficoltà incontrate

Le principali difficoltà incontrate durante lo sviluppo del chatbot utilizzando tecnologie NLP tramite Python sono state legate alla quantità limitata di dati e alla capacità del modello di generalizzare su nuove domande. La scarsità di dati ha limitato l'efficacia del sistema nel rispondere a domande formulate in modo diverso da quelle presenti nel dataset di addestramento, riducendo la capacità del modello di fornire risposte accurate.

Un'altra sfida è stata la gestione delle variazioni linguistiche. Sebbene l'uso di tecniche come TF-IDF abbia aiutato a migliorare l'accuratezza del modello, il sistema faticava a riconoscere domande semanticamente simili se formulate con parole diverse. Questo problema ha evidenziato la necessità di un dataset più ampio e diversificato, che avrebbe permesso al modello di apprendere meglio le varianti linguistiche e di rispondere in modo più coerente.

Infine, il modello non era in grado di mantenere il contesto delle domande precedenti durante conversazioni più lunghe o multi-turno. Questa mancanza di capacità di gestione del contesto ha limitato l'efficacia del chatbot in scenari in cui l'utente poneva domande che richiedevano una memoria delle interazioni precedenti. La soluzione a questo problema sarebbe stata l'adozione di modelli più avanzati, come BERT o GPT, che sono progettati per gestire meglio il contesto delle conversazioni e migliorare la coerenza delle risposte in conversazioni complesse.

2.4 Utilizzo di Voiceflow

Alla fine, la soluzione che si è rivelata più adatta alle esigenze del progetto è stata Voiceflow, una piattaforma che consente la creazione di chatbot e assistenti vocali tramite un'interfaccia visuale e flussi di lavoro personalizzabili. Voiceflow ha permesso di superare le limitazioni delle tecnologie precedentemente esplorate, offrendo una piattaforma semplice ma potente, in grado di integrare modelli di intelligenza artificiale avanzati e flussi di conversazione dinamici.

2.4.1 Integrazione e flessibilità di Voiceflow

Voiceflow ha offerto la possibilità di integrare modelli di intelligenza artificiale avanzati, come ChatGPT, Claude, Copilot e Gemini di Google. La flessibilità della piattaforma ha consentito di creare flussi di conversazione personalizzati, risolvendo i problemi di accuratezza riscontrati con gli altri strumenti.

La piattaforma permette inoltre una facile gestione del chatbot, consentendo iterazioni rapide sulle conversazioni e aggiornamenti in tempo reale. Questo si è rivelato particolarmente utile nel contesto aziendale, dove le esigenze cambiano frequentemente e il chatbot deve essere costantemente aggiornato per rispondere alle nuove richieste degli utenti.

2.4.2 Vantaggi di Voiceflow

Voiceflow ha reso il processo di sviluppo del chatbot più rapido ed efficiente rispetto alle altre soluzioni esplorate. La possibilità di testare e iterare sui modelli di conversazione in modo intuitivo ha facilitato l'implementazione, riducendo al minimo i tempi di sviluppo. Inoltre, Voiceflow offre la possibilità di monitorare le prestazioni del chatbot, consentendo di apportare modifiche tempestive per migliorare la qualità delle risposte e l'esperienza dell'utente.

2.4.3 Valutazione complessiva

La scelta di Voiceflow si è rivelata vincente in termini di efficienza, flessibilità e scalabilità. Il sistema è facilmente adattabile alle esigenze future dell'azienda e può essere espanso per includere funzionalità avanzate come l'integrazione con nuovi modelli di intelligenza artificiale o la gestione di conversazioni più complesse.

Nei capitoli successivi verranno esplorati più nel dettaglio i vantaggi e i costi di gestione di Voiceflow, confrontando questa soluzione con alternative personalizzate e valutando le potenziali evoluzioni del progetto.

Capitolo 3

Varie tecnologie utilizzate

Come anticipato nel capitolo 2 in questo capitolo andremo ad analizzare meglio le varie tecnologie utilizzate all'interno di questo progetto, i vari costi e le varie soluzioni adottate per configurare ogni tecnologia al fine di rendere il tutto pronto all'utilizzo per sviluppare l'applicazione.
In particolare andremo a soffermarci sui i servizi come OpenAI(di conseguenza anche i vari modelli che offre la piattaforma come GPT, claude, Gemini) e Voiceflow.

3.1 OpenAI

Come detto nel preambolo iniziale di questo capitolo, molto importante è stato l'utilizzo della tecnologia offerta da OpenAI. Andremo adesso ad approfondire a fondo l'utilizzo di tale servizio.

3.1.1 Cos'è OpenAI

OpenAI (vedi [5]) è un'organizzazione di ricerca e sviluppo nell'intelligenza artificiale (IA) fondata nel 2015 da vari ricercatori e imprenditori di fama mondiale. OpenAI ha l'obiettivo di sviluppare e promuovere un'IA amichevole (Friendly AI) che possa avvantaggiare tutta l'umanità e lavora per assicurarsi che i progressi nell'IA siano sicuri, trasparenti e accessibili. Nel corso degli anni, OpenAI è diventata nota per i suoi modelli avanzati di linguaggio, come GPT (Generative Pre-trained Transformer) e la sua capacità di innovare nel campo dell'apprendimento automatico.

3.1.2 Obiettivi e Missione

L'obiettivo primario di OpenAI è creare e promuovere un'intelligenza artificiale che sia allineata con i valori umani e che possa essere utilizzata a beneficio di tutta l'umanità. Ciò implica un'attenzione particolare alla sicurezza, all'etica e alla trasparenza dell'IA. OpenAI mira a garantire che le applicazioni di IA avanzata, come GPT-4 e altri modelli di deep learning, siano:

- **Sicure**: Minimizzando i rischi associati alla proliferazione di IA potenti.
- **Accessibili**: Consentendo a una vasta gamma di utenti e organizzazioni di accedere e beneficiare delle tecnologie avanzate.
- **Etiche**: Allineando l'IA ai valori umani e garantendo che le decisioni prese dai modelli siano trasparenti e spiegabili.

3.1.3 Modelli Principali di OpenAI

OpenAI ha sviluppato diversi modelli di linguaggio avanzati, noti per la loro capacità di generare testo, completare frasi, rispondere a domande, e persino creare codice. Tra i modelli principali vi sono:

- **GPT-2 e GPT-3**: Questi modelli sono stati tra i primi di OpenAI a ricevere un'attenzione globale. GPT-3, in particolare, è diventato famoso per la sua capacità di generare testo coerente e naturale, utilizzando oltre 175 miliardi di parametri. È stato il primo modello a essere rilasciato con API commerciali, aprendo la strada a numerose applicazioni.
- **GPT-4**: La generazione successiva rispetto a GPT-3, con una maggiore capacità di generare testo contestualizzato e accurato.
  Utilizzato in applicazioni avanzate, come la creazione di chatbot sofisticati, la generazione automatica di codice e la produzione di contenuti, GPT-4 è conosciuto per la sua accuratezza e la sua capacità di gestire contesti più complessi.
  Nei successivi paragrafi andremo ad affoffondire meglio il modello citato andando a considerare anche i vari costi rispetto ai modelli procedenti come GPT-3.5.
- **Codex**: Un modello di IA specializzato nella generazione di codice, creato per aiutare sviluppatori e programmatori. Codex può comprendere e generare codice in vari linguaggi di programmazione ed è alla base di strumenti come GitHub Copilot.
- **DALL-E**: Un modello di generazione di immagini che crea immagini a partire da descrizioni testuali. DALL-E può combinare concetti e stili artistici diversi, creando immagini uniche e utilizzate in numerosi campi creativi.

Ovviamente il numero dei modelli sono in graduale aumento al passare del tempo dunque in questa sezione sono solamente citati a mio parere i principali modelli che secondo i calcoli visionati sono i più usati a livello globale.

3.1.4 GPT-4: Il Modello di Ultima Generazione di OpenAI

GPT-4 è l'ultima evoluzione della serie di modelli di linguaggio generativi di OpenAI. Rilasciato nel 2023, GPT-4 rappresenta un significativo passo avanti rispetto ai suoi predecessori in termini di capacità di comprensione del linguaggio naturale, accuratezza e versatilità.

Caratteristiche Principali di GPT-4

In questo paragrafo andremo ad illustrare le principali caratterestiche del modello sopracitato.

- **Dimensioni e Capacità Avanzate**: GPT-4 è significativamente più grande di GPT-3 in termini di numero di parametri e complessità della rete neurale, benché OpenAI non abbia divulgato i dettagli specifici. Questa configurazione aumenta la capacità del modello di comprendere contesti complessi e di generare risposte più dettagliate e accurate.
- **Comprensione Contestuale Migliorata**: Una delle caratteristiche più rilevanti di GPT-4 è la sua capacità di mantenere un contesto più ampio e coerente nel corso delle interazioni. Questo aspetto è essenziale per le applicazioni che richiedono continuità, come le sessioni di assistenza clienti o la generazione di contenuti su argomenti articolati e sequenziali.
- **Risposte Affidabili e Accurate**: OpenAI ha introdotto tecniche avanzate per migliorare la qualità delle risposte generate da GPT-4. Rispetto ai modelli precedenti, GPT-4 dimostra una maggiore affidabilità e precisione, riducendo al minimo la probabilità di generare risposte errate o inappropriate. OpenAI ha inoltre perfezionato i dati di addestramento, implementando metodi di filtraggio volti a ridurre i bias e a garantire risposte più neutrali e imparziali.
- **Multimodalità**: Una delle innovazioni di GPT-4 è la capacità di gestire input sia testuali che visivi, ampliando il suo utilizzo ad applicazioni che richiedono la comprensione di immagini oltre che di testo. Questa caratteristica consente a GPT-4 di analizzare immagini, rispondere a domande basate su contenuti visivi e persino di descrivere o interpretare scene complesse.
- **Fine-Tuning e Personalizzazione:** GPT-4 può essere ulteriormente perfezionato mediante il fine-tuning, ovvero un addestramento specifico su dataset personalizzati per adattare il modello a contesti particolari. Tale funzione consente alle organizzazioni di ottimizzare il modello per applicazioni specifiche, come assistenza clienti, consulenza medica e creazione di contenuti, aumentando così l'efficacia delle risposte.

Applicazioni di GPT-4

GPT-4 è utilizzato in una vasta gamma di applicazioni:

- **Assistenza Clienti**: Le aziende impiegano GPT-4 per creare chatbot e assistenti virtuali che rispondono a domande frequenti, risolvono problemi dei clienti e forniscono supporto in tempo reale.
- **Medicina e Scienza**: GPT-4 può analizzare articoli scientifici, riassumere ricerche e persino supportare i medici fornendo informazioni di riferimento basate su un'enorme quantità di dati.
- **Educazione**: Il modello è utilizzato per creare tutor virtuali che assistono gli studenti nelle loro domande e offrono spiegazioni su vari argomenti.
- **Sviluppo Software**: Grazie a strumenti come Codex, che si basa su GPT-4, gli sviluppatori possono ricevere suggerimenti e completamenti di codice, migliorando l'efficienza nel processo di programmazione.

Confronto tra GPT-3.5 e GPT-4

GPT-3.5 e GPT-4 sono entrambi modelli di linguaggio avanzati sviluppati da OpenAI, ma presentano differenze sostanziali sia in termini di capacità che di applicazioni pratiche. GPT-4 rappresenta un'evoluzione significativa rispetto al suo predecessore, portando con sé miglioramenti rilevanti nelle aree della comprensione contestuale, precisione delle risposte e capacità di interazione con input complessi.
GPT-4 è stato progettato con una rete neurale più complessa e con un numero maggiore di parametri rispetto a GPT-3.5, anche se OpenAI non ha reso noti i dettagli precisi sulla sua architettura. Questo aumento di parametri consente a GPT-4 di processare e comprendere contesti più lunghi e dettagliati, migliorando la sua capacità di rispondere a richieste articolate e di mantenere il filo logico durante le conversazioni. Al contrario, GPT-3.5, pur essendo un modello altamente efficace, presenta delle limitazioni in contesti che richiedono il mantenimento di lunghe sequenze di input.
Un'altra differenza fondamentale tra i due modelli riguarda la multimodalità. GPT-4 supporta input non solo testuali ma anche visivi, consentendo al modello di analizzare immagini e rispondere a domande che richiedono una comprensione visiva. Questa caratteristica amplia notevolmente il potenziale di applicazione di GPT-4, rendendolo utile in contesti come l'analisi di immagini mediche, la generazione di descrizioni di immagini, o la comprensione di diagrammi. GPT-3.5, d'altro canto, è limitato esclusivamente all'input testuale, il che ne restringe l'utilizzo a casi d'uso basati solo su testo.

3.1.5 Costi di OpenAI

OpenAI adotta un modello di pricing basato sul consumo (vedi [6]), in cui gli utenti pagano per il numero di token elaborati dai modelli. Un token è un'unità che corrisponde a circa quattro caratteri di testo, inclusi spazi, lettere e simboli. Questo sistema consente una gestione flessibile dei costi, poiché il prezzo finale è determinato dall'effettivo uso del modello, adattandosi a diversi scenari d'uso e alle esigenze di aziende e organizzazioni. Andremo adesso a vedere i costi per i modelli più usati ovvero GPT 3.5 e GPT 4.

Prezzi per GPT-3.5 e GPT-4

Gli utenti possono scegliere tra diversi modelli di linguaggio, ciascuno dei quali ha tariffe differenti in base alla complessità e alle capacità. I costi vengono generalmente calcolati in base a blocchi di 1.000 token. Tutti i prezzi sono espressi in dollari statunitensi.

GPT-3.5

- Tariffa per 1.000 token di input/output: $0.002 - $0.012

GPT-3.5 rappresenta un'opzione economica e accessibile, particolarmente adatta per chatbot di base, prototipi e applicazioni che non richiedono la sofisticazione dei modelli più avanzati. Questo modello è ampiamente utilizzato da startup e piccole imprese che desiderano integrare l'IA a un costo contenuto.

GPT-4 8k (fino a 8.000 token di contesto per richiesta)

- Tariffa per 1.000 token di input: $0.03 - $0.06
- Tariffa per 1.000 token di output: $0.06 - $0.12

GPT-4 32k (fino a 32.000 token di contesto per richiesta)

- Tariffa per 1.000 token di input: $0.06 - $0.12
- Tariffa per 1.000 token di output: $0.12 - $0.24

GPT-4 ha costi più elevati rispetto a GPT-3.5, ma tali costi sono giustificati dalla maggiore potenza computazionale e dalla capacità di comprendere contesti più lunghi e complessi. Di conseguenza, GPT-4 è utilizzato in applicazioni avanzate, come consulenza legale, supporto medico e interazioni clienti prolungate.

Nota sui costi variabili: La variazione delle tariffe per ciascun modello dipende da diversi fattori operativi e tecnici:

- **Tipo di utilizzo e priorità della richiesta**: Tariffe differenti possono essere applicate in base alla priorità e alla complessità dell'elaborazione richiesta.

- **Disponibilità delle risorse computazionali**: In periodi di alta domanda, il costo per token può aumentare a causa della maggiore richiesta di risorse.
- **Contratti personalizzati e volume di token**: OpenAI offre tariffe variabili in base ad accordi specifici e in funzione del volume totale di token richiesto.

Le tariffe per il modello GPT-4 con contesto fino a 8.000 token sono inferiori rispetto a quelle per il modello con contesto di 32.000 token. Questo costo maggiore è dovuto alla capacità del modello da 32k di gestire contesti più lunghi, richiedendo una maggiore potenza computazionale. Pertanto, il modello GPT-4 con 32k è particolarmente indicato per applicazioni che necessitano di contesti conversazionali più dettagliati e analitici.

3.1.6 Considerazioni sui costi

La scelta del modello e del piano di prezzi è determinante per ottimizzare il budget e garantire un'efficace implementazione delle soluzioni di intelligenza artificiale. Mentre GPT-4 rappresenta una soluzione potente per applicazioni complesse che richiedono elevata affidabilità e comprensione contestuale, GPT-3.5 fornisce un'opzione più economica e accessibile per la maggior parte delle applicazioni di base. La struttura dei costi flessibile di OpenAI consente alle organizzazioni di adattare i servizi di IA alle proprie esigenze, bilanciando budget ed efficienza operativa.
In conclusione, OpenAI offre una gamma di soluzioni IA scalabili e accessibili, permettendo agli utenti di scegliere il livello di complessità e capacità del modello in base alle proprie esigenze specifiche. Questo approccio rende le tecnologie di IA di OpenAI accessibili a un vasto spettro di utenti, da sviluppatori individuali a grandi aziende che necessitano di soluzioni avanzate di intelligenza artificiale.

3.2 Voiceflow

Un'altra tecnologia utilizzata per il progetto riguarda Voiceflow. In questa sezione andremo a vedere in modo più dettagliato a cosa serve Voiceflow, perchè viene utilizzato ed i vari costi per il suo corretto funzionamento da parte dei vari utenti.

3.2.1 Cos'è Voiceflow

Voiceflow (vedi [7]) è una piattaforma progettata per creare chatbot interattivi e assistenti vocali, senza la necessità di una programmazione complessa. Lanciata inizialmente per supportare la creazione di skill per Amazon Alexa e Google Assistant, Voiceflow è ora ampiamente utilizzata per sviluppare applicazioni di conversazione più generiche, inclusi chatbot per siti web, app mobili e assistenti virtuali basati su IA, come quelli che si integrano con i modelli di OpenAI.

3.2.2 A cosa serve Voiceflow

Voiceflow è progettato per semplificare la creazione di esperienze conversazionali e per consentire la creazione di chatbot e assistenti vocali che possono essere distribuiti su più canali. I principali usi di Voiceflow includono:

- **Creazione di Chatbot e Assistenti Vocali Personalizzati**: La piattaforma consente agli utenti di costruire flussi conversazionali complessi attraverso una serie di blocchi modulari, ciascuno dei quali rappresenta una fase della conversazione. Questi blocchi includono messaggi di testo, prompt vocali, risposte dinamiche basate su variabili, logiche condizionali e altro ancora. Voiceflow si integra facilmente con i modelli di linguaggio di OpenAI, come GPT-3.5 e GPT-4, consentendo di aggiungere intelligenza avanzata ai chatbot e di migliorare la qualità delle risposte alle richieste degli utenti.
- **Prototipazione Rapida di Esperienze Conversazionali**: Voiceflow offre strumenti di prototipazione rapida, che consentono agli utenti di creare, modificare e testare flussi conversazionali in tempo reale. Questa capacità è particolarmente utile per i team di progettazione UX e di sviluppo prodotto, che possono iterare velocemente sulle idee e ottimizzare il design del flusso in base ai feedback degli utenti.
- **Supporto Multicanale**: Voiceflow consente di creare un unico flusso conversazionale che può essere distribuito su piattaforme diverse, come Alexa, Google Assistant, siti web, app mobili e social media. Questo permette di garantire un'esperienza utente coerente su più punti di contatto, semplificando allo stesso tempo la gestione e la manutenzione del chatbot.

3.2.3 Perchè viene usato Voiceflow

Voiceflow è particolarmente popolare per la facilità d'uso, la flessibilità e la capacità di creare esperienze personalizzate senza dover scrivere codice complesso. Le ragioni principali per cui Voiceflow è scelto da aziende di diverse dimensioni includono:

- **Accessibilità per Professionisti Non Tecnici**: Grazie alla sua interfaccia visiva, Voiceflow consente a designer, marketer, e professionisti di business di creare chatbot e assistenti vocali in autonomia, senza la necessità di competenze di programmazione avanzata.
- **Flessibilità nella Personalizzazione e nell'Integrazione**: Voiceflow consente di personalizzare ogni aspetto dell'esperienza conversazionale, dall'aspetto visivo alla logica dietro le risposte. Gli utenti possono configurare variabili, condizionali, e persino incorporare chiamate API per integrare Voiceflow con altre applicazioni e servizi, come CRM, piattaforme di e-commerce, o servizi di messaggistica.
- **Riduzione dei Tempi di Sviluppo**: Voiceflow semplifica il processo di sviluppo delle interfacce conversazionali, riducendo significativamente i tempi necessari per costruire, testare e distribuire i chatbot. Questa caratteristica è particolarmente vantaggiosa per aziende che vogliono ridurre i costi di sviluppo e accelerare il time-to-market (tempo necessario a inserire nel mercato un prodotto finito).
- **Collaborazione e Gestione dei Progetti**: Voiceflow è ideale per team che necessitano di collaborare e di condividere idee e modifiche in tempo reale. La piattaforma supporta funzionalità di condivisione e gestione dei progetti, consentendo ai membri del team di lavorare contemporaneamente sugli stessi flussi conversazionali proprio all'interno dell'applicativo.

3.2.4 Costi di Voiceflow

Voiceflow offre diversi piani di abbonamento, che vanno dal gratuito a piani per grandi aziende, con costi che variano a seconda delle funzionalità e del livello di supporto richiesti. La struttura di prezzo è pensata per essere scalabile e adattabile alle diverse esigenze dei clienti, garantendo l'accesso a funzionalità di base per i piccoli team e fornendo opzioni avanzate per le organizzazioni più grandi.

- **Piano Gratuito**: Il piano gratuito offre accesso alle funzionalità di base della piattaforma, consentendo di creare e testare flussi conversazionali semplici. Include un numero limitato di integrazioni e di utenti, ed è adatto per singoli utenti e startup che desiderano sperimentare con Voiceflow.
- **Piano Pro**: Questo piano offre funzionalità avanzate, come integrazioni API più estese, accesso a configurazioni predefinite per facilitare lo sviluppo di progetti e una maggiore capacità di gestione degli utenti e delle sessioni di test. Con un costo che si aggira intorno ai \$50-\$60 al mese, rappresenta una soluzione ideale per piccole e medie imprese che richiedono strumenti più completi e flessibili per lo sviluppo e l'ottimizzazione del chatbot.
- **Piano Team**: Il piano Team offre funzionalità di collaborazione tra più utenti, con opzioni di condivisione e gestione del progetto, una maggiore capacità di integrazione e supporto prioritario. È ideale per aziende di dimensioni medio-grandi, e il costo mensile si aggira sui \$100-\$150 al mese.

- **Piano Enterprise**: Questo piano offre soluzioni personalizzate per aziende con elevate necessità di supporto e funzionalità avanzate, come formazione personalizzata e supporto tecnico premium. I costi sono personalizzati in base alle esigenze dell'azienda e possono superare i $500 al mese.

3.2.5 Considerazioni sui Costi e Sulla Sostenibilità

La struttura dei costi di Voiceflow rende la piattaforma accessibile e scalabile, adattandosi sia alle esigenze di singoli utenti e piccole imprese, sia a quelle di grandi organizzazioni. Per le aziende che desiderano implementare intelligenza artificiale avanzata nei loro chatbot, Voiceflow supporta l'integrazione con OpenAI e altri fornitori di IA. È importante notare che i costi associati all'utilizzo di queste integrazioni, come le chiamate API a OpenAI, sono separati dai costi di abbonamento a Voiceflow e vengono determinati dai fornitori esterni, variando in base al volume di utilizzo.
In conclusione, Voiceflow offre un'ampia gamma di strumenti per creare esperienze conversazionali potenti e flessibili. La possibilità di integrare intelligenza artificiale e di distribuire i flussi su più canali rende Voiceflow una soluzione versatile per chiunque voglia costruire chatbot e assistenti vocali, mantenendo sotto controllo i costi e adattando le funzionalità alle proprie necessità aziendali.

Capitolo 4

Soluzione Implementata

In questo capitolo verrà illustrata in dettaglio la soluzione implementata per la realizzazione del progetto. Dopo un'analisi approfondita delle varie opzioni tecnologiche disponibili, si è scelto di adottare Voiceflow come strumento principale per la progettazione e lo sviluppo dell'assistente virtuale, integrato con i modelli avanzati di linguaggio naturale forniti da OpenAI. Questa combinazione è stata selezionata poiché offre una soluzione flessibile e scalabile, che consente di creare un'esperienza conversazionale intuitiva e personalizzabile, adatta sia per interazioni vocali che testuali.
Nella sezione successiva, verranno spiegati nel dettaglio i passi intrapresi per implementare la soluzione, fornendo una visione d'insieme delle metodologie utilizzate e delle motivazioni alla base di ciascuna scelta implementativa. Ogni aspetto del progetto, dalle fasi di progettazione iniziale fino alla configurazione finale, sarà approfondito per evidenziare l'efficacia della piattaforma Voiceflow nell'integrazione con i modelli OpenAI.
Verranno inoltre discussi i vantaggi e le limitazioni della soluzione proposta, confrontando le varie metodologie implementative considerate. Per ogni fase del processo, sarà presentata una spiegazione dettagliata delle tecnologie impiegate, nonché delle scelte progettuali che hanno contribuito a ottimizzare la performance e la qualità dell'interazione conversazionale. In particolare, si andrà a motivare l'utilizzo di una metodologia rispetto ad un'altra, analizzando le opzioni disponibili e valutando le soluzioni in base a criteri di efficienza, scalabilità e facilità di integrazione.
In conclusione, questo capitolo fornirà una visione completa della soluzione implementata, evidenziando come Voiceflow e OpenAI lavorino insieme per creare un chatbot sofisticato e rispondente ai requisiti specifici del progetto.

4.1 Creazione chiave API su OpenAI

Al principio del lavoro è stata necessaria la creazione di una chiave API all'interno del sito di openAI. Dopo essere entrati tramite le proprie credenziali bisogna spostarsi nella sezione API keys e premere il tasto "Create new secret keys", attraverso la pressione di questo tasto all'interno di openAI verrà generata una chiava privata utilizzabile solo dall'effittivo creatore di questa chiave. Formalmente una chiave API (API Key) è un codice unico, solitamente una lunga stringa di caratteri alfanumerici, che funge da identificatore e autorizzazione per l'accesso a un'API (Application Programming Interface). Questa chiave viene fornita da un fornitore di servizi API, come OpenAI, per consentire agli utenti di accedere alle loro funzionalità e servizi in modo sicuro.

4.1.1 Funzione e Scopo di una Chiave API

1. **Autenticazione**: La chiave API conferma l'identità di chi sta effettuando la richiesta. Ogni volta che un'applicazione invia una richiesta a un servizio API, la chiave viene inviata come parte della richiesta per dimostrare che è autorizzata.

2. **Sicurezza e Autorizzazione**: Poiché le API spesso permettono l'accesso a dati sensibili o funzionalità potenti (come ad esempio il modello GPT-4 di OpenAI), l'utilizzo di una chiave API aiuta a garantire che solo utenti autorizzati possano accedere a tali risorse.

3. **Monitoraggio e Limiti di Utilizzo**: Le chiavi API permettono ai fornitori di servizi di monitorare e limitare l'utilizzo dell'API per ciascun utente. Questo è importante sia per evitare abusi che per gestire i costi. Ad esempio, OpenAI può utilizzare la chiave API per tenere traccia delle richieste e applicare limiti di utilizzo, garantendo che ogni utente stia utilizzando il servizio in base al proprio piano.

4.1.2 Legame OpenAI e Voiceflow tramite API key

Nel caso di OpenAI e Voiceflow:

- Quando si configura Voiceflow per usare uno dei modelli di OpenAI, è necessario inserire la propria chiave API, che autorizza Voiceflow a fare chiamate a OpenAI.

- Voiceflow include questa chiave API in ogni richiesta per identificare il proprio account, garantendo che vengano ricevute risposte dal modello usato in modo sicuro e controllato.

In sintesi, la chiave API è come una "password" che permette a un'applicazione di accedere a risorse specifiche su una piattaforma esterna, proteggendo sia i dati che l'infrastruttura del servizio offerto. A seguito sarà possibile vedere un'immagine inerente la creazione di un propria chiave API all'interno di openAI come spiegato in precedenza.

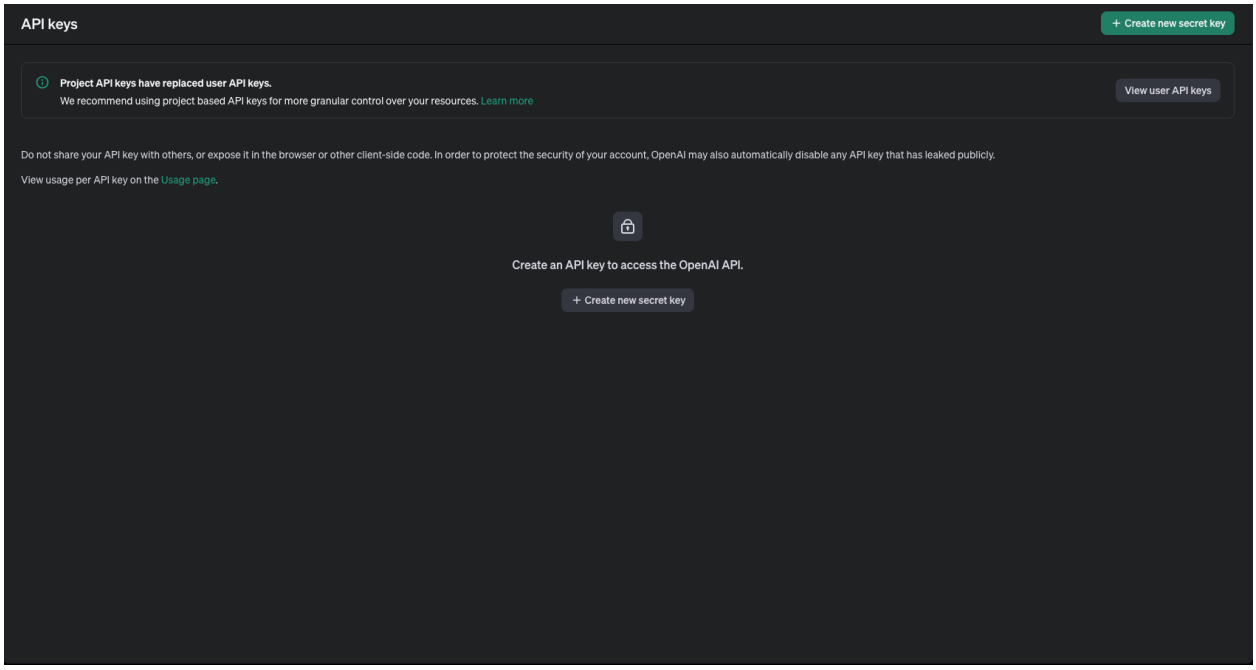

Figura 4.1: Creazione chiave API su openAI

4.2 Creazione flusso conversionale su Voiceflow

Dopo aver generato la chiave API tramite OpenAI, possiamo procedere con l'implementazione del nostro chatbot su Voiceflow, utilizzando uno schema a blocchi per definire il flusso conversazionale. Voiceflow consente di configurare i vari passaggi di una conversazione tramite un'interfaccia visuale basata su blocchi, facilitando la creazione di flussi interattivi senza la necessità di programmare manualmente ogni risposta.
In questo flusso conversazionale, possiamo inserire la chiave API di OpenAI all'interno di un blocco di integrazione. Questo permette a Voiceflow di collegarsi ad un modello tra queli disponibili e inviare i messaggi degli utenti come input. Il modello fornisce risposte generate dinamicamente in base alle specifiche istruzioni fornite nel prompt iniziale, e Voiceflow reindirizza le risposte direttamente agli utenti.
Attraverso la creazione di questo flusso condizionale (vedi figura 4.2), è possibile progettare percorsi di conversazione ramificati che rispondono a domande comuni e interagiscono con i clienti in modo naturale e coerente. Ogni blocco del flusso può essere configurato per rispondere in base a condizioni predefinite o inviare input personalizzati al modello. Questo approccio permette di creare chatbot che offrono un'esperienza su misura, adattandosi alle specifiche esigenze e comportamenti dell'utente.
Voiceflow consente inoltre di integrare blocchi di logica per condurre scelte condizionali, salvare variabili e monitorare il percorso di interazione, rendendo possibile la creazione di conversazioni strutturate. Ciò permette al chatbot non solo di fornire risposte personalizzate, ma anche di adattare il flusso a seconda delle risposte e delle esigenze degli utenti, offrendo un'esperienza interattiva avanzata.

In figura 4.2 verrà mostrato il flusso conversionale implementato formato fondamentalemente da 3 blocchi chiave. Il primo blocco sarà il blocco di benvenuto (welcome) dove ad ogni nuova conversazione il chatbot genera un saluto iniziale all'utente. Il secondo blocco è quello più importante ovvero il blocco di generazone tramite il modello scelto, dove viene generata la risposta in base alla domanda posta tramite una variabile che vedremo in seguito. Il terzo blocco, chiamato blocco "di ritorno", intercetta le domande per le quali il chatbot non ha una risposta predefinita. In questi casi, il chatbot genera una risposta informando l'utente che non è stato possibile trovare una soluzione adeguata. Nelle sezioni successive approfondiremo ulteriormente i diversi blocchi del flusso conversazionale.

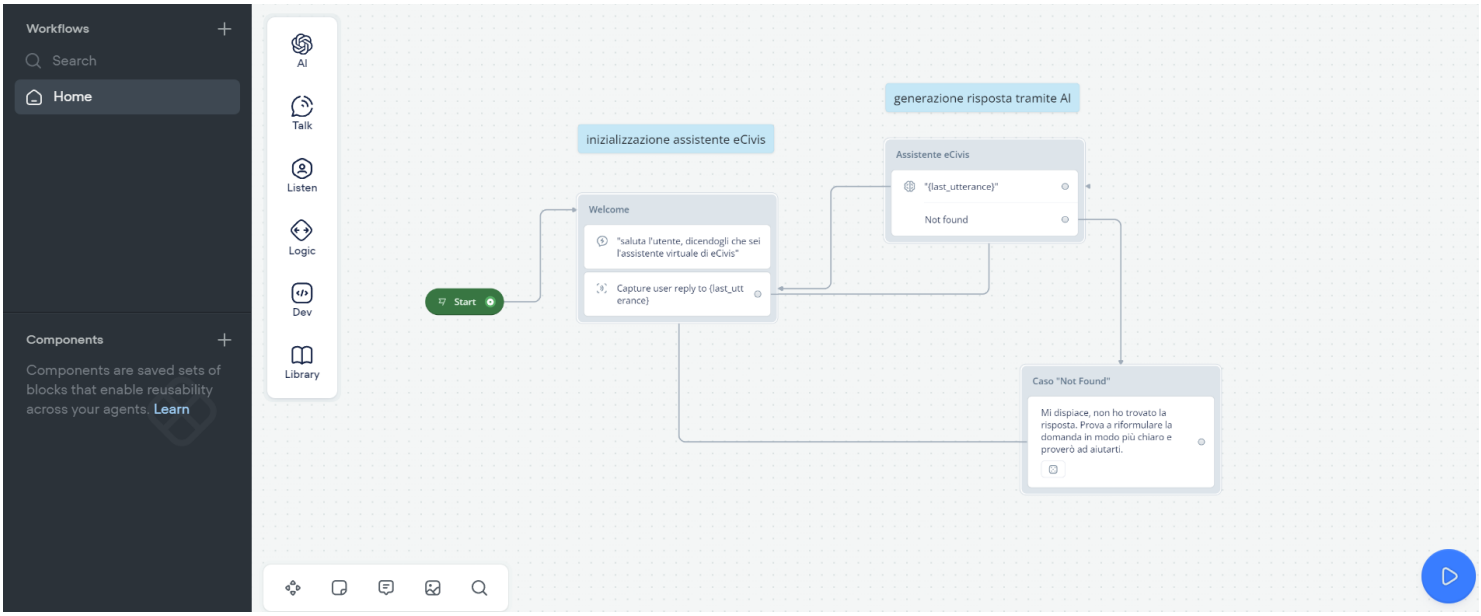

Figura 4.2: Flusso conversazionale su Voiceflow

4.2.1 Blocco Welcome

Come anticipato, andiamo ad analizzare nel dettaglio i vari blocchi del chatbot implementato. Il primo blocco che esamineremo è il blocco di **benvenuto** (Welcome).
Come mostrato in figura, all'interno di questo blocco sono presenti due sottoblocchi fondamentali, rispettivamente chiamati **Response AI** e **Capture**.
Nel sottoblocco **Response AI**, l'utente deve configurare il comportamento del chatbot in termini di risposta. In questa sezione è possibile scegliere se utilizzare il modello di intelligenza artificiale (**AI Model**) o la base di conoscenza (**Knowledge Base**) per gestire la risposta del chatbot. Nel nostro caso, abbiamo optato per l'uso del modello di intelligenza artificiale, impostando una risposta introduttiva in cui il chatbot accoglie l'utente e lo informa di essere l'assistente virtuale di eCivis.

Nel secondo sottoblocco, denominato **Capture**, si definisce una variabile per catturare e memorizzare l'input dell'utente. In questo contesto, utilizziamo la variabile `last_utterance`, che viene impiegata per registrare l'ultima espressione verbale

dell'utente. Questa variabile ci permette di tenere traccia del messaggio più recente inviato dall'utente, il che è essenziale per l'elaborazione delle risposte successive e per mantenere la continuità nella conversazione. Nelle figure 4.3 e 4.4 sono rappresentati i due sottoblocchi appartenenti al blocco Welcome.

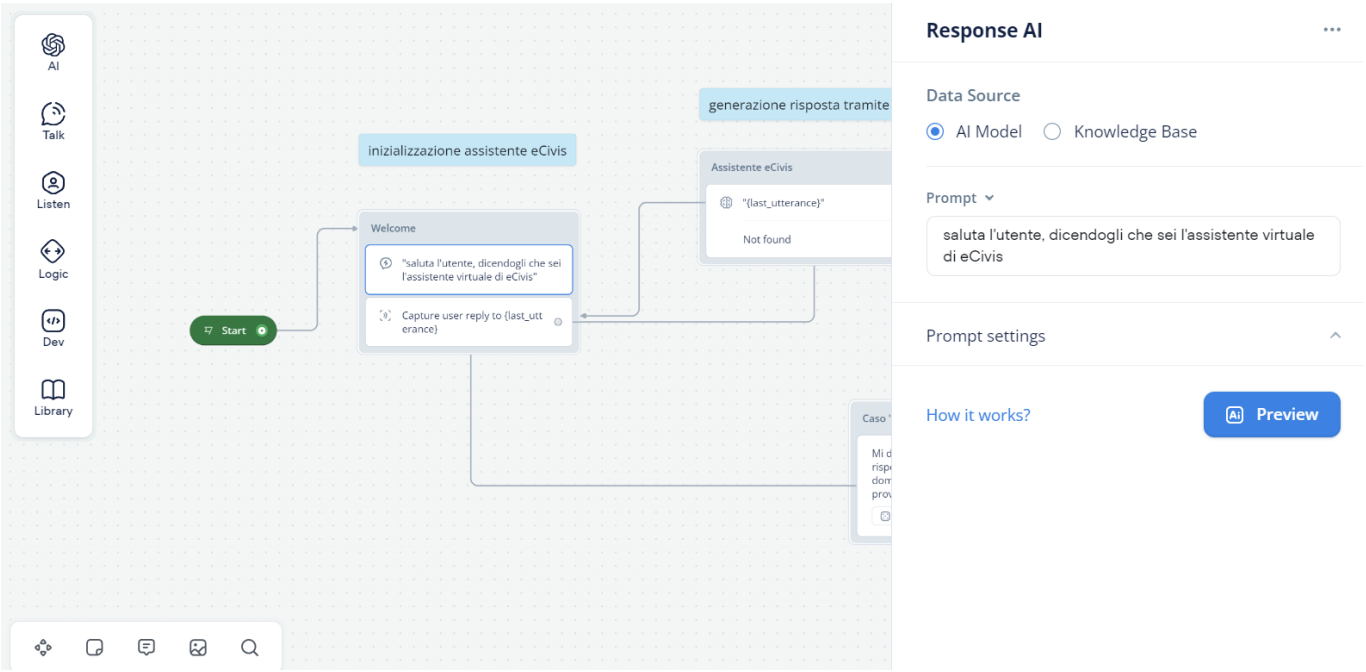

Figura 4.3: sottoblocco Response AI

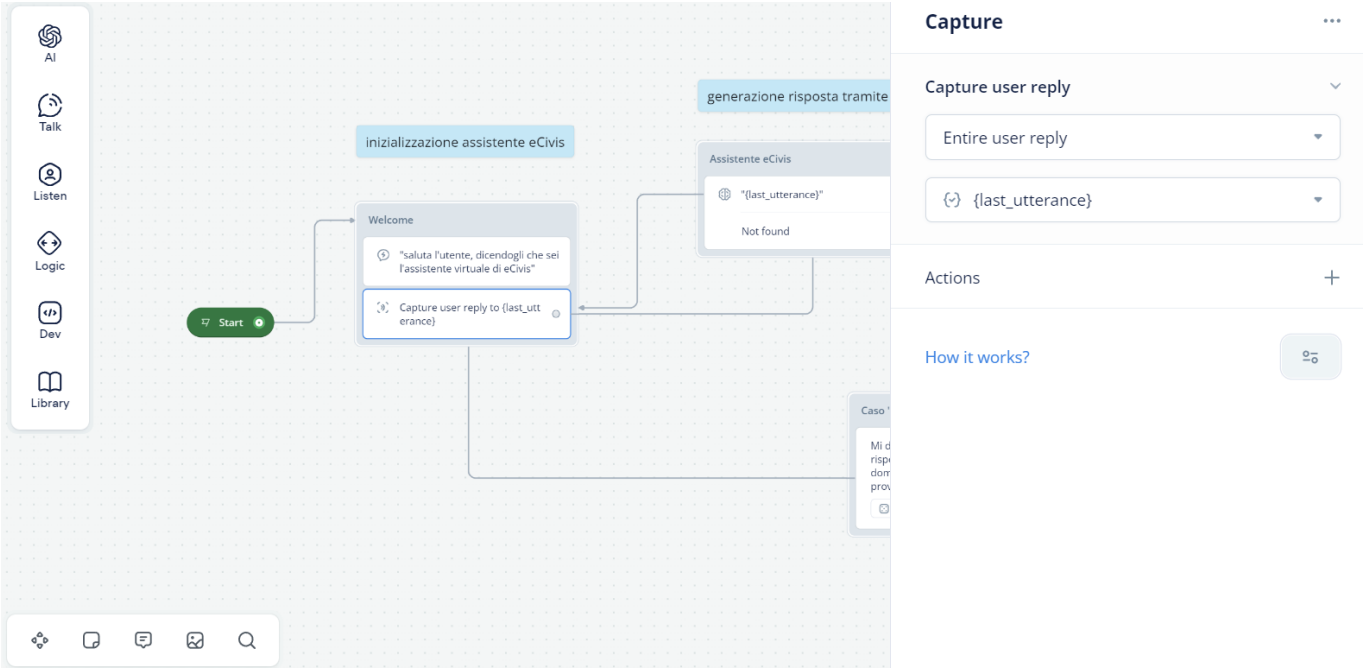

Figura 4.4: sottoblocco Capture

4.2.2 Blocco Modello AI

In questo blocco (vedi figura 4.5), considerato il più importante dell'intero chatbot, avviene la fase cruciale di generazione del linguaggio testuale basata sull'addestramento fornito al modello. È essenziale configurare correttamente diversi parametri affinché il blocco possa operare nel modo desiderato.
Il primo parametro da impostare è il **datasource** (sorgente di dati), che offre una scelta tra la **Knowledge Base** e il **Model AI**. In questo caso, si sceglie di utilizzare la **Knowledge Base**, poiché le informazioni che il chatbot deve elaborare provengono dal file di addestramento implementato dal sottoscritto. La Knowledge Base permette al chatbot di accedere direttamente ai dati specifici forniti durante la fase di training.
Un altro parametro chiave è la **question**, che deve essere configurata per catturare l'input dell'utente. In questa fase, la variabile `last_utterance` viene nuovamente utilizzata, selezionata tra le varie opzioni disponibili all'interno del sistema. Questa variabile è fondamentale in quanto memorizza l'ultima espressione verbale dell'utente, fornendo così al modello il contesto necessario per generare una risposta pertinente.
Infine, è necessario fornire al modello delle istruzioni specifiche, che consistono nel seguire rigorosamente il file di addestramento e rispondere alle domande poste dall'utente in base a tali dati. Qualora non fosse possibile reperire alcuna informazione rilevante nel file di addestramento, il chatbot risponderà con un messaggio di default, informando l'utente che non è stato possibile trovare alcuna corrispondenza nelle informazioni fornite.

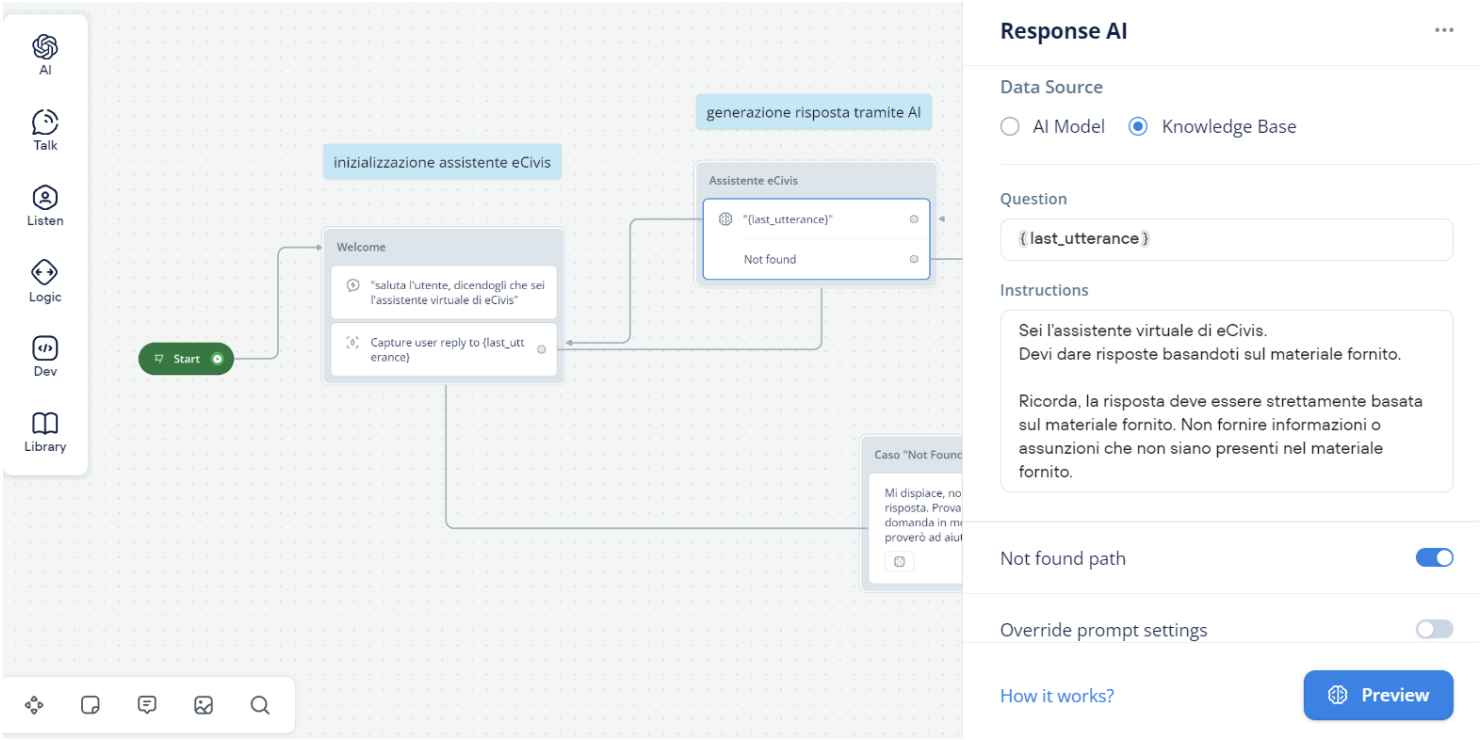

Figura 4.5: blocco model AI

4.2.3 Blocco di ritorno “not-found”

In questo blocco (vedi figura 4.6), che denomineremo **blocco di ritorno**, si gestisce il caso in cui il modello non riesca a catturare alcuna informazione rilevante dal file di addestramento. In tali circostanze, il chatbot non è in grado di fornire una risposta basata sui dati appresi e, di conseguenza, restituisce una risposta di default, segnalando la mancata disponibilità di una risposta appropriata.
La funzione principale di questo blocco è dunque quella di gestire le situazioni di “non risposta” da parte del modello. In pratica, quando il chatbot non riesce a trovare informazioni pertinenti nel file di addestramento o a comprendere la domanda dell’utente, attiva una risposta predefinita per informare l’utente che non sono stati trovati dati utili a generare una risposta valida.
Una volta restituita la risposta di default, il blocco di ritorno è collegato nuovamente al **blocco di benvenuto** (Welcome), dove viene ristabilita una nuova connessione con l’utente corrente. Questo permette al chatbot di riprendere il ciclo di interazione con l’utente, offrendo la possibilità di formulare una nuova domanda o di gestire ulteriori richieste, mantenendo un flusso di conversazione continuo e reattivo.
Grazie a questo collegamento con il blocco di benvenuto, il chatbot è in grado di riavviare il processo di conversazione senza interrompere l’esperienza dell’utente, garantendo una comunicazione più fluida e pronta a nuove interazioni.

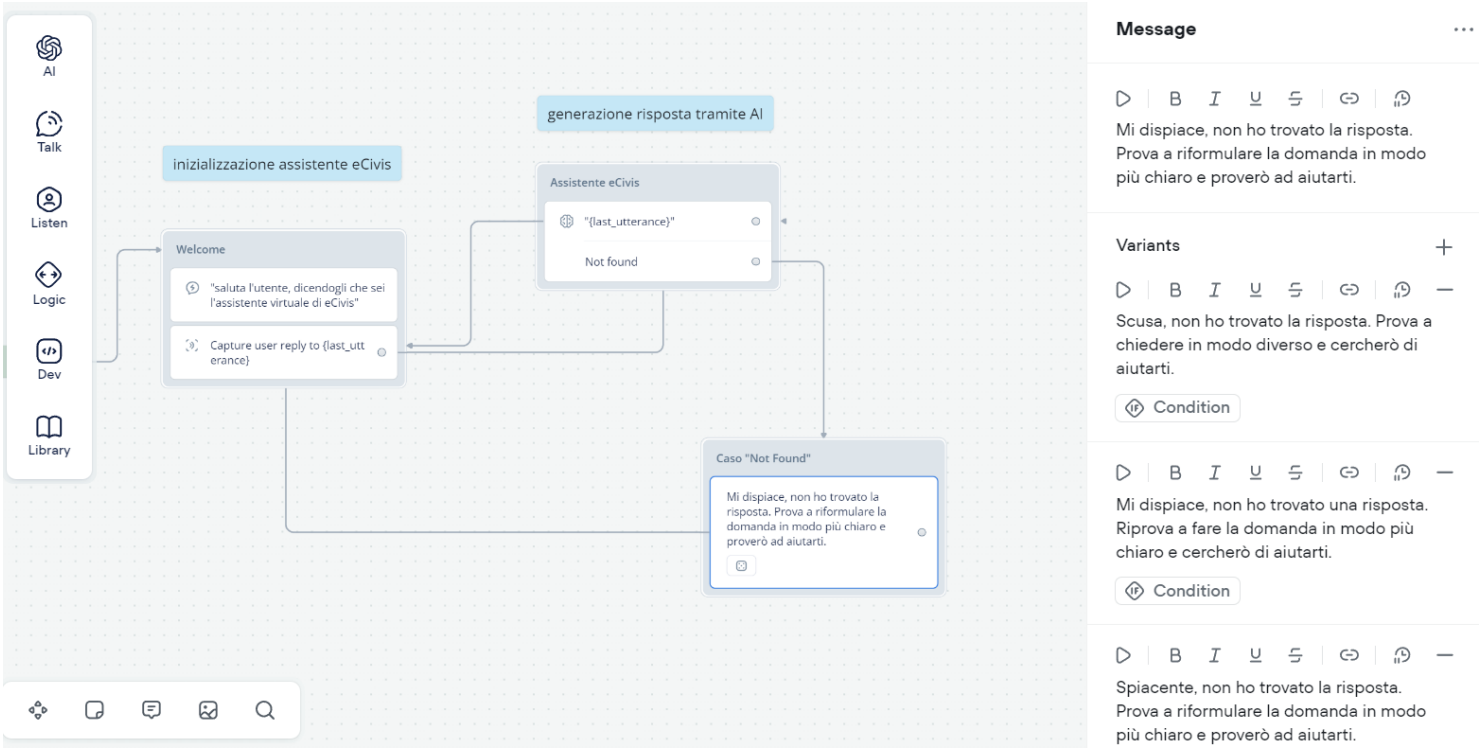

Figura 4.6: blocco not-found

4.3 Utilizzo delle API per Integrare i Modelli di OpenAI

In questa sezione viene illustrata l'integrazione del modello GPT-4 o GPT-3.5 di OpenAI tramite API, utilizzando un esempio di codice Python. Sebbene Voiceflow semplifichi questa operazione attraverso la sua interfaccia grafica, è utile comprendere le dinamiche sottostanti della chiamata API.
Il codice mostrato in figura 4.7 serve a inviare una richiesta POST all'API di OpenAI, specificando il modello desiderato e il messaggio da parte dell'utente. La configurazione include parametri quali l'autenticazione tramite chiave API e la gestione delle risposte JSON per estrarre la risposta generata dal modello. Questo esempio dimostra come sia possibile interagire direttamente con i vari modelli, configurando il comportamento del chatbot attraverso il "prompting", ovvero l'invio di istruzioni nel contesto.
L'integrazione illustrata qui è un esempio diretto di come la tecnologia OpenAI possa essere incorporata in applicazioni tramite API, offrendo la possibilità di espandere chatbot e assistenti digitali con capacità linguistiche avanzate.

```python
import requests
# Definisci l'URL dell'API e la chiave API
api_url = "https://api.openai.com/v1/chat/completions"
api_key = "YOUR_API_KEY"  # Sostituisci con la tua chiave API
# Imposta l'header per l'autenticazione e il tipo di contenuto
headers = {
    "Authorization": f"Bearer {api_key}",
    "Content-Type": "application/json"
}
# Definisci il corpo della richiesta
data = {
    "model": "gpt-4",
    "messages": [
        {
            "role": "system",
            "content": "You are a helpful assistant trained to respond to questions about eCivis."
        },
        {
            "role": "user",
            "content": "What is the role of a chatbot in customer service?"
        }
    ],
    "temperature": 0.7
}
# Invia la richiesta POST
response = requests.post(api_url, headers=headers, json=data)
# Verifica la risposta e stampa il risultato
if response.status_code == 200:
    response_data = response.json()
    chatbot_reply = response_data["choices"][0]["message"]["content"]
    print("Chatbot response:", chatbot_reply)
else:
    print("Request failed with status code:", response.status_code)
    print("Error message:", response.json())
```

Figura 4.7: codice integrazione modello

4.4 Frontend con servizio web

Dopo aver implementato il chatbot e averlo testato approfonditamente, è possibile procedere alla connessione del chatbot con i servizi esterni (vedi figura 4.8) in cui deve essere utilizzato, come nel nostro caso, con i servizi online del sito web di eCivis.
Il processo di integrazione del chatbot con sistemi esterni, specialmente quando si tratta di piattaforme web complesse, può rivelarsi un compito piuttosto impegnativo. Questo richiede spesso competenze tecniche avanzate e la capacità di gestire diversi tipi di servizi e protocolli. Tuttavia, uno dei vantaggi principali offerti da piattaforme come Voiceflow è la possibilità di semplificare questa fase. Voiceflow fornisce agli sviluppatori codice JavaScript pronto per l'uso, facilitando notevolmente l'integrazione del chatbot con altri servizi esterni.
Questa funzionalità non solo rende più agevole l'integrazione con i servizi web, ma consente anche di interagire con API e sistemi backend, aprendo la strada a una vasta gamma di applicazioni. Ad esempio, tramite l'utilizzo del codice JavaScript generato, il chatbot può essere connesso a database esterni, CRM, sistemi di gestione degli ordini, o qualsiasi altro servizio che richieda una comunicazione in tempo reale o l'accesso a dati personalizzati.
Grazie a questa flessibilità, Voiceflow riduce significativamente i tempi di sviluppo necessari per implementare tali integrazioni, rendendo il chatbot utilizzabile su una vasta gamma di piattaforme e dispositivi con minimi sforzi di configurazione. Inoltre, questo approccio rende il chatbot scalabile e facilmente adattabile a nuove esigenze aziendali o tecnologiche, garantendo che esso possa evolvere insieme ai cambiamenti futuri dei sistemi con cui interagisce

```
<script type="text/javascript">
  (function(d, t) {
      var v = d.createElement(t), s = d.getElementsByTagName(t)[0];
      v.onload = function() {
        window.voiceflow.chat.load({
          verify: { projectID: '65f2d3034d8a4fe3f1d01a18' },
          url: 'https://general-runtime.voiceflow.com',
          versionID: 'production'
        });
      }
      v.src = "https://cdn.voiceflow.com/widget/bundle.mjs"; v.type = "text/javascript"; s.parentNode.insertBefore(v, s);
  })(document, 'script');
</script>
```

Figura 4.8: script integrazione chatbot

4.5 Taratura Parametri

In questa sezione analizzeremo la configurazione e l'ottimizzazione dei parametri disponibili nella piattaforma Voiceflow, con l'obiettivo di migliorare l'efficienza e ridurre i costi operativi del chatbot. La taratura dei vari parametri è stata eseguita attraverso una serie di test approfonditi condotti personalmente, al fine di trovare il giusto equilibrio tra prestazioni del modello e sostenibilità economica.
La scelta dei valori ottimali è stata effettuata con l'obiettivo di ottenere un'otti-

mizzazione globale, sia in termini di accuratezza delle risposte del chatbot che in termini di costi legati all'utilizzo delle API di OpenAI. Durante la fase di test, sono stati analizzati diversi scenari operativi per comprendere l'impatto delle variazioni di parametri chiave, come il livello di temperatura del modello, il numero di token utilizzati nelle risposte e il bilanciamento tra qualità e velocità dell'elaborazione.
L'ottimizzazione dei parametri su Voiceflow ha richiesto un attento bilanciamento tra prestazioni e costi; i test condotti hanno consentito di identificare la configurazione più efficiente per il nostro chatbot. Il risultato è un sistema che non solo è in grado di fornire risposte di alta qualità, ma che lo fa in modo sostenibile e ottimizzato dal punto di vista economico. Nella figura 4.9 è possibile vedere i vari parametri da configurare ed ottimizzare, nelle successive sezioni andremo ad approfondire in maniera dettagliata ogni parametro.

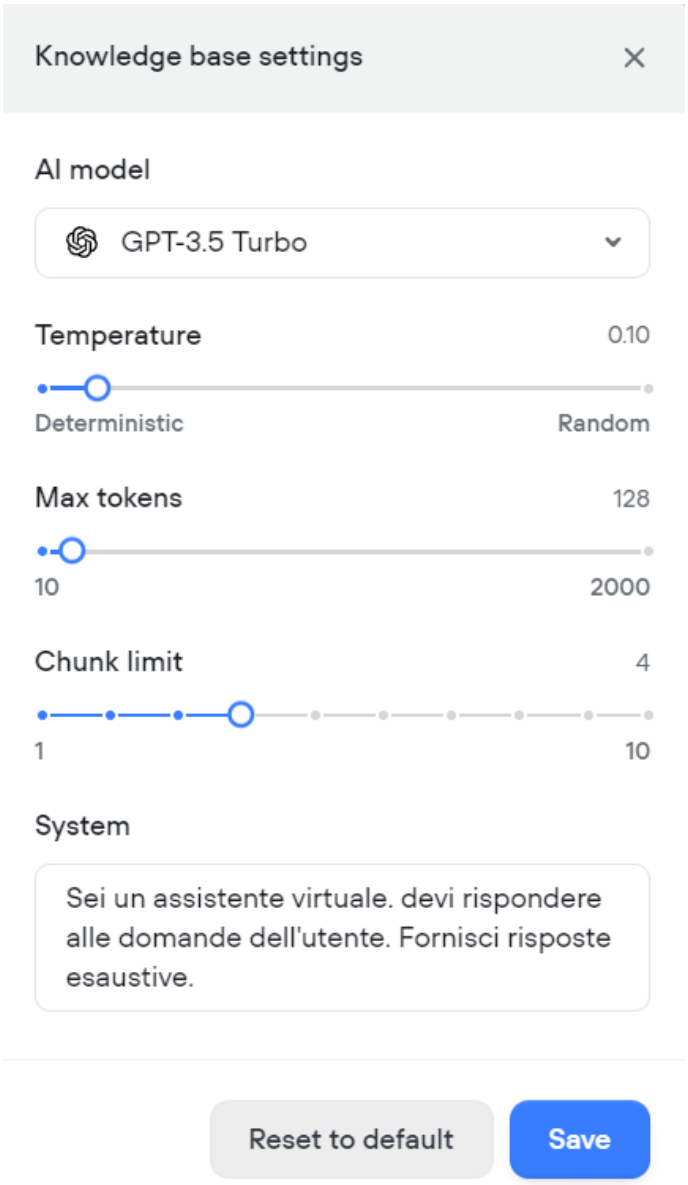

Figura 4.9: script integrazione chatbot

4.5.1 AI model

In questa sezione ci concentreremo sulla modifica del modello utilizzato nel chatbot. Nel caso specifico, è stato deciso di adottare GPT-3.5, uno dei modelli più avanzati e ampiamente utilizzati di OpenAI, per la sua capacità di comprendere e generare testo in modo fluido e coerente. Tuttavia, va sottolineato che esistono diversi modelli alternativi disponibili, come Claude di Anthropic (vedi [8]) e Gemini di Google DeepMind (vedi [9]), ciascuno con le proprie peculiarità e vantaggi a seconda del contesto di utilizzo.
La selezione del modello più adatto dipende dalle esigenze specifiche del progetto, tuttavia è sempre importante considerare le alternative e rimanere flessibili nell'adottare nuovi modelli man mano che le tecnologie AI continuano a evolversi, garantendo che il chatbot possa continuare a migliorare e adattarsi alle mutevoli esigenze aziendali.

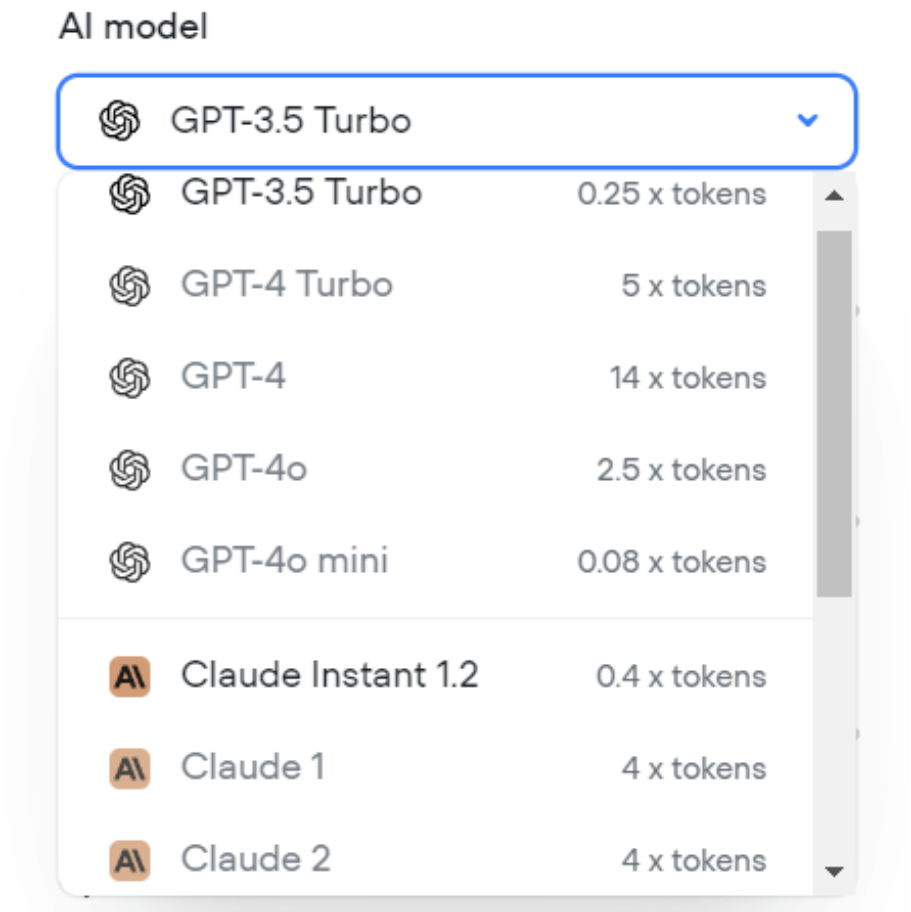

Figura 4.10: scelta dei vari modelli

4.5.2 Temperature

Il parametro "temperature" è uno dei più importanti per influenzare la creatività e la variabilità delle risposte generate dal chatbot. Questo parametro controlla il livello di casualità con cui il modello seleziona le parole successive durante la generazione

del testo.
Quando la temperature è impostata su un valore basso, ad esempio 0.1, il modello produrrà risposte più deterministiche e precise, tendendo a scegliere le parole più probabili in base al contesto. Questo è utile per scenari in cui è richiesta maggiore coerenza e accuratezza, come chatbot per il supporto tecnico o risposte a domande frequenti (FAQ), dove l'affidabilità della risposta è cruciale.
Al contrario, valori di temperature più elevati, come 0.8 o 1.0, aumentano la casualità delle risposte, rendendo il chatbot più creativo. Questo può essere vantaggioso in contesti in cui si desidera una conversazione più naturale e variegata, come chatbot per scopi creativi o per simulazioni di conversazioni complesse.
In pratica, il valore della temperature deve essere tarato a seconda dell'obiettivo del chatbot:

- **Temperature bassa (0.0 - 0.3)**: Risposte precise e coerenti.
- **Temperature media (0.4 - 0.7)**: Bilanciamento tra creatività e accuratezza.
- **Temperature alta (0.8 - 1.0)**: Risposte più creative e meno prevedibili.

L'ottimizzazione della temperature è un processo iterativo e richiede una serie di test per trovare il giusto compromesso tra creatività e coerenza delle risposte in base al contesto d'uso.

4.5.3 Tokens

Il parametro "max tokens" determina la lunghezza massima delle risposte generate dal modello, in termini di token. Un token può rappresentare una parola, parte di una parola o caratteri speciali, a seconda del linguaggio utilizzato. Questo parametro è cruciale perché influenza sia la quantità di informazione fornita dal chatbot che il costo di ogni singola richiesta API, dato che OpenAI addebita i costi in base al numero di token utilizzati.
Impostare un valore appropriato per max tokens è essenziale per evitare che il chatbot generi risposte troppo lunghe o troppo brevi. Un valore troppo basso potrebbe portare a risposte troncate o incomplete, mentre un valore troppo alto potrebbe generare risposte prolisse e non sempre necessarie.

- **Valori bassi di max tokens (ad esempio 50-100)** sono utili per chatbot con risposte concise, come sistemi di domande frequenti (FAQ) o risposte rapide a richieste semplici.
- **Valori medi (tra 150-300)** sono adatti per conversazioni standard, dove è necessario un bilanciamento tra dettaglio e concisione.
- **Valori alti (oltre 500)** possono essere utilizzati per chatbot che devono fornire risposte dettagliate o svolgere compiti complessi come spiegazioni approfondite o analisi di testo.

L'ottimizzazione di questo parametro deve tenere conto non solo dell'esperienza utente, ma anche dei costi associati all'utilizzo del modello, in quanto risposte troppo lunghe potrebbero aumentare sensibilmente i costi operativi del chatbot.

4.5.4 Chunk limit

Il "chunk limit" rappresenta un parametro meno noto, ma altrettanto importante, che limita la dimensione dei blocchi di testo che possono essere processati o restituiti dal modello in una singola chiamata. Questo parametro è essenziale quando il chatbot deve gestire testi o conversazioni lunghe, dove è necessario suddividere il contenuto in "chunk" più piccoli e gestibili per il modello.
Impostare un chunk limit appropriato evita il rischio di sovraccaricare il modello con input troppo grandi, che potrebbero portare a errori o rallentamenti nelle risposte. Un chunk limit ben calibrato permette di suddividere correttamente le informazioni, mantenendo un flusso di conversazione continuo e senza interruzioni.

- **Valori bassi** di chunk limit sono ideali per input brevi o richieste specifiche.
- **Valori medi** possono essere utilizzati in conversazioni normali o per domande che richiedono più dettagli.
- **Valori alti** sono consigliati quando il chatbot deve gestire documenti lunghi o blocchi di testo estesi, suddividendoli in porzioni più piccole per facilitare l'elaborazione.

Un chunk limit troppo elevato potrebbe però portare a sovraccaricare il modello o provocare ritardi nella risposta, quindi anche in questo caso deve essere calibrato attentamente in base alla lunghezza media delle interazioni previste e alla capacità del sistema di supportare input e output più lunghi effettuando vari test.

Capitolo 5

Conclusioni

Questo progetto ha approfondito la progettazione, implementazione e ottimizzazione di un chatbot avanzato basato su intelligenza artificiale, sviluppato tramite la piattaforma **Voiceflow** e integrato con il modello **GPT-3.5** di **OpenAI**.
Il lavoro svolto è stato motivato dall'obiettivo di creare uno strumento di assistenza virtuale capace di rispondere in modo preciso, naturale ed efficiente, riducendo al contempo i costi operativi associati all'uso intensivo delle API.
Un aspetto fondamentale del progetto è stato l'addestramento del chatbot tramite una knowledge base testuale che descriveva accuratamente l'applicazione **eCivis**.
Attraverso questa base di conoscenza, è stata integrata la capacità per il modello di rispondere con precisione alle domande specifiche degli utenti, fornendo risposte mirate e contestualizzate. Come vedremo nella sezione dei risultati ottenuti, la scrittura del testo per l'addestramento del chatbot ha avuto un ruolo fondamentale.
Al fine di evitare ambiguità nelle risposte fornite, è stata eseguita una descrizione dettagliata dell'applicativo eCivis e dei vari servizi richiesti dagli utenti. In questo modo, grazie a una descrizione approfondita e un'attenta taratura dei parametri principali del chatbot, come **temperature**, **max tokens** e **chunk limit**, è stato possibile bilanciare la qualità delle risposte e la sostenibilità economica. Le sezioni seguenti riassumono i risultati raggiunti e delineano le possibili direzioni per sviluppi futuri del progetto.

5.1 Risultati ottenuti

L'implementazione del chatbot e la successiva fase di testing hanno prodotto risultati rilevanti, evidenziando sia le potenzialità della piattaforma **Voiceflow** che le capacità del modello **GPT-3.5**. I principali risultati ottenuti sono descritti di seguito:

- **Efficienza e Pertinenza delle Risposte**: Uno degli obiettivi centrali del progetto era migliorare l'efficienza del chatbot, garantendo risposte tempestive e pertinenti. Grazie alla knowledge base testuale dedicata a **eCivis**, il modello è stato addestrato con informazioni dettagliate sull'applicazione, rendendolo capace di rispondere in modo contestuale e preciso alle domande degli utenti. Questa base di conoscenza ha permesso al chatbot di interpretare e comprendere le richieste specifiche degli utenti relative a eCivis, adattando le risposte in funzione delle domande e fornendo informazioni mirate sui suoi servizi e funzionalità. La scelta di un valore moderato per il parametro temperature ha giocato un ruolo cruciale nel bilanciare creatività e precisione delle risposte: un valore più basso avrebbe prodotto risposte troppo rigide, mentre un valore troppo alto avrebbe generato risposte meno affidabili. La configurazione scelta ha permesso di ottenere risposte fluide e naturali, senza sacrificare la coerenza e la qualità, rendendo l'interazione con il chatbot più piacevole per l'utente.

- **Controllo dei Costi Operativi**: Un altro risultato importante è stato il controllo e la riduzione dei costi operativi grazie all'ottimizzazione del parametro max tokens, che limita il numero di token (unità di testo) utilizzati per ogni risposta. Questa ottimizzazione si è dimostrata essenziale, poiché ogni token genera un costo incrementale nel servizio API, e gestire il numero di token significa poter bilanciare qualità e dettaglio delle risposte con i costi complessivi di utilizzo. Configurando il chatbot per produrre risposte sintetiche ma esaustive, è stato possibile ottenere una risposta efficace con un uso minimo di token, contribuendo così a ridurre le spese associate all'uso continuo e intensivo dell'API. Questa ottimizzazione risulta particolarmente importante per applicazioni aziendali con volumi elevati di interazione.

- **Scalabilità e Adattabilità del Sistema**: Il chatbot si è rivelato altamente scalabile e adattabile a diversi scenari di utilizzo grazie alla modularità offerta da Voiceflow. L'integrazione del parametro chunk limit ha permesso di gestire conversazioni e input lunghi suddividendo il contenuto in blocchi di testo più piccoli e facilmente elaborabili, mantenendo così il chatbot performante anche in contesti che richiedono interazioni più estese. La capacità del chatbot di adattarsi a differenti scenari di interazione, e di rispondere a domande complesse in modo continuativo e senza interruzioni, si è rivelata particolarmente vantaggiosa per applicazioni che richiedono una gestione completa e coerente delle conversazioni. Il sistema, quindi, si presenta come una soluzione versatile, adatta a evolversi insieme alle necessità dell'utente e dell'ambiente di implementazione.

5.2 Sviluppo Futuro

Nonostante i risultati soddisfacenti, ci sono ampie opportunità per migliorare e ampliare le funzionalità del chatbot. Di seguito vengono descritte le principali direzioni di sviluppo che potrebbero essere esplorate in futuro:

- **Integrazione con Altri Modelli di Linguaggio**: Sebbene il progetto attuale utilizzi GPT-3.5, esistono modelli alternativi che potrebbero offrire vantaggi specifici in base alle esigenze. Ad esempio, modelli come Claude di Anthropic e Gemini di Google DeepMind potrebbero migliorare ulteriormente l'esperienza conversazionale in contesti diversi. Claude è noto per l'attenzione alla sicurezza e all'affidabilità, e potrebbe rappresentare una scelta ideale in contesti regolamentati, dove è richiesta maggiore attenzione alla sicurezza dei dati e all'etica delle risposte. Gemini, d'altra parte, offre una stretta integrazione con l'ecosistema Google, rendendolo particolarmente vantaggioso per le aziende che utilizzano già servizi e strumenti Google. Testare questi modelli e integrarli potrebbe rendere il chatbot ancora più versatile, consentendogli di adattarsi a una varietà di contesti aziendali e di mercato.

- **Espansione delle Capacità Multilingue**: Attualmente, il chatbot è ottimizzato per rispondere principalmente in una singola lingua. Tuttavia, un'evoluzione futura potrebbe prevedere l'introduzione di funzionalità multilingue, rendendo il chatbot più accessibile a un pubblico internazionale. Questa funzionalità sarebbe di grande utilità per aziende e organizzazioni che operano in diversi mercati globali, consentendo al chatbot di rispondere automaticamente nella lingua dell'utente. L'implementazione di capacità multilingue comporterebbe sfide tecniche, come la gestione del contesto e la coerenza delle risposte tra lingue differenti, ma permetterebbe di ampliare notevolmente l'utilizzo del chatbot e la sua adattabilità.

- **Integrazione con Dati Esterni e Sistemi Aziendali**: Per migliorare ulteriormente l'efficacia e la rilevanza delle risposte, il chatbot potrebbe essere connesso a database esterni, sistemi CRM (Customer Relationship Manager), o altri sistemi aziendali. Questa integrazione avanzata permetterebbe al chatbot di accedere a dati personalizzati in tempo reale, consentendogli di fornire risposte altamente contestualizzate e rilevanti per ciascun utente. Ad esempio, in un contesto aziendale, il chatbot potrebbe accedere ai dati di acquisto di un cliente per rispondere a domande sui suoi ordini all'interno dell'applicazione eCivis, migliorando così l'esperienza cliente e personalizzando le risposte in base alle esigenze specifiche dell'utente. Questa integrazione rappresenterebbe un ulteriore passo verso un sistema chatbot realmente intelligente e completamente integrato con l'ecosistema aziendale.

- **Ottimizzazione Continua tramite Feedback e Analisi delle Risposte**: Per garantire che il chatbot continui a migliorare nel tempo, potrebbe essere utile integrare un sistema di monitoraggio che raccolga feedback dagli utenti e analizzi le risposte generate. Un approccio del genere permetterebbe di adattare e ottimizzare continuamente i parametri del chatbot in base ai dati

raccolti, migliorando così l'accuratezza delle risposte e aumentando la soddisfazione dell'utente. Implementare una funzionalità di feedback sarebbe utile anche per identificare aree di miglioramento e affinare le risposte in base a contesti d'uso specifici, rendendo il chatbot più preciso e in linea con le esigenze degli utenti.

In conclusione, il progetto ha soddisfatto gli obiettivi iniziali di efficienza, sostenibilità e scalabilità. L'utilizzo della piattaforma **Voiceflow**, abbinato alle capacità del modello **GPT-3.5**, ha consentito di sviluppare un chatbot efficace, flessibile e ottimizzato per un'ampia gamma di applicazioni. Grazie a una gestione attenta dei parametri di configurazione e all'addestramento specifico tramite una knowledge base testuale dedicata a **eCivis**, il chatbot ha raggiunto un equilibrio tra qualità delle risposte e controllo dei costi, dimostrando la validità di un approccio metodico all'ottimizzazione delle risorse in contesti di intelligenza artificiale applicata.
Le possibilità di espansione e perfezionamento del chatbot sono numerose, e la continua evoluzione delle tecnologie di intelligenza artificiale offre ampi margini di miglioramento. Le direzioni di sviluppo suggerite permettono di intravedere come il chatbot possa continuare a evolversi e adattarsi alle mutevoli esigenze del mercato e delle aziende. In questo senso, il chatbot rappresenta non solo uno strumento attuale, ma una base solida per future innovazioni nel campo dell'assistenza virtuale e delle interazioni intelligenti.

Bibliografia

[1] ProjectSRL. *Projectsrl informations*. Disponibile su: `https://www.projectsrl.com/chi-siamo/`

[2] eCivis. *eCivis informations* Disponibile su: `https://www.ecivis.it/`

[3] OpenAI. (sept 2022) Disponibile su: `https://openai.com/index/whisper/`

[4] NLTK Project *NLTK informations* Disponibile su: `https://www.nltk.org/index.html`

[5] OpenAI. (2023). *API Documentation*. Disponibile su: `https://platform.openai.com/docs/`

[6] OpenAI. (2023). *Pricing Information*. Disponibile su: `https://openai.com/api/pricing/`

[7] Voiceflow. (2023). *Documentation for API Integrations*. Disponibile su: `https://docs.voiceflow.com/`

[8] Anthropic. (2023). *Claude: Safe and Steady Language Models*. Disponibile su: `https://www.anthropic.com/`

[9] DeepMind. (2023). *Gemini Language Model Information*. Disponibile su: `https://www.deepmind.com/`