⟨CodeHex16⟩

unipd.codehex16@gmail.com

Specifica Tecnica

Data	21/03/2025
Versione	0.2.0

Sommario

Specifica tecnica

Ruoli

Luca Ribon	Verificatore
Filippo Sabbadin	Redattore
Luca Rossi	Redattore

Registro delle Versioni

Versione	Data	Autore	Cambiamenti	Verificatore
0.2.0	10/04/2025	Filippo Sabbadin	Stesura sezioni iniziali	Luca Ribon
0.1.0	21/03/2025	Luca Rossi	Bozza iniziale struttura	Luca Ribon

Indice

Lista di immagini

Lista di tabelle

1. Introduzione

1.1. Scopo del prodotto

Il progetto consiste nella realizzazione di un **chatbot*** basato su modelli linguistici (**LLM***) pensato per i **fornitori** di beni, come bevande o alimenti, da offrire ai propri clienti. Questo sistema consente ai clienti di ottenere in modo semplice e immediato informazioni dettagliate sui prodotti o servizi disponibili, senza la necessità di contattare direttamente un operatore dell'azienda.

Il chatbot si integra con un'interfaccia dedicata al fornitore*, che permette di:

- Gestire clienti, faq e documenti contenenti le informazioni di riferimento utilizzate dal modello linguistico per generare risposte accurate e personalizzate.
- Personalizzare graficamente la piattaforma tramite l'inserimento del logo aziendale e la selezione di una palette colori.

Per garantire la massima compatibilità e facilità d'uso, il chatbot è accessibile tramite un'interfaccia web*, che può essere utilizzata su qualsiasi dispositivo su cui è installato un browser. I linguaggi principali usati nella webapp* sono HTML*, CSS*, JavaScript*, TypeScript e Python*, linguaggi ampiamente supportati da molti dispositivi.

1.2. Scopo del documento

Lo scopo del documento è fornire una panoramica dettagliata delle scelte progettuali e tecniche adottate per lo sviluppo del sistema. Qui verranno forniti i diagrammi UML delle classi e le scelte architetturali, oltre a una descrizione delle tecnologie utilizzate e delle API* implementate.

1.3. Glossario

Per facilitare la comprensione di questo documento, viene fornito un glossario che chiarisce il significato dei termini specifici utilizzati nel contesto del progetto. Ogni termine di glossario è contrassegnato con un asterisco «*» in apice e collegato direttamente alla pagina web del glossario, permettendo così di accedere immediatamente alla definizione completa del termine. Le definizioni sono disponibili nel documento: `Glossario.pdf`
e nella seguente pagina web: `https://codehex16.github.io/glossario`.

1.4. Riferimenti

1.5. Riferimenti normativi

- Capitolato C7 - Assistente Virtuale Ergon:
  `https://www.math.unipd.it/~tullio/IS-1/2024/Progetto/C7.pdf`
  (ultima consultazione: 03-03-2025);
- Regolamento del progetto didattico:
  `https://www.math.unipd.it/~tullio/IS-1/2024/Dispense/PD1.pdf`
- Norme di progetto:
  `https://codehex16.github.io/docs/2%20-%20PB/Norme-di-Progetto.pdf`
  (versione 1.0.0);

1.6. Riferimenti informativi

- Analisi dei requisiti:
  `https://codehex16.github.io/docs/2%20-%20PB/Analisi-dei-Requisiti.pdf`
  (versione 1.0.0);
- Diagrammi delle classi (UML):
  `https://www.math.unipd.it/~rcardin/swea/2023/Diagrammi%20delle%20Classi.pdf`
- Slide sui pattern architetturali del prof. Cardin:
  - introduzione ai pattern:
    `https://www.math.unipd.it/~rcardin/swea/2022/Software%20Architecture%20Patterns.pdf`
  - pattern creazionali:
    `https://www.math.unipd.it/~rcardin/swea/2022/Design%20Pattern%20Creazionali.pdf`
  - pattern strutturali:
    `https://www.math.unipd.it/~rcardin/swea/2022/Design%20Pattern%20Strutturali.pdf`
  - pattern comportamentali:
    `https://drive.google.com/file/d/1cpi6rORMxFtC91nI6_sPrG1Xn-28z8eI/view?usp=sharing`

- pattern Model View Controller:
  `https://www.math.unipd.it/~rcardin/sweb/2022/L02.pdf`

2. Tecnologie

In questa sezione vengono descritte le tecnologie utilizzate per lo sviluppo del sistema, suddivise in base alla loro funzione e al loro ambito di applicazione. Ogni tecnologia è accompagnata da una breve descrizione e dalle motivazioni che hanno portato alla sua scelta.

2.1. Tecnologie per la codifica

2.1.1. Linguaggi

Linguaggio	Motivazione	Versione
HTML	Utilizzato per la creazione della struttura e del contenuto delle pagine web. È il linguaggio di markup standard per la creazione di pagine web.	5
CSS	Utilizzato per la formattazione e lo stile delle pagine web. Permette di separare il contenuto dalla presentazione, migliorando la manutenibilità del codice.	3
JavaScript	Utilizzato per la creazione di interazioni dinamiche e reattive nelle pagine web.	ECMAScript 2024
Python	Scelto per la sua versatilità e facilità d'uso, è il linguaggio principale per lo sviluppo del back-end. Inoltre supporta e integra esaustivamente tutti i componenti esterni come gli LLM e i database vettoriali e non; inoltre la documentazione relativa a queste integrazioni è ampia e ben strutturata.	3.12
TypeScript	Scelto per la sua tipizzazione statica, migliora la qualità del codice e facilita la manutenzione. È utilizzato in combinazione con Svelte per lo sviluppo del front-end.	5.8.3

Tabella 1: Linguaggi utilizzati

2.1.2. Framework e librerie

Linguaggio	Motivazione	Versione
Svelte	Scelto per la sua semplicità e leggerezza, è il framework utilizzato per il rendering delle pagine del front-end. Permette di creare UI e UX gradevoli con una struttura e semantica del codice che il gruppo ha preferito rispetto ad altri framework.	5.25.10
SvelteKit	Scelto per la sua integrazione con Svelte, è un framework per lo sviluppo di applicazioni web. Permette di gestire implementare funzioni come ottimizzazione delle build, routing in modo semplice.	FastAPI
Framework scelto per la sua facilità nell di API REST. È utilizzato per il back-end per far comunicare tra loro i componenti software.	0.115.12 implementazione	LangChain

Linguaggio	Motivazione	Versione
Libreria scelta per l'integrazione tra modelli AI e database, permette di gestire documenti, contesto e query rivolte all'LLM integrato.	0.9.71	

Tabella 2: Framework e librerie utilizzati

2.1.3. Strumenti e servizi

Linguaggio	Motivazione	Versione
Git	Utilizzato per il versionamento del codice sorgente, permette di tenere traccia delle modifiche e collaborare con altri membri del team	2.49.0
GPT-4o-mini	Il modello utilizzato per il chatbot, scelto in base al prezzo e qualità delle risposte	-
Docker	Utilizzato per suddividere ed eseguire in container l'applicazione, rendendola facilmente distribuibile e scalabile in diversi ambienti	0.21.0 Build 28.0.4 Engine 4.40.0 Desktop 2.34.0 Compose
MongoDB	Database NoSQL utilizzato per memorizzare documenti, cronologia delle conversazioni e utenti. Scelto perché rende più facile e diretto memorizzare i file come formato json	8.0

Linguaggio	Motivazione	Versione
ChromaDB	Database vettoriale per memorizzare e recuperare embedding dei documenti, consentendo ricerche semantiche rapide per fornire risposte contestuali. Scelto per la sua velocità e facilità d'uso, è in grado di gestire grandi volumi di dati e query complesse.	1.0.4

Tabella 3: Strumenti e servizi utilizzati

2.2. Tecnologie per i test

Linguaggio	Motivazione	Versione
Github Actions	Utilizzato per l'integrazione continua e il testing automatico del codice. Permette di eseguire test e controlli di qualità ogni volta che viene effettuata una modifica al codice sorgente o prima di un merge con un altro branch	-
Pytest	Utilizzato per il testing del codice Python, permette di scrivere test in modo semplice e intuitivo. È stato scelto per la sua facilità d'uso e per la sua integrazione con FastAPI	8.3.5

Tabella 4: Tecnologie per il testing utilizzate

3. API

3.1. Endpoint di autenticazione

3.2. Endpoint di chat

3.3. Endpoint di gestione documenti

3.4. Altri endpoint

3.5. Errori e codici di ritorno

4. Architettura Front-end

Per il front-end sono stati utilizzati Svelte*, SvelteKit e TypeScript. SvelteKit è un framework* JavaScript* che integra Svelte e consente di creare interfacce utente reattive e performanti, mentre TypeScript* è un superset* di JavaScript che aggiunge tipizzazione statica al linguaggio, viene utilizzato per la logica di gestione e manipolazione della presentazione dell'applicazione.

4.1. Architettura delle pagine dell'applicazione web

Ogni pagina della webapp è composta da un file Svelte "*page.svelte*", e da un file Typescript "*page.server.ts*". Questa struttura segue il pattern Model-View-Controller* (MVC), in cui il file Svelte rappresenta la "view", mentre il file TypeScript rappresenta il "controller", il "model" invece sono i dati che vengono ricevuti dal back-end tramite le API.

4.1.1. File Svelte

Il file Svelte contiene il codice HTML e il CSS, utilizza le funzionalità di Svelte e le librerie create dal gruppo. Queste librerie contengono vari componenti grafici utilizzati nelle pagina web; inoltre, sfrutta le funzionalità delle pagine dinamiche di Svelte per aggiornare automaticamente l'interfaccia utente in base ai cambiamenti dello stato.

In un certo senso, tutti i file Svelte sono strutturati tramite il pattern *composite*, in quanto sono composti da più componenti che possono essere riutilizzati in altre pagine.

4.1.2. File Typescript

Il file TypeScript funge da *controller* del pattern MVC. Gestisce le chiamate API al back-end, elabora i dati ricevuti e li passa al file Svelte per la visualizzazione. Inoltre, gestisce gli eventi dell'interfaccia utente, come i click sui pulsanti e l'invio dei messaggi.

Ogni file svelte presenta una funzione *load*, che viene eseguita quando la pagina viene caricata.

4.2. Pagina di login

Percorso: */Suppl-AI*/src/routes/login* .
La pagina di login è la prima pagina che l'utente vede quando accede alla webapp.

Consente agli utenti di inserire le proprie credenziali (email e password) per accedere alla piattaforma.
Inoltre permette all'utente di avviare la procedura di recupero password in caso di smarrimento.
Presenta una variabile costante:
- API_URL: contiene l'URL dell'API del database.

Presenta due funzioni:
- *load*: controlla se l'utente è già autenticato tramite un token* cookie*. Se questo token esiste, reindirizza l'utente alla pagina principale.
- *actions*: viene eseguita quando l'utente invia il modulo di login. Prende lo username e password inseriti, e li manda attraverso una richiesta POST alla API per l'autenticazione. In caso di successo, reindirizza l'utente alla pagina principale. In caso di errore, mostra un messaggio di errore all'utente. Il cookie che mantiene l'accesso dura fino a 1 settimana, finita la quale l'utente dovrà reinserire le credenziali.

4.3. Homepage

Percorso: */Suppl-AI/src/routes* .
La homepage è la pagina principale della webapp. Consente agli utenti di visualizzare le chat disponibili e di crearne di nuove. Presenta una barra di navigazione, posta in basso per facilitarne l'utilizzo da dispositivi mobile, per accedere ad altre funzionalità della piattaforma, come la lista delle chat o le informazioni del profilo.
Presenta una variabile costante:
- API_URL: contiene l'URL dell'API del database.

Presenta una funzione:
- *load*: controlla se l'utente è autenticato tramite un token cookie. Se questo token esiste, reindirizza l'utente alla pagina di login. Inoltre, carica le chat disponibili per l'utente autenticato e le mostra nella homepage. Ritorna il token dell'utente e la lista delle chat.

4.4. Pagina account utente

Percorso: */Suppl-AI/src/routes/profilo* .

4.5. Pagina chat

Percorso: */Suppl-AI/src/routes/chat* .
La pagina chat mostra la conversazione tra l'utente e il chatbot. Presenta due variabili costanti:
- API_URL: contiene l'URL dell'API del database.
- LLM_URL: contiene l'URL dell'API del LLM.

Presenta le seguenti funzioni:
- *updateChatNameIfNeeded*: se la chat inizia ad avere più di 2 messaggi, viene mandata una richiesta POST alla API dell'LLM che chiede di cambiare il titolo della chat in base al contesto. Poi viene fatta una richiesta PUT alla API del database per memorizzare il titolo.
- *load*: controlla se l'utente è autenticato tramite un token cookie. Se non lo è, questo viene reindirizzato alla pagina di login. La chat viene caricata tramite un metodo GET alla API del database dove vengono richiesti tutti messaggi della chat.
- *actions*: viene eseguita quando l'utente invia un messaggio. Viene fatta una richiesta POST alla API del LLM per elaborare il messaggio e generare una risposta. La risposta viene poi mostrata nella chat. In caso di errore, la funzione ritorna un messaggio di errore.

5. Architettura logica

6. Architettura di deployment

7. Design pattern utilizzati

8. Diagramma delle classi

9. Database

10. Requisiti funzionali

Di seguito è riportata la tabella con i requisiti funzionali che erano stati stesi nel documento *Analisi dei requisiti*. Si ricorda che i requisiti sono stati divisi in due categorie in base alla lettera con cui finisce il codice del requisito:

- O: obbligatori
- D: desiderabili

10.1. Tracciamento

ID Requisito	Descrizione	Stato
R-01-F-O	L'utente deve poter accedere alla piattaforma	Soddisfatto
R-02-F-O	Nel caso in cui il fornitore acceda per la prima volta alla piattaforma, deve aggiornare la password	
R-03-F-O	Nel caso in cui il cliente acceda per la prima volta alla piattaforma, deve aggiornare la password	
R-04-F-O	Sia utente che fornitore devono poter cambiare la password liberamente	
R-05-F-D	L'utente non autenticato deve poter scegliere se reinserire le credenziali ad ogni accesso o se farle memorizzare alla piattaforma anche dopo il termine della sessione	
R-06-F-O	L'utente non autenticato deve poter recuperare la password nel caso in cui la dimentichi	
R-07-F-O	L'utente non autenticato deve essere notificato in caso di errore nell'inserimento dell'e-mail	

ID Requisito	Descrizione	Stato
R-08-F-O	L'utente non autenticato deve essere notificato in caso di errore nell'inserimento della password o nel caso in cui la password scelta non rispetti il formato richiesto	
R-09-F-O	Sia cliente che fornitore devono avere la possibilità di uscire dal proprio account	
R-10-F-O	Sia cliente che fornitore devono avere la possibilità di vedere la lista di tutte le chat in loro possesso	
R-11-F-O	Sia cliente che fornitore devono avere la possibilità di creare una nuova chat con un contesto pulito, che verrà aggiunta alla lista di quelle già presenti	
R-12-F-D	Sia cliente che fornitore devono avere la possibilità di modificare il titolo di una chat già esistente	
R-13-F-O	Sia cliente che fornitore devono avere la possibilità di aprire una chat singola dalla lista di tutte le chat in loro possesso	
R-14-F-O	Sia cliente che fornitore devono poter scrivere messaggi per comunicare con il chatbot	
R-15-F-O	L'utente che digita il messaggio deve essere avvisato nel caso in cui il messaggio scritto sia troppo lungo	

ID Requisito	Descrizione	Stato
R-16-F-O	Sia cliente che fornitore devono avere la possibilità di scrivere messaggi tramite FAQ preimpostate dal fornitore; se possibile il chatbot suggerirà delle FAQ utili in base al contesto	
R-17-F-O	Sia cliente che fornitore per comunicare con il chatbot devono poter trasmettere il messaggio scritto	
R-18-F-O	Sia cliente che fornitore devono ricevere la risposta elaborata dal chatbot in seguito all'invio di un messaggio; durante l'elaborazione della risposta l'utente deve ricevere un feedback che indica l'elaborazione della risposta	
R-19-F-O	Il cliente deve avere la possibilità di valutare la risposta ricevuta dal chatbot tramite l'opzione 'Pollice su/giù'	
R-20-F-O	Sia cliente che fornitore devono avere la possibilità di eliminare una chat presente nella lista di tutte le chat	
R-21-F-D	L'utente deve avere la possibilità di scegliere tra tema scuro e tema chiaro dell'interfaccia	
R-22-F-O	Il fornitore deve avere la possibilità di modificare la durata dello storico delle chat, ovvero il periodo di tempo massimo per cui vengono memorizzati e tenuti come contesto i messaggi per ogni chat	

ID Requisito	Descrizione	Stato
R-23-F-D	Il fornitore deve avere la possibilità di caricare il proprio logo per personalizzare la propria piattaforma fornita ai clienti	
R-24-F-D	Il fornitore deve avere la possibilità di cambiare i colori principali dell'interfaccia della propria piattaforma	
R-25-F-D	Il fornitore deve avere la possibilità di visualizzare le statistiche relative alle interazioni con il chatbot	
R-26-F-D	Il fornitore deve avere la possibilità di filtrare le statistiche visualizzate	
R-27-F-O	Il fornitore deve avere la possibilità di aggiungere gli account per i propri clienti	
R-28-F-O	Il fornitore deve essere avvisato nel caso in cui stia aggiungendo un account cliente già esistente	
R-29-F-O	Il fornitore deve avere la possibilità di eliminare un account cliente solo dopo aver autorizzato l'eliminazione tramite la propria password	
R-30-F-O	Il fornitore deve avere la possibilità di inserire documenti aziendali in modo da fornire ulteriore contesto all'chatbot	
R-31-F-O	Il fornitore deve sapere quando un file caricato, sia come logo che come documento aziendale, non sia nel formato corretto	

ID Requisito	Descrizione	Stato
R-32-F-O	Il fornitore deve avere la possibilità di visualizzare la lista dei documenti aziendali caricati nella piattaforma	
R-33-F-O	Il fornitore deve avere la possibilità di eliminare un documento aziendale dalla piattaforma solo dopo aver autorizzato l'eliminazione tramite la propria password	
R-34-F-O	Il fornitore deve avere la possibilità di aggiungere delle domande preimpostate nella piattaforma	
R-35-F-O	Il fornitore deve avere la possibilità di visualizzare la lista delle domande preimpostate inserite nella piattaforma	
R-36-F-O	Il fornitore deve avere la possibilità di modificare le domande preimpostate già inserite nella piattaforma	
R-37-F-O	Il fornitore deve avere la possibilità di eliminare le domande preimpostate dalla piattaforma solo dopo aver autorizzato l'eliminazione tramite la propria password	
R-38-F-O	Un utente non autenticato o il cliente che sta utilizzando la piattaforma deve essere avvisato nel caso in cui il sistema non sia raggiungibile, possibilmente specificando il motivo del malfunzionamento	

ID Requisito	Descrizione	Stato
R-39-F-O	Un utente non autenticato o il cliente che sta utilizzando la piattaforma deve essere avvisato nel caso in cui la richiesta che ha inviato contenga dati mancanti o errati; nel caso sia pertinente deve anche essere specificato il problema	

Tabella 5: Requisiti di funzionalità

10.2. Grafico dei requisiti obbligatori soddisfatti

10.3. Grafico dei requisiti desiderabili soddisfatti

11. Conclusioni