Documentazione Progetto Chat Locale

# Indice

1 Indice 2

1 Introduzione 3

1.1 Informazioni sul progetto 3

1.2 Abstract 3

1.3 Scopo 5

2 Analisi 6

2.1 Analisi del dominio 6

2.2 Analisi e specifica dei requisiti 7

2.3 Use case 9

2.4 Pianificazione 9

2.5 Analisi dei mezzi 12

2.5.1 Software 12

2.5.2 Hardware 12

3 Progettazione 13

3.1 Design dell’architettura del sistema 13

3.2 Design dei dati e database 14

3.3 Design delle interfacce 15

Design procedurale 17

4 Implementazione 18

5 Test 29

5.1 Protocollo di test 29

5.2 Risultati test 30

5.3 Mancanze/limitazioni conosciute 30

6 Consuntivo 30

7 Conclusioni 32

7.1 Sviluppi futuri 32

7.2 Considerazioni personali 32

8 Bibliografia 33

8.1 Bibliografia per articoli di riviste: 33

8.2 Bibliografia per libri 33

8.3 Sitografia 33

9 Glossario 34

10 Indice delle figure 34

11 Allegati 34

# Introduzione

## Informazioni sul progetto

In questo capitolo raccogliere le informazioni relative al progetto, ad esempio:

* Allievo: Andrea Negri
* Docente: Geo Petrini
* Scuola: SAMT
* Sezione: Informatica
* Modulo: M306
* Data inizio: 04.09.2024
* Data fine: 18.12.2024

## Scopo

Lo scopo del progetto (scopi didattici/scopi operativi). Dovrebbe descrivere il mandato, ma non vanno ricopiate le informazioni del quaderno dei compiti (che va invece allegato).

Lo scopo di questo progetto è quello di poter chattare localmente grazie ad un’applicazione Web. La chat avrà varie funzionalità tra cui potere vedere i messaggi altrui, scrivere messaggi di testo, mandare foto, mandare audio, mandare una foto/audio con un testo che la descrive, ecc.

Inoltre sarà indispensabile fare un sistema di login e registrazione per tenere traccia degli utenti che vi accedono e per ricordare le loro credenziali.

Dovrà anche essere una chat dove tutti gli utenti loggati ci possono accedere e chattare con

# Analisi

## Analisi del dominio

Questo prodotto sarà un applicativo web che consentirà a due o più utenti di comunicare su un canale privato. Gli utenti potranno scrivere e inviare messaggi in un canale aperto a tutti.

La mia idea era quella di creare la chat con uno stile simile a WhatsApp ma rendendola pubblica in modo da potere dialogare con persone sconosciute di determinati argomenti.

Deve essere una chat adatta a qualsiasi tipo di persona che si voglia interfacciare. Il focus di questo progetto sono persone non competenti sul fattore informatica che approcciano app Web per divertimento o intrattenimento.

Per fare questa chat si possono creare sistemi che possano gestire tramite WebSocket l’applicazione gestendo anche un Database dove andare a salvare i dati dei messaggi e gli utenti. In alternativa si potrebbe scegliere di utilizzare un sistema che comprenda già questa gestione e la gestisca da sola, rendendo il progetto meglio strutturato e più intuitivo da utilizzare come Firebase.

Per questo progetto sono dunque necessari alcune conoscenze di base come quelle dei protocolli, dell’Internet e conoscenze di base di linguaggi come HTML, CSS e Javascript. Bisogna anche conoscere un minimo di JSON dato che se scegliamo Firebase lavora con JSON, oppure MySQL o altri DB se decidiamo di andare con un sistema partendo da 0.

## Analisi e specifica dei requisiti

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-01 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Login con nome utente e password. | | | | |
| **Note** | Deve esistere una pagina di login dove l’utente può fare login e “sbloccare” il suo account con cui può scrivere sulle chat. | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Il login deve essere implementato sia con Google che con un altro dominio | | | | |
| **002** | Per il login deve esistere anche una registrazione con verificazione tramite email | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-02 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Avere un server dove gli utenti si collegano e possono scrivere messaggi | | | | |
| **Note** | Serve un server dove gli utenti possano accedere per chattare con altri utenti | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Utilizzare Firebase | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-03 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Profilo con immagine personalizzata e stato di attività | | | | |
| **Note** | Il profilo di ogni utente dovrà avere un’immagine e uno stato di attività | | | | |
| **Sotto requisiti** | | | | | |
| **001** | La immagine si potrà anche cambiare | | | | |
| **002** | Dovrà esserci una lista di tutte le persone online in quel momento | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-04 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Chat multifunzione | | | | |
| **Note** | Nelle chat oltre a scrivere si potranno inviare immagini e audio con bassa priorità. | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Utilizzare Dropzone.js per immettere i file | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-05 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Storico delle chat di una settimana | | | | |
| **Note** | Le chat dovranno essere salvate per una settimana e poi andare perse | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Le chat dovranno essere salvate quando l’utente finisce di conversare con un altro utente | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-06 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Responsive | | | | |
| **Note** | Il sito dovrà essere responsive | | | | |

**Spiegazione elementi tabella dei requisiti:**

**ID**: identificativo univoco del requisito

**Nome**: breve descrizione del requisito

**Priorità**: indica l’importanza di un requisito nell’insieme del progetto, definita assieme al committente. Ad esempio, poter disporre di report con colonne di colori diversi ha priorità minore rispetto al fatto di avere un database con gli elementi al suo interno. Solitamente si definiscono al massimo di 2-3 livelli di priorità.

**Versione**: indica la versione del requisito. Ogni modifica del requisito avrà una versione aggiornata.

Sulla documentazione apparirà solamente l’ultima versione, mentre le vecchie dovranno essere inserite nei diari.

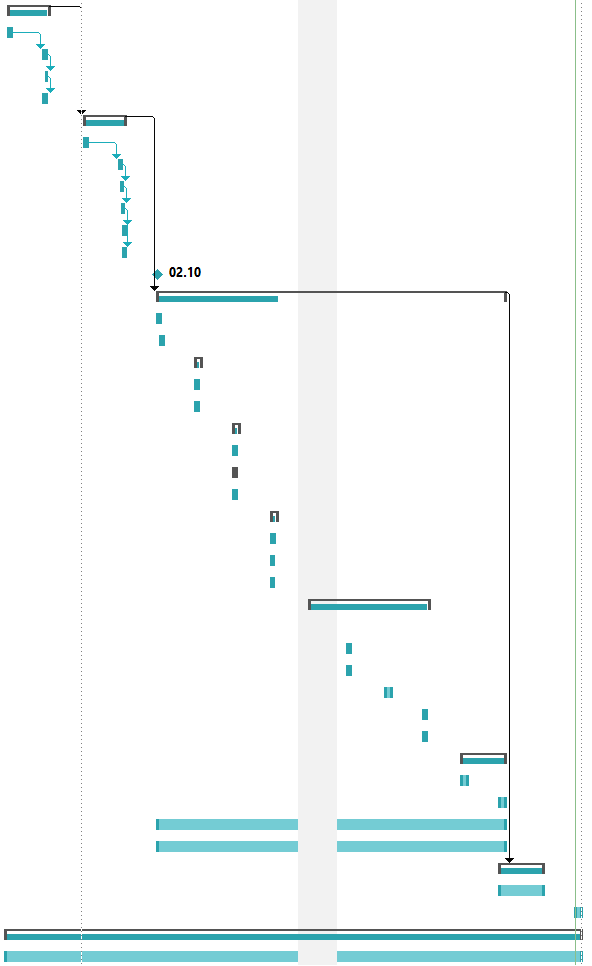
**Note**: eventuali osservazioni importanti o riferimenti ad altri requisiti.

**Sotto requisiti**: elementi che compongono il requisito.

## Use case

\*Tuttavia questa Use case non rappresenta completamente la struttura finale del progetto dato che sono state modificate alcune cose durante l’implementazione del progetto.

## Pianificazione

**Gantt Consuntivo**

## Analisi dei mezzi

Elencare e descrivere i mezzi disponibili per la realizzazione del progetto. Ricordarsi di sempre descrivere nel dettaglio le versioni e il modello di riferimento.

Firebase:

* Firebase Realtime Database
* Firebase Authentication
* Firestore Database

### Software

Backend:

* Firebase (Backend e Servizi Cloud)

Frontend:

* Javascript

Stile e formattazione:

* HTML
* CSS

Schemi:

* Microsoft Visio

IDE di sviluppo:

* Visual Studio Code

Browser utilizzato:

* Firefox

### Hardware

Questo progetto è stato pensato per utenti Windows ma anche altri OS possono interfacciarsi. È anche possibile accedervi da un dispositivo mobile.

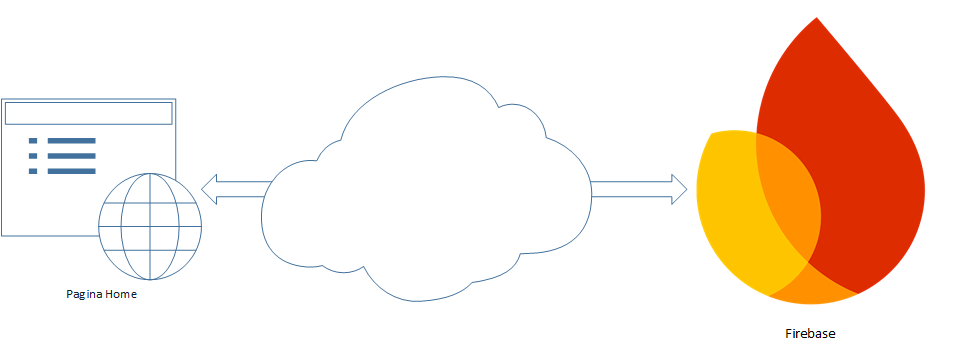
# Progettazione

Questo capitolo descrive esaustivamente come deve essere realizzato il prodotto fin nei suoi dettagli. Una buona progettazione permette all’esecutore di evitare fraintendimenti e imprecisioni nell’implementazione del prodotto.

## Design dell’architettura del sistema

Descrive:

* La struttura del programma/sistema lo schema di rete...
* Gli oggetti/moduli/componenti che lo compongono.
* I flussi di informazione in ingresso ed in uscita e le relative elaborazioni. Può utilizzare *diagrammi di flusso dei dati* (DFD).
* Eventuale sitemap

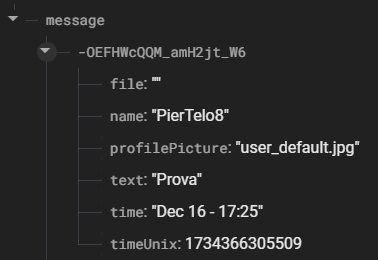
Design dell’architettura del sistema:

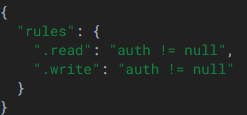
Questo schema rappresenta semplicemente come la pagina della chat comunica con Firebase.

Per l’comunicazione con la chat Firebase Realtime Database utilizza i WebSocket per gestire la sincronizzazione in tempo reale come nel caso di eventi come l’invio di messaggi. Per azioni che non necessitano di cambiamenti real-time il Firestore Database utilizza il protocollo HTTPS REST tramite un API dedicata.

## Design dei dati e database

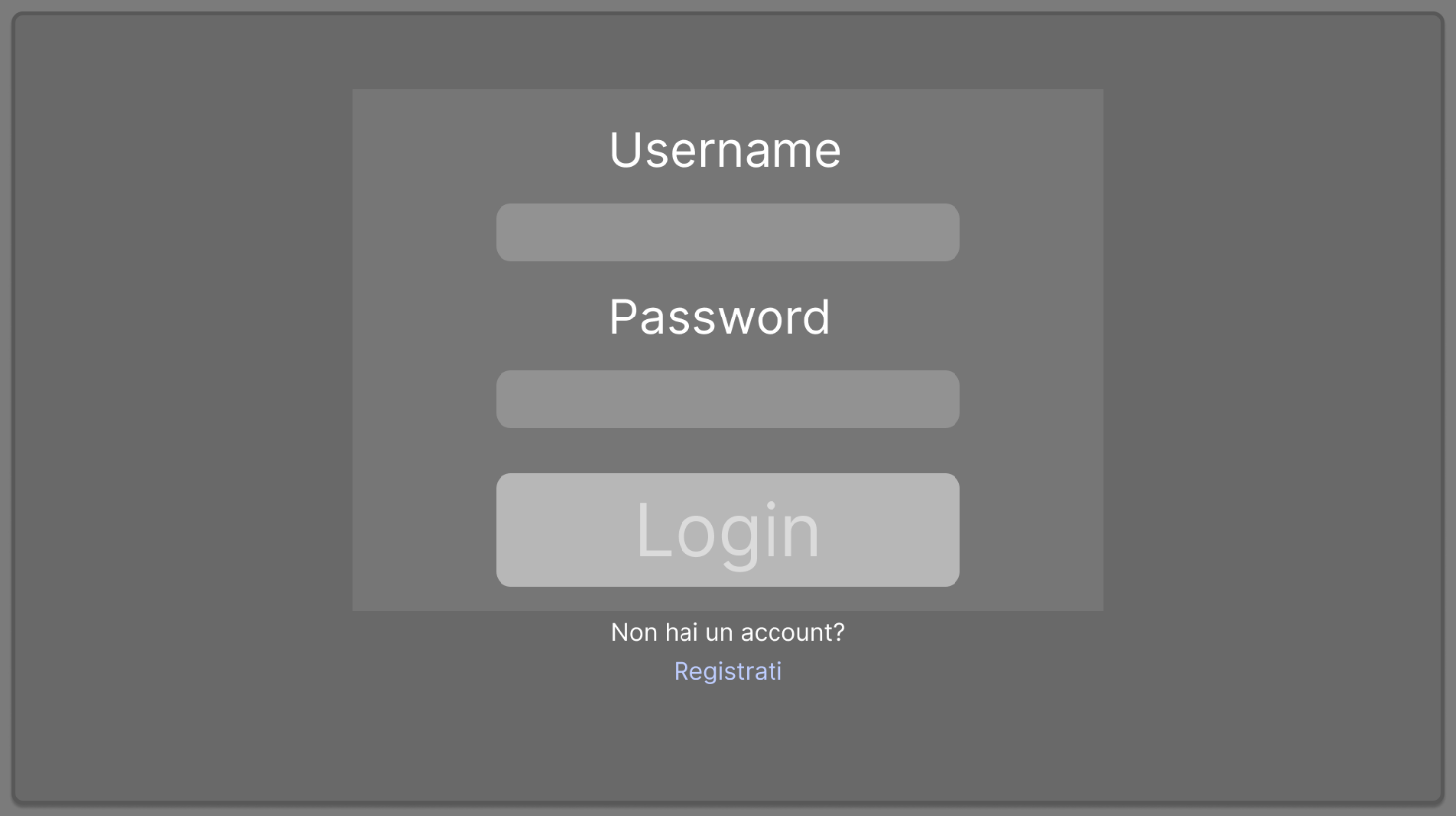
I dati verranno salvati in un database noSQL che salva in dati in una struttura ad albero JSON integrato in Firebase Realtime Database.

Struttura dei dati:

Regole del DB:

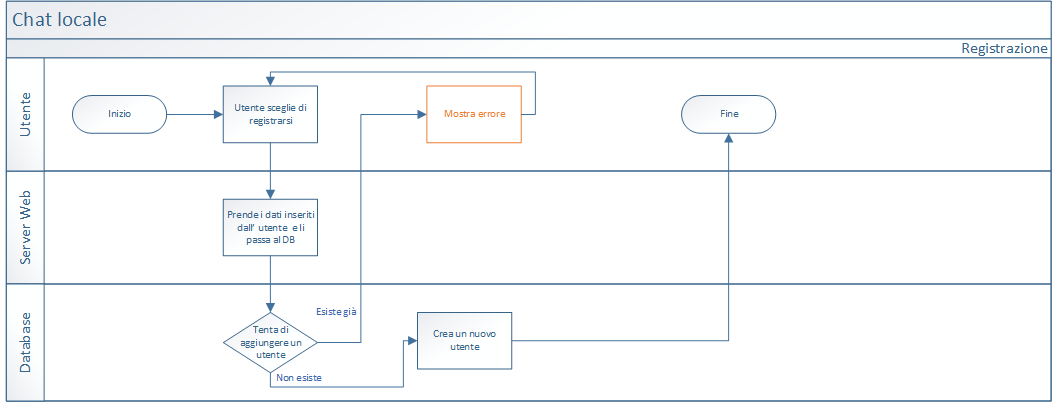
Queste regole danno il diritto solo a chi è autenticato alla chat di leggere o scrivere sul DB

## Design delle interfacce





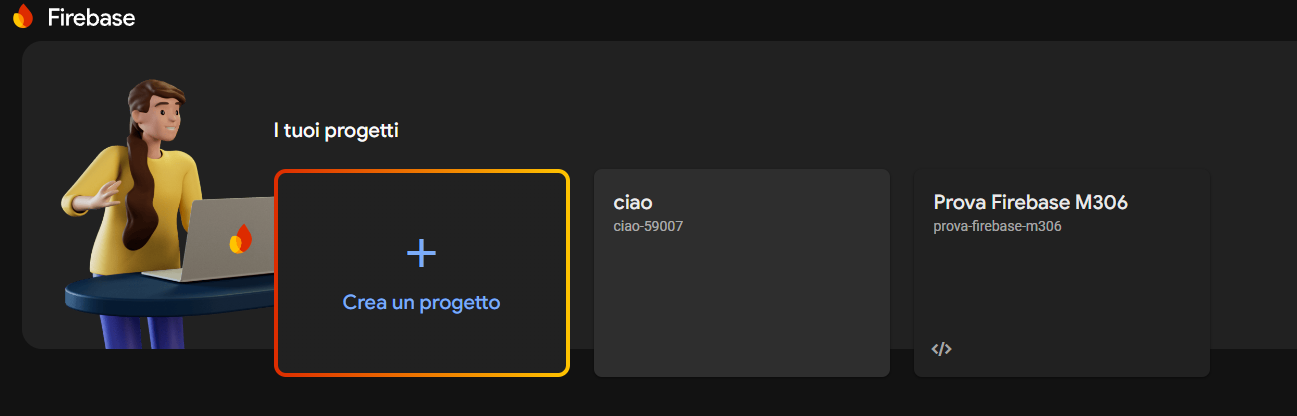
## Design procedurale

Swimlane Chat Locale

\*Tuttavia questa Swimlane non rappresenta la struttura finale del progetto dato che sono state modificate alcune cose durante l’implementazione del progetto.

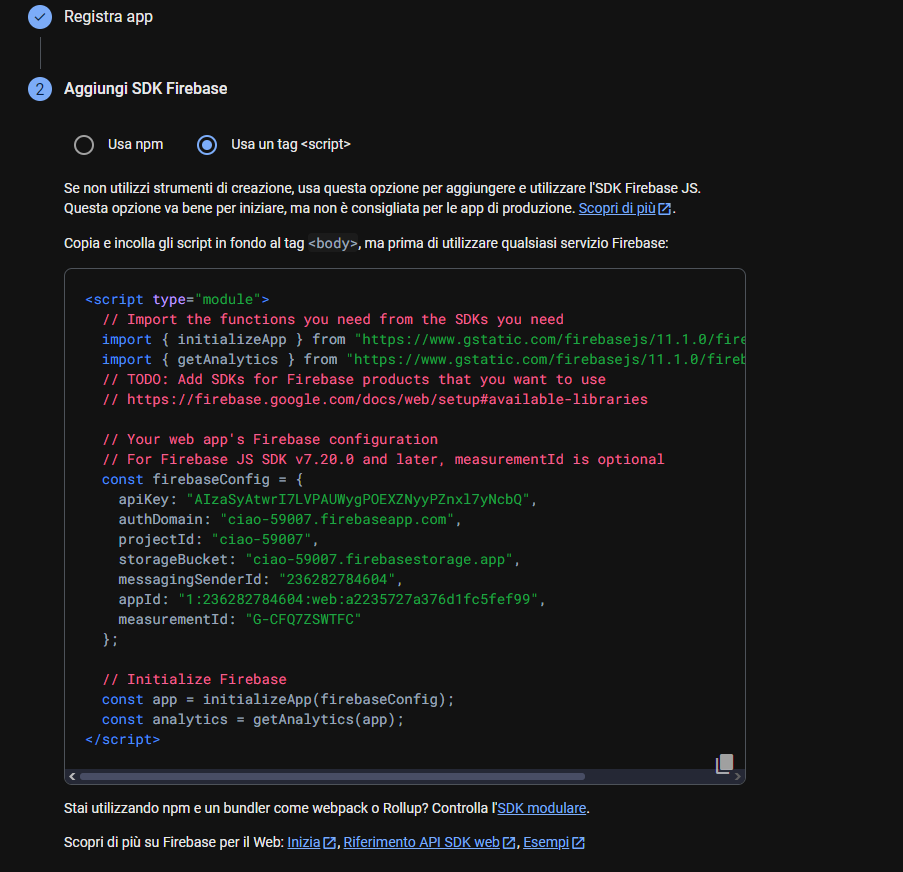
# Implementazione

## Creazione progetto Firebase Web

Come prima cosa recarsi sul sito di Firebase e fare il login. Andare sotto crea un nuovo progetto e inserire il nome che si vuole dare.

Dopodiché aggiungere una nuova applicazione Web da utilizzare per la nostra chat.

Dare un nuovo nome al progetto e continuare la configurazione.

Quando si arriva a “Aggiungi SDK Firebase” bisogna andare sotto “Usa un tag <script>” e copiare tutto in un file così da avere la configurazione che servirà a Javascript per interfacciarsi con Firebase.

Per finire cliccare su “Vai alla console” che vi porterà alla console di Firebase dove avete tutto il necessario per poter gestire Firebase.

## Codice

Premessa: Tutti i file sono contenuti nel package “ChatLocale”

Serve a importare alcuni metodi utili già creati di Firebase alla chat

Per collegare l'applicazione a **Firebase** (in questo caso si tratta del progetto Firebase web**)** è stata configurata la piattaforma tramite le credenziali fornite. La configurazione include:

* **API Key**: necessaria per identificare il progetto.
* **Auth Domain** e **Database URL**: per l'autenticazione e il database in tempo reale.
* **App ID** e **Sender ID**: utili per l'integrazione delle funzionalità di Firebase.

Eccone un esempio:

Nel caso del login aggiungiamo anche questo codice per gestire il login con Google.

Questa costante serve per il Realtime Database di Firebase mentre la seconda per il Firestore Database.



La funzione **loadMessages()** carica i messaggi(in ordine di orario) presi dal Realtime Database e li aggiunge al div che contiene tutti i messaggi. Inoltre etichetta i messaggi inviati dall’utente se sono i propri o di un altro utente (per poi nel CSS cambiare la posizione e il colore).

Metodi e proprietà

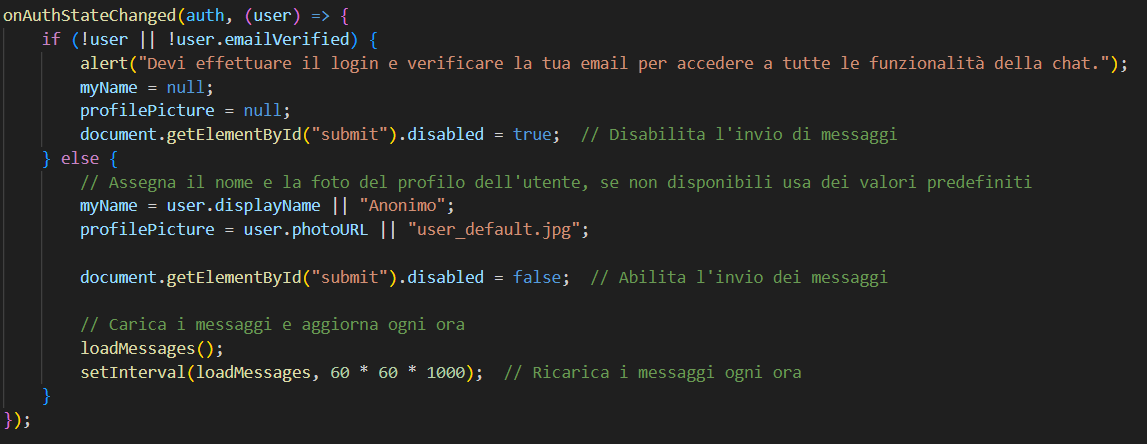
**query()** 🡪È una funzione che permette di combinare un riferimento al database con filtri e ordinamenti.

**ref()** 🡪 È una funzione che viene utilizzata per ottenere un riferimento a un percorso specifico all'interno del database. In questo caso in “message”

**onChildAdded()🡪**Il metodo onChildAdded si attiva per ogni messaggio presente in memoria

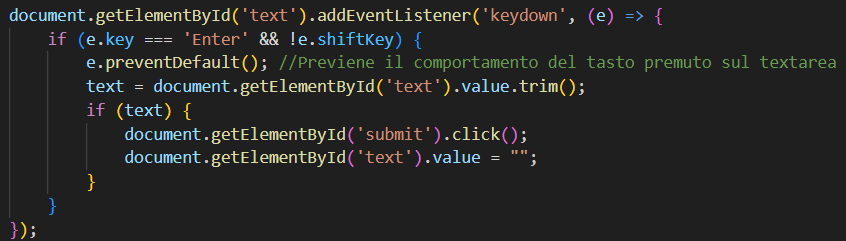
**data.val()** 🡪 data rappresenta il messaggio estratto grazie al onChildAdded() e il val()un metodo fornito dall'API di Firebase che converte i dati del messaggio(Nodo del DB dato che viene utilizzato un DB noSQL che utilizza una struttura a documenti JSON) in un oggetto JavaScript leggibile.

**scrollTop** 🡪scorre automaticamente fino all’ ultimo messaggio così da creare quel effetto di nuovo messaggio che appare.

Questo pezzo di codice gestisce il cambio di stato dell’autenticazione. Serve a decidere se fare vedere o no il contenuto della chat se l’utente è loggato.

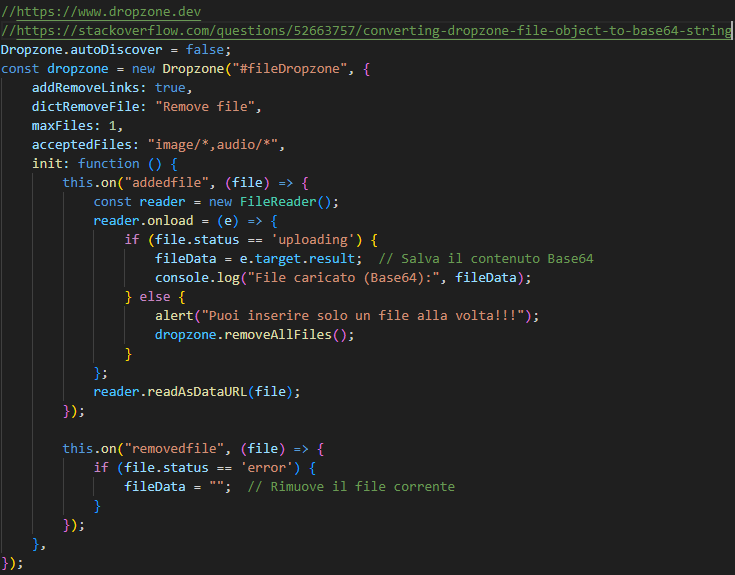
**onAuthStateChanged()** 🡪 È una funzione di Firebase Authentication che viene eseguita ogni volta che lo stato di autenticazione dell'utente cambia.

**setInterval() 🡪** È una funzione che esegue una determinata operazione o funzione a intervalli di tempo definite in essa(il tempo è in millisecondi).

Questa funzione si occupa di simulare il click del bottone di invio del messaggio con il tasto “Enter” della tastiera, lasciando però andare a capo l’utente nel textarea se viene premuto anche shift insieme al tasto “Enter”.

**click()** 🡪 Funzione che simula il click del tasto di invio dei messaggi

**preventDefault()** 🡪 Funzione che previene il comportamento di default di un oggetto HTML(in questo caso del nostro textarea)

Questa porzione di codice serve a gestire l’ input dei file con una libreria chiamata Dropzone.js che migliora il semplice input di html implementando anche la funzione di drag-and-drop

**Configurazione di Dropzone**

* **addRemoveLinks**: Aggiunge un link per rimuovere i file caricati.
* **dictRemoveFile**: Testo del link per rimuovere il file.
* **maxFiles**: Imposta un limite massimo di file caricabili (1 nel nostro caso).
* **acceptedFiles**: Limita il caricamento a file di tipo immagine (image/\*) o audio (audio/\*).

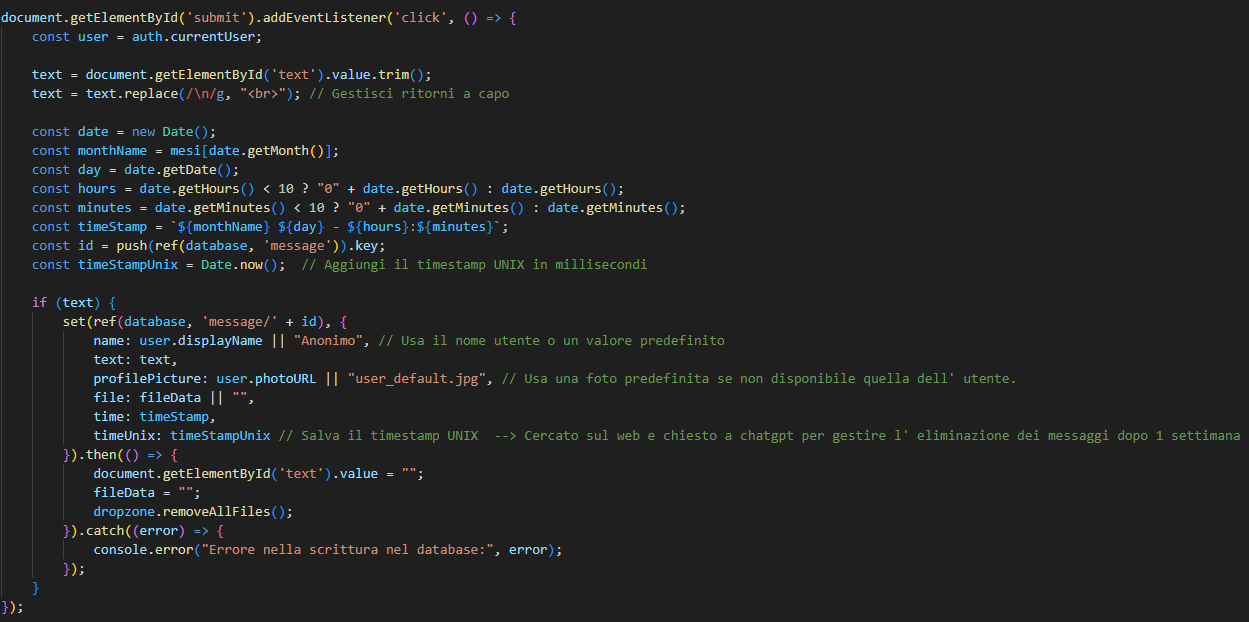
**this.on()** 🡪 È un listener di Dropzone per eventi specifici che avvengono solo durante il ciclo di upload di file.

**FileReader()** 🡪 È un oggetto che permette di leggere file in vari formati direttamente dal browser.

**removeAllFiles()** 🡪 È un metodo di Dropzone che permette di rimuovere tutti i file dalla zona.

**readAsDataURL()** 🡪 È un metodo fornito dalla classe FileReader() che permette di convertire il contenuto di un file in Base64.

Dropzone.autoDiscover = false; 🡪 Serve a disabilitare il comportamento automatico della Dropzone**.**

Questa porzione di codice ha il compito di gestire l’invio dei messaggi tramite il click del bottone di invio(solo se il testo del messaggio non è vuoto).

**auth.currentUser** 🡪 Serve a ottenere le informazioni dell'utente autenticato.

**set()** 🡪 È un metodo di Firebase che ha la funzione di sostituire o come in questo caso aggiungere i dati nella posizione indicata

**push()** 🡪 È un metodo di Firebase che ha la funzione di generare un nuovo id del messaggio così da poterlo usare dopo per salvare sotto quell’ id tramite il comando set()

Questa porzione di codice gestisce il pulsante di logout.



**sessionStorage.removeItem()** 🡪 Serve a togliere la variabile di sessione indicata.

**window.location.replace()** 🡪 Serve a fare il redirect sulla pagina indicata.

Funzione che gestisce interamente la registrazione di un nuovo utente

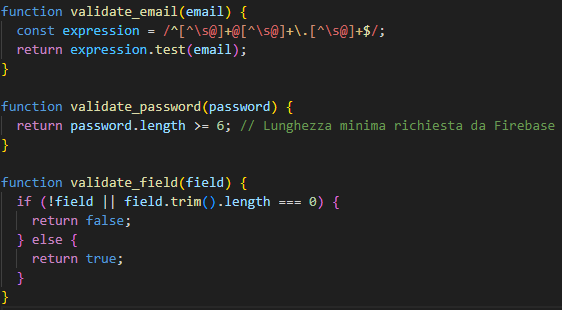
**trim()** 🡪 Funzione che rimuove eventuali spazi vuoti all'inizio e alla fine dei valori.

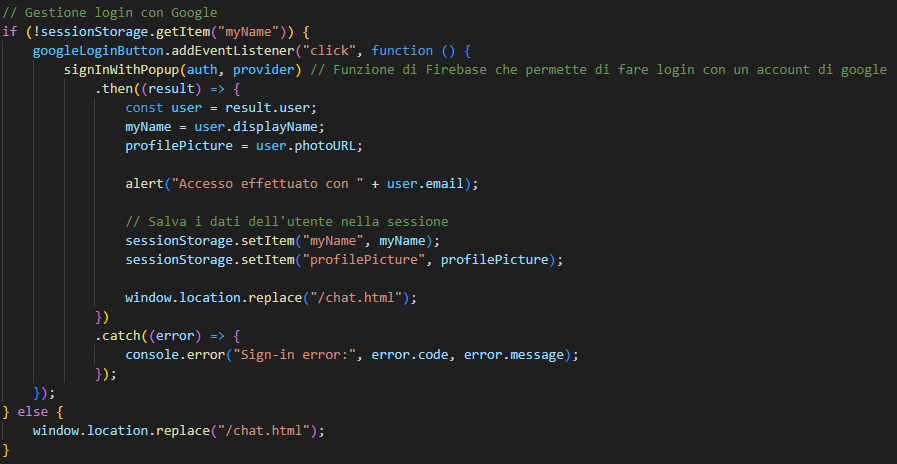
**createUserWithEmailAndPassword()** 🡪 È una funzione di Firebase Authentication che permette di creare un nuovo utente con l'email e la password fornita.

**sendEmailVerification()** 🡪 È una funzione di Firebase Authentication che permette di mandare un email di verifica a chi crea l’ account.

**firestore.collection('users').doc(user.uid).set(userData); 🡪** Questa riga salva i dati dell'utente nel Firestore Database.

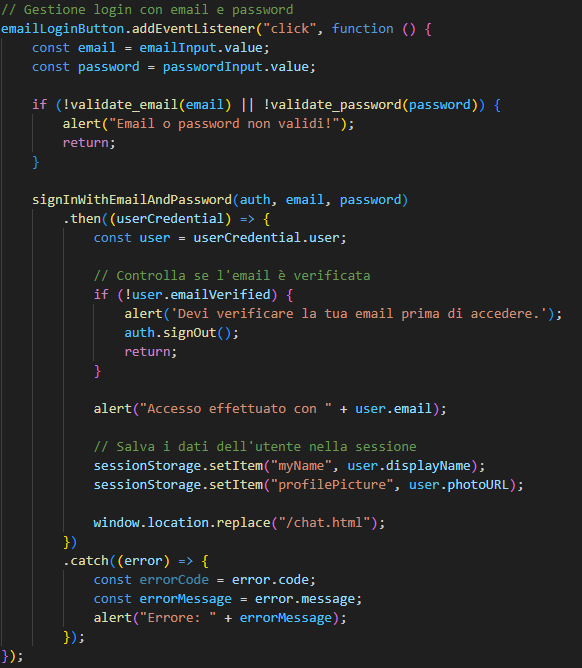
Queste sono le 3 funzioni che uso per validare i dati in ingresso.

**test()** 🡪 È una funzione che serve a utilizzare le regular expression.

Questa è il pezzo di codice che gestisce il login con Google

**sessionStorage.setItem()** 🡪 Serve a creare e popolare una variabile di sessione.

signInWithPopup() 🡪 È una funzione di Firebase Authentication che permette di iscriversi e loggarsi tramite account di Google.

Questo è il pezzo di codice che gestisce il login con l’account registrato tramite mail e password

emailVerified() 🡪 Metodo che ritorna “true” solo se è stata verificata la mail mandata. In questo caso gestisce il fatto che non puoi loggarti con il tuo account appena creato se non confermi la mail.

# Test

## Protocollo di test

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-001  Req-01 | **Nome** | Test login |
| **Descrizione** | Test prova del login. | | |
| **Prerequisiti** | Avere un browser che possa connettersi a Internet e una mail funzionante. | | |
| **Procedura** | 1. Andare sotto la pagina di default e premere su “Registrati” 2. Inserire i dati per la registrazione. 3. Fare login 4. Controllare che sia possibile entrare nella chat | | |
| **Risultati attesi** | Possibilità di vedere gli altri messaggi della chat | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-002  Req-02 | **Nome** | Test invio di un messaggio |
| **Descrizione** | Test di invio di un nuovo messaggio nella chat | | |
| **Prerequisiti** | Avere un browser che possa connettersi a Internet e una mail funzionante. | | |
| **Procedura** | 1. Scrivere qualcosa dentro al textarea 2. Inviare il messaggio | | |
| **Risultati attesi** | Possibilità di scrivere nella chat | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-003  Req-03 | **Nome** | Test stato di attività |
| **Descrizione** | Non implementato | | |
| **Prerequisiti** | Avere un browser che possa connettersi a Internet e una mail funzionante. | | |
| **Procedura** | Non implementato | | |
| **Risultati attesi** | Stato di attività dell’utente | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-004  Req-04 | **Nome** | Test caricamento immagine con Dropzone.js |
| **Descrizione** | Test caricamento immagine (.jpg o .png) con Dropzone.js | | |
| **Prerequisiti** | Avere un browser che possa connettersi a Internet e una mail funzionante. | | |
| **Procedura** | 1. Trascinare sopra alla Dropzone un file salvato in locale. 2. Inviare il messaggio | | |
| **Risultati attesi** | Compare il messaggio con l’immagine. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-005  Req-04 | **Nome** | Test caricamento audio con Dropzone.js |
| **Descrizione** | Test caricamento audio con Dropzone.js | | |
| **Prerequisiti** | Avere un browser che possa connettersi a Internet e una mail funzionante. | | |
| **Procedura** | 1. Trascinare sopra alla Dropzone un file salvato in locale. 2. Inviare il messaggio | | |
| **Risultati attesi** | Compare il messaggio con l’audio. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-006  Req-05 | **Nome** | Test eliminazione messaggi dopo 1 settimana |
| **Descrizione** | Test di eliminazione di ogni messaggio con ciclo di vita di più di 1 settimana. | | |
| **Prerequisiti** | Avere un browser che possa connettersi a Internet e una mail funzionante. | | |
| **Procedura** | 1. Trascinare sopra alla Dropzone un file salvato in locale. 2. Inviare il messaggio | | |
| **Risultati attesi** | Compare il messaggio con l’audio. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-007  Req-06 | **Nome** | Test responsive della pagina |
| **Descrizione** | Test di eliminazione di ogni messaggio con ciclo di vita di più di 1 settimana. | | |
| **Prerequisiti** | Avere un browser che possa connettersi a Internet e una mail funzionante. | | |
| **Procedura** | 1. Trascinare sopra alla Dropzone un file salvato in locale. 2. Inviare il messaggio | | |
| **Risultati attesi** | Compare il messaggio con l’audio. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-008  Req-01 | **Nome** | Test email di verifica |
| **Descrizione** | Test dell’arrivo della mail di verifica della registrazione | | |
| **Prerequisiti** | Avere un browser che possa connettersi a Internet e una mail funzionante. | | |
| **Procedura** | 1. Provare a registrarsi 2. Fare login 3. Andare sotto la email inserita 4. Verificare l’ email 5. Riprovare a fare login | | |
| **Risultati attesi** | Prima darà un errore e dirà che devi verificare la mail e dopo lascerà entrare | | |

## Risultati test

Tabella riassuntiva in cui si inseriscono i test riusciti e non del prodotto finale. Se un test non riesce e viene corretto l’errore, questo dovrà risultare nel documento finale come riuscito (la procedura della correzione apparirà nel diario), altrimenti dovrà essere descritto l’errore con eventuali ipotesi di correzione.

|  |  |  |
| --- | --- | --- |
| **Test Case** | **Risultato ottenuto** | **Stato** |
| TC-001 |  | Passato |
| TC-002 |  | Passato |
| TC-003 | Non implementato | Fallito |
| TC-004 |  | Passato |
| TC-005 |  | Passato |

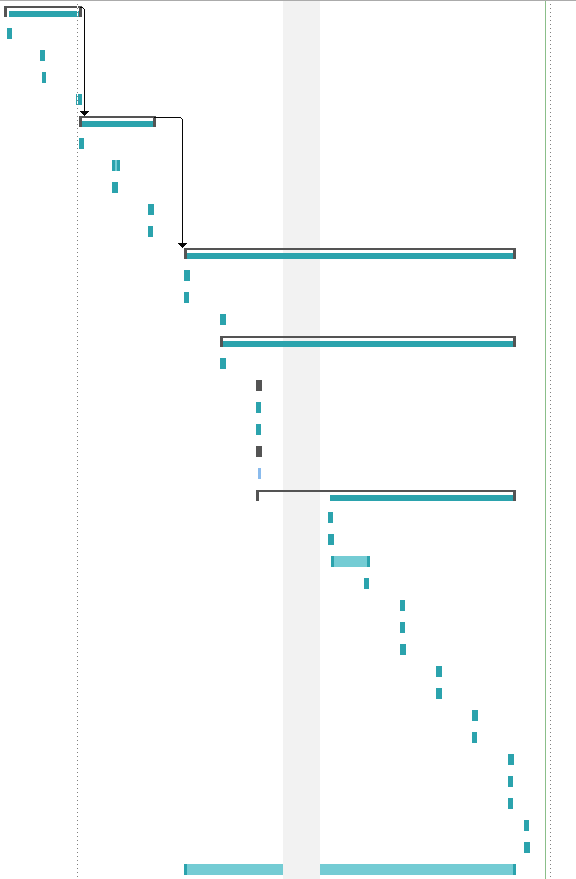
|  |  |  |
| --- | --- | --- |
| TC-006 |  | Passato |
| TC-007 |  | Passato  \*unico problema il textarea che non è completamente corretto |
| TC-008 |  | Passato |

## 

## Mancanze/limitazioni conosciute

Lista degli utenti online e sicurezza nel login.

# Conclusioni

Quali sono le implicazioni della mia soluzione? Che impatto avrà? Cambierà il mondo? È un successo importante? È solo un’aggiunta marginale o è semplicemente servita per scoprire che questo percorso è stato una perdita di tempo? I risultati ottenuti sono generali, facilmente generalizzabili o sono specifici di un caso particolare? ecc.

## Sviluppi futuri

Migliorie sul fattore sicurezza dei dati e login dati che ci sono alcuni modi per aggirarli.

Aggiunta di utenti amministratori e blocco di parole volgari.

Aggiunta di una lista degli amici con

## Considerazioni personali

Da questo progetto ho imparato come strutturare una documentazione di base e i contenuti dentro di essa. Ho imparato a utilizzare Firebase e le sue funzioni cercando su internet tutorial e pagine oppure facendomi aiutare nelle parti più ardue e in cui ero meno sicuro da ChatGPT. Ho anche colmato alcune lacune che avevo dal 2° anno in Javascript. Ho imparato a gestire un progetto in tutta la sua interezza dall’inizio a fine con tanto di diari.

Ho trovato questo progetto molto interessante su tanti punti di vista dato che era una cosa nuova e mi avrebbe aiutato a rendere più stabili le mie conoscenze della materia sviluppando un progetto che mi interessava davvero.

Inoltre non avevo mai usato GIT e ho imparato a usarlo per il version control.

# Bibliografia

## Sitografia

* <https://chatgpt.com/>, ChatGPT, 11-12-2025
* <https://firebase.google.com/docs>, Firebase Documentation, 18-12-2024
* <https://it.wikipedia.org/wiki/WebSocket>, WebSocket Wikipedia, 18-12-2024
* <https://www.youtube.com/watch?v=smHoNUA70q8>, Tutorial Firebase, 18-09-2024
* <https://www.html.it/articoli/firebase/>, Firebase Base Tutorial, 25-09-2024
* <https://www.youtube.com/watch?v=EPlXPdNvQEY>, Tutorial caricamento immagini, 25-09-2024
* <https://www.youtube.com/watch?v=Uhbn1KmiNbg>, Google Authentication Tutorial, 16-10-2024
* <https://firebase.google.com/docs/auth/web/google-signin?hl=it>, Google Authentication Tutorial, 16-10-2024
* <https://www.youtube.com/watch?v=7hB2ASYBkXo>, Delete data Firebase after a week, 20-11-2024

# Glossario

|  |  |
| --- | --- |
| **Termine** | **Significato** |
| WebSocket | È un protocollo basato su TCP che definisce le modalità di scambio tra le reti |

# Allegati

Gantt: