Chat con Firebase

[1 Introduzione 3](#_Toc91112127)

[1.1 Informazioni sul progetto 3](#_Toc91112128)

[1.2 Abstract 3](#_Toc91112129)

[1.3 Scopo 3](#_Toc91112130)

[2 Analisi 4](#_Toc91112131)

[2.1 Analisi del dominio 4](#_Toc91112132)

[2.2 Analisi e specifica dei requisiti 4](#_Toc91112133)

[2.3 Use case 6](#_Toc91112134)

[2.4 Pianificazione 7](#_Toc91112135)

[2.5 Analisi dei mezzi 7](#_Toc91112136)

[2.5.1 Software 7](#_Toc91112137)

[2.5.2 Hardware 7](#_Toc91112138)

[3 Progettazione 7](#_Toc91112139)

[3.1 Design dell’architettura del sistema 7](#_Toc91112140)

[3.2 Design dei dati e database 8](#_Toc91112141)

[3.3 Design delle interfacce 8](#_Toc91112142)

[3.4 Design procedurale 8](#_Toc91112143)

[4 Implementazione 9](#_Toc91112144)

[4.1 Creazione di un progetto su Firebase 9](#_Toc91112145)

[4.2 Autenticazione 10](#_Toc91112146)

[4.2.1 Registrazione 10](#_Toc91112147)

[4.2.2 Login 10](#_Toc91112148)

[4.2.3 Logout 10](#_Toc91112149)

[4.3 Inviare un messaggio 11](#_Toc91112150)

[4.4 Creare un canale 11](#_Toc91112151)

[4.5 Mettere in mostra la lista dei canali e le chat 12](#_Toc91112152)

[4.6 Cambio canale 13](#_Toc91112153)

[4.7 Eliminare un canale 13](#_Toc91112154)

[4.8 Modifica canale 13](#_Toc91112155)

[4.8.1 Aggiungere gli utenti nel gruppo 13](#_Toc91112156)

[4.8.2 Eliminare l’utente nel gruppo 13](#_Toc91112157)

[4.8.3 Modificare la durata del messaggio in un gruppo 13](#_Toc91112158)

[4.9 Eliminazione dei messaggi 13](#_Toc91112159)

[4.10 Broadcast 13](#_Toc91112160)

[4.11 Ban 14](#_Toc91112161)

[4.12 Eliminazione dei vecchi messaggi 14](#_Toc91112162)

[5 Test 15](#_Toc91112163)

[5.1 Protocollo di test 15](#_Toc91112164)

[5.2 Risultati test 19](#_Toc91112165)

[5.3 Mancanze/limitazioni conosciute 20](#_Toc91112166)

[6 Consuntivo 21](#_Toc91112167)

[7 Conclusioni 21](#_Toc91112168)

[7.1 Sviluppi futuri 21](#_Toc91112169)

[7.2 Considerazioni personali 21](#_Toc91112170)

[8 Bibliografia 21](#_Toc91112171)

[8.1 Bibliografia per articoli di riviste: 21](#_Toc91112172)

[8.2 Bibliografia per libri 21](#_Toc91112173)

[8.3 Sitografia 21](#_Toc91112174)

[9 Allegati 21](#_Toc91112175)

# Introduzione

## Informazioni sul progetto

Sono Mattia Pasquini, uno studente del terzo anno di informatica alla Scuola d’Arti e Mestieri a Trevano. Ho realizzato questo progetto dal 09.09.2021 al 23.12.2021, con la supervisione del docente Geo Petrini, durante il modulo 306.

## Abstract

Il progetto si basa sulla comunicazione tra utenti in una piattaforma, utilizzando il database di Firebase. Quindi richiede di conoscere bene il suo meccanismo e cosa potrebbe offrirci questo servizio. Gli utenti devono autenticarsi per poter utilizzare la piattaforma.

Gli utenti hanno una lista dei canali e possono scrivere e leggere i messaggi nei canali, mentre gli amministratori, oltre a scrivere e leggere i messaggi, possono creare, modificare ed eliminare il canale, fare broadcast, bannare gli utenti e configurare la durata dei messaggi nei canali.

I requisiti sono soddisfatti ma nel codice c’è la possibilità di ottimizzarlo.

## Scopo

Questo progetto è stato creato per comunicare tra utenti nei canali in una piattaforma.

# Analisi

## Analisi del dominio

L’applicazione serve per comunicare tra utenti registrati nei canali nella piattaforma realizzata. Simili come Telegram, Discord o Whatsapp, solo che si comunicano tra loro solo con i canali e non comunicano solo tra due utenti specifici. Gli utenti che hanno il ruolo amministratore, hanno la possibilità di creare, modificare ed eliminare il canale, ha la possibilità di fare il broadcast (mandare un messaggio a tutti i gruppi che possiede), gestire il ban ed eliminazione dei messaggi.

L’applicazione è sviluppata in HTML, CSS e Javascript.

È necessario acquisire le conoscenze per saper manipolare i dati del database (fatto in JSON) di Firebase.

## Analisi e specifica dei requisiti

|  |  |
| --- | --- |
| **ID: REQ-001** | |
| **Nome** | Creare banca data dati con firebase per utenti e canali |
| **Priorità** | alta |
| **Versione** | 1.0 |
| **Note** |  |

|  |  |
| --- | --- |
| **ID: REQ-002** | |
| **Nome** | Login, autenticazione con nome e password |
| **Priorità** | alta |
| **Versione** | 1.0 |
| **Note** |  |

|  |  |
| --- | --- |
| **ID: REQ-003** | |
| **Nome** | Definire canali per comunicare tra utenti |
| **Priorità** | alta |
| **Versione** | 1.0 |
| **Note** |  |

|  |  |
| --- | --- |
| **ID: REQ-004** | |
| **Nome** | Amministratore che può creare, modificare e eliminare i canali |
| **Priorità** | alta |
| **Versione** | 1.0 |
| **Note** | Quando un amministratore crea, modifica o elimina il canale, deve essere immediato nel frontend si |

|  |  |
| --- | --- |
| **ID: REQ-005** | |
| **Nome** | Amministratore che può cancellare i messaggi |
| **Priorità** | media |
| **Versione** | 1.0 |
| **Note** |  |

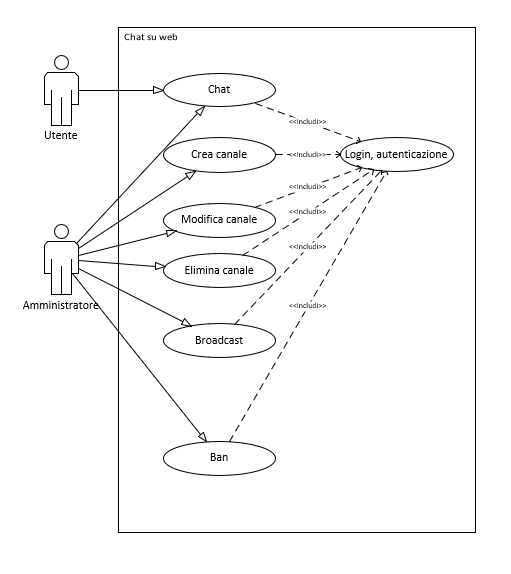
|  |  |
| --- | --- |
| **ID: REQ-006** | |
| **Nome** | Amministratore che può fare broadcast |
| **Priorità** | media |
| **Versione** | 1.0 |
| **Note** |  |

|  |  |
| --- | --- |
| **ID: REQ-007** | |
| **Nome** | Amministratore che può fare ban agli utenti |
| **Priorità** | media |
| **Versione** | 1.0 |
| **Note** |  |

|  |  |
| --- | --- |
| **ID: REQ-008** | |
| **Nome** | Possibilità di configurare la permanenza dei messaggi |
| **Priorità** | bassa |
| **Versione** | 1.0 |
| **Note** |  |

|  |  |
| --- | --- |
| **ID: REQ-009** | |
| **Nome** | Sviluppo web (GUI) |
| **Priorità** | media |
| **Versione** | 1.0 |
| **Note** |  |

## Use case



## Pianificazione

|  |
| --- |
| Figura 1: Diagramma Gannt |

## Analisi dei mezzi

Non è necessario avere un mezzo particolare per la realizzazione del progetto. Si utilizza un text editor e un browser.

### Software

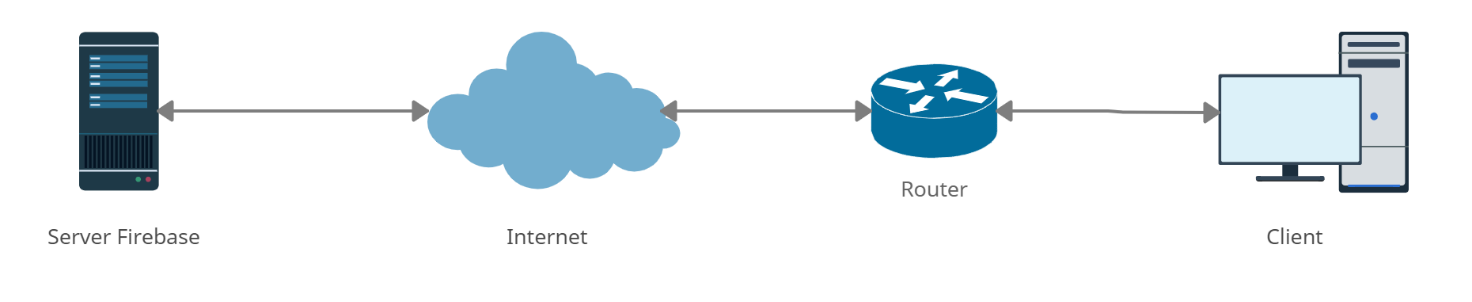
* Visual Studio Code (versione 1.63)
* Firebase (versione 8.2.1)
* Bootstrap (versione 5.1.3)

### Hardware

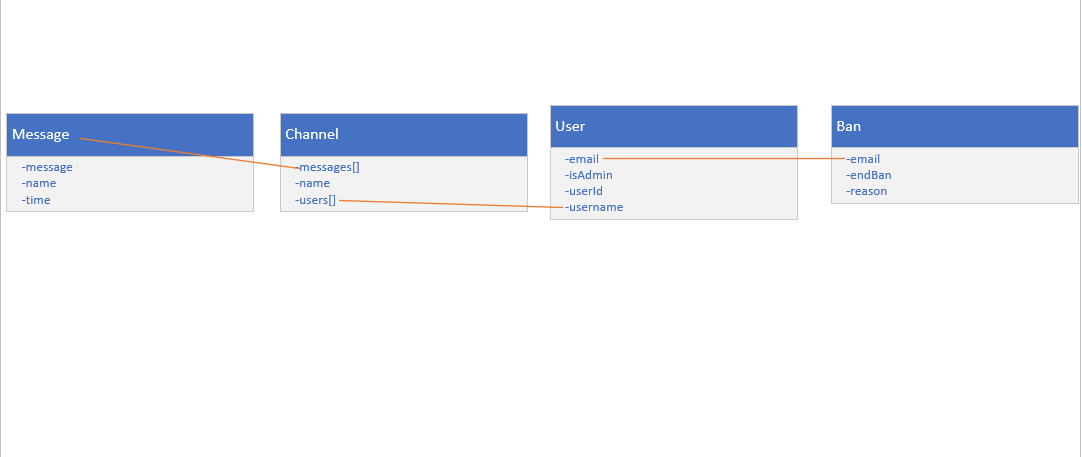
Un dispositivo qualsiasi che abbia una connessione internet e un browser

# Progettazione

## Design dell’architettura del sistema



## Design dei dati e database



Siccome che Firebase utilizza JSON per salvare i dati, è stato creato questo diagramma di classe. Dove ci sono principalmente tre oggetti: *Channel*, *User* e *Ban*. Mentre Message fa parte dell’attributo *messages* nel *Channel*.

## Design delle interfacce

La descrizione delle interfacce per l’utente finale, sia per gli utenti che per gli amministratori. Gli amministratori vengono mostrate le funzionalità come broadcast, e varie impostazioni come creazione di un canale, modificare, eliminare e bannare gli utenti.

Il disegno che viene rappresentato per l’interfaccia è in allegato, con due file *GUI\_ChatFirebase\_1.jpg* e *GUI\_ChatFirebase\_2.jpg*

## Design procedurale

I diagrammi di flusso che vengono rappresentati per questo progetto sono in allegato, il file è *DiagrammiFlussi\_ChatFirebase\_MattiaPasquini.vsdx.*

# Implementazione

## Creazione di un progetto su Firebase

Per creare un progetto su Firebase bisogna andare nel sito <https://console.firebase.google.com/> utilizzando un account Google. Viene chiesto se si vuole abilitare Google Analytics per questo progetto, in questo caso non è necessario perché non viene utilizzato in questo progetto.

Aggiungere un’applicazione web, inserendo il nickname app che viene utilizzato nella Console Firebase per rappresentare l’app.

Creare una banca dati Realtime Database nel sito di Firebase scegliendo la località opportuna che verranno archiviati i dati, in questo caso, *Belgio (europe-west1)*. Poi nelle regole, mettere sia lettura che scrittura come *true* in modo da scrivere e leggere i dati.

[immagine della struttura delle cartelle]

Ogni file html importare le librerie di firebase che verranno utilizzati, per GUI viene utilizzato Bootstrap, importando le librerie.

<script src="https://www.gstatic.com/firebasejs/8.2.1/firebase-app.js"></script>

    <script src="https://www.gstatic.com/firebasejs/8.2.1/firebase-database.js"></script>

    <script src="https://www.gstatic.com/firebasejs/8.2.1/firebase-auth.js"></script>

    <link rel="stylesheet" href="css/bootstrap.min.css">

    <script src="js/bootstrap.min.js"></script>

Ogni script che ha che fare con firebase mettere sempre un costante che contiene i dati della configurazione di firebase, mettere anche l’inizializzazione del firebase e database, che viene già fornito dal sito dopo aver creato il progetto su firebase.

// Your web app's Firebase configuration

const firebaseConfig = {

    apiKey: "…",

    authDomain: "....firebaseapp.com",

    databaseURL: "https://...-default-rtdb.europe-west1.firebasedatabase.app",

    projectId: "...",

    storageBucket: "....appspot.com",

    messagingSenderId: "... ",

    appId: "..."

};

// Initialize Firebase

const app = firebase.initializeApp(firebaseConfig);

// Initialize database

const db = firebase.database();

## Autenticazione

Per l’autenticazione degli utenti del proprio sito, si deve ricordare di abilitare un provider d’accesso per consentire agli utenti di registrarsi.

## Registrazione

Per la registrazione, viene inserito il nome dell’utente, e-mail e password.

C’è la possibilità di impostare che i nomi degli utenti devono essere univoci, ho scelto questa opzione per facilitare quando si legge i dati dal database per aggiungere utente nel gruppo, per eseguire il ban, ecc….

Per farlo si implementa manualmente, possibilmente creando una funzione che viene invocato quando l’utente clicca il bottone per registrarsi.

Se la e-mail e la password rispettano le condizioni, utilizzando il metodo fornito da Firebase, inserire nel database l’utente come userId che contengono le informazioni dell’utente, ossia l’e-mail, isAdmin, userId e username. L’opzione isAdmin è false come default.

Immagine che contiene testo

Descrizione generata automaticamente

## Login

Per login, viene inserito l’e-mail e la password. Viene controllato se l’account esiste, che non sia bannato in quel momento esatto, se non viene rispettato queste condizioni stampa l’errore, altrimenti viene caricata le sezioni per la chat e nascondere la sezione per login.

## Logout

Per eseguire il logout si deve utilizzare il metodo fornito da Firebase che permette di chiudere la sessione dell’utente corrente.

## Inviare un messaggio

Il messaggio, se non è vuoto, viene salvata su database come un nuovo messaggio, salvato come timestamp. E al suo interno c’è il nome dell’utente, il contenuto del messaggio (dall’input che viene scritto dall’utente) e timestamp. Questi dati vengono salivati nel canale corrente nel database.

Immagine che contiene testo

Descrizione generata automaticamente

## Creare un canale

Questa funzione è solo per gli utenti che hanno il ruolo amministratore.

Il canale viene creato avendo il nome del canale, la durata dei messaggi (espresso in ore) e la lista degli utenti che saranno inseriti nel nuovo gruppo (il creatore è già selezionato automaticamente).

Le condizioni per creare un canale sono:

* Nome del canale non dev’essere vuoto e non può essere più lungo di venti caratteri;
* La durata dei messaggi dev’essere maggiore di 0.
* Gli utenti scelti per essere aggiunti (escluso creatore del gruppo) nel gruppo dev’essere minimo uno.

E i dati del canale vengono memorizzati nell’oggetto *Channel*, con nome del canale, gli utenti che fanno parte di quel gruppo e la durata dei messaggi.

Immagine che contiene tavolo

Descrizione generata automaticamente

## Mettere in mostra la lista dei canali e le chat

La parte più critica del progetto è mostrare la lista dei canali e le chat dei canali. Che devono essere aggiornati automaticamente nell’interfaccia da qualsiasi utente (senza ricaricare la pagina) quando vengono subite le aggiunte, modifiche o eliminazione dei dati nel database.

Nell’interfaccia, a sinistra si vede la lista dei canali dove l’utente corrente ne fa parte. E a destra ci sono i “div” dove all’interno ognuno è associata al gruppo e contiene la chat di quel gruppo. E viene mostrata solo il canale selezionato dall’utente (vedi capitolo 4.6), altrimenti non mostra nulla.

function showChannels() {

    //aggiunta canali

    db.ref('channels/').on("child\_added", (snapshot) => {

        var channel = snapshot.val();

//aggiunta utenti nei canali

        db.ref(`channels/${channel.name}/users/`).on("child\_added", (snapshot1) => {

            if (currentUser == snapshot1.val()) {

                //...

                //esegue quando arrivano nuovi messaggi

                db.ref(`channels/${channel.name}/messages/`).on("child\_added", (snapshot2) => {

                    //...

                });

                //esegue quando eliminano i messaggi

                db.ref(`channels/${channel.name}/messages/`).on("child\_removed", (snapshot2) => {

                    //...

                });

            }

        });

        //esegue quando un utente viene rimosso dal gruppo

        db.ref(`channels/${channel.name}/users/`).on("child\_removed", (snapshot1) => {

            if (currentUser == snapshot1.val()) {

                //...

            }

        });

    });

    //ascolto quando elimini il canale

    db.ref('channels/').on("child\_removed", (snapshot) => {

        //...

    });

}

Immaginando lo scenario che questi *listener* verranno letti da qualsiasi client. C’è il *listener* che viene scatenato quando aggiungono i canali. All’interno, siccome che viene creato un canale, c’è un *listener* che viene scatenato quando aggiungono gli utenti in quel gruppo, se l’utente corrente è quello che è stato aggiunto può avere altri due *listener* per stampare il messaggio e sparire il messaggio.

Dopo aver letto tutto il *listene*r per l’aggiunta degli utenti, viene letta anche quando rimuovono gli utenti, se l’utente corrente è quello che viene rimosso dal canale, fa sparire il gruppo dalla lista e chat.

Mentre l’ultimo *listener* che viene scatenato quando un gruppo è eliminato, tutti gli utenti non dovrebbero vedere più quel gruppo dalla lista e la chat.

## Cambio canale

Creare una funzione che viene usato quando clicca un canale nella lista dei canali.

E mettere visibile la chat desiderata e nascondere la chat del canale precedente.

## Eliminare un canale

Questa funzione è solo per gli utenti che hanno il ruolo amministratore.

Una funzione che viene invocato quando si vuole eliminare il canale corrente.

Si deve togliere i dati di quel canale nel database.

## Modifica canale

Questa funzione è solo per gli utenti che hanno il ruolo amministratore.

Per modificare un canale si tratta di aggiungere gli utenti, rimuovere gli utenti e modificare la durata dei messaggi nel gruppo selezionato.

## Aggiungere gli utenti nel gruppo

Nell’interfaccia viene mostrato la lista degli utenti che non fanno parte in quel gruppo. Dopo aver selezionato gli utenti desiderati, c’è un bottone che permette di confermare l’aggiunta degli utenti e aggiunge gli utenti nel gruppo nel database.

## Eliminare l’utente nel gruppo

Per eliminare gli utenti, nell’interfaccia dovrà mostrare gli utenti che fanno parte di quel gruppo e a destra c’è un link per ogni utente, tranne utente corrente, che permette di cliccare se si desidera che quell’utente viene rimosso dal gruppo.

Una volta cliccato, l’array degli utenti in un gruppo nel database, va tolto l’utente selezionato.

## Modificare la durata del messaggio in un gruppo

Le condizioni che devono essere rispettate per poter effettuare la modifica sulla durata dei messaggi del canale sono:

* Se non viene scritto la durata
* Se la durata è maggiore di 0

## Eliminazione dei messaggi

Questa funzione è solo per gli utenti che hanno il ruolo amministratore.

Nel front-end, ogni messaggio ha il suo id, che è associato al timestamp.

Implementare una funzione dove i parametri sono il gruppo e il timestamp e poi rimuovere il messaggio in un gruppo nel database.

## Broadcast

Questa funzione è solo per gli utenti che hanno il ruolo amministratore.

Il messaggio, se non è vuoto, viene salvato in tutti i gruppi che l’utente corrente appartiene nel database.

Ogni gruppo va salvata il messaggio come descritto nel capitolo 4.3.

## Ban

Questa funzione è solo per gli utenti che hanno il ruolo amministratore.

Nell’interfaccia, c’è un input come select, dove si ha la possibilità di scegliere un utente. L’altro input ci dev’essere textarea che viene scritto il motivo del ban, e l’ultimo input, il periodo (espresso in ore) del ban.

Se l’utente non è selezionato, se il motivo non è stata descritta, oppure se il valore del periodo non è scritta o se è minore di uno, dovrebbe stampare l’errore.

Se l’utente selezionato è già bannato, stampare l’errore che è già bannato.

Se le condizioni del ban vengono rispettate, aggiungere i dati nel database come l’e-mail che viene associata dall’utente selezionato, la data della fine del ban espresso in timestamp, e il motivo.

Immagine che contiene testo

Descrizione generata automaticamente

## Eliminazione dei vecchi messaggi

Questa funzione è solo per gli utenti che hanno il ruolo amministratore.

C’è un bottone che fa la “pulizia” di tutti i messaggi che sono durati più della durata del messaggio impostato.

Dal database, verificare tutti i messaggi nel gruppo corrente se la durata è maggiore della sottrazione del tempo corrente e tempo in cui viene inviato il messaggio. Se questa condizione è vera allora eliminare quel messaggio dal database altrimenti non fa nulla.

# Test

## Protocollo di test

Definire in modo accurato tutti i test che devono essere realizzati per garantire l’adempimento delle richieste formulate nei requisiti. I test fungono da garanzia di qualità del prodotto. Ogni test deve essere ripetibile alle stesse condizioni.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-001  REQ-002 | **Nome:** | Login, autenticazione con nome e password |
| **Descrizione:** | Controllare che gli utenti registrano e che i dati vengono salvati correttamente nel database. | | |
| **Prerequisiti:** | - | | |
| **Procedura:** | 1. Andare nella pagina di registrazione 2. Mettere il nome utente, e-mail e password 3. Cliccare il tasto per registrarsi | | |
| **Risultati attesi:** | Nuovo utente viene inserito nel database correttamente | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-002  REQ-002 | **Nome:** | Login, autenticazione con nome e password |
| **Descrizione:** | Controllare che l’utente effettua login con le credenziali corrette. | | |
| **Prerequisiti:** | - | | |
| **Procedura:** | 1. Andare nella pagina per login 2. Mettere le credenziali corrette 3. Cliccare il bottone per effettuare login | | |
| **Risultati attesi:** | Mostrare la sezione principale della piattaforma e nascondere la sezione del login | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-003  REQ-002 | **Nome:** | Login, autenticazione con nome e password |
| **Descrizione:** | Controllare che l’utente effettua login con e-mail non registrato nella piattaforma. | | |
| **Prerequisiti:** | - | | |
| **Procedura:** | 1. Andare nella pagina per login 2. Mettere l’e-mail non registrato nella piattaforma 3. Cliccare il bottone per effettuare login | | |
| **Risultati attesi:** | Stampa l’errore “E-mail or password are not valid” sull’interfaccia. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-004  REQ-002 | **Nome:** | Login, autenticazione con nome e password |
| **Descrizione:** | Controllare che l’utente effettua login con password non corretta. | | |
| **Prerequisiti:** | - | | |
| **Procedura:** | 1. Andare nella pagina per login 2. Mettere l’e-mail non registrato nella piattaforma 3. Cliccare il bottone per effettuare login | | |
| **Risultati attesi:** | Stampa l’errore “E-mail or password are not valid” sull’interfaccia. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-005  REQ-004 | **Nome:** | Amministratore che può creare, modificare ed eliminare i canali |
| **Descrizione:** | Amministratore che crea un canale | | |
| **Prerequisiti:** | - | | |
| **Procedura:** | 1. Aprire model per creare il canale 2. Mettere il nome del canale 3. Selezionare uno o più utenti 4. Mettere la durata 5. Crea il canale | | |
| **Risultati attesi:** | Aggiunge i dati del gruppo correttamente nel database e nell’interfaccia viene aggiunto il gruppo appena creato nella lista. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-006  REQ-004 | **Nome:** | Amministratore che può creare, modificare e eliminare i canali |
| **Descrizione:** | Amministratore che aggiunge un utente in un canale | | |
| **Prerequisiti:** | TC-005 | | |
| **Procedura:** | 1. Aprire model che mostra le informazioni del canale 2. Cliccare il bottone per aprire un altro model che mostra la lista degli utenti per inserire nel gruppo 3. Seleziona uno o più utenti 4. Cliccare il bottone per inserire gli utenti nel gruppo | | |
| **Risultati attesi:** | Aggiorna la lista degli utenti del gruppo correttamente nel database e nell’interfaccia degli utenti appena inseriti nel gruppo viene aggiunto il gruppo nella lista. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-007  REQ-004 | **Nome:** | Amministratore che può creare, modificare e eliminare i canali |
| **Descrizione:** | Amministratore che rimuove un utente in un canale | | |
| **Prerequisiti:** | TC-005 | | |
| **Procedura:** | 1. Aprire model che mostra le informazioni del canale 2. Cliccare un tasto remove per rimuovere un utente | | |
| **Risultati attesi:** | Aggiorna la lista degli utenti del gruppo correttamente (utente rimosso) nel database e nell’interfaccia dell’utente appena rimosso, non viene più visualizzato il gruppo nella lista. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-008  REQ-004 | **Nome:** | Amministratore che può creare, modificare e eliminare i canali |
| **Descrizione:** | Amministratore che elimina il canale | | |
| **Prerequisiti:** | TC-005 | | |
| **Procedura:** | 1. Aprire model che mostra le informazioni del canale 2. Cliccare il bottone dove elimini il canale | | |
| **Risultati attesi:** | Elimina il gruppo eliminato nel database e l’interfaccia di tutti gli utenti che appartenevano in quel gruppo non viene più visualizzato quel gruppo nella lista. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-009  REQ-005 | **Nome:** | Amministratore che può cancellare i messaggi |
| **Descrizione:** | Amministratore che elimina un messaggio | | |
| **Prerequisiti:** | - | | |
| **Procedura:** | 1. Cliccare il tasto per eliminare il messaggio | | |
| **Risultati attesi:** | Elimina il messaggio selezionato nel database e nell’interfaccia non viene visualizzato il messaggio eliminato. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-010  REQ-006 | **Nome:** | Amministratore che può fare broadcast |
| **Descrizione:** | Amministratore che fa broadcast | | |
| **Prerequisiti:** | - | | |
| **Procedura:** | 1. Aprire il model di broadcast 2. Scrivere il testo 3. Cliccare invio | | |
| **Risultati attesi:** | Inserisci quel messaggio a tutti i gruppi che l’utente corrente ne fa parte nel database e nell’interfaccia viene visualizzata il messaggio inviato nei gruppi. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-011  REQ-007 | **Nome:** | Amministratore che può fare ban agli utenti |
| **Descrizione:** | Amministratore che banna un utente correttamente | | |
| **Prerequisiti:** | - | | |
| **Procedura:** | 1. Aprire il model di gestione del ban 2. Selezionare un utente 3. Scrivere il motivo del ban 4. Scrivere la durata del ban che sia un numero positivo | | |
| **Risultati attesi:** | Inserisce i dati corretti nella lista del ban nel database. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-012  REQ-008 | **Nome:** | Possibilità di configurare la permanenza dei messaggi |
| **Descrizione:** | Amministratore che configura la permanenza dei messaggi | | |
| **Prerequisiti:** | - | | |
| **Procedura:** | 1. Aprire il model di informazioni del canale 2. Modificare la durata dei messaggi 3. Cliccare il bottone per salvare la modifica | | |
| **Risultati attesi:** | Stampa sul model “The duration message has changed!” e aggiorna nel database la durata del messaggio del gruppo corrente. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-013  REQ-008 | **Nome:** | Possibilità di configurare la permanenza dei messaggi |
| **Descrizione:** | Amministratore che elimina i messaggi vecchi | | |
| **Prerequisiti:** | TC-013 | | |
| **Procedura:** | 1. Aprire il model di informazioni del canale 2. Cliccare il bottone per eliminare i messaggi vecchi | | |
| **Risultati attesi:** | Stampa sul model “Old messages have been deleted” ed elimina i messaggi vecchi nel gruppo corrente | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-014  REQ-002 | **Nome:** | Login, autenticazione con nome e password |
| **Descrizione:** | Controllare che l’utente bannato effettua login. | | |
| **Prerequisiti:** | TC-12 | | |
| **Procedura:** | 1. Andare nella pagina per login 2. Mettere l’e-mail non registrato nella piattaforma 3. Cliccare il tasto per effettuare login | | |
| **Risultati attesi:** | Stampa l’errore “You are banned until <data>” sull’interfaccia. | | |

## Risultati test

|  |  |
| --- | --- |
| Test Case: | TC-001 |
| Test riuscito: | SI |
| Errore |  |

|  |  |
| --- | --- |
| Test Case: | TC-002 |
| Test riuscito: | SI |
| Errore |  |

|  |  |
| --- | --- |
| Test Case: | TC-003 |
| Test riuscito: | SI |
| Errore |  |

|  |  |
| --- | --- |
| Test Case: | TC-004 |
| Test riuscito: | SI |
| Errore |  |

|  |  |
| --- | --- |
| Test Case: | TC-005 |
| Test riuscito: | SI |
| Errore |  |

|  |  |
| --- | --- |
| Test Case: | TC-006 |
| Test riuscito: | SI |
| Errore |  |

|  |  |
| --- | --- |
| Test Case: | TC-007 |
| Test riuscito: | SI |
| Errore |  |

|  |  |
| --- | --- |
| Test Case: | TC-008 |
| Test riuscito: | SI |
| Errore |  |

|  |  |
| --- | --- |
| Test Case: | TC-009 |
| Test riuscito: | SI |
| Errore |  |

|  |  |
| --- | --- |
| Test Case: | TC-010 |
| Test riuscito: | SI |
| Errore |  |

|  |  |
| --- | --- |
| Test Case: | TC-011 |
| Test riuscito: | - |
| Errore |  |

|  |  |
| --- | --- |
| Test Case: | TC-012 |
| Test riuscito: | - |
| Errore |  |

|  |  |
| --- | --- |
| Test Case: | TC-013 |
| Test riuscito: | - |
| Errore |  |

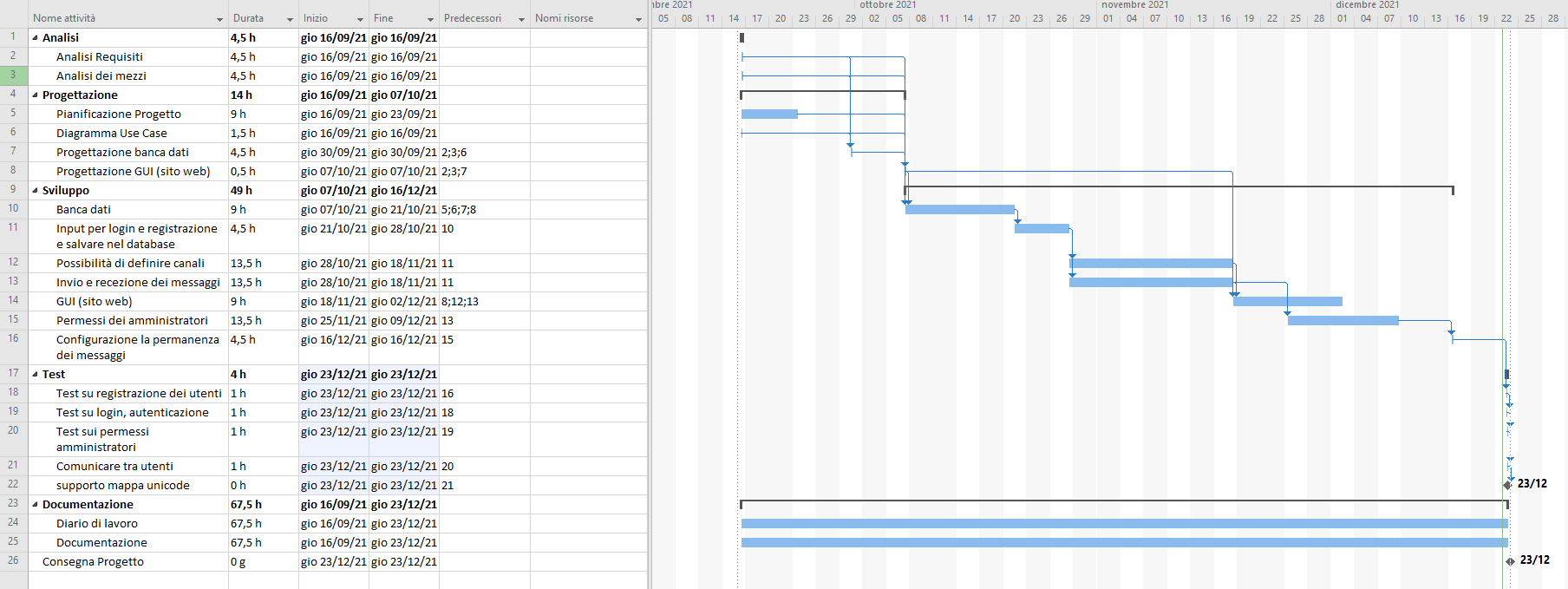
|  |  |
| --- | --- |
| Test Case: | TC-014 |
| Test riuscito: | - |
| Errore |  |

|  |  |
| --- | --- |
| Test Case: | TC-015 |
| Test riuscito: | - |
| Errore |  |

## Mancanze/limitazioni conosciute

Descrizione con motivazione di eventuali elementi mancanti o non completamente implementati, al di fuori dei test case. Non devono essere riportati gli errori e i problemi riscontrati e poi risolti durante il progetto.

# Consuntivo



La figura che viene mostrato è la pianificazione consuntivo, l’andamento del progetto. C’è una grande differenza rispetto alla pianificazione iniziale, la parte dello sviluppo è durato più del previsto mentre il test è durato meno del previsto.

# Conclusioni

Non penso che con questo progetto cambierà il mondo, perché esistono già molte applicazioni simili che sono molto utilizzati e hanno molte più funzionalità utili. Ma potrebbe essere utile per *chattare* in una azienda tra i dipendenti e un amministratore che gestisce i gruppi.

## Sviluppi futuri

C’è la possibilità di aggiungere le ulteriori funzioni per questa applicazione, per esempio mandare le foto, video, GIF e l’audio. Possibilità di segnalare qualche utente. Possibilità di rispondere un messaggio specifico. E migliorare l’interfaccia grafica.

## Considerazioni personali

In questo progetto ho imparato tanto, non ho mai progettato e lavorato un progetto a lungo termine e un po’ complesso come questo progetto. All’inizio sembrava difficile implementare ma col passare del tempo sono riuscito a capire il meccanismo del servizio Firebase che ci offre per sfruttarlo al meglio per questa applicazione.

Nonostante il meglio che avrei potuto fare, sono comunque soddisfatto del mio lavoro che ho fatto per questo progetto soprattutto perché sono riuscito a rispettare tutti i requisiti.

Ho dovuto anche imparare a come gestire i dati nel json, ossia come leggerli e come modificarli. Ho imparato anche a come utilizzare Bootstrap che è utile e comodo per l’interfaccia grafica.

Concludo che con questo progetto mi ha fatto capire quali sono le mie capacità, soprattutto i miei punti deboli che posso migliorarli.

# Bibliografia

## Bibliografia per articoli di riviste:

1. Cognome e nome (o iniziali) dell’autore o degli autori, o nome dell’organizzazione,
2. Titolo dell’articolo (tra virgolette),
3. Titolo della rivista (in italico),
4. Anno e numero
5. Pagina iniziale dell’articolo,

## Bibliografia per libri

1. Cognome e nome (o iniziali) dell’autore o degli autori, o nome dell’organizzazione,
2. Titolo del libro (in italico),
3. ev. Numero di edizione,
4. Nome dell’editore,
5. Anno di pubblicazione,
6. ISBN.

## Sitografia

1. URL del sito (se troppo lungo solo dominio, evt completo nel diario),
2. Eventuale titolo della pagina (in italico),
3. Data di consultazione (GG-MM-AAAA).

* <https://firebase.google.com/>, 30-09-2021
* <https://firebase.google.com/docs/web/setup?authuser=0>, *Aggiungere Firebase al tuo progetto*, 30-09-2021
* <https://www.cometchat.com/tutorials/how-to-build-a-chat-app-with-firebase>, 21-10-2021

**Esempio:**

* http://standards.ieee.org/guides/style/section7.html, *IEEE Standards Style Manual*, 07-06-2008.

# Allegati

Elenco degli allegati, esempio:

* Diari di lavoro
* Codici sorgente/documentazione macchine virtuali
* Istruzioni di installazione del prodotto (con credenziali di accesso) e/o di eventuali prodotti terzi
* Documentazione di prodotti di terzi
* Eventuali guide utente / Manuali di utilizzo
* Mandato e/o Qdc
* Prodotto
* …