Chat con Firebase

1 Introduzione 4

1.1 Informazioni sul progetto 4

1.2 Abstract 4

1.3 Scopo 4

2 Analisi 4

2.1 Analisi del dominio 4

2.2 Analisi e specifica dei requisiti 5

2.3 Use case 6

2.4 Pianificazione 7

2.5 Analisi dei mezzi 8

2.5.1 Software 8

2.5.2 Hardware 8

3 Progettazione 8

3.1 Design dell’architettura del sistema 8

3.2 Design dei dati e database 8

3.3 Design delle interfacce 9

3.4 Design procedurale 10

4 Implementazione 10

4.1 Creazione di un progetto con Firebase 10

4.2 Creazione del progetto 11

4.3 Index.html 11

4.4 Chat.html 11

4.5 Index.js 11

4.5.1 Controllo dei nuovi messaggi 12

4.5.2 registerNewUser 12

4.5.3 loginUser 12

4.5.4 logoutUser 12

4.5.5 sendMessage 12

4.5.6 viewAllUsers 13

4.5.7 addToChannel 13

4.5.8 createChannel 13

4.5.9 channelExists 13

4.5.10 deleteOldMessages 13

4.5.11 loadChannels 13

4.5.12 getChatBox 13

4.5.13 changeChannel 14

4.5.14 showNickname 14

4.5.15 reloadMessages 14

4.5.16 deleteChannel 14

4.5.17 openModifyChannel 14

4.5.18 addToBan 14

4.5.19 banUser 14

4.5.20 loadAdminOption 15

4.5.21 resizeMessagesDiv 15

4.5.22 onEnterSendMessage 15

4.6 Sidebars.js 15

4.6.1 openNav 15

4.6.2 closeNav 15

4.7 Bootstrap.bundle.min.js 15

4.8 Bootstrap.css 15

4.9 Sidebars.css 15

5 Test 16

5.1 Protocollo di test 16

5.2 Risultati test 18

5.3 Mancanze/limitazioni conosciute 19

6 Consuntivo 20

7 Conclusioni 21

7.1 Sviluppi futuri 21

7.2 Considerazioni personali 21

7.3 Sitografia 21

8 Allegati 21

# Introduzione

## Informazioni sul progetto

Sono un’allieva del terzo anno di informatica alla Scuola d’Arti e Mestieri a Trevano. Ho realizzato questo progetto durante il modulo 306 con la supervisione del docente Geo Petrini. Il progetto è iniziato il 09.09.2021 è si è concluso il 23.12.2021.

## Abstract

Lo scopo di questo progetto è quello di realizzare una chat multiutente gestita per canali. L’applicazione permette a qualsiasi utente di registrarsi tramite un indirizzo e-mail. Sarà poi possibile creare nuovi canali specificando dopo quanto tempo i messaggi saranno cancellati ed eventualmente bannare gli utenti. Per gli utenti con ruolo admin è anche possibile inviare messaggi in broadcast su tutti i canali.

Nei requisiti non c’erano limitazioni per il linguaggio da utilizzare quindi ho deciso di sviluppare un’applicazione web utilizzando HTML, Javascript e CSS e l’ausilio di bootstrap. Per le interazioni con il database di firebase ho utilizzato la documentazione fornita sul sito ufficiale e altre guide trovate online.

I requisiti del progetto sono stati soddisfatti però avrei potuto gestire meglio la struttura del database ad oggetti che non avevo mai utilizzato e non l’ho saputa gestire al meglio. Non sono inoltre riuscita a gestire al meglio gli accessi al database.

## Scopo

Il progetto è pensato per utenti che vogliono chattare fra di loro tramite canali.

# Analisi

## Analisi del dominio

L’applicazione permette a chiunque sia registrato nel sito di comunicare tramite canali. Il funzionamento è simile ad’ altri prodotti già esistenti come WhatsApp o Telegram ma senza la possibilità di comunicare direttamente fra due utenti. L’applicazione è stata utilizzata utilizzando principalmente tre linguaggi: HTML, JavaScript e CSS per la parte grafica ed’ è necessario acquisire le conoscenze per manipolare i dati nel database di firebase.

## Analisi e specifica dei requisiti

|  |  |
| --- | --- |
| ID: REQ-01 | |
| Nome | Installazione server Firebase |
| Priorità | alta |
| Versione | 1.0 |
| Sotto requisiti | |
| 001 | Si necessita una maschera di login |
| 002 | Si dovranno poter immettere nuovi allievi |
| 003 | Dovrà essere possibile la ricerca di allievi |

|  |  |
| --- | --- |
| ID: REQ-02 | |
| Nome | Autentificazione degli utenti |
| Priorità | alta |
| Versione | 1.0 |
| Sotto requisiti | |
| 001 | È necessario salvare nome e password |
| 002 | Le password devono essere salvate in modo sicuro sui database di Firebase |

|  |  |
| --- | --- |
| ID: REQ-03 | |
| Nome | Gli amministratori devono poter eseguire determinate azioni in più rispetto agli utenti normali |
| Priorità | 1 |
| Versione | media |
| Sotto requisiti | |
| 001 | Creare canali |
| 002 | Modificare canali |
| 003 | Eliminare canali |
| 004 | Inviare messaggi in broadcast |
| 005 | Possibilità di bannare utenti |

|  |  |
| --- | --- |
| ID: REQ-04 | |
| Nome | Configurazione dei messaggi |
| Priorità | bassa |
| Versione | 1.0 |
| Sotto requisiti | |
| 001 | L’applicazione deve gestire la cancellazione dei messaggi dopo un tempo specificato. |

## Use case

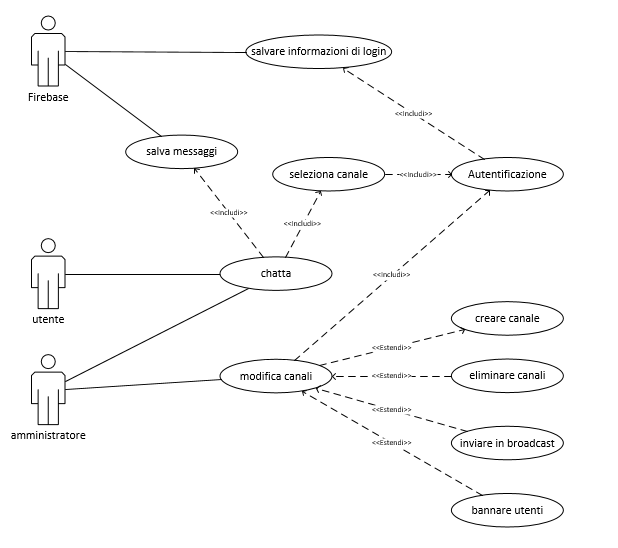


Figura - Use case

## Pianificazione

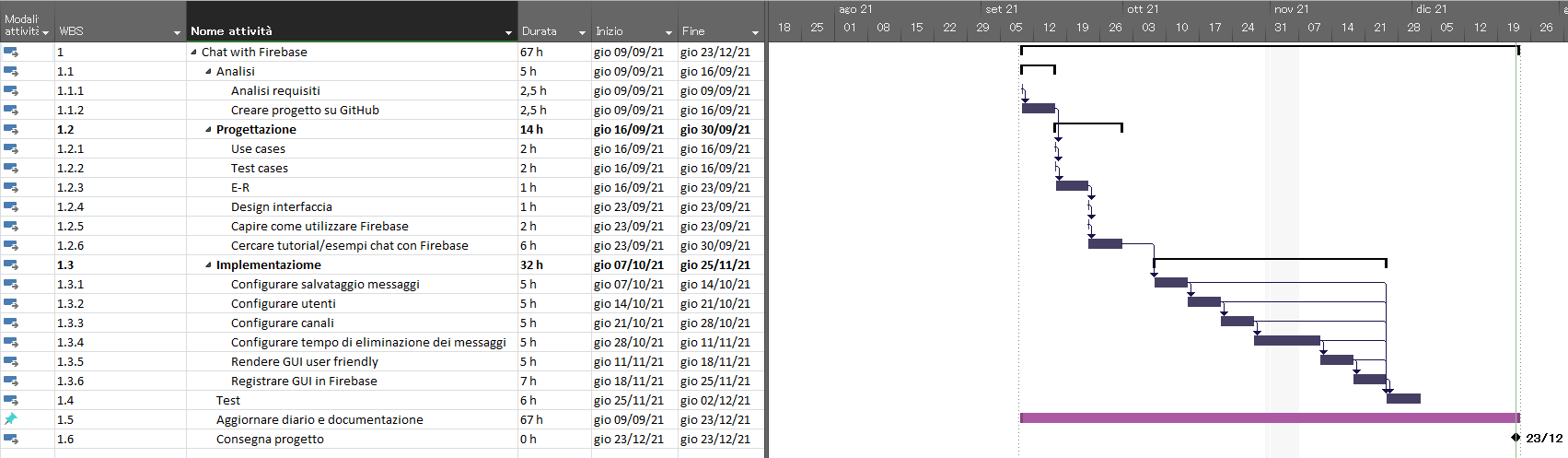


Figura - Gantt iniziale

## Analisi dei mezzi

Per realizzare questo progetto ho utilizzato VS Code come editor.

### Software

* Visual Studio Code (v 1.62)
* Firebase (v 8.2.1)
* Bootstrap (v 5)

### Hardware

L’applicazione è responsive e quindi utilizzabile su qualsiasi dispositivo dotato di un browser e una connessione internet, non è quindi necessario un hardware specifico.

# Progettazione

## Design dell’architettura del sistema

Figura - Schema di rete

Data la necessità di connettersi al database di Firebase per lavorare sono dovuta uscire su internet tramite il proxy della scuola.

## Design dei dati e database

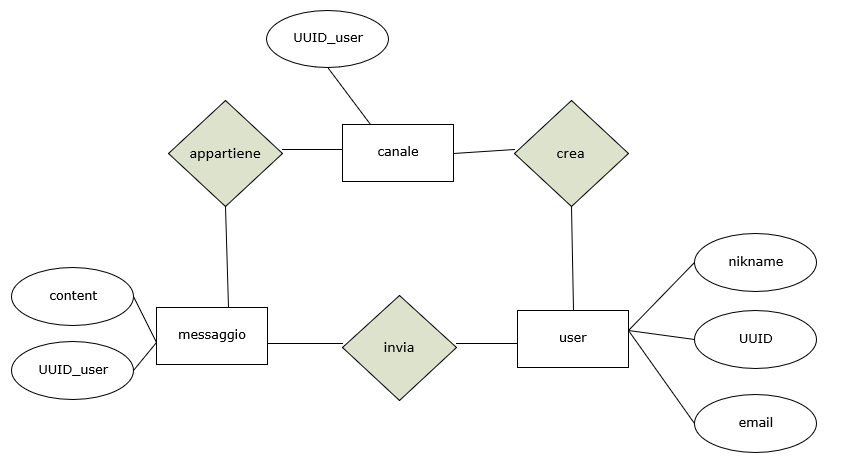


Figura - Schema E-R per il database

Questa era l’idea iniziale prima di essermi documentata scoprendo che firebase utilizza un database ad oggetti e non relazionale. Questa è quindi la versione definitiva in base al funzionamento di firebase.

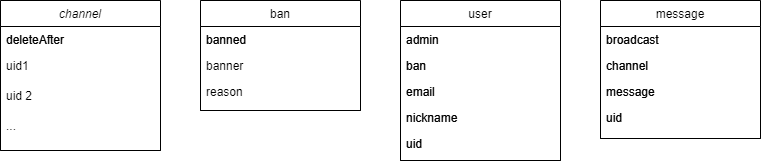


Figura - Diagramma delle classi per il database

## Design delle interfacce

Sulla sinistra ho realizzato la mia progettazione iniziale delle interfacce iniziale nel seguente modo. Invece sulla destra sono gli screen del design finale della pagina web.

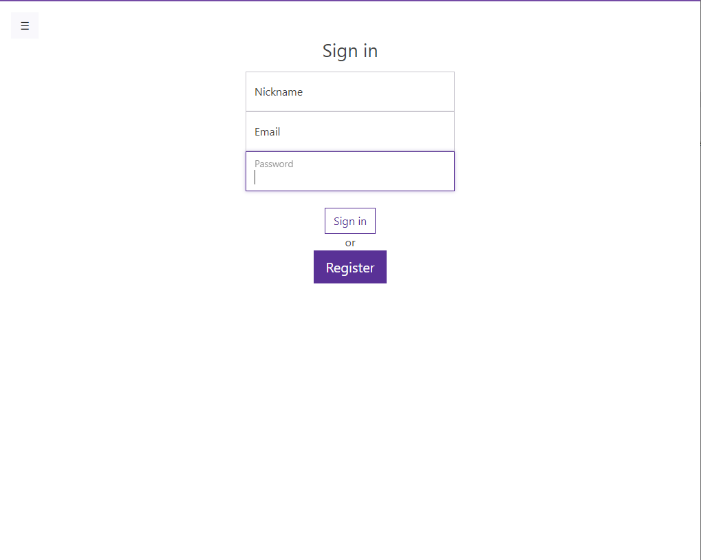


Figura - GUI login finale

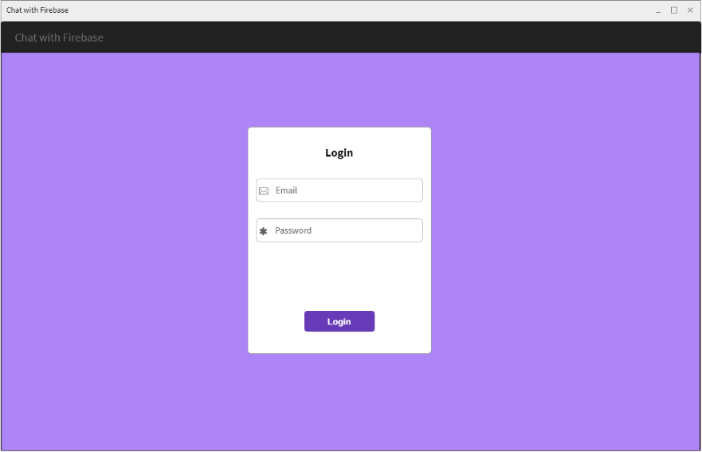


Figura - Progettazione GUI login iniziale

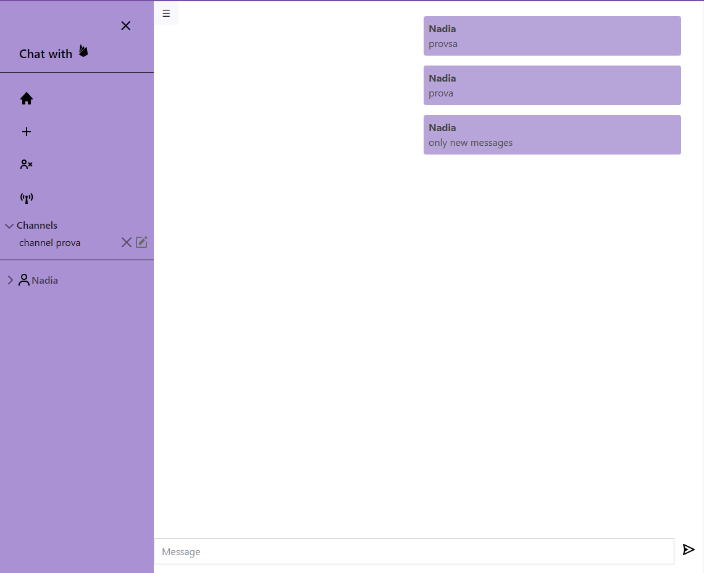


Figura - GUI chat finale



Figura - Progettazione GUI chat iniziale

## Design procedurale

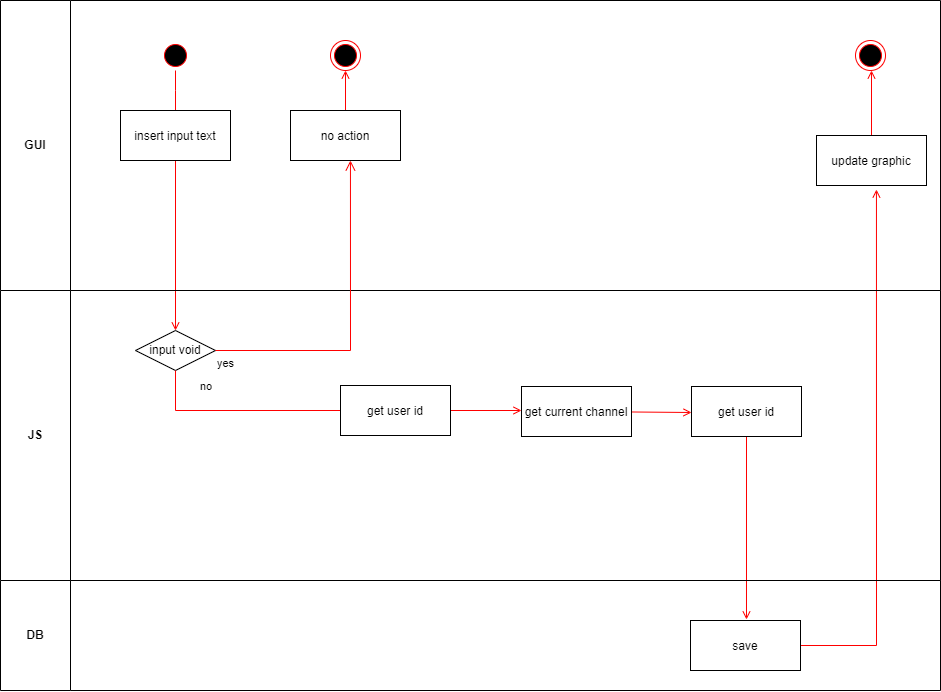
Questo è il diagramma di flusso per l’invio dei messaggi della chat che può essere generalizzato per tutte le chiamate al database di firebase. La comunicazione con Firebase viene gestita tramite un WebSocket.

Figura - Schema diagramma procedurale

# Implementazione

## Creazione di un progetto con Firebase

Per creare un progetto con Firebase aprire il seguente link: <https://console.firebase.google.com/> e con un account Google è possibile accedere all’interfaccia.

Per poter iniziare a programmare l’applicazione è necessario registrarla ricevendo un id e le chiavi di accesso da inserire nei file per connettersi al database.

Serve anche abilitare l’autenticazione tramite indirizzo e-mail e inizializzare il database Realtime di Firebase.

## Creazione del progetto

Figura - Albero delle cartelle dell'applicazione

## Index.html

Figura - Head del file index.html

Nell’head del file ci sono tutti i riferimenti per gli script di JavaScript e CSS. Per firebase ho importato le librerie per l’app, il database e l’autenticazione.

Il file index contiene anche la sidebar laterale con il menu e il form di login o registrazione per accedere alla chat.

## Chat.html

Il file contiene gli elementi per gestire la chat. Nella sidebar laterale c’è la possibilità di creare o cancellare i canali e se si hanno i permessi di amministratore inviare un messaggio in broadcast. Al caricamento della pagina viene sempre mostrata la chat dell’ultimo canale della lista.

## Immagine che contiene testo Descrizione generata automaticamenteIndex.js

Figura - Configurazioni di Firebase

Queste sono le informazioni che permettono all’applicazione di comunicare con firebase. Queste istruzioni si trovano nelle impostazioni di ogni singola applicazione che comunica con firebase.

### Controllo dei nuovi messaggi

Questa chiamata al database utilizza il WebSocket di Firebase che notifica all’applicazione quando è stato aggiunto un nuovo elemento ai messaggi. L’evento richiamato è on child\_added. Questo permette di aggiornare la chat in tempo reale. Viene poi fatto un ciclo per ogni messaggio per capire se è un messaggio da visualizzare perché fa parte del canale corrente ed ‘è quindi da mostrare o no.

### registerNewUser

Immagine che contiene testo

Descrizione generata automaticamenteIl metodo registerNewUser legge il valore degli input del file index.html con nickname, e-mail e password del nuovo utente. Se la password rispetta le condizioni e non esiste un altro account con lo stesso indirizzo e-mail al database sotto gli utenti viene creato un nuovo oggetto che contiene le informazioni dell’utente. L’opzione admin è di default a false e deve essere impostata manualmente.

Figura - Struttura DB per gli utenti

### loginUser

Il metodo loginUser legge il valore degli input del file index.html con email e password. In seguito controlla che l’account esista e che l’utente non sia in quel momento bannato e mostra un messaggio di errore che arriva direttamente da Firebase. Se invece l’utente esiste e può accedere viene caricata la pagina chat.html.

### logoutUser

Tramite il metodo fornito da Firebase viene chiusa la sessione dell’utente corrente.

### sendMessage

Immagine che contiene testo

Descrizione generata automaticamenteIl metodo sendMessage salva nel database un nuovo messaggio che ha come valore il timestamp salvato nel formato UNIX time.

Figura - Struttura DB per i messaggi

L’opzione broadcast specifica se il messaggio deve essere visibile su tutti i canali ed’è quindi stato inviato da un account con un ruolo admin. L’uid è un codice univoco per ogni account per poter risalire a chi ha inviato il messaggio.

### viewAllUsers

Il metodo viewAllUsers permette di visualizzare nel dropdown nel modal all’interno della pagina chat.html per poter aggiungere gli utenti quando si sta creando un nuovo canale. Per farlo vado a leggere i dati dal database su tutti gli utenti esistenti.

### addToChannel

Il metodo addToChannel viene scatenato ogni volta che si clicca su un utente nel dropdown per la creazione di un nuovo canale per poi mostrare sopra al dropdown la lista delle persone che sono state aggiunte.

### createChannel

Immagine che contiene testo

Descrizione generata automaticamenteIl metodo createChannel legge il valore degli input all’interno del modal per la creazione del canale. Il database salverà il nome del canale e l’uid di tutti gli utenti che ne fanno parte, facendo attenzione a comprendere anche l’utente che sta creando il canale. Viene anche salvato il tempo dopo il quale i vecchi messaggi devono essere cancellati (rappresentato in ore).

Figura - Struttura DB per i canali

### channelExists

Il metodo channelExists controlla se nel database esiste già un altro canale con lo stesso nome, se è così mostra nel modal un messaggio che avvisa l’utente di modificare il nome.

### deleteOldMessages

Il metodo deleteOldMessages ha come parametro channel che determina da quale canale cancellare i vecchi messaggi secondo il tempo di cancellazione dei messaggi di quello specifico canale. Per saperlo utilizzo la chiave di ogni messaggio che corrisponde al tempo in UNIX time espresso in millisecondi di quando il messaggio è stato inviato.

### loadChannels

Il metodo loadChannels viene richiamato al caricamento della pagina chat.html e mostra nella sidebar laterale tutti i canali di cui l’utente corrente fa parte.

### getChatBox

Il metodo getChatBox ha come parametri nickname, messaggio, floatRight e broadcast. Serve per creare una nuova riga per i messaggi della chat. Il parametro floatRight viene impostata a true se il messaggio da mostrare è stato inviato dall’utente corrente per essere allineato a destra. Il parametro broadcast se impostato a true aggiunge dopo il nickname una stringa per avvisare che il messaggio è stato inviato in broadcast.

### changeChannel

Il metodo changeChannel modifica il valore della variabile globale currentChannel e ricarica i messaggi per aggiornarli a quelli del canale corrente richiamando il metodo reloadMessages.

### showNickname

Il metodo showNickname mostra nella sidebar laterale il nickname dell’utente corrente. Per fare questo è necessario cercare nel database fra gli utenti registrati quale ha l’uid uguale a quello dell’utente della sessione corrente.

### reloadMessages

Il metodo reloadMessages ricarica i messaggi del canale corrente secondo la variabile globale currentChannel. Per farlo legge tutti i messaggi presenti nel database mostrando a schermo solo quelli che corrispondono al canale corrente e utilizza il metodo getChatBox.

### deleteChannel

Il metodo deleteChannel elimina il canale specificato dal parametro channel dal database di firebase e richiama il metodo loadChannels per rimuovere dalla lista il canale cancellato.

Vengono anche cancellati tutti i messaggi contenuti in quel canale.

### openModifyChannel

Il metodo openModifyChannel quando viene cliccato il tasto di modifica di un canale mostra il modal per la modifica del canale.

### addToBan

Il metodo addToBan viene scatenato ogni volta che si clicca su un utente nel dropdown per il ban di un utente. È possibile bannare un solo utente alla volta. Viene poi mostrato a schermo l’utente selezionato

### Immagine che contiene testo Descrizione generata automaticamentebanUser

Figura - Struttura DB per i ban

Il metodo banUser legge gli input del modal dalla pagina chat.html. L’utente bannato, l’utente che ha bannato, il motivo del ban e la durata vengono salvati nel database.

### loadAdminOption

Il metodo loadAdminOption viene richiamato al caricamento della pagina chat.html e va a leggere dal database se l’utente corrente è admin. Se lo è mostra nella sidebar la possibilità di inviare messaggi in broadcast.

### resizeMessagesDiv

Il metodo resizeMessagesDiv viene richiamato ogni volta che viene ridimensionata la finestra per poter adattare la lunghezza del div della chat rispetto alla dimensione della pagina chat.html.

### onEnterSendMessage

Il metodo onEnterSendMessage viene richiamato al caricamento della pagina chat.html e permette di inviare messaggi in chat premendo il tasto Enter.

## Sidebars.js

Questo file contiene i metodi per la gestione dell’apertura e della chiusura della sidebar.

### openNav

Il metodo gestisce l’animazione all’apertura della sidebar specificando la larghezza che la sidebar deve avere e che il resto della pagina deve avere un margine a sinistra di quella larghezza. Il tempo dell’animazione viene gestito dal CSS.

### closeNav

Il metodo gestisce l’animazione alla chiusura della sidebar specificando la larghezza della sidebar a 0 e il resto della pagina occupa tutto lo spazio disponibile. Il tempo dell’animazione viene gestito dal CSS.

## Bootstrap.bundle.min.js

È la libreria di bootstrap per la gestione degli elementi tramite JavaScript.

## Bootstrap.css

È la libreria di bootstrap per la gestione del CSS. Questa libreria mi è servita nelle pagine html per la gestione della grafica e per realizzare un sito responsive.

## Sidebars.css

Questo file serve per gestire l’animazione e lo stile degli elementi contenuti nella sidebar.

# 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-01  REQ-01 | Nome: | Installazione server Firebase |
| Descrizione: | Controllare che il server sia online e che gestisca correttamente le richieste al DB | | |
| Prerequisiti: | - | | |
| Procedura: | 1. Aprire la console online di firebase 2. Inviare un messaggio all’interno di un canale | | |
| Risultati attesi: | Il messaggio deve apparire nel database di firebase. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case:  Riferimento: | TC-02  REQ-02 | Nome: | Autentificazione degli utenti |
| Descrizione: | Controllare che gli utenti di possano autentificarsi nella piattaforma. | | |
| Prerequisiti: | - | | |
| Procedura: | 1. Aprire la schermata di login 2. Inserire una mail e una password e registrare un nuovo utente con nickname, indirizzo email e password. 3. Cliccare sul tasto di registrazione | | |
| Risultati attesi: | Nella console di firebase nella sezione degli utenti dovrà apparire il nuovo utente appena creato | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case:  Riferimento: | TC-03  REQ-03 | Nome: | Gli amministratori devono avere la possibilità di modificare i canali. |
| Descrizione: | Gli amministratori devono poter creare canali. | | |
| Prerequisiti: | - | | |
| Procedura: | 1. Cliccare il tasto nella sidebar per creare un nuovo canale 2. Inserire il nome del canale e il tempo di cancellazione dei messaggi | | |
| Risultati attesi: | Nel database di firebase dovrebbe apparire il nuovo canale  Nella sidebar sotto la categoria channels | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case:  Riferimento: | TC-04  REQ-03 | Nome: | Gli amministratori devono avere la possibilità di modificare i canali. |
| Descrizione: | Gli amministratori devono poter modificare canali. | | |
| Prerequisiti: | TC-03 | | |
| Procedura: | 1. Nella sidebar di fianco ad un canale cliccare l’icona per la modifica 2. Modificare il nome del canale e il tempo di cancellazione dei messaggi 3. Cliccare il tasto di conferma della modifica | | |
| Risultati attesi: | Il nome del canale nella sidebar è quello nuovo inserito e nel database si vede che anche il tempo di cancellazione è cambiato | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case:  Riferimento: | TC-05  REQ-03 | Nome: | Gli amministratori devono avere la possibilità di modificare i canali. |
| Descrizione: | Gli amministratori devono poter eliminare canali. | | |
| Prerequisiti: | TC-03 | | |
| Procedura: | 1. Nella sidebar di fianco ad un canale cliccare l’icona con la X. | | |
| Risultati attesi: | Il canale scompare sia dal database nella console di firebase e nella lista dei canali nella sidebar. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case:  Riferimento: | TC-06  REQ-03 | Nome: | Gli amministratori devono avere la possibilità di modificare i canali. |
| Descrizione: | Gli amministratori devono poter inviare messaggi in broadcast. | | |
| Prerequisiti: | TC-02 | | |
| Procedura: | 1. Accedere alla chat con un utente con credenziali di admin 2. Aprire il menu laterale 3. Cliccare sull’icona per i messaggi in broadcast 4. Scrivere un messaggio e inviarlo | | |
| Risultati attesi: | Nei canali deve apparire il nuovo messaggio contrassegnato con “by broadcast” dopo il nome dell’utente. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case:  Riferimento: | TC-07  REQ-03 | Nome: | Gli amministratori devono avere la possibilità di modificare i canali. |
| Descrizione: | Gli amministratori devono poter bannare utenti. | | |
| Prerequisiti: | - | | |
| Procedura: | 1. Accedere alla chat con un utente con credenziali di admin 2. Aprire il menu laterale 3. Cliccare sull’icona per il ban di un utente 4. Selezionare un utente da bannare e scrivere il motivo 5. Tornare nella schermata principale con il login e provare ad accedere con l’utente che è appena stato bannato | | |
| Risultati attesi: | Deve apparire un messaggio che mostra fino a quando si è stati bloccati. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case:  Riferimento: | TC-08  REQ-04 | Nome: | Configurazione dei messaggi |
| Descrizione: | L’applicazione deve gestire la cancellazione dei messaggi dopo un tempo specificato. | | |
| Prerequisiti: | - | | |
| Procedura: | 1. Creare una nuova chat con un tempo di cancellazione basso come un minuto 2. Inviare un messaggio | | |
| Risultati attesi: | Il messaggio dopo il tempo specificato non deve più essere visibile | | |

## Risultati test

|  |  |
| --- | --- |
| Test Case: | TC-01 |
| Risultato: | Si |
| Test Case: | TC-02 |
| Risultato: | Sì |
| Test Case: | TC-03 |
| Risultato: | Sì |
| Test Case: | TC-04 |
| Risultato: | No |
| Errore | Non ho avuto tempo sufficiente per capire come modificare i dati nel database di firebase e il requisito non è quindi stato rispettato. |
| Test Case: | TC-05 |
| Risultato: | Sì |
| Test Case: | TC-06 |
| Risultato: | Sì |
| Test Case: | TC-07 |
| Risultato: | Sì |
| Test Case: | TC-08 |
| Risultato: | Sì |
| Test Case: | TC-08 |
| Risultato: | Sì |

## Mancanze/limitazioni conosciute

Nei requisiti doveva essere implementata la possibilità di modificare i canali ma non sono riuscita per mancanza di tempo. Inoltre, non ho gestito l’accesso al database quindi la sicurezza dei dati che vengono registrati non è garantita e chiunque conosca l’indirizzo del database riesce ad accedere ai dati.

# Consuntivo

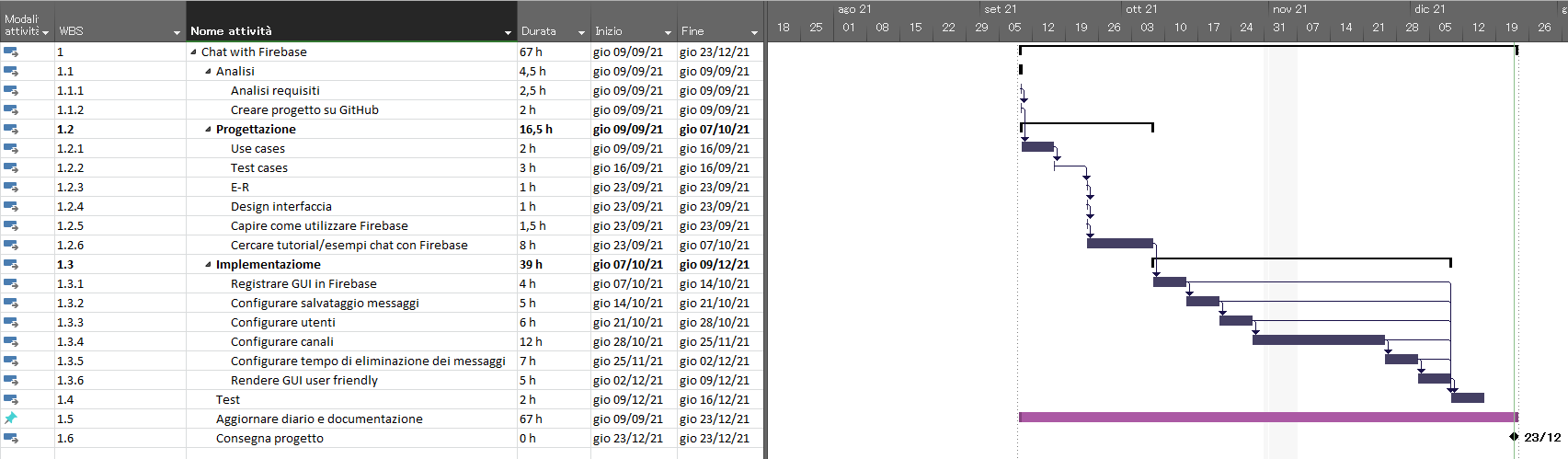


Figura "- Gantt consuntivo

Rispetto alla pianificazione iniziale mi sono resa conto di aver stimato male i tempi e l’ordine delle attività da fare per l’implementazione dato che avevo calcolato delle tempistiche diverse e inserito l’ordine delle attività sbagliato. Sono invece abbastanza soddisfatta della gestione della fase di progettazione che ho saputo valutare meglio. Penso che sia normale che per il primo progetto il Gantt iniziale e consuntivo siano anche molto diversi.

# 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

L’applicazione può essere ampliata aggiungendo per esempio la possibilità di inviare nei canali foto o documenti che possono essere salvati sullo storage online messo a disposizione da firebase. Un’altra aggiunta che sarebbe pratica è quella di avere la possibilità di avere delle chat dirette con gli utenti senza dover creare per forza un canale, questo avvicinerebbe questa applicazione a molte piattaforme per comunicare già esistenti. Potrebbe anche essere implementata la possibilità di autenticarsi con un numero di telefono e ricevere le notifiche dei messaggi direttamente sul telefono, gli sviluppi per questo progetto sono molti anche se già ampiamente implementati da molte piattaforme di comunicazione.

## Considerazioni personali

Con questo primo progetto ho imparato a gestire il tempo a disposizione e ho fatto la mia prima esperienza nel gestire da zero un progetto. Sono soddisfatta del prodotto anche se non ho avuto tempo sufficiente per concludere tutto al meglio e come avrei voluto dato che ci sono alcune mancanze. In questi mesi sono anche riuscita a rendermi meglio conto delle mie capacità e in modo più concreto quello che ho imparato nei primi due anni della scuola che mi hanno permesso di realizzare questa applicazione. Inizialmente riconosco di aver perso tempo che avrei potuto sfruttare in modo diverso ma penso che per il primo progetto sia normale. All’inizio il progetto sembrava molto più complesso di quello che si è poi rivelato anche se ho fatto alcuni grossi cambiamenti che mi hanno semplificato il lavoro come l’utilizzo di bootstrap.

## Sitografia

* <https://wireframepro.mockflow.com> (Consultato il 23.09.2021)
* <https://draw.io> (Consultato il 23.09.2021)
* <https://www.raywenderlich.com/22067733-firebase-tutorial-real-time-chat> (Consultato il 30.09.2021, 07.10.2021)
* <https://getbootstrap.com/docs/5.1/getting-started/introduction/> (Consultato il 07.10.2021)
* <https://firebase.google.com/codelabs/firebase-web#6> (Consultato il 30.09.2021, 07.10.2021)
* <https://getbootstrap.com/docs/5.1/layout/grid/¨> (Consultato il 07.10.2021)
* <https://firebase.google.com/docs/reference/js/v8/firebase.auth.Auth#signinwithemailandpassword> (Consultato il 07.10.2021)
* <https://firebase.google.com/docs/auth/web/password-auth> (Consultato il 07.10.2021)
* <https://medium.com/codingurukul/firebase-for-web-firebase-realtime-database-9280a52ced83> (Consultato il 14.10.2021)
* <https://www.w3schools.com/howto/howto_js_collapse_sidebar.asp> (Consultato il 21.10.2021)
* <https://getbootstrap.com/docs/5.0/examples/sidebars/#> (Consultato il 21.10.2021)
* <https://medium.com/@hasangi/writing-deleting-and-updating-data-in-firebase-realtime-database-with-javascript-f26113ec8c93> (Consultato il 28.10.2021)
* <https://firebase.google.com/docs/reference/js/auth.md#signinwithemailandpassword> (Consultato il 28.10.2021)
* <https://bootswatch.com/> (Consultato il 28.10.2021)
* <https://boxicons.com/> (Consultato il 28.10.2021)
* <https://convertico.com/> (Consultato il 16.12.2021)

# Allegati

Gli allegati di questo progetto sono

* Diari di lavoro
* Codici sorgente
* QdC