

## Lingua dei segni – web trainer

<b>Titolo del progetto</b>	Lingua dei segni – web trainer
<b>Alunno/a</b>	Luca Lorenzon
<b>Classe</b>	I4AA
<b>Anno scolastico</b>	2023/2024
<b>Docente responsabile</b>	Geo Petrini

## 1 Indice

1	Indice .....	2
2	Introduzione .....	4
2.1	Informazioni sul progetto .....	4
2.2	Abstract .....	4
2.3	Scopo .....	4
3	Analisi .....	5
3.1	Analisi del dominio .....	5
3.2	Analisi e specifica dei requisiti .....	5
3.3	Use case .....	9
3.3.1	Utente .....	9
3.3.2	Admin .....	9
3.3.3	Server .....	9
3.4	Pianificazione .....	10
3.5	Analisi dei mezzi .....	11
3.5.1	Software .....	11
3.5.2	Librerie principali .....	11
3.5.3	Hardware .....	12
4	Progettazione .....	13
4.1	Design dell'architettura del sistema .....	13
4.2	Design dei dati e database .....	14
4.2.1	Sign .....	14
4.2.2	Category .....	14
4.2.3	Language .....	14
4.2.4	Word .....	15
4.2.5	User .....	15
4.2.6	History .....	15
4.3	Design delle interfacce .....	16
4.4	Design procedurale .....	21
4.4.1	Swimlane inserimento di un gesto .....	21
4.4.2	Swimlane modalità allenamento .....	23
5	Implementazione .....	25
5.1	Back End .....	25
5.1.1	Struttura delle cartelle .....	25
5.1.2	Database .....	26
5.1.3	REST API .....	28
5.1.4	Socket.IO .....	31
5.1.5	Sistema di riconoscimento .....	33
5.1.6	Protocollo di log .....	34
5.2	Front End .....	35
5.2.1	Struttura Cartelle .....	35
5.2.2	Pagine e navigazione .....	36
5.2.3	Comunicazione con Back End .....	43
5.2.4	Certificati .....	44

5.3	Git Flow .....	45
6	Test .....	46
6.1	Protocollo di test.....	46
6.2	Risultati test .....	56
7	Consuntivo .....	62
7.1	Differenze con la pianificazione .....	63
8	Conclusioni.....	64
8.1	Sviluppi futuri.....	64
8.2	Considerazioni personali.....	64
9	Bibliografia.....	65
9.1	Sitografia .....	65
10	Glossario .....	66
11	Indice delle figure .....	68
12	Allegati.....	69

## 2 Introduzione

---

### 2.1 Informazioni sul progetto

Informazioni generali:

- **Sezione:** Informatica
- **Nome:** Progetto: Lingua dei segni – web trainer
- **Allievo:** Luca Lorenzon
- **Classe:** I4AA
- **Docente Responsabile:** Geo Petrini
- **Perito 1:** Roberto Guidi
- **Perito 2:** Manuele Nolli
- **Data Inizio:** 25.04.2024
- **Data Fine:** 23.05.2024
- **Link Repository:** [http://gitsam.cpt.local/2023\\_2024\\_lpi/lingua-dei-segni-web-trainer](http://gitsam.cpt.local/2023_2024_lpi/lingua-dei-segni-web-trainer)

### 2.2 Abstract

The goal of this project is to create a product that allows users to train in sign language. To achieve this goal, I will create a website where users can select an image or capture in real time from a webcam. The website will use a Machine Learning model to recognize a hand within the image and show the user the words related to the sign they made. On the website, it will be possible to train the user's knowledge by showing an image of the hand and asking the user to replicate the sign. The application will determine if the hand is positioned correctly. Users will also be able to log in to the website to save their learning progress history. In this document it is described the development of the whole project.

### 2.3 Scopo

Lo scopo di questo progetto è quello di creare un sito web dove ci si può allenare ad apprendere il linguaggio dei segni. L'applicativo dovrà avere un'architettura Front End / Back End dove il Back End si occuperà di riconoscere il gesto della mano che l'utente mostra tramite il Front End. Sul sito web sarà anche possibile registrarsi per poter salvare uno storico del proprio apprendimento.

## 3 Analisi

### 3.1 Analisi del dominio

Questo progetto consiste in un sito web dove un utente può imparare e allenare il linguaggio dei segni. I gesti verranno presi dal sito della Federazione Svizzera dei Sordi (<https://www.sgb-fss.ch/signsuisse/it/assistente-di-ricerca-dal-segno>).

Per allenarsi l'utente avrà due opzioni:

- la prima è l'opzione di allenamento dove viene mostrato all'utente un termine e la posizione ideale della mano e lui dovrà replicarla
- la seconda è l'opzione di quiz dove viene mostrato all'utente il termine e lui dovrà mostrare il gesto corretto. Se l'utente ha eseguito l'accesso sul sito, i risultati del quiz verranno salvati all'interno dello storico in modo che l'utente possa controllare la propria precisione nell'eseguire i gesti.

Sul sito l'utente potrà utilizzare una foto rappresentante un gesto con le mani oppure andando a catturare tramite la webcam in real time il movimento delle sue mani. Il riconoscimento della mano verrà eseguita all'interno del Back End utilizzando un modello di Machine Learning.

Il Back End sarà sviluppato con Python mentre il Front End con il framework Sveltekit.

Il riconoscimento della mano verrà fatto facendo distinzione tra la mano destra e la mano sinistra. Quindi se un gesto viene registrato con la mano sinistra verrà riconosciuto solo quando verrà replicato con la mano sinistra e viceversa.

Sul sito sarà anche presente una schermata amministrativa dove un utente autorizzato può registrare dei nuovi gesti con i relativi significati.

Questo sito potrà essere utile a chiunque vorrà imparare il linguaggio dei segni. Esiste già un prodotto simile a questo progetto; per esempio il sito della Federazione Svizzera dei Sordi ha al suo interno gli esempi di come fare tutti i gesti ma non è presente una parte interattiva con l'utente dove può provare lui i gesti.

### 3.2 Analisi e specifica dei requisiti

Req-001	
<b>Nome</b>	Pagina di Registrazione
<b>Priorità</b>	1
<b>Versione</b>	1.0
<b>Note</b>	Sul sito deve essere possibile registrare un account per poter salvare delle statistiche sull'apprendimento dei segni.

### Req-002

<b>Nome</b>	Pagina di Login
<b>Priorità</b>	1
<b>Versione</b>	1.0
<b>Note</b>	Sul sito deve essere possibile accedere ad un account esistente per poter salvare le statistiche sull' apprendimento.

### Req-003

<b>Nome</b>	Pagina di registrazione delle posizioni
<b>Priorità</b>	1
<b>Versione</b>	1.0
<b>Note</b>	La pagina di registrazione delle posizioni della mano sarà visibile solo da un account amministrativo.

#### Sotto requisiti

<b>Req-003_1</b>	Un utente normale non deve poter accedere a questa pagina.
------------------	------------------------------------------------------------

### Req-004

<b>Nome</b>	Pagina di riconoscimento
<b>Priorità</b>	1
<b>Versione</b>	1.0
<b>Note</b>	Questa pagina riconosce la mano mostrata tramite webcam o in un'immagine e mostra il significato.

### Req-005

<b>Nome</b>	Sistema di apprendimento - Allenamento
<b>Priorità</b>	1
<b>Versione</b>	1.0
<b>Note</b>	La pagina di allenamento serve per allenare un utente ad apprendere il linguaggio dei segni mostrando come dovrebbe essere posizionata la mano.

#### Sotto requisiti

<b>Req-005_1</b>	La posizione della mano dovrà idealmente essere in sovrapposizione alla webcam.
------------------	---------------------------------------------------------------------------------

### Req-006

<b>Nome</b>	Sistema di apprendimento - Quiz
<b>Priorità</b>	1
<b>Versione</b>	1.0
<b>Note</b>	La pagina di quiz serve per testare il proprio apprendimento.

### Req-007

<b>Nome</b>	Riconoscimento dei gesti
<b>Priorità</b>	1
<b>Versione</b>	1.0
<b>Note</b>	Il sistema dovrà riuscire a riconoscere le sei posizioni base della mano, due segni per ogni posizione base, due categorie per segno, in lingua italiana e una seconda.

### Req-008

<b>Nome</b>	Riconoscimento Real-time
<b>Priorità</b>	1
<b>Versione</b>	1.0
<b>Note</b>	Il riconoscimento dovrà avvenire in real-time tramite la webcam.

### Req-009

<b>Nome</b>	Storico apprendimento
<b>Priorità</b>	1
<b>Versione</b>	1.0
<b>Note</b>	Dovrà essere presente una pagina con lo storico dell'apprendimento dell'utente.

### Sotto requisiti

<b>Req-009_1</b>	Gli utenti non registrati non devono poter accedere a questa pagina.
------------------	----------------------------------------------------------------------

### Req-010

<b>Nome</b>	Sistema di log
<b>Priorità</b>	2
<b>Versione</b>	1.0
<b>Note</b>	Sul Back End dovrà essere presente un sistema di log con anche dati sulle performance.

Req-011	
<b>Nome</b>	Sito responsive
<b>Priorità</b>	2
<b>Versione</b>	1.0
<b>Note</b>	Il sito deve adattarsi su diversi dispositivi.

Req-012	
<b>Nome</b>	Modifica del margine d'errore
<b>Priorità</b>	3
<b>Versione</b>	1.0
<b>Note</b>	Durante il riconoscimento in tempo reale dovrà essere possibile modificare il margine d'errore utilizzato per il riconoscimento del gesto.

### 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.



### 3.3 Use case

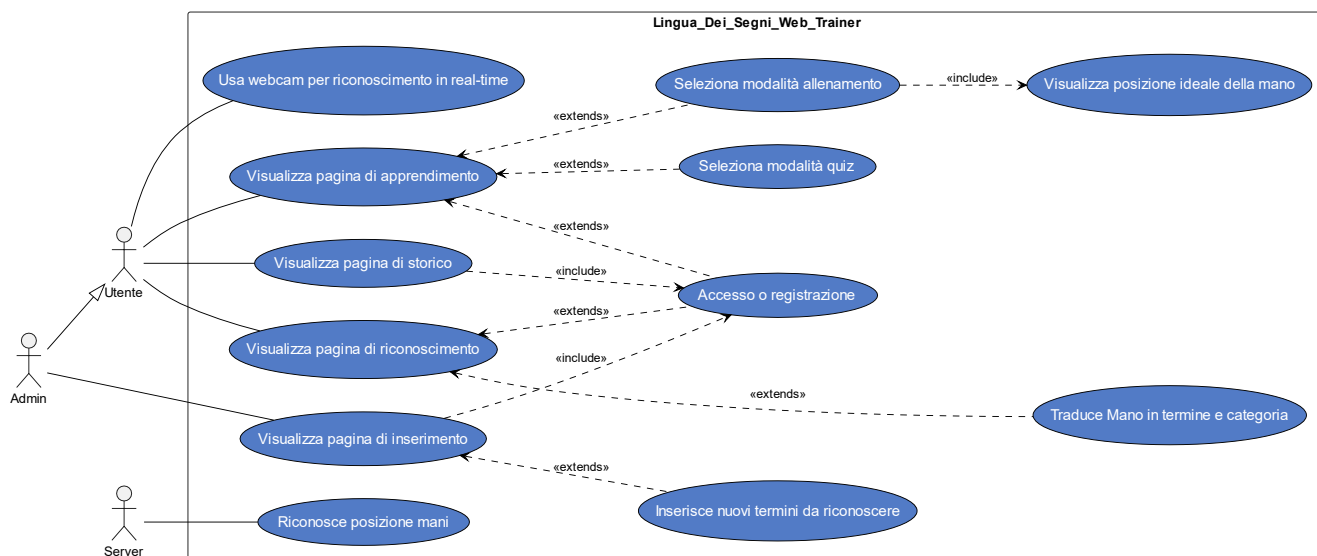


Figura 1 Use Case

In questo progetto sono presenti 3 attori: l'utente, l'admin e il server.

#### 3.3.1 Utente

L'utente, all'interno del sito, può aprire la pagina di riconoscimento e la pagina d'apprendimento senza dover per forza eseguire l'accesso. Invece per accedere alla pagina dello storico è necessario essere registrati sul sito. Sulla pagina di riconoscimento l'utente può ottenere i termini con la categoria di un gesto eseguito. Dalla pagina di apprendimento invece può scegliere tra la modalità allenamento dove viene mostrato un termine con la posizione ideale e lui deve replicarla, oppure la modalità quiz dove viene mostrato all'utente solo il termine e lui deve eseguire il gesto corretto. Inoltre l'utente può utilizzare la webcam per riconoscere la posizione della mano in real-time.

#### 3.3.2 Admin

L'admin può svolgere tutte le attività che può svolgere un utente ma in più può anche accedere alla pagina di registrazione dei nuovi gesti da riconoscere.

#### 3.3.3 Server

Il server invece ha come unico compito quello di riconoscere la posizione delle mani dell'utente.

### 3.4 Pianificazione

Per questo progetto ho deciso di utilizzare il metodo waterfall facendo un diagramma di Gantt. Il diagramma di Gantt l'ho diviso in delle sezioni principali ovvero: Analisi, Pianificazione, Implementazione e Documentazione. Inoltre nella sezione dell'Implementazione sono presenti due sottosezioni: la sezione Autenticazione e la sezione Riconoscimento mano. Le attività di documentazione e dei diari durano tutta la durata del progetto essendo che vengono fatte in parallelo al resto del lavoro.

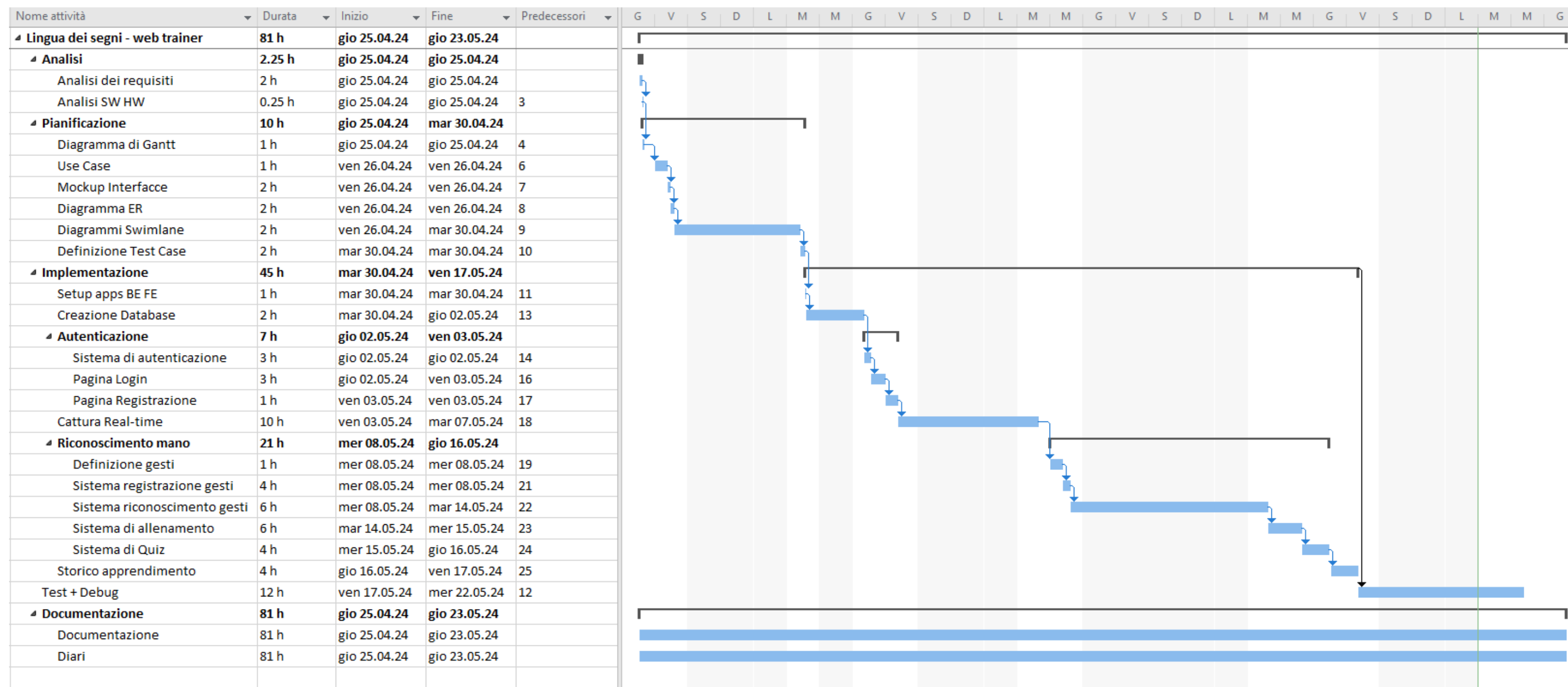


Figura 2 Diagramma di Gantt

## 3.5 Analisi dei mezzi

### 3.5.1 Software

Per lo sviluppo del progetto ho utilizzato i seguenti software:

- Visual Studio Code 1.78.2
- PlantUML VSCode Extension 2.17.5
- Microsoft Word 2019
- Microsoft Project 2019
- MySQL Workbench
- Google Chrome
- Firefox
- Postman 10.24.24

In più ho anche utilizzato i seguenti siti web:

- <https://www.figma.com/>
- <https://dbdiagram.io/>
- <https://lucid.app/>
- <https://regeery.com/en/security/ssl-tools/self-signed-certificate-generator>

### 3.5.2 Librerie principali

#### 3.5.2.1 Back End

- Flask 3.0.3
- Socket.IO 5.11.2
- SQLAlchemy 2.0.29
- Mediapipe 0.10.11

#### 3.5.2.2 Front End

- Svelte 4.2.15
- Tailwind 3.4.3
- Skeleton 2.9.2
- Vite 5.2.10
- Socket.io-client 4.7.5
- Axios 1.6.8

### **3.5.3 Hardware**

Per lo sviluppo di questo progetto mi è stato fornito dalla scuola un PC con le seguenti caratteristiche:

- Sistema Operativo: Windows 10 Enterprise Versione: 22H2
- Processore: Intel(R) Core (TM) i7-9700 CPU @ 3.00GHz
- RAM: 32.0 GB
- GPU: NVIDIA GeForce RTX 2060

In più mi è stata fornita dalla scuola una webcam per poter fare la cattura di immagini sul sito web.

## 4 Progettazione

### 4.1 Design dell'architettura del sistema

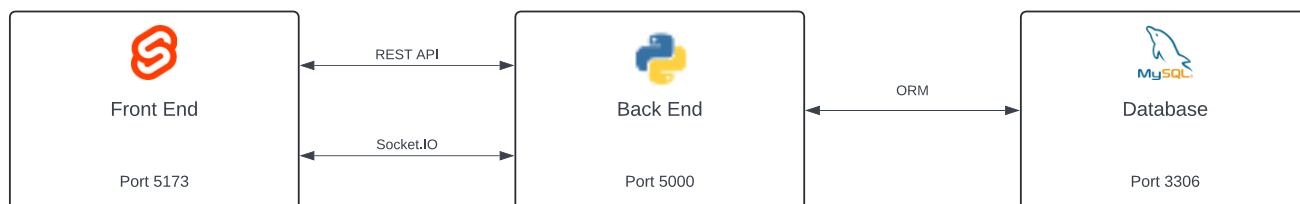


Figura 3 Schema di rete

Questo schema mostra come sarà strutturata l'applicazione. L'applicazione sarà suddivisa in tre parti: il Front End, il Back End e il database.

Per il Front End verrà utilizzato il framework Sveltekit e l'utente potrà raggiungere il sito accedendo sulla porta 5173.

Il Back End verrà sviluppato con Python e il framework Flask e sarà raggiungibile sulla porta 5000.

Il Front End e il Back End comunicheranno tramite una REST API e Socket.IO.

Infine sarà presente anche un database MySQL che sarà raggiungibile sulla porta 3306. Per operare sul database verrà utilizzato un ORM.

## 4.2 Design dei dati e database



Figura 4 Schema logico del database

Il database di quest'applicazione è composto dalle seguenti 6 tabelle.

### 4.2.1 Sign

Questa tabella serve per salvare le informazioni generali sulla posizione di una mano. Come attributi ha un id come chiave primaria, un testo che sarà un JSON convertito in stringa contenente la mappatura della posizione che deve avere la mano all'interno dell'immagine, un numero per identificare a quale posizione di base appartiene la posizione della mano e un numero per identificare quale posizione rappresenta la mappatura.

### 4.2.2 Category

La tabella "category" serve per salvare le categorie dei segni. Questa tabella ha un attributo per identificare la categoria e il nome della categoria.

### 4.2.3 Language

La tabella "language" serve per salvare la lingua delle parole salvate. Questa tabella ha come attributi un id come chiave primaria e il nome della lingua.

#### 4.2.4 Word

Questa tabella serve per contenere i termini salvati che il programma deve riconoscere. Un termine ha come attributo un id come chiave primaria, il termine in sé, a quale gesto della mano è assegnato, a quale categoria fa parte e in che lingua è.

#### 4.2.5 User

Questa tabella serve per salvare gli utenti registrati sul sito. Un utente ha un nome utente identificativo, una password con una lunghezza fissa di 64 caratteri essendo che viene applicato alla password l'algoritmo di hash SHA-256. Inoltre un utente può avere anche un token che viene utilizzato per identificarsi nelle richieste API. Questo token ha una lunghezza fissa di 72 caratteri essendo che viene utilizzato un metodo di salting e di hash per generare i token. Il token ha una data di creazione per poter determinare quando scade. Infine l'utente ha anche la proprietà admin che serve per definire se l'account è un account amministrativo o no.

#### 4.2.6 History

La tabella "history" viene utilizzata per salvare lo storico dell'apprendimento. Un valore dello storico ha come proprietà un id univoco, un valore di precisione nell'esecuzione del gesto, una data di quanto è stato salvato, l'utente che ha salvato il gesto nello storico e quale termine si stava replicando.

### 4.3 Design delle interfacce

logo

Registrati

Nome utente

Password

Conferma password

Accedi

Continua senza account

Registrati

Figura 5 Pagina di registrazione

logo

Accedi

Nome utente

Password

Registrati

Continua senza account

Accedi

Figura 6 Pagina di accesso

Le pagine di registrazione e di accesso sono molto simili fra loro. L'unica differenza è che la pagina di registrazione ha un campo in più per la conferma della password. Queste due pagine servono per registrare o accedere ad un account esistente per poter accedere allo storico del profilo. Da queste pagine si può continuare sul sito senza eseguire l'accesso.



logo

Riconosci Apprendimento Storico **Inserisci**

username

posizione base



Segno



Lingua

IT

Categoria

[Categoria]

Termine +

[Termine] -

[Termine] -

Inserisci

Figura 7 Pagina di inserimento

Questa pagina è visibile solo da un amministratore e serve per registrare dei gesti per poterli poi riconoscere. Per registrare un nuovo segno bisogna prima caricare o scattare una foto e poi selezionare le informazioni del gesto ovvero la posizione base, quale gesto dovrebbe essere, la categoria dei termini che si vogliono inserire, la lingua dei termini e infine uno o più termini corrispondenti al gesto eseguito.

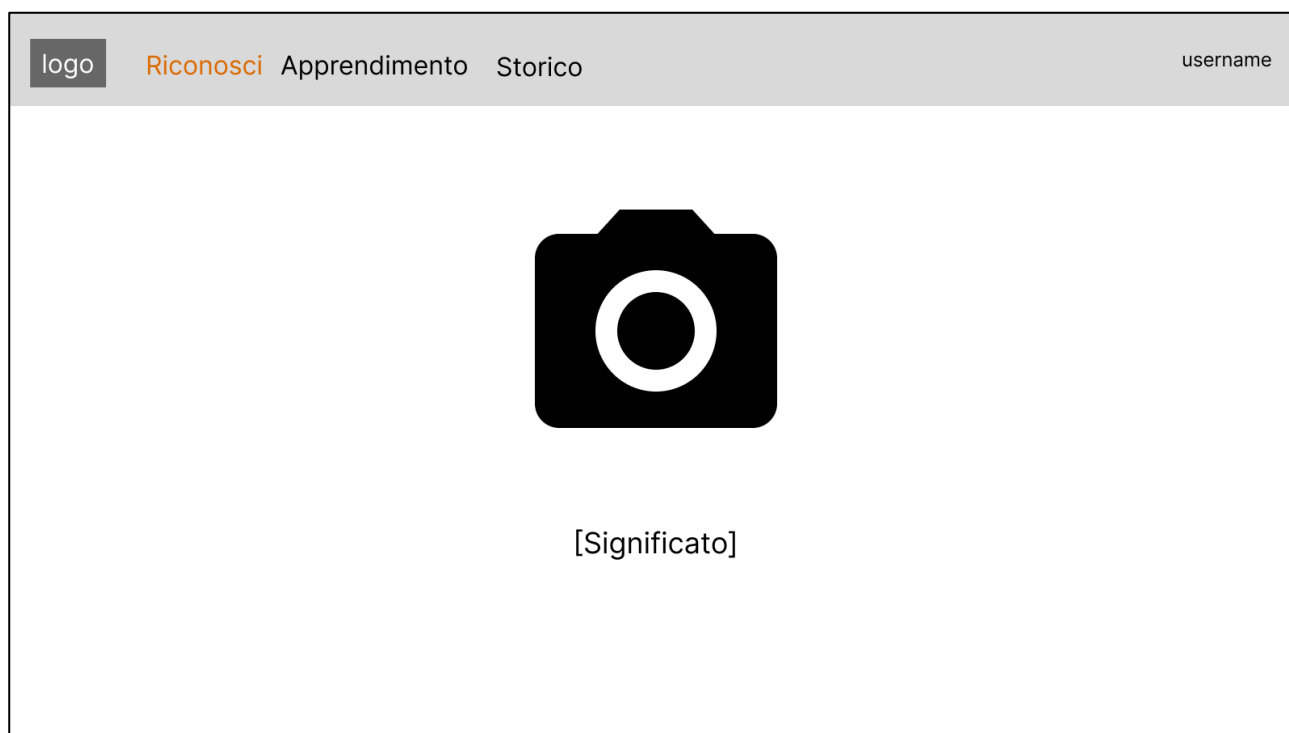


Figura 8 Pagina di riconoscimento

Questa pagina serve per riconoscere i propri gesti eseguiti. Per funzionare si può utilizzare il riconoscimento in tempo reale tramite la webcam oppure caricando una foto da riconoscere. Quando viene rilevato un gesto registrato verranno mostrati i possibili significati a schermo.



Figura 9 Pagina di apprendimento



Figura 10 Pagina di allenamento / Quiz

Dalla pagina di apprendimento si può scegliere tra due modalità: allenamento o quiz. Le schermate delle due modalità sono praticamente le stesse con l'unica differenza che per la modalità di allenamento viene mostrato come deve essere posizionata la mano per fare il gesto corretto. Le schermate hanno al centro lo stream della webcam per poter fare il riconoscimento in tempo reale, sopra viene mostrato quale categoria e termine bisogna indovinare. Sotto la webcam sono presenti due pulsanti per passare al termine successivo o precedente e in mezzo appare un messaggio quando si compie il gesto corretto. Infine in alto a sinistra è presente un tasto per uscire dalla modalità di apprendimento.

logo	Riconosci	Apprendimento	Storico	username
	[categoria]: [termine]	valore migliore	data	
	[categoria]: [termine]	valore migliore	data	
	[categoria]: [termine]	valore migliore	data	
	[categoria]: [termine]	valore migliore	data	

Figura 11 Pagina dello storico

La pagina dello storico è visualizzabile soltanto se si esegue l'accesso sul sito. Nello storico vengono salvate le statistiche di apprendimento dei segni mostrando l'immagine del gesto, la categoria e il termine su cui si stava allenando l'utente assieme ai miglior valori ottenuti e le date di quando sono stati ottenuti.

## 4.4 Design procedurale

### 4.4.1 Swimlane inserimento di un gesto

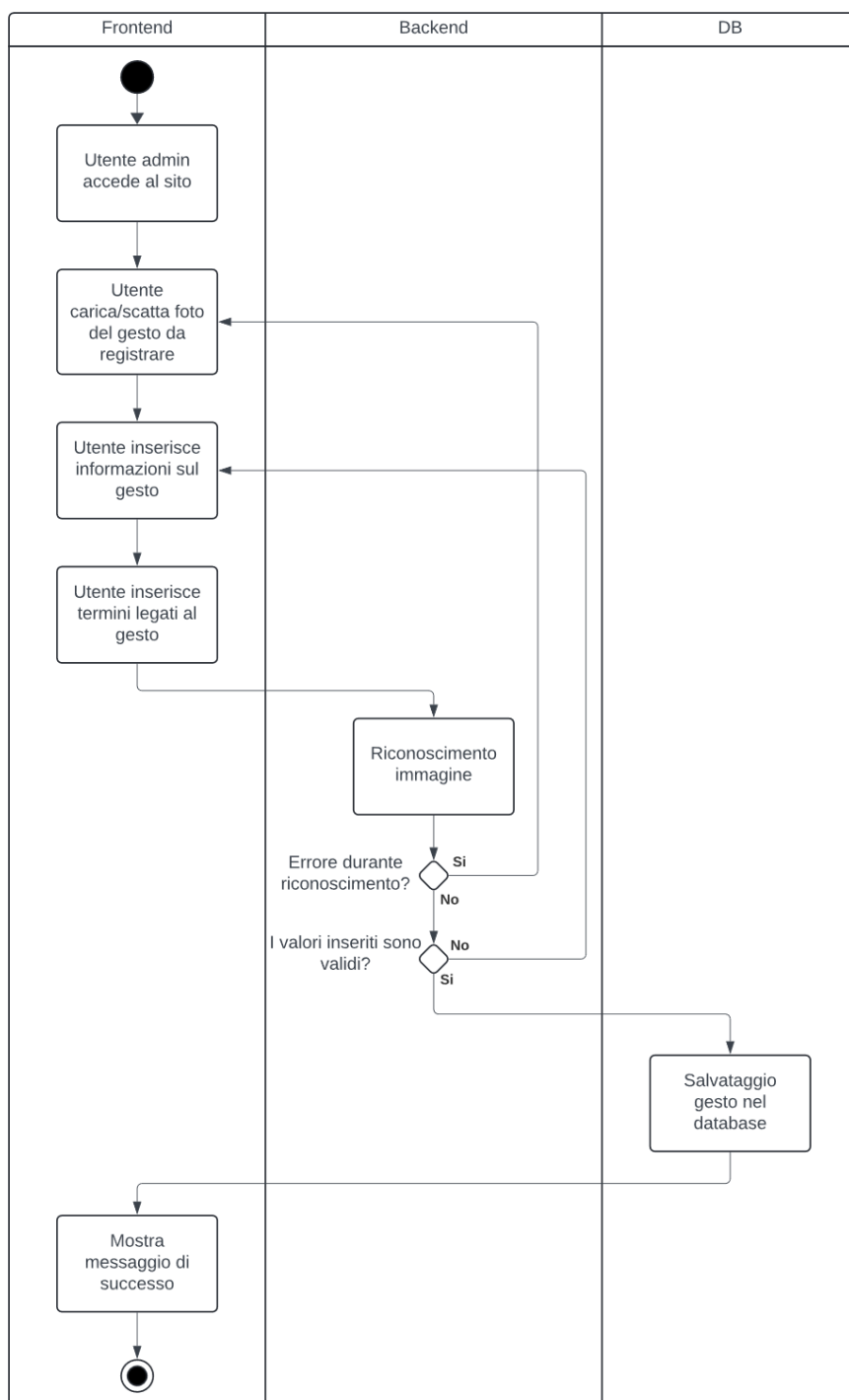


Figura 12 UML inserimento gesto

Questo schema mostra il funzionamento dell'inserimento di un nuovo gesto da riconoscere dal sistema. Il risultato finale che si vuole ottenere è quello di aver salvato nel database la mappa dei punti della mano nella posizione corretta.

Per prima cosa un utente amministratore deve accedere al sito web e aprire la pagina di inserimento. Da qui l'amministratore può caricare o scattare una foto dove si vede bene la mano nella posizione che si vuole salvare. Dopo aver scelto l'immagine viene richiesto di inserire la posizione base del gesto, selezionare quale gesto si sta cercando di replicare e i termini con la rispettiva lingua e categoria che si vogliono registrare.

Una volta inviato tutto al server viene riconosciuta la mano all'interno della foto e viene controllato anche che tutti gli altri valori passati siano validi. Se tutto va a buon fine viene salvato nel database il nuovo gesto e di conseguenza il sistema sarà capace di riconoscere anche questo gesto. Una volta salvato il gesto nel database all'amministratore verrà mostrato un messaggio di successo.

#### 4.4.2 Swimlane modalità allenamento

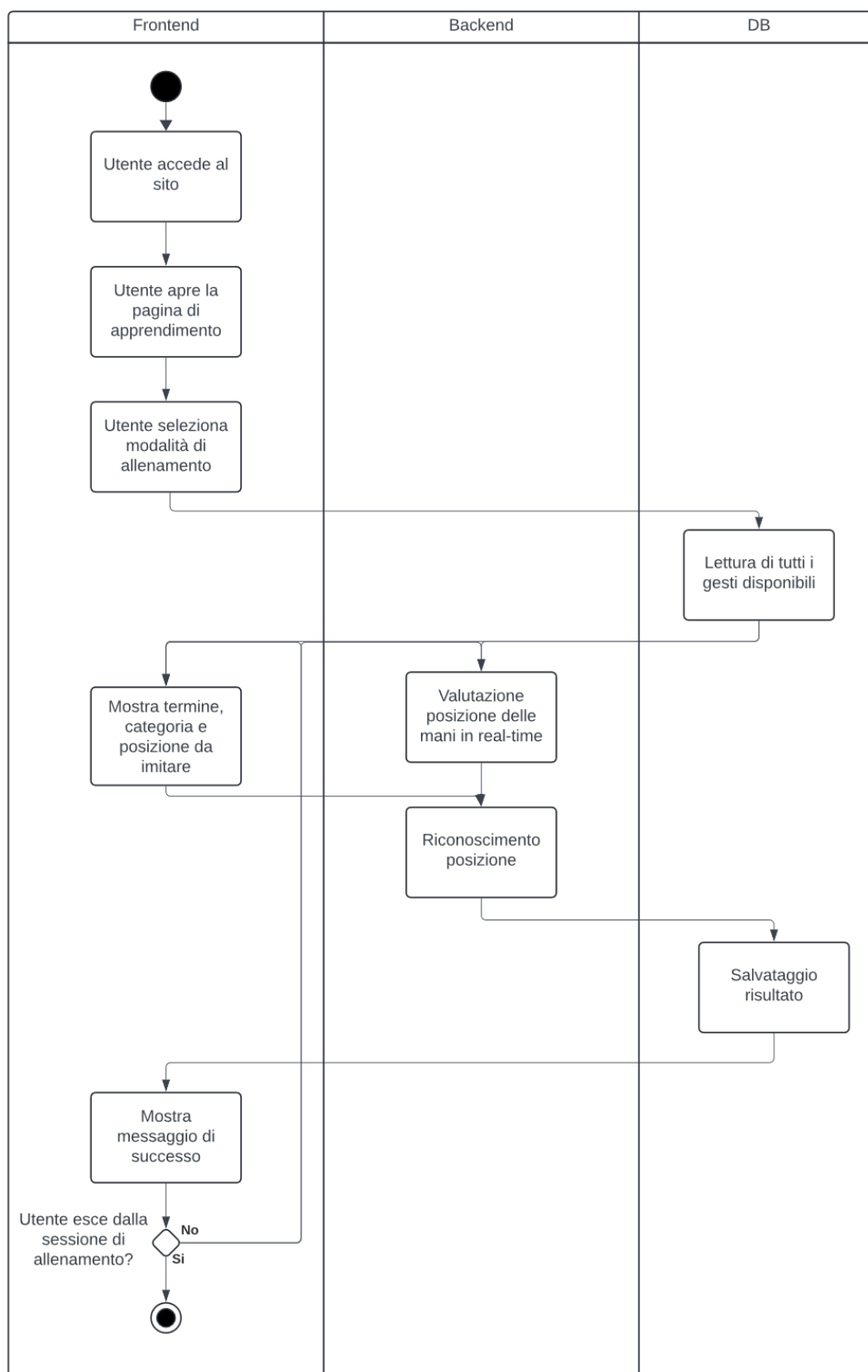


Figura 13 UML modalità allenamento

Questo schema mostra il funzionamento della modalità di allenamento all'interno del sito. Il risultato che si vuole ottenere è una pagina che aiuta l'utente a posizionare le mani in modo corretto per imitare i gesti al meglio.

Per farlo come prima cosa un utente qualsiasi deve accedere al sito web e aprire la pagina di apprendimento. Da qui va selezionata la modalità allenamento. Quando viene avviata questa modalità vengono letti dal database tutti i termini salvati che vengono poi mostrati a schermo.

All'utente infatti viene mostrato il termine con la sua categoria e un'immagine di esempio di come deve posizionare la mano. Nel mentre viene avviata la cattura tramite la webcam e i frame iniziano a venir inviati al server.

Il server, ricevendo i frame della webcam valuta in tempo reale la posizione della mano dell'utente e controlla che sia simile a quella richiesta. Una volta che il gesto viene riconosciuto viene mostrato all'utente un messaggio di successo e l'utente può passare alla parola successiva, a quella precedente oppure uscire dalla sessione di allenamento.



## 5 Implementazione

### 5.1 Back End

Per il Back End ho deciso di utilizzare Python (3.10.6) utilizzando il framework Flask. Ho optato per questa soluzione perché conoscevo già queste tecnologie. Utilizzando il framework Flask ho creato una REST API per poter fare una parte di comunicazione con il Front End. Per comunicare non ho utilizzato solo la REST API ma anche Socket.IO per poter fare una comunicazione bilaterale che mi è servita per implementare certe funzionalità.

Il database che ho utilizzato è un database MySQL e per interfacciarmi ho utilizzato l'ORM SQLAlchemy.

#### 5.1.1 Struttura delle cartelle

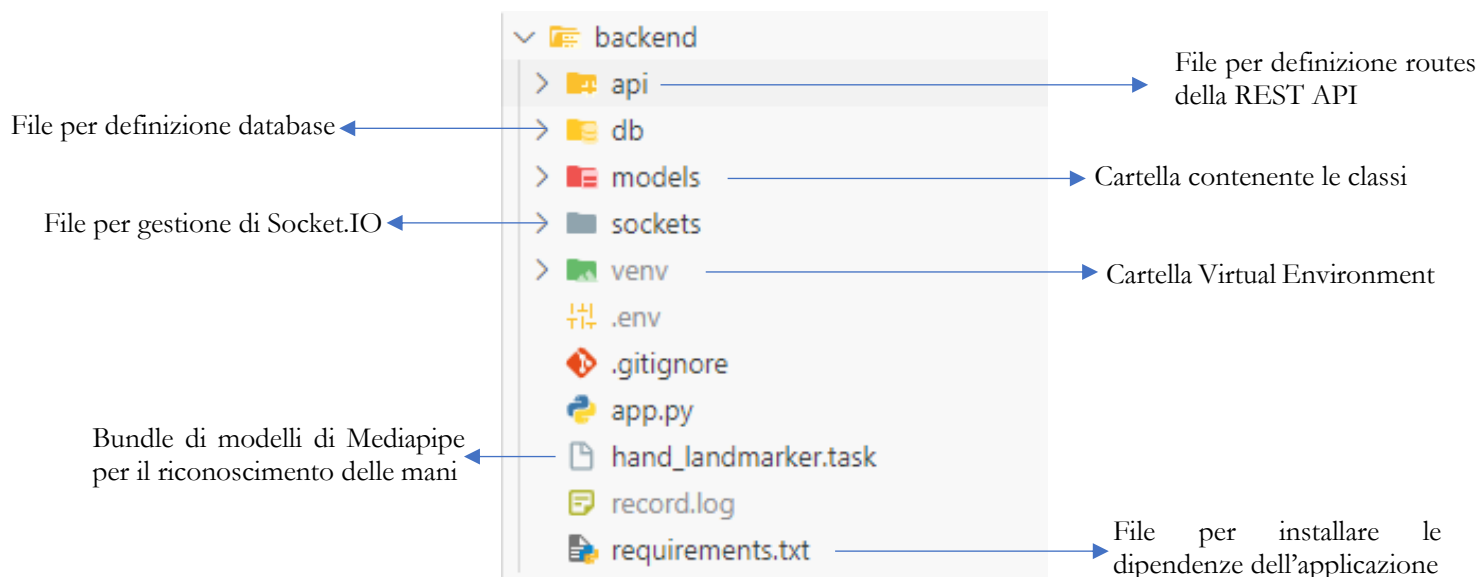


Figura 14 Struttura della cartella Back End

## 5.1.2 Database

Per poter lavorare con il database come prima cosa bisogna configurare l'applicazione Flask passandogli la stringa di connessione al database che è stata salvata all'interno di un file env.

```
app.config['SQLALCHEMY_DATABASE_URI'] = os.getenv('DB_CONNECTION')
```

Come visto dal diagramma ER il database è composto da 6 tabelle.

### 5.1.2.1 User

```
class User(db.Model):  
    username = db.Column(db.String(20), primary_key=True)  
    password = db.Column(db.String(64).with_variant(CHAR(64), "mysql", "mariadb"))  
    token = db.Column(db.String(72).with_variant(CHAR(72), "mysql", "mariadb"))  
    token_creation = db.Column(db.DateTime)  
    admin = db.Column(db.Integer().with_variant(TINYINT(), "mysql", "mariadb"))  
    histories = db.relationship('History', backref='user')
```

Questa è la definizione della tabella User. Questa tabella ha le proprietà username, password, token, token\_creation e admin. Il campo “histories” serve per poter creare la relazione tra la tabella user e la tabella history. Questa classe possiede anche dei metodi che servono per gestire l'autenticazione dell'utente. Questi metodi verranno spiegati in seguito.

### 5.1.2.2 Category

```
class Category(db.Model):  
    id = db.Column(db.Integer, primary_key=True, autoincrement=True)  
    name = db.Column(db.String(20))  
    words = db.relationship("Word", backref='category')
```

La tabella “Category” possiede le proprietà id e name. Anche in questa tabella è presente un campo che serve per poter creare una foreign key ovvero il campo words.

### 5.1.2.3 Language

```
class Language(db.Model):  
    id = db.Column(db.Integer, primary_key=True, autoincrement=True)  
    name = db.Column(db.String(20))  
    words = db.relationship("Word", backref='language')
```

La tabella “Language”, a livello di struttura è uguale alla tabella Category. Infatti anche questa tabella ha come proprietà un id e il nome e possiede il campo words per poter fare da foreign key.

### 5.1.2.4 Sign

```
class Sign(db.Model):
    id = db.Column(db.Integer, primary_key=True, autoincrement=True)
    base_position = db.Column(db.Integer())
    sign_position = db.Column(db.Integer())
    map = db.Column(LONGTEXT)
    words = db.relationship("Word", backref='sign')
```

Questa è la definizione della tabella “Sign”. Questa tabella ha come proprietà un id, il numero della posizione iniziale, il numero che rappresenta il gesto e una mappatura del gesto. Anche in questa tabella è presente il campo words per poter fare da riferimento.

### 5.1.2.5 Word

```
class Word(db.Model):
    id = db.Column(db.Integer, primary_key=True, autoincrement=True)
    word = db.Column(db.String(20))
    category_fk = db.Column(db.Integer, db.ForeignKey('category.id'))
    language_fk = db.Column(db.Integer, db.ForeignKey('language.id'))
    sign_fk = db.Column(db.Integer, db.ForeignKey('sign.id'))
    histories = db.relationship('History', backref='word')
```

La tabella “Word” ha come proprietà un id e la parola. Oltre a queste proprietà ha 3 riferimenti ad altre tre tabelle ovvero category\_fk, language\_fk e sign\_fk. Anche questa tabella ha un campo per fare da foreign key alla tabella history.

### 5.1.2.6 History

```
class History(db.Model):
    id = db.Column(db.Integer, primary_key=True, autoincrement=True)
    value = db.Column(db.Float)
    date = db.Column(db.DateTime)
    user_fk = db.Column(db.String(20), db.ForeignKey('user.username'))
    word_fk = db.Column(db.Integer, db.ForeignKey('word.id'))
```

Questa è la definizione della tabella “History”. Come proprietà ha un id, un valore di precisione e una data di quando è stato salvato il dato. Questa tabella ha anche due proprietà che fanno riferimento ad altre due tabelle ovvero i campi user\_fk e word\_fk.

### 5.1.3 REST API

La REST API viene utilizzata per fare una parte di comunicazione tra il Front End e il Back End. Ogni richiesta da come risposta un JSON e uno status code che verranno poi gestiti dal Front End. In alcuni casi i JSON inviati come risposta sono vuoti perché al Front End si basa solo sullo status code per eseguire certe azioni.

#### 5.1.3.1 Autenticazione

Per gestire l'autenticazione delle richieste vengono generati dei token nel momento in cui l'utente fa login. Questi token vengono salvati all'interno del database e passati all'utente in modo che per le richieste successive si possa autenticare con quello invece che dover inviare ad ogni richiesta il nome utente e la password.

```
def create_auth_token(self):
    salt = bcrypt.gensalt()
    token = bcrypt.hashpw(self.username.encode('utf-8'), salt)
    self.token = token
    self.token_creation = datetime.now()
    db.session.commit()
```

Questo è il metodo che viene utilizzato per creare i token. Questo metodo si trova all'interno della classe User, la stessa che utilizza l'ORM per creare la tabella. Per generare i token viene utilizzato un metodo di salting per renderli più sicuri. Oltre al token nel database viene anche salvata la data di creazione essendo che i token hanno una scadenza. Una volta che il token è scaduto bisogna rieseguire l'accesso per rinnovare il token.

```
@staticmethod
def verify_auth_token(token):
    user = User.query.filter_by(token=token).first()
    if not user:
        logger.info(f'token {token} is not valid')
        return False
    if user.token_creation < datetime.today() - timedelta(days=30):
        logger.info(f'token for user {user.username} expired, need to login')
        return False
    return user
```

Per verificare che un token sia valido viene utilizzato questo metodo che controlla se nel database è presente il token che è stato utilizzato per autenticarsi e se esiste viene controllato che non sia scaduto.

```
@auth.verify_password
def verify_password(usr_or_tkn, psw):
    # check token validity
    user = db_models.User.verify_auth_token(usr_or_tkn)
    if not user:
        # check password validity
        user = db_models.User.query.filter_by(username = usr_or_tkn).first()
        if not user or not user.verify_password(psw):
            return False
    g.user = user
    return True
```

Per verificare l'autenticazione, prima di eseguire ogni richiesta che la richiede, viene eseguita questa funzione che controlla come prima cosa che il token sia valido, e nel caso non lo fosse controlla se esiste un utente che ha come username il primo parametro passato. Se esiste viene verificato che la password corrisponda con quella salvata nel database. Se anche in questo caso l'autenticazione fallisce perché la password non è corretta oppure perché non esiste un utente con il nome utente passato, verrà ritornato all'utente un errore con lo status code 401.

### 5.1.3.2 Routes

#### 5.1.3.2.1 Login

La route /api/login utilizza il metodo GET e permette ad un utente di eseguire l'accesso. Questa richiesta richiede l'autenticazione e nel caso sia valida viene ritornato all'utente il token con il quale si può autenticare per le prossime richieste.

#### 5.1.3.2.2 Register

La route /api/register utilizza il metodo POST e serve per permettere ad un utente di creare un proprio account sul sito. Per registrare un nuovo account sono necessari un nome utente e una password. Il nome deve essere lungo almeno quattro caratteri, al massimo venti e non deve essere già utilizzato da un altro utente. La password invece deve essere lunga almeno sei caratteri e contenere almeno un numero e un carattere speciale. Per verificare che la password sia valida viene utilizzata una regular expression.

```
PSW_PATTERN = re.compile("(^(?=.*[0-9])(?=.*[!@#$%^&*])[a-zA-Z0-9!@#$%^&*]{6,}$")
```

Se tutti i criteri vengono soddisfatti l'account viene creato all'interno del database.

#### 5.1.3.2.3 User

La route /api/use utilizza il metodo GET e serve per ottenere delle informazioni riguardanti l'utente. Questa richiesta necessita l'autenticazione e se questa è valida viene ritornato all'utente il suo nome utente e la sua proprietà admin per poter conoscere i suoi permessi.

#### 5.1.3.2.4 History

La route `/api/history` utilizza il metodo GET e serve per raccogliere tutti i dati salvati nello storico dell'utente. Questa richiesta necessita l'autenticazione essendo che lo storico è legato all'utente.

#### 5.1.3.2.5 History add

La route `/api/history/add` utilizza il metodo POST e serve per salvare un nuovo valore all'interno dell'history. Prima di aggiungere il nuovo valore viene controllato che quell'utente non abbia già un valore salvato per la parola che sta salvando in quel momento. Se esiste già un valore viene controllato che sia più piccolo del nuovo valore. Nel caso lo fosse viene cambiato il valore all'interno del database con quello nuovo. Se invece non esiste ancora un valore, questo viene salvato normalmente.

#### 5.1.3.2.6 Insert

La route `/api/insert` utilizza il metodo POST e serve per registrare un nuovo gesto all'interno del sito. Questa route necessita l'autenticazione e che l'utente sia un amministratore. In questa route viene utilizzato il sistema di riconoscimento della mano per salvare i punti di posizione della mano all'interno del database. Oltre alla mappatura viene salvato anche la posizione base del gesto, un numero che serve per risalire quale gesto rappresenta la mappatura e tutti i suoi termini legati, salvando la lingua del termine, la categoria e il termine in sé.

#### 5.1.3.2.7 Recognize

La route `/api/recognize` utilizza il metodo POST e serve per riconoscere il gesto presente in una singola immagine caricata. Per richiamare questa route non è necessario essere autenticati sul sito. Se il gesto dell'utente è un gesto registrato gli verranno ritornati tutti i termini rappresentati dal gesto con anche le categorie e le lingue.

## 5.1.4 Socket.IO

Socket.IO è stato utilizzato principalmente per la funzionalità del riconoscimento in real-time tramite la webcam. Questi sono gli eventi implementati sul Back End per Socket.IO.

### 5.1.4.1 start\_recognition

Questo evento serve per collezionare i dati del database come per esempio tutti i gesti salvati per accedere alla mappatura. Questo evento viene chiamato all'avvio del riconoscimento in real-time in modo che dopo non c'è più bisogno di andare a leggere questi dati dal database.

```
global signs
global languages
global categories
signs = db_models.Sign.query.all()
languages = db_models.Language.query.all()
categories = db_models.Category.query.all()
```

### 5.1.4.2 stream

Questo evento invece serve per ricevere i frame della webcam per poi andare a riconoscere la mano. Infatti questo evento, dopo aver verificato che il formato dell'immagine sia valido, richiama il sistema di riconoscimento per mappare la mano all'interno del frame. Dopodiché viene confrontata la mappatura della mano con tutte le mappature salvate all'interno del database. Se i gesti corrispondono vengono inviati all'utente tutti i termini legati al gesto assieme alla categoria e alla lingua.

```
for s in signs:
    saved_map = json.loads(s.map)
    if utils.validate_maps(json_pos, saved_map, threshold):
        data = collect_sign_data(s.id)
        results.append({'base_position': s.base_position, 'sign_position': s.sign_position, 'words': data})
```

### 5.1.4.3 stream\_learn

Questo evento serve per ricevere i frame della webcam quando si è nella modalità di allenamento. Infatti in questo caso il gesto dell'utente non viene confrontato con tutti i gesti salvati nel database ma solo con quello che è stato mostrato all'utente. Se le due mappe corrispondono viene inviato all'utente un messaggio positivo, in modo che capisca che il gesto è stato eseguito correttamente.

```
json_pos = utils.cord_to_json(pos.hand_world_landmarks)
saved_map = json.loads(signs[data['id']-1].map)

if utils.validate_maps(json_pos, saved_map):
    emit('real_time_learn_msg', {'msg': 'OK'})
```

### 5.1.4.4 stream\_quiz

Questo evento serve per ricevere i frame catturati dalla webcam quando si è nella modalità quiz. Il processo che viene eseguito è simile a quello della modalità di allenamento, con l'unica differenza che invece che controllare se il gesto dell'utente sia uguale a quello da imitare viene soltanto calcolata una media tra i delta per poter calcolare la precisione dell'utente nell'eseguire il gesto.

```
json_pos = utils.cord_to_json(pos.hand_world_landmarks)
saved_map = json.loads(signs[data['id']-1].map)

delta = utils.calc_deltas(json_pos, saved_map)
```

### 5.1.4.5 learn\_next\_word

Questo evento serve sempre durante la modalità di apprendimento per richiedere una nuova parola da apprendere. Per farlo viene generato un indice randomico dalla lista di parole in modo da non avere sempre la stessa sequenza durante l'apprendimento. Il numero random però non può essere lo stesso generato la volta prima, in modo da evitare di non avere la stessa parola due volte di fila.

```
words = db_models.Word.query.all()
diff_index = True
logger.info(f"Getting next word", request.remote_addr)
while diff_index:
    w_index = random.randint(0, len(words)-1)
    diff_index = w_index == data['wordId']-1
word = words[w_index]
w_data = collect_word_data(word)
```



### 5.1.5 Sistema di riconoscimento

Per il riconoscimento della mano ho utilizzato la libreria Mediapipe che permette di utilizzare dei modelli Machine Learning già esistenti. Per questo progetto è stato utilizzato un bundle contenente due modelli Machine Learning: il primo che serve per riconoscere il palmo della mano e il secondo che serve per riconoscere i punti di riferimento di una mano (vedi immagine sotto). Grazie a questi due modelli si riescono a calcolare le coordinate di questi punti in modo da ottenere una mappatura della mano.

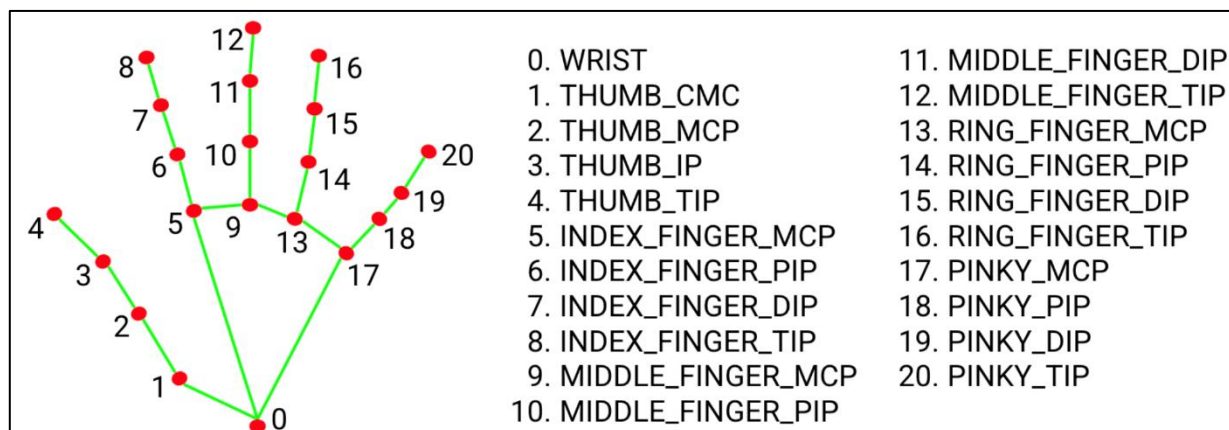


Figura 15 Punti di riferimento della mano

Le coordinate che vengono salvate sono relative al centro della mano in modo che dove è posizionata la mano all'interno della foto non è rilevante per il confronto. Per verificare che due mappe corrispondano viene fatta la differenza tra le coordinate e se le tre differenze (x, y, z) sono minori di un delta viene considerato il punto nella posizione corretta. Infine viene verificato che tutti i punti siano posizionati nel modo corretto e se lo sono il gesto viene considerato lo stesso nelle due mappe.

```
for i, v in enumerate(current_map):
    diff_x = abs(current_map[i]['x'] - saved_map[i]['x'])
    diff_y = abs(current_map[i]['y'] - saved_map[i]['y'])
    diff_z = abs(current_map[i]['z'] - saved_map[i]['z'])

    pos_ok.append(diff_x < sign_threshold and diff_y < sign_threshold and diff_z < sign_threshold)
```

Per scegliere il valore di default del delta è stata calcolata la differenza delle coordinate con due mappature molto simili e con i risultati ottenuti è stato determinato il delta. Quando si esegue il riconoscimento in tempo reale è possibile modificare il delta in modo da rendere più o meno preciso il controllo.

Per il calcolo della precisione (il valore che viene salvato nello storico) vengono prese le differenze delle coordinate e si fa una media. Più il valore ottenuto è basso più l'esecuzione è stata corretta. Per questo calcolo viene anche preso in considerazione che certi punti devono avere meno peso di altri e quindi viene fatta una media ponderata.

```
for i, v in enumerate(current_map):
    diff_x = abs(current_map[i]['x'] - saved_map[i]['x'])
    diff_y = abs(current_map[i]['y'] - saved_map[i]['y'])
    diff_z = abs(current_map[i]['z'] - saved_map[i]['z'])
    d = (diff_x + diff_y + diff_z) / 3
    if i in lower_weights:
        d *= lower_w
    f_delta += d
return f_delta / (len(current_map)-(len(lower_weights)*lower_w))
```

Prima di venir salvato nel database il delta viene trasformato in una percentuale.

```
def convert_delta(d):
    return 100-((d*100)/(def_threshold*100)*100)
```

## 5.1.6 Protocollo di log

Sul Back End è presente un protocollo di log, in modo da tenere traccia delle operazioni eseguite dall'utente. Il formato dei log è il seguente:

```
[LIVELLO] - UTENTE - - [gg/m/aaaa hh:mm:ss] MESSAGGIO
```

In tutta l'applicazione vengono usati solo tre livelli dei log ovvero info, warning e error. Il protocollo di log viene usato principalmente dalla REST API per salvarsi le operazioni eseguite dall'utente ed eventuali errori avvenuti. Nella parte di comunicazione con Socket.IO invece sono presenti pochi log perché se vengono fatti dei log durante lo stream della webcam, i log si riempiono subito con messaggi quasi del tutto uguali.

I log, oltre ad apparire nel terminale dell'applicazione, vengono scritti all'interno di un file chiamato record.log. Questo file può pesare un massimo di dieci MB. Una volta che il file si riempie del tutto viene fatta una rotazione dei log, ovvero il file viene rinominato e viene creato un nuovo file record.log dove verranno scritti i nuovi log. La rotazione arriva ad un massimo di due file extra, dopodiché vengono eliminati i log più vecchi.

### 5.1.6.1 Performance

Assieme al protocollo di log è presente una funzione che misura il tempo di certe operazioni, e una volta finita l'operazione viene stampato un log che indica la durata.

```
def performance(self, msg, func, *args):
    bf = utils.current_time_ms()
    res = func(*args)
    af = utils.current_time_ms()
    self.app.logger.info(f'PERFORMANCE - [{utils.get_formatted_date()}] {msg}: {af-bf} ms')
    return res
```

## 5.2 Front End

Per il Front End ho deciso di utilizzare il framework Sveltekit utilizzando Typescript. Assieme a Sveltekit ho utilizzato la libreria Skeleton per lavorare alla grafica. Questa libreria utilizza anche Tailwind, ovvero un framework CSS.

### 5.2.1 Struttura Cartelle

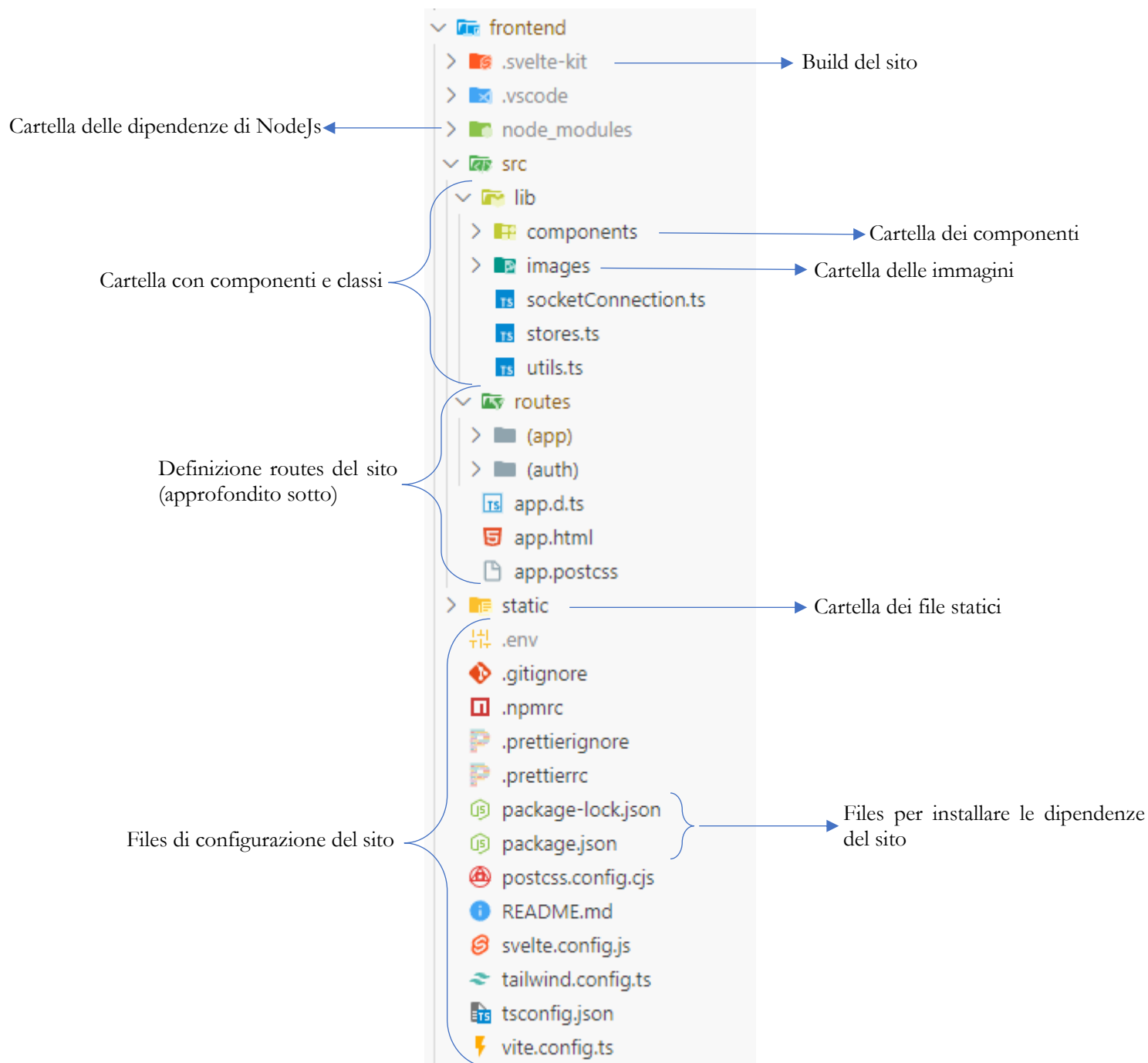


Figura 16 Struttura della cartella Front End

## 5.2.2 Pagine e navigazione

Per gestire lo stile del sito ho utilizzato un tema messo a disposizione da Skeleton. Questo tema utilizza la seguente palette di colori.



Figura 17 Palette dei colori

### 5.2.2.1 Navigazione

All'interno del sito sono presenti diverse pagine: la pagina principale/riconoscimento, la pagina di apprendimento, la pagina di allenamento, la pagina del quiz, la pagina di resoconto del quiz, la pagina dello storico, le pagine di inserimento (la prima per la cattura dell'immagine e la seconda per la selezione dei dati) e le pagine di autenticazione (accesso e registrazione).

Per la navigazione tra le pagine Sveltekit utilizza un *filesystem-based router* ovvero le route del sito vengono definite dalle cartelle all'interno del progetto.

Ogni cartella necessita al suo interno un file `+page.svelte` che è il file contenente la struttura della pagina. I file `+layout.svelte` servono per definire un layout da applicare a tutte le pagine all'interno di un gruppo. All'interno del sito sono presenti due gruppi: il gruppo `app` e il gruppo `auth` essendo che le pagine di autenticazione necessitavano un layout diverso rispetto alle altre pagine.

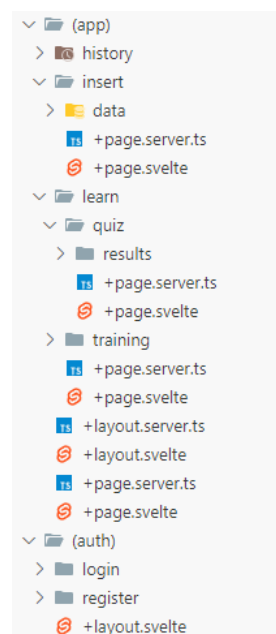


Figura 18 Definizione routes del sito

Per navigare tra le pagine del sito è presente una barra di navigazione. Le pagine di riconoscimento e di apprendimento sono accessibili da tutti, la pagina dello storico solo da chi ha eseguito l'accesso e la pagina di inserimento solo da un account admin.

Il logo del sito (visibile alla sinistra della pagina di navigazione) è stato generato con Microsoft Copilot.



Figura 19 Barra di navigazione

### 5.2.2.2 Pagine di autenticazione

Le pagine di autenticazione, ovvero le pagine di accesso e registrazione, sono molto simili tra di loro a livello grafico. Infatti l'unica differenza è che la pagina di registrazione ha un campo in più per poter confermare la password.

Figura 20 Pagina di registrazione

Nel caso in cui la registrazione o l'accesso non vada a buon fine, in basso apparirà un messaggio che avviserà l'utente e i campi di testo verranno svuotati. Questo messaggio andrà via in automatico dopo 3 secondi.

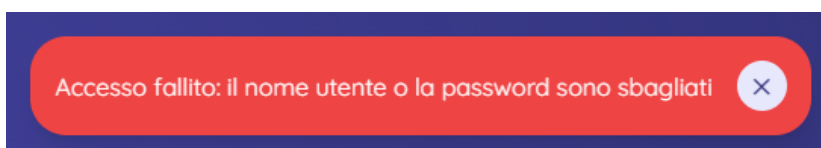


Figura 21 Errore d'autenticazione

### 5.2.2.3 Pagina di riconoscimento

La pagina di riconoscimento è la pagina principale del sito. In questa pagina è possibile aprire la webcam per riconoscere i gesti in tempo reale oppure caricare una foto e riconoscere il gesto da questa.

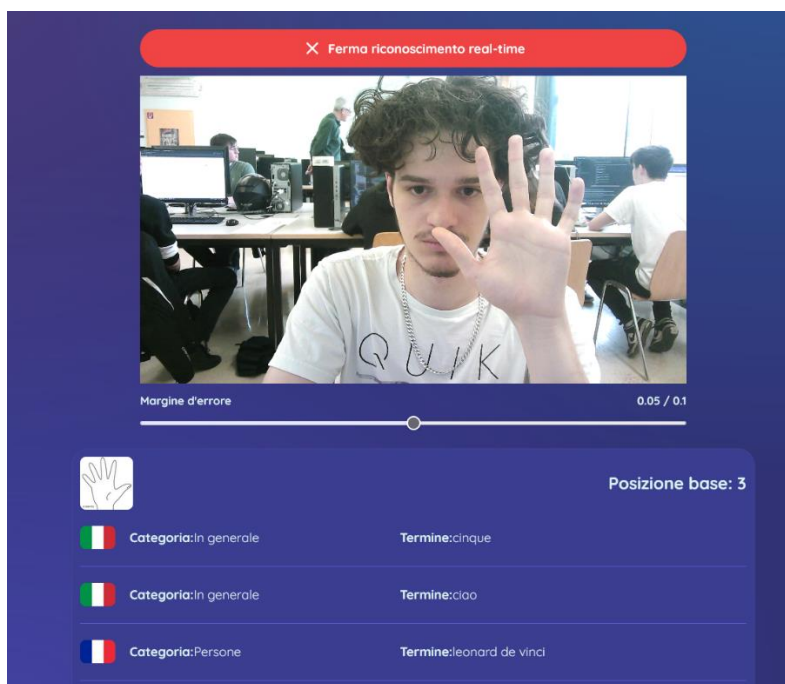


Figura 22 Riconoscimento in tempo reale

Quando un gesto viene riconosciuto verranno mostrati i termini legati al gesto sotto alla webcam assieme ad un'immagine del gesto riconosciuto, il numero della posizione base, la lingua e la categoria. Sotto la webcam è anche possibile modificare il margine d'errore (il delta) che viene utilizzato per il riconoscimento in modo da poter cambiare la precisione del riconoscimento dei gesti.

```
let interval = setInterval(() => {
  const canvas = document.createElement("canvas");
  canvas.width = videoSource.videoWidth;
  canvas.height = videoSource.videoHeight;
  canvas.getContext("2d").drawImage(videoSource, 0, 0, canvas.width, canvas.height);
  let img = canvas.toDataURL("image/jpeg");
  io.emit('stream', {'img':img, 'threshold': tValue});
}, 1000/fps);
```

Questo intervallo serve per inviare i frame catturati dalla webcam al server tramite Socket.IO. Per accedere ad un frame, lo stream della webcam viene salvato all'interno di un canvas per poi convertirlo in un'immagine. La trasmissione viene effettuata a 30 fps.

#### 5.2.2.4 Pagine di apprendimento

Le pagine di apprendimento sono 3: la pagina principale dove si può selezionare se si vuole cominciare la modalità d'allenamento o la modalità di quiz, la pagina per la modalità allenamento e la pagina per la modalità quiz. Le pagine di allenamento e di quiz sono molto simili a livello grafico.

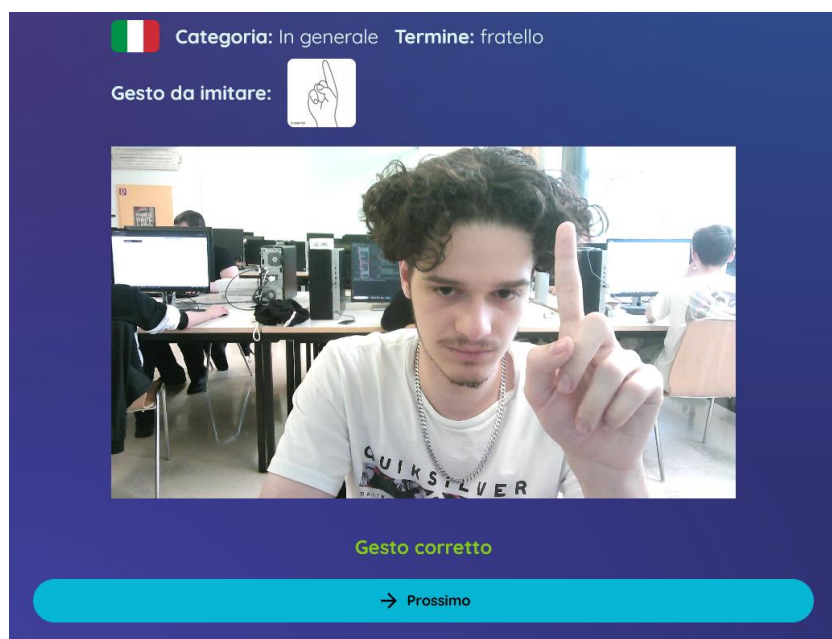


Figura 23 Pagina di allenamento

La differenza principale a livello grafico tra la pagina d'allenamento e quella di quiz è che nella pagina di quiz non viene mostrato all'utente la foto del gesto da imitare e non viene comunicato all'utente quando ha eseguito il gesto corretto. I risultati del quiz vengono mostrati solo alla fine del quiz (dalla durata di 10 termini) mostrando all'utente la percentuale di precisione ottenuta nei diversi termini.

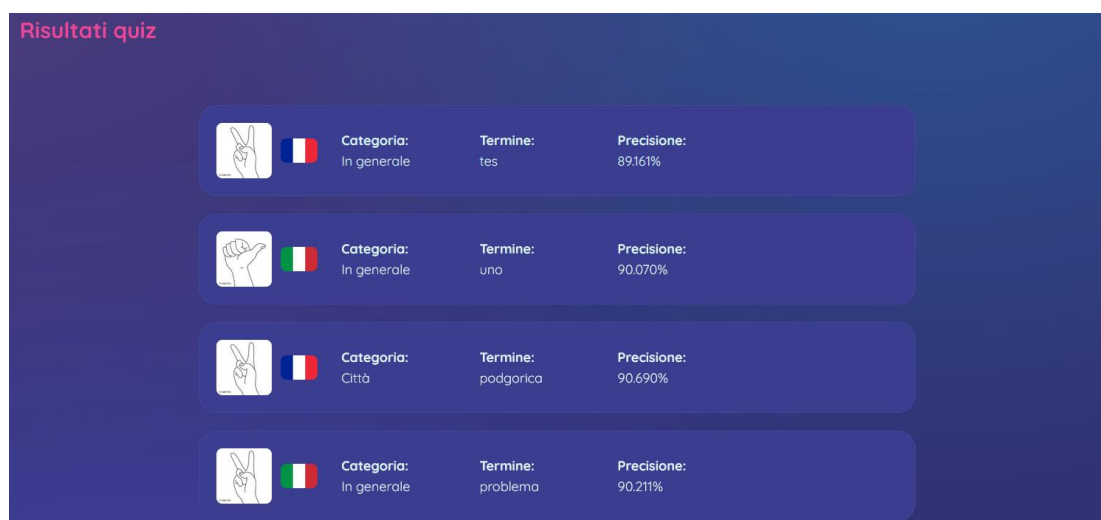


Figura 24 Pagina di risultato del quiz



### 5.2.2.5 Pagina dello storico

La pagina serve per visualizzare le proprie statistiche di apprendimento dei segni. Questa pagina è accessibile soltanto se si ha eseguito l'accesso. Nello storico si possono vedere tutti i termini che sono stati eseguiti durante i quiz mostrando la percentuale di precisione più alta ottenuta eseguendo il gesto legato al termine. Inoltre è anche mostrata la data e l'ora dell'esecuzione del valore migliore.

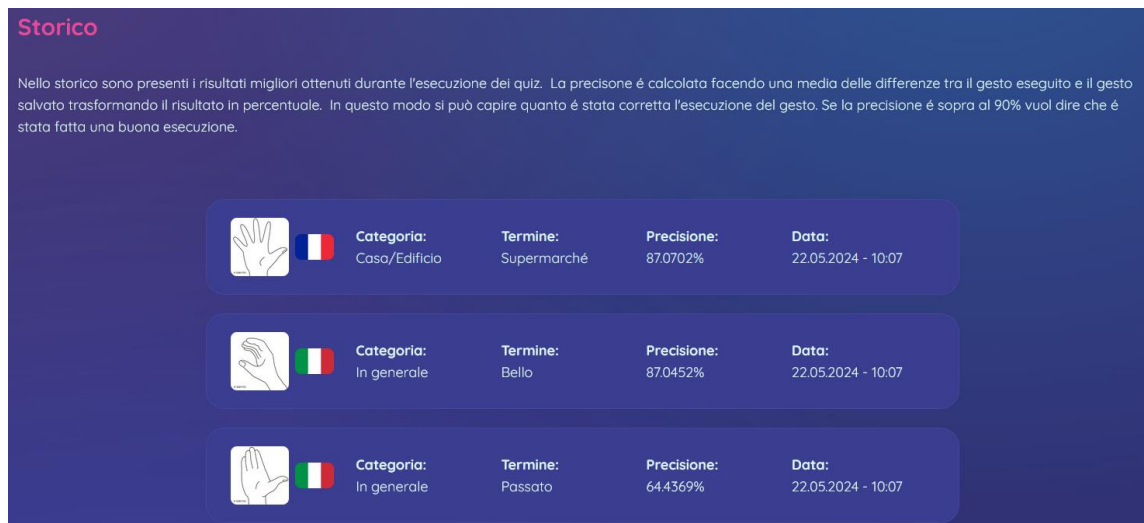


Figura 25 Pagina dello storico

Nel caso che nello storico non sono presenti dei valori, verrà mostrato all'utente un messaggio per informarlo.

Non ci sono valori presenti nello storico. Fai un quiz per salvare all'interno dello storico i tuoi risultati

Figura 26 Messaggio storico vuoto



### 5.2.2.6 Pagine di inserimento

Come detto in precedenza le pagine per l'inserimento di un nuovo gesto sono due: una prima pagina per la cattura o l'upload dell'immagine e la seconda per la selezione del gesto eseguito nella foto e l'inserimento dei termini legati al gesto.

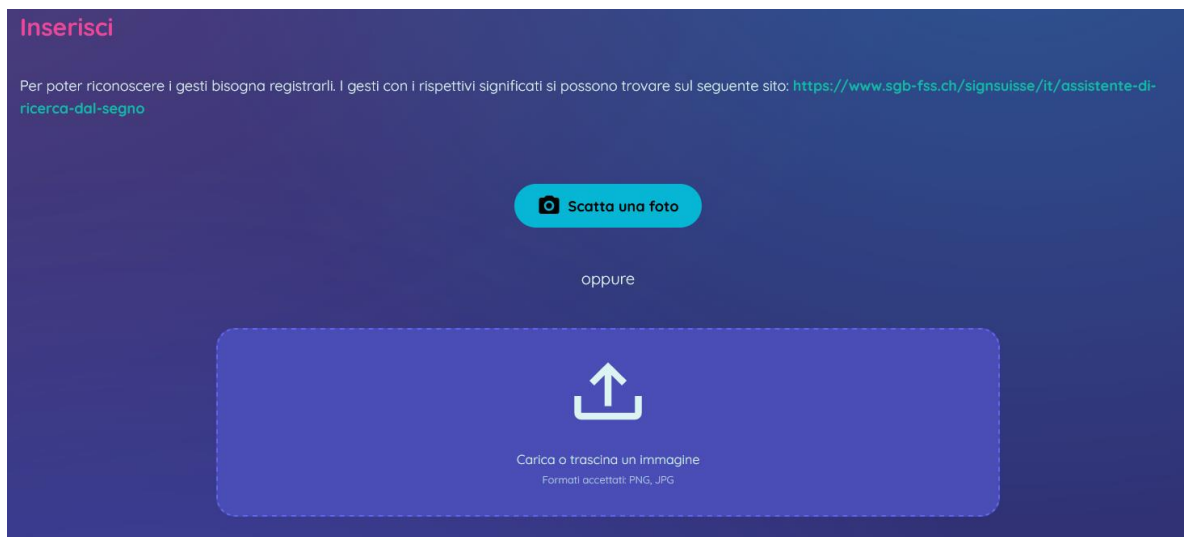


Figura 27 Pagina di inserimento (cattura immagine)

Così è come si presenta la schermata di inserimento. Una volta scelta l'immagine, prima di inserire il resto dei dati, viene chiesta la conferma all'utente di utilizzare l'immagine.

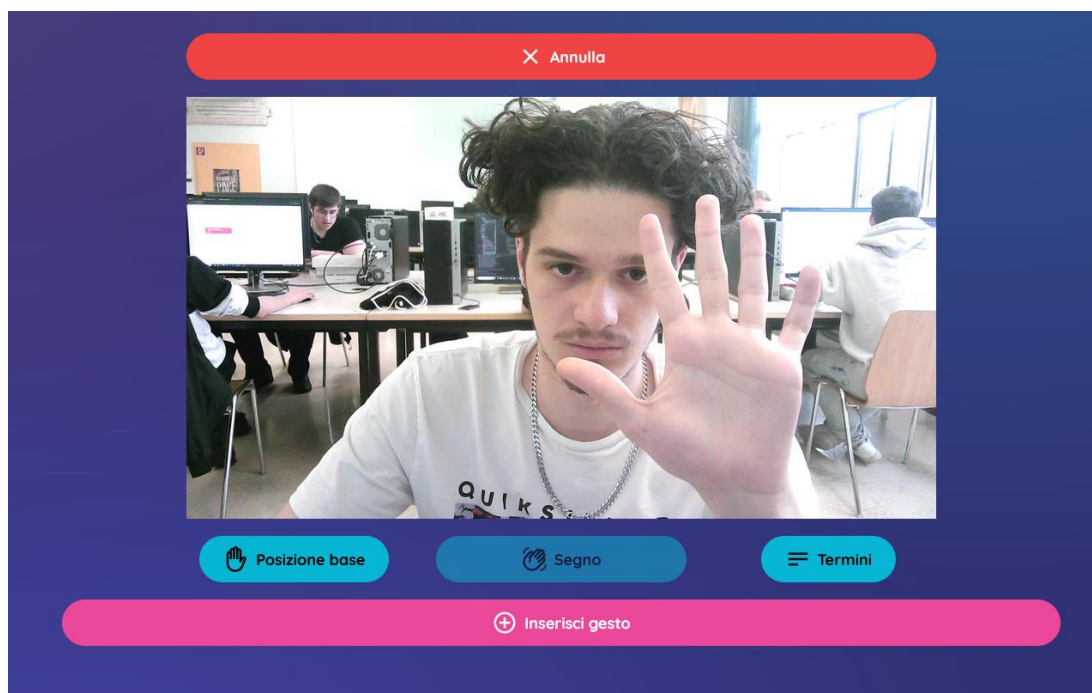


Figura 28 Schermata di inserimento (dati)

Una volta scelta l'immagine bisogna selezionare a quale posizione base appartiene il gesto. Una volta scelta la posizione base, si deve anche selezionare il segno che si sta eseguendo. Infine si possono inserire i termini selezionando anche una categoria e una lingua (scegliendo tra italiano e francese).



Figura 29 Selezione del gesto eseguito

Figura 30 Inserimento dei termini

### 5.2.2.7 Responsive

Se il sito viene aperto da un telefono, la grafica del sito subirà delle modifiche per permettere una visualizzazione ordinata del sito. La differenza più evidente è la modifica del menu di navigazione; infatti da telefono la barra di navigazione non conterrà più i pulsanti per accedere alle altre pagine ma sarà presente solo un pulsante che apre un menù laterale.

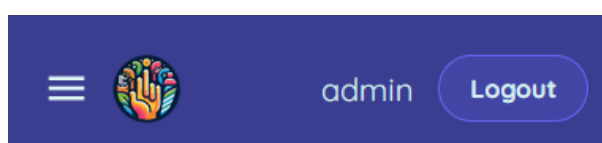


Figura 31 Barra di navigazione da telefono

## 5.2.3 Comunicazione con Back End

### 5.2.3.1 REST API

Per fare le richieste sulla REST API viene utilizzata la libreria axios. All'interno del Front End è presente una funzione che crea un oggetto di tipo AxiosRequestConfig che viene utilizzato per inviare una richiesta.

```
export function configRequest(method:string,url:string,auth: RequestAuth, data: any = {}):AxiosRequestConfig{
  let config: AxiosRequestConfig = {
    method: method,
    url: url,
    auth: auth,
    data: data,
    headers: { "Content-Type": "multipart/form-data" },
    httpsAgent: new https.Agent({
      rejectUnauthorized: false
    })
  };
  return config;
}
```

In questo modo non bisogna riscrivere per ogni richiesta dei parametri che rimangono sempre uguali (come headers e httpsAgent).

#### 5.2.3.1.1 Gestione errori

Quando una richiesta non va a buon fine viene ritornato all'utente un messaggio d'errore. Questo messaggio viene controllato in uno switch che ritorna una stringa da mostrare all'utente in modo che capisca qual è stato il problema.

```
case "username-length-invalid":
  return { 'msg': 'La lunghezza del nome utente non é valida. Il nome utente deve essere lungo da 4 a 20 caratteri' }
```

### 5.2.3.2 Socket.IO

Per la parte di comunicazione in tempo reale è stato utilizzato Socket.IO. Gli eventi su cui ascolta il Front End vengono iscritti nel mount della pagina e prima di farlo vengono cancellati tutti gli eventi registrati in precedenza.

```
onMount(() =>{
  io.off();
  io.on('real_time_learn_msg', (res: any) => {
    if(unlockButton) return;
    switch(res.msg){
      case 'OK':
        unlockButton = true;
        msg = "Gesto corretto";
        break;
      case 'NOK':
        msg = "Gesto sbagliato";
        break;
    }
  })
  io.on('new_word', (res: any) => {
    word = res;
    signId = res.sign_id
  })
  io.emit('start_recognition');
  nextWord();
})
```

### 5.2.4 Certificati

Per questioni di sicurezza, per poter accedere alla webcam, è richiesto l'utilizzo del protocollo HTTPS. Per questo motivo è stato implementato un certificato self-signed in modo da poterci accedere.

```
export default defineConfig({
  server: {
    https: {
      key: '../signlanguage-privateKey.key',
      cert: '../signlanguage.crt',
    }
  },
  plugins: [sveltekit(), purgeCss()]
});
```

### 5.3 Git Flow

Per gestire le versioni del progetto è stato utilizzato Git Flow. Questo consiste di creare almeno una branch separata sul repository Git dove vengono implementate le funzionalità dell'applicativo, e quando si ha sviluppato qualcosa di concreto la branch di sviluppo viene unita alla branch principale (master). In questo modo nella branch principale vengono inserite solo le funzionalità complete.

```
PS E:\lingua-dei-segni-web-trainer> git branch
* develop
master
```

Figura 32 Branch di Git

Purtroppo mi sono accorto a fine progetto, guardando l'albero dei commit di Git, che non ha funzionato tutto come avrebbe dovuto, essendo che quasi tutti i commit risultano stati fatti sulla branch master anche se sono stati fatti sulla branch di develop.

```
* dae7d97 (HEAD -> develop, origin/develop) bug fixes
* cec3dbb aggiunte gestione permesso webcam
* e1221b7 pulizia codice
* d19cabb bug fixes
* 11ab82d (origin/master, origin/HEAD, master) diari doc
* 52b1a5c fix diario
* 93ccfb4 doc e diario
* 22d453f fixes
* 5f87f15 gestione errori OK
* da8c33f gestione errori
* e5e77fd log performance
* 0a74d1d bug fixes + doc
* 608c9c3 test case quasi finiti + doc + use case corretto
* 6e7548e protocollo di log OK
* 90bda28 responsive OK
* 3c5b7f6 sito responsive + miglioramento protocollo di log
* 351588c slider threshold
* 404e192 funzionalità aggiuntive
* 1a61634 gestione errori cominciata
* 3722938 pagina history OK
* deb2593 salvataggio quiz history
* d6fb1ba bbug fixes
* 04bc6c7 bug fixes
* 769efef pagine di apprendimento OK
* 1fbb5d0 fixes
* 3c00d71 sistema riconoscimento da immagine singola quasi OK
* 41296f1 sistema di riconoscimento OK
* 94a4bc4 pulizia
* 0328f00 inserimento OK
* cda5e89 insert page circa OK + inizio richiesta API insert
* 8d82b15 modifica data diario
* f3bb320 Modifica componenti per webcam + pagina di inserimento
* d05aa57 doc + componente FileUploader
* 96e9878 pagina di riconoscimento + cattura in real-time via webcam
* 4988e36 auth system OK on FE
* e37eae2 grafica iniziale sito (accesso e registrazione)
* ab73e6e auth system on backend + ssl cert
* 353998f db created via ORM
* ad15218 frontend project created
* 94c46a6 merged repos
* 9efd5b8 modifica diario
* 855faa7 Merge branch 'develop'
| \
| * 03c908f folder app
| * 17f7b7c repo
| * 955a4a5 prova
| * 5c91090 commit before switch branch
| * 570f54e finita definizione testcase
| * ca90f32 Use case, Mockup, ER, Swimlane e inizio definizione testcase
| /
* f40b92a temp file
* 808d64d diagramma di Gantt, doc e diario
* c0a63b8 Update README.md
```

Figura 33 Albero dei commit

## 6 Test

### 6.1 Protocollo di test

<b>Test Case Riferimento</b>	TC-001 REQ-001	<b>Nome</b>	Registrazione account
<b>Descrizione</b>	Bisogna registrare un nuovo account sul sito		
<b>Prerequisiti</b>	<ol style="list-style-type: none"> <li>1. Aprire il sito web</li> <li>2. Aprire la pagina di registrazione</li> </ol>		
<b>Procedura</b>	<ol style="list-style-type: none"> <li>1. Inserire un nome utente e una password due volte</li> <li>2. Premere il tasto di registrazione</li> </ol>		
<b>Risultati attesi</b>	L'account dovrà venir creato con successo e l'utente verrà portato sulla pagina iniziale. In alto a destra dovrà anche essere visibile il nome utente.		

<b>Test Case Riferimento</b>	TC-002 REQ-001	<b>Nome</b>	Registrazione con account non valido
<b>Descrizione</b>	Bisogna registrare un account con dei valori non validi		
<b>Prerequisiti</b>	<ol style="list-style-type: none"> <li>1. Aprire il sito web</li> <li>2. Aprire la pagina di registrazione</li> </ol>		
<b>Procedura</b>	<ol style="list-style-type: none"> <li>1. Inserire un nome utente e una password non validi</li> <li>2. Premere il tasto di registrazione</li> </ol>		
<b>Risultati attesi</b>	L'account non dovrà venir creato e dovrà apparire anche un messaggio d'errore che ricorda all'utente le convenzioni del nome utente e della password.		

<b>Test Case Riferimento</b>	TC-003 REQ-002	<b>Nome</b>	Accesso ad un account
<b>Descrizione</b>	Bisogna eseguire l'accesso ad un account esistente		
<b>Prerequisiti</b>	<ol style="list-style-type: none"> <li>1. Aprire il sito web</li> <li>2. Aprire la pagina di accesso</li> <li>3. Deve esistere un account</li> </ol>		
<b>Procedura</b>	<ol style="list-style-type: none"> <li>1. Inserire un nome utente e una password di un account già esistente</li> <li>2. Premere il tasto di accesso</li> </ol>		
<b>Risultati attesi</b>	L'utente verrà riportato sulla pagina principale e in alto a destra sarà visibile il suo nome utente. Inoltre se l'utente apre la pagina dello storico potrà visualizzare le sue statistiche.		

<b>Test Case Riferimento</b>	TC-004 REQ-002	<b>Nome</b>	Accesso sbagliato
<b>Descrizione</b>	Bisogna tentare di accedere ad un account con delle credenziali sbagliate		
<b>Prerequisiti</b>	<ol style="list-style-type: none"> <li>1. Aprire il sito web</li> <li>2. Aprire la pagina di accesso</li> <li>3. Deve esistere un account</li> </ol>		
<b>Procedura</b>	<ol style="list-style-type: none"> <li>1. Inserire un nome utente e una password non validi</li> <li>2. Premere il tasto di accesso</li> </ol>		
<b>Risultati attesi</b>	Dovrà venir mostrato all'utente un messaggio d'errore che lo avvisa che le credenziali sono errate.		

<b>Test Case Riferimento</b>	TC-005 REQ-003	<b>Nome</b>	Registrazione di un nuovo gesto
<b>Descrizione</b>	Bisogna registrare un nuovo gesto		
<b>Prerequisiti</b>	<ol style="list-style-type: none"> <li>1. Aprire il sito web</li> <li>2. Eseguire l'accesso con un utente amministratore</li> <li>3. Aprire la pagina di registrazione</li> </ol>		
<b>Procedura</b>	<ol style="list-style-type: none"> <li>1. Scattare una foto con il gesto della mano visibile</li> <li>2. Inserire la posizione base del gesto</li> <li>3. Selezionare il gesto che si vuole registrare</li> <li>4. Inserire i termini con la rispettiva categoria e lingua</li> <li>5. Premere il tasto di invio</li> </ol>		
<b>Risultati attesi</b>	L'immagine inviata e verrà controllata la posizione della mano che poi verrà salvata all'interno del database. All'utente verrà mostrato un messaggio di successo.		

<b>Test Case Riferimento</b>	TC-006 REQ-003	<b>Nome</b>	Registrazione di un nuovo gesto senza selezionare tutti i dati
<b>Descrizione</b>	Bisogna provare a registrare un nuovo gesto non selezionando tutti i dati, per vedere come si comporta l'applicativo.		
<b>Prerequisiti</b>	<ol style="list-style-type: none"> <li>1. Aprire il sito web</li> <li>2. Eseguire l'accesso con un utente amministratore</li> <li>3. Aprire la pagina di registrazione</li> </ol>		
<b>Procedura</b>	<ol style="list-style-type: none"> <li>1. Caricare una foto con il gesto della mano visibile</li> <li>2. Inserire la posizione base del gesto</li> <li>3. <b>Non</b> selezionare il gesto che si vuole registrare</li> <li>4. Inserire i termini con la rispettiva categoria e lingua</li> <li>5. Premere il tasto di invio</li> </ol>		
<b>Risultati attesi</b>	Dovrà apparire un messaggio d'errore che comunica all'utente che deve inserire tutti i dati per registrare un nuovo gesto.		



<b>Test Case Riferimento</b>	TC-007 REQ-003	<b>Nome</b>	Registrazione di un nuovo gesto con una foto senza mano visibile
<b>Descrizione</b>	Bisogna registrare un nuovo gesto utilizzando una foto dove le mani non sono visibili, per vedere come si comporta l'applicativo.		
<b>Prerequisiti</b>	<ol style="list-style-type: none"> <li>1. Aprire il sito web</li> <li>2. Eseguire l'accesso con un utente amministratore</li> <li>3. Aprire la pagina di registrazione</li> </ol>		
<b>Procedura</b>	<ol style="list-style-type: none"> <li>1. Caricare una foto dove non si vede alcuna mano</li> <li>2. Inserire la posizione base del gesto</li> <li>3. Selezionare il gesto che si vuole registrare</li> <li>4. Inserire i termini con la rispettiva categoria e lingua</li> <li>5. Premere il tasto di invio</li> </ol>		
<b>Risultati attesi</b>	Dovrà apparire un messaggio d'errore che comunica all'utente che non è stata trovata alcuna mano all'interno dell'immagine.		

<b>Test Case Riferimento</b>	TC-008 REQ-003	<b>Nome</b>	Accesso alla pagina di registrazione con un account non amministrativo
<b>Descrizione</b>	Bisogna tentare di accedere alla pagina di registrazione dei gesti con un account normale o senza aver eseguito l'accesso.		
<b>Prerequisiti</b>	<ol style="list-style-type: none"> <li>1. Aprire il sito web</li> <li>2. Eseguire l'accesso con un utente non amministrativo</li> </ol>		
<b>Procedura</b>	<ol style="list-style-type: none"> <li>1. Provare ad accedere alla pagina di registrazione modificando l'URL a mano andando sulla pagina /insert.</li> </ol>		
<b>Risultati attesi</b>	Il sito web non dovrà permettere l'accesso alla pagina facendo un redirect sulla pagina iniziale.		

<b>Test Case Riferimento</b>	TC-009 REQ-004	<b>Nome</b>	Pagina di riconoscimento gesti
<b>Descrizione</b>	Bisogna controllare che il riconoscimento della mano sia possibile sia in tempo reale che tramite una foto.		
<b>Prerequisiti</b>	<ol style="list-style-type: none"> <li>1. Aprire il sito web</li> <li>2. Aprire la pagina di riconoscimento</li> <li>3. Deve essere registrato almeno un gesto</li> </ol>		
<b>Procedura</b>	<ol style="list-style-type: none"> <li>1. Avviare catture tramite la webcam</li> <li>2. Eseguire dei gesti salvati</li> <li>3. Chiudere la webcam</li> <li>4. Caricare una foto con all'interno un gesto</li> </ol>		
<b>Risultati attesi</b>	In entrambi i casi verranno riportati a schermo i termini legati al gesto.		

<b>Test Case Riferimento</b>	TC-010 REQ-004	<b>Nome</b>	Upload file formato non valido
<b>Descrizione</b>	Bisogna controllare che viene verificato il formato dei file caricati per il riconoscimento tramite immagine.		
<b>Prerequisiti</b>	<ol style="list-style-type: none"> <li>1. Aprire il sito web</li> <li>2. Aprire la pagina di riconoscimento</li> <li>3. Deve essere registrato almeno un gesto</li> </ol>		
<b>Procedura</b>	<ol style="list-style-type: none"> <li>1. Creare un file di testo</li> <li>2. Rinominarlo cambiando l'estensione in .png</li> <li>3. Caricarlo sul sito per il riconoscimento tramite immagine</li> <li>4. Confermare l'invio</li> </ol>		
<b>Risultati attesi</b>	Dovrà apparire un errore che avvisa l'utente che il file è in un formato non valido.		

<b>Test Case Riferimento</b>	TC-011 REQ-005	<b>Nome</b>	Prova sistema di allenamento
<b>Descrizione</b>	Bisogna testare il sistema di allenamento		
<b>Prerequisiti</b>	<ol style="list-style-type: none"> <li>1. Aprire il sito web</li> <li>2. Aprire la pagina di apprendimento</li> <li>3. Selezionare “Avvia allenamento”</li> <li>4. Deve essere registrato almeno un gesto</li> </ol>		
<b>Procedura</b>	<ol style="list-style-type: none"> <li>1. Eseguire il gesto mostrato</li> <li>2. Cliccare “Prossimo” per passare al gesto successivo</li> <li>3. Eseguire un gesto diverso da quello mostrato</li> </ol>		
<b>Risultati attesi</b>	<p>Quando si esegue il gesto mostrato dovrà apparire un messaggio che dice all'utente che il gesto è stato riconosciuto. Quando si passa al gesto successivo il messaggio dovrà sparire, apparire il nuovo gesto da imitare e ricomincerà il riconoscimento della mano. Se il gesto che esegue l'utente non corrisponde a quello che deve eseguire, verrà mostrato un messaggio all'utente che gli comunica che il gesto non è stato riconosciuto.</p>		

<b>Test Case Riferimento</b>	TC-012 REQ-006	<b>Nome</b>	Prova sistema di quiz
<b>Descrizione</b>	Bisogna testare il sistema di quiz		
<b>Prerequisiti</b>	<ol style="list-style-type: none"> <li>1. Aprire il sito web</li> <li>2. Aprire la pagina di apprendimento</li> <li>3. Selezionare “Avvia quiz”</li> <li>4. Deve essere registrato almeno un gesto</li> </ol>		
<b>Procedura</b>	<ol style="list-style-type: none"> <li>1. Mostrare il gesto corrispondente al termine mostrato</li> <li>2. Cliccare “Prossimo” per passare al gesto successivo</li> <li>3. Ripetere fino al termine del quiz.</li> </ol>		
<b>Risultati attesi</b>	<p>Quando si termina il quiz si dovrà vedere la pagina di resoconto del quiz con mostrate le statistiche del quiz.</p>		

<b>Test Case Riferimento</b>	TC-013 REQ-007	<b>Nome</b>	Riconoscimento gesto fallito
<b>Descrizione</b>	Bisogna vedere come si comporta l'applicativo quando non riconosce un gesto sia in tempo reale che dà un'immagine		
<b>Prerequisiti</b>	<ol style="list-style-type: none"> <li>1. Aprire il sito web</li> <li>2. Aprire la pagina di riconoscimento</li> <li>3. Deve essere registrato almeno un gesto</li> </ol>		
<b>Procedura</b>	<ol style="list-style-type: none"> <li>1. Avviare catture tramite la webcam</li> <li>2. Eseguire dei gesti non salvati</li> <li>3. Chiudere la webcam</li> <li>4. Caricare una foto rappresentante un gesto non salvato</li> <li>5. Inviare l'immagine.</li> </ol>		
<b>Risultati attesi</b>	Quando il programma non riesce a riconoscere il gesto durante il riconoscimento in tempo reale non deve mostrare nulla, in attesa che l'utente esegua un gesto corretto. Mentre se si carica una foto rappresentante un gesto non salvato verrà comunicato all'utente che il gesto non è stato riconosciuto.		

<b>Test Case Riferimento</b>	TC-014 REQ-008	<b>Nome</b>	Riconoscimento in real time
<b>Descrizione</b>	Bisogna verificare che il riconoscimento dei gesti avvenga in tempo reale tramite la webcam.		
<b>Prerequisiti</b>	<ol style="list-style-type: none"> <li>1. Aprire il sito web</li> <li>2. Aprire la pagina di riconoscimento</li> <li>3. Deve essere registrato almeno un gesto</li> </ol>		
<b>Procedura</b>	<ol style="list-style-type: none"> <li>1. Eseguire dei gesti ripresi dalla webcam</li> <li>2. Verificare che i possibili significati vengano mostrati quasi in real time</li> </ol>		
<b>Risultati attesi</b>	I significati dovranno apparire a schermo in tempo reale.		

<b>Test Case Riferimento</b>	TC-015 REQ-009	<b>Nome</b>	Salvataggio statistiche apprendimento
<b>Descrizione</b>	Bisogna verificare che, nel caso si sia registrati, le statistiche di apprendimento nella modalità quiz vengano salvate.		
<b>Prerequisiti</b>	<ol style="list-style-type: none"> <li>1. Aprire il sito web</li> <li>2. Eseguire l'accesso con un account esistente</li> <li>3. Deve essere registrato almeno un gesto</li> </ol>		
<b>Procedura</b>	<ol style="list-style-type: none"> <li>1. Avviare e completare il Quiz un paio di volte</li> <li>2. Aprire la pagina dello storico</li> </ol>		
<b>Risultati attesi</b>	Nello storico dovranno essere presenti i gesti eseguiti nei Quiz con un valore di precisione riferito all'esecuzione del gesto. Dovrà essere salvato solo il valore più alto per ogni gesto.		

<b>Test Case Riferimento</b>	TC-016 REQ-009	<b>Nome</b>	Accesso alla pagina dello storico senza eseguire l'accesso.
<b>Descrizione</b>	Bisogna tentare di accedere alla pagina dello storico senza aver eseguito l'accesso.		
<b>Prerequisiti</b>	<ol style="list-style-type: none"> <li>1. Aprire il sito web</li> </ol>		
<b>Procedura</b>	<ol style="list-style-type: none"> <li>1. Provare ad accedere alla pagina dello storico cambiando l'URL andando sulla pagina /history.</li> </ol>		
<b>Risultati attesi</b>	L'utente dovrà venir riportato alla pagina principale.		

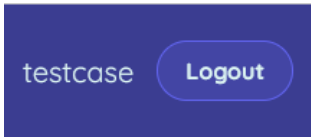
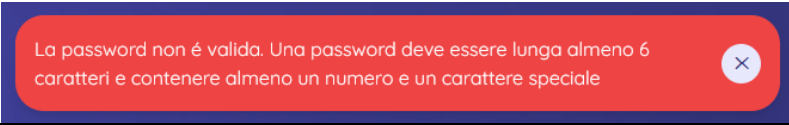
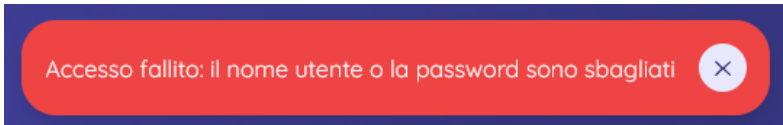
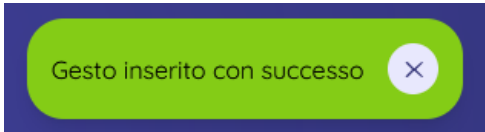
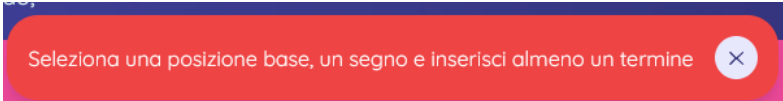
<b>Test Case Riferimento</b>	TC-017 REQ-010	<b>Nome</b>	Verifica sistema di log
<b>Descrizione</b>	Bisogna verificare che sul Back End sia presente un sistema di log che mostri diverse operazioni eseguite		
<b>Prerequisiti</b>	<ol style="list-style-type: none"> <li>1. Aprire il sito web</li> </ol>		
<b>Procedura</b>	<ol style="list-style-type: none"> <li>1. Registrare un nuovo account</li> <li>2. Aprire la pagina di riconoscimento</li> <li>3. Caricare e inviare un'immagine contenente un gesto valido</li> <li>4. Caricare e inviare un'immagine che non contiene alcun gesto</li> <li>5. Caricare e inviare un'immagine di un formato non valido</li> <li>6. Caricare e inviare un'immagine contenete un gesto non registrato</li> </ol>		
<b>Risultati attesi</b>	All'interno dei log del Back End dovranno essere presenti dei messaggi che mostrino le operazioni eseguite dall'utente.		

<b>Test Case Riferimento</b>	TC-018 REQ-011	<b>Nome</b>	Verifica sito responsive
<b>Descrizione</b>	Bisogna verificare che il sito web sia visualizzabile in modo ordinato anche da un dispositivo mobile.		
<b>Prerequisiti</b>	1. Aprire il sito web da un dispositivo mobile		
<b>Procedura</b>	1. Controllare pagina di registrazione 2. Controllare pagina di login 3. Eseguire accesso con un'utente amministratore 4. Controllare pagina di riconoscimento 5. Eseguire il riconoscimento sia in tempo reale che tramite una foto 6. Controllare pagina di apprendimento 7. Controllare la pagina di allenamento 8. Controllare e finire un quiz per controllare anche la pagina di resoconto 9. Controllare pagina dello storico 10. Controllare pagina di inserimento 11. Inserire un nuovo gesto		
<b>Risultati attesi</b>	Tutte le pagine dovranno visualizzarsi in modo ordinato.		

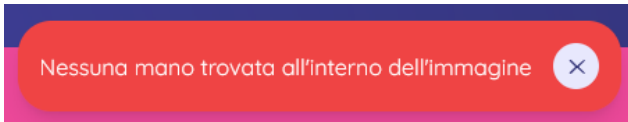

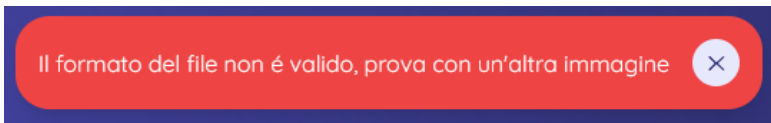

<b>Test Case Riferimento</b>	TC-019 REQ-012	<b>Nome</b>	Modifica margine d'errore
<b>Descrizione</b>	Bisogna verificare che la modifica del margine d'errore funzioni		
<b>Prerequisiti</b>	1. Aprire il sito web 2. Aprire la pagina di riconoscimento 3. Deve essere registrato almeno un gesto		
<b>Procedura</b>	1. Avviare catture tramite la webcam 2. Eseguire dei gesti salvati con il margine d'errore sul valore di default 3. Aumentare il margine d'errore 4. Diminuire il margine d'errore		
<b>Risultati attesi</b>	Più il margine diminuisce più il sistema di riconoscimento è preciso e quindi far riconoscere il gesto diventerà più difficile perché bisogna essere più precisi. Se invece si aumenta il margine d'errore il sistema sarà meno preciso e quindi il gesto viene riconosciuto anche se l'esecuzione dell'utente non è ottimale.		

<b>Test Case Riferimento</b>	TC-020 REQ-007	<b>Nome</b>	Verifica di riconoscimento
<b>Descrizione</b>	Bisogna valutare la precisione nel riconoscimento dei gesti. Per farlo verranno registrati diversi gesti e dopo verranno testati tutti. Con questo test si riescono a visualizzare i limiti dell'applicativo.		
<b>Prerequisiti</b>	<ol style="list-style-type: none"> <li>1. Aprire il sito web</li> <li>2. Eseguire l'accesso con un account amministratore</li> </ol>		
<b>Procedura</b>	<ol style="list-style-type: none"> <li>1. Aprire pagina di inserimento</li> <li>2. Registrare due segni per ogni posizione base (12 in totale) con almeno due termini per lingua ad ogni gesto</li> <li>3. Aprire pagina di riconoscimento</li> <li>4. Eseguire tutti e 12 i gesti</li> </ol>		
<b>Risultati attesi</b>	Valutare se l'applicativo riesce a registrare tutte le mappature e se il sistema di riconoscimento funziona per tutti i gesti indicando il valore più basso di margine d'errore applicato prima che il sistema cominci a non riconoscere più il gesto.		

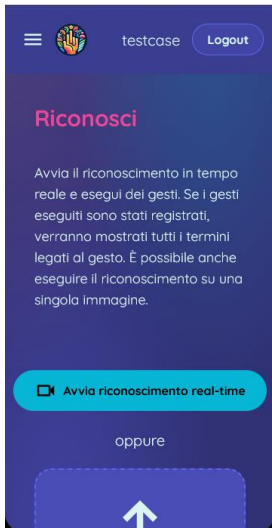
## 6.2 Risultati test



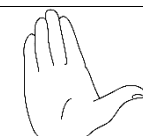
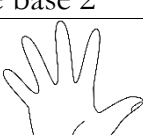
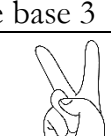
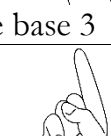
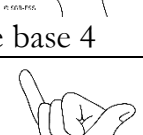

Test Case	Risultato ottenuto	Stato	Data
TC-001	<p>Quando si registra un nuovo account, se tutti i valori sono corretti, si viene portati alla pagina principale e in alto a destra è visualizzabile il nome utente.</p> 	Passato	17.05.2024
TC-002	<p>Inserendo dei valori non validi, non si viene portati alla pagina principale ma si svuotano i campi di testo e apparirà un messaggio d'errore in basso alla finestra che ricorda le convenzioni che devono avere nome utente e password.</p> 	Passato	17.05.2024
TC-003	<p>Quando si esegue l'accesso con dei valori validi di un utente già esistente si viene portati alla pagina principale e in alto a destra è sempre visibile il nome utente del profilo alla quale si ha fatto l'accesso</p>	Passato	17.05.2024
TC-004	<p>Quando si esegue con dei valori sbagliati i campi di testo vengono svuotati e in basso alla finestra apparirà un messaggio d'errore.</p> 	Passato	17.05.2024
TC-005	<p>Quando si registra un gesto con tutti i valori validi viene mostrato all'utente un messaggio di successo.</p> 	Passato	17.05.2024
TC-006	<p>Quando si prova a registrare un nuovo gesto non inserendo tutti i valori (per esempio lasciando via la foto del segno chi si sta registrando) verrà mostrato all'utente un errore e la registrazione non avverrà.</p> 	Passato	17.05.2024






TC-007	<p>Quando si prova a registrare un nuovo gesto, ma dall'immagine non viene rilevata alcuna mano, l'inserimento non avverrà e verrà però mostrato all'utente un messaggio d'errore.</p> 	Passato	17.05.2024
TC-008	<p>Se si prova ad accedere alla pagina di inserimento con un utente non amministrativo modificando l'URL verrà eseguito un redirect che riporta l'utente alla pagina principale.</p>	Passato	17.05.2024
TC-009	<p>Sia tramite la webcam che tramite un'immagine, se il gesto viene riconosciuto, vengono mostrati all'utente i termini legati al gesto.</p> 	Passato	17.05.2024
TC-010	<p>Quando si carica un file di testo rinominato con l'estensione .png viene mostrato all'utente un errore.</p> 	Passato	23.05.2024
TC-011	<p>Quando si avvia un allenamento viene mostrato il gesto da eseguire. Quando si esegue il gesto corretto in basso verrà mostrato un messaggio e il pulsante per andare avanti si abiliterà. Invece, quando il gesto eseguito non è corretto, verrà mostrato un messaggio diverso e il pulsante per andare avanti sarà disabilitato.</p> 	Passato	17.05.2024

TC-012	<p>Mentre si sta eseguendo il quiz non viene mostrato all'utente se sta eseguendo i gesti in modo corretto, viene fatto solo quando il quiz è finito. Alla fine del quiz viene mostrata una pagina di resoconto che mostra la precisione nell'esecuzione dei gesti.</p> 	Passato	17.05.2024
TC-013	<p>Quando si esegue un gesto non riconosciuto durante il riconoscimento in tempo reale non viene mostrato niente all'utente in attesa che esegua un gesto conosciuto. Mentre quando si carica un'immagine che ritrae un gesto non salvato viene mostrato all'utente un messaggio d'errore.</p> 	Passato	17.05.2024
TC-014	<p>Quando si avvia il riconoscimento in tempo reale i termini dei gesti vengono mostrati in tempo reale, cambiando appena si cambia gesto.</p>	Passato	17.05.2024
TC-015	<p>Quando si apre lo storico si possono vedere tutti i risultati migliori ottenuti nei quiz. I valori nello storico sono in ordine dalla precisione più alta alla più bassa.</p> 	Passato	17.05.2024
TC-016	<p>Se si prova ad accedere alla pagina dello storico senza eseguire l'accesso modificando l'URL verrà eseguito un redirect che riporta l'utente alla pagina principale.</p>	Passato	17.05.2024
TC-017	<p>Quando si eseguono delle operazioni sul sito, sul Back End verranno salvati nei log dei messaggi per visualizzare cosa è stato fatto dall'utente.</p> <pre>[INFO] - 127.0.0.1 - - [23/May/2024 09:14:40] Request for /api/user [INFO] - 127.0.0.1 - - [23/May/2024 09:14:51] Loaded data from database or recognition [INFO] - 127.0.0.1 - - [23/May/2024 09:14:59] Request for /api/recognize [INFO] - 127.0.0.1 - - [23/May/2024 09:14:59] Sign recognized, collecting data from database [INFO] - 127.0.0.1 - - [23/May/2024 09:14:59] Sign recognized, collecting data from database [INFO] - PERFORMANCE - [23/May/2024 09:14:59] Time for recognize sign: 42 ms [INFO] - 127.0.0.1 - - [23/May/2024 09:15:07] Request for /api/recognize [WARNING] - 127.0.0.1 - - [23/May/2024 09:15:07] No hand found while trying to recognize sign [INFO] - PERFORMANCE - [23/May/2024 09:15:07] Time for recognize sign: 19 ms [INFO] - 127.0.0.1 - - [23/May/2024 09:15:14] Request for /api/recognize [ERROR] - 127.0.0.1 - - [23/May/2024 09:15:14] File format is not valid [INFO] - 127.0.0.1 - - [23/May/2024 09:18:36] Request for /api/recognize [WARNING] - 127.0.0.1 - - [23/May/2024 09:18:36] No sign match with user's sign [INFO] - PERFORMANCE - [23/May/2024 09:18:36] Time for recognize sign: 47 ms</pre>	Passato	21.05.2024

TC-018	<p>Tutte le pagine si visualizzano in ordine in modo da poter utilizzare il sito anche da un dispositivo mobile.</p> 	Passato	17.05.2024
TC-019	<p>Se si cambia il margine d'errore durante il riconoscimento in tempo reale, più lo si abbassa e più il sistema faticherà a riconoscere il gesto essendo che più si abbassa il margine d'errore e più il gesto deve essere preciso. Mentre se si alza il margine d'errore il sistema sarà meno preciso e quindi riconosce i gesti anche se non sono veramente simili.</p>	Passato	17.05.2024
TC-020	<p>Il primo limite riscontrato è che registrando i due gesti per la prima posizione base, la mano non viene quasi mai riconosciuta, infatti sono riuscito a registrare soltanto un gesto invece che due per la prima posizione base. Questo problema è causato dai modelli Machine Learning essendo che provando a caricare le stesse foto che ho utilizzato sul mio sito sul sito ufficiale di Mediapipe, anche in quel caso la mano non veniva riconosciuta.</p> <p>Un altro problema riscontrato (non causato da me) è che sul sito <a href="https://www.sgb-fss.ch/signsuisse/it/assistente-di-ricerca-dal-segno">https://www.sgb-fss.ch/signsuisse/it/assistente-di-ricerca-dal-segno</a> è presente solo un gesto per la quinta posizione base che ha 2 termini in italiano. Quindi uno dei due gesti legati alla quinta posizione base ha solo 1 termine in italiano.</p> <p>A parte questi due errori, tutto il resto ha funzionato come previsto. Infatti tutti gli altri gesti sono stati registrati correttamente e tutti i gesti registrati vengono riconosciuti.</p> <p>Nella tabella sotto sono presenti i test dei gesti registrati con riportato se il gesto è stato riconosciuto e il margine d'errore minimo con il quale il sistema è riuscito a riconoscere il gesto.</p>	Quasi Passato	21.05.2024

Gesto	Stato	Margine minimo
 Posizione base 1	Riconosciuto	0.019
 Posizione base 2	Riconosciuto	0.009
 Posizione base 2	Riconosciuto	0.014
 Posizione base 3	Riconosciuto	0.016
 Posizione base 3	Riconosciuto	0.027
 Posizione base 4	Riconosciuto	0.019
 Posizione base 4	Riconosciuto	0.020
 Posizione base 5	Riconosciuto	0.026

 Posizione base 5	Riconosciuto	0.018
 Posizione base 6	Riconosciuto	0.037
 Posizione base 6	Riconosciuto	0.043

## 7 Consuntivo

Qua sotto è presente il Gantt consuntivo. Si possono notare delle differenze con il Gantt preventivo.

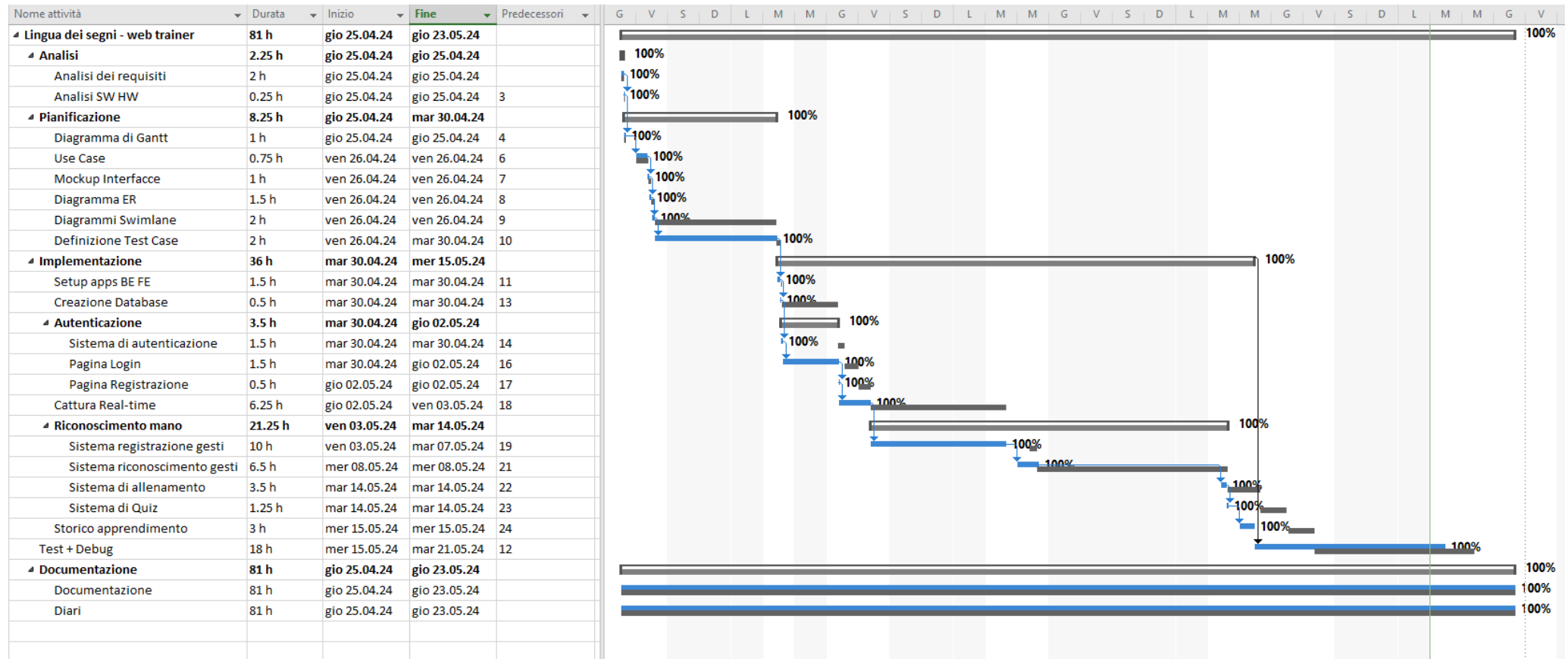


Figura 34 Diagramma di Gantt consuntivo

## 7.1 Differenze con la pianificazione

Come si vede dal diagramma di Gantt consuntivo, ci sono state parecchie differenze rispetto a quanto avevo pianificato. Questo mi ha permesso di avere più tempo da dedicare (6 ore in più) ai test e a fare debug.

La fase dove ho risparmiato più tempo è stata la fase di implementazione. Infatti, in totale, ho impiegato 9 ore in meno del previsto per implementare l'applicativo. Però non per tutte le attività della fase d'implementazione ho risparmiato tempo: per implementare il sistema di registrazione dei gesti ho impiegato 6 ore in più del previsto, probabilmente causato dal fatto che durante la pianificazione non avevo ancora bene in chiaro come doveva funzionare la registrazione dei gesti. A parte quest'attività, in quasi tutte le altre ho impiegato meno del previsto, essendo che tante funzionalità sapevo già come implementare.

Per la fase di analisi invece ho impiegato il tempo che avevo previsto, mentre per la pianificazione del progetto ci ho impiegato un ora e quarantacinque minuti in meno di quanto avevo previsto.

## 8 Conclusioni

Il progetto è stato concluso con successo. Infatti tutti i requisiti richiesti sono stati soddisfatti. L'applicativo è utilizzabile da chiunque sia su un PC che su un dispositivo mobile e può servire per chi necessita o vuole imparare i gesti del linguaggio dei segni.

### 8.1 Sviluppi futuri

A questo progetto si potrebbero fare delle modifiche come:

- Aggiungere la possibilità di registrare termini in altre lingue (come per esempio in tedesco)
- Aggiungere la possibilità di cambiare lingua del sito (scegliendo tra le lingue che si possono apprendere)
- Aggiungere la possibilità di scegliere una lingua per la modalità di Allenamento o Quiz, in modo da potersi allenare per una lingua specifica
- Aggiungere dei filtri all'interno dello storico
- Aggiungere la possibilità di modificare i gesti e i termini per l'amministratore
- Aggiungere la modifica del margine d'errore anche durante le modalità di apprendimento e durante il riconoscimento tramite una foto.
- Aggiungere per l'amministratore la gestione degli utenti in modo che possa rendere altri utenti degli amministratori
- Migliorare la scelta delle parole durante le modalità di apprendimento, essendo che per il momento è random
- Aggiungere una terza modalità di apprendimento, disponibile solo per chi ha eseguito l'accesso, dove vengono proposte solo le parole dove l'utente ha un valore di precisione basso salvato nello storico.

### 8.2 Considerazioni personali

Concluso questo progetto posso ritenermi soddisfatto del risultato che sono riuscito ad ottenere. Durante il progetto non ho riscontrato problemi nell'implementare delle funzionalità.

Grazie a questo progetto ho migliorato le mie conoscenze di Python e di Typescript. Anche se tutte le tecnologie che ho utilizzato le conoscevo già, ho comunque ampliato le mie conoscenze. Infatti ho approfondito l'utilizzo di Flask, Socket.IO, SQLAlchemy per creare un Back End in Python e ho ampliato le mie conoscenze di Sveltekit per il Front End. In generale mi sono trovato bene a lavorare con le tecnologie che ho scelto e non ho avuto troppe difficoltà a svolgere questo progetto.



## 9 Bibliografia

---

### 9.1 Sitografia

- <https://svelte.dev/>, *documentazione di Svelte*, 25.04.2024
- <https://kit.svelte.dev/>, *documentazione di Sveltekit*, 25.04.2024
- <https://www.skeleton.dev/>, *documentazione di Skeleton*, 25.04.2024
- <https://tailwindcss.com/>, *documentazione di Tailwind*, 25.04.2024
- <https://flask.palletsprojects.com/en/3.0.x/>, *documentazione di Flask*, 25.04.2024
- <https://flask-sqlalchemy.palletsprojects.com/en/3.1.x/>, *documentazione di SQLAlchemy*, 25.04.2024
- <https://www.sgb-fss.ch/signsuisse/it/assistente-di-ricerca-dal-segno>, *sito della Federazione Svizzera dei Sordi*, 25.04.2024
- [https://ai.google.dev/edge/mediapipe/solutions/vision/hand\\_landmarker](https://ai.google.dev/edge/mediapipe/solutions/vision/hand_landmarker), *documentazione Mediapipe*, 25.04.2024
- <https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow>, *documentazione Git flow*, 26.04.2024
- <https://socket.io/docs/v4/client-initialization/>, *documentazione Socket.IO-client*, 02.05.2024
- <https://icon-sets.iconify.design/>, *sito con icons*, 02.05.2024
- <https://uiverse.io/>, *loaders open source fatti in CSS*, 03.05.2024
- <https://stackoverflow.com/questions/1838873/visualizing-branch-topology-in-git>, *show git commit tree*, 23.05.2024

## 10 Glossario

Termine	Significato
Back End	Il Back End è la parte dell'applicazione che non si interfaccia con l'utente, quindi dove i dati vengono elaborati e, nel caso, salvati nel database.
Branch di Git	Una branch serve per isolare una parte di sviluppo in modo da non andare a modificare il progetto principale. Una volta che la funzionalità è funzionante in modo corretto si può unire la branch al progetto principale.
Certificato self-signed	Un certificato SSL self-signed è un certificato ritenuto non sicuro perché non è stato distribuito da una vera autorità di certificazione ma viene creato per essere utilizzato durante lo sviluppo di un applicativo.
File env	Un file env è un file contenente delle configurazioni dell'applicazione, delle variabili d'ambiente o delle informazioni sensibili.
Framework	Il framework è una struttura già esistente per facilitare lo sviluppo di un software in modo da non partire da zero.
Front End	Il Front End è la parte dell'applicazione che si interfaccia con l'utente, quindi la GUI e la raccolta dei dati che poi vengono inviati al Back End.
JSON	<b>Javascript Object Notation:</b> è un formato di rappresentazione dei dati basato sul linguaggio Javascript.
Modello di Machine Learning	Un modello di Machine Learning è una struttura che impara dalle informazioni dei dati forniti. Con questi dati il modello cerca di trovarci una sequenza e prova a farci delle predizioni. Nel caso di un modello pre-allenato si intende un modello alla quale sono già stati dati una grande quantità di dati e quindi dovrebbe riuscire ad ottenere dei risultati validi.
ORM	<b>Object Relational Mapping:</b> è una tecnica per poter creare utilizzare un database tramite la programmazione ad oggetti. Ogni tabella del database è una classe all'interno del codice.
Regular Expression	Una regular expression serve per vedere se una stringa soddisfa delle condizioni (per esempio contenere almeno un carattere speciale).

REST API	<b>Representational State Transfer API:</b> è uno stile di architettura di API basata su http dove per ogni richiesta bisogna inviare l'autenticazione (con username e password o con un token identificativo) e viene ritornata la risorsa richiesta in JSON o in XML.
Salting	Il salting è una tecnica che aggiunge una stringa random ad un dato sensibile prima di andare ad applicarci un algoritmo di hash. In questo modo non si può fare reverse engineering sul dato cryptato per ottenere il dato sensibile.
Socket.IO	Socket.IO è una libreria che permette una comunicazione bidirezionale basata su degli eventi tra un client e un server.
Token	Un token è una stringa di caratteri che viene utilizzata per autenticarsi senza dover utilizzare ad ogni richiesta le proprie credenziali.
Typescript	Typescript è un'estensione del linguaggio Javascript rendendolo un linguaggio tipizzato.
Virtual Environment	Un Virtual Environment in Python è un ambiente isolato dove si possono installare le dipendenze in modo da non installarle a livello globale per python.

## 11 Indice delle figure

Figura 1 Use Case .....	9
Figura 2 Diagramma di Gantt .....	10
Figura 3 Schema di rete.....	13
Figura 4 Schema logico del database.....	14
Figura 5 Pagina di registrazione .....	16
Figura 6 Pagina di accesso .....	16
Figura 7 Pagina di inserimento.....	17
Figura 8 Pagina di riconoscimento .....	18
Figura 9 Pagina di apprendimento.....	19
Figura 10 Pagina di allenamento / Quiz.....	19
Figura 11 Pagina dello storico .....	20
Figura 12 UML inserimento gesto.....	21
Figura 13 UML modalità allenamento .....	23
Figura 14 Struttura della cartella Back End.....	25
Figura 15 Punti di riferimento della mano .....	33
Figura 16 Struttura della cartella Front End .....	35
Figura 17 Palette dei colori.....	36
Figura 18 Definizione routes del sito.....	36
Figura 19 Barra di navigazione.....	36
Figura 20 Pagina di registrazione .....	37
Figura 21 Errore d'autenticazione .....	37
Figura 22 Riconoscimento in tempo reale.....	38
Figura 23 Pagina di allenamento .....	39
Figura 24 Pagina di risultato del quiz .....	39
Figura 25 Pagina dello storico .....	40
Figura 26 Messaggio storico vuoto.....	40
Figura 27 Pagina di inserimento (cattura immagine).....	41
Figura 28 Schermata di inserimento (dati).....	41
Figura 29 Selezione del gesto eseguito.....	42
Figura 30 Inserimento dei termini .....	42
Figura 31 Barra di navigazione da telefono.....	42
Figura 32 Branch di Git .....	45
Figura 33 Albero dei commit .....	45
Figura 34 Diagramma di Gantt consuntivo .....	62

## 12 Allegati

QdC	1_QdC/QdC-LPI24-GePe-Lorenzon Luca.pdf
Abstract	2_Abstract/Abstract.pdf
Diari di lavoro	4_Diari/
Prodotto	5_Sito_o_applicativo/
Schema database	7_Allegati/Database/Diagramma Db.png
Diagrammi di Gantt	7_Allegati/Pianificazione/
Diagrammi Swimlane	7_Allegati/Swimlane/Swimlane.svg
Mockup	7_Allegati/Mockup/
Schema di rete	7_Allegati/SchemaRete/SchemaRete.svg
Use Case	7_Allegati/UseCase/