

# **MEMORIZER**



**LORENZO GRASSI**

**PROGETTO PROGRAMMAZIONE AVANZATA – 15/01/2025**

# 1 Caso d'uso

Una delle parti principali dell'apprendimento di una lingua straniera è la memorizzazione delle parole, ma non sempre è facile tenere traccia di tutte le parole che si incontrano e cercare di ricordarle. Tra i metodi più comuni ci sono scrivere una lista oppure scrivere delle flash cards. *Memorizer* combina questi due metodi dando la possibilità all'utente di inserire tutte le parole che si intendono ricordare, e ha una funzionalità di quiz per aiutare e rendere più interattivo il processo di memorizzazione.

La semplicità del sistema e le impostazioni di personalizzazione lo rendono anche adattabile alle esigenze degli utenti: è possibile sia memorizzare direttamente le parole nella lingua straniera, oppure memorizzarne la traduzione, a seconda di cosa si inserisce nel campo *word* e *translation* (è necessario inserire in *word* quello che si vuole memorizzare). Anche se è pensato per memorizzare parole, e i termini presenti all'interno fanno riferimento a questo uso, è possibile infatti utilizzare *Memorizer* anche per memorizzare altre informazioni come date o nomi, interpretando a piacere i campi *word* e *translation* presenti.

## 2 Client

L'applicazione client è pensata per essere minimalistica ma funzionale e intuitiva: sono infatti presenti in tutte le schermate delle shortcut per rendere più veloce la navigazione e l'utilizzo delle pagine, come ad esempio le finestre di avviso che possono essere chiuse premendo *Enter*.

### 2.1 Home

La schermata *Home* contiene un bottone per accedere a tutte le schermate principali del progetto.

## MEMORIZER

Show Words

Add Words

Memorize

Settings

Le schermate sono anche accessibili tramite tastiera: *L* apre la lista delle parole, *+* apre la pagina per inserire nuove parole, *Enter* apre la funzione *Memorize*, mentre *S* è per aprire le impostazioni.

All'apertura della pagina viene verificata la connessione con il server: in caso di connessione fallita viene mostrato un messaggio di avviso. Lo stesso messaggio viene mostrato anche quando si prova ad aprire la lista delle parole oppure la funzione *Memorize* quando si è disconnessi.

## 2.2 Word List

Word List è la pagina che consente di visualizzare tutte le parole inserite e di modificarle e cancellarle. Per ogni parola, oltre ai campi *word*, *description* e *translation* sono presenti altre tre colonne che ricordano i progressi dell'utente: *quizzed* rappresenta il numero di volte che la parola è stata chiesta nel quiz, *guessed* indica quante volte è stata indovinata, mentre *confidence*, che viene calcolata come  $\text{guessed} / (\text{quizzed} + 1.5)$  indica quanto l'utente si ricorda la parola. Questo valore verrà utilizzato da *Memorize* (dettagli più avanti).

[< BACK](#)

### WORD LIST

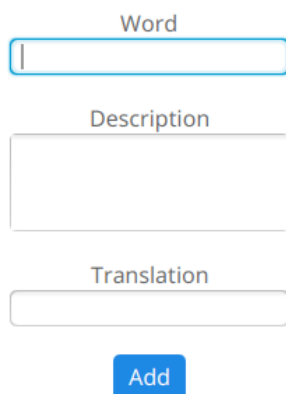
Word	Description	Translation	Confidence	Quizzed	Guessed	
Apple	A round fruit with red, green,...	Mela	56.3%	50	29	^
Ball	A round object typically used ...	Palla	47.5%	49	24	
Book	A set of written or printed pa...	Libro	2.2%	44	1	
Cat	A small domesticated carnivo...	Gatto	25.3%	38	10	
Clock	A device for measuring and d...	Orologio	83.2%	85	72	
Desk	A piece of furniture with a flat...	Scrivania	67.2%	58	40	
Dog	A domesticated carnivorous ...	Cane	72.0%	11	9	
Egg	An oval object laid by female ...	Uovo	77.6%	41	33	
Elephant	A large mammal with a trunk ...	Elefante	56.9%	67	39	
Fan	A device with rotating blades ...	Ventilatore	80.3%	62	51	
Flower	The seed-bearing part of a pl...	Fiore	14.0%	27	4	v

Le righe della tabella sono interattive: il click col tasto destro apre un menù contestuale con le opzioni per visualizzare i dettagli, modificare, aggiungere e cancellare la parola selezionata, e un'opzione per aggiungere una nuova parola senza dover aprire la pagina dedicata.

Anche qua sono presenti shortcut: *Enter* apre i dettagli, *Delete* elimina, doppio click col sinistro per modificare, e *+* per aggiungere una parola nuova.

## 2.3 Edit Word e Add Word

### ADD WORD



The 'Add Word' form consists of three input fields stacked vertically. The first field is labeled 'Word' and contains a single vertical bar cursor. The second field is labeled 'Description' and is empty. The third field is labeled 'Translation' and is empty. Below these fields is a blue button labeled 'Add'.

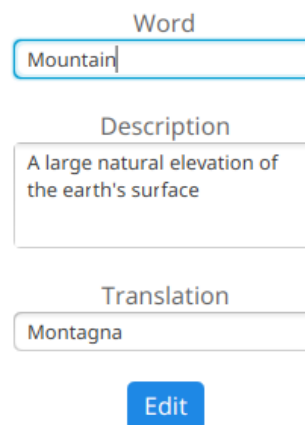
Le schermate *Edit Word* e *Add Word*, molto simili, presentano un form per inserire o modificare *word*, *description* e *translation* di una parola. Le schermate sono navigabili interamente tramite tastiera: è possibile usare *Tab* per spostarsi tra i vari campi e *Enter* per confermare e inserire o modificare la parola.

Le schermate *Edit Word* e *Add Word* si chiudono da sole una volta che la parola è stata modificata o inserita quando aperta tramite *Word List*, mentre in caso

in cui si acceda a *Add Word* tramite la schermata *Home* i campi si svuotano e la schermata rimane aperta per velocizzare l'inserimento di più parole consecutivamente.

In caso di mancata connessione viene visualizzato un messaggio di avviso.

### EDIT WORD



The 'Edit Word' form is similar to the 'Add Word' form but includes a 'Word' field with the text 'Mountain' entered. The 'Description' field contains the text 'A large natural elevation of the earth's surface'. The 'Translation' field contains the text 'Montagna'. Below these fields is a blue button labeled 'Edit'.

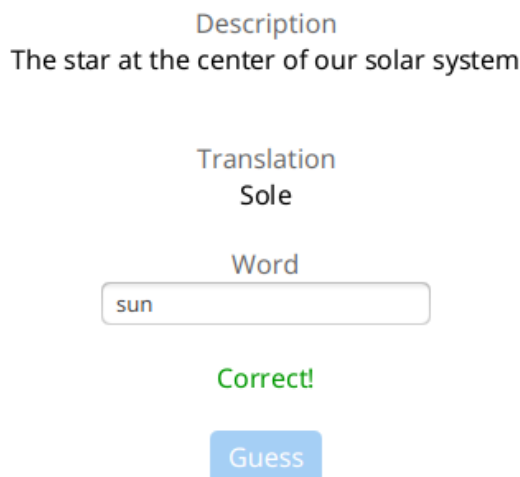
## 2.4 Memorize

La schermata *Memorize* aiuta l'utente a memorizzare le parole tramite un quiz interattivo. L'applicazione mostra la *description* e *translation* di una parola, e viene chiesto all'utente di inserire la parola. Una volta che l'utente clicca su *guess* oppure preme *Enter* il sistema visualizza un feedback che mostra se la parola inserita è corretta o no. Alla fine del quiz viene visualizzato un messaggio che riporta il numero di parole indovinate, e un voto per la prestazione dell'utente.

La lunghezza del quiz è personalizzabile, come anche quali

< BACK

### MEMORIZE



The 'Memorize' form displays a 'Description' field with the text 'The star at the center of our solar system'. Below it is a 'Translation' field with the text 'Sole'. The 'Word' field contains the text 'sun'. Below the 'Word' field is a green text label 'Correct!' and a blue button labeled 'Guess'.

parole l'utente vuole che vengano testate.

Quando si esce dal quiz i risultati vengono inviati al server per memorizzare i progressi, in caso di disconnessione viene visualizzato un messaggio di avviso.

## 2.5 Settings

È presente infine una schermata per modificare le impostazioni.

[< BACK](#)

### SETTINGS

☒ Show a confirmation box when deleting words

Quiz length:  words

☒ Always prioritize memorizing the words you know the least

Load data

Save

Le configurazioni presenti sono la possibilità di visualizzare un messaggio di conferma per la cancellazione delle immagini, la lunghezza del quiz e quali parole vengono testate: se si seleziona l'ultima checkbox *Memorizer* mostrerà le parole con *confidence* più bassa in modo da impararle prima, altrimenti vengono scelte parole casualmente. Le checkbox sono selezionabili e deselectionabili da tastiera tramite la pressione dei tasti *Y* e *N*, mentre *Enter* salva e chiude la pagina. Per navigare tra le varie impostazioni è possibile premere *Tab*.

Le impostazioni sono salvate in un file locale, non sono inviate al server. Se il file non è presente viene creato con le impostazioni di default.

*Load Data* serve per inizializzare il database e caricarci dei dati di esempio per verificare il funzionamento del programma: questo sovrascrive qualsiasi dato precedentemente salvato.

## 3 Server

Il server è relativamente semplice e si occupa della connessione col database e

dell'esecuzione delle query. Il server espone i seguenti endpoint:

<b>GET /words</b>	ritorna tutte le parole, compresi i valori di <i>guessed</i> e <i>quizzed</i>
<b>DELETE /word</b>	prende una stringa <u>word</u> come parametro e cancella la parola che ha quella stringa nel campo <i>word</i>
<b>POST /word</b>	aggiunge una parola, che deve essere passata nel body, compresa di <i>word</i> , <i>description</i> , e <i>translation</i>
<b>POST /quiz-result</b>	prende un parametro <u>guessed</u> , e un array di stringhe nel body, e aggiorna il database aumentando di uno il campo <i>quizzed</i> , e anche il campo <i>guessed</i> se <u>guessed</u> è true, per ogni parola nell'array (confronta la stringa con il campo <i>word</i> )
<b>PATCH /word</b>	prende un parametro <u>originalWord</u> e aggiorna la parola che ha come <i>word</i> <u>originalWord</u> con i dati di una nuova parola che devono essere passati nel body
<b>GET /training-data</b>	prende due parametri: un intero <u>n</u> e un booleano <u>random</u> , e ritorna <u>n</u> parole a caso se <u>random</u> è true, altrimenti ritorna le <u>n</u> parole con <i>confidence</i> più bassa
<b>POST /initialize-database</b>	inizializza il database e inserisce i dati di prova a partire da uno script SQL su un file
<b>GET /test</b>	serve per testare la connessione, ritorna solo un messaggio di ok

## 4 Dettagli tecnici e di implementazione

Il progetto è stato implementato per Java 17.

### 4.1 Comunicazione e database

Il client e il server comunicano in HTTP sulla porta 9696. Il server inoltre comunica col DBMS MySql sulla porta 3306.

Le credenziali per accedere a MySql sono username: root, password: root. Il server usa il database di nome 620944 che è composto da una singola tabella *Words* che ha cinque campi: *word*, *description*, *translation*, *quizzed*, *guessed*. Le colonne *quizzed* e *guessed* devono essere settate come automaticamente zero, perché il server esegue l'inserimento delle parole tramite il *WordRepository* che si basa sulla classe *Word*, che non ha questi campi. Questo è fatto per non dover inserire sempre quei dati nella richiesta HTTP, che sarebbero sempre zero. Il server assume che il database sia impostato correttamente: è importante eseguire un'inizializzazione del database prima di usare *Memorizer* tramite il bottone *Load Data* sul client, che esegue lo script SQL che crea correttamente la tabella e ci inserisce dei dati di prova.

## 4.2 Implementazione Client

Il client, implementato con la piattaforma JavaFX, ha una classe *Controller* per ogni schermata dell'applicazione, che gestisce gli oggetti grafici e la logica. La classe *ServiceRequest*, basata su *HttpClient*, rende molto semplice inviare le richieste HTTP al server, gestisce tutti gli aspetti di connessione, e lancia un'eccezione *DisconnectedException* in caso di errore di connessione. Le richieste vengono eseguite tutte su thread aggiuntivi per non bloccare l'interfaccia grafica, e il client mostra graficamente lo stato di attesa e blocca le azioni dell'utente per evitare effetti indesiderati. Infine la classe *Window* permette la creazione di finestre e il caricamento di nuove schermate sulla finestra in corso, con la possibilità di passare una funzione al caricatore delle pagine per trasferire parametri al *Controller* della nuova pagina. Questa classe viene usata anche per creare messaggi di avviso e di conferma.

## 4.3 Implementazione Server

Il server è basato sul framework Spring e gestisce le richieste HTTP tramite la classe *ServiceController*. L'interazione col database è fatta attraverso le interfacce *WordWithInfoRepository* e *WordRepository* che accedono alla stessa tabella ma con la possibilità di ottenere tutti o solo alcuni campi dei record, per avere oggetti più piccoli quando i dati aggiuntivi non sono necessari. I metodi più articolati dei repository sono implementati come JPQL query.

# 5 Utilizzo di AI durante l'implementazione

Durante l'implementazione del progetto è stato usato lo strumento ChatGPT-4o per velocizzare e rendere più efficiente il debugging, e occasionalmente anche la scrittura del codice. I prompt utilizzati sono del tipo: "I wrote a JavaFX application but it crashes when I open this view, here's the error message: [ERROR MESSAGE] and here's the fxml and controller class code [CODE]. Can you help me fix it?"

L'AI è anche stata utilizzata per la popolazione del database con questo prompt: "[SQL TABLE DEFINITION] I have this SQL table. Fill it with a few dozen rows where *word* is a word in English, *description* is the definition of the word and *translation* is the Italian translation; *quizzed* and *guessed* should be random integers but *quizzed* is always  $\geq$  *guessed*."

The screenshot shows a web browser window with a ChatGPT chat interface. The user's message is in Italian, asking for help with a JavaFX application error. The user provides a Java code snippet and an SQL query. The AI's response explains the issue: a case-sensitivity problem in the database table name. The AI suggests explicitly specifying the table name in the @Table annotation.

MySQL Driver Error Fix x +

https://chatgpt.com/c/677c0a02-4314-...

ChatGPT

java

```
@Query("SELECT w FROM Word w")
List<Word> findAllWords();
```

This will produce a cleaner query in the logs:

```
sql
SELECT word, description, translation FROM Words
```

okay the problem is my table's name is Words and the query says words (lowercase w)

Ah, got it! This happens because Hibernate typically assumes that table and column names are lowercase unless explicitly specified. If your table is named `Words` (with an uppercase `W`) and Hibernate is querying `words` (lowercase), it leads to the issue you're seeing.

**Fixing the Case-Sensitivity Issue**

1. Explicitly Specify the Table Name

Update the `@Table` annotation in your `Word` entity to explicitly set the table name as `Words` (case-sensitive):

java

Message ChatGPT

ChatGPT can make mistakes. Check important info.