

UNIVERSITÀ DEGLI STUDI DI BARI ALDO MORO

DIPARTIMENTO DI INFORMATICA

CORSO DI LAUREA IN INFORMATICA

RELAZIONE TECNICA IN

SISTEMI AD AGENTI

EMOGAME

**Relatore:**

Prof. Berardina de Carolis

**Gruppo composto da:**

Andriani Claudio e Coppolecchia Dario

Anno Accademico 2021-2022

# 1. INTRODUZIONE

**Idea di progetto**

Per la realizzazione del caso di studio, abbiamo realizzato un software che ha come obiettivo finale quello di essere di supporto ai terapisti per il riconoscimento e l’analisi delle emozioni dei pazienti dati degli stimoli, questo viene effettuato mostrando al paziente diversi media come immagini, video, testi o audio, scelti da un terapista, in modo tale da cogliere le emozioni provate dal sottoposto attraverso il riconoscimento delle espressioni del volto. Il progetto consiste in un full-stack website che permette ai terapisti di:

1. Aggiungere nuovi account terapisti
2. Gestire gli stimoli
3. Gestire i giochi
4. Avviare un gioco

Durante la partita viene registrato il volto del paziente e viene anche generato un CSV che potrà essere salvato in locale sul dispositivo, per poter effettuare l’analisi in un secondo momento dagli addetti.

# 2. LINGUAGGI E LIBRERIE SCELTE

**Linguaggi front-end e librerie**

I linguaggi scelti per la realizzazione dell’interfaccia utente per questo caso di studio sono:

* HTML5 / JSX
* CSS3
* JavaScript

La UI è stata realizzata utilizzando il framework React, includendo le librerie:

* axios, che permette di astrarre le AJAX request;
* react-Webcam, che fornisce il componente per il supporto alla webcam del dispositivo
* react-CSV, che fornisce i componenti necessari per l’utilizzo dei file CSV

**Linguaggi back-end e librerie**

Per quanto riguarda il lato back-end è stato utilizzato come WebServer node.js che per mette di gestire il lato server tramite Javascript. Le librerie utilizzate sono:

* express, utilizzato per gestire le richieste da e verso il front-end
* express-fileupload, adoperato per poter caricare dei file sul client
* cors, definisce un metodo con cui le applicazioni Web dei clienti caricate in un dominio possono interagire con le risorse situate in un dominio differente
* get-video-duration, permette di ottenere la durata di un video selezionato
* get-audio-duration, permette di ottenere la durata di un file audio selezionato
* mysql2, fornisce le funzioni per poter interagire con un database MySQL
* js-sha256, fornisce la funzione per poter applicare l’algoritmo di crittografia SHA256 ad un dato

**Linguaggio per il database**

Il lato server è stato gestito tramite il DBMS (Database Management System) MySQL.

# 3. STRUTTURA DELLE CARTELLE

Il progetto, reperibile su [GitHub](https://github.com/ClaudioAndriani/EmoGame), è suddiviso in tre cartelle principali:

* backend: per la gestione del server
* database: contente il modello e lo script sql per la creazione del DB
* UI: contente tutti i componenti per la realizzazione dell’interfaccia utente

Immagine che contiene testo

Descrizione generata automaticamente

Immagine che contiene testo

Descrizione generata automaticamenteAll’interno della cartella ‘backend’ ci sono le seguenti cartelle/file:

* node\_modules (non presente nella repo)
* stimuli: contiene tutti gli stimoli caricati dagli utenti (ne sono presenti alcuni di default per testing)
* package-lock.json e package.json: contengono le informazioni del server node
* server.js: è lo script del server Express

Nella cartella ‘database’ vi sono i seguenti file:

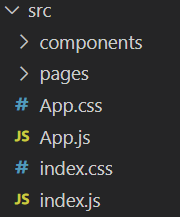
* Immagine che contiene testo, arancia

  Descrizione generata automaticamenteDBCreationScript.sql: lo script per la creazione del DB
* EmoDBModel.mwb: il modello del DB modificabile tramite mySQL Workbench

Immagine che contiene testo

Descrizione generata automaticamenteL’ultima sottocartella principale è suddivisa nel seguente modo:

* node\_modules (non presente nella repo)
* public: contiene gli asset e il file ‘index.html’ che è la pagina principale del sito
* src: contenente tutti i componenti dell’interfaccia
* viste interfaccia: contiene la prima versione delle bozze di tutte le pagine dell’interfaccia utente
* gitignore: indica i file da ignorare quando si effettua un push sulla repo di GitHub
* package-lock.json e package.json: contengono le informazioni del progetto React

La cartella ‘src’ contente tutti i componenti è strutturata in questo modo:

* components: che a sua volta è suddivisa in tre cartelle contenenti i componenti principali del progetto: manage\_games, manage\_stimulus e play\_game
* pages: contiene tutti gli script delle pagine realizzate
* app.css: il file css contenente gli script dello stile del sito
* app.js: il componente principale che gestisce il routing delle diverse pagine
* index.css: contiene lo script di stile di default principale
* index.js: lo script della pagina principale del sito

# 4. UI: Pagine principali

Immagine che contiene testo

Descrizione generata automaticamenteLo script App.js effettua il routing alle seguenti pagine:

* AccessDenied.js: contiene il messaggio di errore nel caso in cui un utente non abbia i permessi per accedere altre pagine
* AddTherapist.js: contiene il form e logica per la registrazione di un nuovo account terapista
* AnalyzeGame: contiene la base per lo sviluppo futuro dell’analisi dei risultati dei giochi
* ErrorPage.js: contiene il messaggio di errore se cerca di accedere ad un URL non esistente
* Header.js: contiene la componente che gestisce l’header dell’interfaccia utente, cioè il logo del sito e la NavBar per accedere alle varie pagine del sito in modo rapido
* Home.js: componente che permette di accedere alle varie sezioni del sito
* LoginModal.js: contiene il form per il login di un account terapista e permette di accedere al sito anche come paziente (con permessi limitati)
* ManageGames.js: permette di accedere alle sottosezioni per la gestione dei vari giochi presenti nel server
* ManageStimulus.js: permette di accedere alle sottosezioni per la gestione dei vari stimoli presenti nel server
* PlayGame.js: pagina in cui si gioca ad un gioco presente nel server

Il paziente verrà indirizzato alla pagina PlayGame.js e quindi non avrà i permessi per accedere alle altre sezioni del sito.

# 5. UI: Componenti – manage\_stimulus

Immagine che contiene testo

Descrizione generata automaticamente- AddNewStimulus.js: contiene il form e la logica per all’aggiunta di un nuovo stimolo (eventualmente anche una categoria) sul server

- DeleteStimulus.js: il componente che permette la cancellazione di stimoli precedentemente caricati sul server

- PlayStimulus.js: permette di visualizzare lo stimolo selezionato

- Stimulus.js: usato in StimulusGroup.js per rappresentare e gestire un singolo stimolo

- StimulusGroup.js: rappresenta un raggruppamento dei vari stimoli, utilizzato in VisualizeStimulus.js, VisualizeGames.js e SelectGameVisualizationMenu.js

- StimulusMenu.js: permette di accedere alle sottosezioni per la gestione degli stimoli

- VisualizeStimulus.js: permette di visualizzare gli stimoli presenti nel server suddivisi in varie categorie, da riprodurre o cancellare

# Immagine che contiene testo Descrizione generata automaticamente6. UI: Componenti – manage\_games

* DeleteGamesModal.js: modale che chiede conferma all’utente di un eventuale cancellazione di un gioco dal server
* EditGame.js: permette di modellare la struttura di un gioco (creato al momento o precedentemente esistente), nello specifico la durata dei singoli stimoli, l’ordine di essi ed eventualmente aggiungerne o rimuoverne alcuni.
* GamesMenu.js: : permette di accedere alle sottosezioni per la gestione dei giochi
* GStimulusRow.js: componente utilizzato in EditGame.js per rappresentare un singolo stimolo e permette di modificarne l’indice, la durata e aggiungere uno stimolo in coda a questo o di rimuoverlo dal gioco
* VisualizeGames.js: permette di visualizzare l’elenco dei giochi presenti nel server, mostrando il nome del gioco e i suoi stimoli, inoltre, permette, a seconda della sezione in cui viene aperto, permette di eseguire le seguenti operazioni:
  + aggiunta
  + modifica
  + cancellazione

di un gioco.

# Immagine che contiene testo Descrizione generata automaticamente7. UI: Componenti – play\_games

* PlayGamesStimulus.js: componente che gestisce la ‘partita’, quindi, permette all’utente di visualizzare in sequenza i vari stimoli del gioco presenti nel server, così impostato dai terapisti (sequenza e durata dei singoli stimoli)
* SaveMatch.js: permette di salvare l’eventuale registrazione della webcam e il CSV generato con i relativi dati di ogni stimolo presente nel gioco
* SelectGameVisualizationMenu.js: permette la scelta di un gioco presente nel server di cui effettuare la partita, inoltre ha delle opzioni che possono essere spuntate come la possibilità di registrare la webcam del sottoposto e di mostrare nome dello stimolo e del gioco durante la partita.

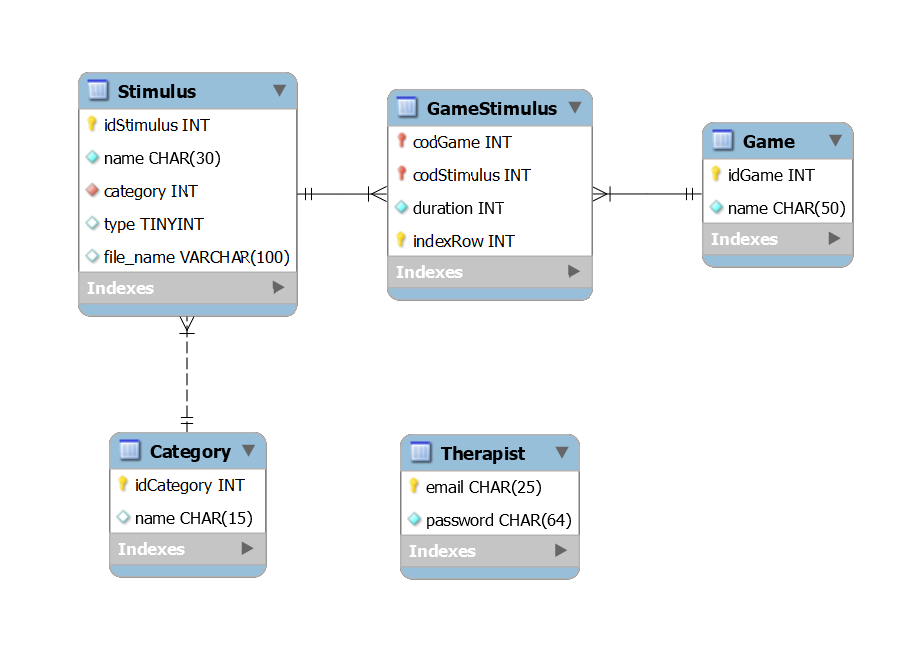
# 8. Server: funzionalità

Il server permette di effettuare le seguenti operazioni:

* “/add\_stimulus”: permette l’aggiunta di uno stimolo tramite richiesta del client all’interno del server, se non vi è lo stesso stimolo già presente nel server.
* “/is\_stimulus\_name\_in\_db”: effettua il controllo se un nome di uno stimolo inviato dal client è presente nel database
* “/check\_register\_therapist”: effettua il controllo durante il tentativo di registrazione di un terapista sull’email inserita, controllando se è presente o meno
* “/register\_therapist”: permette di registrare un nuovo account terapista l’e-mail non è stata già utilizzata in precedenza
* “/check\_login”: effettua il controllo sul tentativo di login da parte dell’utente controllando la presenza delle credenziale inserite nel database
* “/get\_stimulus\_categories”: restituisce l’elenco dei nomi delle categorie presenti nel database
* “/add\_stimulus\_category”: permette l’aggiunta di una nuova categoria, se non vi è già presente all’interno del database
* “/get\_all\_stimulus”: permette di ottenere l’elenco degli stimoli e il nome della categoria a cui appartengono
* “/get\_stimulus\_file”: permette di ottenere un file binario nella cartella *stimuli* dato un percorso dal client
* “/get\_stimulus\_text”: permette di ottenere un file di testo nella cartella *stimuli* dato un percorso dal client
* “/is\_gameName\_already\_present”: controlla se il nome dato dal client per un determinato gioco sia presente nel database
* “/add\_update\_game”: permette l’aggiunta o l’aggiornamento di un game all’interno del database
* “/delete\_stimuli”: permette la cancellazione di uno o più stimoli dal database il nome
* “/get\_stimulus\_duration”: ottiene la durata di un determinato stimolo all’interno del database
* “/get\_all\_games”: invia al client tutti i giochi e i suoi attributi presenti nel database
* “/get\_game\_stimulus\_rows”: permette di ottenere un singolo gioco e i suoi stimoli presenti nel database
* “/delete\_game”: effettua la cancellazione di un determinato gioco presente nel database

# 9. Database

La struttura del database è stata definita il seguente modello E/R:



La tabella GameStimulus reifica la relazione molti a molti delle tabelle Stimulus e Game, in cui la chiave primaria è una chiave composta da codGame, codStimulus e indexRow che sono le rispettive chiavi primarie più la riga dell’indice dello stimolo all’interno del gioco (inclusa per permettere allo stesso stimolo di essere presente più volte nello stesso gioco). La tabella Therapist non ha alcuna relazione, ma potrebbe essere aggiunta se si vuole estendere il progetto tenendo traccia di quale terapista effettui operazioni sul database. Stimulus ha una relazione di molti a uno con Category, che uno stimolo può far parte di una singola categoria ma una categoria può avere all’interno più stimoli; una possibile estensione potrebbe essere quella di modificare la relazione in un molti a molti permettendo ad uno stimolo di corrispondere a più categorie contemporaneamente.