Relazione

Tema e sezioni principali

Il sito creato è una libreria cinematografica in cui l'utente può, dopo essersi registrato o dopo aver effettuato l'accesso, informarsi sulle caratteristiche di un film quali trama, attori e i loro ruoli, direttore, genere narrativo, un'area commenti e un campo in cui l'utente può lasciare i suoi commenti.

Il sito è diviso in 4 macro aree:

- La pagina di login/registrazione nella quale l'utente può registrarsi al sito oppure effettuare il login
- Una bacheca nella quale sono esposte le locandine dei film che una volta cliccate rimandano alla pagina del film stesso
- La pagina del film selezionato nella quale sono presenti tutte le informazioni del film l'area commenti e l'area in cui scrivere i nuovi commenti che verranno subito visualizzati
- La pagina profilo utente in cui sono presenti i suoi dati e tutti i commenti fatti dall'utente raggruppati per film

Funzionalità

- Login: è costituita da due sezioni principali: la sezione registrazione (6 campi da compilare) e la sezione login (3 campi da compilare). In entrambi i casi i dati vengono acquisiti e validati. Nel caso in cui un utente già registrato cerchi di registrarsi nuovamente con gli stessi dati, gli verrà chiesto di utilizzare la funzione di login. Nel caso opposto ossia di un utente che cerca di acceder senza essersi prima registrato gli verrà chiesto di registrarsi. Se uno dei due procedimenti va a buon fine si viene reindirizzati sulla pagina rancid.php.

- Rancid: è costituita dal banner, titolo di benvenuto o bentornato a seconda del tipo di login eseguito, un link alla propria pagina profilo e infine la bacheca con tutti i film del sito.

- Film: una volta selezionata e cliccata la locandina di un film si viene reindirizzati alla pagina del film che è costituita da: titolo, valutazione, trama, area commenti, area dove poter scrivere il proprio commento, un link per tornare alla home e infine sulla destra una colonna con la locandina del film contenente le informazioni relative .

- Profilo: divisa in due sezioni: una centrale dove si possono visualizzare tutti i propri commenti raggruppati per film e un link per tornare alla home; una laterale dove sono presenti le informazioni dell'utente più un tasto per modificare i campi genere e data di nascita.

- Logout: l'utente si scollega dal sito, viene distrutta la vecchia sessione con tutti i suoi relativi campi e ne viene fatta partire una nuova e si viene reindirizzati alla pagina di login

Caratteristiche

- Login : i dati vengono acquisiti tramite due funzioni javascript distinte a seconda che si sia cliccato il pulsante "Registrati" o il pulsante "Accedi". In entrambi i casi si controlla la validità dei campi (campi significativi e data valida altrimenti compaiono opportuni messaggi di errore e se i controlli vanno a buon fine i dati vengono mandati tramite metodo POST a verify_login.php la quale si occupa di acquisire i dati e di contattare il database:

    - Registrazione: stabilita la connessione al database viene eseguita una query che ritorna gli utenti associati a quei dati: se il conteggio delle righe è uguale a zero allora l'utente è effettivamente nuovo al sito perciò (dopo aver cryptato la password), tutti i dati vengono inseriti nel database e viene creato un oggetto json con un campo registrazione uguale a true altrimenti il campo viene messo a false. Lato javascript si controlla questo valore e:
        - True => si viene reindirizzati a Rancid.php
        - False => viene stampato a video che l'utente è già registrato

    - Accesso: stabilita la connessione al database viene eseguita una query che ritorna l'utente associato a nome e cognome inseriti, se il numero di righe è zero l'utente non si è ancora registrato e si setta il campo vuoto a true dell'oggetto json, altrimenti si verifica che la password sia corretta e si setta il campo accesso dell'oggetto json a true . Lato javascript si controlla questo valore e:
        - Vuoto => utente non registrato
        - True => si viene reindirizzati a Rancid.php
        - False => viene stampato a video password errata

- Rancid: è la bacheca dove sono esposti tutte le locandine cinematografiche del sito. Si controlla che l'utente si sia loggato dopodiché stabilita la connessione al database

si esegue una query dalla quale si ricavano tutti i nomi dei film e tramite i nomi si ricavano le locandine dei film.

- Film: si controlla che l'utente si sia loggato e si carica lo scheletro della pagina. Una volta caricata l'intera pagina film.js si occupa di riempirla con tutti i suoi dati: vengono eseguite 6 funzioni per recuperare il rank, la trama, i commenti, il genere, il direttore e gli attori. Tutte queste funzioni comunicano tramite metodo POST con la pagina ajax.php la quale risponde con oggetti json. Un'ulteriore funzione si occupa di pubblicare i commenti lasciati dagli utenti e di registrarli nel database.

- Profilo: si controlla che l'utente si sia loggato dopodiché si carica lo scheletro della pagina. Una volta caricata la pagina si recuperano dal database, tramite due funzioni javascript , le informazioni dell'utente più tutti i suoi commenti raggruppati per film. Inoltre è presente un pulsante per modificare i dati: se i nuovi dati sono tutti validi, si aggiorna il database e la pagina, altrimenti viene stampato a schermo un opportuno messaggio di errore.

Front-end

- File: 5 cartelle più tutti i file php.
    - Css: contiene 3 fogli di stile uno generico, uno per rancid.php e uno per profilo.php
    - Db: contiene config.php (sono salvati i dati di accesso al database) e rancid.sql (file per creare il database del sito)
    - File: contiene una sottocartella Trama dove sono salvate tutte le trame dei film
    - Img: contiene tutte le immagini del sito e le locandine dei film
    - Js: contiene la libreria JQuery, il foglio js di film.php, il js di profilo.php e il js di login.php
    - Index.php: include top,login e bottom .php
    - Rancid.php: include top,function,bottom .php
    - Film.php. include top,function,bottom .php
    - Profilo.php. include top,function,bottom .php
    - Function.php: contiene la funzione per connettersi al database più tutte le query/funzioni utilizzate per ricavare le informazioni visualizzate nel sito, la funzione di reindirizzamento e la funzione loggato
    - Top.php: contiene il banner del sito
    - Bottom.php: contiene i badge del validatore html e css più il pulsante logout visibile solamente una volta effettuato l'accesso che richiama logout.php
    - Login.php: contiene lo scheletro hml di index.php

    - Logout.php: include function.php e quando viene richiamata da bottom.php distrugge la sessione, ne avvia una nuova e reindirizza a index.php
    - Verify_login.php: usata dalla index per validare i dati e inserirli nel database
    - Ajax.php: usata per ricavare tutti i dati del film selezionato in rancid.php e per ricavare i dati del profilo

- Top: banner => banner posto centralmente in alto che si estende per tutta la larghezza della pagina
- Logout: pulsante per scollegarsi dal sito
- Bottom: contiene le immagini dei validatori html e css

- Index:
    - Presentazione/Contenuto: Top, una sezione centrale contente il nome del sito, un messaggio flash di istruzioni, l'area di registrazione e l'area di accesso con i rispettivi campi, infine bottom
    - Comportamento: una volta cliccati i pulsanti "Registrati" o "Accedi", una funzione javascript per pulsante si occupa di reindirizzare l'utente alla home del sito oppure, in caso di fallimento di eseguire nuova la procedura di accesso

- Rancid:
    - Presentazione: Top, una sezione centrale di benvenuto/a o bentornato/a con il proprio nome, una barra contenente il pulsante per andare al profilo, una bacheca contenente tutti i film del sito con le rispettive locandine,logout e bottom
    - Contenuto: tutti i film più relativa locandina
    - Comportamento: il pulsante "Profilo" reindirizza alla pagina profilo tramite funzione javascript

- Profilo:
    - Presentazione: Top, Nome e cognome allineati a sinistra, un'area con tutti i commenti fatti raggruppati per film, una colonna sulla destra con tutte le informazioni dell'utente, un link per tornare alla home, logout e bottom
    - Contenuto: tramite query si ricavano tutti i commenti fatti dall'utente e tramite group by si raggruppano per film; anche le informazioni dell'utente vengono ricavate tramite query
    - Comportamento: il pulsante "Modifica" consente di aprire una sorta di menù in cui si possono modificare i dati inseriti dall'utente tramite funzione

javascript; solamente i campi compilati verranno modificati e una volta cliccato il pulsante “Invia” (comparso dopo aver premuto “Modifica”), i campi compilati sono acquisiti e se validi, sostituiscono i vecchi valori; il pulsante “back” reindirizza alla home page

- Film:
    - Presentazione: Top, colonna centrale divisa in una colonna sinistra (100 pixel di larghezza) e una destra (250 pixel);
        - Sx: titolo, rank del film, trama, area commenti, area scrittura commenti, pulsante back
        - Dx: Locandina del film, genere, Direttore, Attori (questi ultimi sono organizzati in sezioni e sono delle liste)
        - Infine logout e bottom
    - Contenuto: una volta caricata la pagina tramite javascript e ajax si recuperano tutti i contenuti:
        - Il rank qualora sia valido, viene rappresentato oltre che in percentuale tramite una barra colorata che ha larghezza uguale al rank in pixel moltiplicato 10
        - La trama è un paragrafo e si ricava tramite lettura del relativo file
        - Area commenti: area centrale con tutti i commenti posti in una griglia di n righe x 3 colone
        - Genere, direttore e Attori vengono ricavati tramite query e organizzati in liste
    - Comportamento: il pulsante “Pubblica” consente, tramite funzione javascript e richiesta ajax, di acquisire il commento inserirlo nel database e visualizzarlo a video

Back-end

- Db: nome: Rancid -> 9 tabelle
    - Actors
        - Id: int(11) PRIMARYKEY
        - First_name: varchar(100)
        - Last_name: varchar(100)
        - Gender: char(1)
        - Film_count: int(11)
    - Commenti
        - Id: int(11) PRIMARYKEY AUTO_INCREMENT
        - Commento: text
        - Film_id: int(11)
        - Nome_autore: varchar(100)

        - Cognome_autore: varchar(100)
    - Director
        - Id: int(11) PRIMARYKEY
        - Firts_name: varchar(100)
        - Last_name: varchar(100)
    - Directors_genres
        - Director_id: int(11) INDICE
        - Genre: varchar(100)
        - Prob: float
    - Movies
        - Id: int(11) PRIMARYKEY
        - Name: varchar(100)
        - Year: int(11)
        - Rank: float
    - Movies_directors
        - Director_id: int(11) INDICE
        - Movie_id: int(11) INDICE
    - Movie_genres
        - Movie_id: int(11)
        - Genre: varchar(100)
    - Roles
        - Actor_id: int(11) INDICE
        - Movie_id: int(11) INDICE
        - Role: varchar(100)
    - Utenti
        - Id int(11) PRIMARYKEY AUTO_INCREMENT
        - Nome: varchar(100)
        - Cognome: varchar(100)
        - Password: varchar(100)
        - Genere: char(1)
        - Giorno: int(2)
        - Mese: int(2)
        - Anno: int(2)

- Function.php: contiene 12 funzioni:
    - Redirect
        - Input: 2 stringhe:
            - url: la pagina a cui reindirizzare
            - flash_messagge: il messaggio che verrà stampato nell'apposito div in risposta ad un'azione dell'utente
        - Contenuto: se "flash_message" è valido si setta il campo "flash" di _SESSION con "flash_message", si reindirizza l'utente alla pagina "url" e si chiude la pagina attuale

- Remove_
  - Input
    - Movie: tipo stringa, rappresenta il nome di un film
  - Contenuto: sostituisce tutte le occorrenze del carattere "_" con " " (spazio)

- Connection
  - Input
    - Config: array associativo contenente tutti i dati del database
  - Contenuto: recupera tutti i dati da "config", crea un nuovo oggetto PDO che servirà per eseguire query e ritorna l'oggetto stesso. In caso di errore una try catch si occupa di stampare a video un messaggio di errore

- Query_utente
  - Input
    - Db: oggetto PDO per la connessione al database
    - Nome e cognome: tipo stringa
  - Contenuto: se l'oggetto db è valido viene eseguita una query che ritorna i dati dell'utente "nome" e "cognome" e ritorna gli stessi dati. In caso di errore una try catch setta a null il valore di ritorno

- Query
  - Input
    - Db: oggetto PDO per la connessione al database
    - Query: la query che si vuole eseguire
    - Movie: nome film
  - Contenuto: se db è valido prepara ed esegue la query e ne ritorna il valore. In caso di errore una try catch setta a null il valore di ritorno

- Queryprofilo
  - Input
    - Db: oggetto PDO per la connessione al database
    - Nome e cognome: tipo stringa
  - Contenuto: se db è valido vengono recuperati i commenti associati all'utente raggruppati per film e si ritornano. In caso di errore una try catch setta a null il valore di ritorno

- Loggato
  - Contenuto: controlla che il campo "_SESSION["nome"]" sia settato per evitare che gli utenti saltino la pagina di accesso e in caso reindirizza alla pagina iniziale

- Read_trama
  - Input
    - Nome: tipo stringa, il nome del file trama associato al film
  - Contenuto: se il nome è valido apre il file associato al film selezionato contente la trama, la acquisisce, la salva in una stringa e viene ritornata altrimenti ritorna null

- Require_id
  - Input
    - Db: oggetto PDO per la connessione al database
    - Filme: tipo stringa
  - Contenuto: se db è valido ritorna l'id associato al film. In caso di errore una try catch setta a null il valore di ritorno

- Write_comment
  - Input
    - Db: oggetto PDO per la connessione al database
    - Nome, cognome e testo: tipo stringa
    - Id: tipo intero
  - Contenuto: se db è valido si esegue la quote su tutti i campi, dopodiché si inserisce il commento nel database, altrimenti si ritorna null. In caso di errore una try catch setta a null il valore di ritorno

- Modificaprofilo
  - Input
    - Query: la query che viene eseguita
    - Modifica, nome e cognome: tipo stringa
  - Contenuto: si stabilisce la connessione al database, dopodiché si esegue la query che viene passata (si vanno a modificare dei campi del profilo dell'utente). In caso di fallimento si ritorna null

- Read_commenti

        - Input
            - Query: la query che viene eseguita
            - Nome: tipo stringa
        - Contenuto: se db è valido vengono ritornati tutti i commenti associati al film. In caso di errore si ritorna null

- Index:
    - Client-side:
        - MyFunctionsx e myFunctiondx: si occupano di mostrare o nascondere il campo password dell'area registrazione (MyFunctionsx) o dell'area accesso (MyFunctiondx). Per mostrare il campo si acquisisce il campo password e si cambia il tipo da password a text o viceversa per nascondere il campo

        - Profilo: reindirizza alla pagina profilo

        - Agganciaerrore
            - Input
                - Stringa: il messaggio di erroere da stampare
                - Iderroreaggancio: l'id che identifica l'elemento a cui agganciare il messaggio di errore
                - Errore: l'id del messaggio di errore
            - Contenuto: si crea una label, si aggiunge l'id "errore" come attributo, si scrive il testo del messaggio di errore e si aggancia la label all'elemento individuato da "Iderroreaggancio"

        - Isdata
            - Input giorno, mese e anno
            - Contenuto: si controlla che la data sia valida e si ritorna true oppure false

        - Puliscierr
            - Si rimuove l'elemento avente id uguale ad err se è presente

        - Errorerispo
            - Input
                - Str: la stringa contenente il messaggio di errore

- Contenuto: se non è già presente un elemento avente id uguale a “errorerispo”, si crea un elemento h1 con id uguale a “errorerispo”, testo uguale a “str” e lo si aggancia dopo il banner

- Functionreg
  - Contenuto: chiama puliscierr(), acquisisce i vari campi dell’area registrazione, se non sono validi chiama agganciaerrore() , altrimenti viene fatta una richiesta ajax a verify_login.php tramite metodo POST in cui si setta il campo registrazione a true più tutti i dati acquisiti. Si controlla la risposta del server:
    - Risp.erroredb == false
      - Risp.registrazione == true => reindirizzamento a rancid.php
      - Risp.registrazione == false => si rimuove l’elemento con id “flash” e lo si ricrea con il nuovo messaggio ossia “utente già registrato, utilizza la sezione accesso”
    - Risp.erroredb == true si chiama errorerispo() e si comunica all’utente che c’è stato un errore di connessione al database

- Functionacc
  - Contenuto: come functionreg ma con meno campi da acquisire (3 invece di 6). Anche in questo caso si fa una richiesta a verify_login.php:
    - Risp.erroredb == false
      - Risp.vuoto== true=> si rimuove l’elemento con id “flash” e lo si ricrea con il nuovo messaggio ossia “utente non registrato, utilizza la sezione registrazione”
      - Risp.login == false => si rimuove l’elemento con id “flash” e lo si ricrea con il nuovo messaggio ossia “Password errata”
      - Risp.login == true => reindirizzamento a rancid.php
    - Risp.erroredb == true si chiama errorerispo() e si comunica all’utente che c’è stato un errore di connessione al database

- Rancid
  - Client-side
    - Controlla che la sessione si attiva e in caso ne avvia una nuova; richiama loggato() -> controlla che l’utente abbia eseguito l’accesso o il login; richiede la connessione al database ed esegue la query per recuperare tutti i nomi dei film, tramite i quali si caricano nell’area bacheca le locandine che sono dei link alla pagina del film stesso

- Film
    - Client-side
        - Controlla che la sessione si attiva e in caso ne avvia una nuova; richiama loggato() -> controlla che l'utente abbia eseguito l'accesso o il login; rimuove il carattere "_" dal nome del film salvato in "_GET", lo salva in "_SESSION" e viene usato come titolo della pagina. Viene creato l'intero scheletro della pagina.

        - Errorerispo: se non esiste nessun elemento con id = "errorerispo", lo si crea e si setta con messaggio di errore la stringa passata come paramentro

        - Una volta carica la pagina vengono eseguite 6 funzioni javascript

        - Rank: fa una richiesta tramite POST a ajax.php passando il campo rank a true e attende la risposta.
            - Risp.erroredb == false: si crea un div e un paragrafo
                - Se il rank == null -> si assegna l'id "rankgreen" al div e si impone la larghezza a 100 pixel in questo modo, poiché il rank è nullo non apparirà nessuna barra a rappresentare il rank
                - Se rank != null -> si usa assegna l'id "rankred" che sarà largo quanto il numero del rank moltiplicato 100 pixel
                - Il tutto viene agganciato al tag con id "rankeste"

            - Risp.erroredb == true: si richiama errorerispo() passando come parametro il messaggio di errore

        - Read_trama: fa una richiesta tramite POST a ajax.php passando il campo trama a true e attende la risposta.
            - Risp.erroredb == false: crea un paragrafo, e gli assegna il valore della risposta ajax, infine lo aggancia a "#trama"
            - Risp.erroredb == true: si richiama errorerispo() passando come parametro il messaggio di errore

        - Read_commenti: fa una richiesta tramite POST a ajax.php passando il campo commenti a true e attende la risposta.
            - Risp.erroredb == false: per ogni commento valido si crea un div con classe "commento", composto da un titolo (nome e cognome

dell'autore del commento) e un testo (il commento) e lo si aggancia a "#commenti"
- Risp.erroredb == true: si richiama errorerispo() passando come parametro il messaggio di errore

- Read_genere: fa una richiesta tramite POST a ajax.php passando il campo generea a true e attende la risposta.
  - Risp.erroredb == false: viene creata una unsigned list dove ogni genere è un elemento della lista e la si aggancia a "#genere"
  - Risp.erroredb == true: si richiama errorerispo() passando come parametro il messaggio di errore

- Read_direttore: fa una richiesta ajax tramite POST a ajax.php passando il campo direttore a true e attende la risposta.
  - Risp.erroredb == false: viene creata una unsigned list dove ogni direttore è un elemento della lista e la si aggancia a "#direttore"
  - Risp.erroredb == true: si richiama errorerispo() passando come parametro il messaggio di errore

- Read_attori: fa una richiesta ajax tramite POST a ajax.php passando il campo attoria true e attende la risposta.
  - Risp.erroredb == false: viene creata una unsigned list dove ogni attore con rispettivo ruolo è un elemento della lista e la si aggancia a "#attori"
  - Risp.erroredb == true: si richiama errorerispo() passando come parametro il messaggio di errore

- Pubblica: viene chiama ogni volta che un utente fa un commento. Si recupera nome, cognome dell'autore e testo del commento e si passano tramite metodo POST ad ajax.php
  - Risp.erroredb == false: si crea un div composto da un titolo (nome e cognome dell'autore) e un testo (il commento) e lo si aggancia a "#commenti"
  - Risp.erroredb == true: si richiama errorerispo() passando come parametro il messaggio di errore

- Profilo:
  - Client-side
    - Controlla che la sessione si attiva e in caso ne avvia una nuova; richiama loggato() -> controlla che l'utente abbia eseguito l'accesso o il login e viene creato lo scheletro della pagina

- Modificagenere/modificagiorno/modificamese/modificaanno: sono tutte funzioni che vanno a prendere il rispettivo campo della lista con le informazioni dell'utente, lo eliminano e ne creano uno nuovo che ha come valore la variabile passata come parametro

- Isdata: controlla che la data passata come parametro sia una data valida

- Pulisci: elimina l'elemento con id "textedit" ossia i 4 campi in cui è possibile scrivere se si vuole modificare le informazioni (genere e data di nascita) dell'utente, e crea il pulsante send associandogli la funzione javascript edit(). Aggancia il tutto a "#dx"

- Inviodb: fa una richiesta tramite POST a ajax.php passando come parametri la query (una update), la modifica (il campo che deve essere aggiornato) e il nome e cognome dell'utente
    - Se risp.errore == true viene chiamata errorerrispo con un opportuno messaggio di errore
    - Altrimenti se
        - Mod == "genere" viene chiamata modificagenere()
        - Mod == "giorno" viene chiamata modificagiorno()
        - Mod == "mese" viene chiamata modificamese()
        - Mod == "anno" viene chiamata modificaanno()

- Errorerispo
    - Input
        - Str: la stringa contenente il messaggio di errore

    - Contenuto: se non è già presente un elemento avente id uguale a "errorerispo", si crea un elemento h1 con id uguale a "errorerispo", testo uguale a "str" e lo si aggancia dopo il banner.
        - Se fallisce viene chiamata errorerrispo con un opportuno messaggio di errore
        - Se termina con successo: viene chiamata pulisci()
        - Se termina con errore: viene chiamata errorerrispo con un opportuno messaggio di errore
        - Al termine si ritorna il valore della risposta del database

- Replace: sostituisce tutti i caratteri "_" con spazi bianchi e ritorna la stringa modificata

- Edit: crea le 4 aree in cui l'utente può scrivere se vuole modificare il campo genere, data, mese o anno. Crea anche il tasto invia e gli associa la funzione javascript send(). Aggancia il tutto a "#textedit"

- Info: fa una richiesta tramite metodo POST ad ajax.php passando profiloinfo con valore true.
    - In caso di fallimento viene chiamata errorerrispo con un opportuno messaggio di errore

    - In caso di successo
        - Se risp.erroredb == false: si crea la lista formata da 2 colonne (il nome del campo a sinistra e a destra il suo valore) e 6 righe (nome, cognome, genere, giorno, mese, anno)

        - Se risp.erroredb == true: viene chiamata errorerrispo con un opportuno messaggio di errore

- filmcommenti: fa una richiesta tramite metodo POST ad ajax.php passando filmcommenti con valore true.
    - In caso di fallimento viene chiamata errorerrispo con un opportuno messaggio di errore

    - In caso di successo
        - Se risp.erroredb == false: si crea il primo commento, costituito dal nome del film seguito dal commento inserito in una lista e lo si aggancia all'area commenti, dopodiché se il successivo riguarda lo stesso film lo si inserisce nella lista altrimenti si crea un nuovo commento. Il tutto finché non terminano i commenti

        - Se risp.erroredb == true: viene chiamata errorerrispo con un opportuno messaggio di errore

- Verify_login
    - Server-side: se la sessione non è attiva, viene avviata; richiede i dati per accedere al database:

        - Se _POST[“registrazione”] è attivo, lo distrugge, acquisisce tutte le variabili da _POST (7 in totale), vengono sostituiti eventuali caratteri html e si stabilisce la connessione al database:
            - Se db è valido: si richiama query_utente()
                - Se il numero di righe ritornate è uguale a 0 si crypta la password, si esegue la quote sui 6 restanti campi, si inseriscono i dati nel database e si ritorna un oggetto json :
                    - Erroredb => il suo valore a seconda del valore di ritorno della query (false se la query va a buon fine, true altrimenti)
                    - Registrazione => true
                - Altrimenti json
                    - Erroredb => false
                    - Registrazione => false
            - Altrimenti json
                - Erroredb => true
                - Registrazione => false

        - Se _POST[“accesso”] è attivo, lo distrugge, acquisisce tutte le variabili da _POST (3 in totale) e si stabilisce la connessione al database:
            - Se db è valido: si richiama query_utente()
                - Se $vet è valido
                    - Se il numero di righe è 0 => json
                        - Erroredb => false
                        - Vuoto => true
                    - Altrimenti si verifica la password
                        - Se è corretta json
                            - Erroredb => false
                            - Login => true
                        - Altrimenti json
                            - Erroredb => false
                            - Login => false

        - Altrimenti json
            - Erroredb => true
    - Altrimenti json
        - Erroredb => true
        -

- Ajax
    - Server-side
        - Si richiedono di dati del database
            - Se db è valido
                - Se e settato _POST[rank]
                    - Si esegue la query per ricavarsi il rank
                        - Se ritorna un rank valido errore=false
                        - Altrimenti errore =true
                    - Si ritorno un json
                        - Erroredb => valore di errore
                        - Rank => valore del rank
                - Se e settato _POST[trama]
                    - Si richiama la read_trama
                        - Se ritorna una trama valida errore=false
                        - Altrimenti errore =true
                    - Si ritorno un json
                        - Erroredb => valore di errore
                        - trama=> valore di trama
                        -
                - Se e settato _POST[commenti]
                    - Si richiama la read_commenti
                        - Se ritorna dei commenti validi errore=false
                        - Altrimenti errore =true
                    - Si ritorno un json
                        - Erroredb => valore di errore
                        - commenti=> valore dei commenti
                        -
                - Se e settato _POST[genere]
                    - Si esegue la query sul genere dei film
                        - Se ritorna dei generi validi errore=false
                        - Altrimenti errore =true
                    - Si ritorno un json
                        - Erroredb => valore di errore
                        - genere=> valore dei generi

- Se e settato _POST[direttore]
  - Si esegue la query sui direttori dei film
    - Se ritorna dei direttori validi errore=false
    - Altrimenti errore =true
  - Si ritorno un json
    - Erroredb => valore di errore
    - direttori=> valore dei direttori
    -
- Se e settato _POST[attori]
  - Si esegue la query sugli attori dei film
    - Se ritorna degli attori validi errore=false
    - Altrimenti errore =true
  - Si ritorno un json
    - Erroredb => valore di errore
    - attori=> valore degli attori
    -
- Se e settato _POST[commento]
  - Si sostituiscono eventuali caratteri html dal commento, si richiede l'id del film
    - Se ritorna un id valido
      - Errore =false
      - si richiama la write_comment() per inserire il commento nel database
        - Se il valore di ritorno == true json erroredb => valore di errore,nome =>valore del nome, cognome => valore del cognome
        - Altrimenti errore = true
    - Altrimenti errore = true
  - Si ritorno un json
    - Erroredb => valore di errore

- Se e settato _POST[profiloinfo]
  - Si esegue la query sulle informazioni dell'utente
    - Se ritorna delle informazioni valide errore=false
    - Altrimenti errore =true

        - Si ritorno un json
            - Erroredb => valore di errore
            - nome=> valore dele nome
            - Cognome => valore del cognome
            - Genere => valore del genere
            - Giorno => valore del giorno
            - Mese => valore del mese
            - Anno => valore dell'anno
            -
    - Se e settato _POST[film_commenti]
        - Si esegue la query sui commenti dell'autore ai film
            - Se ritorna dei commenti validi errore=false
            - Altrimenti errore =true
            - Se row è valido
                - Json
                    - Erroredb => valore di errore
                    - Commento => valore del commento
                    - Film => valore del film
            - Altrimenti json
                - Erroredb => valore di erroredb

    - Se e settato _POST[querydb] e _POST["modifica"] e _POST["nomedb"] e _POST["cognomedb"]
        - Si sostituiscono eventuali caratteri html di _POST["modifica"]
        - Si esegue la query per modificare i dati del profilo
            - Se ritorna true errore=false
            - Altrimenti errore =true
        - Si ritorno un json
            - Erroredb => valore di errore
- Altrimenti
- Json
    - Erroredb => true