Samuele Perrotta 943460

Relazione Progetto TWEB Anno 2022/2023

TEMA DEL SITO

Il sito tratta di una società (Cars On Track), che permette di prenotare dei giri in pista con delle macchine e dei circuiti messi a disposizione dall'azienda stessa e di scrivere delle recensioni sull'esperienza fatta.

SEZIONI PRINCIPALI

Pagina di Login, che permette di accedere al sito.

Pagina di Registrazione, che permette di creare un nuovo account.

Home, che permette di visualizzare tutte le auto, le piste e le recensioni.

Prenota, che permette di prenotare un giro in pista.

Lascia Una Recensione, che permette di scrivere una recensione.

Esci, che permette la disconnessione dal sito.

Modifica, riservato all'amministratore, che permette di aggiungere ed eliminare auto e piste.

FUNZIONALITÀ

Login/Logout: Per fare il Login nel sito bisogna inserire l'email e la password negli appositi campi del form di accesso. Vengono effettuati controlli lato client. Il form è contenuto nel file *index.php* collegato al file JavaScript *index.js*, che gestisce il suo comportamento. Per accedere bisogna cliccare sul bottone "Accedi". Si genera un evento click, gestito da JavaScript, e Ajax invierà una richiesta allo script *logIn.php*, il quale, una volta controllati i campi e verificato che l'utente esiste attraverso l'invocazione della query in "searchUser", ritorna una risposta al client, il quale con la funzione di successo "logIn" aprirà l'homepage del sito. Per eseguire il Logout si deve premere un link in alto a destra "Esci", che richiama la pagina *logOut.php*, la quale eliminerà la sessione e aprirà di nuovo la pagina di accesso.

Registrazione: Per effettuare la registrazione bisogna premere il link "Registrati", posto in basso a destra nel form di accesso, il quale aprirà la pagina *pageRegistration.php*, che conterrà un form di inserimento dati, quali Nome, Cognome, Codice Fiscale ecc… e due pulsanti al fondo. Uno "Reimposta" per ripulire i campi e uno "Registrati" per registrarsi. Cliccato "Registrati", dopo aver verificato lato client i valori dei campi, si genera un evento click gestito da JavaScript, che genera una richiesta Ajax allo script *subscription.php*, che controllerà i valori dei campi lato server e inserirà l'utente nel database con la query in "insertUser" e poi aprirà l'homepage *homepage.php*.

Ruoli: I ruoli sono stati implementati aggiungendo una colonna "Admin" nella tabella "users" nel database. Se ha valore 1 l'utente è un amministratore, se ha valore 0 è un utente semplice. Ogni volta che una pagina del sito viene caricata completamente, viene

inviata una richiesta Ajax allo script *verifyAdmin.php*, che ritorna i dati dell'utente loggato, grazie alla query in "verifyAdmin". JavaScript controlla il valore dell'attributo "Admin" dell'utente. Se è 0 rimuove il link "Modifica" che manda a *pageManagementAdmin.php*, altrimenti lo mostra.

Gestione del contenuto generato dall'utente: Il contenuto generato dall'utente comprende: 1) La visualizzazione di una lista di tutte le macchine, una di tutte le piste e una di tutte le recensioni del database. Si accede a queste liste premendo una delle tre immagini dell'homepage, che fanno partire una richiesta Ajax agli script *showcars.php*, *showTracks.php* e *showsReviews.php*, che ritornano al client, tramite query invocate lato server, dei JSON con tutti i dati richiesti. 2) Un form per prenotare uno o più giri in pista. Ci sono dei campi da compilare, controllati lato client, e un pulsante "Prenota", cliccato si genera un evento click che invia una richiesta Ajax allo script *addReservation.php*, che controlla i valori dei campi lato server e inserisce la prenotazione nel database con la query in "insertReservation". 3) Un form per scrivere recensioni. Ci sono dei campi da compilare, controllati lato client, e un pulsante "Invia recensione", una volta cliccato si genera un evento click che invia una richiesta Ajax allo script *addReview.php*, che controlla i valori dei campi lato server e inserisce la recensione nel database con la query in "insertReview".

CARATTERISTICHE

Usabilità: L'usabilità del sito è stata assicurata con più accorgimenti. 1) Quando un utente immette dei dati errati o lascia un campo vuoto, come ad esempio un'email inesistente nell'accesso o nella registrazione o una data antecedente il giorno odierno nella prenotazione o il testo della recensione lasciato vuoto, viene visualizzato un messaggio di errore in rosso sotto al form. 2) Quando un utente accede al sito viene stampato, in alto nella barra di navigazione, un messaggio "Ciao *nome_utente*". 3) Il sito ha un colore di sfondo blu e bianco gestito con i gradienti (*linear_gradient()*) e le scritte sono in nero o in bianco quando lo sfondo non è bianco. 4) Quando l'utente effettua il Logout viene visualizzato in alto un messaggio in rosso di avvenuto scollegamento, che piano piano scompare. 5) Quando un utente effettua una prenotazione o quando invia una recensione o quando l'amministratore aggiunge o elimina auto o piste, viene visualizzato in alto un messaggio in rosso di avvenuta operazione, che piano piano scompare. 6) Nella registrazione se i campi sono sintatticamente corretti e non sono vuoti, sono colorati di verde, altrimenti di rosso. In generale i messaggi sono chiari, brevi, i colori non danno fastidio e il design è intuitivo.

Interazione/animazione: L'interazione/animazione è stata gestita con l'uso del metodo *animate({opacity: '0'}, 2000)* nel file JavaScript *reservation.js*, per prenotare una corsa, che viene attivato quando si preme il pulsante "Tenta la fortuna", il quale, una volta premuto, crea un array con 4 frasi che indicano un premio diverso e 2 che indicano nessuna vittoria, nasconde se stesso rendendo la sua opacità a zero con un tempo di 2000 millisecondi e, accedendo in maniera randomica ad una posizione dell'array, fa comparire una delle 6 frasi

in rosso. Inoltre il messaggio di avvenuto logout e i messaggi di avvenuta prenotazione, recensione, eliminazione, aggiunta, usano anche il metodo *animate({opacity: '0'}, 3000)*.

Sessioni: All'inizio del file in *index.php* e *pageRegistration.php* se non è presente una sessione, viene creata(*session_start()*). Nelle altre pagine che mostrano contenuto all'utente, come *pageManagementAdmin.php, homepage.php, pageReview.php* e *pageReserve.php*, se non è presente una sessione viene creata e si mostra effettivamente il contenuto della pagina solo se i dati di sessione "email" e "password" dell'array associativo "$_SESSION" sono settati, altrimenti si viene reindirizzati alla pagina di accesso *index.php*. Per scollegarsi, nel file *logOut.php*, viene fatto flush dei dati (*session_unset()*) e la sessione viene eliminata (*session_destroy()*) e si è reindirizzati alla pagina di accesso. Negli script php che hanno bisogno del dato di sessione "email" in "$_SESSION", si riprende la sessione corrente con *session_start()* e si usa il dato.

Interrogazione del database: Le interrogazioni del database vengono mantenute in un file comune *commonQuery.php*, che contiene la connessione al database e tutte le query utilizzate nel sito, ad esempio per accedere, quindi per verificare se un utente è nel database, o per visualizzare tutte le specifiche di auto e piste, quindi facendosi ritornare dal database tutti i campi di auto e piste disponibili.

Validazione dei dati in input: I dati immessi dagli utenti vengono sempre validati lato client, con formule regex, input type appositi, clausola required e lato server, con controlli *htmlspecialchars()*, *filter_input()*, lunghezza esatta dei valori dei campi, verifica campi vuoti, funzione di hash *md5()* per criptare password, *quote()* prima di immettere valori nel database.

Sicurezza: Tutte le richieste al server vengono fatte con metodo POST, si usano funzioni *htmlspecialchars()* per evitare attacchi XSS e *quote()* e *filter_input()* per SQL Injection.

Presentazione: Viene usato uno schema a 3 colonne nella homepage e a colonna singola centrata nelle pagine della prenotazione e della recensione. Nella pagina di accesso e di registrazione si utilizza uno schema a due colonne centrate, con a sinistra un testo e a destra il form di accesso o registrazione. Il sito utilizza un layout flessibile e risulta essere chiaro e intuibile per via dei colori (blu, bianco, nero, rosso) e della posizione degli elementi sullo schermo, alcuni con posizione fissa, come la barra di navigazione o il footer.

FRONT-END

Separazione presentazione/contenuto/comportamento: La **presentazione** del sito viene gestita con 3 file CSS: *homepageStyle.css, indexStyle.css e registrationStyle.css*. Il primo si occupa dello stile dell'homepage e delle pagine di prenotazione e recensione, mentre gli altri due rispettivamente della pagina di accesso e di registrazione. Vengono inclusi nei file attraverso il tag HTML *<link href="percorso_css">*. Il **contenuto** del sito viene mantenuto nelle pagine php *pageManagementAdmin.php, homepage.php, pageReview.php*, *pageReserve.php, index.php* e *pageRegistration.php*, con i tag HTML e includono

header.html e *footer.html* grazie a *include_once()*. Il **comportamento** del sito viene gestito dai file JavaScript *index.js, homepage.js, managementAdmin.js, registration.js, reservation.js, review.js*, che vengono inclusi nei file php attraverso il tag HTML *<script src="percorso_js">*.

Soluzioni cross-platform: Il sito usa un layout flessibile, utilizzando il più possibile misure percentuali, per permettere una corretta visualizzazione in base allo schermo su cui viene visualizzato il sito.

Organizzazione file e cartelle di progetto: I file hanno un nome significativo. Se un file si riferisce all'homepage si chiamerà XXXhomepageXXX.css/php/js e ogni file è stato inserito in una cartella denominata con il nome del tipo del file. La cartella img è stata classificata in 4 sottocartelle che differenziano immagini di auto, piste, recensioni e icone/loghi. La cartella php è stata classificata in 4 sottocartelle che differenziano azioni dell'admin, utente, registrazione e accesso. Quelle più generali sono fuori da queste 4 sottocartelle. L'unico file fuori da tutte le cartelle e *index.php*.

Soluzioni html/css/javascript degne di nota: La creazione di un bottone "mostra/nascondi password", con l'utilizzo degli eventi *mousedown()* e *mouseup()* in JavaScript. Quando si tiene premuto il pulsante "Mostra password" il valore di type passa da password a text e il testo del bottone diventa "Nascondi password", quando si rilascia il bottone tutto torna come prima.

BACK-END E COMUNICAZIONE FRONT/BACK-END

Architettura generale classi/funzioni php: L'architettura generale delle classi php è la seguente: Le pagine che usano tag HTML e che mostrano contenuto all'utente contengono all'inizio del file il controllo della sessione, nel mezzo il contenuto della pagina, subito dopo il tag <body> l'inclusione dell'*header.html* e alla fine del file l'inclusione del *footer.html*. Le pagine lato server invisibili all'utente contengono il controllo che la richiesta al server sia di tipo POST, riprendono la sessione, recuperano i valori passati da client, controllano i dati (sicurezza), richiamano la query necessaria e gestiscono il valore di ritorno della query restituendo al client una risposta di tipo text o json.

Schema del database:

cars (ID, NumeroTelaio, Marchio, Modello, Anno, Cavalli, Prezzo, Descrizione)

reservations (ID, *Auto*, *pista*, *Utente*, data, Ora)

reviews (ID, *Autore*, *Auto*, *Circuito*, Commento, Grado)

tracks (ID, Nome, Città, Lunghezza, NumeroCurve)

users (ID, Nome, Cognome, CF, Sesso, DataNascita, Email, Password, Admin)

chiave primaria *chiave esterna*

Descrizione delle funzioni remote: Tutte le funzioni remote vengono invocate con una richiesta di tipo POST che inizia sempre con la seguente URL *http://localhost/progetto/php/nome_funzione_remota*. Essendo di tipo POST non verranno passati i parametri come query string nell'URL. Ogni Funzione remota viene richiamata tramite Ajax, dunque in caso di errori si eseguirà il metodo *ajaxfailed()* che visualizzerà in un *alert()* lo stato e la risposta del server.
Elenco delle funzioni remote che ritornano al client un dato di tipo text:
logIn.php: Funzione che permette di verificare se un dato utente esiste. I parametri passati sono email e password e invoca la funzione *searchUser()*. Ritorna "0" in caso non esista, "1" in caso esista, "-1" se il formato di email e/o password è errato. La funzione di callback Ajax "logIn" reindirizza l'utente all'homepage o stampa un errore.
addCar.php: Funzione che permette di aggiungere un'auto al database. I parametri passati sono telaio, marchio, modello, anno, cavalli, prezzo e descrizione e invoca la funzione *insertCar()*. Ritorna "0" in caso di fallimento, "1" in caso di successo, "-1" se c'è un campo vuoto. La funzione di callback Ajax "addCar" stampa un messaggio di avvenuta aggiunta o stampa un errore.
addTrack.php: Funzione che permette di aggiungere una pista al database. I parametri passati sono nome, città, lunghezza e NumeroCurve e invoca la funzione *insertTrack()*. Ritorna "0" in caso di fallimento, "1" in caso di successo, "-1" se c'è un campo vuoto. La funzione di callback Ajax "addTrack" stampa un messaggio di avvenuta aggiunta o stampa un errore.
removeCar.php: Funzione che permette di rimuovere un'auto dal database. Il parametro passato è telaio e invoca la funzione *removeCar()*. Ritorna "0" in caso di fallimento, "1" in caso di successo, "-1" se c'è un campo vuoto. La funzione di callback Ajax "removeCar" stampa un messaggio di avvenuta rimozione o stampa un errore.
removeTrack.php: Funzione che permette di rimuovere una pista dal database. Il parametro passato è id e invoca la funzione *removeTrack()*. Ritorna "0" in caso di fallimento, "1" in caso di successo, "-1" se c'è un campo vuoto. La funzione di callback Ajax "removeTrack" stampa un messaggio di avvenuta rimozione o stampa un errore.
subscription.php: Funzione che permette di registrare un nuovo utente nel database. I parametri passati sono name, surname, CF, sesso, date, email, password, admin e invoca la funzione *insertUser()*. Ritorna "0","-1","-2","-3" in caso di fallimento, "1" in caso di successo. La funzione di callback Ajax "registration" reindirizza l'utente all'homepage o stampa un errore.
addReservation.php: Funzione che permette di aggiungere una prenotazione al database. I parametri passati sono marchio, modello, pista, data e ora e invoca la funzione *insertReservation()*. Ritorna "0","-1","-2","-3","-4","-5","-6"in caso di fallimento, "1" in caso di successo. La funzione di callback Ajax "addReservation" stampa un messaggio di avvenuta prenotazione o stampa un errore.
addReview.php: Funzione che permette di aggiungere una recensione al database. I parametri passati sono marchio, modello, track, review, degreePreference, data e ora e

invoca la funzione *insertReview()*. Ritorna “0”,”-1”,”-2”,”-3”,”-4”,”-5” in caso di fallimento, ”1” in caso di successo. La funzione di callback Ajax “sendReview” stampa un messaggio di avvenuto invio recensione o stampa un errore.

Elenco delle funzioni remote che ritornano al client un dato di tipo json:
showCars.php: Funzione che permette di visualizzare tutte le auto nel database. Invoca la funzione *viewTable()*. Ritorna un json nel formato {“name”:dato;”name2”:dato2 ecc…”}. La funzione di callback Ajax “viewCars” visualizza a video un elenco in forma tabellare con tutti i dati delle auto.
showReviews.php: Funzione che permette di visualizzare tutte le recensioni nel database. Invoca la funzione *viewTablerevieww()*. Ritorna un json nel formato {“name”:dato;”name2”:dato2 ecc…”}. La funzione di callback Ajax “viewReviews” visualizza a video un elenco in forma tabellare con tutti i dati delle recensioni.
showTracks.php: Funzione che permette di visualizzare tutte le piste nel database. Invoca la funzione *viewTable()*. Ritorna un json nel formato {“name”:dato;”name2”:dato2 ecc…”}. La funzione di callback Ajax “viewTracks” visualizza a video un elenco in forma tabellare con tutti i dati delle piste.
verifyAdmin.php: Funzione che permette di ritornare i dati dell’utente loggato. Invoca la funzione *verifyAdmin()*. Ritorna un json nel formato {“name”:dato;”name2”:dato2 ecc…”}. La funzione di callback Ajax “verifyAdmin” mostra un saluto con il nome dell’utente e mostra o nasconde il link “Modifica” a seconda se è loggato un utente semplice o l’amministratore.

Tutte le funzioni remote richiamo altre funzioni che eseguono delle interrogazioni al database. Queste ultime funzioni sono contenute nel file *commonQuery.php*.