**LA FORCHETTA**

**Progetto di Applicazione Web e Cloud**

Anno accademico 2016-2017

Università degli Studi di Milano



Realizzato da **Alessandro Frangiamone**

**Introduzione**

Il progetto consiste nella realizzazione dell’applicazione web “La Forchetta”.

Un portale che permette di gestire la ricerca e la prenotazione di ristoranti sparsi su tutto il territorio italiano.

La Forchetta gestisce un numero non conosciuto a priori di ristoranti.

Il progetto è composto da due macro-scenari:

**- Gestione dei ristoranti.**

**- Gestione delle prenotazioni di utenti registrati.**

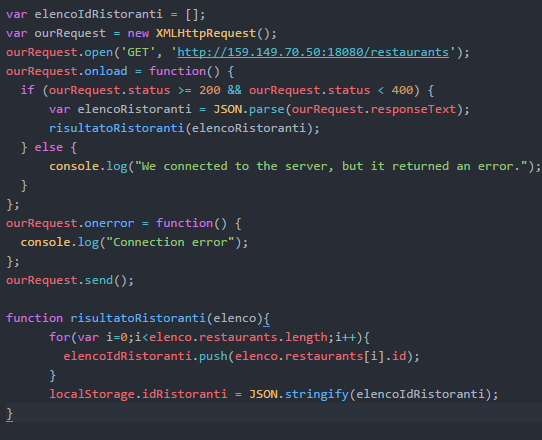
Nelle pagine seguenti verrà spiegato nel dettaglio, anche con porzioni di codice, le scelte effettuate per ogni problematica nata nello sviluppo di questo sito.

Prima di vedere il codice nel dettaglio sottolineo che non è stato utilizzato **nessun** template o codice trovato in rete, ma l’intero sito è stato creato **completamente** ad hoc con le conoscenze impartite nelle lezioni di teoria e laboratorio del corso, più diversi approfondimenti cercati in rete.

**API**

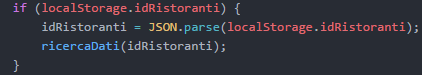
Per la realizzazione del sito, e in particolare per il ricevimento dei dati relativi ai ristoranti è stata utilizzata l’API fornita al momento inziale del progetto.

Allo startup del sito viene stabilita una connessione con l’API, e vengono richiesti tutti gli id dei ristoranti, che vengono poi salvati nel localStorage nel vettore elencoIdRistoranti.



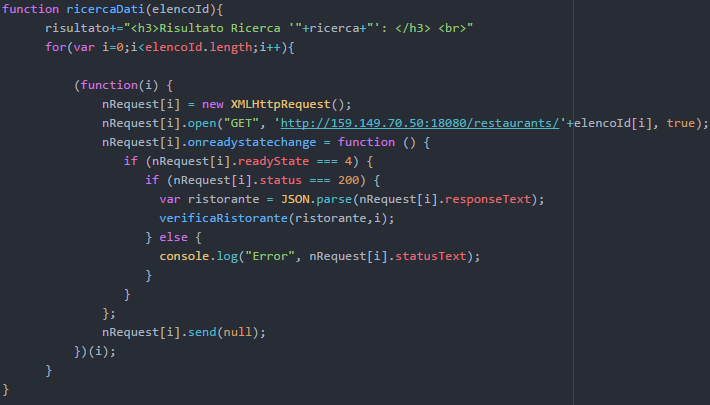
Nel momento in cui si effettua una ricerca viene stabilita una nuova connessione mediante l’API utilizzando stavolta però gli id salvati precedentemente nel localStorage.

Infatti, come prima cosa viene caricata la funzione init nella pagina di ricerca e viene guardato se è presente il vettore contenente gli id (localStorage.elencoId).



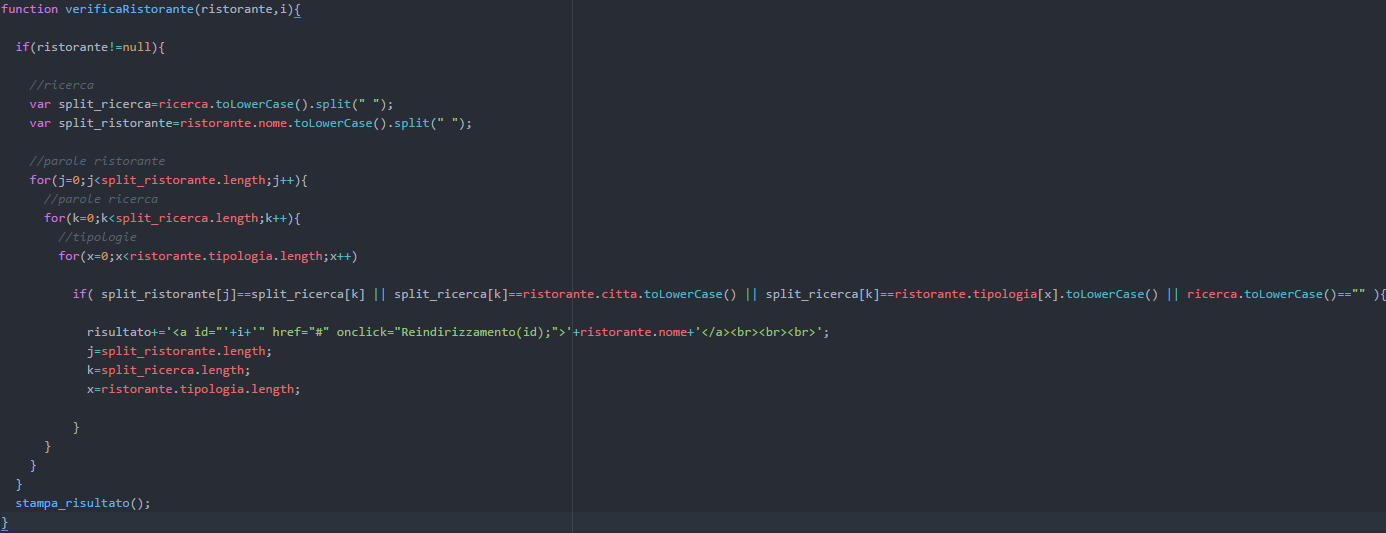
In caso affermativo viene chiamata la funzione ricerca dati alla quale viene passato il vettore con l’elenco degli id.

Nella funzione viene effettuata una nuova chiamata all’API in cui viene passato uno per uno l’id dei ristoranti e per ogni ristorante viene chiamata la funzione verifica Ristorante passando come argomento l’oggetto ristorante ricevuto dall’output dell’API e l’indice i del ristorante.

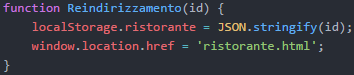


La funzione verificaRistorante scorre ogni ristorante e verifica se ci sono corrispondenze tra la ricerca e il ristorante passato nella funzione, quindi viene controllato nome, tipologia di ristorante e zona geografica.

In caso di corrispondenza viene scritto su pagina il nome del ristorante e il link che permette di accedere al ristorante stesso.



Una volta che poi si clicca su uno dei ristoranti risultati della ricerca viene invocata la funzione Reindirizzamento(id) che in sostanza salva l’id del ristorante cliccato e lo inserisce nel localStorage in una variabile (localStorage.ristorante).

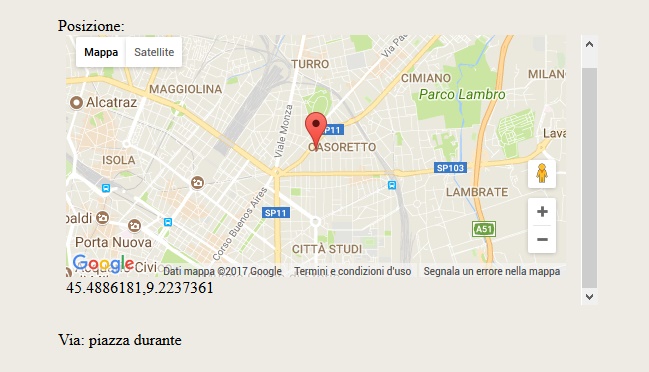


Nella pagina ristorante poi in seguito vengono stampate tutte le informazioni relative al ristorante che vedremo più nel dettaglio nelle pagine successive.

Oltre a questa sono state utilizzate altre due api per funzioni secondarie nel sito.

**Google Maps API:**

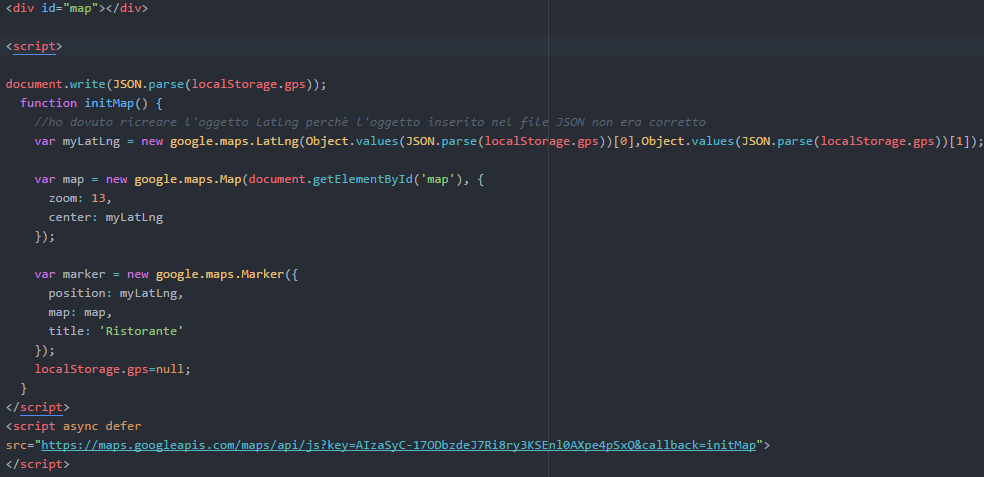
Tra i dati relativi ai ristoranti in alcuni di questi sono presenti le coordinate di latitudine e longitudine, così avendo questi dati ho voluto inserire nel sito la possibilità di avere una mappa che stabilisce il punto in cui è posizionato il ristorante.



Nel dettaglio a livello di codice nella pagina ristorante (dopo aver selezionato il ristorante scelto) vengono salvate nel localStorage le informazioni relative alla longitudine e latitudine, se sono fornite.



Queste informazioni vengono utilizzate nelle pagine mapRistorante



In cui viene creato un nuovo oggetto google.maps.LatLng in cui alla creazione gli passiamo come argomento la latitudine e longitudine che abbiamo inserito nel localStorage (in particolare nella variabile gps di quest’ultimo).

Di seguito viene creata l’oggetto per la mappa vera e propria (google.maps.Map), in cui vengono specificati i parametri di zoom della mappa e il punto centrale della mappa composto dal nostro oggetto precedentemente creato contenente la latitudine e longitudine.

Infine viene creato anche il marker sulla mappa.

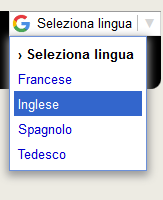
Questa pagina viene richiamata ad ogni selezione di un ristorante ed è contenuta nella pagina di quest’ultimo mediante la creazione di un oggetto che richiama la pagina per la creazione della mappa (in data), questo oggetto poi assume l’output della pagina mapRistorante.



**Google Traslate API:**

Oltre alla mappa per localizzare il ristorante è possibile tradurre la pagina in4 diverse lingue oltre l’italiano (Inglese, Francese, Tedesco e Spagnolo).

È possibile scegliere la lingua desiderata scegliendola dal menù a tendina posto nella parte destra dell’intestazione del ristorante.



Una volta premuto sulla lingua corrispondente viene tradotto in automatico tutto il contenuto della pagina del ristorante che si sta consultando in quel momento.



Per implementare questa API è sufficiente creare un div nella pagina di visualizzazione del ristorante in html, che poi corrisponderà alla posizione del menù a tendina che ci permetterà di scegliere la lingua in cui tradurre la pagina.

Sempre nella stessa pagina è necessario inserire una funzione in cui viene creato e inizializzato l’oggetto google.traslate.TraslateElement dove vengono specificati i parametri della lingua “originale”, le lingue in cui si vuole tradurre la pagina e il layout del menù a tendina.

Questa funzione viene passata poi all’indirizzo di google (presente nell’src dello script) che genererà l’output di traduzione della pagina.

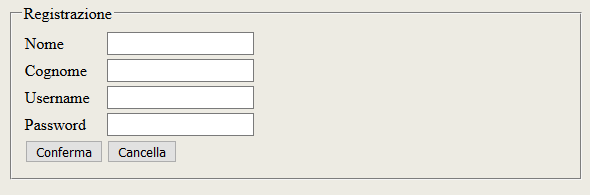
**Operazioni Significative**

Le principali operazioni significative del sito sono:

* **Registrazioni e Login**
* **Ricerca**
* **Prenotazione**

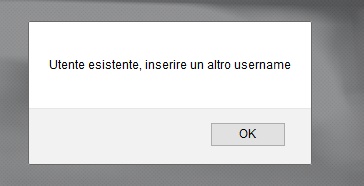
Ognuna di queste andremo ad analizzare singolarmente.

**Registrazione e Login**

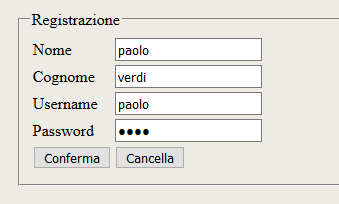
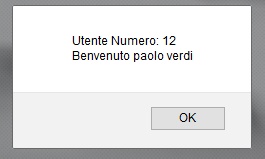
La **registrazione** nel sito consiste nella semplice compilazione di un form che si trova nella Homapage principale dei visitatori, quindi tutti gli utenti che non hanno effettuato il Login. 

I campi necessari per la registrazione sono: nome, cognome, username e password.

L’unico vincolo imposto nella registrazione è che ogni account deve avere un Username diverso, altrimenti comparirà un messaggio di errore.

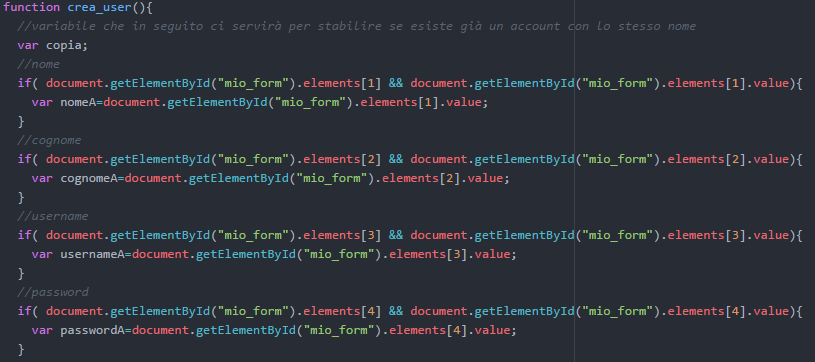


Se invece verrà inserito un username non presente nel sito l’account verrà creato.



Per quanto riguarda il codice cliccando sul bottone “Conferma” verrà invocata la funzione crea\_user().

Questa funzione prima di tutto prende i valori di ogni casella del form di registrazione e viene creata una variabile per ogni dato.

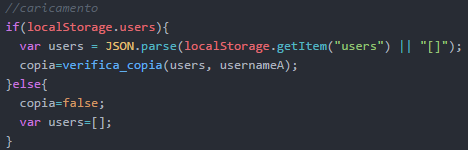


Successivamente si controlla se esiste nel localStorage un vettore di oggetti utente, in sostanza si controlla se il nostro utente che si sta registrando è il primo a registrarsi o ce ne sono già altri.

Se ci sono già utenti registrati viene invocata la funzione verifica\_copia alla quale vengono passati come argomenti l’array degli utenti già esistenti e l’username dell’utente che si vuole registrare in questo momento.

La funziona verifica\_copia restituisce un booleano, la variabile copia prende il valore di questo booleano restituito.

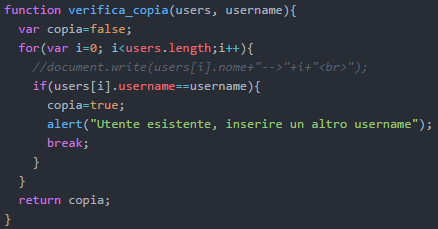
Altrimenti viene settata la variabile copia direttamente a false e viene creato un array vuoto, questa parte ovviamente solo nel caso non ci siano ancora utenti registrati al sito, quindi viene eseguita solo per il primo utente registrato.



Nella funzione verifica\_copia come prima cosa viene setta la variabile copia a false poi viene eseguito un ciclo su tutti gli utenti per cercare un utente con lo stesso username dell’utente che si sta registrando.

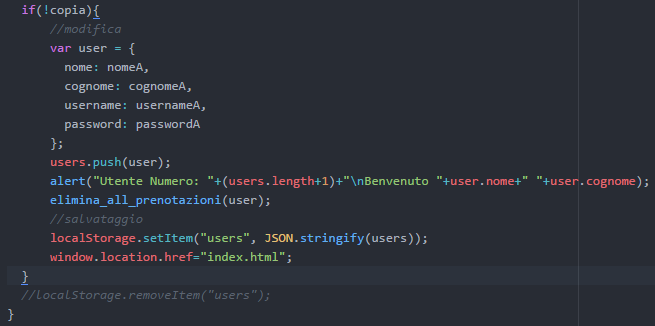
In caso di corrispondenza viene visualizzato su schermo dall’utente una finestra con la scritta: “Utente Esistente, inserire un altro username” e copia viene settato a true, altrimenti rimane false.

Infine la funzione restituisce la variabile copia.



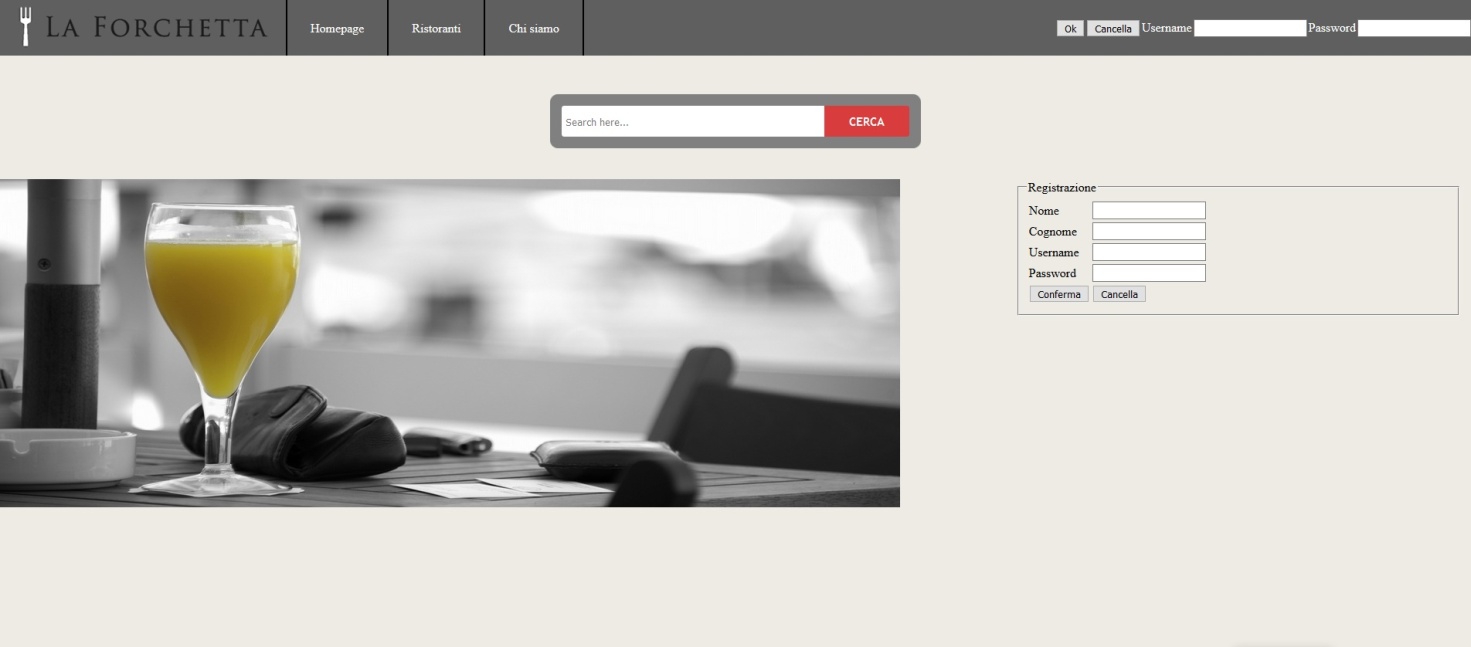
Proseguendo per la funzione principale crea\_user nel caso la variabile copia sia settata a false, quindi non siano state trovate corrispondenze riguardanti l’username, viene creato un oggetto user composto dai campi: nome, cognome username e password, e viene inserito nell’array users.

In sostanza viene creato il nuovo utente vero e proprio mediante l’oggetto user che lo rappresenta.



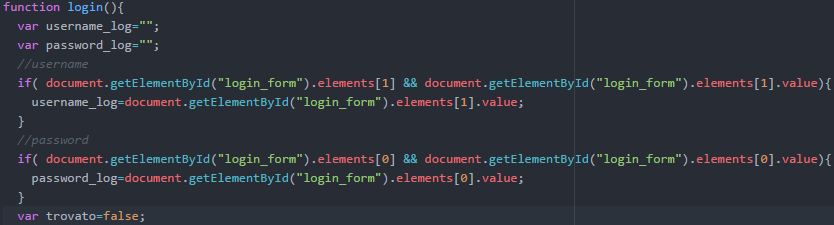
Il **Login** è presente sempre nella stessa pagina di Hompage per gli utenti del sito non ancora loggati, si trova in alto a destra nella navbar e vengono chiesti come di consueto Username e Password.



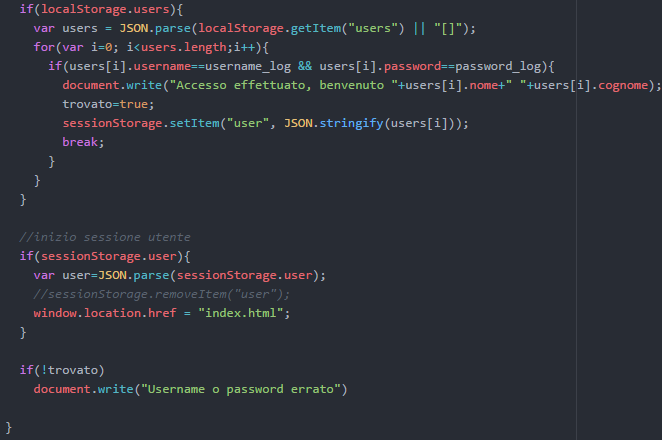


Cliccando il pulsante di Conferma viene invocata la funzione login().

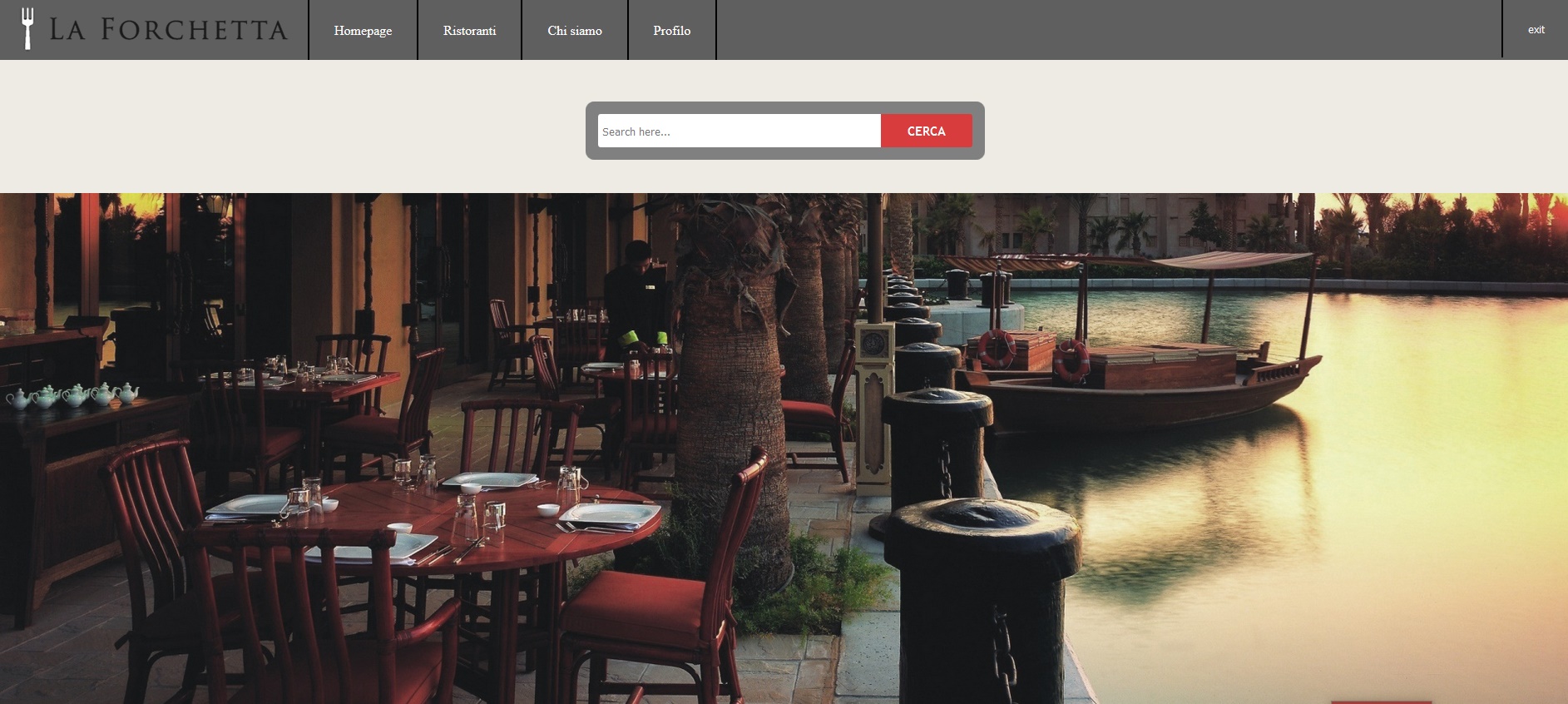
Come per la funzione crea\_user la prima parte della funzione login consiste nell’indicizzare in delle variabili, username e password inserite dall’utente e la variabile trovato a false (variabile booleana che rappresenta se è stata trovata una corrispondenza tra account esistenti e username/password inseriti).



Poi di seguito una semplice ricerca per corrispondenza su tutti gli utenti presenti nel sito, appena viene trovata una corrispondenza viene aperta una sessione nella quale viene salvata la variabile utente e viene ricaricata la Hompage.



Una volta loggati la Hompage risulta diversa, non appare più il form di registrazione a destra, ma una grande immagine centrale.



Nella navbar viene eliminato pure la parte relative dal login sostituito da una voce profilo che porterà al profilo dell’utente loggato e un pulsante “exit” per effettuare il logout dal proprio account.

**Ricerca**

Tutte le pagine del sito, a parte la pagine Profilo, presentano la barra di ricerca nella quale è possibile cercare ristoranti per nome, tipologia o area geografica.

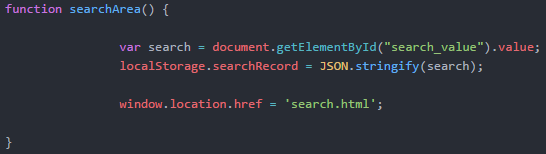


Oltre la barra di ricerca nella navbar è presente una sezione “Ristoranti” in cui vengono restituiti tutti i ristoranti del sito come risultato della ricerca.



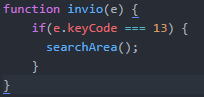
Sono presenti due metodi di ricerca nel sito:

* Premendo il bottone cerca, verrà chiamata la funzione searchArea(), in cui viene salvato all’interno del localStorage (localStorage.searchRecord) quello che l’utente ha cercato all’interno della search box, e subito di seguito viene caricata la pagina “search.html”.



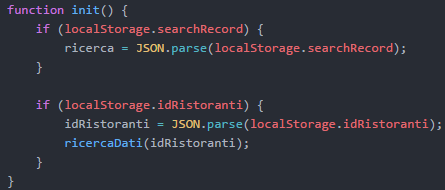
* Premendo il tasto invio, è stata implementata una funzione in cui al premere un qualsiasi tasto all’interno della search box viene richiamata la funzione invio(e) dove e rappresenta il tasto premuto.

Se il tasto premuto corrisponde a “Invio” verrà a sua volta chiamata la funzione searchArea().



Dopo aver eseguito la ricerca si aprirà la pagina search.html nella quale viene precaricata la funzione init().

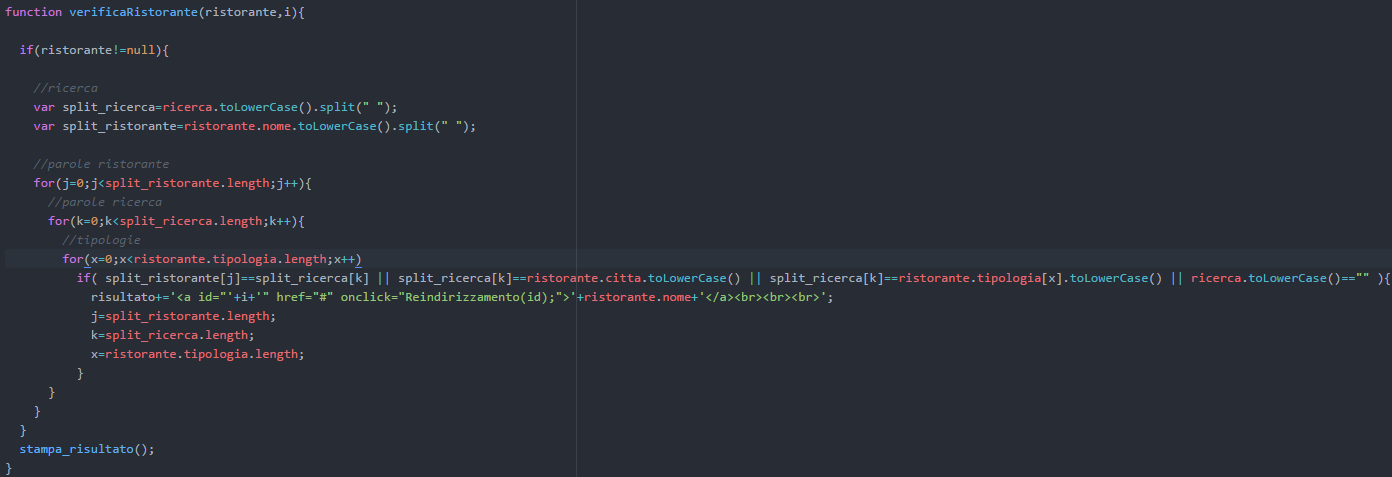
In questa funzione viene semplicemente salvato in una variabile globale il contenuto di localStorage.searchRecord e richiamata la funzione ricerca Dati(idRistoranti), dove idRistoranti rappresenta il vettore con inseriti tutti gli id dei ristoranti presenti sul sito.



La funzione ricerca Dati stabilisce una connessione con l’API per il prelievo dei dati riguardante i ristoranti.

I dati dei ristoranti verranno prelevati uno ad uno e per ogni dato prelevato verrà richiamata la funzione verificaRistorante() con i parametri ristorante, che rappresenta l’oggetto ristorante ed i, che rappresenta l’id del ristorante iesimo.

In verificaRistorante avviene una ricerca per corrispondenza tra la variabile ricerca, contenente la/le parole ricercate dall’utente nella serch box, e nome, tipologia o città contenuti nell’oggetto ristorante.

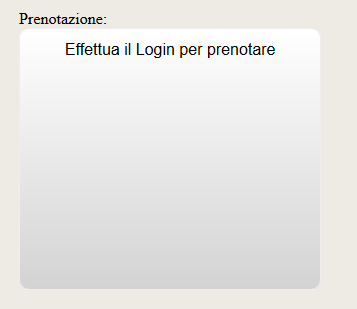


Il risultato sarà una serie di nomi e link dei ristoranti per cui si è trovata una corrispondenza, questi inseriti in un div al momento della chiamata della funzione stampa\_risultato().

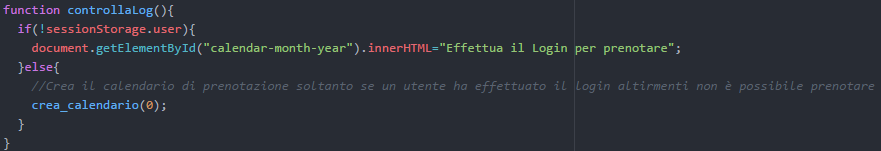


**Prenotazione**

Le prenotazioni a differenza della visualizzazione dei ristoranti può essere fatta solamente per gli utenti regolarmente registrati e loggati, infatti è stato posto un controllo che pone un blocco alle prenotazioni per gli utenti non registrati.



A livello di codice viene controllato se è presente la variabile user all’interno del sessionStorage.

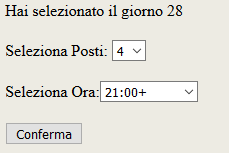


Mentre per gli utenti loggati, viene visto un calendario mensile con le date in cui è possibile prenotare, il calendario è composto da 3 mesi a partire dal giorno attuale (quindi settembre, ottobre e novembre in questo caso).

Ovviamente è posto un controllo che non permette di prenotare nei giorni già passati.



Selezionato il giorno si aprirà sotto il calendario un piccolo menù che impone l’inserimento dei dati di prenotazione, quindi numero di posti (max 8) e ora (2 fascie di orari disponibili) entrambi inseribili tramite un menù a tendina, questo per minimizzare la possibilità di errore da parte degli utenti.



Confermata la prenotazione, se è andata a buon fine, verrà visualizzato un messaggio in rosso sotto al menù come conferma dell’avvenuta prenotazione.



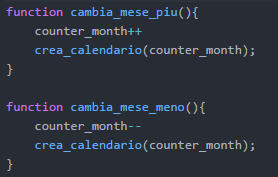
Nel caso in cui non sono più disponibili posti nel ristorante viene visualizzato un messaggio d’errore.



Riguardo al codice il calendario è creato con la funzione crea\_calendario(x), dove x è il parametro che setta il mese attuale e serve per scorrere di mese in mese (settembre-ottobre-novembre ecc.), infatti nella parte alta del calendario, sulla destra del mese visualizzato sono presenti due pulsanti “>” e “<” avanti e indietro che permettono di spostarsi tra i mesi.



Questi due bottoni però hanno il vincolo di rimanere in un range di 3 mesi quindi avanti sparirà al terzo mese, mentre indietro sparirà al primo mese (ovvero quello attuale). Entrambi i bottoni chiamano due funzioni a specifiche cambia\_mese\_piu() e cambia\_mese\_meno().



La funzione crea\_calendario() al suo interno richiama get\_calender() funzione che si occupa di creare il calendario vero e proprio con i giorni relativi ad ogni mese.

Ognuno di questi giorni è composto da un bottone in cui all’onclick viene chiamata la funzione dettagli\_prenotazione(id) dove id rappresenta il numero del giorno cliccato.

Dettagli\_prenotazione() si occupa di creare il menù per effettuare la richiesta di prenotazione, quindi per l’inserimento del numero dei posti e dell’ora.

Nella parte inferiore di questo menù è presente un bottone Conferma che premuto invoca la funzione Prenota(giorno).



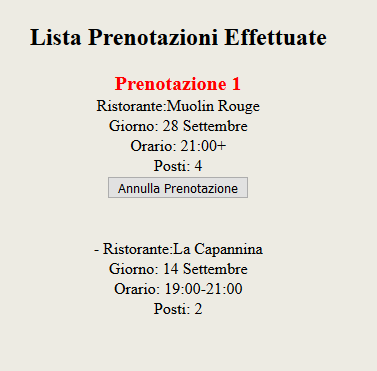
In prenota() viene controllato se ci sono ancora posti liberi per il numero indicato quel giorno, per questo compito viene chiamata la funzione conta Posti().

Nel caso in cui ci sia disponibilità per quel numero di posti viene creata la prenotazione vera e propria che consiste nella creazione di un oggetto prenotazione composto dagli attributi: utente, ristorante, mese, giorno, orario, posti e id.

Questo oggetto prenotazione viene inserito nel localStorage in un array di oggetti prenotazione che costituiscono il sistema cardine per il funzionamento del meccanismo di prenotazione.

Tutte le prenotazioni effettuate sono consultabili sulla propria pagina del profilo cliccando l’analogo bottone “Profilo” presente nella navbar.

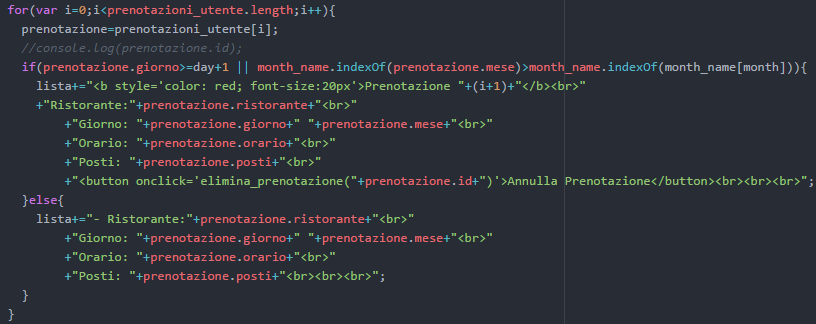
È possibile inoltre annullare le proprie prenotazione, fatte ad esempio per errore, l’unico vincolo sull’eliminazione della prenotazioni è che funziona solo se annullato 24 ore prima della prenotazione, in caso contrario non sarà possibile annullare la prenotazione ed entrerà nello storico prenotazioni dell’account.



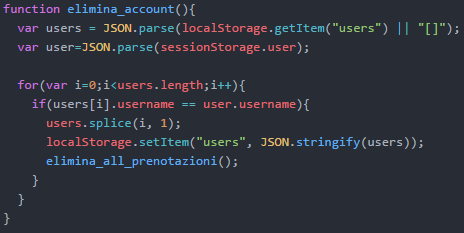
Sempre nella pagina profilo è presente sulla sinistra un bottone “Elimina Account”, che permette di eliminare definitivamente il proprio account e di conseguenza tutte le prenotazioni effettuate con quell’account.



A livello di codice appena viene caricata la pagina Profilo dell’utente viene invocata la funzione lista\_prenotazioni() che esegue una ricerca tra tutte le prenotazioni in memoria e cerca quelle corrispondenti all’utente loggato.



Invece cliccando il pulsante a destra della pagina profilo “Elimina Account” viene invocata la funzione elimina\_account(), che si occupa di cercare ed eliminare l’utente dall’array di utenti in memoria e di richiamare la funzione elimina\_all\_prenotaizoni()



che elimina tutte le prenotazioni del dato account facendo una ricerca tra tutte le prenotazioni.



**Conclusione**

Questi sono alcuni degli aspetti principali legati al progetto “La Forchetta”, non è stato riportato il codice nella sua interezza ma piccole parti significative che vanno a focalizzare le principali scelte effettuate durante la progettazione del sito.