Lorenzo Tabasso Matricola: 812499

Relazione del progetto di Tecnologie Web

1. Analisi generale

Tema del sito

Kinon è un sito e-commerce di un negozio di Torino che tratta oggetti fotografici.
I brand principali trattati da Kinon sono Nikon, che produce macchine fotografiche, Nikkor, che produce le ottiche per macchine fotografiche marchiate Nikon, e Manfrotto, che produce treppiedi. Il sito web di Kinon è stato progettato per offrire ai suoi utenti la possibilità di sfogliare il catalogo dei prodotti, ed eventualmente, poterli acquistare in remoto, con la possibilità di venire a ritirarli in negozio. Infine, il sito è stato progettato in inglese per permettere a tutti gli utenti di tutte le nazionalità di poter usufruire dei servizi offerti.

Sezioni principali e Schema di navigazione di base

Le sezioni principali sono così articolate:

1. **Login**: l'utente come prima cosa quando raggiunge il sito web di Kinon deve eseguire il login. Se l'utente non dispone di un account, potrà cliccare sulla voce "Registrati" per ottenerne uno, e procedere successivamente al login. Dopo che l'utente avrà effettuato il login inserendo le proprie credenziali, sarà caricata la homepage, dove l'utente troverà una serie di menù per interagire con gli oggetti nel catalogo. I menù e le loro funzionalità sono elencati qui di seguito.

2. **Registrazione**: come già menzionato, un utente che non depone di un account, avrà la possibilità di crearne uno tramite la pagina di registrazione, accessibile dalla voce "Registrati" nella pagina di login.

3. **Homepage**: una volta effettuato il login, l'utente verrà reindirizzato nella homepage del sito. In questa pagina, l'utente potrà visualizzare i prodotti selezionando con un click la rispettiva **categoria**. Le categorie di oggetti che Kinon tratta sono 3: DSLRs (Digital Single Lens Reflex, corpi macchina), Lenses (ottiche per i copri macchina), e Tripods (Trepiedi per i corpi macchina). Se l'utente cliccherà su un prodotto per vedere le sue caratteristiche, ed eventualmente per acquistarlo, verrà automaticamente reindirizzato alla pagina del singolo prodotto, contenente tutti i dettagli su di esso.

4. **Prodotto**: in questa pagina l'utente potrà visualizzare le specifiche del prodotto e il suo prezzo. Al momento dell'acquisto del prodotto selezionato, l'utente dovrà inserire la quantità del prodotto selezionato. Cliccando sul bottone "Add to cart" l'oggetto selezionato verrà automaticamente inserito nel carrello nella quantità selezionata.

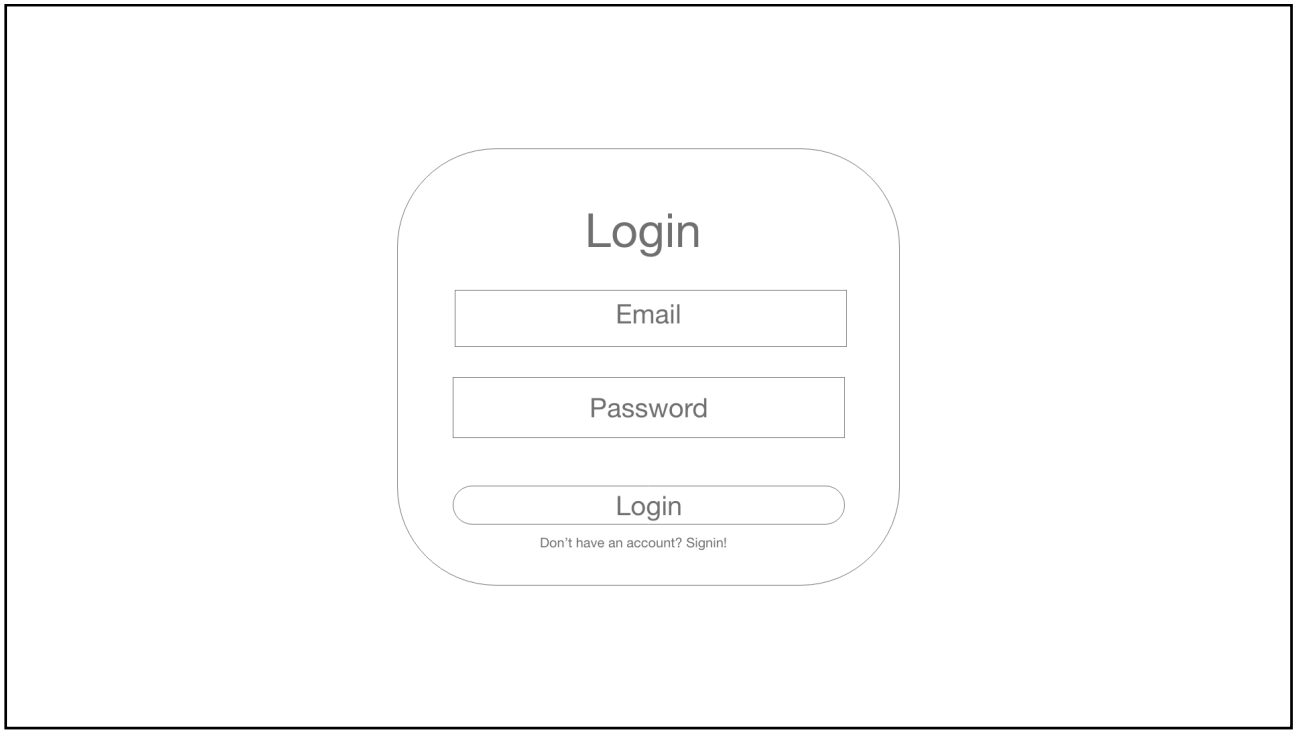
5. **Carrello**: in questa pagina verranno visualizzati tutti i prodotti che l'utente ha inserito nel carrello e desidera acquistare.

Lorenzo Tabasso Matricola: 812499

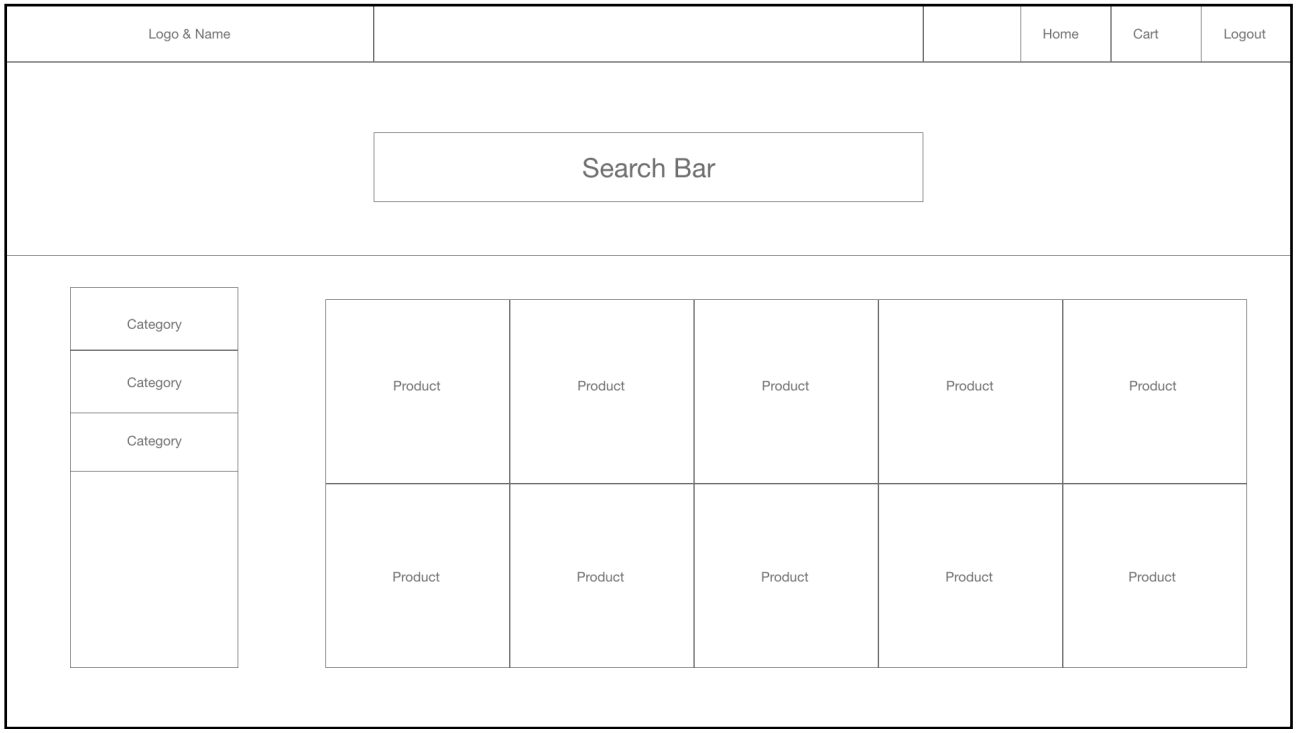
Mockup

I mockup sono disponibili anche nella cartella tweb/final_project/Relazione/Mockups

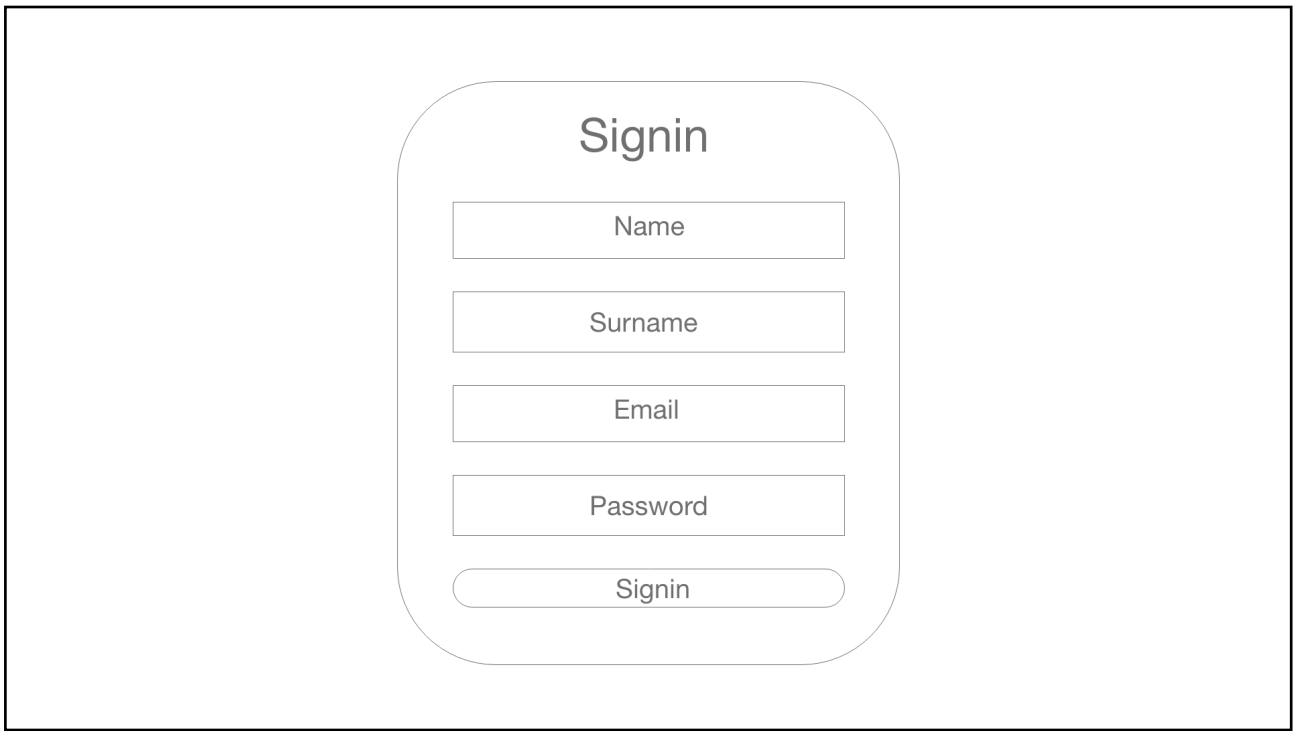
INDEX.PHP



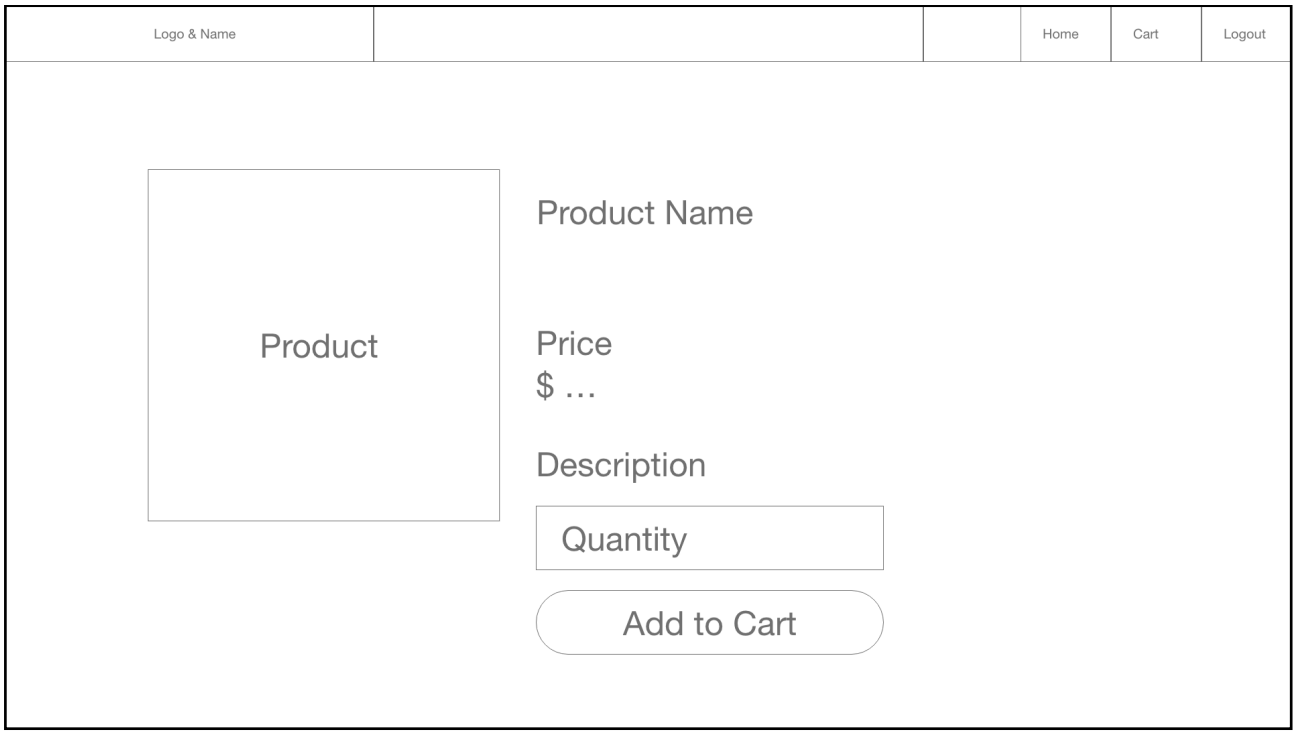
HOME.PHP



SIGNIN.PHP

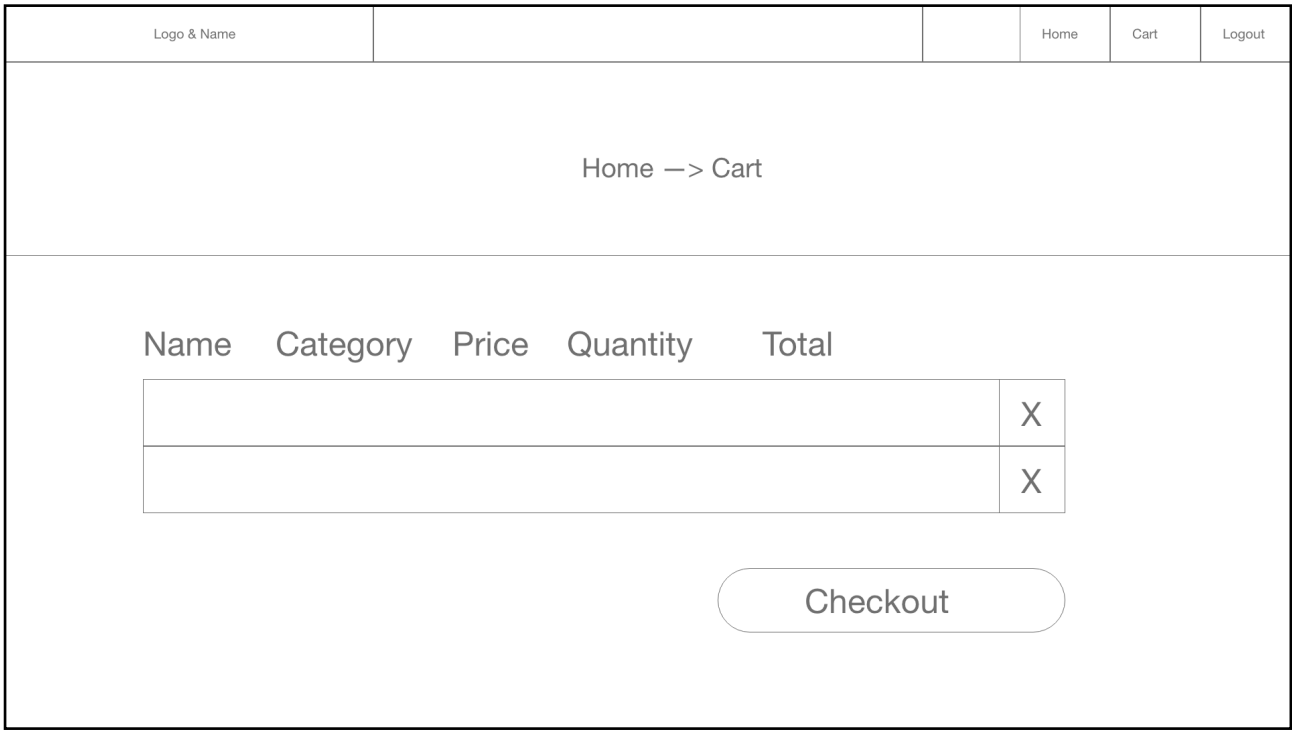


PRODUCT.PHP



Lorenzo Tabasso Matricola: 812499

CART.PHP



2. Parte Tecnica

Funzionalità

1. **Login**: il login è stato implementato nella pagina index.php, che viene caricata come prima pagina dal server. Una volta inserite le credenziali di accesso, cliccando sul tasto “Login”, verrà generata una chiamata Ajax dal client (implementato in client.js) al server, il contenuto della chiamata dal client al server sarà composto dall’email dell’utente e dalla sua password, inserite come campi in una richiesta POST. Il server, ricevuti questi dati, verificherà la presenza delle rispettive credenziali dell’utente nel database MySQL, e risponderà alla chiamata, in caso di successo esso consegnerà un file JSON contenente l’email dell’utente, mentre in caso di insuccesso il file JSON conterrà un errore. Il client, nel momento in cui la chiamata sarà terminata con successo, indirizzerà l’utente alla homepage (implementata dal file home.php), e in caso di insuccesso, stamperà a video il messaggi odi errore contenuto nel JSON.

2. **Logout**: Quando l’utente desidera eseguire il logout, premerà sul tasto “logout” nella barra superiore. All’interno di questo tasto, è presente un link alla pagina logout.php, la quale prima di effettuare il logout, eliminerà gli ordini ancora presenti nel carrello eseguiti dall’utente, e procederà alla chiusura della sessione, con le operazioni di session_unset() e session_destroy(). Infine, a logout avvenuto, l’utente verrà indirizzato nuovamente alla pagina di Login.

3. **Registrazione**: La registrazione è implementata sul server dalla pagina signin.php. In questa pagina è contenuto un form HTML, che una volta compilato correttamente (in particolare, la password deve essere lunga almeno 6 caratteri), invierà (cliccando sull’opportuno tasto “signin”) i dati del form (nome, cognome, email e password) tramite una chiamata Ajax al lato server-side, dove verranno elaborate dalla pagina submit.php, la quale controllerà se i dati inseriti sono validi. Se l’email dell’utente non è già stata inserita nel database, il server risponderà inviando un JSON contenente l’email dell’utente al client, altrimenti, il file JSON conterrà l’errore generato.

4. **Contenuto generato dall’utente**: il contenuto generato dall’utente corrisponde alla pagina cart.php, che consiste nel carrello. Quando un utente nella pagina del singolo prodotto clicca sul bottone “Add to cart”, l’oggetto selezionato viene aggiunto nella quantità inserita nel carrello. Tramite le due funzioni cart_count() e cart() contenute nel client.js, il DOM della pagina del carrello cart.php viene popolato con i dati selezionati dall’utente precedentemente. In questa maniera, quando l’utente vorrà visitare il proprio carrello, queste due funzioni avranno già caricato il DOM della pagina del carrello in modo adeguato.

Caratteristiche

1. **Usabilità e interazione/animazione**: Per raggiungere l’obbiettivo di una buona usabilità ho privilegiato l’uso di Bootstrap per la realizzazione dell’interfaccia grafica e delle animazioni. Ho inoltre aggiunto alcuni elementi stilistici (per esempio nella schermata di Login), visibili al fondo del foglio di stile style.css, e ho inserito alcune animazioni tramite diversi script Javascript.

2. **Sessioni**: Ho posizionato nel corretto ordine le istruzioni session_start(), session_unset(), e session_destroy() nei file: config.php, index.php, logout.php, e. cart.php .

3. **Interrogazione del database**: Ho deciso di implementare le interrogazioni al database tramite un oggetto PDO, con gli obiettivi di avere maggior chiarezza nella lettura del codice, e di essere protetto da attacchi di SLQ injection. E’ possibile trovare delle interrogazioni al database nei file product.php, logout.php, submit.php, action.php, deletecommand.php, e config.php .

4. **Validazione dati di input**: La validazione dei dati in input è eseguita dal file submit.php, il quale controlla che i dati inseriti dall'utente in fase di login e di registrazione siano dei dati validi e non esistano all'interno del database al momento dell'inserimento.

5. **Sicurezza**: Il sito è protetto da qualsiasi attacco di SLQ injection, poiché l'interrogazione del database è stata effettuata tramite l'uso di un oggetto PDO.

6. **Presentazione**: Ho implementato il sito in modo che avesse si presentasse in un layout responsivo, chiaro e leggibile.

Front-end

1. **Separazione presentazione/contenuto/comportamento**: Nello scrivere l'intero sito ho usato uno stile unobtrusive allo scopo di maggiore chiarezza e di bellezza del codice. Tutte le funzioni Javascript sono state inserite all'interno dei rispedivi file.js

2. **Soluzioni cross-platform**: Il sito web è perfettamente funzionante su svariate piattaforme di browsing, in modo particolare, esso è stato testato su Safari (versione 11.0.3), Firefox (versione 58.0.1) e Google Chrome (versione 63.0.3239.132). Unica funzionalità non mantenuta nell'ambiente cross-platform è la barra di navigazione superiore, contenente i link del menù di navigazione quali il tasto "Home", il tasto "Shopping Cart" e il tasto "Logout"

3. **Organizzazione file e cartelle di progetto**: Internamente, la cartella del progetto è divisa in varie directory, ordinate in base alla tipologia di file che contengono (ad esempio le cartelle css, img, js, ecc.). Unica eccezione: i file php, questi ultimi sono infatti suddivisi in 2 cartelle: "inc", che contiene il file di configurazione e il file he contiene tutte le funzioni comuni a tutti i file php, e "include" che contiene i file header.php, footer.php, e nav.php. I rimanenti file php sono stati invece posizionati nella cartella principale.

Back-end e comunicazione front/back-end

1. **Architettura generale classi/funzioni php**: Le funzioni PHP sono così di seguito articolate:

    1. ***Function.php***: contiene un unica funzione "output" la quale restituisce il risultato degli script server-side alla chiamata Ajax che ha richiesto i dati al server.

    2. ***Config.php***: al suo interno è presente un oggetto PDO il quale connette il server al database di MySQL.

    3. ***Submit.php***: contiene al suo interno due script di verifica delle credenziali dell'utente, uno per il login, il quale controlla che le credenziali esistano all'interno del database, e uno per la registrazione, che controlla che le credenziali scelte dall'utente non siano già presenti all'interno del database. Entrami gli script restituiscono al client tramite la funzione output un oggetto JSON contenente il l'email dell'utente in caso di script effettuato con successo, e un oggetto JSON con un messaggio di errore altrimenti.

    4. ***Action.php***: all'interno di action vi sono sei funzioni. Ognuna di queste funzioni restituisce un oggetto JSON usato da client.js per popolare il DOM della pagina corrispondente. Si veda come esempio la seconda funzione, alla linea 17 del file action.php, che ritorna alla corrispondente chiamata Ajax del client un oggetto JSON contenente tutti i dati di ogni prodotto presente nel database che rientra all'interno di una determinata categoria.

2. **Schema del database**: Il database è strutturato da 5 tabelle elencate di seguito, la chiave primaria di ogni tabella è sottolineata:

    1. **Category(id, name, icon)**: contenente le categorie dei prodotti.

    2. **Command(id, id_produit, quantity, dat, statut, id_user)**: dove ogni riga contiene i dati sullo stato di un di un ordine

    3. **Pictures(id, picture, id_product)**: contenente i path relativi del'immagine di un prodotto, e l'id del prodotto collegato a essa.

    4. **Product(id, id_category, name, description, price, id_picture, thumbnail)**: ogni riga contiene le informazioni di un prodotto

    5. **Users(id, email, firstname, lastname, password)**: ogni riga contiene le credenziali di un singolo utente.

3. **Condizioni di errore: come sono gestite**:
    1. In caso di credenziali di login errate, l'utente verrà avvisato da un messaggio di errore sotto il form di login nella schermata di login.
    2. In casi di credenziali di registrazione già esistenti, l'utente verrà avvisato da un messaggio di errore sotto il form di login nella schermata di login.

4. **Funzioni di callback lato javascript/Ajax**: Nel file client.js sono presenti le seguenti funzioni di callback:
    1. Funzione anonima (client.js, linea 15): essa effettua il parsing dei dati ottenuti tramite JSON e se non vi sono errori nelle credenziali id login, indirizza l'utente nella homepage, altrimenti inserisce nel div con id "display_error"il messaggio di errore.

    2. Funzione anonima (client.js, linea 38): in caso di successo, essa indirizza l'utente alla homepage

    3. Funzione anonima (client.js, linea 52): in caso di successo, essa utilizza i dati che ha ricevuto dallo script php server-side per popolare il DOM delle categorie all'interno della homepage.

    4. Funzione anonima (client.js, linea 65): in caso di successo, essa esegue il parsing dell'oggetto JSON ricevuto dallo script server-side, e lo utilizza per caricare il DOM dei prodotti nella homepage.

    5. Funzione anonima (client.js, linea 84): in caso di successo, essa esegue il parsing dell'oggetto JSON ricevuto dallo script server-side, e lo utilizza per caricare il DOM dei prodotti appartenenti a una determinata categoria nella hompage. Essa si attiva quando viene effettuato un click su una categoria di prodotti.

    6. Funzione anonima (client.js, linea 100): in caso di successo, essa esegue il parsing dell'oggetto JSON ricevuto dallo script server-side, e lo utilizza per caricare il DOM dei prodotti nella homepage il tuo nome corrisponde alla keyword digitata nella barra di ricerca.

    7. Funzione anonima (client.js, linea 123): in caso di successo, essa aggiorna il conteggio degli oggetti inseriti nel carrello dall'utente.

    8. Funzione anonima (client.js, linea 134): in caso di successo, essa esegue il parsing dell'oggetto JSON ricevuto dallo script server-side, e lo utilizza per caricare il DOM dei prodotti presenti all'interno del carrello.