

# Relazione Progetto TWEB 2022/2023

Robustelli Luca

## **Tema**

il tema del sito fa riferimento ad un e-commerce per la vendita di chiavi di attivazione per videogiochi .

Ci sono due tipologie di utenti principali che interagiscono con il sito web:

- Utente:colui che acquista le key dei videogiochi.
- Amministratore:aggiunge e rimuove giochi e key all'interno del sito.

## **Sezioni**

Il sito è diviso in queste macro sezioni:

-home:home page del sito internet in cui sono presenti le ultime aggiunte e i giochi più in tendenza del momento.

-login:pagina di login dove effettuare l'accesso

-register: pagina di registrazione

-shop:in base alla console scelta,viene visualizzata la lista dei videogiochi disponibili relativa ad essa

-profilo:pagina del profilo dell'utente dove è possibile visualizzare e cambiare le proprie informazioni(es:nome utente,password ecc), e visualizzare la propria lista di acquisti.

### Solo lato utente:

-cart:questa sezione rappresenta il carrello dell'utente che mostra tutti giochi in esso contenuti e da dove si puo' accedere alla sezione del pagamento

-payment:questa è la vera e propria pagina del pagamento dove alla fine si visualizzeranno le chiavi di attivazione dei videogiochi acquistati

### Solo lato amministratore:

-admin games:la pagina rappresenta la pagina dell'amministratore dove puo' inserire o eliminare nuovi videogiochi e gestirne le chiavi

-admin key:dalla sezione precedente si arriva qui grazie al tasto gestisci key di ogni videogioco nell'elenco .Questa pagina da possibilità di inserire nuove chiavi o eliminarne di esistenti per il gioco specificato

## **Funzionalità**

### -NavBar

è presente una navbar con all'interno delle voci che fanno riferimento alle piattaforme di gioco e alla

### -Registrazione

Per registrarsi va compilato un form con: email,password, nome e cognome.

Nome e Cognome possono avere solamente caratteri(maiuscole e minuscole)

mentre email deve rispettare,appunto,il formato d'indirizzo di posta elettronica(\*\*\*@\*\*\*.). Il

campo Password deve rispettare certi criteri: essere superiore o uguale ad 8 caratteri di

lunghezza ,contenere almeno una maiuscola,un carattere speciale e un numero.

Una volta compilato il form e premuto il tasto di registrazione,viene effettuato la validazione lato client dei dati e poi attraverso una \$\_POST spediti al server. Arrivati al server vengono effettuate ulteriori validazioni e se non vengono riscontrati problemi,l'utente viene registrato e reindirizzato alla pagina di home con nuove voci all'interno della navbar(profilo,logout e cart).

Login uno form con email e password per accedere. Anche in questo caso vengono effettuati controlli sia server-side e client-side.

### Shop

La sezione shop mostra tutte i videogiochi che sono disponibili per quella piattaforma,dove quest'ultima viene scelta attraverso le voci presenti in NavBar e mandata al server, tramite una chiamata \$\_GET ,passando come parametro una variabile platform. Inoltre,dopo aver ottenuto la lista di videogiochi,è possibile filtrare la lista con 3 tipologie di filtri:

-checkbox Console:una o più checkbox in base al numero di Console di una Piattaforma

-menu-dropdw Ordinamento:prezzo crescente,decrescente,Alfabetico -Genere:generi dei videogames

I filtri Console e Genere vengono ottenuti attraverso delle query SELECT che estraggono dalle tabelle Genere e Console tutte le tuple.

Ogni videogioco quando cliccato, porta alla sua pagina tramite una \$\_GET,passando come parametro l'id del gioco

### Profilo videogioco:

Questa è una pagina informativa che fornisce dettagli chiave su un videogioco, tra cui il suo nome, la piattaforma, il prezzo e una breve descrizione della trama o delle caratteristiche ed inoltre fornisce uno stato sulla quantità delle chiavi ancora disponibili.È possibile aggiungere al carrello solamente se si è accessi e se si è amministratori.

### Carrello:

Questa pagina viene creata in base alla variabile di sessione creata durante l'accesso e visualizza il carrello dell'utente con tutti i videogiochi aggiunti in precedenza e un riepilogo della somma totale che dovrà pagare,in questa sezione è possibili aumentare o diminuire la quantità o rimuovere un videogioco dal carrello.

### Acquisto Key:

Durante il processo di acquisto della chiave di gioco, ci saranno tre fasi da completare prima di poter visualizzare le chiavi acquistate. Nel primo passaggio, sarà necessario selezionare la carta di credito da utilizzare come metodo di pagamento. Senza prendere questa decisione, non sarà possibile procedere alle fasi successive.

Una volta scelta la carta di credito, si passerà al secondo passaggio, dove verrà riassunto il costo totale dell'acquisto, inclusi i dettagli della carta di credito utilizzata per il pagamento.

Infine, nella fase finale, saranno visualizzate le chiavi di gioco acquistate per ciascun videogioco. Da qui, l'unica opzione disponibile sarà tornare alla pagina principale del sito.

## Funzionalità Amministratore

**Gestione videogiochi:** Questa sezione è accessibile solo agli amministratori, permettendo loro di aggiungere nuovi videogiochi al catalogo. È possibile inserire le seguenti informazioni per ogni nuovo gioco: nome, descrizione, prezzo, genere e immagine. Inoltre, gli amministratori hanno la possibilità di eliminare un videogioco esistente dal catalogo o di accedere alla sezione key.

**Gestione Chiavi:** Qui, puoi aggiungere, eliminare e monitorare lo stato delle chiavi per il videogioco in oggetto. Puoi vedere chiaramente quali chiavi sono state acquistate e quali no, semplificando così la gestione dell'inventario delle chiavi del gioco.

## Usabilità

Per quanto riguarda l'usabilità si è cercato di mantenere uno stile verticale, in modo tale da mantenere sempre l'attenzione dell'utente al centro della pagina.

Le azioni eseguite sulla pagina, dall'utente o dall'amministratore, generano dei feedback attraverso risposte standardizzate inviate dal server, che vengono mostrate all'interno di swallow o attraverso il supporto di chiamate API, in alert.php. In questo file sono presenti le due funzioni, sendAlert e displayAlert, le quali settano e mostrano a video i feedback mandati dal server.

Si è preferito utilizzare un contrasto di colori di bianco (scritte) su sfondo scuri e in modo tale da permettere una facile lettura del testo.

## Interazione/Animazione

All'interno della pagina di home è presente un'animazione di un carosello, che in autonomia, mostra all'utente tutti i tre giochi che al momento sono in più voga (quelli più acquistati). Ogni 3 secondi il carosello mostra un gioco diverso e in qualsiasi momento l'utente può decidere quale gioco visualizzare attraverso l'utilizzo delle frecce che si trovano ai lati.

I tre giochi più in voga vengono estratti dal database per mezzo di una query eseguita sulla tabella purchase, che fa un group by sull'id dei giochi acquistati e count sui gruppi.

L'animazione del carosello è stata realizzata con l'utilizzo delle funzioni di fadeout e fadeIn di jQuery. Con fadeout eseguiamo un effetto di dissolvenza di uscita che dura 300ms, dopo di che chiamiamo una funzione (di callback) che rimuove l'immagine del gioco mostrata in quel momento. Invece, con fadeIn eseguiamo nuovamente un effetto di dissolvenza di entrata che dura 300ms, mostrando l'immagine del nuovo gioco.

Le informazioni dei 3 giochi vengono salvate all'interno di una classe chiamata .game che con l'utilizzo della funzione .find(.game) ci permette di estrarli.

Per quanto riguarda .slick quest'ultima è una funzione appartenente ad un Plugin di JQuery, chiamato appunto Slick, che aggiunge le due frecce ai lati del carosello permettendoci di scegliere quale gioco visualizzare.

## Sessioni

Il sito web utilizza delle variabili di sessioni per mantenere lo stato dell'utente.

Ogni qual volta che l'utente si registra o logga con il suo account, viene inizializzata una sessione e generate quattro variabili di sessione:

-user:

-cart:

-cards:

-purchase:

Ogni qual volta che l'utente effettua il logout, le variabili di sessione lato server vengono unsetate e distrutte.

## Interrogazioni del Database

Le chiamate al database vengono effettuate attraverso funzioni all'interno delle classi del model (visto che si è deciso di adottare una struttura MVC).

## Validazioni:

I dati inseriti nei form vengono validati due volte:

-lato client:utilizzando regex e required all'interno dei html

-lato server:con il supporto di una classe ServerValidator che utilizza metodi statici per controllare gli input dell'utente.

## Sicurezza

Per garantire la massima sicurezza dei dati nei nostri modelli, abbiamo utilizzato una crittografia avanzata AES-256-CBC per proteggere le chiavi dei videogiochi e i numeri CVV delle carte. Le password degli utenti sono state salvate nel database in forma di hash, rendendo difficile la loro decifrazione. Inoltre, abbiamo implementato una funzione "preventXSS" per proteggere le pagine web da attacchi XSS. Queste misure garantiscono la sicurezza dei dati e delle informazioni sensibili dell'applicazione.

## Presentazione

Nel nostro progetto, abbiamo organizzato gran parte dello stile del sito nel file 'header.css', anche se è importante notare che quasi tutte le viste hanno un proprio file CSS dedicato. Inoltre, abbiamo validato il nostro codice CSS per garantirne la correttezza.

## FRONT END

Il sito può essere utilizzato anche da utenti che non hanno un account registrato ma la loro navigazione è limitata dalla visita dello shop e dalla pagina di presentazione del videogioco,qual'ora si cerchi di accedere a view protette si viene automaticamente reindirizzati verso la pagina di login o alla homepage.Un caso particolare è la pagina del pagamento che dopo aver terminato tutti gli step , se nell'eventualità si clicchi sul tasto indietro del browser si tornerà alla pagina del carrello vuoto.

Ogni contenuto delle pagine html (escluso il login e la pagina di registrazione) viene generato da una chiamata ajax in modo da tenere le pagine sempre aggiornate qual'ora ci siano cambiamenti nel database.

Nel sito web, si fa uso di codice PHP in maniera limitata, fatta eccezione per l'inclusione dei file di intestazione (header.php) e di chiusura (footer.php), oltre che per l'implementazione di controlli per la protezione dell'accesso alle pagine protette. È importante notare che l'eccezione principale riguarda la pagina header.php, la quale contiene codice PHP finalizzato all'aggiornamento dinamico dello stato nella barra di navigazione, in base alle interazioni dell'utente e al suo ruolo.

## BACK END

Questa è una descrizione della comunicazione delle varie chiamate che servono per generare il contenuto nelle pagine di view.

Le risposte ajax ritornano sempre un output con formato:

- {'success':true , 'message':<messaggio da parte del server>,....(un numero di dati che cambia in base a quello che serve per aggiornare la vista)}
- {'success':false , 'message':<messaggio da parte del server>}

Di seguito elencherò tutte le chiamate post e get che vengono fatte su ogni pagina e userò il suffisso ajax per identificarle:

home\_view->home.js->home.php:

- GET:recupero informazioni per videogiochi piu'aquistati e le ultime aggiunte (ajax) login\_view->login.php:
- POST:elaborazione dati inseriti nel form di login se con successo reindirizzamento verso homepage altrimenti segnalazione errore
- register\_view->register.php:
- POST:elaborazione dati inseriti nel form di registraizione se con successo reindirizzamento verso homepage altrimenti segnalazione errore
- cart\_view->cart.js->cart.php o update\_cart.php:
- GET:recupero informazione carello dell'utente(axax)
- POST:cambiamenti all'interno del carello come rimozione o aumento della quantità(axax)
- shop\_view->shop.js->shop.php:
- GET:recupero informazioni dallo shop in base alla piattaforma scelta e/o ai filtri scelti.(ajax)
- gameProfile\_view->-game\_profile.js>game\_profile.php:
- GET:recupero informazioni dal videogioco scelto dallo shop(axax)
- POST:aggiunta al carello qual'ora il videogioco fosse disponibile(axax) profile\_view->profile.js->profile.php:
- GET:recupero informazioni dall'utente e (se utenti normali)visualizzazione acquisti e carte possedute(axax)
- POST:modifiche alle informazini personali e aggiunta di nuove carte di pagamento(axax)
- payment\_view->payment.js->buy.php:
- GET:recupero carte dell'utente(axax)
- POST:riepilogo informazione di acquisto e visualizzazione key comprate(axax)
- admin\_games\_view->admin\_games.js->admin\_games.php:
- GET:recupero informazioni di tutti i videogiochi all'interno del database e/o reindirizzamento verso pagina del videogiochi per gestione key (ajax)
- POST:eliminazione o inserimento di videogiochi(axax)

admin\_key\_view.php->admin\_key.js->admin\_key.php:

- GET:recupero informazioni delle chaivi e dello stato di esse e informazioni del videogioco in esame all'interno del database (ajax)
- POST:eliminazione o inserimento di nuove chiavi di attivazione (ajax)

### Struttura del database:

cards(id, card\_number, card\_holder\_name, card\_holder\_lastname, cvv, expiring\_date, user\_id)

carts(id, game\_id, user\_id, quantity)

games(id, title, platform, genre, price, description, cover) genre(genre)

```
keys_(id, key_value, game_id, acquired) platform(platform)
purchases(id, order_date, game_id, user_id, key_id, card_id)
users(id, email, password, first_name, last_name, role)
```