

Gianluca Napoletano

0334000098



Progetto di sistema informativo per negozio di videogiochi GN SHOP

1. Analisi e progettazione concettuale
2. Progettazione Logica.
3. Implementazione Sistema Informativo.

Pagina 4

Pagina 7

Pagina 14

Traccia d'Esame – Progetto di Sistema Informativo per Negozio di Videogiochi

Il progetto prevede la creazione di un sistema informativo completo per un negozio di videogiochi, il quale obiettivo principale, sarà quello di modernizzare la gestione delle vendite e del noleggio, sostituendo tutte le procedure manuali con un'applicazione web intuitiva ed efficiente. Il centro del sistema sarà un database SQL che raccoglierà tutte le informazioni necessarie per gestire le attività commerciali, storico di vendite e noleggi e i servizi offerti ai clienti. Il database conterrà un catalogo di tutti i videogiochi disponibili nel negozio, includendo le informazioni di base come titolo, genere e disponibilità, e anche i relativi prezzi di vendita e noleggio. Per ogni videogioco, sarà visualizzato lo stato in tempo reale: se è disponibile per la vendita, disponibile per il noleggio o non disponibile.

Gli utenti del sistema saranno clienti e dipendenti, i primi, ovvero i clienti del negozio, possono essere registrati come clienti standard o clienti fidelizzati tramite il programma fedeltà, i dipendenti; invece, saranno divisi in base al ruolo che ricoprono. I clienti che aderiranno al programma fidelity avranno accesso a prezzi di noleggio ridotti o promozioni dedicate. Saranno memorizzati dati personali come nome, indirizzo e-mail, numero di telefono e, per i clienti fidelizzati, il tipo di abbonamento attivo e la sua validità. L'accesso all'applicazione sarà possibile tramite i dati personali.

La gestione delle vendite e dei noleggi sarà una funzionalità centrale dell'applicazione, consentendo il monitoraggio del ciclo di vita delle transazioni: dalla selezione di un videogioco all'acquisto o alla restituzione di un noleggio, inclusi eventuali ritardi, visto che saranno salvate le date di inizio e fine noleggio. Il sistema manterrà uno storico di tutti gli ordini eseguiti dai clienti, facilitando il servizio clienti.

All'interno dell'app, sarà presente anche una sezione dedicata alle opinioni e alle recensioni sui videogiochi e sul servizio offerto dal negozio, nella quale i clienti potranno condividere commenti sui videogiochi acquistati o noleggiati, contribuendo così alla valutazione totale del negozio.

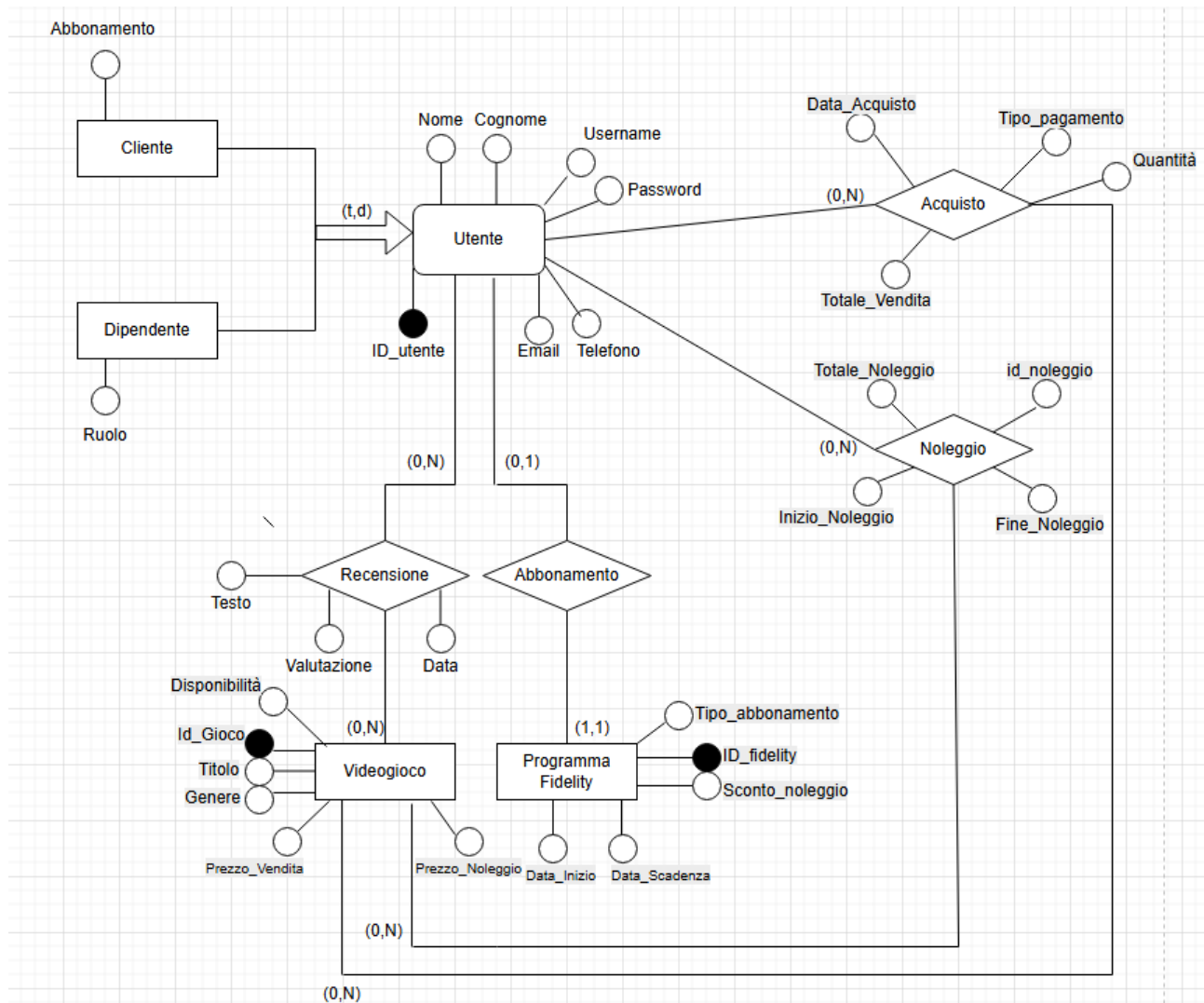
Nel caso in cui, il negozio organizzerà eventi speciali o promozioni (tornei o sconti stagionali), saranno gestiti tramite il sistema.

Verranno infatti pubblicate informazioni sugli eventi e sulle promozioni in un'apposita sezione. Un archivio storico conserverà la memoria degli eventi passati.

Infine, ci sarà una sezione dedicata ai dipendenti. A ciascun dipendente saranno associate le transazioni gestite (vendite e noleggi) e il sistema ne monitorerà il ruolo e l'organizzazione interna.

Obiettivi del Progetto

1. Analisi e progettazione concettuale



È stata progettata una gerarchia tra gli utenti del sistema, composta da un'entità padre Utente e due entità figlie: Cliente e Dipendente.

La specializzazione è di tipo totale e disgiunta, ovvero ogni utente è obbligatoriamente un cliente o un dipendente, ma non entrambi.

Vincoli della specializzazione:

- Copertura (Totale): ogni utente è o un cliente o un dipendente
- Disgiunzione: un utente non può appartenere a entrambe le categorie

Attributi comuni (entità padre Utente):

ID_utente, Nome, Cognome, E-mail, Password, Telefono

Attributi specifici:

Dipendente: ha un attributo "Ruolo"

Cliente: contiene un attributo o un collegamento che indica se è abbonato al programma fidelity

In questo caso, la distinzione tra Cliente Standard e Cliente Fidelity è stata assorbita all'interno dell'entità Cliente, evitando la creazione di due entità

separate. Questa scelta è stata fatta per semplificare il modello, dato che le differenze tra le due tipologie di clienti sono limitate e gestibili tramite attributi opzionali o relazioni (es. presenza o meno di un abbonamento).

Relazioni speciali:

- Solo i Dipendenti possono registrare vendite e noleggi
- Solo i Clienti con abbonamento attivo possono accedere al sistema di noleggio
- Tutti i clienti possono lasciare recensioni
- Le transazioni sono modellate con:
 - Vendita gestita da → Dipendente
 - Vendita effettuata da → Cliente

MODELLO ER RISOLTO

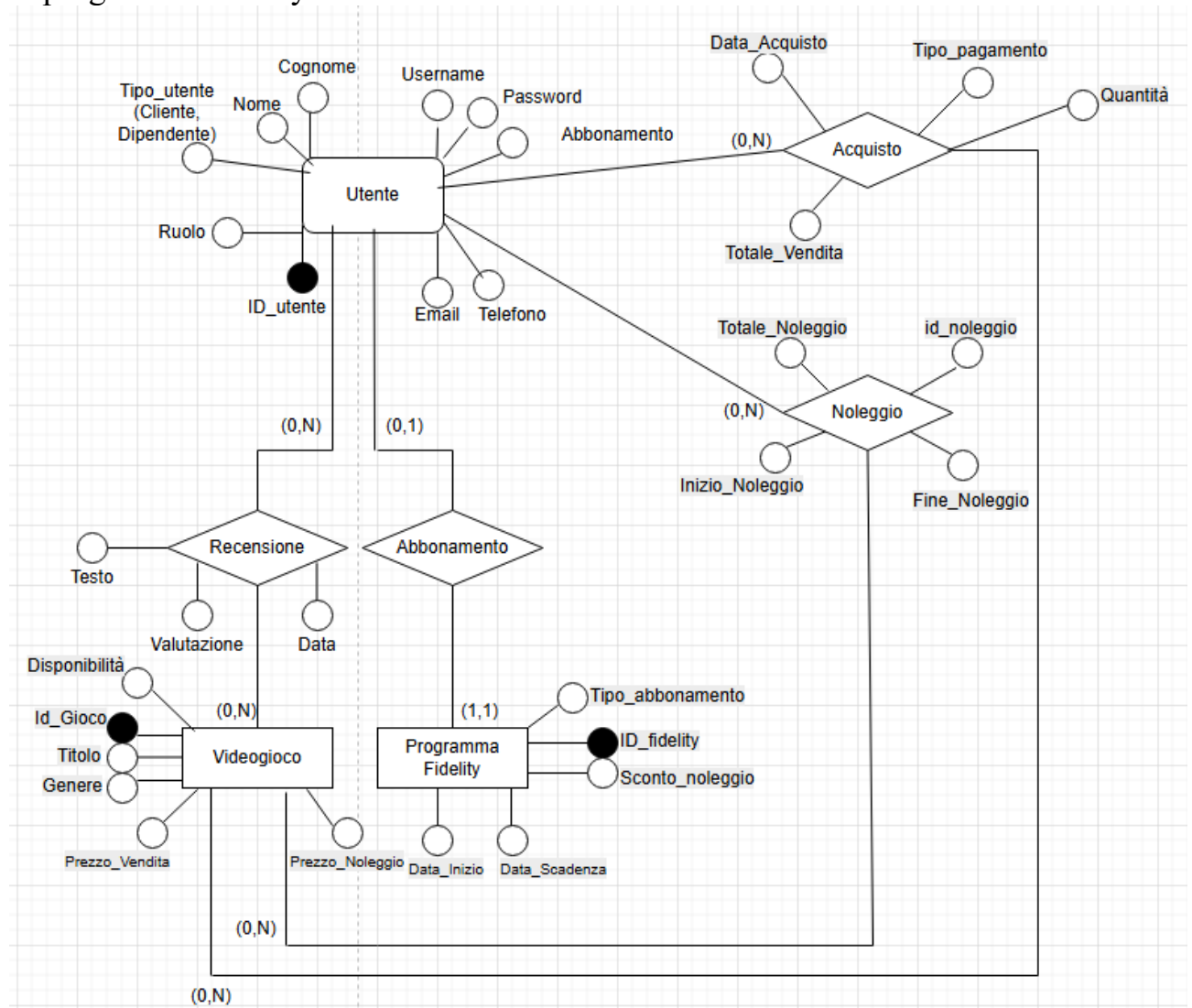
Per la generalizzazione, sono state valutate due strategie per rappresentare la gerarchia tra gli utenti del sistema: Cliente e Dipendente.

- Accorpamento in padre: tutti gli utenti vengono gestiti in un'unica tabella, con un attributo "ruolo" per distinguerli. È una soluzione semplice, ma può portare a campi non utilizzati (NULL) e minore chiarezza.
- Accorpamento in figlie: si creano entità separate per ogni tipo di utente, ereditando gli attributi comuni da Utente. È più chiaro, ma comporta una maggiore complessità nelle relazioni.

Nel progetto si è scelto un approccio intermedio: è stata mantenuta la specializzazione tra Utente, Cliente e Dipendente, ma la distinzione tra Cliente Standard e Cliente Fidelity è gestita all'interno dell'entità Cliente, tramite un attributo o una relazione con l'abbonamento.

Questa scelta semplifica il modello, evitando entità aggiuntive, ma consente comunque di distinguere i comportamenti dei clienti in base alla loro adesione

al programma fidelity.



2. Progettazione logica

Dal modello E-R è stato possibile derivare il modello logico relazionale:

Utente(ID_Utente(PK), Nome, Cognome, username, Password, Email, Telefono, Abbonamento, Tipo_cliente, ID_fidelity(FK))

Nome Campo	Descrizione	Tipo	Vincoli
ID_Utente	Identificativo univoco dell'utente	Integer (PK)	Primary Key, Not Null, Unique
Nome	Nome dell'utente	Varchar(50)	Not Null
Cognome	Cognome dell'utente	Varchar(50)	Not Null
Username	Nome utente per l'accesso	Varchar(50)	Unique, Not Null
Password	Password dell'utente	Varchar(128)	Not Null
Email	Indirizzo email	Varchar(254)	Unique, Not Null
Telefono	Numero di telefono	Varchar(20)	Not Null
Abbonamento	Indica se il cliente è fidelizzato	Boolean	Default: False
Tipo_cliente	Specifica il tipo di cliente (standard/fidelity)	Enum	Valori: ['Standard', 'Fidelity']
ID_fidelity	Riferimento all'abbonamento fidelity	Integer (FK)	Foreign Key, Nullable

Programma Fidelity (Id_abbonamento(PK), Tipo_abbonamento, Data_Inizio, Data_Fine, Sconto_noleggio, id_utente(FK))

Nome Campo	Descrizione	Tipo	Vincoli
ID_abbonamento	Identificativo univoco del programma fidelity	Integer (PK)	Primary Key, Not Null, Unique
Tipo_abbonamento	Tipo di abbonamento (es. base, premium)	Varchar(50)	Not Null
Data_Inizio	Data di attivazione dell'abbonamento	Date	Not Null
Data_Fine	Data di scadenza dell'abbonamento	Date	Not Null
Sconto_noleggio	Percentuale di sconto applicata ai noleggi	Decimal(5,2)	Not Null
ID_Utente	Riferimento all'utente che ha attivato l'abbonamento	Integer (FK)	Foreign Key verso Cliente, Not Null

Videogioco (Id_Gioco(PK), Titolo, Genere, Prezzo_Vendita, Prezzo_Noleggio, Disponibilità)

Nome Campo	Descrizione	Tipo	Vincoli
ID_Gioco	Identificativo univoco del videogioco	Integer (PK)	Primary Key, Not Null, Unique
Titolo	Titolo del videogioco	Varchar(100)	Not Null
Genere	Genere del videogioco (es. azione, sport, RPG...)	Varchar(50)	Not Null
Prezzo_Vendita	Prezzo per l'acquisto del videogioco	Decimal(6,2)	Not Null
Prezzo_Noleggio	Prezzo per il noleggio del videogioco	Decimal(6,2)	Not Null
Disponibilità	Stato di disponibilità (es. disponibile, non disponibile)	Enum o Boolean	Not Null

Acquisto (Totale, Data_Acquisto, Tipo_Pagamento, Quantità, id_utente, id_gioco)

Totale	Importo totale dell'acquisto	Decimal(8,2)	Not Null
Data_Acquisto	Data in cui è stato effettuato l'acquisto	Date	Not Null
Tipo_Pagamento	Metodo di pagamento utilizzato	Enum	Valori: ['Contanti', 'Carta', 'PayPal'], Not N
Quantità	Quantità di articoli acquistati	Integer	Not Null, CHECK (Quantità ≥ 1)
ID_Utente	Riferimento al cliente che ha effettuato l'acquisto	Integer (FK)	Foreign Key verso Cliente, Not Null
ID_Gioco	Riferimento al videogioco acquistato	Integer (FK)	Foreign Key verso Videogioco, Not Null

Noleggio (Totale_noleggio, Inizio_Noleggio, Fine_Noleggio, Id_utente(FK), Id_gioco(FK))

Totale_Noleggio	Importo totale del noleggio	Decimal(8,2)	Not Null
Inizio_Noleggio	Data di inizio del noleggio	Date	Not Null
Fine_Noleggio	Data di fine del noleggio	Date	Not Null
ID_Utente	Riferimento al cliente che ha effettuato il noleggio	Integer (FK)	Foreign Key verso Cliente, Not Null
ID_Gioco	Riferimento al videogioco noleggiato	Integer (FK)	Foreign Key verso Videogioco, Not Null

Recensione (Testo, Valutazione, data, Id_cliente, Id_gioco)

Testo	Testo della recensione	Text	Not Null
Valutazione	Valutazione data alla recensione	Integer	Not Null
Data	Data della recensione	Date	Not Null
Id_cliente	Riferimento al cliente che ha scritto la recensione	Integer (FK)	Foreign Key verso Cliente, Not Null
Id_gioco	Riferimento al videogioco recensito	Integer (FK)	Foreign Key verso Videogioco, Not Null

VINCOLI INTERRELAZIONALI

- Un cliente per poter effettuare ordini si deve registrare

→ Vincolo di esistenza (obbligatorietà)

- Un cliente può recensire un prodotto solo se lo ha acquistato o noleggiato

→ Vincolo di dipendenza (condizionale)

- Un cliente può recensire un evento solo se vi ha partecipato

→ Vincolo di dipendenza (condizionale)

- Un cliente può visualizzare i dettagli del proprio ordine solo se ha effettuato quell'ordine

→ Vincolo di autorizzazione (dipendenza da relazione)

- Solo i clienti con abbonamento attivo possono accedere al sistema di noleggio

→ Vincolo di autorizzazione (condizionale)

- Solo i dipendenti possono registrare vendite e noleggi

→ Vincolo di ruolo (autorizzazione)

- Ogni recensione deve essere associata a un videogioco e a un cliente

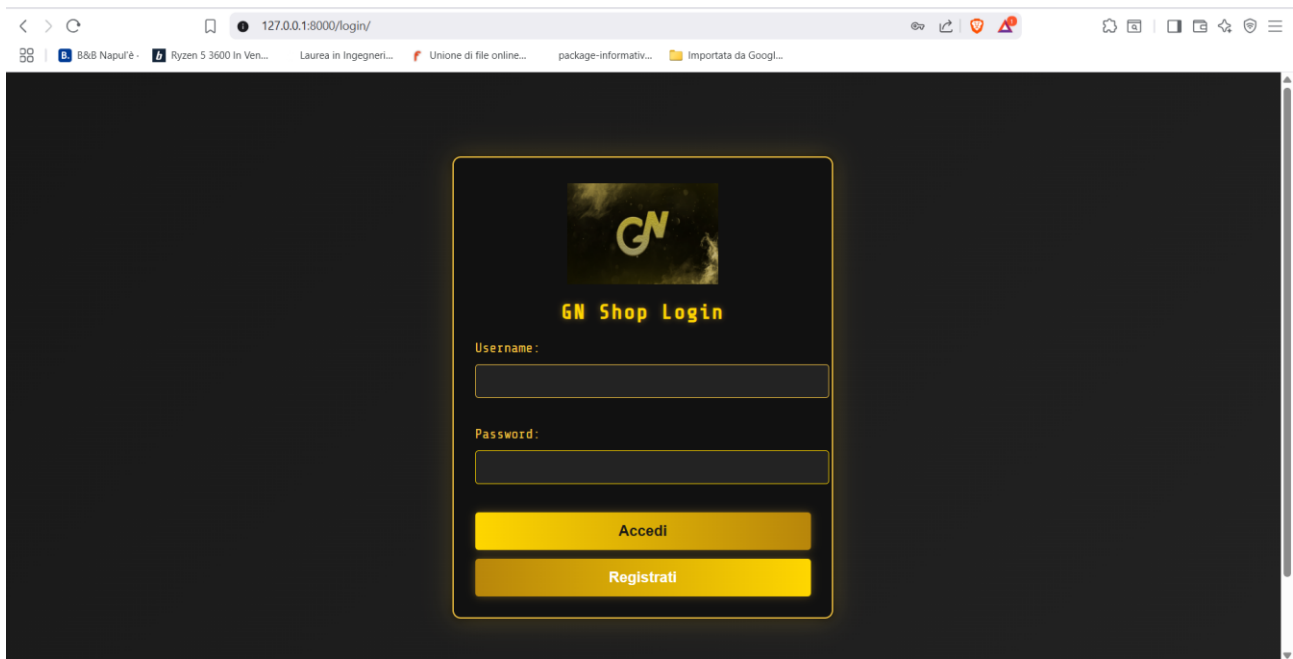
→ Vincolo di integrità referenziale

3. Implementazione del sistema informativo

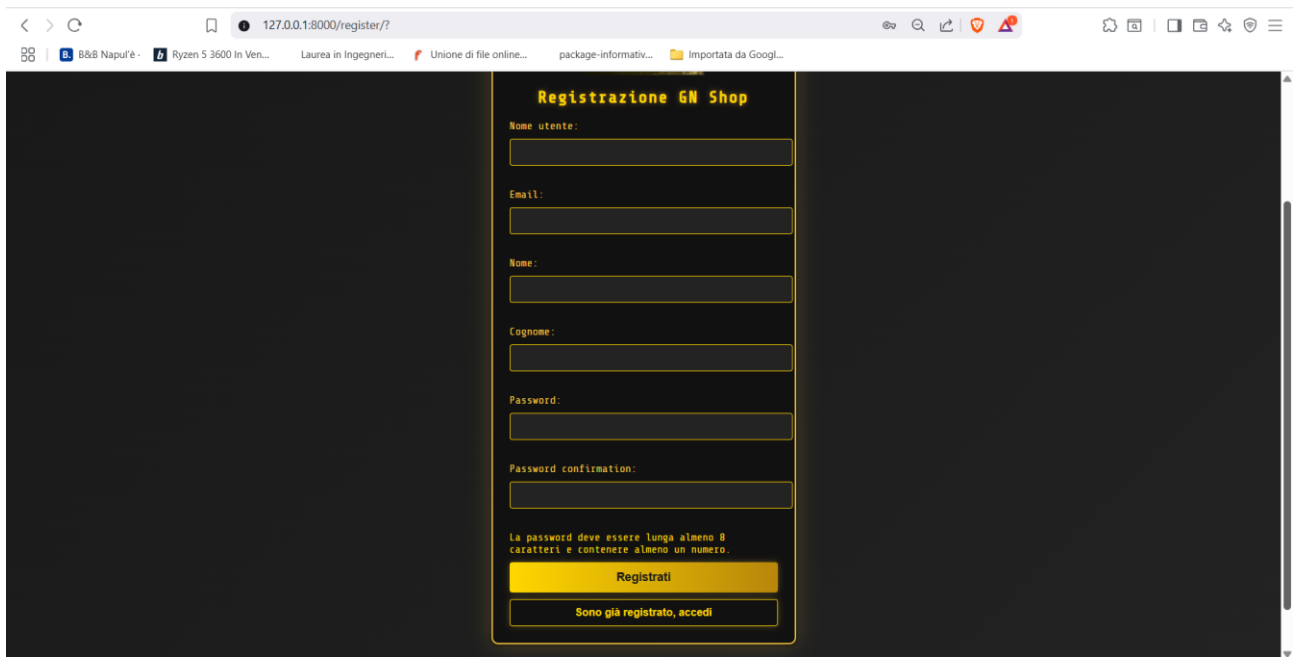
Utilizzando **Django**, è stata realizzata un'applicazione che permette la gestione dei dati modellati e l'accesso alle seguenti funzionalità:

- Visualizzazione del catalogo dei videogiochi.
- Registrazione e autenticazione degli utenti
- Registrazione e gestione di vendite e Noleggi (assegnazione e restituzione).
- Inserimento e consultazione di recensioni.
- Visualizzazione eventi e partecipazione agli eventi
- Storico degli ordini per utente.
- Accesso totale agli ordini, tramite sezione staff

SCHERMATA DI LOGIN

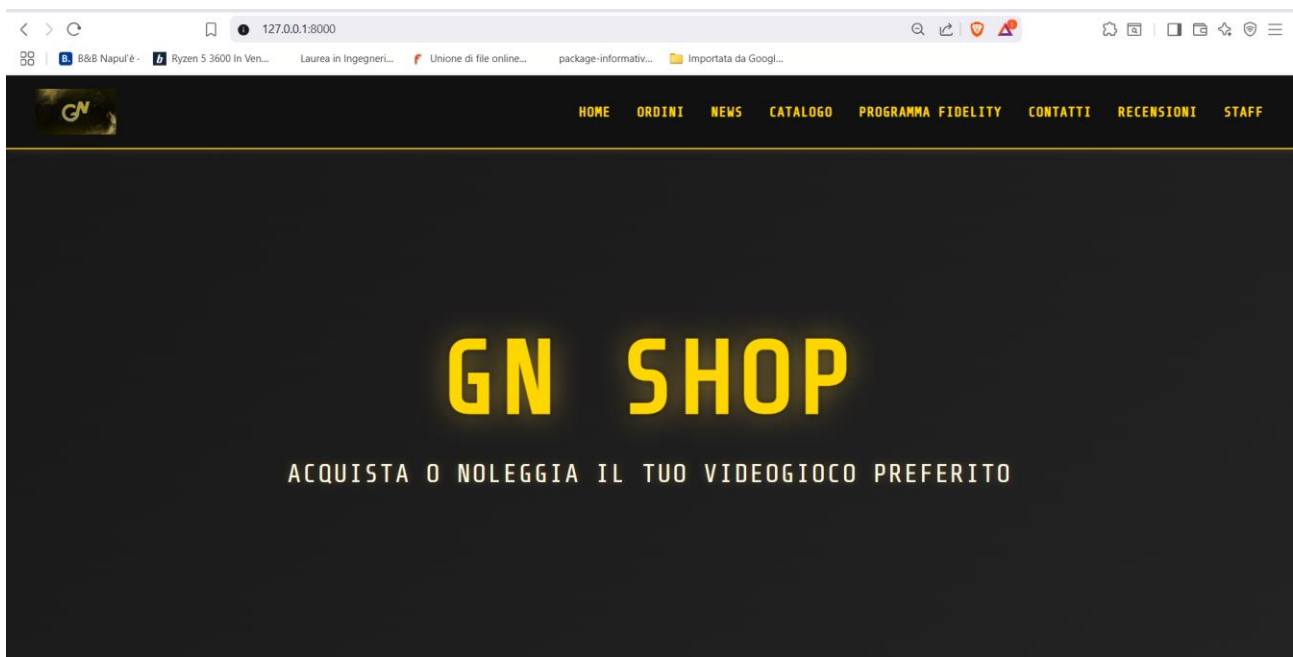


SCHERMATA DI REGISTRAZIONE

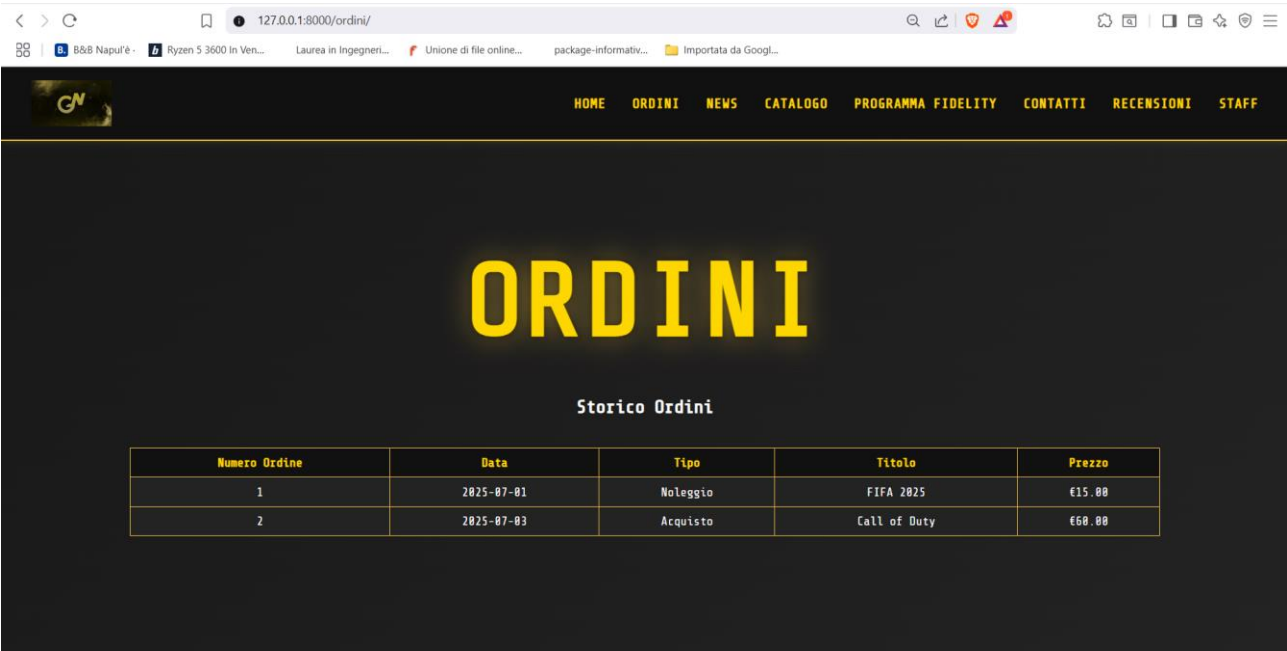


The screenshot shows a web browser window with the URL `127.0.0.1:8000/register/`. The page is titled "Registrazione GN Shop" and contains a registration form with the following fields: "Nome utente:", "Email:", "Nome:", "Cognome:", "Password:", and "Password confirmation:". Below the form, there is a note: "La password deve essere lunga almeno 8 caratteri e contenere almeno un numero." At the bottom of the form, there are two buttons: "Registrati" (highlighted in yellow) and "Sono già registrato, accedi".

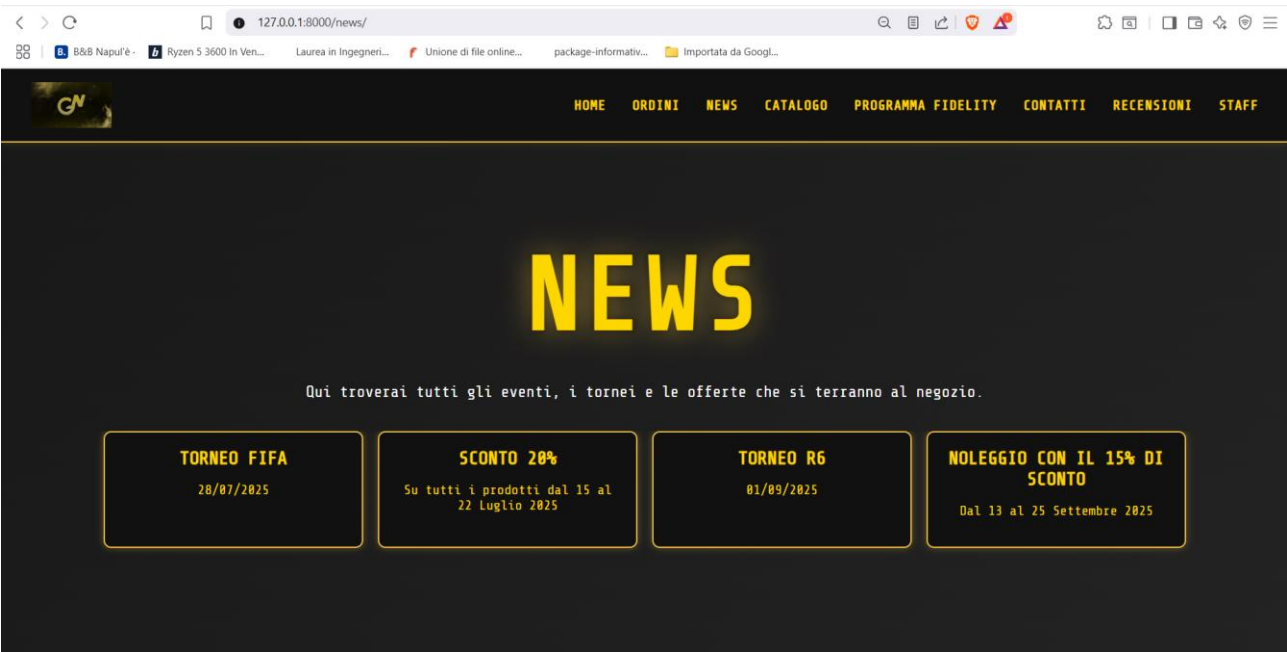
SCHERMATA HOME



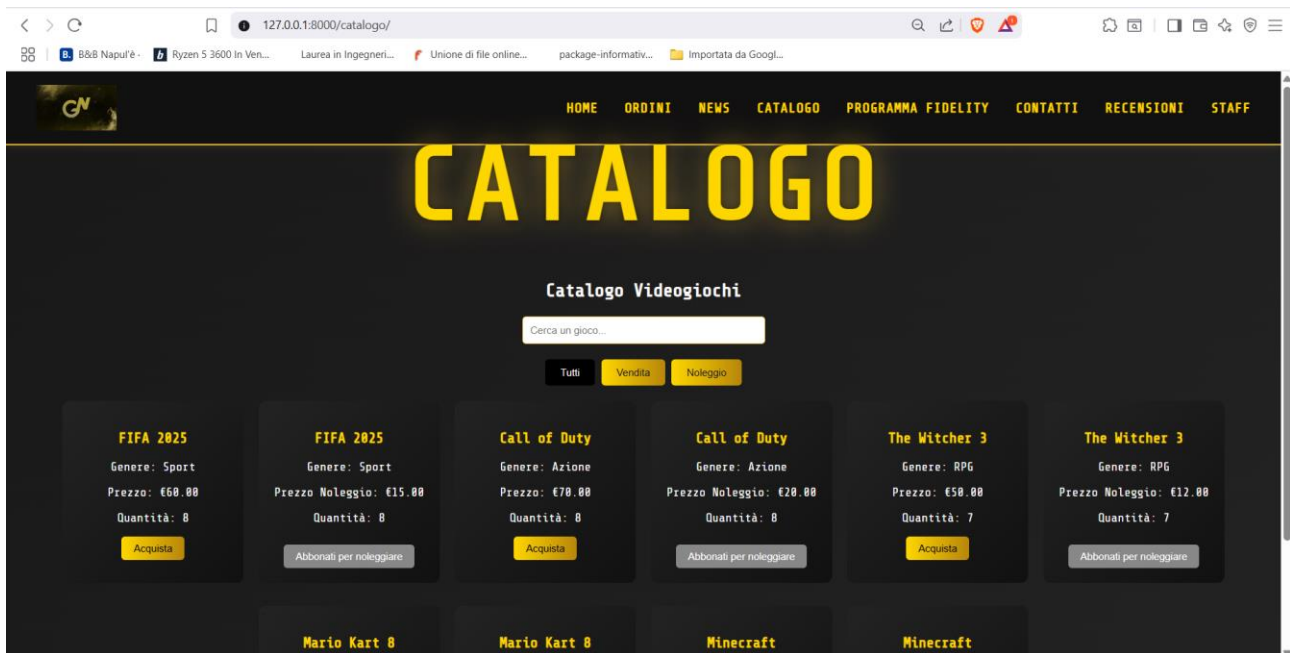
SCHERMATA ORDINI



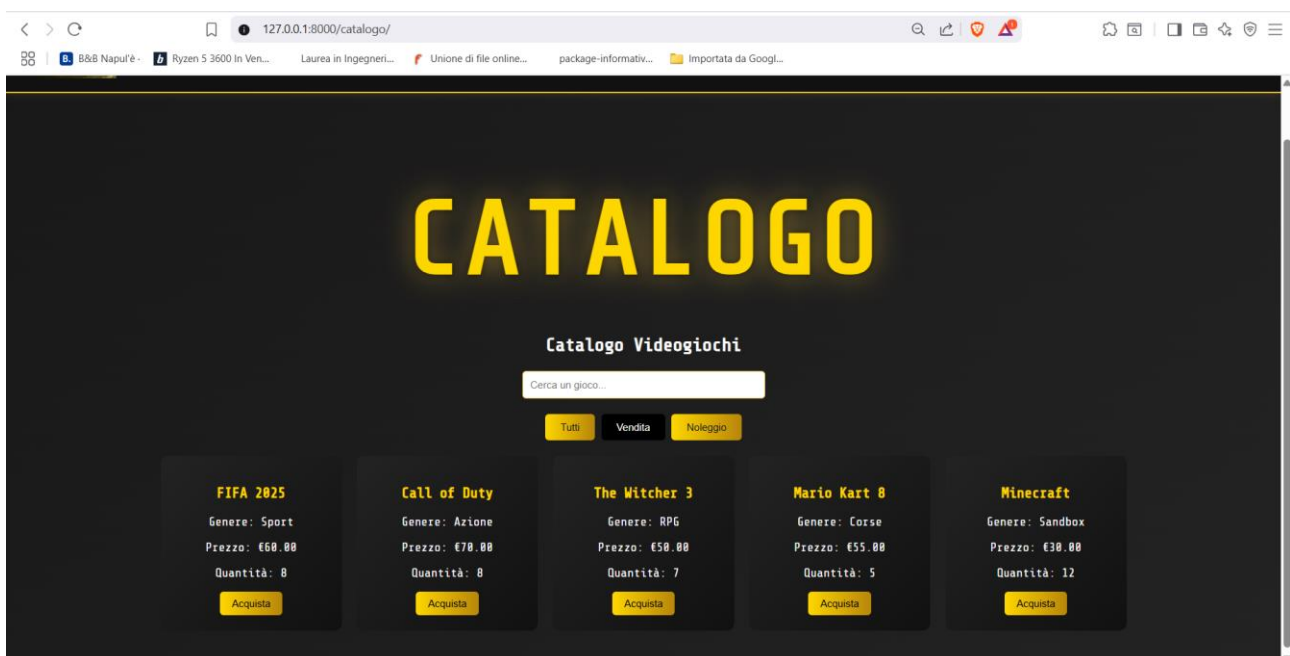
SCHERMATA NEWS



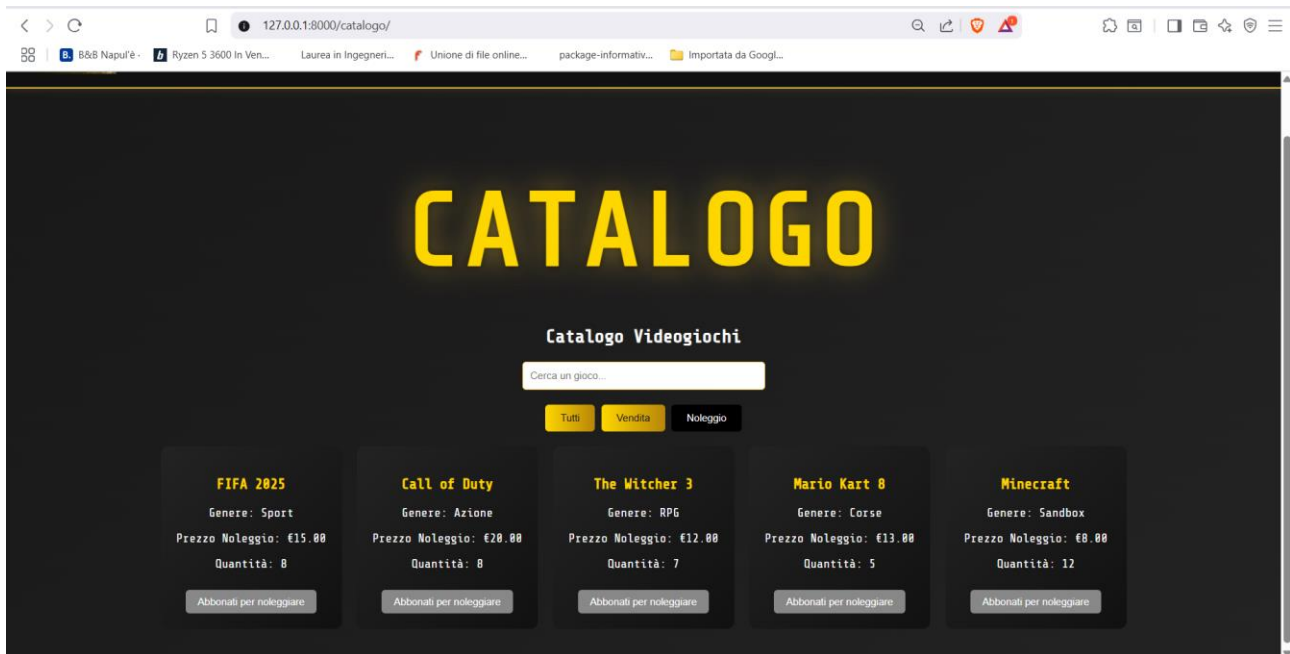
SCHERMATA CATALOGO



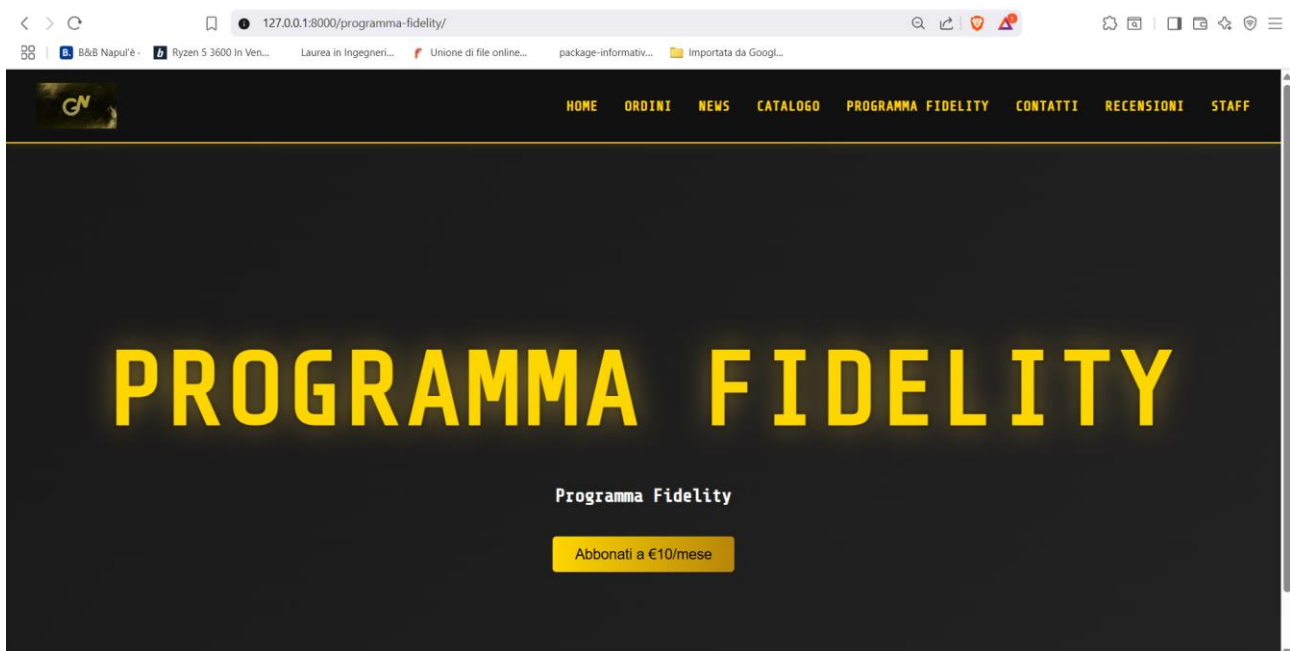
SCHERMATA CATALOGO FILTRO VENDITA



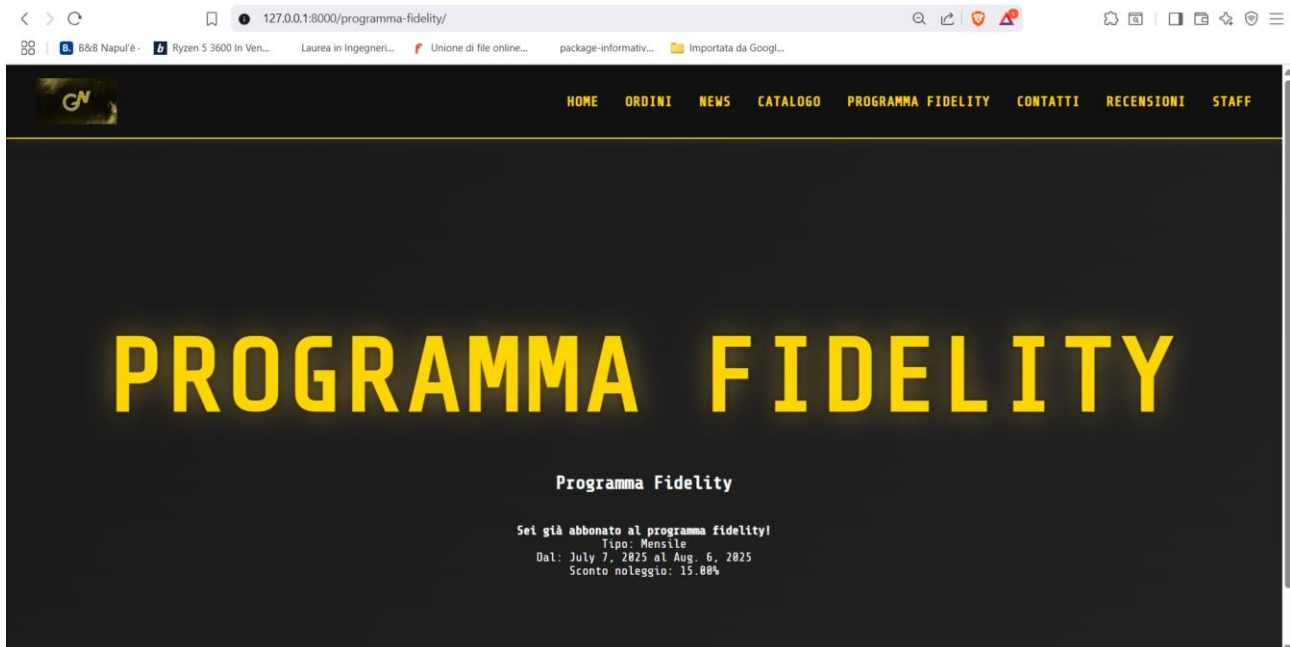
SCHERMATA CATALOGO FILTRO NOLEGGIO (UTENTE NON ABBONATO)



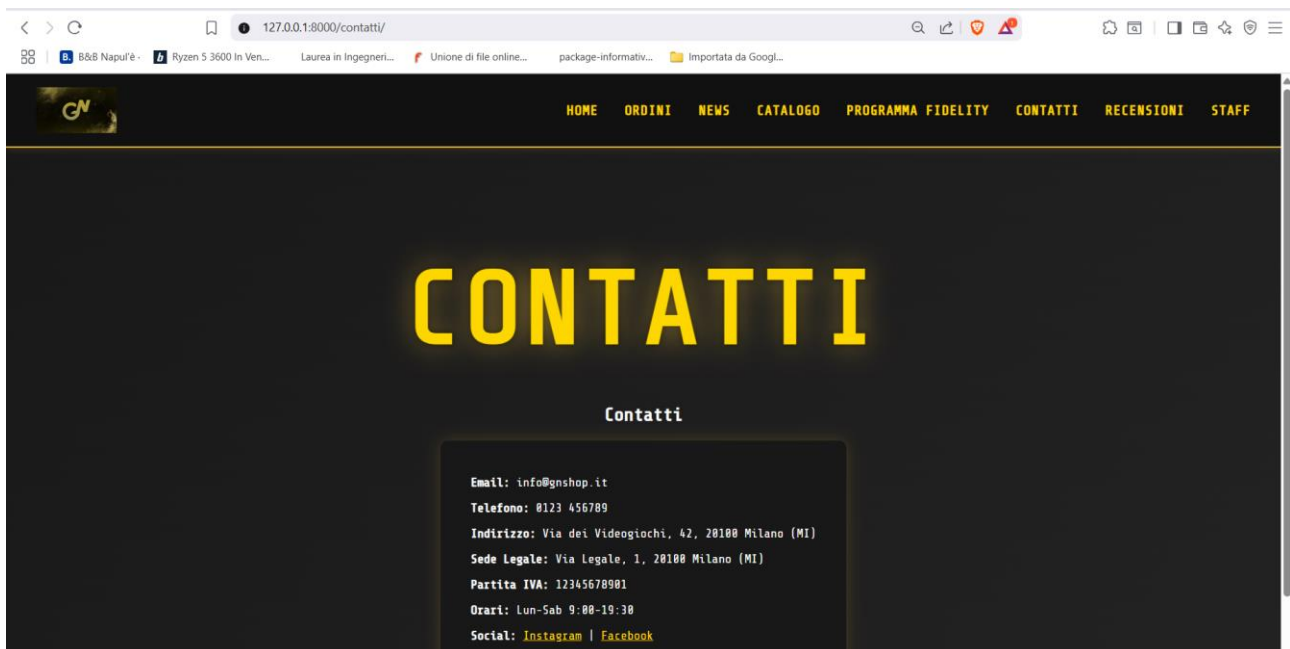
SCHERMATA PROGRAMMA FIDELITY NON ABBONATO



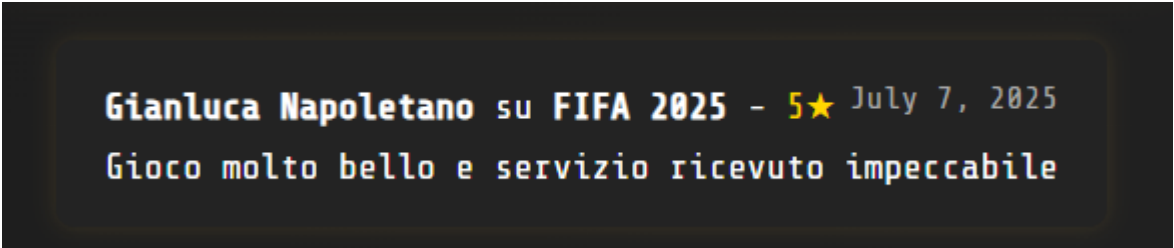
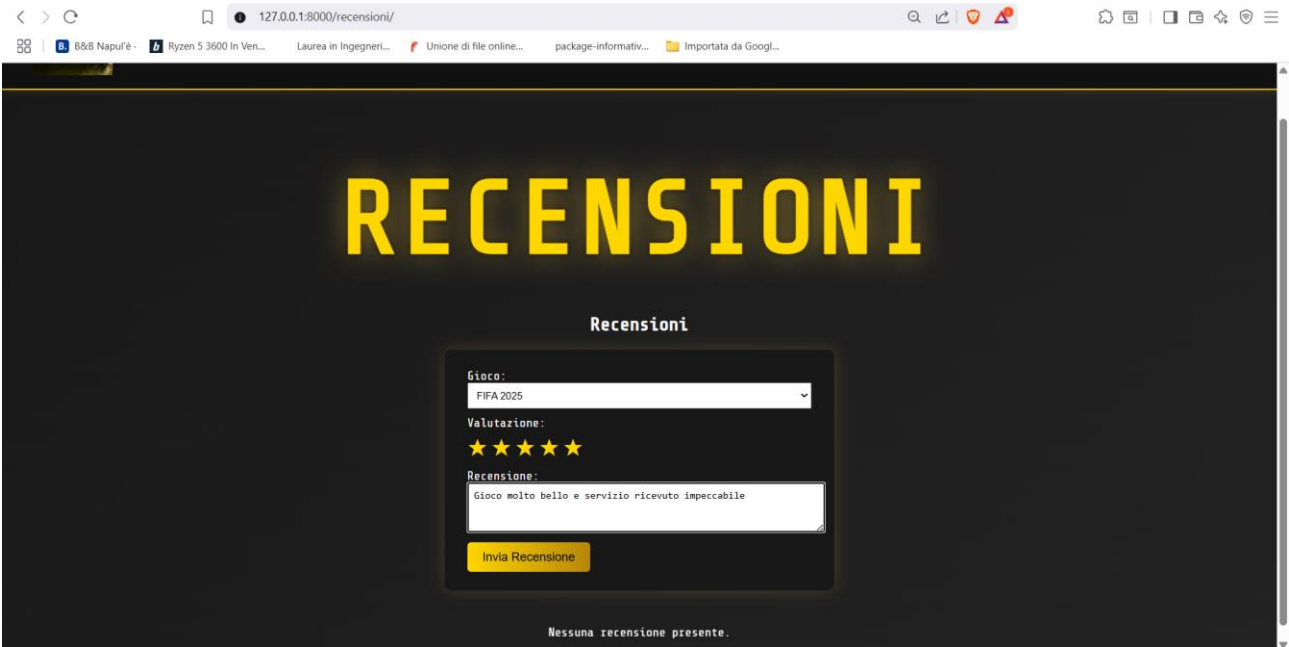
SCHERMATA PROGRAMMA FIDELITY ABBONATO



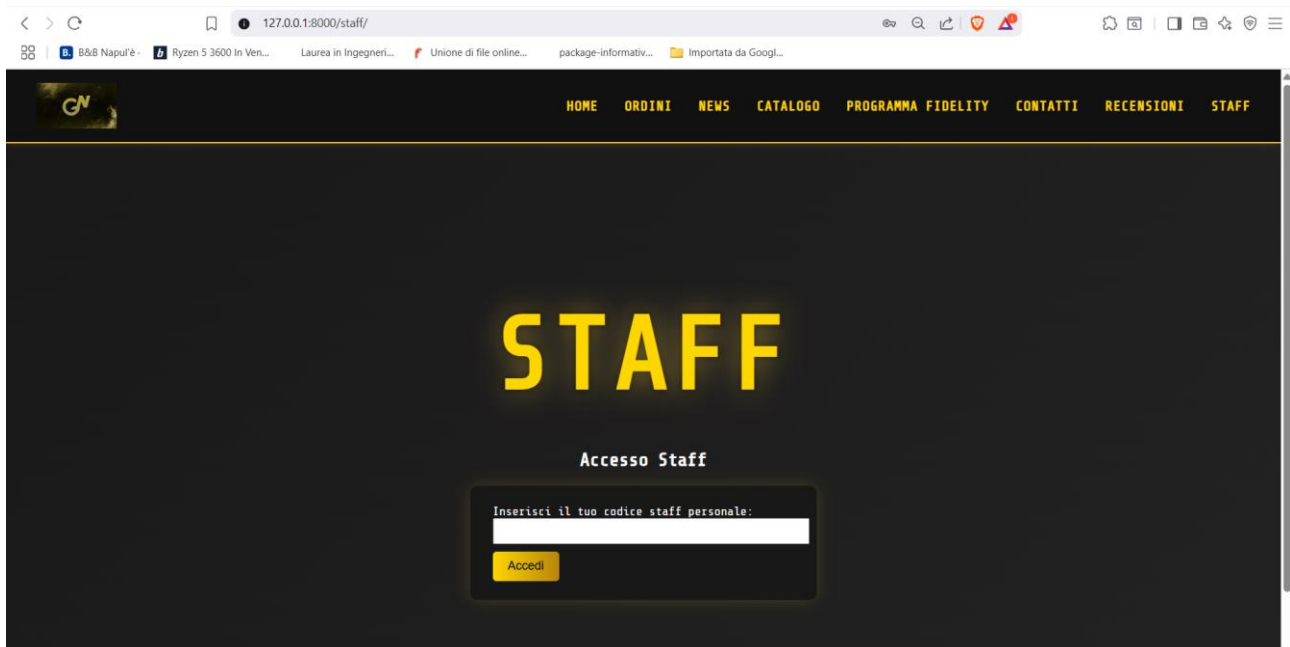
SCHERMATA CONTATTI

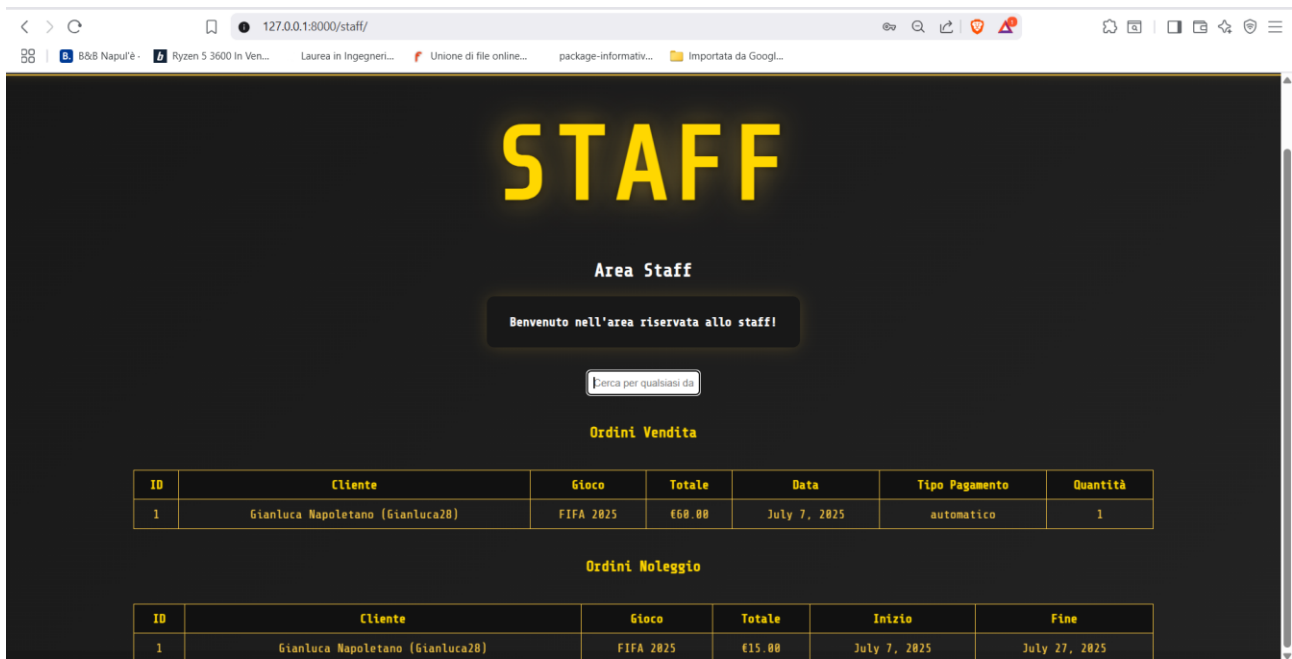


SCHERMATA RECENSIONI



SCHERMATA STAFF (I codici dipendenti vanno inseriti da database per poter accedere alla sezione riservata)





Sicurezza dell'Applicazione

Nello sviluppo dell'applicazione GN SHOP, è stata considerata l'importanza della sicurezza dei dati degli utenti. Sebbene non ancora implementate, sono previste misure di protezione come la gestione sicura delle password tramite algoritmi di hashing (es. MD5 o SHA-256) e l'uso del token CSRF per prevenire attacchi di tipo Cross-Site Request Forgery. Inoltre, si prevede l'introduzione di controlli lato server per garantire che solo utenti autenticati possano accedere alle sezioni riservate dell'applicazione, come la gestione ordini o l'area staff.

Prevenzione SQL Injection

Una delle vulnerabilità più comuni nelle applicazioni web è la SQL Injection, che si verifica quando input utente non filtrati vengono inseriti direttamente nelle query SQL. Per prevenire questo rischio, si prevede l'adozione dell'ORM di Django, che consente di costruire query parametrizzate in modo sicuro, evitando l'inserimento diretto di dati potenzialmente pericolosi.