

Traccia d'Esame – Progetto di sistema informativo per la gestione ordini della Tenuta Beltrani

Il progetto ha come obiettivo la realizzazione di un sistema informativo moderno e completo, destinato a digitalizzare la gestione delle vendite dei prodotti agricoli della Tenuta Beltrani, superando le attuali procedure manuali e obsolete. Il sistema si concretizzerà in un'applicazione web intuitiva, progettata per migliorare l'efficienza operativa e offrire un'esperienza utente avanzata.

Il cuore del sistema sarà costituito da un database relazionale SQL, progettato per organizzare in modo strutturato tutte le informazioni necessarie alla gestione dell'attività. Al suo interno sono presenti, tra le altre, una sezione dedicata al catalogo prodotti: contenente nome, descrizione, formato, prezzo, disponibilità per ciascun centro distributivo e immagine, e un modulo per la gestione degli ordini, con funzionalità di tracciamento dello stato (in attesa, spedito, consegnato, annullato) e consultazione dello storico ordini per ogni utente.

Il sistema prevederà due principali tipologie di clienti: utenti privati, per i quali saranno registrati nome e cognome, e utenti commerciali, per cui verranno archiviati dati come ragione sociale e partita IVA. L'accesso all'applicazione avverrà tramite credenziali personali, con profili configurati in modo differenziato a seconda della categoria di appartenenza.

La gestione degli ordini rappresenta una funzionalità chiave, permettendo di tracciare l'intero ciclo di vita del prodotto, dall'azienda agricola di origine alla destinazione finale, incluse eventuali proroghe o ritardi nella consegna. Tutti i dati saranno consultabili nello storico, utile sia per finalità amministrative che di supporto al cliente.

Gli utenti avranno la possibilità di lasciare valutazioni e commenti relativi ai prodotti acquistati, contribuendo a migliorare la trasparenza e la reputazione aziendale.

La Tenuta Beltrani organizza periodicamente eventi esperienziali, come visite al frantoio durante la molitura o percorsi guidati nelle campagne durante la raccolta. Il sistema permetterà di pubblicare informazioni dettagliate sugli eventi, gestire le iscrizioni online, registrare le presenze e conservare un archivio storico con materiali associati.

Data la presenza di due punti di distribuzione, uno in Puglia e uno in Veneto, il sistema includerà una sezione per la distribuzione geografica degli ordini, in modo da indirizzarli al centro logistico più vicino. Questo migliorerà la tempestività nelle consegne e offrirà un riferimento chiaro agli utenti per l'assistenza.

Obiettivo del Progetto

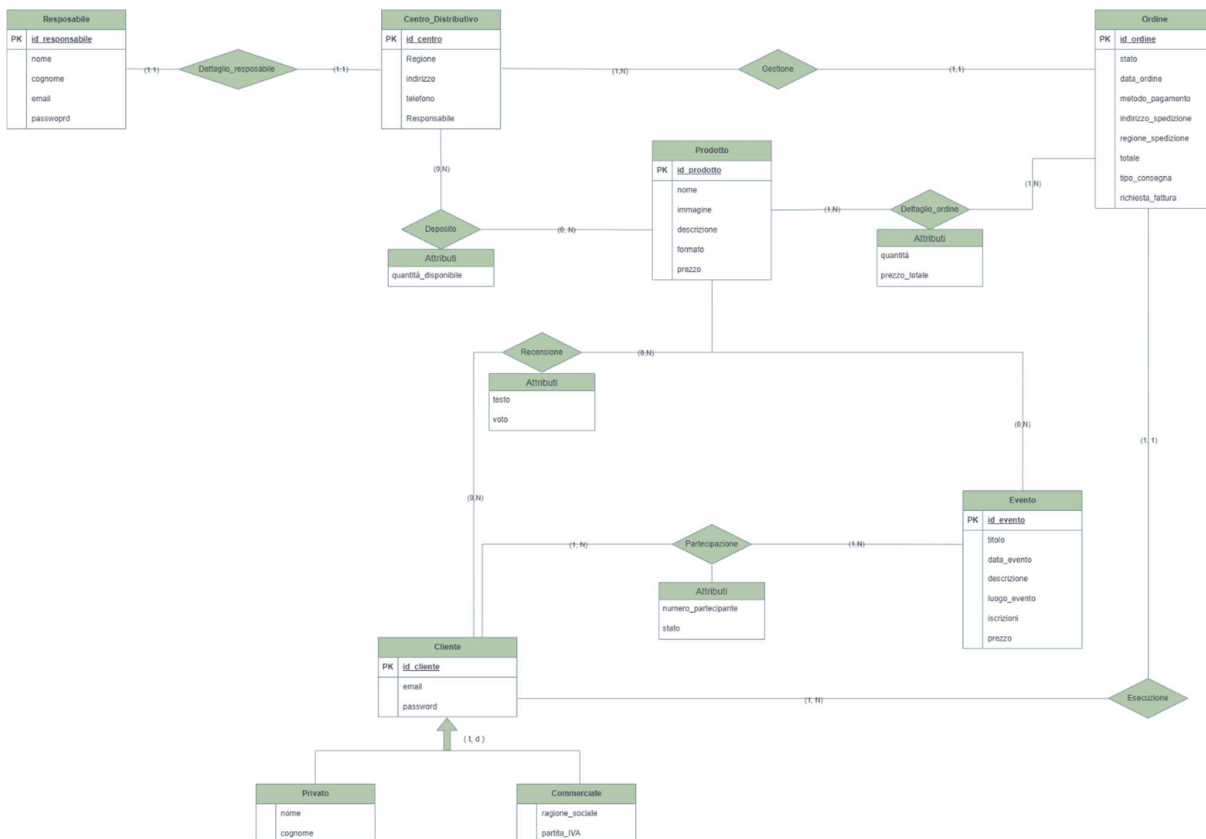
1. Analisi e progettazione concettuale

Modello E/R

Il sito è incentrato principalmente sulla gestione delle interazioni con i clienti. Questi ultimi potranno consultare un catalogo prodotti completo di descrizioni dettagliate, immagini, formati disponibili, prezzi e quantità presenti nel centro distributivo.

Una volta selezionati i prodotti desiderati, sarà possibile procedere all'ordine, che potrà essere monitorato tramite un'apposita area riservata. L'accesso a quest'ultima avverrà mediante credenziali inserite nel momento della registrazione.

I responsabili della tenuta avranno a disposizione un pannello dedicato che consentirà loro di aggiornare lo stato degli ordini e degli eventi, nonché di accedere a tutte le informazioni relative ai clienti registrati.



Inizialmente è stato realizzato un modello E/R con un'entità padre e due entità figlie, collegate da una relazione di specializzazione totale e disgiunta.

Relazione Cliente → Privato & Commerciale

- Vincolo di copertura: Totale → Ogni cliente è sempre un privato o un commerciale.
- Vincolo di disgiunzione: Disgiunta → Un cliente è o privato o commerciale, mai entrambi.

La specializzazione dell'entità Cliente è stata così definita:

- Attributi comuni: id_cliente, email, password
- Privato: nome, cognome
- Commerciale: ragione sociale, Partita IVA

Soluzioni possibili alla generalizzazione

Per la generalizzazione dell'entità Cliente, si possono adottare due strategie principali nel passaggio dal modello concettuale al modello logico/relazionale:

1. Soluzione 1: Accorpamento in padre

In questo approccio, la generalizzazione ingloba anche le specializzazioni, creando un'unica tabella Cliente che contiene tutti gli attributi comuni e specifici delle entità figlie (Privato e Commerciale). Gli attributi specifici vengono resi opzionali (accettano valori NULL se non pertinenti) e si aggiunge un attributo Ruolo per distinguere il tipo di cliente.

Cliente(id_cliente, email, password, ruolo [privato | commerciale], nome, cognome, ragione_sociale, partita_iva)

Vantaggio:

- Una sola tabella da gestire.
- Maggiore semplicità nelle query generiche su tutti i clienti.

Svantaggio:

- Presenza di molti valori NULL.
- Minor integrità semantica rispetto alle entità figlie.

2. Soluzione 2: Accorpamento in figlie

In questo approccio, le entità specializzate (Privato e Commerciale) ereditano gli attributi dell'entità padre Cliente. Si creano quindi due tabelle distinte, ognuna con tutti gli attributi necessari (sia comuni che specifici).

Vantaggi:

- Nessun attributo con valore NULL.
- Maggiore chiarezza semantica: ogni tabella rappresenta un tipo ben definito di cliente.

Svantaggi:

Ridondanza nelle relazioni: entrambe le tabelle (Privato e Commerciale) devono essere collegate separatamente a tabelle esterne tramite relazioni come:

- Partecipazione → verso la tabella Eventi
- Eseguire → verso la tabella Ordine
- Recensione → verso la tabella Prodotto

Questo può aumentare la complessità delle query e della gestione delle chiavi esterne.

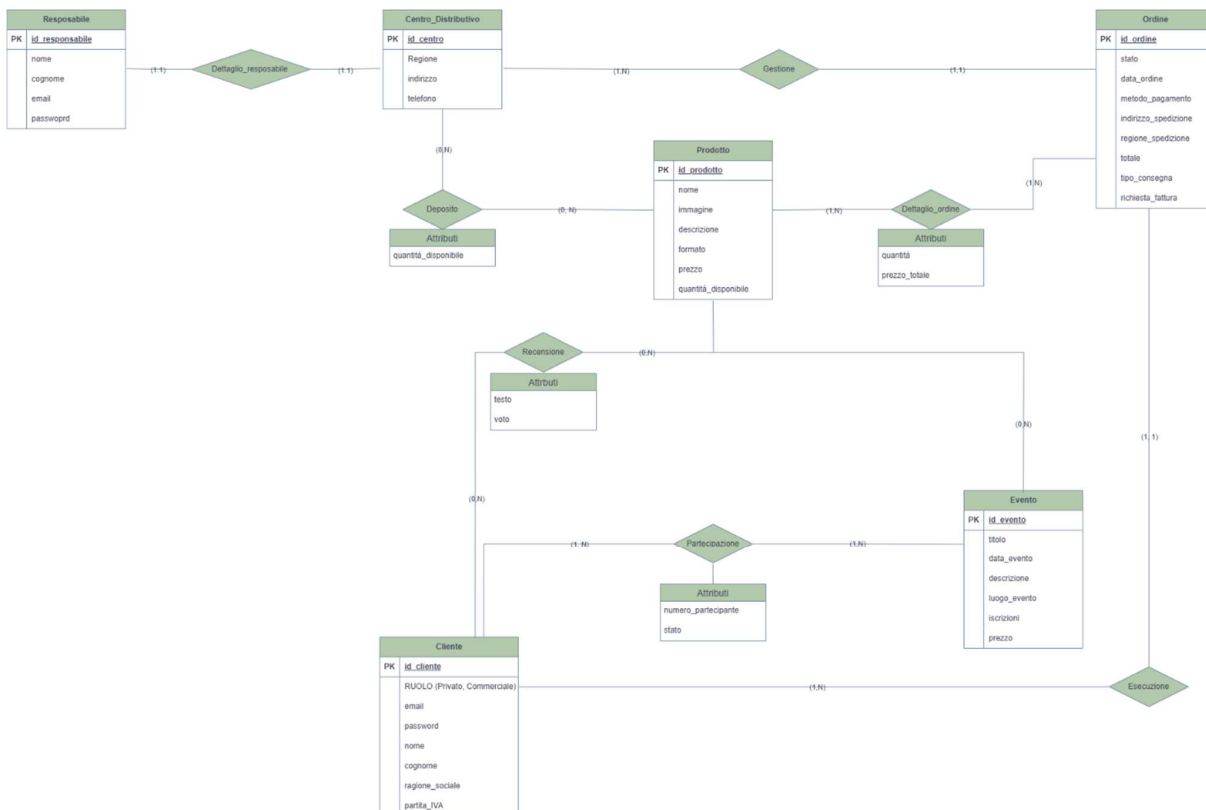
Modello E/R ottimizzato

Si è scelto di risolvere la generalizzazione in modo che assorbisse le specializzazioni, inglobando gli attributi delle entità figlie all'interno dell'entità padre. La decisione è motivata dal fatto che le operazioni previste dal sistema non richiedono la gestione separata delle diverse tipologie di cliente. Sono stati quindi inglobati gli attributi di Commerciale e Privato nell'entità generale Cliente, aggiungendo un attributo RUOLO per distinguerle, in quanto non si riscontrano differenze strutturali rilevanti.

Questo approccio prevede una sola tabella per Cliente, con attributi opzionali specifici per Privato e Commerciale, con l'aggiunta dell'attributo Ruolo per distinguerli. Per i clienti che non riguardano una delle due specializzazioni, gli attributi non pertinenti assumono valore NULL nel database relazionale risultante.

- Cliente: id_cliente, email, password, ruolo (Privati, Commerciale)
 - Privato: nome, cognome
 - Commerciale: ragione sociale, Partita IVA

Nel modello E/R questi attributi saranno visualizzati come parte dell'entità Cliente, ma vincolati dalla specializzazione.



2. Progettazione logica

Centro_Distributivo (id_centro, regione (Veneto, Puglia), indirizzo, telefono, id_responsabile:responsabile)

Responsabile (id_responsabile, nome, cognome, email, password)

Prodotto (id_prodotto, nome, immagine, prezzo, formato, descrizione)

Ordine(id_ordine, stato(attesa, spedito, consegnato, annullato), data_ordine, metodo_pagamento(cartà di credito, paypal, bonifico) , indirizzo_spedizione, regione_spedizione, totale, tipo_consegna(standard, espresso), richiesta_fattura, id_centro:centro_distribuzione, id_cliente: cliente)

Cliente (id_cliente, email, password, RUOLO (privato, commerciale), nome, cognome, ragione_sociale, Partita_IVA)

Evento (id_evento, titolo, data_evento, descrizione, luogo_evento, iscrizione, prezzo)

Dettaglio_ordine (id_dettaglio_ordine, ordine:id_ordine, id_prodotto:prodotto, quantità, prezzo_totale)

Partecipazione(id_partecipazione, numero_partecipanti, stato(confermato, annullato, attesa), id_cliente:cliente, id_evento:evento)

Recensione (id_recensione, testo, valutazione, id_cliente:cliente, id_prodotto:prodotto, id_evento:evento)

Deposito(id_deposito, quantità_disponibile, id_prodotto:prodotto, id_centro: centro_distribuzione)

Entità

CLIENTE				
Nome Campo	Descrizione	Tipo	Lunghezza	Vincoli
Id	Identificativo univoco del Cliente	Integer, AutoIncrement	–	Primary Key, Unique, Not null
Email	Indirizzo email univoco del cliente	Varchar	254	Unique, Not null
Password	Password del cliente	Varchar	128	Not null
Nome	Nome del cliente (solo per i clienti privati)	Varchar	50	Not null, se ruolo = "Privato"
Cognome	Cognome del cliente (solo per i clienti privati)	Varchar	50	Not null, se ruolo = "Privato"
Ragione sociale	Ragione sociale (solo per clienti commerciali)	Varchar	100	Not null, se ruolo = "Commerciale"
Partita IVA	Partita IVA (solo per clienti commerciali)	Varchar	20	Not null, se ruolo = "Commerciale"

Ruolo	Tipologia del cliente (Privato o Commerciale)	Enum	–	Not null, valori ammessi: ['Privato', 'Commerciale']
-------	--	------	---	---

Ruolo: può assumere come valori privato o commerciale

- Se viene scelto Privato si potranno compilare solo i campi nome e cognome, mentre ragione sociale e partita iva rimarranno di tipo null
- Se viene scelto Commerciale si potranno compilare solo i campi ragione sociale e partita iva, mentre nome e cognome rimarranno di tipo null

Vincolo di Tupla

- Un utente non può registrarsi due volte con la stessa email

PRODOTTO				
Nome Campo	Descrizione	Tipo	Lunghezza	Vincoli
Id	Identificativo univoco del prodotto	Integer, AutoIncrement	–	Primary Key, Unique, Not null
Nome	Nome del prodotto	Varchar	100	Not null
Descrizione	Descrizione testuale del prodotto	Longtext	illimitata (TEXT)	Not null
Formato	Formato/confezione del prodotto	Varchar	50	Not null
Prezzo	Prezzo del prodotto	Integer	max_digits =10, dec=2	Not null
Immagine	Immagine del prodotto	Varchar	Percorso dell'immagine	Not null

ORDINE				
Nome Campo	Descrizione	Tipo	Lunghezza	Vincoli
Id	Identificativo univoco dell'ordine	Integer, AutoIncrement	11	Primary Key, Unique, Not null
Data ordine	Data in cui è stato effettuato l'ordine	Date	–	Not null

Stato	Stato dell'ordine	Enum	–	Not null ,valori ammessi: ['Attesa', 'Spedito', 'Consegnato','Annullato']
Metodo pagamento	Metodo scelto per il pagamento	Enum	–	Not null, valori ammessi: ['Carta di credito', 'PayPal', 'Bonifico']
Tipo consegna	Tipo di spedizione scelta	Enum	–	Not null , valori ammessi: ['Standard', 'Espresso']
Indirizzo spedizione	Indirizzo dettagliato di spedizione	Varchar	255	Not null
Totale	Totale economico dell'ordine	Int	max_digits =10, dec=2	Not null
Richiesta fattura	Indica se è richiesta la fattura (1 = Sì, 0 = No)	Boolean	1	Not null
Id centro	Centro logistico da cui parte la spedizione	Foreign Key	–	Not null, riferimento a Centro_Distributivo(id)
Id cliente	Cliente che ha effettuato l'ordine	Foreign Key	–	Not null, riferimento a Cliente(id)

CENTRO DISTRIBUTIVO				
Nome Campo	Descrizione	Tipo	Lunghezza	Vincoli
Id	Identificativo univoco del centro distributivo	Integer, AutoIncrement	–	Primary Key, Unique, Not null
Regione	Regione in cui si trova il centro	Enum	–	Not null, valori ammessi: ['Veneto', 'Puglia']
Indirizzo	Indirizzo completo del centro	Varchar	255	Not null
Telefono	Numero di telefono del centro	Varchar	20	Not null

Id responsabile	Riferimento al responsabile assegnato al centro	Foreign Key	–	Not null, riferimento a responsabile(id), on delete cascade
-----------------	---	-------------	---	---

EVENTO				
Nome Campo	Descrizione	Tipo	Lunghezza	Vincoli
Id	Identificativo univoco dell'evento	Integer, AutoIncrement	–	Primary Key, Unique, Not null
Titolo	Titolo dell'evento	Varchar	100	Not null
Data evento	Data in cui si svolge l'evento	Date	–	Not null
Descrizione	Descrizione testuale dell'evento	Text	illimitata (TEXT)	Not null
Luogo evento	Luogo in cui si tiene l'evento	Varchar	255	Not null
Iscrizione	Indica il numero di iscrizioni possibili	Int	255	Not null
Prezzo	Prezzo per partecipare all'evento	Decimal	max_digits=8, dec=2	Not null

RESPONSABILE				
Nome Campo	Descrizione	Tipo	Lunghezza	Vincoli
Id	Identificativo univoco del responsabile	Integer, AutoIncrement	–	Primary Key, Unique, Not null
Nome	Nome del responsabile	Varchar	100	Not null
Cognome	Cognome del responsabile	Varchar	100	Not null
Email	Indirizzo email (univoco) del responsabile	Email	254	Unique, Not null
Password	Password di accesso	Varchar	128	Not null

Relazioni

DETTAGLIO ORDINE				
Nome Campo	Descrizione	Tipo	Lunghezza	Vincoli
Id	Identificativo univoco della riga di dettaglio ordine	Integer, Autoincrement	20	Primary Key, Unique, Not null
Quantità	Quantità del prodotto ordinato	Integer	10	Not null, CHECK (quantita ≥ 0)
Prezzo totale	Prezzo totale (quantità × prezzo unitario) del prodotto nell'ordine	Integer	max_digits=10, dec=2	Not null
Id ordine	Riferimento all'ordine associato	Foreign Key	–	Not null, riferimento a tenuta_ordine(id), On delete cascade
Id prodotto	Riferimento al prodotto incluso nell'ordine	Foreign Key	–	Not null, riferimento a prodotto(id), On delete cascade

Relazione N:M tra Ordine e Prodotto

- Un ordine può contenere uno o più dettagli d'ordine, ciascuno riferito a un prodotto specifico.
- Un prodotto può comparire in uno o più dettagli d'ordine, associato a diversi ordini.

Il dettaglio ordine non può esistere senza un ordine e un cliente associato.

PARTECIPAZIONE				
Nome Campo	Descrizione	Tipo	Lunghezza	Vincoli
Id	Identificativo univoco della partecipazione	Integer, AutoIncrement	–	Primary Key, Unique, Not null
Stato	Stato della partecipazione	Enum	–	Not null, valori ammessi:

	(confermato, annullato, attesa)			['confermato', 'annullato', 'attesa']
Numero partecipanti	Numero di partecipanti associati alla registrazione	Integer	255	Not null, ≥ 0 (valore positivo incluso 0)
Id evento	Evento a cui si riferisce la partecipazione	Foreign Key	–	Not null, riferimento a evento(id), On delete cascade
Id cliente	Cliente che partecipa all'evento	Foreign Key	–	Not null, riferimento a cliente(id), On delete cascade

Relazione N:M tra Cliente e Evento

- Un cliente può partecipare a uno o più eventi.
- Un evento può avere uno o più clienti iscritti.

La partecipazione non può esistere senza un evento e un cliente associato.

DEPOSITO				
Nome Campo	Descrizione	Tipo	Lunghezza	Vincoli
Id	Identificativo univoco del deposito	Integer, AutoIncrement	–	Primary Key, Unique, Not null
Quantità	Quantità disponibile del prodotto nel deposito	Integer	255	Not null, ≥ 0 (valore positivo incluso 0)
Id centro	Riferimento al centro distributivo dove è stoccato il prodotto	Foreign Key	–	Not null, riferimento a centro distributivo(id), On delete cascade
Id prodotto	Riferimento al prodotto presente nel deposito	Foreign Key	–	Not null, riferimento a prodotto(id), On delete cascade

Relazione N:M tra Centro Distributivo e Prodotto

- Un centro distributivo può depositare uno o più prodotti al proprio interno.
- Un prodotto può essere depositato in uno o più centri distributivi.

Il deposito non può esistere senza un centro e un prodotto associato.

RECENSIONE				
Nome Campo	Descrizione	Tipo	Lunghezza	Vincoli
Id	Identificativo univoco della recensione	Integer, AutoIncrement	–	Primary Key, Unique, Not null
Testo	Testo della recensione	Text	–	Not null
Valutazioni	Valutazione numerica (ad esempio da 1 a 5)	Integer	1	Not null, generalmente range 1–5
Id cliente	Cliente che scrive la recensione	Foreign Key	–	Not null, riferimento a cliente(id), On delete cascade
Id prodotto	Prodotto recensito (opzionale)	Foreign Key	–	Nullable, riferimento a prodotto(id), On delete cascade
Id evento	Evento recensito (opzionale)	Foreign Key	–	Nullable, riferimento a evento(id), On delete cascade

Relazione N:M tra Cliente e Prodotto oppure Evento

- Un cliente può fare da 0 a N recensioni (cioè può non averne fatte oppure averne fatte tante).
- Un prodotto o un evento può ricevere da 0 a N recensioni (cioè può non avere recensioni oppure averne tante).

Le recensioni non possono esistere senza un cliente e un Prodotto o un Evento associato.

Vincoli interrelazionali

- Un cliente per poter effettuare ordini o partecipare ad eventi si deve registrare, in modo che i suoi dati vengano salvati all'interno della tabella cliente.
- Un cliente può recensire un prodotto, solo se lo ha acquistato (cioè ha un ordine associato che contiene quel prodotto) oppure un evento, solo se vi ha partecipato (cioè è presente in Partecipazione).
- Un cliente per poter visualizzare i dettagli del proprio ordine deve avere eseguito un ordine, ovvero deve avere un ordine associato al quel dettaglio ordine.

3. Implementazione del sistema informativo

L'applicazione, sviluppata con Django, ha l'obiettivo di gestire in modo efficace i dati e le funzionalità necessarie all'attività della Tenuta Beltrani, coprendo sia gli aspetti operativi che quelli legati all'esperienza utente. Il sistema integra diverse funzionalità fondamentali, scelte tra quelle previste nella descrizione iniziale e adattate alle esigenze aziendali:

Funzionalità attualmente implementate

- Registrazione e autenticazione dei clienti (differenziati per ruolo):
Sistema di login sicuro con distinzione tra clienti privati e attività commerciali
- Registrazione e gestione degli ordini (spedizione e consegna):
Gli utenti possono creare ordini, specificando metodo di pagamento, tipo di consegna e tracciarne lo stato (in attesa, spedito, consegnato, annullato).
- Amministrazione degli ordini in base alla regione:
Gli ordini vengono assegnati e visualizzati secondo la regione di spedizione, consentendo un'amministrazione logistica efficiente. All'interno del sito, ciascuno dei due centri distributivi dispone di una propria area dedicata per la gestione degli ordini.
- Inserimento e consultazione di recensioni:
Gli utenti possono lasciare feedback sui prodotti acquistati e sugli eventi a cui hanno preso parte, migliorando l'esperienza generale e la trasparenza.
- Storico degli ordini per cliente
Ogni utente ha accesso a un archivio completo dei propri ordini, consultabile in qualsiasi momento.

Funzionalità previste per sviluppi futuri

- Visualizzazione del catalogo prodotti:
Sarà introdotta una sezione per consultare il catalogo completo, con schede dettagliate per ogni prodotto (descrizione, immagini, caratteristiche tecniche).
- Iscrizione agli eventi esperienziali:
Verrà implementata una funzionalità per visualizzare e iscriversi agli eventi organizzati dall'azienda (visite guidate, degustazioni, ecc.).
- Ricerca avanzata nel catalogo:
È prevista una funzionalità di filtraggio avanzato per cercare i prodotti in base a tipologia, formato e altri parametri specifici.
- Gestione dei depositi per centro distributivo:
Il sistema sarà ampliato con un modulo per la gestione dell'inventario all'interno di ciascun centro, utile al monitoraggio delle disponibilità e al rifornimento.

Descrizione della web app

È stata implementata una web app che permette agli utenti di autenticarsi per ordinare i prodotti gestiti dai lavoratori dell'azienda.

Login Personale

The screenshot shows a web browser window with the URL 127.0.0.1:8000/personale. The page header includes the company logo 'Azienda Agricola Tenuta Beltrani' and navigation links for 'Area Amministrativa' and 'Area Personale'. The main content area is titled 'Accedi all'area personale' and contains a login form with the following fields:

- Email:
- Password:
- Accedi button
- Link: Non ancora registrato? [Registrati](#)

Nell'area personale, i clienti possono visualizzare i propri dati di accesso, effettuare nuovi ordini, consultare lo storico degli ordini, per tenere sotto controllo il loro stato e inserire una recensione per ogni prodotto, fornendo così un feedback diretto ai venditori.

Home Personale

The screenshot shows a web browser window with the URL 127.0.0.1:8000/login. The page header includes the company logo 'Azienda Agricola Tenuta Beltrani' and navigation links for 'Area Amministrativa' and 'Area Personale'. The main content area is titled 'Benvenuto tizio ieri' and contains a 'Dati anagrafici' section with the following fields:

- Nome:
- Cognome:
- Email:
- Address:

On the left side, there is a 'MENU' section with the following links:

- Home
- Esegui ordini
- Storico degli ordini
- Inserisci recensioni
- Logout

Ogni cliente può scegliere con cura i prodotti desiderati, indicandone la quantità, l'indirizzo di spedizione e la regione. Queste informazioni consentono al sistema di smistare gli ordini nei due centri distributivi. Inoltre, è possibile selezionare il metodo di pagamento e richiedere la fattura, se necessario. In caso di ordine effettuato con successo, viene visualizzata una pagina di conferma.

Esegui Ordini

Ordini - Tenuta Beltrani

127.0.0.1:8000/ordine

Italiano

Area Amministrativa

Azienda Agricola
Tenuta Beltrani

Area Personale

HOME PRODOTTI EVENTI RECENSIONI

Benvenuto nell'Area Personale

MENU

- Home
- Esegui ordini
- Storico degli ordini
- Inserisci recensioni
- Logout

Scegli i prodotti

Lattina d'olio da 5 litri

Prezzo: €45.00

Formato: 5L

Quantità:
Nessuna

Inserimento dettagli ordine

Pagamento - Tenuta Beltrani

127.0.0.1:8000/pagamento

Italiano

Area Amministrativa

Azienda Agricola
Tenuta Beltrani

Area Personale

HOME PRODOTTI EVENTI RECENSIONI

Benvenuto nell'Area Personale

MENU

- Home
- Esegui ordini
- Storico degli ordini
- Inserisci recensioni
- Logout

Dati spedizione e pagamento

Indirizzo di spedizione:

Regione di spedizione:
... Seleziona una regione ...

Tipo di consegna:
Standard

Metodo di pagamento:
Carta di credito

☐ Desidero ricevere la fattura

Conferma Ordine

Storico degli ordini

Benvenuto nell'Area Personale

MENU

- Home
- Esigui ordini
- Storico degli ordini
- Inserisci recensioni
- Login

I tuoi Ordini

Data	Totale	Stato	Metodo Pagamento	Fattura	Prodotti
June 10, 2025	56.00 €	Spedito	PayPal	No	<ul style="list-style-type: none"> 1 Litro d'olio da 5 litri x 1 1 Latticchia ricotta 5.500 gr x 1 1 Vasetto di funghi 300 gr x 1
June 5, 2025	234.90 €	Spedito	Carta di credito	No	<ul style="list-style-type: none"> 1 Litro d'olio da 5 litri x 3 1 Carbone da 500 gr x 2 1 Latticchia ricotta 5.500 gr x 2 1 Vasetto di funghi 300 gr x 1
June 2, 2025	135.90 €	Spedito	PayPal	No	<ul style="list-style-type: none"> 1 Litro d'olio da 5 litri x 1 1 Litro d'olio da 3 litri x 1 1 Carbone da 500 gr x 1

Realizzato da Lara Maggiulli - Progetto Universitario - Università degli Studi di Napoli Parthenope © 2025

Inserimento recensioni

Benvenuto nell'Area Personale

MENU

- Home
- Esigui ordini
- Storico degli ordini
- Inserisci recensioni
- Login

Inserisci Recensione

Valutazione (1-5):

Testo:

Tipo di recensione:

Dal lato amministrativo, i responsabili dei due centri distributivi possono accedere all'area dedicata con le proprie credenziali per gestire gli ordini, visualizzarli e aggiornare il loro stato in tempo reale.

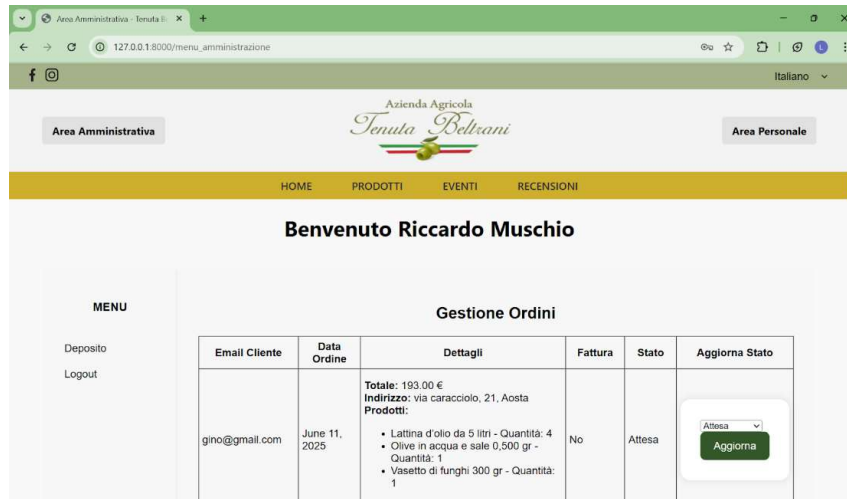
Login Responsabili

Accedi all'area amministrativa

Email:

Password:

Menu Responsabili



Sistema di gestione delle sessioni

Il sistema di gestione delle sessioni è stato implementato utilizzando i cookie di sessione, o cookie temporanei. Si tratta di file di testo che il browser salva sul computer dell'utente per la durata della sessione di navigazione e che vengono eliminati automaticamente alla chiusura del browser. Non avendo una data di scadenza specifica, questi cookie vengono utilizzati per identificare l'utente durante la sessione, ad esempio memorizzando la sua email. Questa informazione viene aggiornata nel momento in cui effettua l'accesso un altro cliente o un responsabile.

L'email memorizzata nella sessione viene poi utilizzata per interrogare il database, identificando l'utente o il responsabile all'interno delle rispettive tabelle personalizzate, e permettendo il recupero dei dati associati (come ordini, richieste, gestione prodotti, ecc.).

L'uso dei cookie di sessione è stato necessario in quanto, per il progetto, si è scelto di non utilizzare il sistema di autenticazione predefinito di Django (Abstractuser), ma di creare tabelle personalizzate per gestire separatamente clienti e responsabili. Di conseguenza, non è stato adottato il meccanismo integrato di login/logout di Django, bensì un sistema di autenticazione su misura, che prevede il salvataggio dell'email nella sessione per identificare l'utente attualmente loggato. In particolare, per gestire l'accesso all'applicazione web da uno stesso dispositivo, si è resa necessaria la separazione delle due sessioni. Ciò è stato possibile grazie all'utilizzo della funzione `request.session.pop('email', None)` nel logout del cliente e `request.session.pop('admin_email', None)` nel logout del responsabile, che rimuovono rispettivamente l'email dell'utente loggato senza interferire con la sessione dell'altro.

Esempio pratico:

Inizialmente si inseriscono i dati necessari per il login

The screenshot shows a login form titled 'Accedi all'area personale'. It has two input fields: 'Email:' with the value 'gigio@gmail.com' and 'Password:' with a masked password '****'. Below the fields is a green 'Accedi' button. At the bottom, there is a link that says 'Non ancora registrato? [Registrati](#)'.

Grazie all'utilizzo dell'applicazione Wireshark è possibile analizzare il flusso del traffico HTTP. In particolare, si può osservare la presenza del parametro sessionid, che identifica la sessione dell'utente autenticato.



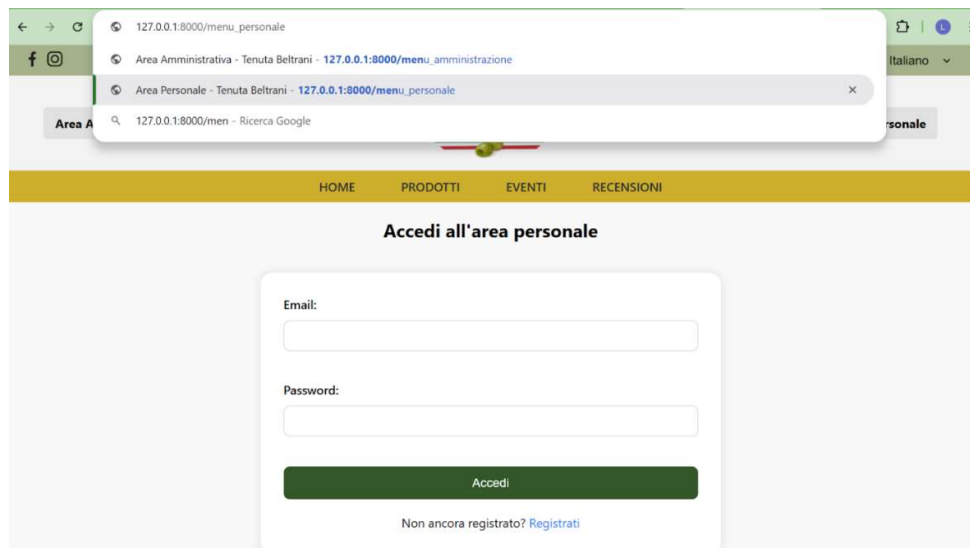
```
Wireshark - Segui flusso HTTP (tcp.stream eq 4) - Adapter for loopback traffic capture

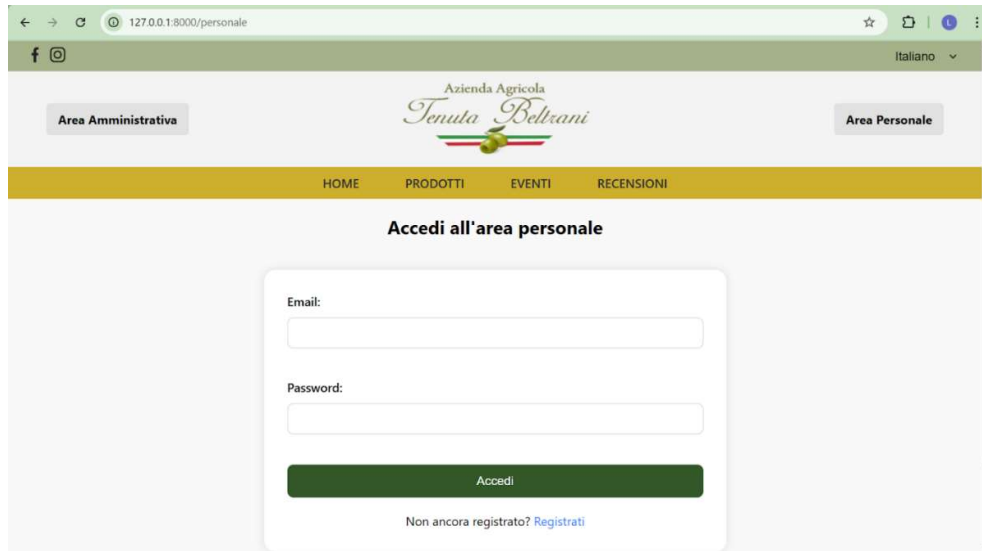
POST /login HTTP/1.1
Host: 127.0.0.1:8000
Connection: keep-alive
Content-Length: 122
Cache-Control: max-age=0
sec-ch-ua: "Google Chrome";v="137", "Chromium";v="137", "Not(A)Brand";v="24"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "Windows"
Origin: http://127.0.0.1:8000
Content-Type: application/x-www-form-urlencoded
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/137.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: http://127.0.0.1:8000/personale
Accept-Encoding: gzip, deflate, br, zstd
Accept-Language: it-IT,it;q=0.9,en-US;q=0.8,en;q=0.7
Cookie: csrfmiddlewaretoken=uh9TK0vNDmraqRZEuW8zgfQSEJmVJRPT2vgnu8Ivw7Tz079xEthSH1RAEWZLOKsQ&email=gigio%4@gmail.com&password=1234
sessionid=0isfo8y7h1bgtjt71nwgpkd6oq7dnrt
```

Inoltre, dopo il logout, se si tenta di accedere a una pagina dell'area amministrativa o personale senza aver effettuato il login, si viene automaticamente reindirizzati alla pagina di login corrispondente. Questo comportamento è dovuto al fatto che, nel momento in cui viene richiesta una di queste pagine tramite URL, viene effettuato un controllo sul cookie di sessione, in particolare sulla presenza di un flag che verifica se l'utente è autenticato, come ad esempio `is_authenticated`

```
if not request.session.get('is_authenticated'):
    return redirect('personale')
```

Esempio pratico





Implementazione SQL injection

Nel contesto dello sviluppo web, una delle vulnerabilità più comuni e pericolose è la **SQL Injection**. Questo tipo di attacco si verifica quando un'applicazione costruisce query SQL inserendo direttamente i dati provenienti dall'utente, senza effettuare un adeguato filtraggio o uso di meccanismi di protezione.

Codice correlato:

```
def login_vulnerabile(request): 1 usage  Lara Maggiulli
    if request.method == "POST":
        email = request.POST.get("email")
        password = request.POST.get("password")

        # Query SQL vulnerabile
        query = f"SELECT * FROM tenuta_cliente WHERE email = '{email}' AND password = '{password}'"

        with connection.cursor() as cursor:
            cursor.execute(query)
            row = cursor.fetchone()

        if row:
            return render(request, template_name="menu_personale.html", context={"cliente": row})
        else:
            return render(request, template_name="personale.html", context={"error_message": "Credenziali non valide"})
    else:
        return render(request, template_name="personale.html")
```

Questa funzione gestisce il login di un utente. Riceve email e password tramite POST e costruisce una query SQL concatenando direttamente i dati utente nella stringa SQL.

query = f"SELECT * FROM tenuta_cliente WHERE email = '{email}' AND password = '{password}'"

Qui, i dati immessi nel form (email e password) vengono inseriti direttamente nella query SQL, rendendo il sistema vulnerabile a SQL Injection, ovvero l'inserimento di comandi SQL da parte dell'utente per manipolare la query a proprio favore.

In particolare, l'interazione diretta con il database avviene attraverso l'oggetto `cursor`, che rappresenta il canale di comunicazione tra Django e il database relazionale. Il `cursor` viene utilizzato per eseguire comandi SQL e per recuperare i risultati delle query. Nello specifico, l'oggetto viene aperto utilizzando il costrutto `with connection.cursor() as cursor:`, che ha il vantaggio di gestire automaticamente l'apertura e la chiusura del canale di comunicazione con il database, evitando il rischio di lasciare connessioni aperte che potrebbero compromettere le prestazioni dell'applicazione.

Una volta eseguita la query SQL con `cursor.execute(query)`, viene utilizzato il metodo `cursor.fetchone()` per ottenere il risultato. Questo metodo restituisce una singola riga del risultato, che viene salvata nella variabile `row`. Se la query restituisce un risultato (cioè se l'utente esiste con le credenziali fornite), `row` conterrà una tupla con i dati dell'utente. In caso contrario, se non viene trovata alcuna corrispondenza, `row` sarà uguale a `None`.

SQLmap

SQLmap è uno strumento open-source di penetration testing specializzato nell'identificazione e nello sfruttamento automatico delle vulnerabilità SQL injection all'interno di applicazioni web. È ampiamente utilizzato da analisti di sicurezza e sviluppatori per testare la sicurezza dei propri sistemi in modo efficace e automatizzato.

Esempio pratico:

La vulnerabilità dell'applicazione web è testabile eseguendo il seguente comando SQLmap nel terminale:

```
python sqlmap.py -u "http://127.0.0.1:8000/personale_vulnerabile" --data "email=test&password=test" --batch -dbs
```

Spiegazione dei parametri:

- `-u`: specifica l'URL del target vulnerabile;
- `--data`: invia i parametri POST email e password, simulando un form di login;
- `--batch`: esegue in modalità automatica, accettando le impostazioni predefinite;
- `--dbs`: richiede a SQLmap di elencare i database disponibili.

Risultato dell'esecuzione del comando

Questi sono i tipi di SQL Injection che si possono provare:

- Union-Based SQLi

Esempio di payload: `' UNION SELECT NULL,NULL-`

Nel log si vede che è stato testato l'attacco UNION-based, ma senza successo:

- [11:43:40] [INFO] testing 'Generic UNION query (NULL) - 1 to 10 columns'
- [11:43:40] [WARNING] POST parameter 'email' does not seem to be injectable

- Boolean-Based Blind SQLi

Esempio di payload: ' OR 1=1-- o ' AND 1=0--

Questa tecnica verifica se la risposta cambia in base a condizioni vere o false. Nel log è evidente che sono stati testati payload boolean-based per i parametri email e password:

- [11:43:39] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
- [11:43:39] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
- [11:43:40] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
- [11:43:40] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'

- Time-Based Blind SQLi

Il tool tenta attacchi basati sul ritardo della risposta, usando funzioni come SLEEP() o IF() per capire se il parametro è vulnerabile anche senza errori evidenti.

Esempio payload: ' OR IF(1=1,SLEEP(5),0)--

Nel log vediamo che sono stati effettuati test time-based per vari database:

- [11:43:39] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)'
- [11:43:39] [INFO] testing 'PostgreSQL > 8.1 AND time-based blind'
- [11:43:39] [INFO] testing 'Microsoft SQL Server/Sybase time-based blind (IF)'
- [11:43:39] [INFO] testing 'Oracle AND time-based blind'

- Error-Based SQLi

Si sfruttano errori del database per estrarre informazioni.

Esempio di payload: ' AND EXTRACTVALUE(1, CONCAT(0x3a, (SELECT database())))--

Nel log sono presenti test di error-based injection per diversi DBMS:

- [11:43:39] [INFO] testing 'MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
- [11:43:39] [INFO] testing 'PostgreSQL AND error-based - WHERE or HAVING clause'
- [11:43:39] [INFO] testing 'Microsoft SQL Server/Sybase AND error-based - WHERE or HAVING clause (IN)'
- [11:43:39] [INFO] testing 'Oracle AND error-based - WHERE or HAVING clause (XMLType)'

- Stacked Queries

Permettono di eseguire più query in una sola chiamata, separandole con ;.

Esempio di payload: '; DROP TABLE users--

Nel log sono testate diverse varianti per DBMS comuni:

- [11:43:39] [INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'
- [11:43:40] [INFO] testing 'Microsoft SQL Server/Sybase stacked queries (comment)'
- [11:43:40] [INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)'

- Bypass WAF con Tamper Scripts

Il tool segnala che nessuno dei parametri testati sembra vulnerabile con i test standard e suggerisce di aumentare il livello e rischio di test o usare script di tampering per aggirare i filtri:

- [11:43:41] [CRITICAL] all tested parameters do not appear to be injectable. Try to increase values for '--level'/'--risk' options if you wish to perform more tests.

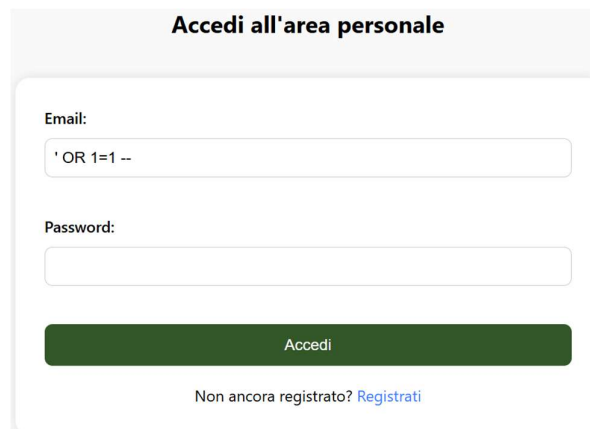
- [11:43:41] [CRITICAL] If you suspect that there is some kind of protection mechanism involved (e.g. WAF) maybe you could try to use option '--tamper' (e.g. '--tamper=space2comment') and/or switch '--random-agent'

Esempio di attacco SQL Injection

Supponiamo che un utente malevolo inserisca come **email**:

' OR 1=1 --

Questa è una tautologia, perché l'espressione $1=1$ è sempre vera, e fa sì che la condizione nel WHERE diventi sempre vera. Di conseguenza, la query restituirà risultati indipendentemente dal valore originale del parametro. In questo modo, si può capire se la query è vulnerabile a SQL Injection osservando la risposta del server. Inoltre, anche il parametro password può risultare vulnerabile, poiché l'operatore - commenta tutto ciò che segue nella query, annullando quindi il controllo sulla password.



Accedi all'area personale

Email:

' OR 1=1 --

Password:

Accedi

Non ancora registrato? [Registrati](#)

È stato possibile eseguire un attacco di SQL Injection poiché nel form della pagine HTML è stato utilizzato l'attributo novalidate, che disattiva la validazione lato client, e il campo email è stato definito con type="text", invece di type="email". Questo consente all'utente di inserire input arbitrari, incluso codice SQL malevolo, senza alcun controllo preliminare.

```
<form method="post" action="/personale_vulnerabile" novalidate>
  {% csrf_token %}
  <div class="mb-3">
    <label for="email" class="form-label">Email:</label>
    <input type="text" class="form-control" id="email" name="email" required>
  </div>

  <div class="mb-3">
    <label for="password" class="form-label">Password:</label>
    <input type="password" class="form-control" id="password" name="password" required>
  </div>
  {% if error_message %}
    <div class="alert alert-danger" role="alert">
      {{ error_message }}
    </div>
  {% endif %}

  <div class="d-grid mb-3">
    <button type="submit" class="btn btn-primary">Accedi</button>
  </div>

  <div class="mb-0" align="center">Non ancora registrato?
    <a href="/registration_page" style="color: #007bff;">Registrati</a>
  </div>
</form>
```

Nel terminale di Django è stata eseguita la seguente query:

Executing query: SELECT * FROM tenuta_cliente WHERE email = " OR 1=1 --' AND password = "

Poiché OR 1=1 è sempre vero e tutto ciò che segue dopo -- viene ignorato come commento, la condizione sulla password viene bypassata. Di conseguenza, anche senza inserire una password valida, la pagina viene mostrata con successo e riesco ad accedere comunque.



Soluzione 1: ORM di Django

Il modo più sicuro e consigliato in Django per gestire il login e prevenire SQL Injection è utilizzare l'ORM, che costruisce query parametrizzate automaticamente, evitando così l'inserimento diretto di dati non filtrati nelle query SQL.

cliente = Cliente.objects.get(email=email)

In questa funzione:

- I dati dell'utente (email) vengono passati all'ORM di Django tramite `Cliente.objects.get(email=email)`, che utilizza query parametrizzate, impedendo SQL Injection.
- La password viene preventivamente hashata con MD5
- La sessione salva lo stato dell'utente autenticato, evitando di dover effettuare nuovamente il controllo.

Per concludere, utilizzare l'ORM di Django è il modo più efficace per evitare vulnerabilità da SQL Injection, perché non si costruiscono query SQL manualmente concatenando input utente.

Sicurezza dell'Applicazione

Per garantire la sicurezza e la privacy degli utenti, sono state implementate diverse misure di protezione all'interno dell'applicazione:

- Gestione sicura delle password:

Le password degli utenti non vengono mai salvate in chiaro nel database. Vengono invece sottoposte a una funzione di hashing, in particolare utilizzando l'algoritmo MD5. Questo garantisce che, anche in caso di accesso non autorizzato al database, le credenziali non siano direttamente leggibili.

hashed_password = hashlib.md5(password.encode()).hexdigest()

- Protezione contro attacchi CSRF (Cross-Site Request Forgery):

Per prevenire attacchi di tipo CSRF, ovvero tentativi di inviare richieste fraudolente da parte di terzi a nome dell'utente autenticato, viene utilizzato il token CSRF su tutte le pagine che includono form HTML. Questo è possibile inserendo il tag {% csrf_token %} all'interno dei form Django, che genera automaticamente un token univoco per ciascuna sessione. In questo modo, eventuali tentativi di spoofing vengono bloccati.

Sommario

Traccia d'Esame – Progetto di sistema informativo per la gestione ordini della Tenuta Beltrani

Obiettivi del Progetto

1. Analisi e progettazione concettuale: Modello E/R.	Pagina 2
2. Progettazione Logica.	Pagina 4
3. Implementazione Sistema Informativo.	Pagina 11