

**Università degli studi UNIMORE**  
**Facoltà di Scienze Informatiche**

**Progetto di Tecnologie Web**

**SmartStore di Gabriele Sitta (matr. 165340)**

**Anno Accademico 2024-25**

## **Indice:**

<b>1. Traccia dettagliata che illustra le funzionalità richieste</b>	<b>pag. 3</b>
<b>2. Diagrammi</b>	<b>pag. 5</b>
2.1 Diagrammi dei casi d'uso	pag. 5
2.2 Diagramma di utilizzo	pag. 5
2.3 Diagramma delle classi gymapp	pag. 6
2.4 Diagramma delle classi – useradmin	pag. 8
2.5 Diagramma delle classi – userauths	pag. 9
<b>3. Tecnologie utilizzate</b>	<b>pag. 9</b>
3.1 Django	pag. 9
3.2 Javascript	pag. 9
3.3 Bootstrap	pag. 10
3.4 Ajax	pag. 10
<b>4. Descrizione del progetto</b>	<b>pag. 10</b>
4.1 Utenti presenti	pag. 10
4.2 Gestione ordini	pag. 11
4.3 Vendere un prodotto	pag. 12
4.4 Ruolo dell'Utente Admin	pag. 12
4.5 Homepage	pag. 12
4.6 Metodo di recensire Prodotto e Venditore	pag. 12
4.7 Pagina contattaci	pag. 12
<b>5. Recommendation System</b>	<b>pag. 13</b>
<b>6. Test</b>	<b>pag. 14</b>
6.1 Test Login	pag. 14
6.2 Test inserimento articolo	pag. 15
6.3 Test reindirizzamento e login	pag. 16
6.4 Test rimuovere elemento dal carrello	pag. 16
<b>7. Screenshot</b>	<b>pag. 17</b>
7.1 Homepage	pag. 17
7.2 Profilo Utente	pag. 18
7.3 Dashboard vendite	pag. 18
7.4 Profilo di Utente Fornitore	pag. 19
7.5 Visualizzazione del carrello	pag. 19
7.6 Pagina di checkout	pag. 20
7.7 Fattura	pag. 20
7.8 Risultati di una ricerca	pag. 21
7.9 Dettagli Prodotto	pag. 21
7.10 Dettagli Fornitore	pag. 22
7.11 Pagina in base alla categoria selezionata	pag. 22
7.12 Recensione Prodotto	pag. 23
7.13 Recommendation System	pag. 23
<b>8. Conclusioni</b>	<b>pag. 24</b>

## 1- Traccia dettagliata che illustra le funzionalità richieste

Il progetto prevede la realizzazione di un portale e-commerce interamente dedicato alla vendita di prodotti per la palestra e di accessori correlati al fitness. L'obiettivo è offrire un'esperienza di navigazione ricca e intuitiva, capace di guidare l'utente attraverso un catalogo di articoli pensati, sia per professionisti del settore, sia per appassionati di attività fisica. Grazie a una serie di strumenti e a funzionalità integrate, il sito si propone di facilitare la ricerca dei prodotti e la gestione degli acquisti, rendendo il processo d'acquisto rapido e sicuro.

### Tipologie di utenti:

All'interno della piattaforma è prevista la presenza di quattro principali categorie di utenti:

- anonimi
- registrati
- fornitori
- admin

A ognuna di queste corrispondono specifiche possibilità di interazione con il sito.

### Utenti anonimi

- **Navigazione libera:** gli utenti che non dispongono di un account possono esplorare liberamente l'intero catalogo prodotti, consultando immagini e descrizioni delle varie proposte.
- **Ricerca e consultazione dei prodotti:** tramite un motore di ricerca interno, gli utenti anonimi possono individuare rapidamente gli articoli di interesse, cercando per categorie, fasce di prezzo o altre caratteristiche.
- **Lettura delle schede informative:** oltre a visualizzare il prodotto, gli utenti hanno la possibilità di approfondire i dettagli tecnici e guardare i dettagli del fornitore.

### Utenti registrati

- **Area personale:** l'utente che decide di registrarsi, creando un proprio profilo, può accedere ad una sezione privata nella quale visualizza e modifica i propri dati personali, come nome utente, numero di telefono e immagine di profilo.
- **Acquisto di prodotti e accessori:** oltre alla consultazione, questi utenti possono acquistare direttamente gli articoli desiderati, inserendoli nel carrello, gestendo in modo autonomo gli ordini e il pagamento.
- **Recensioni dei prodotti e dei fornitori:** l'utente ha la facoltà di lasciare una valutazione e un commento sul prodotto e sul fornitore. Ciò contribuisce a creare una community di clienti informati e a offrire feedback utili ad altri acquirenti.

### Utenti fornitori

- **Inserimento di prodotti in vendita:** i fornitori, una volta riconosciuti come tali dall'Admin, possono caricare nuovi articoli corredati di immagini, prezzi e descrizioni dettagliate, arricchendo così la varietà del catalogo.
- **Visualizzare la propria area riservata:** i fornitori troveranno una propria area riservata con i propri dati personali, articoli in vendita e recensione fornita dagli utenti.

- **Accesso al “Profilo del Fornitore”:** i fornitori possono modificare il logo personalizzato, nome dell’azienda, descrizione e indirizzo fisico.

## **L’utente Admin**

La figura dell’amministratore ricopre un ruolo cruciale nel garantire il corretto funzionamento del portale.

Nello specifico, l’Admin può:

- **Autorizzare o negare il ruolo di fornitore:** Valuta le richieste inviate dagli utenti che desiderano vendere i propri articoli, accettandole o rifiutandole.
- **Gestire i prodotti in vendita:** Ha la facoltà di modificare ed eliminare i prodotti inseriti dai fornitori, qualora non rispettino i termini o la qualità richiesta dalla piattaforma.
- **Gestione area riservata:** All’interno di una sezione dedicata, l’Admin tiene traccia dei prodotti in vendita da parte dei fornitori, monitora gli acquisti effettuati, potendo così controllare in tempo reale l’andamento della propria attività.

## **Caratteristiche dei prodotti**

Ogni articolo è dotato di:

- **Numero identificativo**, garantisce l’unicità nel catalogo.
- **Immagine**, fornisce all’utente una rappresentazione visiva del prodotto.
- **Prezzo**, indicato.
- **Descrizione**, illustra le caratteristiche tecniche e i possibili benefici per chi lo utilizza.

## **Altre funzionalità del sito**

Per rendere l’esperienza d’acquisto ancora più completa, il portale comprende una serie di servizi aggiuntivi:

### **1. Motore di ricerca avanzato**

Il sito mette a disposizione un’efficace barra di ricerca, che consente di trovare rapidamente i prodotti desiderati.

### **2. Recommendation System**

Basandosi sugli acquisti precedenti di ciascun utente e sulle preferenze espresse, il sistema di raccomandazione suggerisce automaticamente prodotti potenzialmente interessanti o complementari a quelli già acquistati. In questo modo, ogni utente riceve consigli personalizzati, migliorando l’esperienza di shopping online. Inoltre sotto ad ogni prodotto sono presenti gli articoli consigliati della stessa categoria.

### **3. Newsletter automatica**

Tutti gli utenti registrati ricevono una newsletter con gli aggiornamenti sugli ultimi articoli inseriti in piattaforma. Questa funzione contribuisce a mantenere alto il coinvolgimento e a favorire acquisti ricorrenti.

## **Funzionalità facoltative scelte per arricchire il progetto**

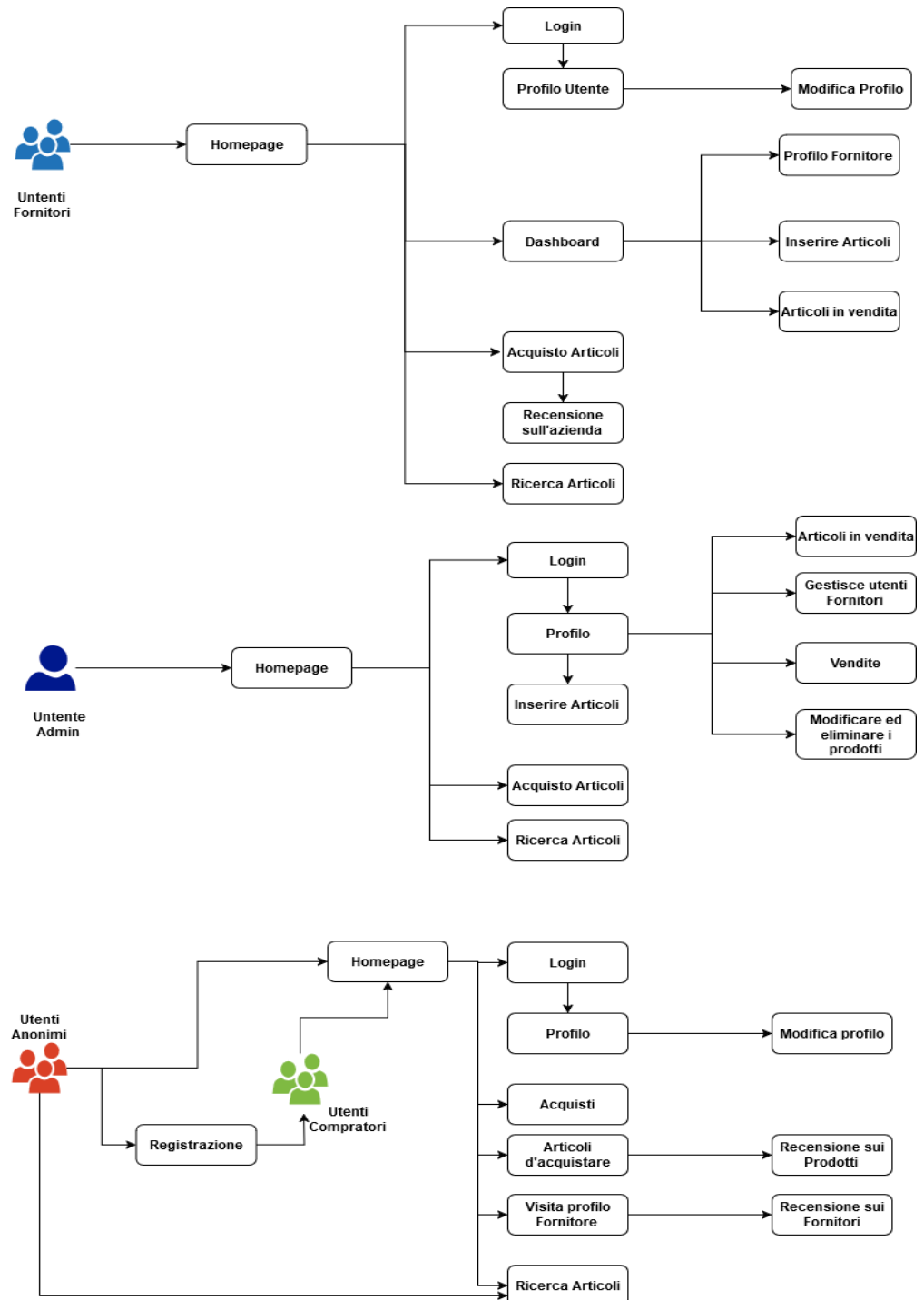
- **Carrello virtuale:** Durante la navigazione, l’utente può inserire nel carrello uno o più prodotti, valutare il prezzo complessivo, modificare le quantità e procedere al

pagamento quando desidera. Questo strumento semplifica l'esperienza d'acquisto, poiché consente di salvare temporaneamente i prodotti di interesse.

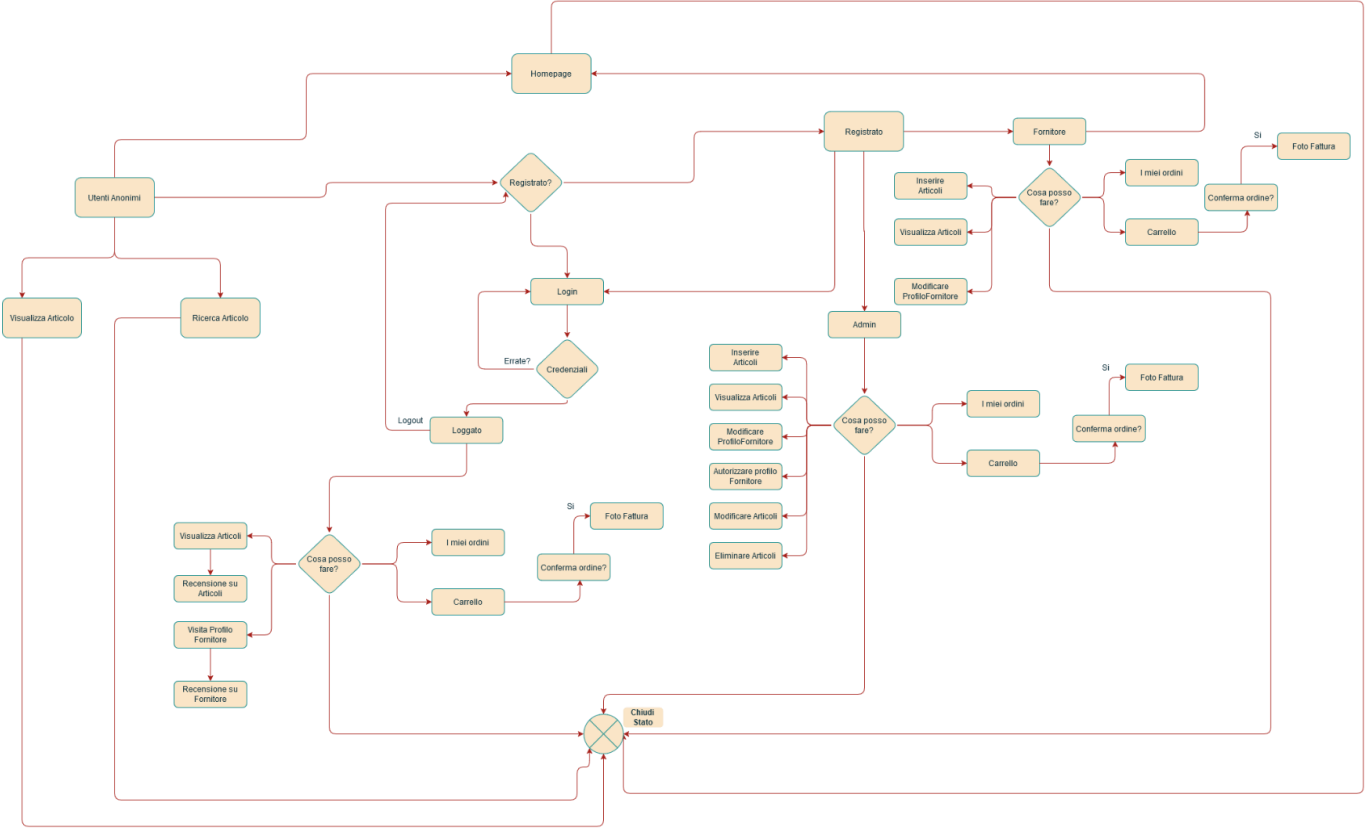
- **Pagamento tramite bonifico:** L'e-commerce richiede di effettuare il pagamento tramite bonifico bancario e scaricare la ricevuta in formato JPG.

## 2.Diagrammi

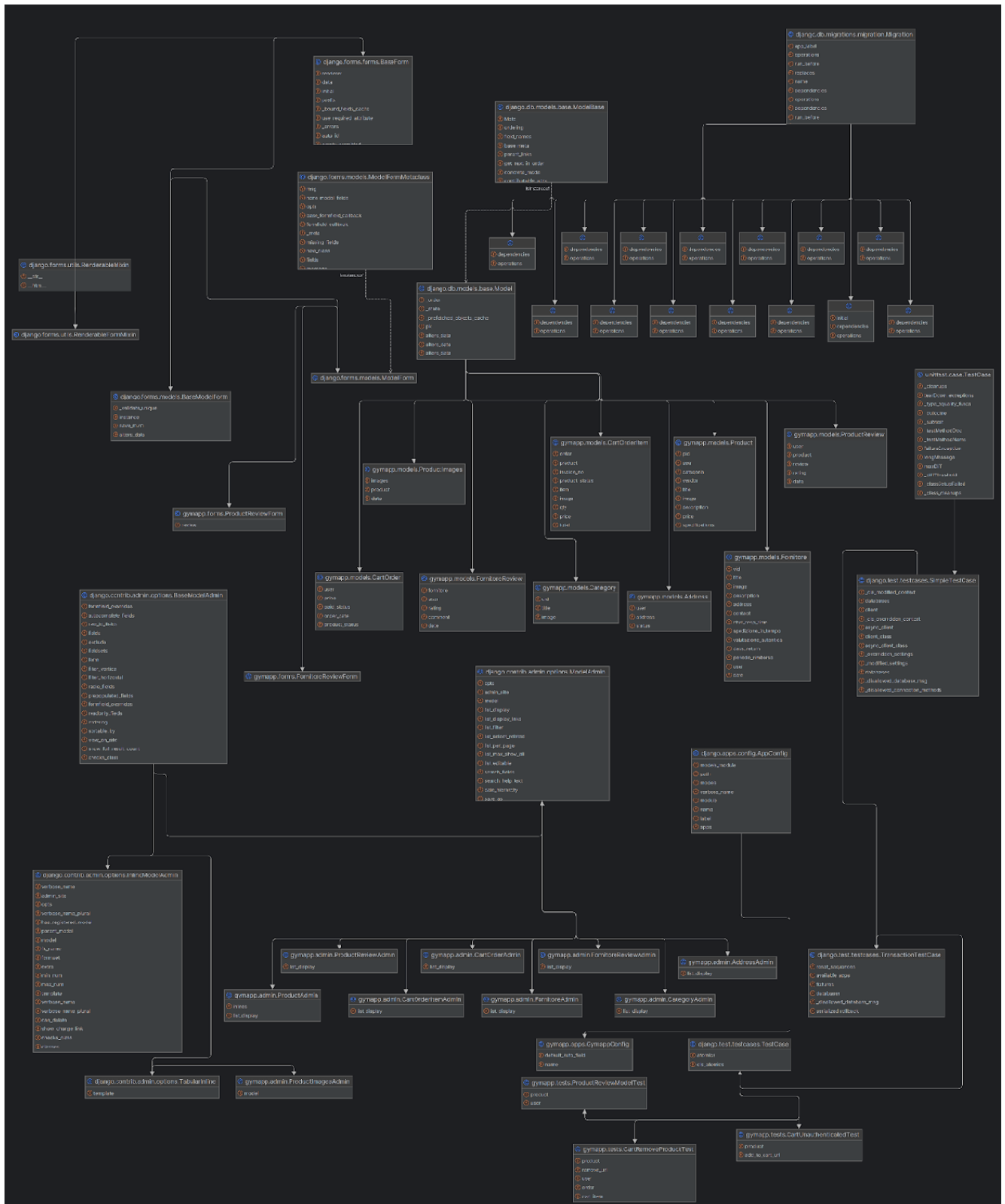
### 2.1 Diagramma dei casi d'uso



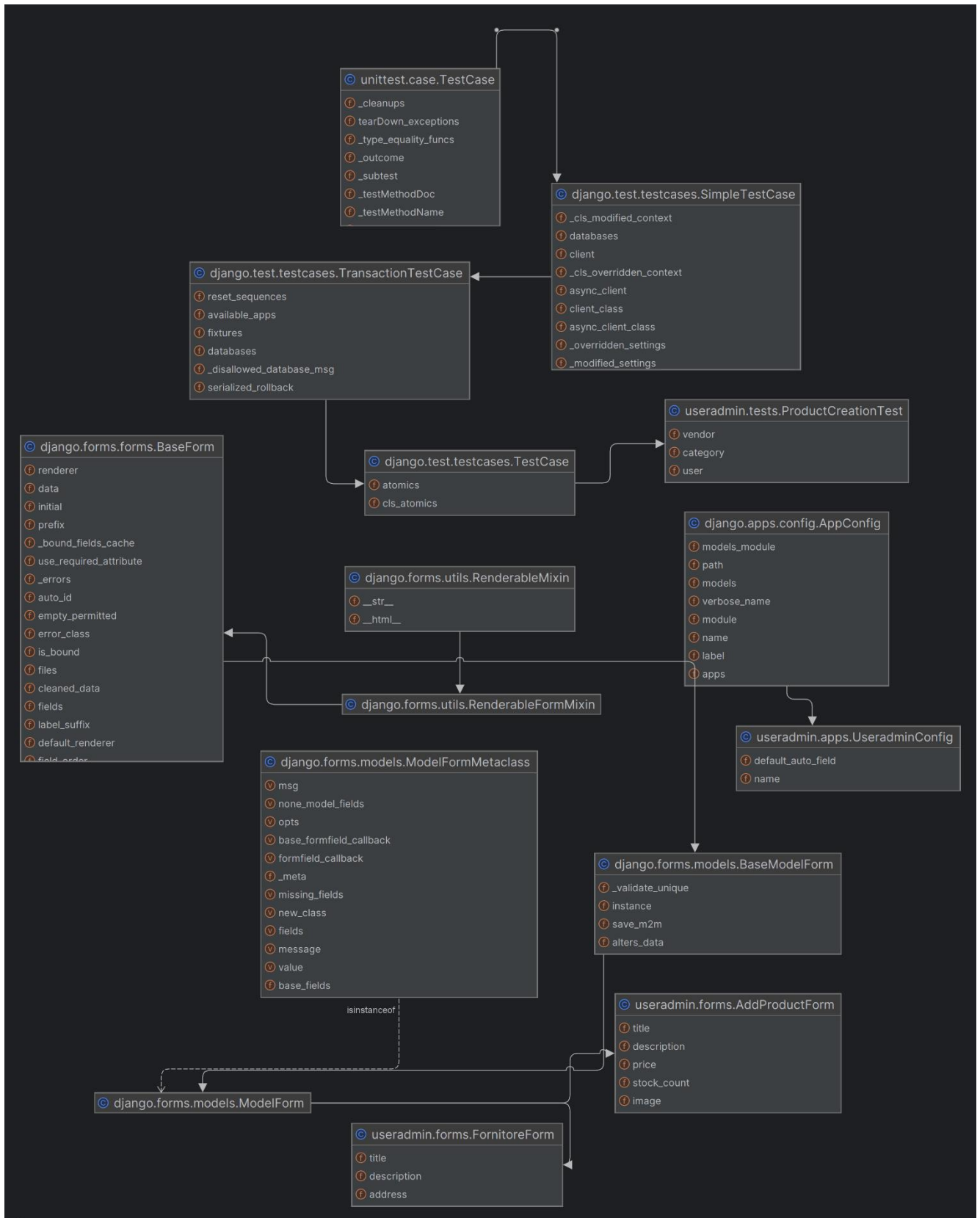
## 2.2 – Diagramma di utilizzo



## 2.3 – Diagramma delle classi-gymapp

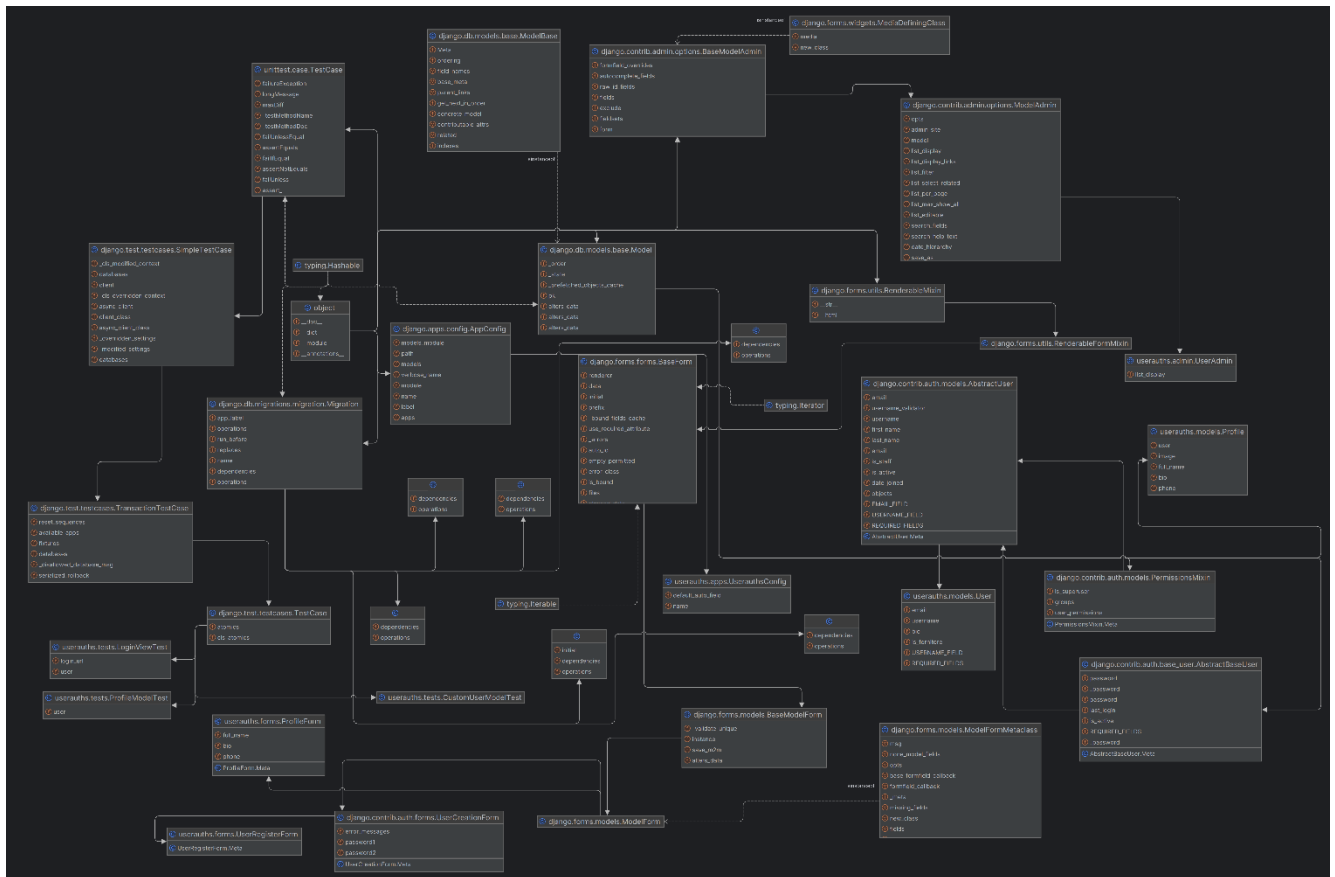


## 2.4- Diagramma delle classi – useradmin





## 2.5- Diagramma delle classi – userauths



### 3– Technologie usate

### 3.1 – Django

Il progetto è costruito con Django, un popolare framework di tipo MVC (Model View Controller) per lo sviluppo di applicazioni web in Python.

Django fornisce, di default, un'interfaccia di amministrazione che facilita la gestione del database dell'applicazione.

Una delle principali caratteristiche di Django consiste nella possibilità di riutilizzare agevolmente i suoi componenti, grazie all'architettura che suddivide il progetto in “app”. Questa suddivisione favorisce sia la leggibilità del codice sia la sua manutenibilità.

Dal punto di vista dell'MVC, la parte di "Controller" è implementata attraverso i file 'urls.py', che mappano gli indirizzi URL sulle relative richieste degli utenti. Le "View" sono funzioni (o metodi) Python che gestiscono la logica dell'applicazione, definendo il comportamento delle pagine in base alle azioni compiute dall'utente. Infine, la sezione "Model" si basa su classi Python che rappresentano le tabelle del database, rendendo più semplice e chiara l'interazione con i dati.

## 3.2 – Bootstrap

Bootstrap è un framework front-end ampiamente utilizzato per progettare template e interfacce web, soprattutto in ottica responsive, cioè in presenza di contenuti dinamici che devono adattarsi a diversi dispositivi. Grazie alla sua raccolta di componenti predefiniti, consente di sviluppare progetti in modo rapido ed efficiente.

## 3.3 – Javascript

JavaScript è un linguaggio di programmazione interpretato e orientato agli oggetti. È ampiamente utilizzato come linguaggio di Scripting client-side nelle pagine web, dove gestisce sia le funzionalità che il comportamento delle interfacce utente. In particolare, è responsabile delle interazioni dinamiche che si verificano quando l'utente esegue determinate azioni o eventi.

## 3.4 – Ajax

Ajax (Asynchronous JavaScript and XML) è una tecnica di sviluppo che consente di creare applicazioni web interattive. Grazie allo scambio asincrono di dati in background tra browser e server, è possibile aggiornare dinamicamente una pagina web senza richiederne il ricaricamento esplicito da parte dell'utente. In questo modo, le nuove informazioni vengono caricate senza interrompere o modificare il normale comportamento della pagina.

## 4. Descrizione del progetto

Ho sviluppato un sito di e-commerce specializzato, concepito per rendere la ricerca e la vendita di prodotti quanto più semplice e immediata possibile.

Il progetto è suddiviso in quattro applicazioni principali:

- **userauths:** gestisce la registrazione degli utenti e l'amministrazione degli indirizzi associati ai vari account.
- **gymapp:** si occupa delle funzionalità legate alle pagine del sito, come la homepage o la sezione vendite, ed è qui che si sono incontrate le maggiori difficoltà di implementazione.
- **gymcommerce:** rappresenta la base fondante del progetto, da cui prendono forma gli altri componenti.
- **useradmin:** dedicata alla gestione della pagina riservata ai venditori e all'admin.

### 4.1 – Utenti presenti

- **Utente/Anonimo:** non autenticato, può comunque esplorare liberamente il sito e visualizzare i prodotti insieme ai relativi dettagli. Non può, invece, leggere le recensioni né procedere all'acquisto finché non si registra.

- **Utente/Acquirente:** una volta effettuata la registrazione, può acquistare prodotti e consultare una sezione di “articoli consigliati” personalizzata, presente nella pagina iniziale, basata sulla cronologia degli acquisti.
- **Utente/Venditore:** dopo aver contattato l’amministratore per ottenere i permessi necessari, può mettere in vendita i propri prodotti direttamente sulla piattaforma e modificare la propria pagina venditore.
- **Admin:** è il supervisore del sistema, abilitato a cancellare o modificare gli articoli, autorizzare i venditori, controllare che i prodotti inseriti siano appropriati, gestire le registrazioni degli acquirenti e monitorare ordini e pagamenti.

## 4.2 – Gestione ordini

Solo gli utenti registrati e connessi al sito hanno la facoltà di acquistare prodotti. Nel momento in cui un utente aggiunge uno o più articoli al proprio carrello, il sistema genera automaticamente un nuovo ordine. Qualora invece il carrello fosse vuoto, viene mostrato il messaggio: “Il carrello è vuoto.”

È stata inoltre predisposta una pagina denominata “carrello”, che evidenzia i prodotti selezionati dall’utente. All’interno di questa pagina, sono mostrati:

- Il totale degli articoli pronti all’acquisto
- Un pulsante “Elimina prodotto”, che consente di rimuovere articoli specifici
- Tre ulteriori pulsanti: uno per avviare la procedura di checkout, uno per aggiornare le quantità dei prodotti e uno per continuare lo shopping, ritornando alla homepage.

Nel caso in cui l’utente desideri completare l’ordine, il sistema lo reindirizza a una pagina dedicata all’inserimento dei dati di spedizione e fatturazione. In questa sezione, è possibile specificare:

- Nome e cognome
- Indirizzo di spedizione (via, città, CAP, eventuale indirizzo secondario opzionale)
- Numero di telefono
- Indirizzo e-mail (con la possibilità di modificarlo rispetto a quello usato in fase di login)
- Annotazioni aggiuntive (opzionali)
- Metodo di pagamento scelto.

Dopo aver concluso il checkout, l’utente ha la possibilità di scaricare la fattura in formato JPG. Inoltre, il sistema invia automaticamente un’e-mail all’acquirente contenente un riepilogo dell’ordine effettuato.

### **4.3 – Vendere un prodotto**

La vendita di un prodotto può essere effettuata, in modo analogo all'acquisto, unicamente da utenti registrati, autenticati e autorizzati dall'amministratore. L'operazione avviene tramite il pulsante "Aggiungi prodotto", presente all'interno della dashboard dedicata a Fornitori e Admin.

Nel momento in cui il venditore decide di inserire un nuovo articolo, deve fornire tutti i dettagli necessari: nome, descrizione, immagine rappresentativa, categoria, eventuali categorie di prodotti correlati, quantità disponibili in magazzino e azienda produttrice. Durante la procedura, il sistema genera automaticamente un ID univoco per il prodotto; tutti i campi appena menzionati sono obbligatori da compilare.

Una volta completato il processo di inserimento, il prodotto diventa immediatamente visibile a qualsiasi utente del sito. Ogni volta, viene inviata una notifica tramite newsletter a tutti gli iscritti, informandoli della disponibilità del nuovo articolo. Il venditore, infine, ha la possibilità di visualizzare il proprio prodotto sia nell'elenco generale degli articoli in vendita sia all'interno del proprio profilo personale.

### **4.4 – Ruolo dell'Utente Admin**

L'amministratore del sistema svolge un ruolo di primaria importanza, in quanto ha la facoltà di conferire ad un utente registrato lo status di *Venditore*. Inoltre, l'Admin può intervenire sui prodotti inseriti dai venditori, modificandoli o eliminandoli qualora non risultino conformi ai requisiti stabiliti.

### **4.5 – Homepage**

Nella *homepage* sono presenti una barra di navigazione, un elenco di categorie di prodotti (che evidenziano la suddivisione in diverse classi) e i pulsanti per la registrazione e il login dell'utente. Qualora l'utente sia autenticato, nella parte inferiore della pagina compare una sezione dedicata ai prodotti "Consigliati per te".

### **4.6 –Metodo di recensire Prodotto e Venditore**

L'utente registrato ha la possibilità di consultare la scheda dettagliata dell'articolo/venditore di proprio interesse e di lasciare una recensione, assegnando un punteggio da 1 a 5 stelle (dove 1 indica una valutazione negativa e 5 un'eccellente qualità del prodotto/venditore). Analogamente, è possibile accedere alla scheda di un venditore e fornire una valutazione e un commento sulla sua affidabilità.

### **4.7 – Pagina contattaci**

Gli utenti hanno la possibilità di contattare l'amministratore per segnalare eventuali problemi o richiedere assistenza. In più, gli aspiranti venditori devono rivolgersi all'Admin per ottenere l'autorizzazione necessaria alla vendita all'interno della piattaforma.

## 5. Recommendation System

Nel progetto è stato implementato un sofisticato **sistema di raccomandazione**, progettato per suggerire prodotti di possibile interesse all'utente in base ai suoi acquisti precedenti. Questo sistema si articola in due viste principali:

1. ***get\_recommendations***: analizza lo storico degli acquisti dell'utente e utilizza questi dati come criterio per proporre nuovi articoli affini ai suoi interessi.
2. ***product\_detail\_view***: mostra, all'interno della scheda di ogni prodotto, una selezione di articoli appartenenti alla stessa categoria, al fine di offrire suggerimenti pertinenti.

Il processo di selezione si basa sul monitoraggio del numero totale di prodotti acquistati dall'utente, generando una lista di raccomandazioni composta esclusivamente da articoli che rispettano determinati criteri predefiniti. Successivamente, il sistema verifica quali articoli siano già stati acquistati in passato e, qualora le condizioni stabilite siano soddisfatte, aggiunge il prodotto alla sezione “**Consigliati per te**”. Gli articoli selezionati vengono infine mostrati in basso nella *homepage* dell'utente, presentando suggerimenti personalizzati per migliorare e arricchire l'esperienza di acquisto.

```
#Recommendation System
def get_recommendations(user):
    """
    Raccomanda prodotti basati sulla categoria degli articoli già acquistati dall'utente.
    """
    if not user.is_authenticated:
        return Product.objects.filter(status=True, featured=True)[:5] # Prodotti in evidenza per utenti non autenticati

    # Recupera tutte le categorie dei prodotti acquistati dall'utente
    purchased_items = CartOrderItem.objects.filter(
        order__user=user,
        order__product_status="archived" # Solo ordini completati
    ).values_list('product__categoria', flat=True).distinct()

    # Raccomanda prodotti dalla stessa categoria degli articoli acquistati
    recommended_products = Product.objects.filter(
        categoria__in=purchased_items, # Prodotti con la stessa categoria
        status=True, # Solo prodotti attivi
        in_stock=True # Solo prodotti in stock
    ).exclude(
        cartorderitem__order__user=user # Escludi prodotti già acquistati dall'utente
    ).annotate(
        relevance=Count('id')
    ).order_by('-relevance')[:5] # Limita a 5 prodotti

    return recommended_products
```

```
#Product Review form
if request.method == 'POST':
    form = ProductReviewForm(request.POST)
    if form.is_valid():
        review = form.save(commit=False)
        review.product = prodotto
        review.user = request.user # Assicurati che l'utente sia autenticato
        review.save()
        return redirect('gymapp:dettagli_prodotto', pid=pid) # Ricarica la pagina prodotto
else:
    form = ProductReviewForm()
```

## 6 – Test

### 6.1 –Test Login

Il login di un utente viene controllato attraverso il “LoginViewTest” che verifica la correttezza dell’inserimento delle credenziali; provato sia con credenziali corrette che errate.

Naturalmente se le credenziali vengono inserite correttamente, si effettua il login.

```
class LoginViewTest(TestCase):
    def setUp(self):
        """
        Configura il contesto per il test creando un utente di prova.
        """
        # Crea un utente con credenziali valide
        self.user = get_user_model().objects.create_user(
            email="testuser@example.com", # Email dell'utente
            username="testuser",          # Nome utente
            password="testpassword",      # Password dell'utente
            bio="Test bio"                # Bio dell'utente
        )
        # Ottiene l'URL della view di login utilizzando il nome della route
        self.login_url = reverse('userauths:sign-in')

    def test_login_with_invalid_credentials(self):
        """
        Testa il login con credenziali errate.
        """
        # Esegue una richiesta POST alla view di login con email e password errate
        response = self.client.post(self.login_url, {
            'email': 'wrongemail@example.com', # Email non registrata
            'password': 'wrongpassword'        # Password sbagliata
        })
        self.assertEqual(response.status_code, 302) # Controlla che venga restituito un reindirizzamento
        self.assertRedirects(response, reverse('gymapp:base')) # Controlla che reindirizzi alla home
        # Estrae i messaggi di risposta generati
        messages = list(response.wsgi_request.messages)
        self.assertEqual(len(messages), 1) # Controlla che ci sia un solo messaggio
        # Verifica che il messaggio contenga l'avviso per credenziali errate
        self.assertIn("L'utente wrongemail@example.com non esiste.", str(messages[0]))

    def test_login_with_correct_credentials(self):
        """
        Testa il login con credenziali corrette.
        """
        # Esegue una richiesta POST alla view di login con email e password corrette
        response = self.client.post(self.login_url, {
            'email': 'testuser@example.com', # Email valida
            'password': 'testpassword'        # Password valida
        })
        self.assertEqual(response.status_code, 302) # Controlla che venga restituito un reindirizzamento
        self.assertRedirects(response, reverse('gymapp:base')) # Controlla che reindirizzi alla home
        # Estrae i messaggi di risposta generati
        messages = list(response.wsgi_request.messages)
        self.assertEqual(len(messages), 1) # Controlla che ci sia un solo messaggio
        # Verifica che il messaggio confermi l'accesso riuscito
        self.assertIn("Hai fatto l'accesso correttamente.", str(messages[0]))
        # Controlla che l'utente sia autenticato
        self.assertTrue(response.wsgi_request.user.is_authenticated)
```



## 6.2- Test inserimento articolo

Attraverso il “ProductCreationTest” si verifica che i campi vengano inseriti correttamente e che i campi siano tutti completi (nessun campo vuoto).

```
class ProductCreationTest(TestCase):
    def setUp(self):
        """
        Imposta il contesto per il test:
        - Crea un utente fornitore.
        - Crea una categoria di prodotto.
        - Crea un fornitore associato all'utente.
        """
        # Creazione di un utente di tipo fornitore
        self.user = get_user_model().objects.create_user(
            email="fornitore@example.com", # Email del fornitore
            username="fornitore",          # Username del fornitore
            password="password123",        # Password del fornitore
            is_fornitore=True              # Flag per identificare il fornitore
        )

        # Creazione di una categoria di test
        self.category = Category.objects.create(
            title="Manubri", # Nome corretto del campo
        )

        # Creazione di un fornitore di test associato all'utente
        self.vendor = Fornitore.objects.create(
            user=self.user,                # Utente associato al fornitore
            title="Palestra Test",          # Nome del fornitore
            description="Descrizione del fornitore", # Descrizione
            contact="1234567890"            # Contatto del fornitore
        )

    def test_create_product(self):
        """
        Test per verificare che un prodotto venga creato correttamente
        con tutti i campi richiesti.
        """
        # Creazione di un prodotto di test
        product = Product.objects.create(
            user=self.user,                # Utente che crea il prodotto
            categoria=self.category,        # Categoria del prodotto
            vendor=self.vendor,             # Fornitore del prodotto
            title="Manubrio 10kg",          # Titolo del prodotto
            image="manubrio.jpg",          # Immagine del prodotto
            description="Manubrio di test da 10kg", # Descrizione del prodotto
            price=25.99,                   # Prezzo del prodotto
            specifications="Specifiche di test", # Specifiche tecniche
            stock_count="20",              # Quantità in magazzino
            product_status="available",     # Stato del prodotto
            in_stock=True,                  # Disponibilità
            featured=False,                 # Prodotto in evidenza (False)
            digital=False,                  # Prodotto non digitale (False)
            categ_type="manubrio"           # Tipo di categoria
        )

        # Verifica che il prodotto sia stato creato correttamente
        self.assertEqual(Product.objects.count(), 1) # Controlla che ci sia un solo prodotto nel database
        self.assertEqual(product.title, "Manubrio 10kg") # Verifica il titolo
        self.assertEqual(product.price, 25.99) # Verifica il prezzo
        self.assertEqual(product.stock_count, "20") # Verifica la quantità in magazzino
        self.assertTrue(product.in_stock) # Verifica che il prodotto sia disponibile
        self.assertEqual(product.categ_type, "manubrio") # Verifica il tipo di categoria
```

## 6.3 – Test reindirizzamento oggetto

Attraverso il “CartUnauthenticatedTest” il sistema verifica che l’utente sia autenticato prima di consentire l’acquisto di un articolo. Qualora l’utente non abbia effettuato l’accesso, viene automaticamente reindirizzato alla pagina di login.

Se l’utente è già autenticato, il sistema procede con un ulteriore controllo sulla correttezza dell’inserimento dell’articolo, verificando che i dati siano validi e conformi ai requisiti previsti prima di completare l’acquisto.

```
class CartUnauthenticatedTest(TestCase):
    def setUp(self):
        """
        Configura il contesto del test creando un prodotto di esempio.
        """
        self.product = Product.objects.create(
            title="Manubrio da palestra",
            price=29.99,
            stock_count=10
        )
        self.add_to_cart_url = reverse('gymapp:add_to_cart', args=[self.product.pk])

    def test_add_to_cart_redirects_if_not_authenticated(self):
        """
        Testa che un utente non autenticato venga reindirizzato alla pagina home
        quando tenta di aggiungere un prodotto al carrello.
        """
        response = self.client.post(self.add_to_cart_url, {
            'quantity': 1
        })
        # Controlla che l'utente sia reindirizzato alla pagina di login
        login_url = reverse('userauths:sign-in') # Modifica questo in base al tuo URL di login
        self.assertRedirects(response, f"{login_url}?next={self.add_to_cart_url}")
```

## 6.4 – Test per rimuovere un elemento dal carrello

Attraverso il “CartRemoveProductTest” si crea un utente di prova, aggiunge un articolo al carrello che viene successivamente rimosso (dal carrello).

```
class CartRemoveProductTest(TestCase):
    def setUp(self):
        # Crea un utente di prova
        self.user = get_user_model().objects.create_user(
            email="testuser@example.com", # Email dell'utente di prova
            username="testuser",          # Username dell'utente di prova
            password="testpassword"       # Password dell'utente di prova
        )
        # Autentica l'utente appena creato
        self.client.login(email="testuser@example.com", password="testpassword")

        # Crea un prodotto di prova con un titolo e un prezzo specificati
        self.product = Product.objects.create(title="Test Product", price=10.99)

        # Crea un ordine associato all'utente di prova
        self.order = CartOrder.objects.create(user=self.user)

        # Crea un elemento del carrello associato al prodotto e all'ordine
        self.cart_item = CartOrderItem.objects.create(
            product=self.product, # Prodotto associato all'elemento del carrello
            qty=1,                # Quantità del prodotto
            total=10.99,          # Totale calcolato per il prodotto
            order=self.order      # Ordine a cui appartiene l'elemento del carrello
        )

        # Definisce l'URL per rimuovere l'elemento dal carrello, utilizzando il primary key dell'elemento
        self.remove_url = reverse('gymapp:remove_from_cart', args=[self.cart_item.pk])

    def test_remove_product_from_cart(self):
        # Effettua una richiesta POST per rimuovere l'elemento dal carrello
        response = self.client.post(self.remove_url)

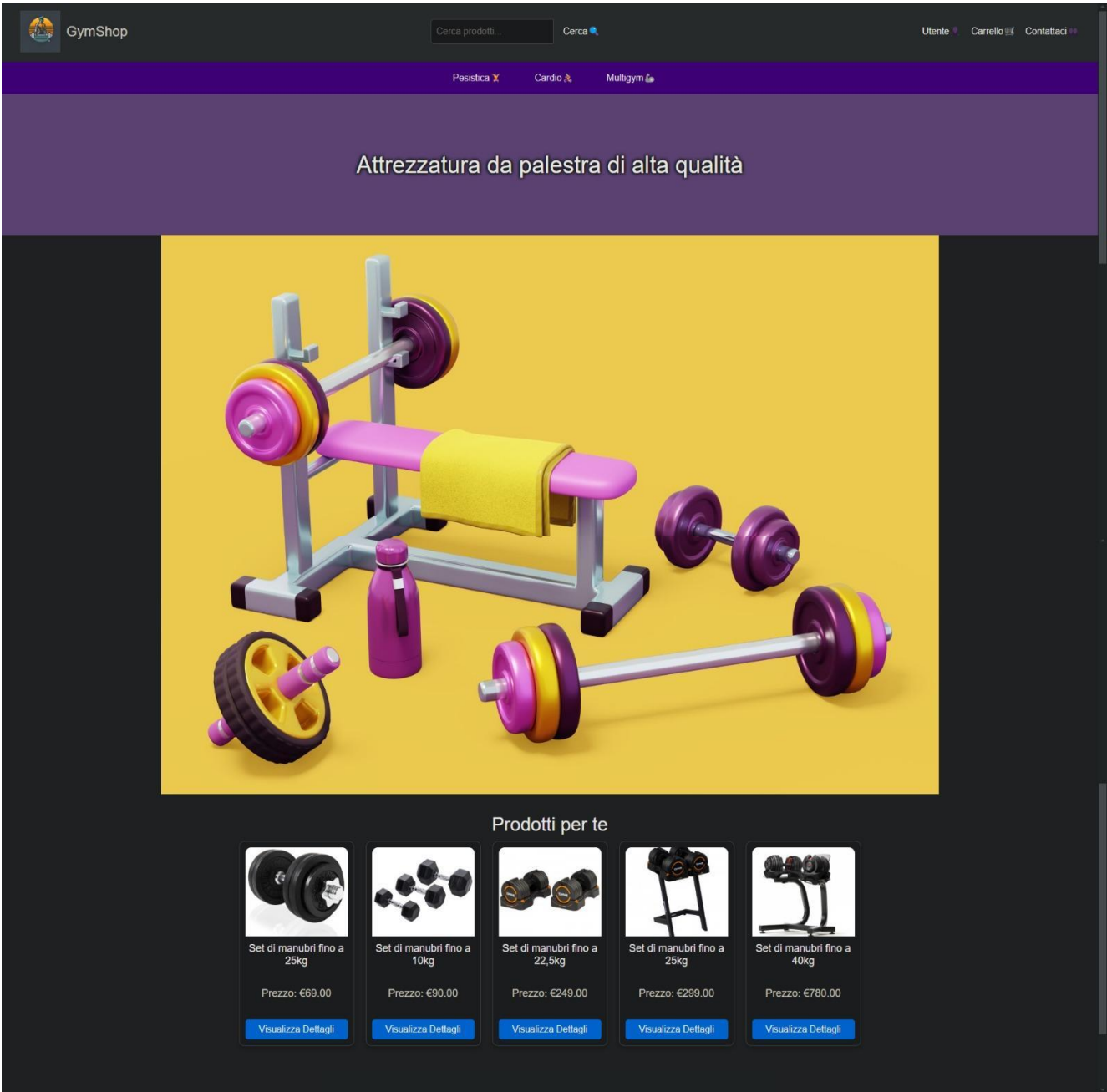
        # Controlla che l'elemento del carrello sia stato eliminato dal database
        self.assertFalse(CartOrderItem.objects.filter(pk=self.cart_item.pk).exists())

        # Verifica che la risposta sia un reindirizzamento (codice HTTP 302)
        self.assertEqual(response.status_code, 302)
```



# 7– Screenshot

## 7.1 – Homepage



## 7.2 – Profilo utente



## 7.3 – Dashboard Vendite

### Dashboard

Tutti i dati sulla tua attività qui

Dashboard

Prodotti

Profilo Fornitore

Aggiungi Prodotto

Utenti

Logout

Utenti iscritti al sito

10

Ordini

25

Prodotti

47

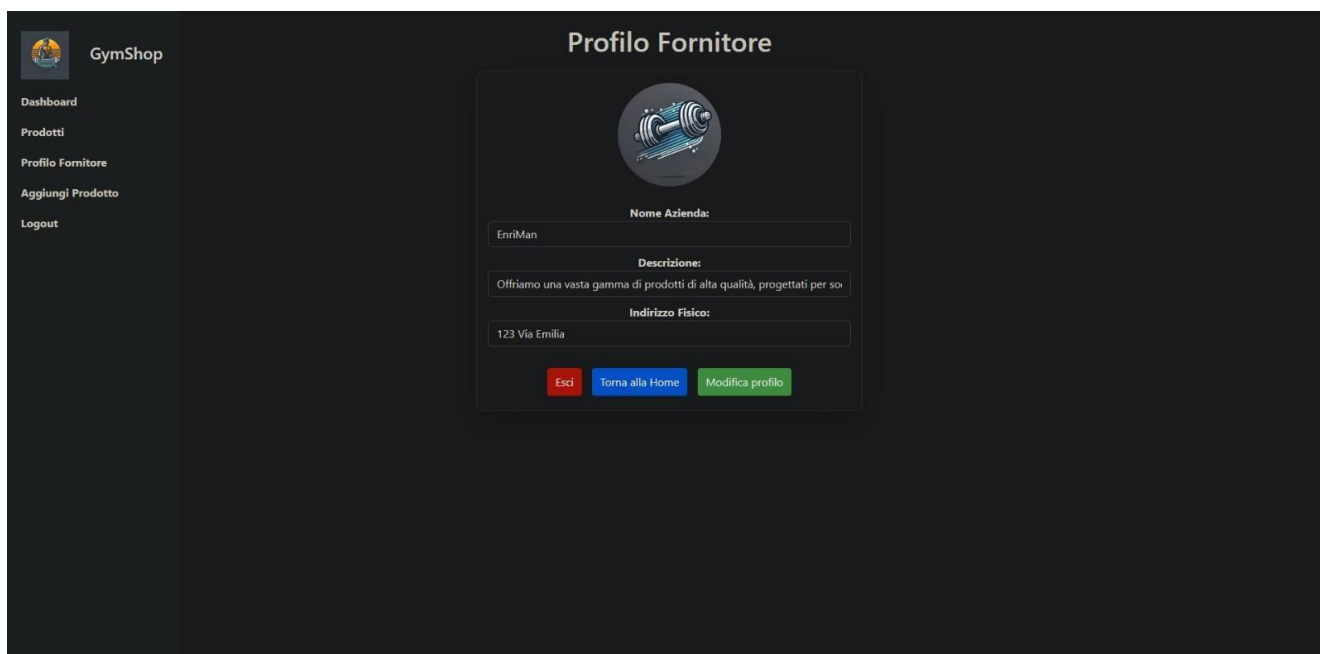
Tutte le categorie di prodotti

12

### Ordini recenti

ID Ordine	Acquirente	Data	Total	Stato Pagamento
#sku4697	gabbuz10@gmail.com	Jan. 27, 2025, 3:52 p.m.	€84.00	Not Paid
#sku4697	gabbuz10@gmail.com	Jan. 30, 2025, 7:39 a.m.	€84.00	Not Paid
#sku4735	gabbuz10@gmail.com	Jan. 30, 2025, 7:54 a.m.	€369.00	Not Paid
#sku4697	enri@gmail.com	Feb. 1, 2025, 11:24 a.m.	€84.00	Not Paid
#sku2171	enri@gmail.com	Feb. 1, 2025, 11:25 a.m.	€123.00	Not Paid
#sku4697	gabbuz10@gmail.com	Jan. 30, 2025, 8:08 a.m.	€28.00	Not Paid
#sku2171	gabbuz10@gmail.com	Feb. 3, 2025, 8:37 a.m.	€369.00	Not Paid
#sku4735	gabbuz10@gmail.com	Feb. 3, 2025, 2:41 p.m.	€123.00	Not Paid
#sku1935	gabbuz10@gmail.com	Feb. 3, 2025, 2:42 p.m.	€639.00	Not Paid
#sku7272	gabbuz10@gmail.com	Feb. 3, 2025, 2:51 p.m.	€329.00	Not Paid
#sku2171	gabbuz10@gmail.com	Feb. 3, 2025, 2:56 p.m.	€123.00	Not Paid
#sku4697	gabbuz10@gmail.com	Feb. 3, 2025, 2:56 p.m.	€28.00	Not Paid
#sku4697	gabbuz10@gmail.com	Feb. 3, 2025, 2:57 p.m.	€28.00	Not Paid


## 7.4 – Profilo di Utente Fornitore



**GymShop**

- Dashboard
- Prodotti
- Profilo Fornitore
- Aggiungi Prodotto
- Logout

### Profilo Fornitore



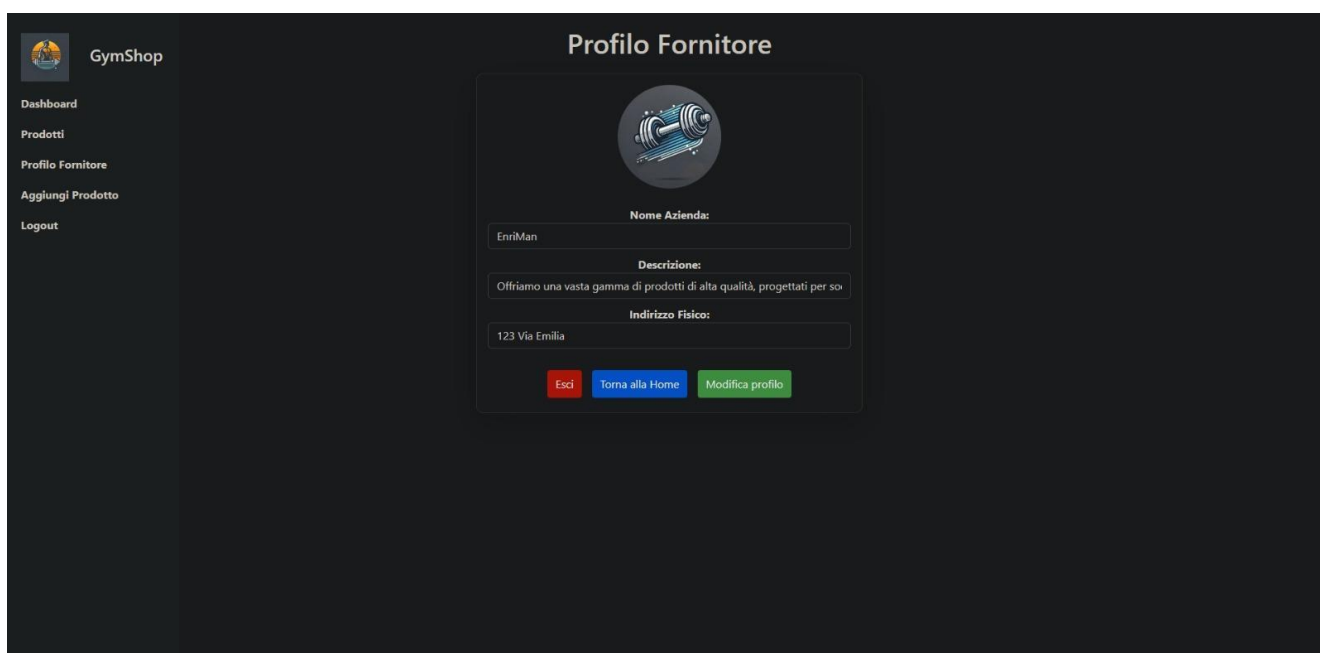
**Nome Azienda:**  
EnriMan

**Descrizione:**  
Offriamo una vasta gamma di prodotti di alta qualità, progettati per so

**Indirizzo Fisico:**  
123 Via Emilia

[Esci](#) [Torna alla Home](#) [Modifica profilo](#)


## 7.5 – Visualizzazione del carrello



**GymShop**

- Dashboard
- Prodotti
- Profilo Fornitore
- Aggiungi Prodotto
- Logout

### Profilo Fornitore



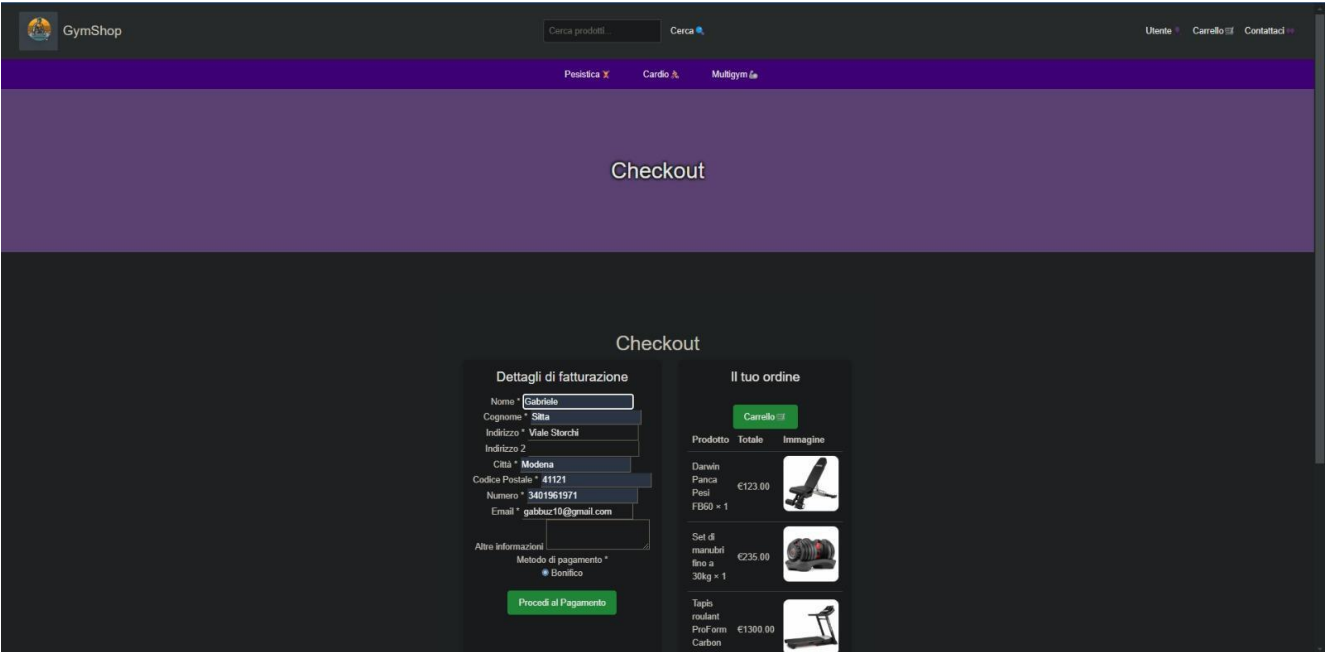
**Nome Azienda:**  
EnriMan

**Descrizione:**  
Offriamo una vasta gamma di prodotti di alta qualità, progettati per so

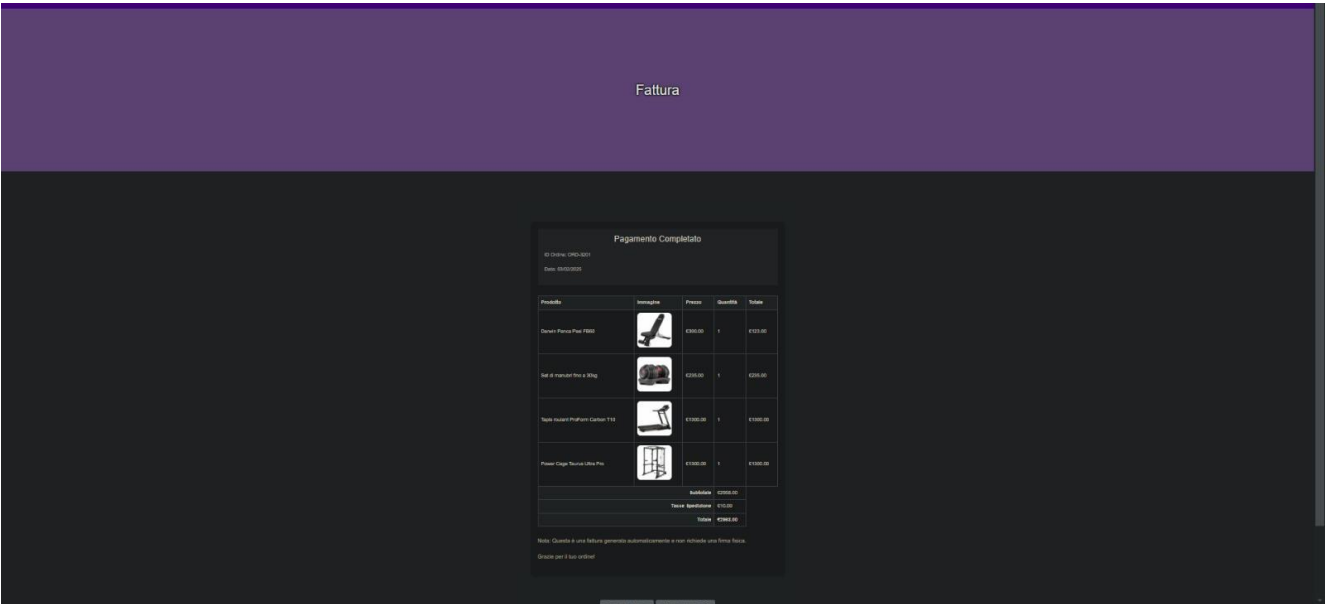
**Indirizzo Fisico:**  
123 Via Emilia

[Esci](#) [Torna alla Home](#) [Modifica profilo](#)

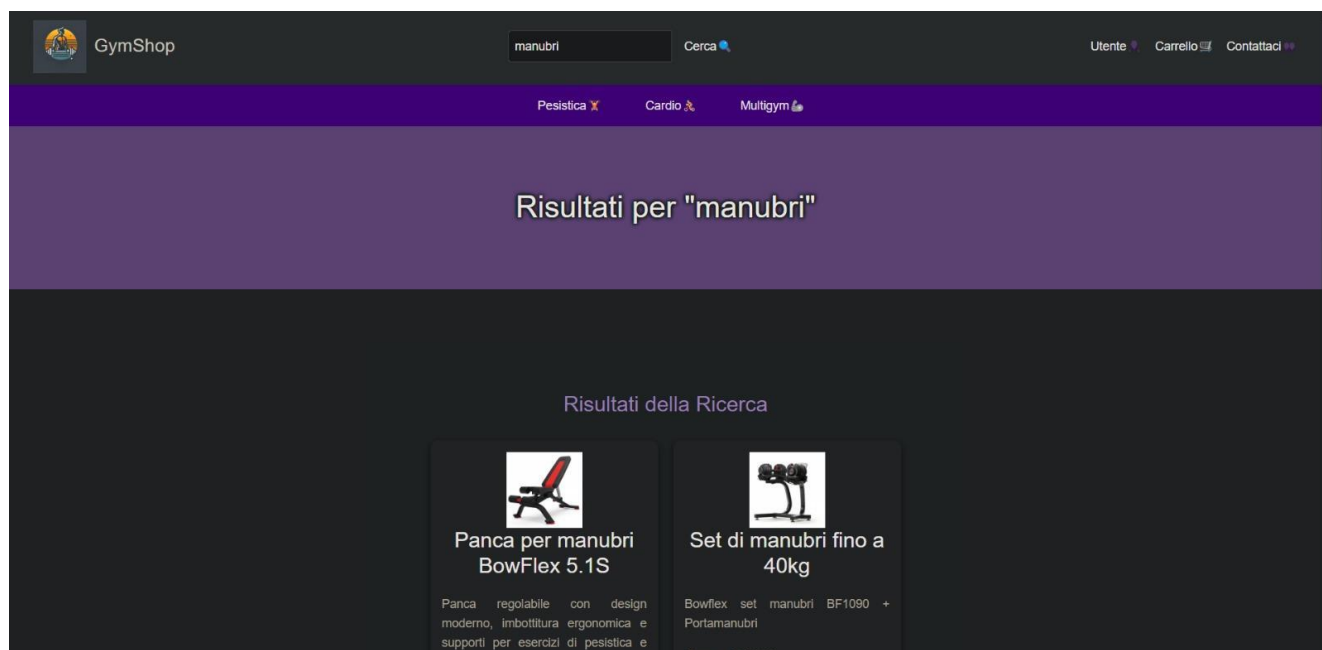
## 7.6 – Pagina di Checkout



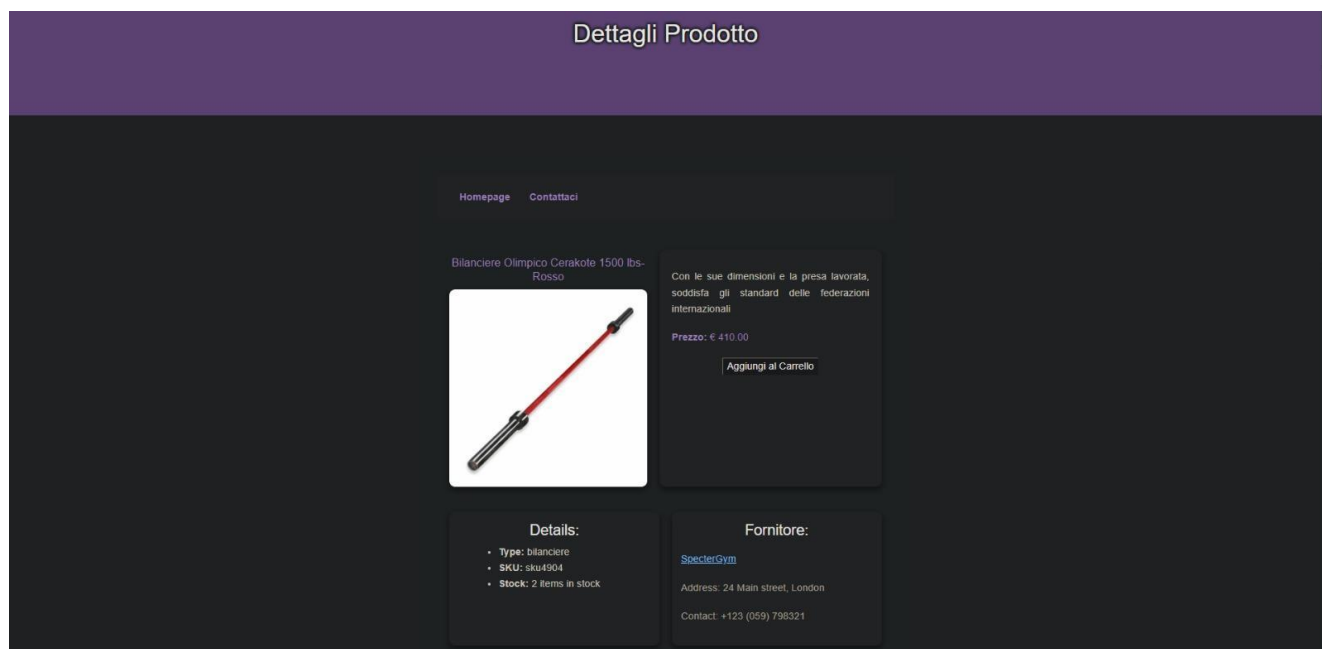
## 7.7 – Fattura



## 7.8 – Risultati di una ricerca



## 7.9 – Dettagli Prodotto



## 7.10 – Dettagli Fornitore

Pesistica

Cardio

Multigym

Dettagli Fornitore

Prodotti di SpecterGym

Homepage

Contattaci:specter@gmail.com

Informazioni sul Fornitore

Nome: SpecterGym

Descrizione: SpecterGym: qualità in un nome

Indirizzo: 24 Main street, London

Telefono: +123 (059) 798321

Fornitore Dal: 2025

mailto:specter@gmail.com

## 7.11 – Pagina in base alla categoria selezionata

Cerca prodotti...

Cerca

Utente

Carrello

Contattaci

Pesistica

Cardio

Multigym

Panche

[Darwin Panca Pesì FB60](#)

Panche

Panca inclinabile e regolabile, dotata di supporti per le gambe, ideale per addominali e allenamenti di forza

€300.00

Lugym

Aggiungi al carrello

[Taurus Studio Panca Pesì B970](#)

Panche

Panca da palestra regolabile, robusta e versatile, perfetta per esercizi di sollevamento pesi e workout domestici

€549.00

Beasgym

Aggiungi al carrello

[Taurus Panca Pesì B930](#)

Panche

Panca inclinabile con struttura stabile e imbottiture di alta qualità. Adatta per una vasta gamma di esercizi fitness

€600.00

Gymappo

Aggiungi al carrello

[Panca per manubri BowFlex 5.1S](#)

Panche

Panca regolabile con design moderno, imbottitura ergonomica e supporti per esercizi di pesistica e allenamenti del core

€300.00

Lugym

Aggiungi al carrello

[Set di panca per pesi Duke Fitness](#)

Panche

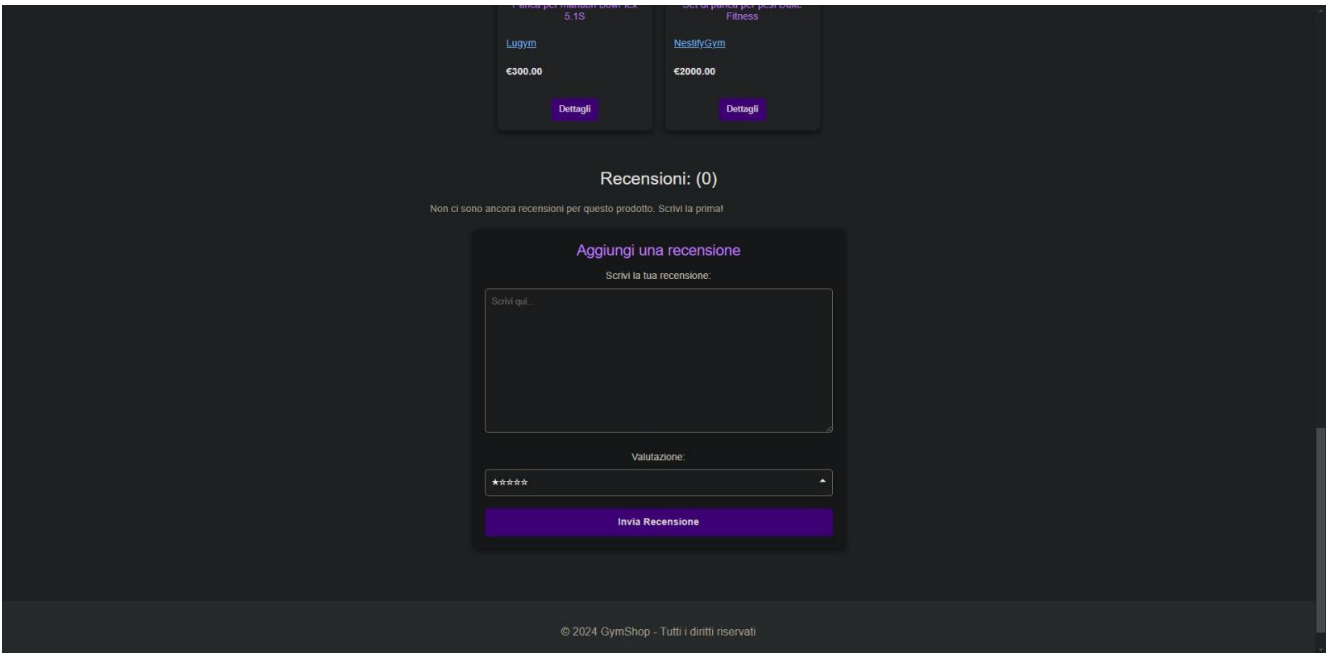
Stazione multifunzione con panca regolabile, supporti per bilanciere e set di pesi inclusi. Ideale per esercizi completi di potenziamento muscolare

€2000.00

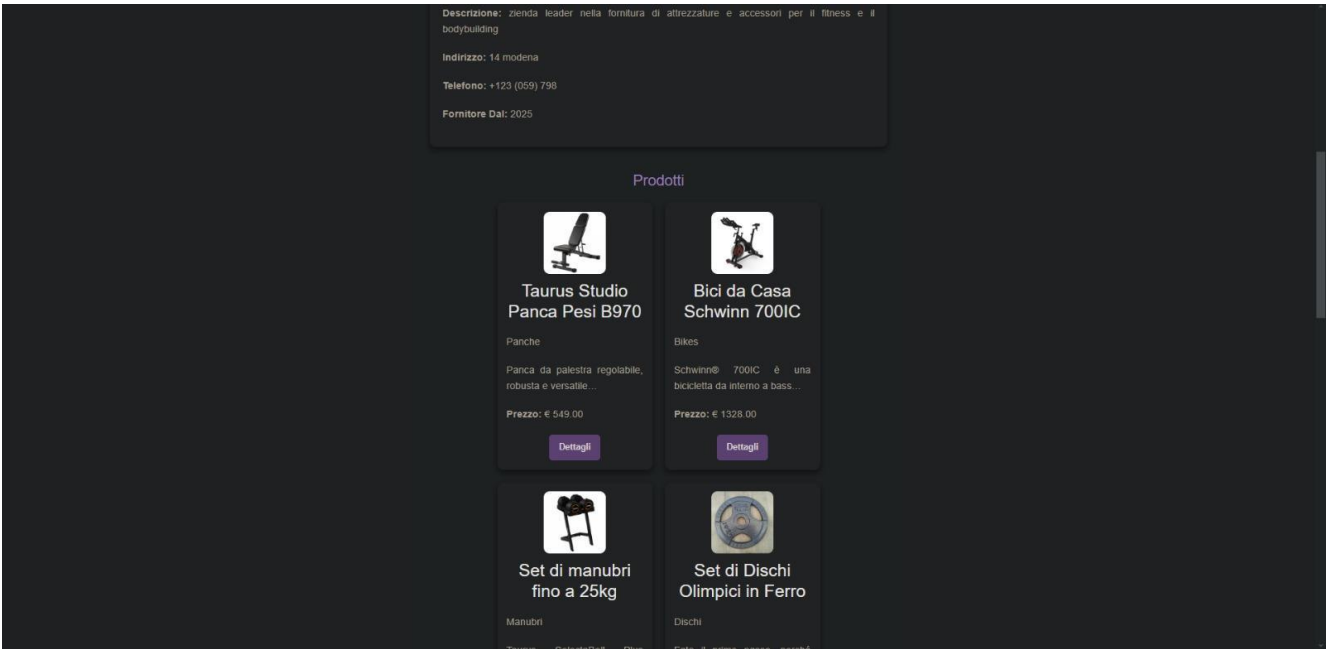
NeslityGym

Aggiungi al carrello

## 7.12 – Recensione Prodotto



## 7.13 – Recommendation System



## 8 – Conclusioni

Al termine di questo percorso, provo una grande soddisfazione per il lavoro svolto e per tutto ciò che ho imparato lungo il cammino. È stata la mia prima esperienza con **Django** e, per la prima volta, mi sono trovato ad affrontare da solo un progetto di questa portata. Se all'inizio mi sembrava una sfida complessa e quasi insormontabile, oggi posso guardare al risultato con orgoglio, consapevole di aver acquisito competenze e sicurezza che prima non avevo.

Questo progetto è stato molto più di un esercizio tecnico: è stato un viaggio di crescita, fatto di tentativi, errori e successi. Ho imparato a **lavorare con metodo, a gestire le difficoltà con pazienza e a trovare soluzioni creative ai problemi incontrati**. Ogni ostacolo superato è diventato un piccolo traguardo, ogni revisione ha reso il progetto più solido e ogni errore si è trasformato in un'opportunità di miglioramento.

So che ci sarebbero stati altri aspetti da esplorare, come **l'integrazione delle procedure di reso e rimborso o l'aggiunta di un forum per la community**, ma ciò non sminuisce il valore di ciò che ho costruito. Anzi, mi lascia la voglia di continuare a migliorare, di ampliare il progetto e di sperimentare nuove soluzioni.

Questa esperienza mi ha insegnato che la programmazione non è solo un insieme di regole e codice, ma una forma di creatività, un modo per dare vita a idee e progetti concreti. E, più di tutto, mi ha dato la consapevolezza che, con impegno e determinazione, posso affrontare qualsiasi sfida.

Gabriele Sitta

Matricola 165340