Relazione Tecnica relativa alla Web Application "Gruppo d’Acquisto"

Descrizione del Progetto

Il progetto prevedeva la realizzazione di un sito web che offre un servizio di prenotazione di prodotti per un gruppo di acquisto. La web application consente agli utenti di eseguire le seguenti operazioni:
1. Accesso alla piattaforma di prenotazione tramite email e password.
2. Visualizzazione di un elenco di prodotti disponibili e delle quantità disponibili per ogni lotto.
3. Prenotazione di quantità specifiche di un certo lotto.
4. Visualizzazione dell'elenco dei lotti e delle quantità prenotate dall'utente.

Strumenti Utilizzati

Per la realizzazione del sito web sono stati impiegati i seguenti strumenti:

- **Visual Studio Code**: Utilizzato per la scrittura e il testing del codice.
- **HTML** con libreria **Bootstrap 5 e Jinja**: Utilizzato per la visualizzazione lato front-end. Bootstrap 5 è stato impiegato per creare un'interfaccia utente responsiva e moderna.
- **JavaScrip**t: Utilizzato per popolare dinamicamente le pagine HTML, migliorando l'interattività e la reattività dell'applicazione.
- **Python 3.12**: Utilizzato per la realizzazione delle applicazioni di creazione e interrogazione dei database, nonché per la creazione delle API da cui richiedere i dati. Le librerie utilizzate includono ***`os`, `json`, `datetime`, `locale`, `re`, `functools`***.
- **MySQL**: Utilizzato come database per memorizzare informazioni sugli utenti, prodotti, lotti e prenotazioni.
- **Flask**: Utilizzato come framework web per gestire il collegamento tra Python e il database. Le librerie Flask utilizzate includono:
  - **Flask-SQLAlchemy**: Per gestire l'interazione con il database in modo ORM (Object-Relational Mapping).
  - **sqlalchemy-serializer**: Per serializzare gli oggetti del database in JSON.
  - **flask-bcrypt**: Per la gestione sicura delle password, mediante hashing.
  - **flask-limiter**: Per implementare il rate limiting e proteggere l'applicazione da abusi e attacchi di forza bruta.

Funzionalità Implementate

1. **Autenticazione:**
   - Gli utenti possono registrarsi e accedere alla piattaforma utilizzando email e password.
   - Le password sono gestite in modo sicuro utilizzando la libreria `flask-bcrypt` per hashing.

2. **Visualizzazione Prodotti:**
   - Gli utenti possono visualizzare un elenco di prodotti disponibili, inclusi dettagli come quantità disponibili per ogni lotto.

3. **Prenotazione:**
   - Gli utenti possono prenotare quantità specifiche di prodotti da lotti disponibili.
   - Le prenotazioni vengono memorizzate nel database e associate all'utente che le ha effettuate.

4. **Visualizzazione Prenotazioni:**
   - Gli utenti possono visualizzare un elenco dei lotti e delle quantità prenotate.

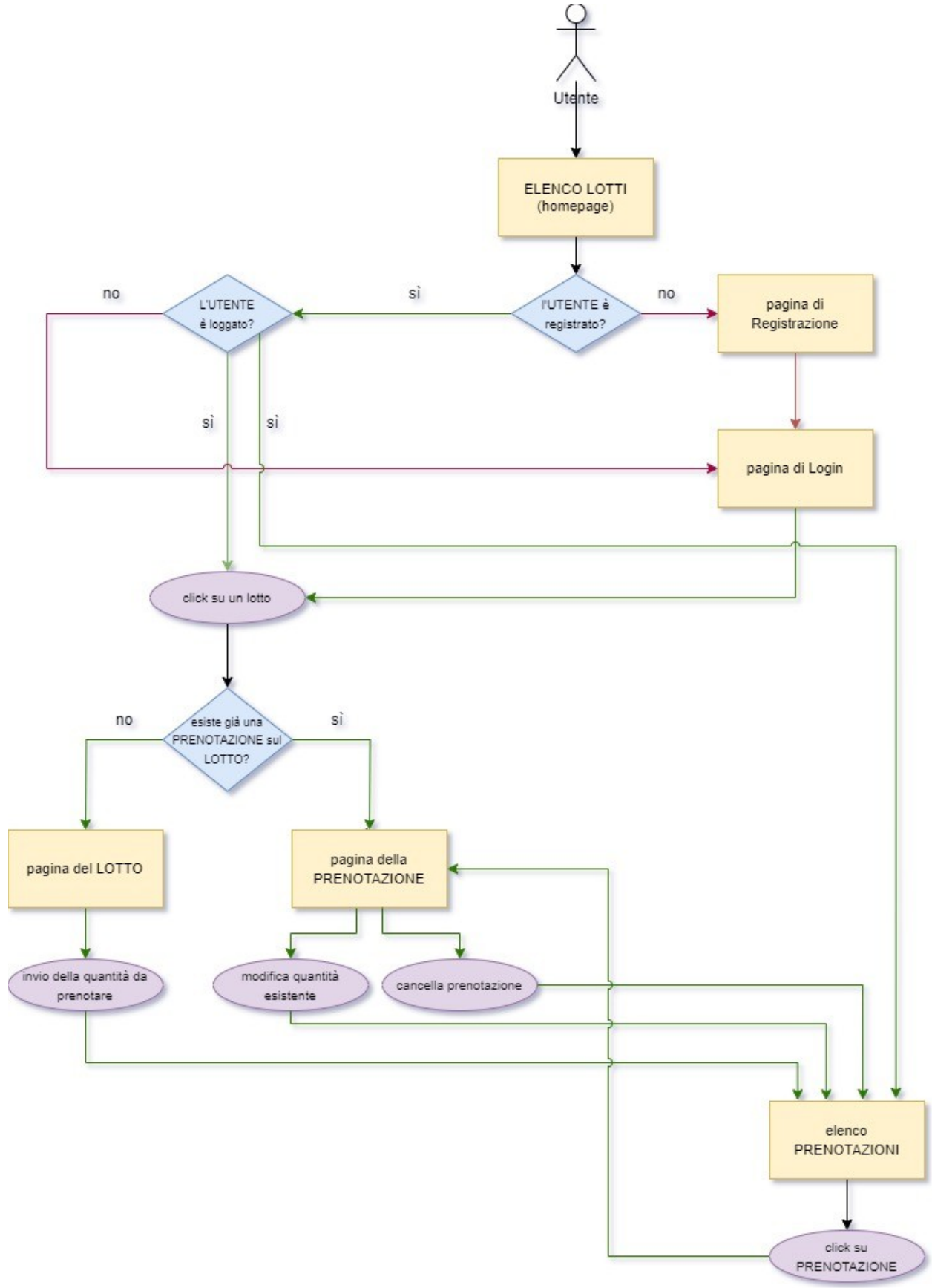
5. **Prerogative amministratore (admin):**
   - Inserimento di nuovi produttori.
   - Inserimento di nuovi prodotti.
   - Inserimento di nuovi lotti.
   - Operazioni sugli utenti.

Struttura dell’applicazione

```
gruppo_acquisto/
|
├── app.py                     # File principale dell'applicazione
├── settings.py                # Configurazioni dell'applicazione
├── models.py                  # Definizioni dei modelli del database
├── rehash_passwords.py        # File per il rehash delle password
├── requirements.txt           # File dei requisiti per le dipendenze
├── database/                  # Directory per database e file dati (Json o CSV)
|   ├── data_json/             # Cartella per i file Json
|   ├── db.sqlite3             # File database
├── static/                    # Directory per file statici (CSS, JS, immagini)
|   ├── imgs/                  # Cartella per le immagini
|   ├── scripts/
|       ├── lotti.js           # File JavaScript per la gestione dei lotti
|       ├── prenotazioni.js    # File JavaScript per la gestione delle prenotazioni
|   ├── styles/
|   |   ├── style.css          # File CSS personalizzato
├── templates/                 # Directory per i template HTML
|   ├── _layout.html           # Template di layout base
|   ├── home.html              # Template della homepage
|   ├── login.html             # Template della pagina di login
|   ├── registrazione.html     # Template della pagina di registrazione
|   ├── prenotazione.html      # Template della pagina di prenotazione
|   ├── prenotazioni.html      # Template dell'elenco prenotazioni
|   ├── lotto.html             # Template della pagina di un lotto
|   ├── nuovo_produttore.html  # Template per aggiungere un nuovo produttore
|   ├── nuovo_prodotto.html    # Template per aggiungere un nuovo prodotto
|   ├── nuovo_lotto.html       # Template per aggiungere un nuovo lotto
|   ├── gestisci_utenti.html   # Template per la gestione degli utenti
|   ├── includes/
|   |   ├── flash.html         # Template per i messaggi flash
|   |   ├── footer.html        # Template per il footer
|   |   ├── head.html          # Template per l’head
|   |   ├── navbar.html        # Template per la navbar
```

Flusso dell’applicazione “Gruppo d’Acquisto” - UTENTE

Il diagramma di flusso rappresenta il percorso logico che un utente segue all'interno dell'applicazione web "Gruppo d’Acquisto" per visualizzare e prenotare lotti di prodotti. Di seguito è riportata una descrizione dettagliata delle varie sezioni del diagramma:



1. **Inizio**
    - **Utente**: L'utente accede alla homepage dell'applicazione dove può vedere l'elenco dei lotti disponibili.

2. **Verifica Registrazione e Login**
    - **L'UTENTE è registrato?**
        - **No**: L'utente viene reindirizzato alla pagina di registrazione.
        - **Sì**: L'utente viene verificato se è loggato.
    - **L'UTENTE è loggato?**
        - **No**: L'utente viene reindirizzato alla pagina di login.
        - **Sì**: L'utente può continuare a interagire con l'elenco dei lotti.

3. **Interazione con i Lotti**
    - **Click su un lotto**: L'utente clicca su un lotto per vedere i dettagli.
        - **Esiste già una PRENOTAZIONE sul LOTTO?**
            - **No**: L'utente viene indirizzato alla pagina del lotto.
                - Nella **pagina del LOTTO**, l'utente può:
                    - **Inviare la quantità da prenotare**: Inserire la quantità di prodotto desiderata e inviare la prenotazione.
            - **Sì**: L'utente viene indirizzato alla **pagina della PRENOTAZIONE** dove può:
                - **Modifica quantità esistente**: Modificare la quantità prenotata precedentemente.
                - **Cancella prenotazione**: Cancellare la prenotazione esistente.

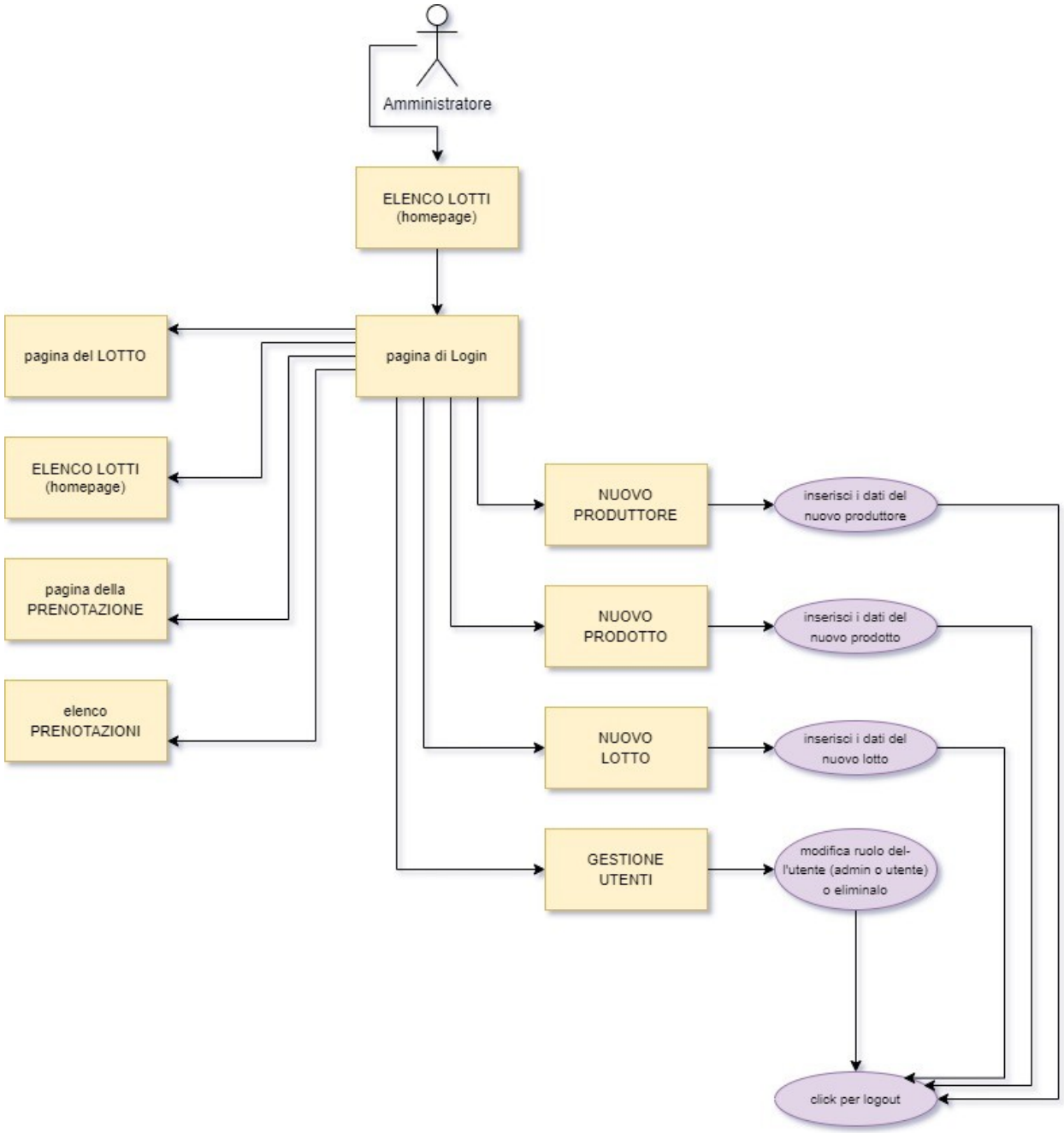
4. **Elenco Prenotazioni**
    - Dopo aver eseguito una delle operazioni sulla prenotazione (modifica o cancellazione), l'utente viene indirizzato all'**elenco PRENOTAZIONI** dove può vedere tutte le prenotazioni effettuate.

5. **Conclusione**
    - L'utente può cliccare su una prenotazione specifica nell'elenco per visualizzare o modificare ulteriormente i dettagli della prenotazione.

La presenza della navbar permette all’utente di effettuare il logout da qualsiasi pagina dell’applicazione.
Questo flusso assicura che l'utente abbia un'esperienza coerente e intuitiva durante l'interazione con l'applicazione "Gruppo d’Acquisto".

Flusso dell’applicazione “Gruppo d’Acquisto” - AMMINISTRATORE

Il diagramma rappresenta il flusso di navigazione e le operazioni principali che un amministratore può effettuare all'interno di un'applicazione di gestione GAS (Gruppo di Acquisto Solidale). Di seguito è riportata una spiegazione dettagliata del diagramma:



1. **Amministratore**:
    - L'utente con ruolo amministrativo ha accesso a tutte le funzionalità dell'applicazione.
    - L'amministratore può accedere all'elenco dei lotti, alla gestione dei produttori, prodotti, lotti e utenti.

2. **Elenco Lotti (Homepage)**:
    - Questa è la pagina principale a cui l'amministratore ha accesso dopo il login.
    - Da qui, l'amministratore può navigare verso altre pagine.

3. **Pagina di Login**:
    - La pagina di accesso per l'amministratore.
    - **L’amministratore deve effettuare il login per accedere alle altre funzionalità dell'applicazione.**

1. **Pagine Lotto, Prenotazione e Prenotazioni**
    - L'amministratore può interagire come un utente normale

4. **Nuovo Produttore**:
    - L'amministratore può inserire le informazioni necessarie per aggiungere un nuovo produttore al sistema.

5. **Nuovo Prodotto**:
    - L'amministratore può aggiungere nuovi prodotti associandoli ai produttori esistenti.

6. **Nuovo Lotto**:
    - L'amministratore inserisce i dettagli del lotto, come la quantità, la data di consegna e il prezzo.

7. **Gestione Utenti**:
    - L'amministratore può modificare i ruoli degli utenti (ad esempio, da utente a amministratore) o eliminarli dal sistema.
    - L’eliminazione è possibile solo se l’utente non ha effettuato prenotazioni.

8. **Logout**:
    - L'amministratore può uscire dall'applicazione in qualsiasi momento.

Funzionalità Specifiche

- **Inserimento Dati**:
    - Ogni pagina dedicata all'inserimento dati (produttore, prodotto, lotto) permette all'amministratore di aggiungere nuovi elementi al sistema.
- **Modifica e Eliminazione Utenti**:
    - Nella pagina di gestione utenti, l'amministratore ha la possibilità di modificare i ruoli degli utenti esistenti o eliminarli dal sistema.

Creazione dell'utente amministratore di default

Nel file `app.py`, viene creato un utente amministratore di default all'avvio dell'applicazione.
La creazione di un amministratore di default all'avvio dell'applicazione è una pratica comune per garantire che ci sia sempre un utente con privilegi amministrativi che possa gestire e configurare l'applicazione. Questo è particolarmente utile nei seguenti scenari:

- **Installazione Iniziale**: Quando l'applicazione viene avviata per la prima volta, l'amministratore di default fornisce un modo per accedere immediatamente alle funzionalità di gestione.
- **Recupero da Errori**: Se tutti gli altri account amministrativi vengono accidentalmente eliminati o disabilitati, l'account amministratore di default funge da back-up.
- **Facilità di Accesso**: Fornisce un accesso immediato e un punto di partenza per configurare ulteriormente il sistema e creare altri account amministrativi o utenti.

Per inserire altri amministratori occorre prima creare un nuovo utente, e successivamente, mediante accesso da amministratore, andare a mofificare il ruolo nella pagina gestione utenti.

Considerazioni di Sicurezza

- **Modifica della Password Predefinita**: È buona norma cambiare la password predefinita (`Ciotola_1`) subito dopo l'installazione per garantire la sicurezza.

Inserimento dei dati esistenti in formato JSON nel Database

Durante lo sviluppo dell'applicazione "Gruppo d’Acquisto", è stato necessario popolare il database con dati iniziali, provenienti da file JSON, contenenti informazioni su produttori, prodotti e lotti. Questo processo di inserimento è fondamentale per garantire che l'applicazione disponga dei dati necessari per funzionare correttamente sin dal primo avvio.

I file JSON utilizzati per popolare il database sono strutturati in modo da rappresentare i dati necessari per ciascuna entità del sistema.
Per inserire questi dati nel database, è stato creato uno script Python che legge i file JSON e popola le rispettive tabelle nel database.
L'inserimento di dati esistenti nel database da file JSON è un passaggio cruciale per garantire che l'applicazione disponga di dati iniziali necessari per il suo funzionamento. Questo approccio permette una facile migrazione e aggiornamento dei dati, facilitando la gestione e la manutenzione del sistema nel lungo termine. Utilizzando JSON, è possibile mantenere una struttura dati chiara e facilmente leggibile, semplificando il processo di caricamento e verifica dei dati.

Protezione delle Password

La protezione delle password è stata implementata utilizzando la libreria `flask-bcrypt`.
Questo assicura che le password degli utenti siano conservate in modo sicuro nel database, rendendo difficile per gli attaccanti ottenere le password anche se il database fosse compromesso.

Hashing delle Password:

- Quando un utente si registra, la sua password viene hashata utilizzando l'algoritmo bcrypt prima di essere salvata nel database. Questo processo include la generazione di un "salt" univoco per ogni password, che viene combinato con la password stessa prima di essere hashato. Il risultato è una stringa hashata che rappresenta la password dell'utente.

Verifica delle Password:

- Durante il login, la password inserita dall'utente viene hashata con lo stesso algoritmo e "salt" e confrontata con l'hash memorizzato nel database. Se i due hash corrispondono, l'utente viene autenticato con successo.

Re-hashing delle Password Esistenti:

- È stato scritto uno script per re-hashare le password esistenti (*rehash_passwords.py*). Questo script legge tutte le password dal database, verifica se sono già hashate correttamente e, se necessario, le re-hasherà utilizzando il nuovo algoritmo.

Pagine HTML

Per generare le pagine HTML è stato usato Jinja, un motore di template per Python comunemente utilizzato con Flask per generare dinamicamente HTML da variabili Python.
Nell'applicazione "Gruppo d'Acquisto", Jinja è utilizzato per renderizzare le pagine HTML con i dati provenienti dal backend.
Ecco alcuni esempi di come è stato utilizzato nei template:

1. **Estendere Template**: Utilizzo Jinja per estendere un layout base `_layout.html`. Questo permette di definire un layout comune per tutte le pagine e riutilizzarlo in vari template.

2. **Blocchi di Template**: Utilizzo blocchi di template (`{% block ... %}{% endblock %}`) per definire sezioni di HTML che possono essere sovrascritte nei template che estendono `_layout.html`.

3. **Loop e Condizioni**: Utilizzo i costrutti di controllo di Jinja come loop (`{% for ... %}`) e condizioni (`{% if ... %}`) per iterare sui dati e mostrare contenuti condizionali.

4. **Iniezione di Variabili**: Utilizzo Jinja per iniettare variabili Python direttamente nell'HTML (`<td>{{ utente.id }}</td>`).

5. **URL di Flask**: Utilizzo la funzione `url_for` di Flask all'interno di Jinja per generare URL dinamici.

Jinja è stato utilizzato estensivamente nell'applicazione "Gruppo d'Acquisto" per generare HTML dinamico, incorporare variabili Python, eseguire loop e condizioni, e gestire l'estensione dei template. Questo approccio permette di separare la logica di business (gestita in Python) dalla presentazione (gestita in HTML), rendendo il codice più organizzato e mantenibile.

Considerazioni Finali

La web application *"Gruppo d’Acquisto"* è stata sviluppata utilizzando un insieme di tecnologie moderne e strumenti di sviluppo avanzati, garantendo un'interfaccia utente intuitiva e una gestione sicura ed efficiente dei dati. L'uso di Flask e delle librerie associate ha permesso di creare un'applicazione robusta e scalabile, in grado di gestire efficacemente le esigenze di un gruppo di acquisto.Re