



MATTIA MONTIS

WEB APP PER LA GESTIONE DI UTENTI

[H T T P S : / / G I T H U B . C O M / M A T T I A M O N T I S](https://github.com/mattiamontis)



MATTIA MONTIS

INTRODUZIONE

Il progetto consiste nello sviluppo di una web application che consente la gestione di utenti, con particolare attenzione alla sicurezza delle credenziali. Questo progetto nasce dall'esigenza di ampliare la conoscenza personale per poter lavorare come programmatore backend e all'unione con un'altra passione, la cyber sicurezza, dimostrando competenze nello sviluppo di applicazioni web sicure utilizzando Python e il framework Flask.

OBIETTIVI

L'obiettivo principale è creare un'applicazione che permetta agli utenti di registrarsi, effettuare il login e visualizzare un elenco degli utenti registrati.

Particolare attenzione è stata posta alla gestione delle password, che vengono salvate in formato hash utilizzando un algoritmo sicuro come SHA-256 tramite la libreria Werkzeug. Implementando anche un piccolo database dove memorizzare gli account degli utenti.

Register

← → ⌂ Archivio C:/Users/monti/OneDrive/Desktop/my_project/templates/register

Registrati

Username:

Password:

Conferma Password:

[Registrati](#)

Login

← → ⌂ Archivio C:/Users/monti/OneDrive/Desktop/my_project/templates/login

Login

Username

Password

[Login](#)



INDICE

1-Struttura del progetto

2-Tecnologie e Librerie Utilizzate

3-Sviluppo e Problematiche

4-Risultati e Considerazioni Finali



STRUTTURA DEL PROGETTO



La struttura del progetto è stata organizzata per mantenere chiarezza e modularità:

```
my_project/
├── app.py
├── templates/
│   ├── login.html
│   ├── register.html
│   └── view_users.html
└── static/
```

- **app.py**: Contiene il codice Python per la gestione delle rotte e la logica dell'applicazione.
- **templates/**: Cartella dedicata ai file HTML, creati con l'ausilio di un'intelligenza artificiale per ottimizzare il tempo di sviluppo.
- **static/**: Riservata per eventuali file statici (es. immagini, CSS, JavaScript).
- **venv/**: Ambiente virtuale per isolare le dipendenze del progetto.



TECNOLOGIE E LIBRERIE UTILIZZATE



- Flask: Framework leggero e versatile per lo sviluppo di applicazioni web.
- SQLite: Database relazionale utilizzato per memorizzare i dati degli utenti. È stato scelto per la sua semplicità e facilità di configurazione in progetti di piccole dimensioni.
- Werkzeug: Libreria Python utilizzata per generare e verificare gli hash delle password.
- Jinja2: Utilizzato per il rendering dinamico delle pagine HTML.



SVILUPPO E PROBLEMATICHE

Durante lo sviluppo del progetto, sono stati affrontati e risolti diversi problemi:

1. Sicurezza delle password:
2. Inizialmente, il progetto memorizzava le password in chiaro, ma per garantire la sicurezza degli utenti, è stato implementato l'algoritmo di hashing SHA-256 tramite la funzione `generate_password_hash` di Werkzeug. Questo garantisce che le password non siano accessibili neanche in caso di violazione del database.
3. Gestione degli utenti duplicati:
4. Un problema ricorrente era la possibilità di registrare più utenti con lo stesso username. È stato introdotto un controllo lato backend per verificare l'esistenza del nome utente prima di procedere con la registrazione, migliorando l'esperienza utente e l'affidabilità dell'applicazione.
5. Feedback all'utente:
6. Durante i test iniziali, la mancanza di messaggi esplicativi in caso di errori (es. password non corrispondenti) rendeva l'applicazione poco intuitiva. Sono stati aggiunti messaggi di errore personalizzati per guidare meglio l'utente.
7. Visualizzazione degli utenti:
8. La pagina `view_users.html` è stata creata per mostrare un elenco di tutti gli utenti registrati. Questo ha richiesto l'implementazione di un'interfaccia chiara e semplice, con una tabella HTML dinamica generata tramite Jinja2. È stato necessario adattare il database per garantire la visualizzazione sicura delle informazioni, come il salvataggio delle password in formato hash.
9. Configurazione dell'ambiente virtuale:
10. Configurare un ambiente virtuale (`venv`) su Windows 11 ha richiesto un'attenzione particolare, specialmente per evitare conflitti tra versioni di Python e librerie. Una volta configurato correttamente, il `venv` ha permesso di isolare le dipendenze del progetto e di garantire la replicabilità dell'ambiente.

The screenshot displays three windows illustrating the application's development environment and user interface.

- Registration Page:** A browser window titled "Registrati" shows fields for "Username" (mett), "Password" (.....), and "Conferma Password" (.....). Below the form is a "Registrati" button.
- User List Page:** A browser window titled "Users List" shows a table with columns "Username" and "Password (Hash)". It lists three users: met (scrypt:32768:8:1\$fmRX0JdHlwDgaec\$197139d9de34f3f89f12f9733ecfcee24538230c690cc002dbbee2c0741567df49a47383810a), met (scrypt:32768:8:1\$LFSDZFvMSbBzIDFC\$1da1484a92acc8eb80c093397ab70aa0c744efcb6974b55e65963087093963fa473ca15c0), and mett (scrypt:32768:8:1\$YWu4r9v4Zq01e1rf\$b5c6a8ae1b4839ba3eebf3bb04ecc43b501a31dec678d1059f60e215e1d8a254b603aa0f2a5c). The URL in the address bar is 127.0.0.1:5000/view_users.
- Registration Error:** A browser window titled "register" shows an error message: "Il nome utente esiste già. Scegli un nome utente diverso." The URL is 127.0.0.1:5000/register.
- Windows PowerShell:** A terminal window titled "Windows PowerShell" shows the command `python app.py` being run. The output indicates the application is serving on port 5000: "Serving Flask app 'app'". It also includes a warning about using this as a production server: "WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead." and "Running on http://127.0.0.1:5000".



Risultati e Considerazioni Finali

L'APPLICAZIONE È STATA SVILUPPATA CON SUCCESSO, RISPETTANDO GLI STANDARD DI SICUREZZA PER LA GESTIONE DELLE CREDENZIALI DEGLI UTENTI. SEBBENE IL PROGETTO SIA ATTUALMENTE LIMITATO ALLE FUNZIONALITÀ DI BASE, ESSO RAPPRESENTA UNA SOLIDA BASE PER ULTERIORI SVILUPPI, COME:

- IMPLEMENTAZIONE DI UN SISTEMA DI AUTENTICAZIONE A DUE FATTORI (2FA).
- PASSAGGIO A FRAMEWORK PIÙ AVANZATI COME DJANGO PER UNA MAGGIORE SCALABILITÀ.
- INTRODUZIONE DI TEST AUTOMATICI PER MIGLIORARE L'AFFIDABILITÀ DEL CODICE.

CONCLUSIONI

QUESTO PROGETTO NON SOLO DIMOSTRA LE COMPETENZE TECNICHE NELLA PROGRAMMAZIONE BACKEND E NELLA GESTIONE DELLA SICUREZZA, MA RIFLETTE ANCHE LA CAPACITÀ DI AFFRONTARE E RISOLVERE PROBLEMI PRATICI DURANTE LO SVILUPPO. È UNA TESTIMONIANZA CONCRETA DEL MIO IMPEGNO A CREARE APPLICAZIONI ROBUSTE E SICURE, QUALITÀ ESSENZIALI PER UN PROGRAMMATORE BACKEND ORIENTATO ALLA CYBERSECURITY.



MATTIA MONTIS

THANK YOU

[HTTPS://GITHUB.COM/MATTIAMONTIS](https://github.com/mattiamontis)