

Piattaforma di Gestione di Tutoraggio

Alessandro Catenacci

July 8, 2024

Contents

| | | |
|-----------|--|----------|
| 1 | Introduzione | 2 |
| 2 | Funzionalità Richieste | 2 |
| 3 | Requisiti Non Funzionali | 2 |
| 3.1 | Autorizzazioni | 3 |
| 4 | Descrizione del Progetto dell'Applicazione | 4 |
| 4.1 | Diagramma delle classi UML | 4 |
| 4.2 | Diagramma E/R | 4 |
| 5 | Tecnologie Usate e Motivazione | 6 |
| 5.1 | Database | 6 |
| 5.1.1 | Esempi di Query per la Ricerca dei Tutor | 6 |
| 6 | Organizzazione Logica dell'Applicazione | 6 |
| 6.1 | Suddivisione in Moduli | 7 |
| 7 | Frontend | 7 |
| 8 | Backend | 8 |
| 9 | Test Eseguiti | 8 |
| 10 | Risultati | 8 |

List of Tables

| | | |
|---|---|---|
| 1 | Autorizzazioni degli utenti per funzionalità principali | 3 |
|---|---|---|

List of Figures

| | | |
|---|---|----|
| 1 | Diagramma dei casi d'uso per le prenotazioni | 4 |
| 2 | Diagramma Entità-Relazione del Database | 5 |
| 3 | Homepage | 9 |
| 4 | Pagina di registrazione | 9 |
| 5 | Dashboard dello studente con elenco dei tutor disponibili | 10 |
| 6 | Profilo del tutor con recensioni e valutazioni | 10 |
| 7 | Sistema di prenotazione delle lezioni | 11 |

1 Introduzione

Il progetto sviluppato rappresenta una piattaforma di gestione di tutoraggio, concepita per facilitare l'organizzazione di lezioni private in presenza tra studenti e tutor. Questo sistema integra una serie di funzionalità essenziali per garantire un'esperienza utente ottimale e una gestione efficiente delle lezioni. Di seguito, forniamo una descrizione dettagliata delle funzionalità e delle caratteristiche implementate, accompagnata dai dettagli tecnici necessari per una comprensione completa e approfondita del sistema.

La piattaforma non solo si propone di semplificare l'interazione tra studenti e tutor, ma anche di introdurre una serie di strumenti che migliorano la qualità dell'insegnamento e dell'apprendimento. L'obiettivo principale è quello di creare un ambiente dinamico e interattivo, in cui gli utenti possano facilmente trovare e prenotare lezioni e ricevere feedback costruttivo.

2 Funzionalità Richieste

Questa sezione descrive in dettaglio le funzionalità richieste per l'applicazione, fornendo un'analisi approfondita di ciascun componente chiave:

- **Registrazione e gestione del profilo per studenti e tutor:** Ogni utente può creare un profilo personalizzato scegliendo il proprio ruolo (studente o tutor). I profili includono informazioni personali, qualifiche, materie di insegnamento (per i tutor) e preferenze di apprendimento (per gli studenti). Gli utenti possono caricare una foto del profilo e aggiornare le loro informazioni in qualsiasi momento.
- **Sistema di prenotazione delle lezioni con gestione delle disponibilità:** Gli studenti possono cercare tutor disponibili in base alle materie insegnate e alla loro disponibilità. Una volta trovato il tutor adatto, possono prenotare lezioni private inserendo la data e l'orario desiderati. Il sistema permette anche la modifica e la cancellazione delle prenotazioni, mantenendo traccia dello storico delle lezioni per una gestione più efficiente.
- **Recensioni e valutazioni dei tutor da parte degli studenti:** Dopo ogni lezione, gli studenti possono lasciare recensioni e valutazioni sui tutor basate sulla loro esperienza. Questo feedback è visibile agli altri utenti, e i tutor possono visualizzare le recensioni ricevute e rispondere per migliorare il proprio servizio.
- **Gestione delle liste d'attesa per lezioni già prenotate:** Se un tutor è già prenotato per un determinato slot, gli studenti possono inserirsi in una lista d'attesa. Il sistema notificherà automaticamente gli studenti quando una lezione diventa disponibile a seguito di una cancellazione.
- **Notifiche via email per conferme, modifiche e cancellazioni di prenotazioni:** Il sistema invia automaticamente notifiche via email per confermare le prenotazioni, notificare modifiche o annullamenti, e informare gli studenti sulla disponibilità da lista d'attesa. In questo modo, gli utenti sono sempre aggiornati sugli eventi che li riguardano.
- **Sistema di ricerca avanzata per trovare tutor in base a materie, disponibilità e valutazioni:** La piattaforma include un sistema di ricerca avanzata che permette agli studenti di filtrare i tutor in base alle materie insegnate, alla disponibilità e alle valutazioni ricevute. Questo garantisce un processo di selezione rapido ed efficace, aiutando gli studenti a trovare il tutor più adatto alle loro esigenze.

3 Requisiti Non Funzionali

Questa sezione descrive i requisiti non funzionali derivati dalle funzionalità richieste, garantendo che il sistema funzioni efficacemente e fornisca un'esperienza utente ottimale.

- **Performance e Scalabilità:**
 - Il sistema deve essere in grado di gestire un numero abbastanza grande di utenti.
 - Le operazioni di ricerca e prenotazione devono essere eseguite in tempi rapidi, garantendo una risposta immediata e informativa di successo/fallimento agli utenti.

- **Sicurezza:**

- Il sistema deve garantire l'autenticazione e l'autorizzazione adeguate per prevenire accessi non autorizzati.

- **Usabilità:**

- L'interfaccia utente deve essere intuitiva e facile da usare, garantendo che anche gli utenti con scarse competenze tecnologiche possano navigare e utilizzare il sistema senza difficoltà.

- **Manutenibilità ed Estensibilità:**

- Il codice del sistema deve essere scritto in modo modulare per facilitare la manutenzione e l'aggiornamento.
- Il sistema deve essere progettato per consentire l'aggiunta di nuove funzionalità senza necessità di riscrivere interi moduli.

3.1 Autorizzazioni

Il sistema utilizza il Django Admin e una dashboard di default per la gestione delle autorizzazioni e delle operazioni amministrative. Il Django Admin consente agli utenti con privilegi di superuser di eseguire operazioni amministrative come la gestione degli utenti, la supervisione delle prenotazioni e l'accesso a tutte le funzionalità del backend.

Di seguito è descritta la differenza tra i permessi concessi agli utenti anonimi, tutor e studenti. È inoltre fornita una tabella riepilogativa delle autorizzazioni per ciascun tipo di utente.

| Funzionalità | Utente Anonimo | Studente | Tutor |
|---------------------------------------|----------------|----------|---------------------|
| Registrazione e gestione profilo | No | Sì | Sì |
| Visualizzazione profili tutor | Sì | Sì | Sì |
| Prenotazione lezioni | No | Sì | No |
| Modifica e cancellazione prenotazioni | No | Sì | No |
| Visualizzazione storico lezioni | No | Sì | Sì |
| Recensioni e valutazioni tutor | No | Sì | No |
| Accesso a lista d'attesa | No | Sì | No |
| Notifiche via email | No | Sì | Sì |
| Sistema di ricerca avanzata | No | Sì | Sì |
| Visualizzazione recensioni tutor | Sì | Sì | Sì |
| Risposta alle recensioni ricevute | No | No | Sì |
| Accesso a Django Admin | No | No | No (Solo Superuser) |

Table 1: Autorizzazioni degli utenti per funzionalità principali

Utenti Anonimi: Gli utenti non registrati hanno accesso limitato alla piattaforma. Possono visualizzare la homepage e le informazioni generali, ma non possono accedere alle funzionalità principali come la prenotazione delle lezioni o la gestione dei profili.

Studenti: Gli studenti registrati possono creare e gestire il proprio profilo, cercare e prenotare lezioni con i tutor, lasciare recensioni e valutazioni, accedere alla lista d'attesa e ricevere notifiche via email per conferme, modifiche e cancellazioni di prenotazioni.

Tutor: I tutor registrati possono creare e gestire il proprio profilo, visualizzare e rispondere alle recensioni ricevute, gestire la propria disponibilità per le lezioni e ricevere notifiche via email. Tuttavia, non possono prenotare lezioni come gli studenti.

Superuser (Django Admin): I superuser hanno accesso completo alla dashboard di Django Admin, consentendo loro di gestire tutti gli aspetti della piattaforma, inclusa la supervisione degli utenti e delle prenotazioni, la gestione delle autorizzazioni e la visualizzazione delle statistiche di utilizzo.

4 Descrizione del Progetto dell'Applicazione

4.1 Diagramma delle classi UML

Il diagramma delle classi UML rappresenta la struttura del sistema e le relazioni tra le diverse classi. Questo diagramma fornisce una panoramica chiara delle entità principali e delle loro interazioni, facilitando la comprensione della logica di business e delle funzionalità implementate.

In un esempio di diagramma delle classi UML, possiamo vedere le classi principali per quanto riguarda la fase di prenotazione, con annessa modifica e cancellazione. Queste classi includono i form e le viste per la gestione delle prenotazioni, per semplicità non sono state incluse le classi che provengono dai modelli del database, già rappresentate in altro modo nel diagramma E/R.

Usando la Django CBV (Class Based Views) per la gestione delle richieste HTTP, le classi delle viste sono state progettate per gestire le operazioni CRUD (Create, Read, Update, Delete) sulle prenotazioni. I form sono utilizzati per la validazione dei dati inseriti dagli utenti e per la creazione di nuove prenotazioni o la modifica di quelle esistenti.

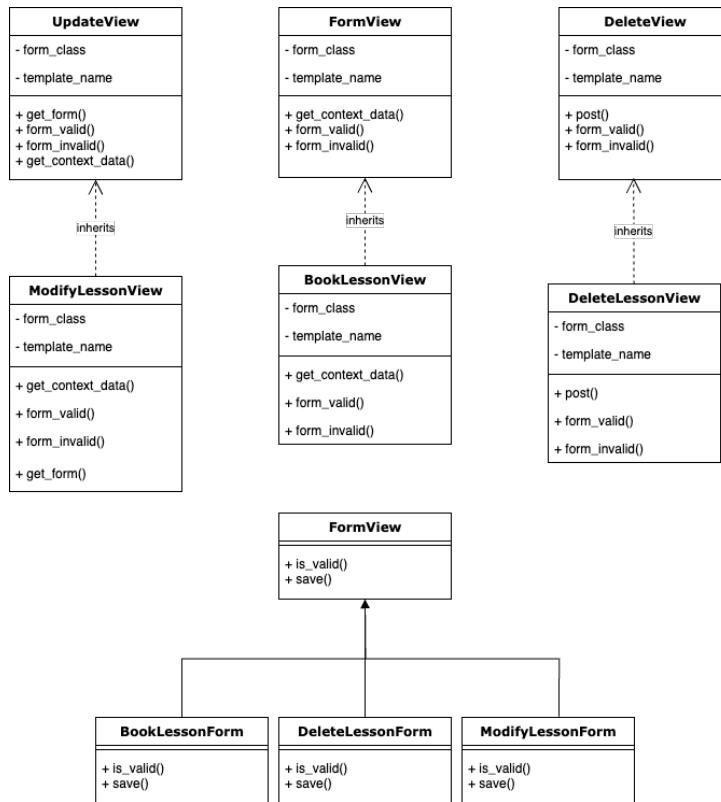


Figure 1: Diagramma dei casi d'uso per le prenotazioni

4.2 Diagramma E/R

Il diagramma Entità-Relazione (E/R) rappresenta la struttura del database dell'applicazione, mostrando le entità principali e le relazioni tra di esse. Questo diagramma fornisce una panoramica chiara della struttura dei dati e delle relazioni tra le diverse entità. Le entità principali riguardano i tipi di utenti, gli schemi per le lezioni e le prenotazioni e le recensioni.

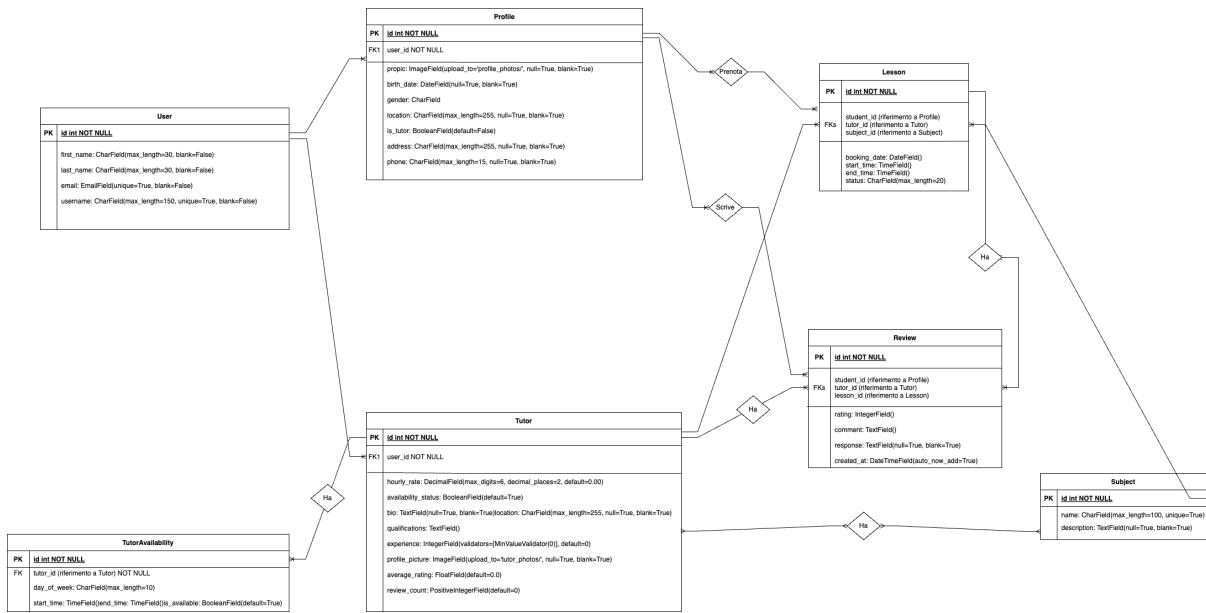


Figure 2: Diagramma Entità-Relazione del Database

5 Tecnologie Usate e Motivazione

5.1 Database

La scelta di SQLite come Database Management System (DBMS) per questo progetto è stata guidata da diversi fattori:

- **Semplicità e Facilità d'Uso:** SQLite è un database relazionale leggero e self-contained, il che lo rende estremamente facile da configurare e utilizzare. Non richiede un server separato, eliminando la necessità di gestire complessi setup di database server.
- **Portabilità:** Essendo un singolo file, un database SQLite può essere facilmente trasferito tra diverse piattaforme, rendendolo ideale per lo sviluppo locale e per applicazioni più piccole che non necessitano di un database distribuito.
- **Prestazioni:** Per applicazioni con un carico di lavoro moderato, SQLite offre prestazioni eccellenti. È ottimizzato per letture veloci e può gestire un buon numero di connessioni simultanee senza problemi.
- **Supporto Integrato:** SQLite è supportato nativamente da Django, il framework utilizzato per questo progetto, il che semplifica ulteriormente l'integrazione e la gestione del database.

5.1.1 Esempi di Query per la Ricerca dei Tutor

Di seguito sono riportati alcuni esempi di query che vengono eseguite nel contesto delle viste di Django per la ricerca dei tutor. Questi esempi mostrano come vengono filtrati i dati basandosi sui criteri di ricerca forniti dall'utente.

```
1 # Filtra per posizione
2 if location:
3     queryset = queryset.filter(user__profile__location__icontains=location)
4
5 # Filtra per valutazione minima
6 if min_rating is not None:
7     queryset = queryset.filter(average_rating__gte=min_rating)
8
9 # Filtra per esperienza
10 if experience is not None:
11     queryset = queryset.filter(experience__gte=experience)
12
13 # Filtra per disponibilità
14 if available_on_day or (available_from and available_to):
15     availability_filter = Q(availabilities__is_available=True)
16     if available_on_day and available_on_day != '_____':
17         availability_filter &= Q(availabilities__day_of_week=available_on_day.
18         capitalize())
19     if available_from and available_to:
20         availability_filter &= Q(availabilities__start_time__lte=available_from,
21                                 availabilities__end_time__gte=available_to)
22     queryset = queryset.filter(availability_filter).distinct()
23
24 return queryset
```

6 Organizzazione Logica dell'Applicazione

Il sito possiede un'unica applicazione centrale chiamata core, in core si sviluppano tutte le viste che sono organizzate in package per facilitare la manutenzione e la comprensione del codice. Stessa cosa per i templates e i forms, i file hanno nomi comuni e riconoscibili per facilitare la navigazione e la comprensione del codice. Per esempio un file che si occupa di gestire la prenotazione di una lezione si chiamerà `register.py`, e avrà `forms/register.py`, `templates/register.html` e `views/register.py`.

6.1 Suddivisione in Moduli

Concettualmente, l'applicazione è suddivisa in moduli distinti per gestire le diverse funzionalità e i flussi di lavoro. Questa suddivisione modulare facilita la manutenzione e l'estensione del sistema, consentendo ai team di sviluppo di lavorare in modo indipendente su parti specifiche dell'applicazione.

Abbiamo 7 moduli principali, main che riguarda sostanzialmente la homepage, auth riguarda le procedure di login e logout, registration le procedure di registrazione, search e contact rispettivamente la funzione di ricerca avanzata dei tutor e la gestione delle prenotazioni, profile si occupa di tutto ciò che riguarda la gestione del profilo, review riguarda la gestione delle recensioni.

7 Frontend

Le tecnologie utilizzate per lo sviluppo del frontend comprendono i classici JavaScript, HTML e CSS, integrati dai componenti offerti da Bootstrap 5.3 e dalla libreria di utilità jQuery, ampiamente utilizzata per funzioni come la registrazione e la modifica del profilo. JavaScript è utilizzato in quasi tutti i template per rendere l'interfaccia più user-friendly e per gestire le view che usano AJAX e jQuery.

Per esempio, in `templates/edit_profile.html`, JavaScript è ampiamente utilizzato per:

1. Gestione dei File Input (Caricamento dell'Immagine del Profilo)

- Quando l'utente clicca sul pulsante "Change Picture", viene simulato un click sull'input file nascosto `propic-input` tramite l'attributo `onclick="document.getElementById('propic-input').click();"`.
- L'evento `change` sull'input file (`#propic-input`) attiva la funzione `previewImage(event)`, che utilizza un `FileReader` per caricare e mostrare l'immagine selezionata in anteprima.

2. Reset dell'Immagine del Profilo

- Il pulsante con `onclick="resetProfilePicture()"` chiama la funzione `resetProfilePicture()`, che ripristina l'immagine del profilo a quella predefinita e aggiorna il valore dell'input nascosto `delete-propic` per indicare che l'immagine è stata rimossa.

3. Aggiunta e Rimozione delle Disponibilità

- La funzione `addAvailability()` crea dinamicamente nuovi elementi del modulo per le disponibilità dell'utente (giorno della settimana, ora di inizio e fine, e disponibilità) e li aggiunge alla pagina. Utilizza JavaScript per gestire il conteggio dei moduli (`availabilityCounter`) e aggiornare il numero totale di moduli nel campo nascosto `id_form-TOTAL_FORMS`.
- La funzione `removeAvailability(index)` gestisce la rimozione delle disponibilità nascondendo l'elemento corrispondente e marcandolo come eliminato, aggiornando il campo `id_form-${index}-DELETE`.

4. AJAX per la Gestione dei Moduli

- Quando il modulo principale (`#main-form`) viene inviato, l'evento `submit` viene intercettato da JavaScript per prevenire l'invio predefinito. Invece, viene utilizzato AJAX (`$.ajax`) per inviare i dati del modulo al server senza ricaricare la pagina. Se la risposta ha successo, viene mostrata una notifica con `showNotification(response.message)`. Se ci sono errori, vengono gestiti da `handleFormErrors(response.errors)`.
- Similmente, il modulo per il cambio della password (`#password-form`) viene gestito tramite AJAX per aggiornare la password senza ricaricare la pagina.

5. Notifiche

- La funzione `showNotification(message)` mostra un messaggio di notifica temporaneo nella pagina, aggiungendo e rimuovendo una classe CSS per far apparire e scomparire il messaggio dopo 3 secondi.

6. Gestione degli Errori del Modulo

- La funzione `handleFormErrors(errors)` raccoglie e mostra eventuali errori di validazione del modulo ricevuti dal server.

8 Backend

Per il backend è stato utilizzato principalmente django, con alcune librerie di supporto come `django.core.cache` per la gestione delle code. Un'altra menzione particolare va al modulo per l'invio di notifiche via email (`django.core.mail`) implementate nei casi di prenotazione/modifica e cancellazione di lezioni e anche in caso di disponibilità di lezioni in lista d'attesa.

9 Test Eseguiti

Per garantire la qualità del software, sono stati eseguiti vari tipi di test:

- **Unit Testing:** Test delle singole unità di codice per verificarne il corretto funzionamento. Questi test sono fondamentali per assicurare che ogni componente dell'applicazione funzioni correttamente in isolamento, riducendo il rischio di bug.

Le funzionalità testate includono:

- Popolazione del database.
- Registrazione e login degli utenti
- Creazione e modifica dei profili
- Visualizzazione dei profili
- Template di registrazione
- Cambio password
- Prenotazione, modifica e cancellazione delle lezioni
- Notifiche via email per conferme e cancellazioni di prenotazioni

10 Risultati

Gli screenshot più interessanti dell'applicazione includono:

- **Homepage:** Schermata principale del sito che mostra le schede dei primi tre tutor selezionati per ordine di rating positivi. In alto a sinistra della navbar c'è il tasto "home" mentre al centro c'è il tasto per raggiungere la ricerca avanzata mentre sulla destra c'è la gestione del profilo e il register.
- **Pagina di registrazione:** Una schermata user-friendly che consente agli utenti di registrarsi alla piattaforma.
- **Dashboard dello studente con lezioni prenotate:** Un'interfaccia intuitiva che permette agli studenti di visualizzare e selezionare le lezioni, con opzioni di scrivere recensioni per quelle complete e modifica per quelle ancora da svolgersi.
- **Profilo del tutor con recensioni e valutazioni:** Una pagina dettagliata che mostra le qualifiche del tutor, le materie insegnate e le recensioni degli studenti, aiutando gli utenti a fare scelte informate.
- **Sistema di prenotazione delle lezioni:** Una funzione integrata che consente agli studenti di prenotare lezioni con facilità, visualizzando la disponibilità in tempo reale.

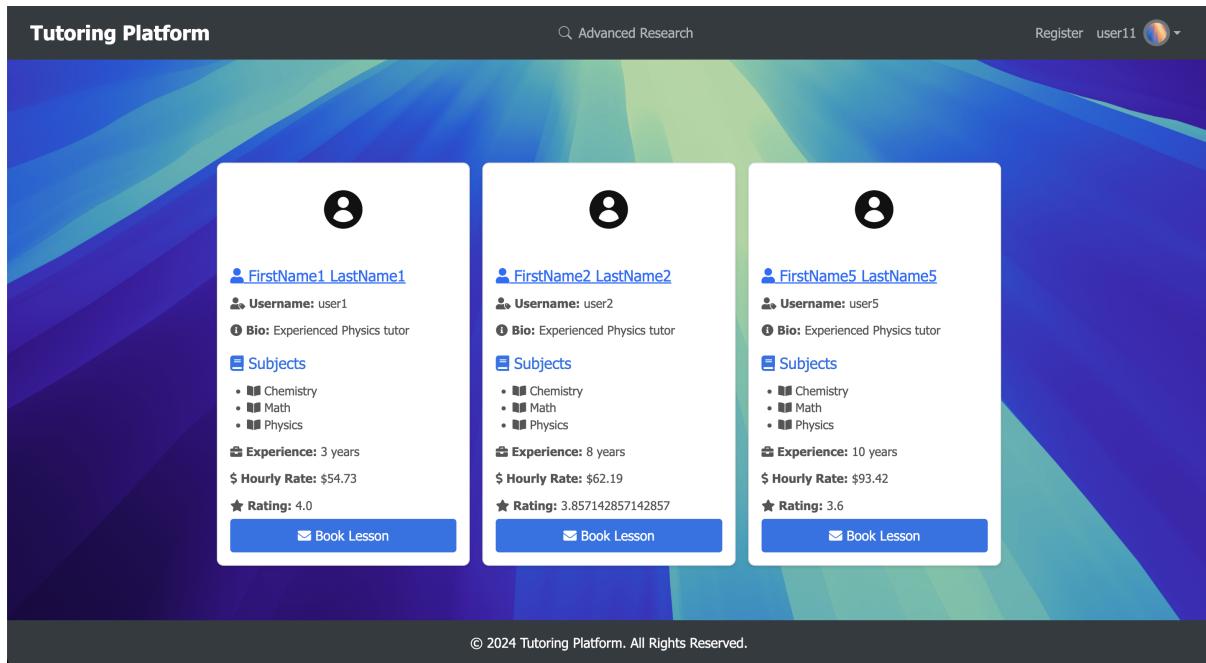


Figure 3: Homepage

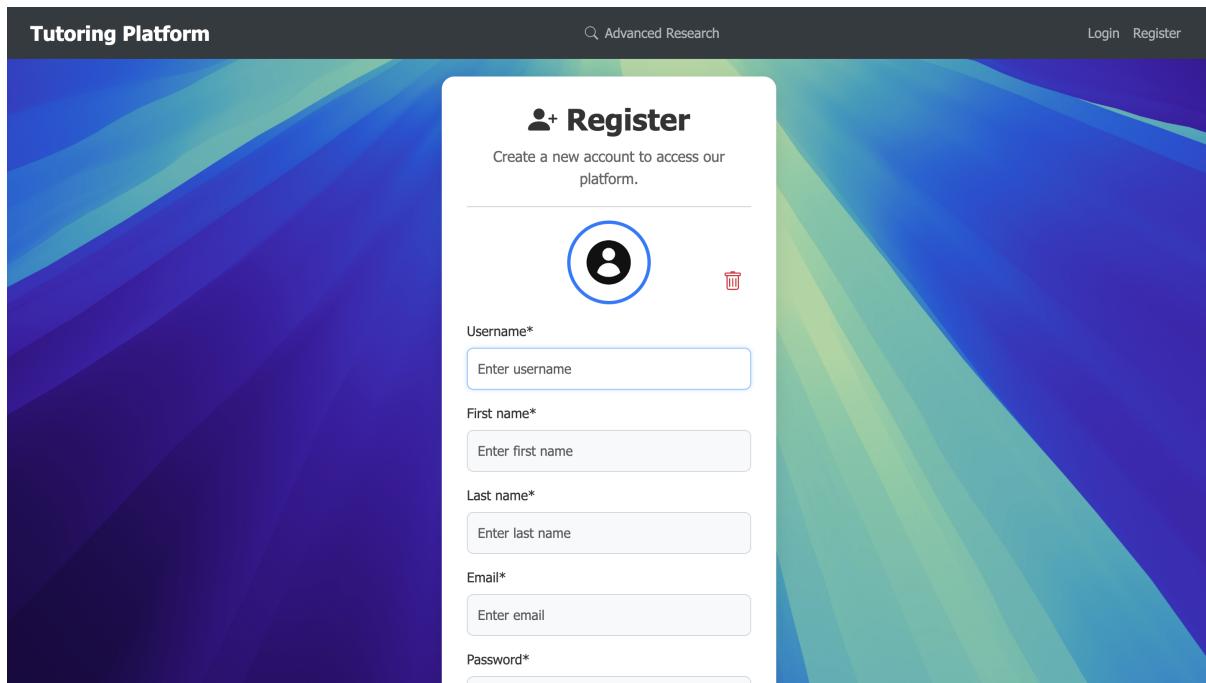


Figure 4: Pagina di registrazione

The screenshot shows the 'Profile Settings' section of the student dashboard. At the top, there are navigation links for 'Profile', 'Account', 'Lessons' (which is currently selected), and 'Settings'. Below this is a table titled 'Lessons' with two rows of data. The first row represents a lesson booked for July 29, 2024, from 11:05 a.m. to 12:50 p.m., taught by FirstName5 LastName5 in Physics, with status 'Booked'. It includes 'Edit' and 'Delete' buttons. The second row represents a completed lesson on July 6, 2024, from 9 a.m. to 5 p.m., taught by FirstName5 LastName5 in Chemistry, with status 'Completed'. It includes a 'Leave a Review' button. At the bottom of the page is a blue button labeled 'Edit Profile'.

Figure 5: Dashboard dello studente con elenco dei tutor disponibili

The screenshot shows the 'Public Profile' section of the tutor's dashboard. At the top, there are navigation links for 'Profile', 'Tutor' (selected), 'Reviews' (highlighted in blue), and 'Lessons'. Below this is a section titled 'Total Review Score' showing 3.6 stars. Underneath is a heading 'Reviews' followed by a list of five reviews from different users. Each review includes the user's name, their rating (e.g., 4 stars, 5 stars, 1 star, 3 stars, 5 stars), a comment like 'Great lesson!', and the subject of the lesson (e.g., Chemistry, Physics).

Figure 6: Profilo del tutor con recensioni e valutazioni

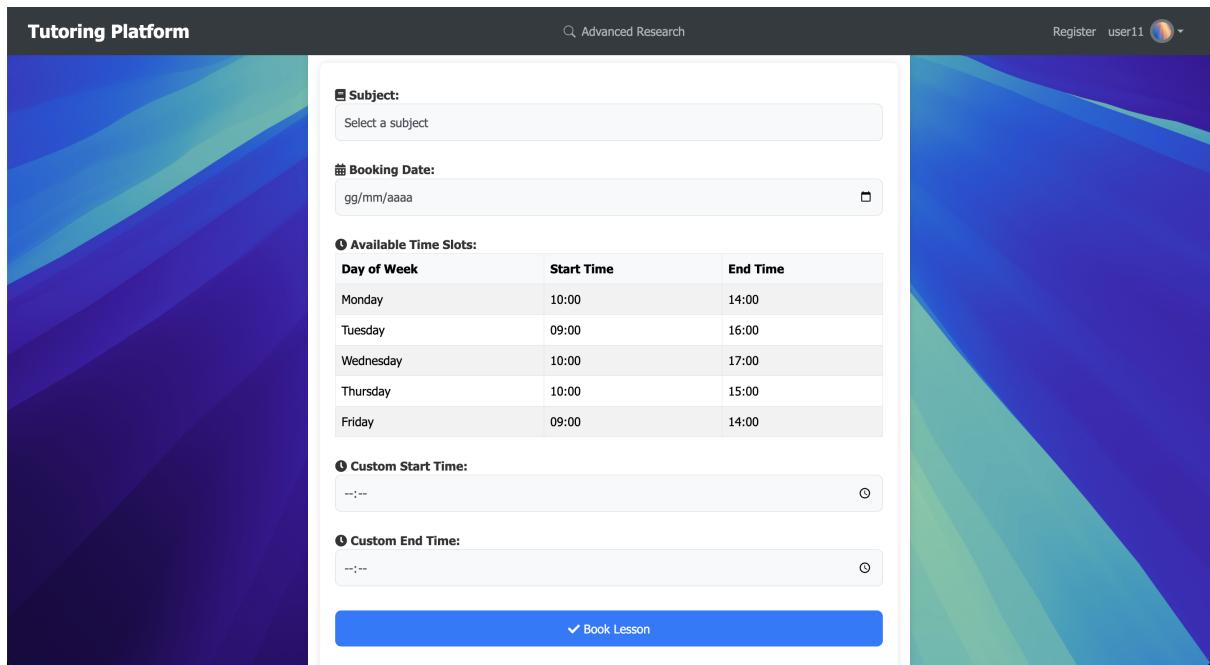


Figure 7: Sistema di prenotazione delle lezioni