



Università  
Ca' Foscari  
Venezia



# Informazioni progetto **Basi di Dati**

**GESTIONE ESAMI UNIVERSITARI**

# SOMMARIO:

SOMMARIO.....	1
1. Introduzione.....	2
2. Funzionalità Principali.....	3
3. Progettazione Concettuale.....	5
4. Progettazione Logica.....	7
5. Scelte Progettuali.....	8
6. Descrizione pagine applicazione.....	9
7. Risorse.....	14

*Disclaimer: questa non è la documentazione inviata per questo esame ma soltanto una parte, Questo progetto è condiviso pubblicamente su GitHub e potrebbe contenere errori, difetti o mancanze.*

# 1. Introduzione

In questo progetto abbiamo sviluppato una web application dedicata alla gestione degli esami universitari.

Come prima cosa abbiamo creato uno schema concettuale che successivamente abbiamo trasformato nello schema logico per la base di dati su cui si basa la nostra applicazione.

In merito al Database, ci siamo appoggiate a PostgreSQL.

PostgreSQL è un database a oggetti in cui l'informazione è rappresentata in forma di oggetti, noto anche con l'abbreviazione "Postgres". Si distingue nel settore delle basi di dati per le sue caratteristiche uniche nel suo genere, che lo pongono per alcuni aspetti all'avanguardia nel settore delle basi di dati.

Per quanto riguarda il back-end della nostra applicazione, abbiamo utilizzato PostgreSQL, Flask e SQLAlchemy per gestire in modo ottimale l'intera parte server della nostra applicazione.

Per quanto riguarda il front-end della nostra applicazione, abbiamo fatto affidamento su HTML, CSS e JavaScript. Abbiamo adottato un design minimale ma accattivante, con l'obiettivo di offrire agli utenti la massima semplicità nell'utilizzo della nostra piattaforma.



In questa sezione forniremo una breve panoramica delle funzionalità generali della nostra applicazione. Successivamente, approfondiremo dettagliatamente ciascuna di esse nel capitolo 2.

Gli utenti della nostra applicazione, si suddividono principalmente in due categorie: i professori e gli studenti.

I professori avranno la possibilità di eseguire diverse azioni, tutte finalizzate alla gestione degli esami universitari, ad esempio potranno creare degli esami con le prove che li compongono, registrare le valutazioni delle prove svolte dagli studenti e altro ancora.

Gli studenti avranno accesso alla gestione della loro carriera universitaria, comprese funzioni quali l'iscrizione agli esami, la visualizzazione e l'accettazione delle valutazioni relative a tali esami, e altro ancora.

## 2. Funzionalità Principali

Di seguito verranno descritte le funzionalità principali che abbiamo implementato nella nostra applicazione.

- **Registrazione e login dell'utente:**

- Un utente non registrato non può accedere all'applicazione, deve quindi creare un nuovo account registrandosi tramite apposito form.
- Una volta creato l'account in cui l'utente specificherà il proprio ruolo (professore o studente) avrà accesso alla propria area riservata che sarà diversa in base al ruolo inserito in fase di registrazione e avrà diverse funzionalità.

- **Creazione di Esami e prove:**

- I docenti possono creare nuovi esami.
- Un esame per esistere dev'essere composto da almeno una prova, che sarà creata subito dopo l'esame.

- **Configurazione delle Prove:**

- I docenti possono configurare le prove, definendo i criteri per la valutazione finale.
- Ogni prova può avere una diversa tipologia (scritto, orale, progetto...) e un diverso tipo di valutazione peso del voto, bonus al voto).

- **Assegnazione Data:**

- Il professore assegna a tutte le prove che ha creato la data che preferisce in cui ogni prova venga sostenuta (ad eccezione di weekend o date già trascorse).

- **Visualizzazione e modifica:**

- Il professore può visualizzare tutte le prove da lui e dagli altri professori create, con i loro dettagli e la data a loro assegnata, potrà inoltre apportare modifiche alle prove da lui create in caso di errore o altro.

- **Prove e Superamento:**

- Ogni prova ha una data di superamento e una data di scadenza (la data di scadenza è calcolata in base alla data di superamento).
- Se una prova viene superata in un appello successivo, invalida la prova precedente.

- **Iscrizione Prove:**

- Ogni studente può iscriversi alle prove create dai professori in uno degli appelli disponibili.

- **Visualizzazione e cancellazione:**

- Lo studente può visualizzare tutte le prove a cui si è iscritto e può decidere di cancellare la sua iscrizione fino a 3 giorni prima.

- **Registrazione dei Voti parziali:**

- Quando è passata la data in cui una prova è stata sostenuta, il professore può inserire le valutazioni di tutti gli studenti iscritti.

- **Registrazione dei Voti totali:**

- Quando tutte le prove sono superate e ancora valide, è possibile registrare il voto dell'esame.

- **Visualizzazione dello Stato:**

- Il sistema fornisce la possibilità di visualizzare lo stato degli studenti, incluse le prove valide superate e lo storico delle prove sostenute.

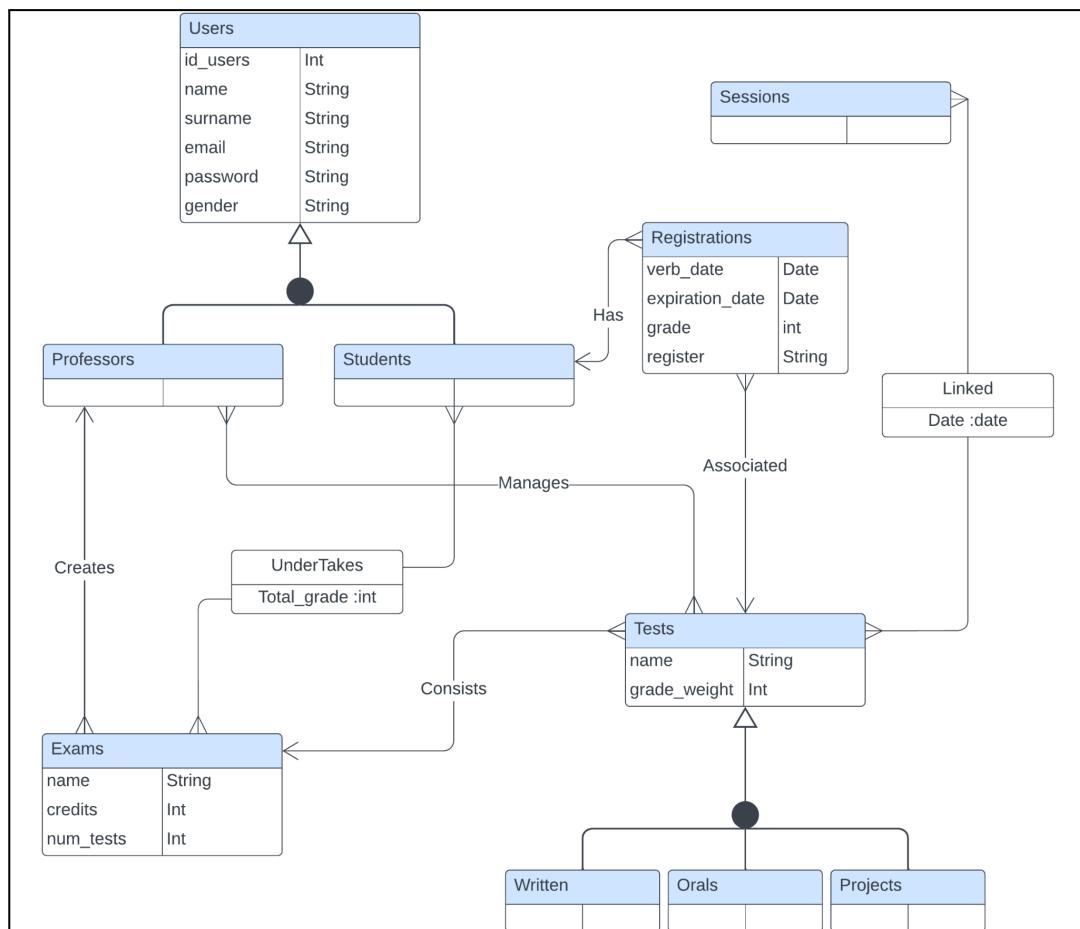
- **Pagina dedicata gestione errori:**

- Al fine di migliorare la trasparenza e la gestione degli errori nel nostro progetto, abbiamo deciso di implementare una pagina dedicata. Questa pagina sarà specificamente progettata per affrontare eventuali errori che potrebbero verificarsi durante l'esecuzione del sistema. È importante sottolineare che la creazione di quest'ultima non riflette una mancanza di considerazione da parte nostra, ma piuttosto un'attenzione riguardo all'utente che andrà ad utilizzare la nostra applicazione perché trovandosi in questa pagina avrà la possibilità di svolgere comunque varie azioni come tornare indietro, eseguire il logout, andare alla homepage, oppure andare nel suo profilo.

### 3. Progettazione Concettuale

Per convenzione abbiamo creato i nostri schemi (e l'applicazione) in lingua inglese.

Di seguito mostriamo lo schema che abbiamo creato nella **Progettazione Concettuale**:



Lo schema concettuale è composto da un insieme di entità e associazioni.

Di seguito una breve spiegazione delle nostre collezioni.

#### CLASSI

- **Users**: Per ogni utente, verranno salvati un insieme di dati essenziali, tra cui un codice identificativo, il nominativo completo, l'indirizzo email, la password di accesso, il genere e il ruolo.  
L'identificativo univoco sarà utilizzato come chiave primaria per distinguere in modo univoco ciascun utente nel sistema.  
L'indirizzo email e la password saranno le credenziali utilizzate durante la procedura di accesso all'applicazione, garantendo l'autenticazione sicura dell'utente.  
Infine, il ruolo svolge un ruolo importante nell' identificare se un utente è uno studente o un docente.
- **Exams**: Per ogni esame verrà salvato: il nome dell'esame, il numero dei crediti ad esso assegnati, il numero dei test di cui esso è composto e il voto totale. Ogni esame sarà

identificato da un codice univoco. Il voto totale rappresenterà la media pesata dei voti ottenuti nei singoli test.

- **Tests:** Di ogni test verrà salvato: il nome del test, il suo tipo (scritto-orale-progetto), il bonus e il peso che quel voto avrà sull'esame finale.
- **Registrations:** Di ogni registrazione verrà salvato: la data, la sua scadenza,e il suo voto.
- **Sessions:** Di ogni sessione verrà salvata la data.

## SOTTOCLASSI

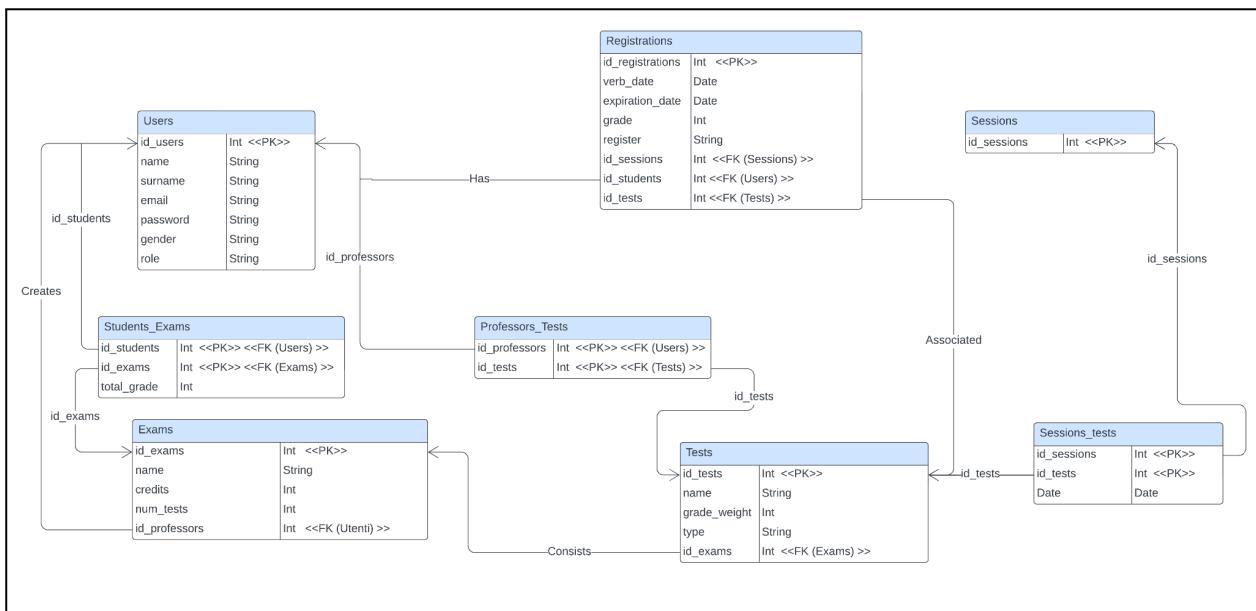
- **Professors e Students** sono sottoclassi di Users.
- **Written Orals e Projects** sono sottoclassi di Tests.

## ASSOCIAZIONI

Le Associazioni sono: **Creates,Consists,Manages,Has,Associated,Linked**.

- Professors → **Creates** → Exams (**1:N**)  
Un professore crea uno o più esami.  
Un esame è creato da un professore.
- Exams → **Consists** → Tests (**1:N**)  
Un esame consiste di una o più prove.  
Una prova compone un esame.
- Professors → **Manages** → Tests (**N:N**)  
Un professore gestisce uno o più esami.  
Un esame è gestito da uno o più professori.
- Students → **Has** → Registrations (**1:N**)  
Uno studente ha una o più registrazioni.  
Una registrazione è di uno studente.
- Students → **UnderTakes** → Exams(**N:N**)  
Uno studente svolge uno o più esami.  
Un esame è svolto da uno o più studenti.
- Registrations → **Associated** → Tests (**N:1**)  
Una prova è associata ad una o più registrazioni.  
Una registrazione è associata ad una prova.
- Tests → **Linked** → Sessions (**N:N**)  
Una prova è associata(linked) ad una o più sessioni.  
Una sessione è associata ad una o più prove.

# 4. Progettazione Logica



Il nostro modello Concettuale è stato tradotto in Quello Logico utilizzando le regole e i metodi visti durante il corso, di seguito un approfondimento di alcuni punti degni di attenzione.

## RAPPRESENTAZIONE GERARCHIE TRA CLASSI

Abbiamo utilizzato il modo della **relazione unica** per rappresentare le sottoclassi: **Professors - Students - Written - Orals e Projects**.

## TRASFORMAZIONE DELLE ASSOCIAZIONI N:N

Abbiamo trasformato poi le relazioni **Manages**, **UnderTakes** e **Linked** arrivando ad avere le classi: **Sessions\_tests**, **Students\_Exams** e **Professors\_tests**.

## PRIMARY KEYS (PK) e FOREIGN KEYS (FK)

- **Users**: id\_users (PK)
- **Exams**: id\_exams (PK), id\_professors (FK Users)
- **Tests**: id\_tests (PK), id\_exams (FK Exams)
- **Registrations**: id\_registrations (PK), id\_sessions (FK Sessions), id\_students (FK Users), id\_tests (FK Tests)
- **Sessions**: id\_sessions (PK)
- **Sessions\_tests**: id\_sessions (PK) (FK Sessions), id\_tests (PK) (FK Tests)
- **Students\_Exams**: id\_students (PK) (FK Users), id\_exams (PK) (FK Exams)
- **Professors\_tests**: id\_professors (PK) (FK Users), id\_tests (PK) (FK Tests)

# 5. Scelte Progettuali

## SQLAlchemy

Per creare le nostre query abbiamo utilizzato SQLAlchemy, una potente libreria ORM (Object-Relational Mapping).

### I Vantaggi delle query SQLAlchemy:

- **Astrazione**

SQLAlchemy offre un'interfaccia Python di alto livello per interagire con i database, astraendo la complessità SQL sottostante. Questa astrazione semplifica l'interazione del database, riducendo la necessità di codice specifico del database.

- **Leggibilità e manutenibilità**

Il codice SQLAlchemy tende ad essere più leggibile e più facile da mantenere rispetto alle query SQL grezze. Sfrutta classi e oggetti Python per rappresentare tabelle e righe di database, rendendo il codice intuitivo e autoesplicativo.

- **Portabilità**

Con SQLAlchemy, puoi passare senza problemi tra vari motori di database (ad esempio, SQLite, PostgreSQL, MySQL) semplicemente modificando un'impostazione di configurazione. Questa portabilità può far risparmiare notevoli sforzi di sviluppo.

- **Sicurezza (Protezione dalle SQL-injection)**

SQLAlchemy fornisce una difesa integrata contro gli attacchi di SQL injection.

L'input dell'utente viene automaticamente sanificato proteggendo il sistema.

## WTForms

Nella realizzazione del nostro progetto, per quanto riguarda la gestione dei form abbiamo utilizzato i **WTForms**:

WTForms è una libreria flessibile di convalida e rendering di moduli per Python sviluppo web. Uno dei principali vantaggi dell'utilizzo di WTForms è la:

- **Sicurezza (Protezione da Cross-Site Request Forgery (CSRF))**

Può funzionare con qualsiasi framework e modello web motore che scegli. Supporta la convalida dei dati, la protezione CSRF, internazionalizzazione (I18N) e altro ancora.

## RUOLI

Nel nostro progetto abbiamo poi deciso di implementare i ruoli in PostgreSQL, quelli che abbiamo implementato sono i seguenti: admin, professor, student.

PostgreSQL gestisce i permessi di accesso al database utilizzando il concetto di ruoli .

Un ruolo può essere considerato come un utente del database o un gruppo di utenti del database, a seconda di come è impostato il ruolo. I ruoli possono possedere oggetti di database (ad esempio

tabelle e funzioni) e possono assegnare privilegi su tali oggetti ad altri ruoli per controllare chi ha accesso a quali oggetti. Inoltre, è possibile concedere l'appartenenza ad un ruolo ad un altro ruolo, consentendo così al ruolo membro di utilizzare i privilegi assegnati ad un altro ruolo.

## FLASK LOGIN

Nel nostro progetto abbiamo utilizzato Flask-Login. Esso fornisce la gestione delle sessioni utente per Flask. Si occupa delle attività di accesso, disconnessione e memorizzazione delle sessioni degli utenti per periodi di tempo prolungati.

Flask-Login:

- Memorizza l'ID dell'utente attivo nella sessione di Flask e consente di effettuare facilmente il log-in e il log-out.
- Consente di limitare le visualizzazioni agli utenti connessi (o disconnessi). (login\_required)
- Aiuta a proteggere le sessioni degli utenti dal furto dei cookie.

Sempre per quanto riguarda l'autenticazione, per conservare la password nel database in modo sicuro e non in chiaro abbiamo utilizzato la libreria **flask\_bcrypt** che fornisce utilità di hashing bcrypt per l'applicazione.

Abbiamo utilizzato **Flask Blueprints** per poter suddividere il progetto in pagine diverse in modo da facilitarne la lettura.

## 6. Descrizione pagine applicazione

In questa sezione andremo a dare una spiegazione più dettagliata delle pagine e del funzionamento della nostra applicazione.

Aprendo la nostra applicazione si viene reindirizzati alla **homepage** dove viene descritta la nostra università e dove si possono trovare il login e il signup.

Schiacciando su **Signup** ci si può registrare e successivamente essere indirizzati al login.

Con il **Login** si può accedere al menù principale dei professori, se chi ha effettuato l'accesso è un professore, altrimenti al menù degli studenti se chi ha effettuato l'accesso è uno studente.

→ **Menù Professori:** In alto troveremo un menù con "Home","Profile","Logout"

Scrollando in basso troveremo le seguenti Pagine:

- **Exams Creation:** in questa pagina il professore potrà creare gli esami che vuole e le prove di cui sono composti.

- **Tests dates assignment:** in questa pagina il professore può decidere in quali date si terranno le prove create da lui in precedenza.
- **Tests Created:** in questa pagina il professore può visualizzare e anche modificare la lista delle prove che ha creato e la data che gli ha assegnato.
- **Tests grade registration:** in questa pagina il professore avrà la lista di studenti che hanno partecipato ad una sua prova e sarà in grado di inserire la valutazione che gli ha assegnato.
- **Students status records:** in questa pagina il professore è in grado di visualizzare l'elenco degli studenti con le prove da loro effettuate, anche se il voto è insufficiente,basta che la loro validità sia ancora attiva.
- **Tests grade registration:** in questa pagina il professore è in grado di visualizzare l'elenco degli studenti che sono in grado di avere il voto dell'esame registrato e registrarlo.
- **Account:** in questa pagina il professore può accedere al suo profilo.
- **Help:** in questa pagina vengono spiegate tutte le pagine del menù e determinate regole a cui deve attenersi il professore.

→ **Menù Studenti:** In altro troveremo un menù con "Home","Profile","Logout"

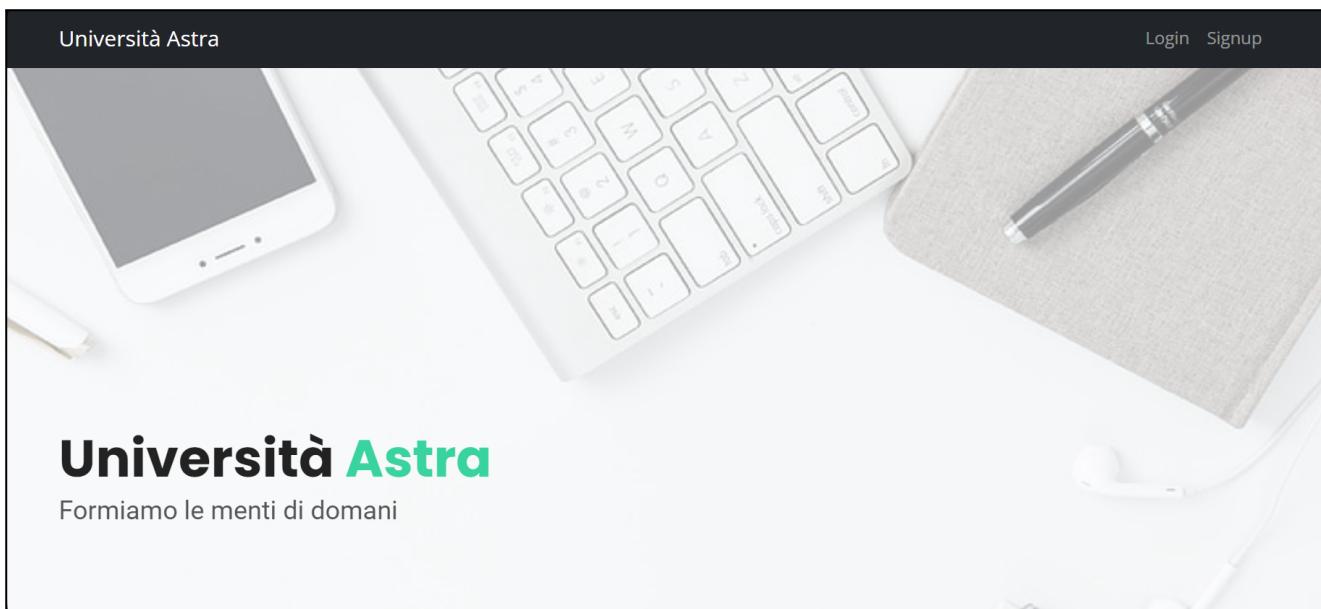
Scrollando in basso troveremo le seguenti Pagine:

- **Registration for Tests:** in questa pagina uno studente può iscriversi ad una o più prove.
- **Reservations made:** in questa pagina uno studente può visualizzare l'elenco delle prove a cui si è iscritto e può decidere se annullare la sua registrazione se la data della prova è almeno a tre giorni di distanza.
- **Results board:** in questa pagina uno studente può visualizzare le prove che ha svolto con esito positivo.
- **Career:** in questa pagina uno studente può visualizzare tutti gli esami che ha passato nella sua carriera ed i loro dettagli.

A titolo esemplificativo di seguito mostriamo solo qualche schermata del nostro progetto.

La nostra applicazione verrà mostrata nella sua interezza nel video dimostrativo consegnato.

## Homepage, la schermata iniziale



Università Astra

Login Signup

# Università Astra

Formiamo le menti di domani

## Riguardo la nostra università:

L'Università Astra è un'istituzione d'avanguardia che ridefinisce l'educazione superiore nel XXI secolo. Situata in un campus moderno e accogliente, questa università si distingue per l'approccio all'avanguardia alla formazione e per l'attenzione costante all'evoluzione delle esigenze globali e alle sfide emergenti.

Missioni e Valori: La nostra missione è quella di fornire un'educazione di alta qualità che prepari gli studenti a eccellere in un mondo in rapida trasformazione. I nostri valori fondamentali includono l'innovazione, l'accessibilità, l'integrità e la collaborazione.

Programmi Accademici Avanzati: Offriamo una vasta gamma di programmi accademici che spaziano dalla



Programmi Accademici Avanzati: Offriamo una vasta gamma di programmi accademici che spaziano dalle scienze umane alle tecnologie emergenti, garantendo che gli studenti abbiano accesso a un'istruzione di livello mondiale in settori chiave. I nostri docenti sono esperti nel loro campo e sono impegnati a guidare gli studenti verso l'eccellenza.

Tecnologie all'Avanguardia: L'Università Innovativa del Futuro integra tecnologie all'avanguardia in ogni aspetto dell'esperienza educativa. Dalle aule intelligenti ai laboratori virtuali, forniamo agli studenti gli strumenti necessari per affrontare le sfide del futuro.

L'Università Innovativa del Futuro è un faro di eccellenza nell'istruzione superiore, preparando gli studenti a guidare il futuro con intelligenza, creatività e responsabilità. Se sei pronto a fare parte di una comunità che abbraccia l'innovazione e l'evoluzione, l'Università



## Signup, la pagina di registrazione

### Sign Up

First Name

Last Name

Email Address

Password

Role

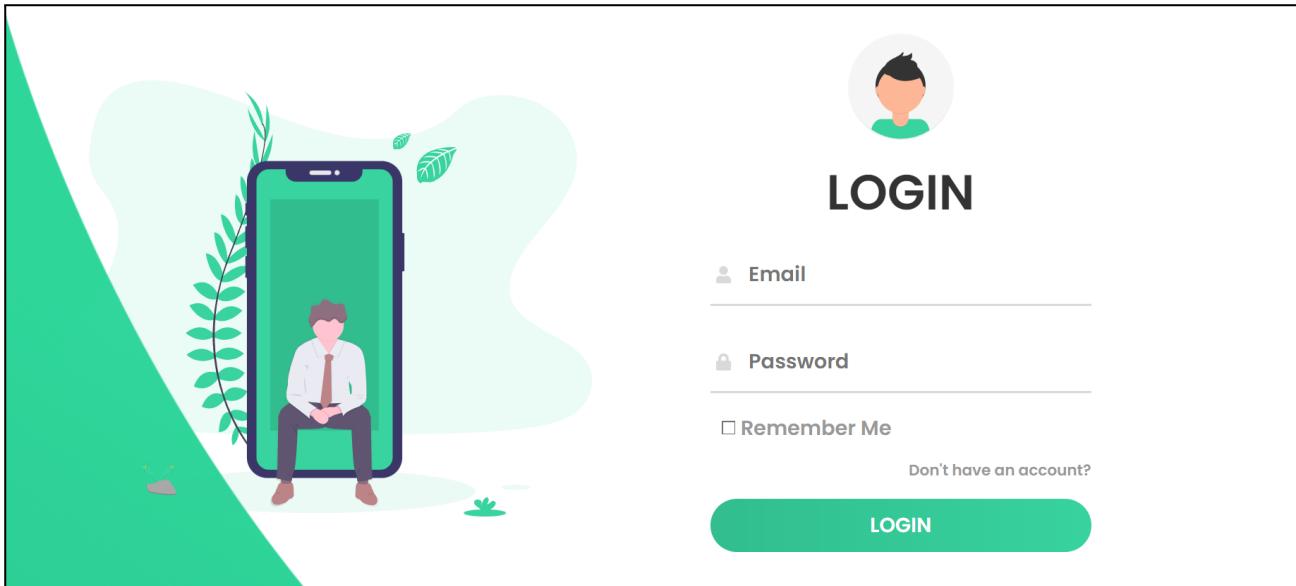
Gender

Female

Male

Other

## Login, la pagina di accesso



## Menù studenti, la pagina principale per gli studenti

The student menu home page has a green header bar with the text "Università Astra" on the left and "Home Profile Logout" on the right. A decorative green plant graphic is on the left side. The main content area features a large "Welcome bilico" message. On the right, there is a photograph of a person's hands typing on a laptop keyboard. The background of the main area is light gray.

The student menu services page has a green decorative graphic on the left. It contains four service cards with rounded corners: 1. "Registration for Tests": See the list of the available sessions and Subscribe. 2. "Reservations made": View the list of reservations made. 3. "Results board": View all the tests you passed with their details. 4. "Career": See all the exams you passed in your career with their details. At the bottom right, there is a footer with the text "Università Astra", "A108 Adam Street", "New York, NY 5350212", and "Phone: +39 358 461 3395".

## Menù professori, la pagina principale per i professori

The screenshot shows the main menu for professors. At the top, there's a green header bar with the text "Università Astra" on the left and "Home Profile Logout" on the right. Below the header is a decorative background image featuring a potted plant on the left and a laptop keyboard on the right. In the center, the text "Welcome simone" is displayed in a large, bold, black font.

This screenshot shows the main menu for professors with eight management options arranged in two rows:

- Exams Creation**: Create the exam and the tests it consists of.
- Tests dates assignment**: Assign a session date to the tests created.
- Tests Created**: Here you can see the list of tests you created and their associated dates.
- Tests grades registration**: View the list of tests taken by students and enter ratings.
- Students status record**: View the student's test and exam record.
- Exams grades registration**: Enter the final grade of the exams taken by the students.
- Account**: Here you can access your profile.
- Help**: Here is an explanation of the pages in this web application.

## Error, pagina di gestione errori

The screenshot shows an error page with a central message box. The message box contains the word "Oops!" in a large, bold, black font at the top. Below it, a paragraph of text reads: "If you are seeing this page, it's because you tried to insert some incorrect data, or the page you are trying to access is empty." At the bottom of the message box is a blue "Go Back" button. The background features a stylized illustration of books and papers.

## 7. Risorse

Per la creazione del nostro progetto abbiamo utilizzato le lezioni frontali ed i lucidi messi a disposizione dal professore.

Altre risorse che abbiamo utilizzato:

- Real Python
- Flask-SQLAlchemy
- Flask SQLAlchemy (with Examples) - Python Tutorial
- Flask-Login
- BBBootstrap
- Stack Overflow
- Lucid.app

Università Astra

