

**UNIVERSITÀ DEGLI STUDI DI NAPOLI**  
**PARTHENOPE**  
**DIPARTIMENTO DI INGEGNERIA**



**CORSO DI LAUREA IN INGEGNERIA E SCIENZE  
INFORMATICHE PER LA CYBERSECURITY**

**TITOLO PROGETTO**

**UniLabManager**

**DOCENTE**

**PROFESSORE DANIELE GRANATA**

**STUDENTE**

**Andrea Di Palo, 0334000176**

**ANNO ACCADEMICO 2024/2025**

## Descrizione del progetto:

Il progetto *UniLabManager* è un sistema web progettato per gestire in maniera centralizzata le attività didattiche e sperimentali svolte all'interno dei laboratori universitari.

L'obiettivo principale è offrire un'interfaccia efficiente per la **creazione e supervisione di progetti scientifici**, la **gestione di esperimenti**, e la **prenotazione di attrezzature e laboratori**, con ruoli differenziati per **professori, studenti e tecnici**.

## Obiettivi:

- Permettere la **registrazione differenziata** degli utenti secondo il proprio ruolo (Professore, Studente, Tecnico)
- Consentire al Professore di creare progetti sperimentali e associarvi esperimenti
- Fornire la possibilità di **prenotare attrezzature e laboratori** per ogni esperimento
- Offrire una **dashboard personalizzata** per ciascun utente
- Garantire **sicurezza dei dati**, autenticazione tramite matricola e accessi controllati

## Requisiti Funzionali

### RF1 – Registrazione dell'utente

Il sistema deve permettere la registrazione di nuovi utenti specificando il ruolo (Professore, Studente, Tecnico), tramite un form personalizzato per ciascun tipo. Durante la registrazione, viene verificata e salvata una matricola unica con formato specifico (P12345 per professori, S12345 per studenti, T12345 per tecnici).

### RF2 – Autenticazione

Il sistema deve consentire l'autenticazione degli utenti registrati tramite matricola e password.

### **RF3 – Accesso a dashboard personalizzata**

Dopo l'autenticazione, ogni utente accede ad una dashboard diversa a seconda del ruolo:

- I professori visualizzano i propri progetti sperimentali e possono crearne di nuovi.
- Gli studenti accedono a contenuti didattici o prenotazioni (in futuro).
- I tecnici visualizzano le prenotazioni da gestire.

### **RF4 – Creazione progetto sperimentale (Professore)**

Un professore autenticato può creare nuovi progetti sperimentali inserendo titolo, descrizione, obiettivi, date, ecc.

### **RF5 – Visualizzazione ed eliminazione dei progetti**

I professori possono visualizzare l'elenco dei propri progetti e scegliere di eliminarne uno se necessario.

### **RF6 – Creazione esperimenti associati a un progetto**

Per ogni progetto, il professore può creare uno o più esperimenti, specificando descrizione, obiettivi, materiali e date.

### **RF7 – Prenotazione laboratorio e attrezzature**

Durante la creazione di un esperimento, il sistema consente al professore di prenotare un laboratorio e selezionare una o più attrezzature, evitando conflitti di orario o doppie prenotazioni.

### **RF8 – Verifica disponibilità risorse**

Il sistema deve impedire la prenotazione di laboratori o attrezzature già occupati nella stessa fascia oraria.

### **RF9 – Partecipazione agli esperimenti (Studente)**

Il sistema deve consentire agli **studenti autenticati** di visualizzare l'elenco dei **progetti disponibili** e di **richiedere la partecipazione** agli esperimenti collegati. L'accesso a queste funzionalità è riservato solo agli utenti con ruolo "studente" che hanno effettuato il login.

### **RF10 – Gestione limite partecipanti per progetto sperimentale**

Ogni progetto sperimentale deve avere un campo che indica il **numero massimo di studenti ammessi**. Il sistema deve impedire ulteriori iscrizioni quando il limite è stato raggiunto

### **RF11 – Logout**

Il sistema deve permettere in ogni momento il logout dell'utente.

### **RF12 - Gestione Attrezzatura**

Il tecnico deve poter aggiungere e/o modificare lo stato delle attrezzature

### **RF13 - Modifica di Progetti ed Esperimenti**

Il sistema deve permettere a un professore autenticato di modificare i dettagli di un progetto esistente (es. titolo, descrizione, numero di posti) o di un esperimento (es. obiettivi, materiali). La modifica delle date o delle risorse di un esperimento richiederà una nuova verifica della disponibilità per evitare conflitti.

### **RF14 - Sistema di Notifiche Automatiche**

Il sistema deve inviare notifiche agli utenti al verificarsi di eventi rilevanti. Ad esempio:

- Un professore riceve una notifica quando uno studente si iscrive a un suo progetto.
- Uno studente riceve una notifica se un esperimento a cui partecipa viene modificato o annullato dal professore.
- Un professore viene notificato se un'attrezzatura che ha prenotato viene messa "In manutenzione" dal tecnico.

### **RF15 - Caricamento Risultati e Feedback per gli Esperimenti**

Gli studenti devono avere la possibilità di caricare un file (es. un report in PDF, un foglio di calcolo con i dati) per ogni esperimento a cui hanno partecipato.

Successivamente, il professore responsabile deve poter visualizzare i file caricati, lasciare un commento di feedback e segnare l'esperimento come "Completato" per lo studente.

## RF16 - Visualizzazione Calendario Prenotazioni

Il sistema deve offrire una vista calendario per una migliore gestione delle risorse.

- **Professori:** Possono visualizzare un calendario personale con le proprie prenotazioni di laboratori e attrezzature.
- **Tecnici:** Possono visualizzare un calendario generale del laboratorio di cui sono responsabili per avere una visione chiara dell'occupazione giornaliera e settimanale.

## RF17 - Gestione Profilo Utente Personale

Ogni utente autenticato (Professore, Studente, Tecnico) deve poter accedere a una pagina "Il Mio Profilo" per visualizzare i propri dati personali. In questa sezione, deve essere in grado di eseguire azioni come la modifica della propria password o dell'indirizzo email.

## Matrice di tracciabilità dei requisiti

ID	Descrizione	Implementato	Non implementato
RF1	Registrazione differenziata degli utenti (Professore, Studente, Tecnico).	SI	
RF2	Autenticazione degli utenti registrati tramite matricola e password.	SI	
RF3	Accesso a una dashboard personalizzata a seconda del ruolo.	SI	
RF4	Creazione di nuovi progetti sperimentali da parte del professore.	SI	
RF5	Visualizzazione ed eliminazione dei propri progetti da parte dei professori.	SI	
RF6	Creazione di esperimenti associati a un progetto.	SI	

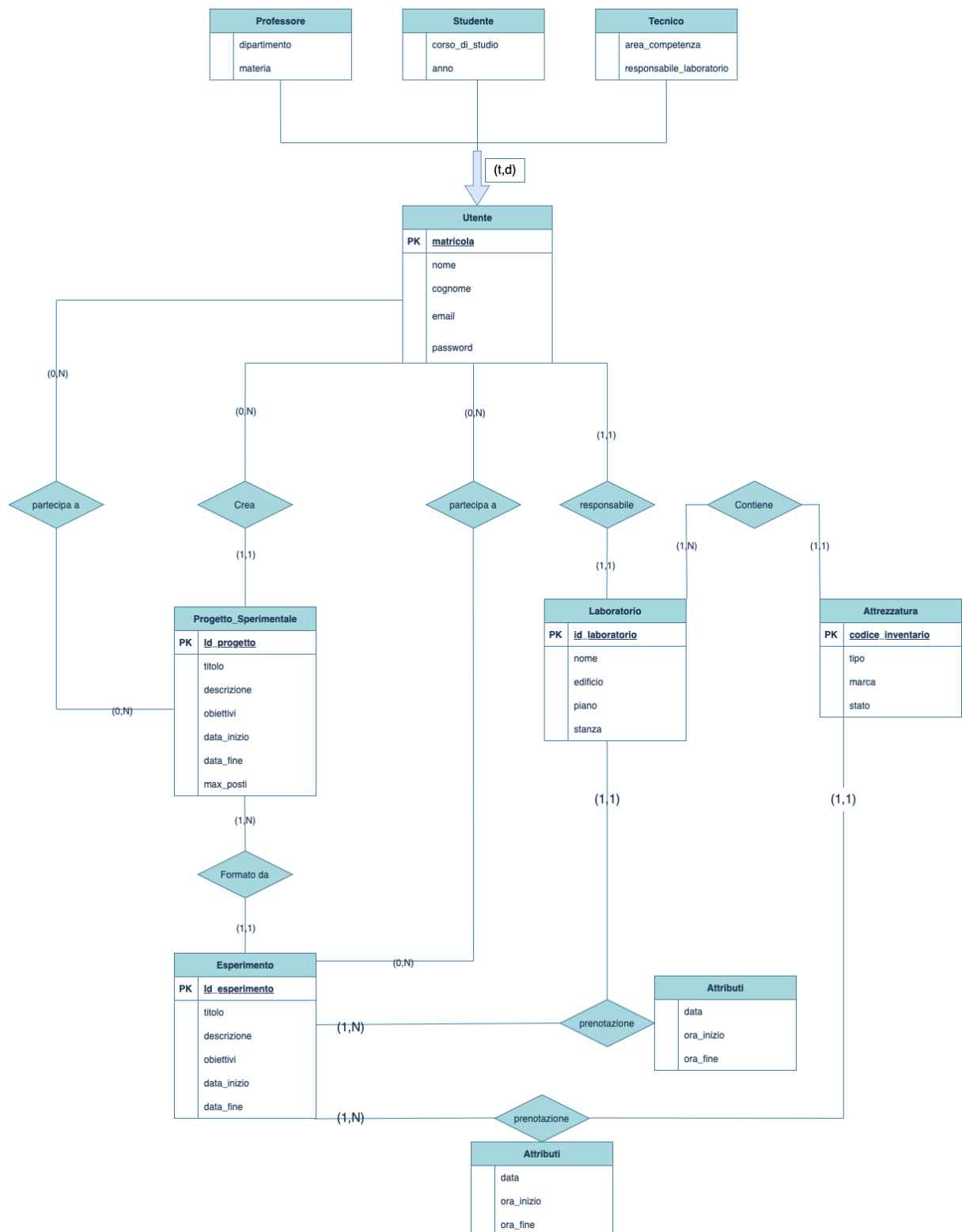
<b>RF7</b>	Prenotazione di laboratorio e attrezzature durante la creazione di un esperimento.	SI	
<b>RF8</b>	Verifica della disponibilità delle risorse per impedire doppie prenotazioni.	SI	
<b>RF9</b>	Partecipazione degli studenti ai progetti e agli esperimenti disponibili.	SI	
<b>RF10</b>	Gestione del limite massimo di partecipanti per ogni progetto.	SI	
<b>RF11</b>	Possibilità per l'utente di effettuare il logout in ogni momento.	SI	
<b>RF12</b>	Gestione delle attrezzature (aggiunta e modifica stato) da parte del tecnico.	SI	
<b>RF13</b>	Modifica di Progetti ed Esperimenti esistenti da parte del professore.		NO
<b>RF14</b>	Sistema di Notifiche Automatiche per eventi rilevanti (nuove iscrizioni, modifiche, etc.).		NO
<b>RF15</b>	Caricamento Risultati e Feedback per gli esperimenti da parte di studenti e professori.		NO
<b>RF16</b>	Visualizzazione Calendario delle prenotazioni per professori e tecnici.		NO
<b>RF17</b>	Gestione del Profilo Utente per modificare i propri dati (password, email).		NO

## Modello Concettuale (E-R)

Il servizio web è pensato per la gestione dei progetti e delle attività sperimentali nei laboratori universitari.

1. I **professori** possono creare progetti, definire esperimenti e prenotare laboratori e attrezzature.
2. Gli **studenti**, una volta autenticati, possono visualizzare i progetti disponibili, prenotarsi e monitorare le attività a cui partecipano.
3. I **tecnici** hanno accesso a un pannello dedicato per gestire la disponibilità delle attrezzature e supervisionare le prenotazioni.

Ogni utente accede con credenziali univoche e visualizza solo le funzionalità legate al proprio ruolo. Il sistema garantisce un'organizzazione efficiente, evitando sovrapposizioni e semplificando la comunicazione tra studenti, docenti e tecnici.





Inizialmente è stato realizzato un modello E/R con un'entità padre e tre entità figlie, collegate da una relazione di specializzazione totale e disgiunta.

1. Relazione Profilo → Professore & Studente & Tecnico
2. Vincolo di copertura: Totale → Ogni Profilo è sempre un Professore, uno Studente o un Tecnico.
3. Vincolo di disgiunzione: Disgiunta → Un Profilo è o Professore o Studente o Tecnico, mai una combinazione delle tre.

La specializzazione dell'entità Profilo è stata definita in questo modo:

- Attributi comuni: matricola, nome, cognome, email
- Attributi privati: password

## Gestione della generalizzazione:

### Accorpamento in padre

Nel progetto **UniLabManager**, per la gestione delle diverse tipologie di utenti (Professore, Studente, Tecnico) si è deciso di adottare una strategia di **generalizzazione con accorpamento nel padre**, che come rappresentato nel modello E-R, consiste nel mappare l'entità genitore **Utente** e tutte le sue specializzazioni in un'unica tabella fisica. La distinzione dei ruoli è affidata all'attributo omonimo **Ruolo**.

### Motivazioni della scelta:

#### 1. Semplificazione architetturale

L'utilizzo di un'unica tabella **Utente** rende lo schema del database semplice, lineare e facile da comprendere. Si evita la complessità di dover gestire e mantenere più tabelle correlate per rappresentare una singola gerarchia, riducendo i possibili errori e facilitando la manutenzione.

#### 2. Efficienza

Poiché tutti i dati degli utenti sono in un'unica tabella, non è necessario eseguire operazioni di JOIN per recuperare le informazioni complete di un **Utente**. Ad esempio, per ottenere il nome di uno studente (attributo comune) e il suo anno di corso (attributo specifico), è sufficiente una singola e veloce.

## **Svantaggi:**

### **1. Presenza di Valori NULL:**

La tabella **Utente** contiene colonne specifiche per certi ruoli che rimarranno NULL per altri. Ad esempio, la colonna `area_competenza` sarà NULL per gli studenti.

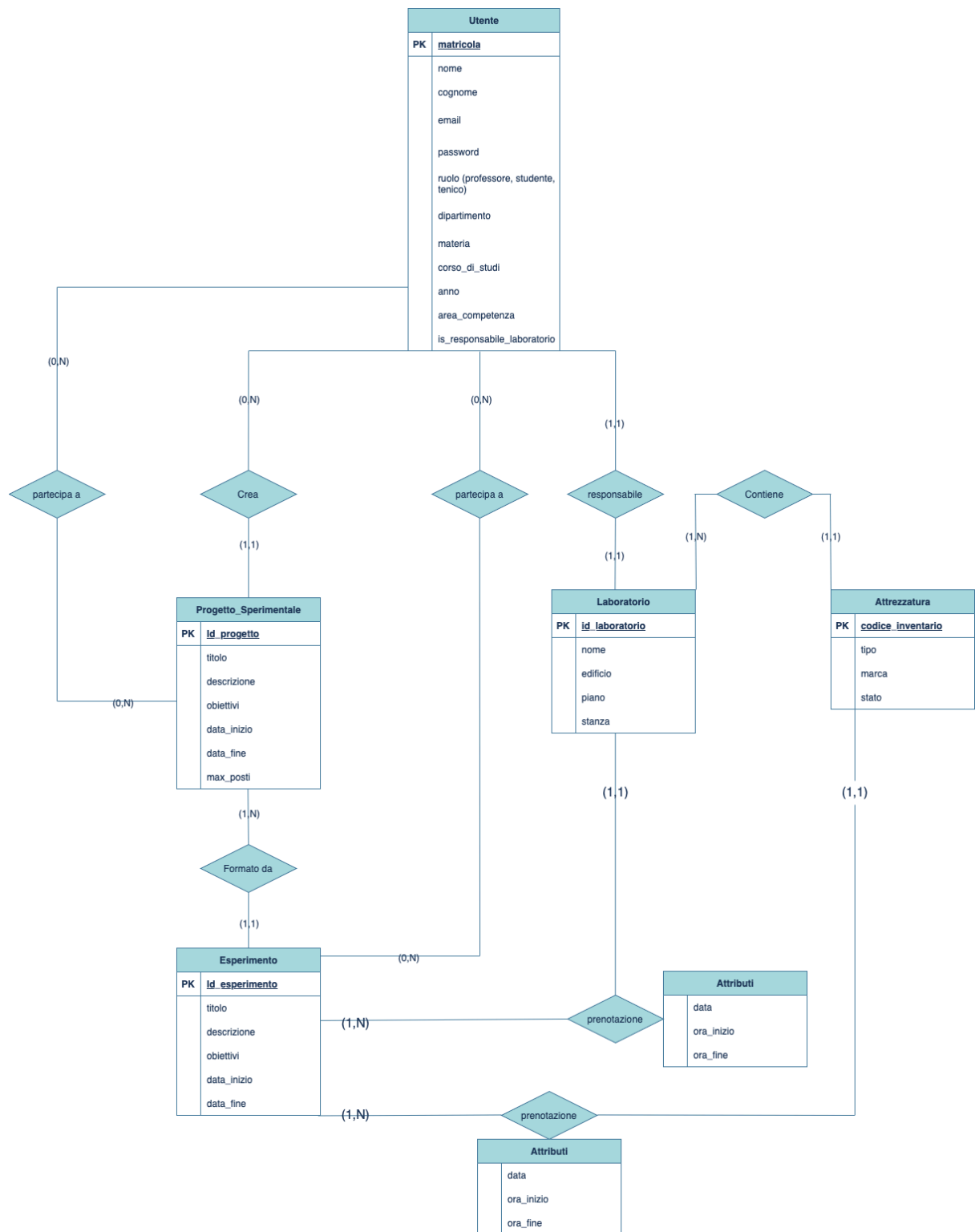
### **2. Mitigazione:**

Nel contesto del progetto, il numero di attributi specifici per ciascun ruolo è limitato. Lo spreco di spazio è trascurabile. L'integrità dei dati (ad esempio, assicurarsi che uno studente abbia sempre un anno definito) viene gestita a livello di logica applicativa o tramite vincoli di controllo a livello di codice.

## **Aspetti implementativi e vincoli:**

- Un **Utente** può essere solo o professore, o studente, o tecnico . Quindi non può essere contemporaneamente professore e studente. Questo vincolo è garantito dal fatto che l'attributo **Ruolo** può contenere un solo valore.
- **Generalizzazione totale:**  
Ogni **Utente** nel sistema deve obbligatoriamente appartenere a un ruolo. Questo viene imposto definendo l'attributo **ruolo** come obbligatorio (NOT NULL) al momento della creazione di un nuovo utente.
- Le relazioni come Crea (tra **Utente** e **Progetto\_Sperimentale**), partono dall'unica tabella **Utente**. Sarà compito della logica applicativa verificare che il ruolo dell'utente che avvia la relazione sia appropriato (es. solo un 'professore' può creare un progetto)

## Modello E-R finale



## Modello logico

Traducendo il modello E-R finale, si ottiene il seguente schema relazionale

- **\*\*Utente\*\***(\_matricola\_, nome, cognome, email, password, ruolo, dipartimento, materia, corso\_di\_studi, anno, area\_competenza, is\_responsabile)
- **\*\*Laboratorio\*\***(\_id\_laboratorio\_, nome, edificio, piano, stanza, responsabile:Utente)
- **\*\*Attrezzatura\*\***(\_codice\_inventario\_, tipo, marca, stato, laboratorio:Laboratorio)
- **\*\*ProgettoSperimentale\*\***(\_id\_progetto\_, titolo, descrizione, obiettivi, data\_inizio, data\_fine, max\_posti, docente:Utente)
- **\*\*Esperimento\*\***(\_id\_esperimento\_, titolo, descrizione, obiettivi, data\_inizio, data\_fine, progetto:ProgettoSperimentale)
- **\*\*PartecipazioneProgetto\*\***(\_id\_partecipazione\_prog\_, progetto:ProgettoSperimentale, studente:Utente)
- **\*\*PartecipazioneEsperimento\*\***(\_id\_partecipazione\_esp\_, esperimento:Esperimento, studente:Utente)
- **\*\*PrenotazioneLaboratorio\*\***(\_id\_prenotazione\_lab\_, data, ora\_inizio, ora\_fine, docente:Utente, laboratorio:Laboratorio, esperimento:Esperimento)
- **\*\*PrenotazioneAttrezzatura\*\***(\_id\_prenotazione\_attr\_, data, ora\_inizio, ora\_fine, docente:Utente, attrezzatura:Attrezzatura, esperimento:Esperimento)

## 1) UTENTE

Nome Campo	Descrizione	Tipo	Lunghezza	Vincoli
Matricola	Identificativo dell'utente	Char	6	Primary Key, Not null, Unique
Nome	Nome dell'utente	Varchar	100	Not null
Cognome	Cognome dell'utente	Varchar	100	Not null
Email	Email dell'utente	Varchar	150	Not null, Unique
Password	Password dell'utente	Varchar	128	Not null
Ruolo	Tipologia di utente (Professore, Studente, Tecnico)	Enum	-	Not null, valori ammessi ['Professore', 'Studente', 'Tecnico']
Dipartimento	Dipartimento di appartenenza	Varchar	100	Not null se ruolo = 'Professore'
Materia	Materia insegnata	Varchar	100	Not null se ruolo = 'Professore'
Corso di Studi	Corso di Studi di appartenenza	Varchar	100	Not null se ruolo = 'Studente'
Anno	Anni di corso	Int 1<x<5	-	Not null se ruolo = 'Studente'
Area di competenza	Competente in	Varchar	100	Not null se ruolo = 'Tecnico'
Is_responsabile	Responsabilità laboratorio	Bool	-	Not null se ruolo = 'Tecnico'

### Ruolo

Un utente deve essere o un Professore o uno Studente o un Tecnico:

- Se l'utente è un professore si compileranno i campi matricola, nome, cognome, email, password, ruolo, **dipartimento** e **materia**; i campi restanti saranno di tipo Null;
- Se l'utente è uno studente si compileranno i campi matricola, nome, cognome, email, password, ruolo, **corso di studi** e **anno**; i campi restanti saranno di tipo Null;
- Se l'utente è un tecnico si compileranno i campi matricola, nome, cognome, email, password, ruolo, **area di competenza** e **is\_responsabile**; i campi restanti saranno di tipo Null;

### Vincoli di tupla

Un utente non può registrarsi due volte con la stessa matricola o mail.

## 2) LABORATORIO

Nome Campo	Descrizione	Tipo	Lunghezza	Vincoli
Id Laboratorio	Codice identificativo del laboratorio	Int	5	Primary key, not null, unique
Matricola	Matricola del tecnico responsabile	Char	6	Foreign Key → Utente(matricola), Not null, unique, on delete cascade, limit_choice(ruolo= tecnico)
Nome	Nome del laboratorio	Varchar	100	Not null
Edificio	Edificio in cui si trova il laboratorio	Varchar	100	Not null
Piano	Piano dell'edificio	Integer	–	Not null
Stanza	Numero della stanza	Varchar	20	Not null

### Vincoli di tupla

- Ad un tecnico può essere associato un solo laboratorio;

## 3) ATTREZZATURA

Nome Campo	Descrizione	Tipo	Lunghezza	Vincoli
Codice_inventario	Codice inventario identificativo	Int	5	Primary Key, Unique, Not null
Laboratorio	Id laboratorio a cui è assegnata	Int	5	Foreign Key → Laboratorio(id), Not null, on delete cascade
Tipo	Tipo di attrezzatura	Varchar	100	Not null
Marca	Marca dell'attrezzatura	Varchar	100	Not null
Stato	Stato dell'attrezzatura (es. disponibile)	Enum	-	Not null, valori ammessi ['Funzionante', 'In manutenzione', 'Non disponibile']

## 4) PRENOTAZIONE LABORATORIO

### Relazione N:M tra Esperimento e Laboratorio

La relazione tra un esperimento e il laboratorio in cui si svolge è di tipo **multi-a-multi** (N:M). La logica è la seguente:

- Un singolo **Esperimento**, per essere eseguito, richiede la prenotazione di un laboratorio in una specifica data e fascia oraria.
- Un singolo **Laboratorio** può essere utilizzato per ospitare numerosi e diversi esperimenti, solo se le loro prenotazioni non si sovrappongono temporalmente.

Questa relazione viene risolta attraverso la tabella **PrenotazioneLaboratorio**. Tale tabella non si limita a collegare **Esperimento** e **Laboratorio**, ma include anche le informazioni temporali (data, ora inizio, ora fine) e il docente responsabile della prenotazione

Nome Campo	Descrizione	Tipo	Lunghezza	Vincoli
Id prenotazione laboratorio	Identificativo prenotazione	Int	10	Primary key, not null, unique
Matricola	Matricola del professore che ha effettuato la prenotazione	Char	6	Foreign Key → Utente(matricola), Not null, on delete cascade, limit_choice (ruolo=professore)
Laboratorio	Id del laboratorio prenotato	Int	5	Foreign Key → Laboratorio(id), Not null, on delete cascade
Esperimento	ID dell'esperimento associato	Int	5	Foreign Key → Esperimento (Id_esperimento), Not null, on delete cascade
Data	Data della prenotazione	Date	—	Not null

Ora_inizio	Ora di inizio della prenotazione	Time	–	Not null
Ora_fine	Ora di fine della prenotazione	Time	–	Not null

## 5) PRENOTAZIONE ATTREZZATURA

### Relazione N:M tra Esperimento e Attrezzatura

Nel sistema, la relazione tra un esperimento e le attrezzature necessarie per svolgerlo è di tipo **multi-a-molti (N:M)**, infatti:

- Un singolo **Esperimento** può richiedere l'utilizzo di una o più attrezzature diverse per essere eseguito
- Una specifica **Attrezzatura** può essere prenotata e utilizzata per diversi esperimenti, a condizione che le fasce orarie non si sovrappongano

Dunque per rappresentare questa relazione è necessaria una tabella ponte tra **Esperimento** e **Attrezzatura** che include la data, gli orari della prenotazione e il docente responsabile

Nome Campo	Descrizione	Tipo	Lunghezza	Vincoli
Id prenotazione attrezzatura	Identificativo prenotazione	Int	10	Primary key, not null, unique
Matricola	Matricola del professore che ha effettuato la prenotazione	Char	6	Foreign Key → Utente(matricola), Not null,on delete cascade, limit_choice (ruolo=professore)
Codice Inventario	Codice inventario attrezzatura prenotata	Int	5	Foreign Key → Attrezzatura (codice_inventario), Not null,on delete cascade
Esperimento	Id dell'esperimento associato	Int	5	Foreign Key → Esperimento (Id_esperimento), Not null,on delete cascade



Data	Data della prenotazione	Date	–	Not null
Ora_inizio	Ora di inizio della prenotazione	Time	–	Not null
Ora_fine	Ora di fine della prenotazione	Time	–	Not null

## 6) PROGETTO SPERIMENTALE

Nome Campo	Descrizione	Tipo	Lunghezza	Vincoli
Id progetto	Identificativo univoco del progetto	Int	5	Primary Key, Unique, Not null
Matricola	Matricola del professore responsabile	Char	6	Foreign Key →Utente(matricola), Not null,on delete cascade, limit_choice (ruolo=professore)
Titolo	Titolo del progetto	Varchar	100	Not null
Descrizione	Descrizione dettagliata	Text	–	Not null
Obiettivi	Obiettivi del progetto	Text	–	Not null
Data_inizio	Data di inizio del progetto	Date	–	Not null
Data_fine	Data di fine del progetto	Date	–	Not null
Max_posti	Numero massimo di partecipanti	Int	default = 1 1<x<30	Not null, Check > 0

## 7) PARTECIPAZIONE PROGETTO

### Relazione N:M tra Utente (Studente) e Progetto Sperimentale

Nel sistema, la relazione che lega gli studenti ai progetti sperimentali è di tipo **molti-a-molti (N:M)**. Questa cardinalità deriva dalla seguente logica:

- Uno **Studente** può iscriversi e partecipare a uno o più progetti sperimentali durante la sua carriera accademica
- Un **Progetto Sperimentale** è pensato per accogliere un numero definito di partecipanti, quindi può avere più studenti iscritti, fino al raggiungimento del numero massimo di posti

Inoltre, per evitare iscrizioni duplicate, uno studente può iscriversi solo una volta ad un determinato progetto.

Dunque per rappresentare questa relazione è necessaria una tabella ponte tra **Utente (studente)** e **Progetto Sperimentale**

Nome Campo	Descrizione	Tipo	Lunghezza	Vincoli
Id partecipazione prog	Identificativo della partecipazione al progetto	Int	5	Primary key, unique, not null
Id progetto	ID del progetto a cui si partecipa	Int	5	Foreign Key → Progetto_Sperimentale (Id_progetto), Not null, on delete cascade
Matricola	Matricola dello studente partecipante	Char	6	Foreign Key → Utente(matricola), Not null, on delete cascade, limit_choice (ruolo=studente)

## 8) ESPERIMENTO

Nome Campo	Descrizione	Tipo	Lunghezza	Vincoli
Id_esperimento	Identificativo univoco dell'esperimento	Int	5	Primary Key, Unique, Not null
Id progetto	ID del progetto associato	Int	5	Foreign Key → Progetto_Sperimentale (Id_progetto), Not null, on delete cascade
Titolo	Titolo dell'esperimento	Varchar	100	Not null
Descrizione	Descrizione dettagliata	Text	200	Not null
Obiettivi	Obiettivi specifici	Text	100	Not null
Data_inizio	Data di inizio dell'esperimento	Date	–	Not null
Data_fine	Data di fine dell'esperimento	Date	–	Not null

## 9) PARTECIPAZIONE ESPERIMENTO

### Relazione N:M tra Utente (Studente) e Esperimento

La relazione che registra la partecipazione degli studenti ai singoli esperimenti è di tipo **multi-a-molti (N:M)**. Questa cardinalità si basa su:

- Uno **Studente** può partecipare a uno o più esperimenti, sia all'interno dello stesso progetto sperimentale sia in progetti diversi.
- Un **Esperimento** è progettato per essere svolto da più studenti, spesso da tutti quelli iscritti al progetto a cui l'esperimento appartiene.

Per tracciare quale studente partecipa a quale esperimento, si utilizza la tabella associativa **PartecipazioneEsperimento**. Questa tabella funge da registro, creando un collegamento univoco tra un **Utente** (con ruolo Studente) e un'istanza di **Esperimento**. Inoltre, il sistema impone il vincolo logico per cui uno studente può partecipare a un esperimento solo se è già iscritto al progetto sperimentale associato.

Nome Campo	Descrizione	Tipo	Lunghezza	Vincoli
Id partecipazione esp	Identificativo della partecipazione all'esperimento	Int	5	Primary key, unique, not null
Id esperimento	ID dell'esperimento prenotato	Int	–	Foreign Key →Esperimento (Id_esperimento), Not null, on delete cascade
Matricola	Matricola dello studente	Char	6	Foreign Key →Utente(matricola), Not null,on delete cascade, limit_choice(ruolo=studente)

## Vincoli di integrità e relazionali

Il sistema implementa una serie di vincoli di integrità che garantiscono la coerenza logica dei dati attraverso più relazioni. Oltre ai vincoli di integrità referenziale e di dominio definiti direttamente a livello di database (come UNIQUE, FOREIGN KEY con azioni ON DELETE e vincoli sui ruoli), il sistema implementa le seguenti regole di business più complesse a livello applicativo, tramite la logica delle viste e dei form di Django

I vincoli principali sono:

- **Coerenza partecipazione progetto-esperimento:** Uno studente può richiedere la partecipazione a un esperimento solo ed esclusivamente se risulta già iscritto al progetto sperimentale a cui l'esperimento appartiene. Questo previene l'accesso a esperimenti da parte di studenti non autorizzati.
- **Proprietà delle risorse:** Un professore può modificare o eliminare solo i progetti e gli esperimenti da lui creati. Il sistema impedisce che un docente possa interferire con le attività di un collega.
- **Unicità della partecipazione ai progetti:** Per garantire l'integrità dei dati, uno studente può iscriversi una sola volta a un determinato progetto sperimentale. Il sistema impedisce iscrizioni duplicate allo stesso progetto da parte del medesimo studente.
- **Disponibilità e stato delle Risorse:** La prenotazione di un'attrezzatura per un esperimento è permessa solo se lo stato dell'attrezzatura è *'Funzionante'*. Il sistema blocca la prenotazione di attrezzature in stato *'In manutenzione'* o *'Non disponibile'*.

- **Prevenzione conflitti di prenotazione:** Il sistema è progettato per impedire la prenotazione di laboratori o attrezzature che risultano già occupati nella stessa data e fascia oraria. Al momento della creazione di un esperimento, viene eseguita una verifica della disponibilità delle risorse per evitare doppie prenotazioni e conflitti.

## Implementazione del servizio web

L'applicazione web **UniLabManager** è stata sviluppata utilizzando il framework **Django** e utilizzando il database **SQLite** in modo da creare una piattaforma centralizzata per la gestione delle attività didattiche e sperimentali nei laboratori universitari. L'obiettivo è offrire un'interfaccia efficiente per la creazione di progetti, la gestione di esperimenti e la prenotazione di risorse, rispettando i ruoli e i permessi di professori, studenti e tecnici, come definito nei requisiti di progetto.

A differenza del sistema di autenticazione predefinito di Django, si è scelto di implementare un sistema di gestione degli utenti personalizzato basato su un singolo modello **Utente**. Questo modello contiene un campo **ruolo** che distingue le tre tipologie di utenti (**Professore, Studente, Tecnico**), permettendo di mantenere una struttura del database semplice e di gestire gli accessi tramite logica implementata nelle views di Django. La gestione della sessione utente si basa sull'ID di sessione di Django, in cui vengono memorizzati **matricola** e **ruolo** dopo un login andato a buon fine.

## Funzionalità implementate

- **Registrazione e Autenticazione (RF1, RF2):** Il sistema offre form di registrazione separati per Professori, Studenti e Tecnici. Ogni form valida il formato specifico della matricola (es. P12345 per i professori) tramite espressioni regolari definite nei forms (es. ProfessoreForm). L'autenticazione avviene tramite la vista *login\_view*, che verifica la corrispondenza tra matricola e password (precedentemente hashata).
- **Dashboard Personalizzate (RF3):** Dopo il login, ogni utente viene reindirizzato a una dashboard specifica per il suo ruolo.
  - **Professore:** La *dashboard\_professore* visualizza l'elenco dei progetti creati e permette di crearne di nuovi o di eliminarli.
  - **Studente:** La *dashboard\_studente* mostra sia i progetti a cui lo studente è iscritto sia quelli ancora disponibili per l'iscrizione.

- **Tecnico:** La *dashboard\_tecnico* mostra le attrezzature del laboratorio di cui è responsabile e le prenotazioni imminenti.
- **Gestione Progetti e Esperimenti (RF4, RF5, RF6):** I professori possono creare progetti tramite la *crea\_progetto\_view*. Per ogni progetto, possono visualizzare un dettaglio e aggiungere esperimenti tramite la *crea\_esperimento\_view*. L'eliminazione di progetti e esperimenti è gestita rispettivamente da *elimina\_progetto\_view* e *elimina\_esperimento\_view*, che includono controlli di autorizzazione.
- **Prenotazione e Verifica Disponibilità Risorse (RF7, RF8):** Durante la creazione di un esperimento, il *CreaEsperimentoForm* permette al professore di selezionare un laboratorio e le attrezzature. Il form include una logica di validazione che impedisce la prenotazione di risorse già occupate nella stessa fascia oraria, soddisfacendo il requisito di verifica dei conflitti.
- **Partecipazione ai Progetti (RF9, RF10):** Gli studenti possono iscriversi a un progetto tramite la *partecipa\_progetto\_view*. Questa funzione controlla che l'utente non sia già iscritto e, soprattutto, verifica che il numero massimo di partecipanti (*max\_posti*) non sia stato raggiunto, impedendo ulteriori iscrizioni.
- **Gestione Attrezzature (RF12):** I tecnici responsabili di un laboratorio possono aggiungere nuove attrezzature (*aggiungi\_attrezzatura\_view*) e modificarne lo stato (es. "In manutenzione") tramite la *modifica\_stato\_attrezzatura\_view*.
- **Logout (RF11):** La funzione *logout\_view* permette a tutti gli utenti di terminare la propria sessione in modo sicuro, cancellando tutti i dati dalla sessione del server.

## Sviluppi futuri

Per completare la piattaforma UniLabManager sono stati pianificati i seguenti sviluppi, basati sui requisiti funzionali identificati nella fase di analisi ma non ancora implementati:

- **Modifica di Progetti ed Esperimenti (RF13):** Verrà introdotta la possibilità per i professori di modificare i dettagli dei progetti e degli esperimenti già creati.
- **Sistema di Notifiche Automatiche (RF14):** Si prevede di implementare un sistema di notifiche per informare gli utenti su eventi rilevanti che li riguardano. Esempi di notifiche includono:
  - Un professore riceve una notifica quando uno studente si iscrive a un suo progetto.

- Uno studente viene notificato se un esperimento a cui partecipa viene modificato o annullato.
- Un professore viene avvisato se un'attrezzatura che ha prenotato viene messa "In manutenzione" da un tecnico.
- **Caricamento Risultati e Feedback per gli Esperimenti (RF15):** Sarà sviluppata una funzionalità che permetterà agli studenti di caricare file (es. report in PDF, fogli di calcolo) per ogni esperimento a cui hanno partecipato. A loro volta, i professori avranno un'interfaccia per visualizzare i file caricati, lasciare un commento di feedback e segnare l'esperimento come "Completato" per quello specifico studente.
- **Visualizzazione Calendario Prenotazioni (RF16):** Per migliorare la gestione e la visualizzazione delle risorse, verrà implementata una vista a calendario. Questo permetterà una consultazione più intuitiva delle prenotazioni. In particolare:
- **Gestione Profilo Utente Personale (RF17):** Verrà creata una pagina "Il Mio Profilo" accessibile a ogni utente autenticato. In questa sezione, l'utente potrà non solo visualizzare i propri dati personali ma anche eseguire operazioni di modifica, come l'aggiornamento del proprio indirizzo email o il cambio della password, attraverso appositi form sicuri

## Sicurezza implementata

Sono state implementate le seguenti misure per proteggere l'applicazione e i dati degli utenti:

- **Gestione Sicura delle Password:** Le password non sono mai salvate in chiaro. Durante la registrazione, la funzione *make\_password* di Django, richiamata nel metodo *save* dei form di registrazione (es. *BaseUserRegistrationForm*), esegue l'hashing della password con l'algoritmo **PBKDF2** e **SHA256** prima di salvarla nel database. La verifica durante il login avviene tramite la funzione *check\_password*.
- **Protezione da CSRF (Cross-Site Request Forgery):** Tutti i form che eseguono operazioni di modifica (login, registrazione, creazione di progetti) sono protetti tramite l'uso del tag `{% csrf_token %}` nei template html. Questo meccanismo, garantisce che le richieste provengano effettivamente dall'applicazione e non da siti esterni malevoli.
- **Controllo degli Accessi e Autorizzazione:** L'accesso alle varie funzionalità è strettamente controllato in base al ruolo e alla proprietà delle risorse:

1. **Controllo basato sul ruolo:** Ogni vista controlla il ruolo dell'utente memorizzato in sessione (es. *if request.session.get('ruolo') != 'Professore'*) prima di eseguire qualsiasi operazione, reindirizzando al login in caso di accesso non autorizzato.
2. **Controllo basato sulla proprietà:** Per operazioni critiche come l'eliminazione di un progetto, il sistema verifica che l'utente che effettua la richiesta sia l'effettivo proprietario della risorsa. Ad esempio, in *elimina\_progetto\_view*, viene controllato che la matricola del docente associato al progetto coincida con quella salvata in sessione (*if progetto.docente.matricola != request.session.get('matricola')*).

## Analisi di vulnerabilità

### 1) Manipolazione degli URL

Una delle vulnerabilità più comuni nelle applicazioni web si verifica quando un'applicazione espone un riferimento diretto a un oggetto (come un ID nel database) in un URL, senza effettuare adeguati controlli di autorizzazione.

Questo può permettere a un utente di accedere, modificare o eliminare dati di altri utenti semplicemente manipolando l'URL.

Nel progetto è stata creata una vista appositamente vulnerabile, *elimina\_progetto\_view\_vulnerabile*, per dimostrare questo attacco.

### Codice vulnerabile (*views.py*)

```
#CODICE CHE PERMETTE UN ATTACCO DI TIPO MANIPOLAZIONE URL
<%=/progetto/{progetto_id}/elimina-vulnerabile/
def elimina_progetto_view_vulnerabile(request, progetto_id): 1 usage 2 Andrea0333
    progetto = get_object_or_404(ProgettoSperimentale, id_progetto=progetto_id)

    #DISATTIVIAMO LA PROTEZIONE
    # if progetto.docente.matricola != request.session.get('matricola'):
    #     messages.error(request, "Non sei autorizzato a eseguire questa operazione.")
    #     return redirect('dashboard_professore')

    if request.method == 'POST':
        progetto.delete()
        messages.success(request, message=f"Il progetto '{progetto.titolo}' è stato eliminato tramite manipolazione URL.")
        return redirect('dashboard_professore')

    # SENZA IL CONTROLLO, LA VISTA PROCEDE E MOSTRA LA PAGINA DI CONFERMA
    return render(request, template_name='professore/conferma_eliminazione.html', context={'progetto': progetto})
```



## Scenario di attacco

1. Un professore (Professore A) con matricola *P12345* vuole eliminare un suo progetto, che ha *id\_progetto=3*. L'URL per farlo è *http://127.0.0.1:8000/progetto/3/elimina-vulnerabile/*.
2. Un altro professore (Professore B) con matricola *P23456* intuisce la struttura dell'URL.
3. Il Professore B può tentare di eliminare il progetto del Professore A semplicemente visitando l'URL *http://127.0.0.1:8000/progetto/3/elimina-vulnerabile/* e confermando l'operazione. Poiché la vista *vulnerabile* non controlla se l'utente loggato (*P23456*) è il proprietario del progetto (*id\_progetto=3*).

## Soluzione e mitigazione del rischio

La vulnerabilità viene risolta introducendo il controllo di proprietà dell'oggetto, come implementato nella vista *elimina\_progetto\_view*:

```
<%=/progetto/{progetto_id}/elimina/
def elimina_progetto_view(request, progetto_id): 1 usage  ⚡ Andrea0333 *

    #solo professori autenticati
    if not request.session.get('is_authenticated') or request.session.get('ruolo') != 'Professore':
        return redirect('login')

    progetto = get_object_or_404(ProgettoSperimentale, id_progetto=progetto_id)

    #il professore può eliminare solo i propri progetti importante per evitare manipolazioni url
    if progetto.docente.matricola != request.session.get('matricola'):
        messages.error(request, message: "Non sei autorizzato a eseguire questa operazione.")
        return redirect('dashboard_professore')

    if request.method == 'POST':
        progetto.delete()
        return redirect('dashboard_professore')

    context = {
        'progetto': progetto
    }
    return render(request, template_name: 'professore/conferma_eliminazione.html', context)
```

## 2) SQL injection

Questo tipo di attacco si verifica quando un utente malintenzionato riesce a inserire comandi SQL all'interno di un campo di input (come un form di login), manipolando le query eseguite dal server per accedere, modificare o distruggere dati in modo non autorizzato. Per dimostrare questa vulnerabilità e, soprattutto, l'efficacia delle contromisure adottate in *UniLabManager*, è stata creata una funzione di login appositamente vulnerabile.

### Codice vulnerabile (*views.py*)

```
<< //login-vulnerabile/
def login_vulnerabile_view(request): 1 usage  ⤴ Andrea0333 *
    if request.method == 'POST':
        matricola = request.POST.get('matricola')
        password = request.POST.get('password')

        # QUERY SQL VULNERABILE
        # L'input dell'utente viene inserito direttamente nella stringa della query.
        query = f"SELECT * FROM gestionelaboratoriuniversitari utente WHERE matricola = '{matricola}' AND password = '{password}'"

        with connection.cursor() as cursor:
            cursor.execute(query)
            utente = cursor.fetchone()

        if utente:
            return redirect('dashboard_professore')
        else:
            messages.error(request, message="Credenziali non valide.")
            return redirect('login_vulnerabile')

    return render(request, template_name='login_vulnerabile.html')
```



### Accesso Vulnerabile (Solo Dimostrazione)

Questa pagina è intenzionalmente vulnerabile a SQL Injection.

Matricola:

Password:

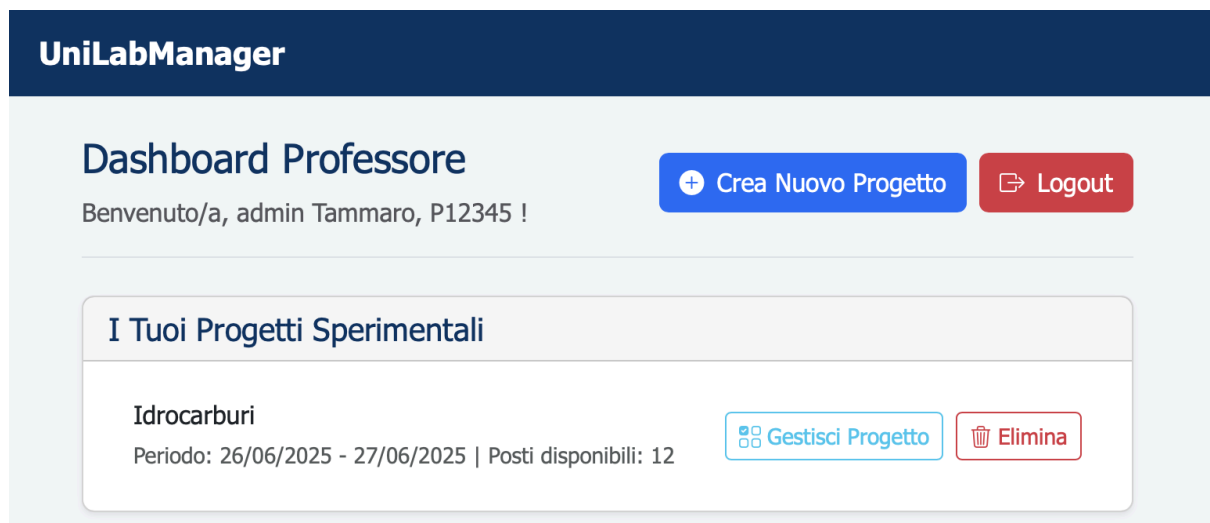
Accedi

**Scenario di attacco manuale:** un utente può facilmente bypassare l'autenticazione inserendo **' OR 1=1 --**. La query che verrà eseguita dal server è:

**SELECT \* FROM gestionelaboratoriuniversitari\_utente WHERE matricola = '' OR 1=1 -- ' AND password = 'qualsiasi\_cosa'**

**'OR 1=1'**: Questa condizione è **sempre vera**, quindi la clausola **WHERE** restituirà un risultato (il primo utente della tabella).

--: In SQL, questo è un commento. Tutto ciò che segue (**AND password = ...**) viene ignorato dal database. Infatti accediamo alla dashboard del primo utente in tabella, in questo caso un professore.



## Soluzione e mitigazione

La vulnerabilità descritta è prevenuta nel progetto grazie all'uso dell'**Object-Relational Mapper (ORM) di Django**, come implementato nella vista **login\_view** sicura.

```
def login_view(request):
    if request.method == "POST":
        matricola = request.POST.get('matricola')
        password = request.POST.get('password')

        if not matricola or not password:
            return render(request, 'login.html', context={'error_message': "Matricola e password sono obbligatorie."})

        try:
            utente = Utente.objects.get(matricola=matricola)

            if check_password(password, utente.password):
                # Credenziali valide, salviamo i dati in sessione
                request.session['matricola'] = utente.matricola
                request.session['ruolo'] = utente.ruolo
                request.session['is_authenticated'] = True

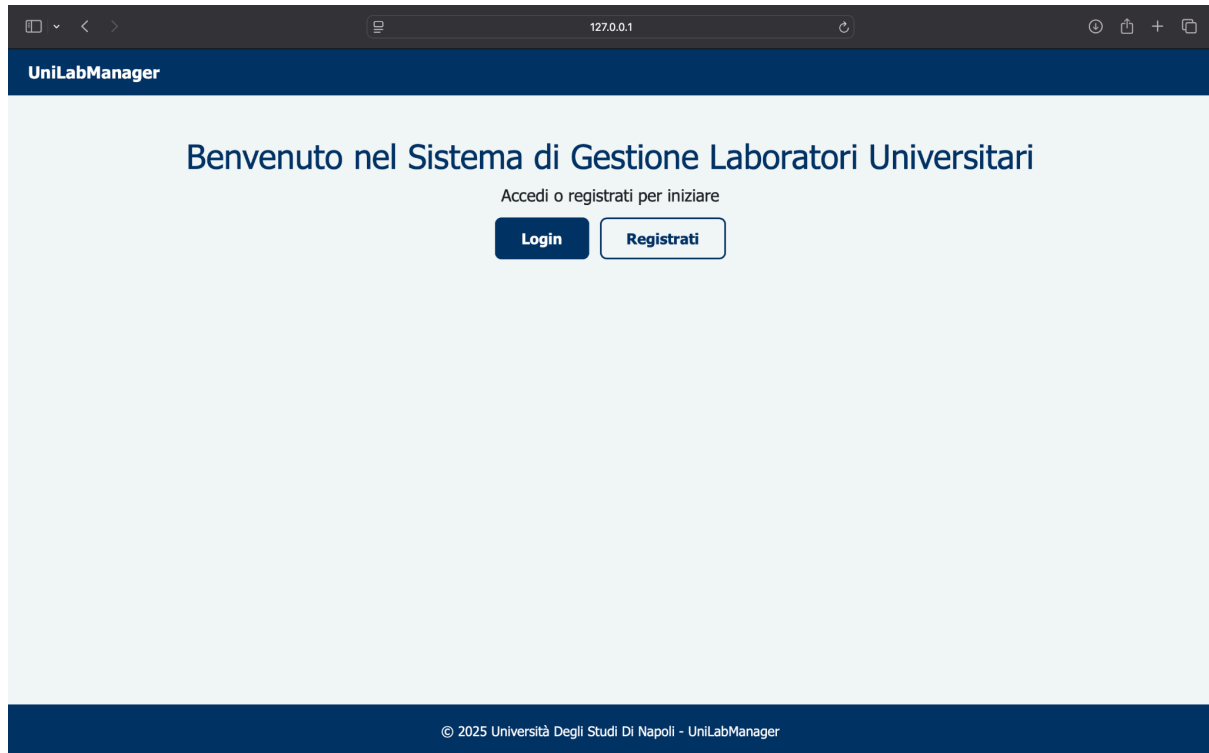
                # indirizzamento basato sul ruolo dell'utente
                if utente.ruolo == 'Professore':
                    return redirect('dashboard_professore')
                elif utente.ruolo == 'Studente':
                    return redirect('dashboard_studente')
                elif utente.ruolo == 'Tecnico':
                    return redirect('dashboard_tecnico')
            else:
                return render(request, 'login.html', context={'error_message': "Credenziali non valide, riprova."})

        except Utente.DoesNotExist:
            return render(request, 'login.html', context={'error_message': "Credenziali non valide, riprova."})

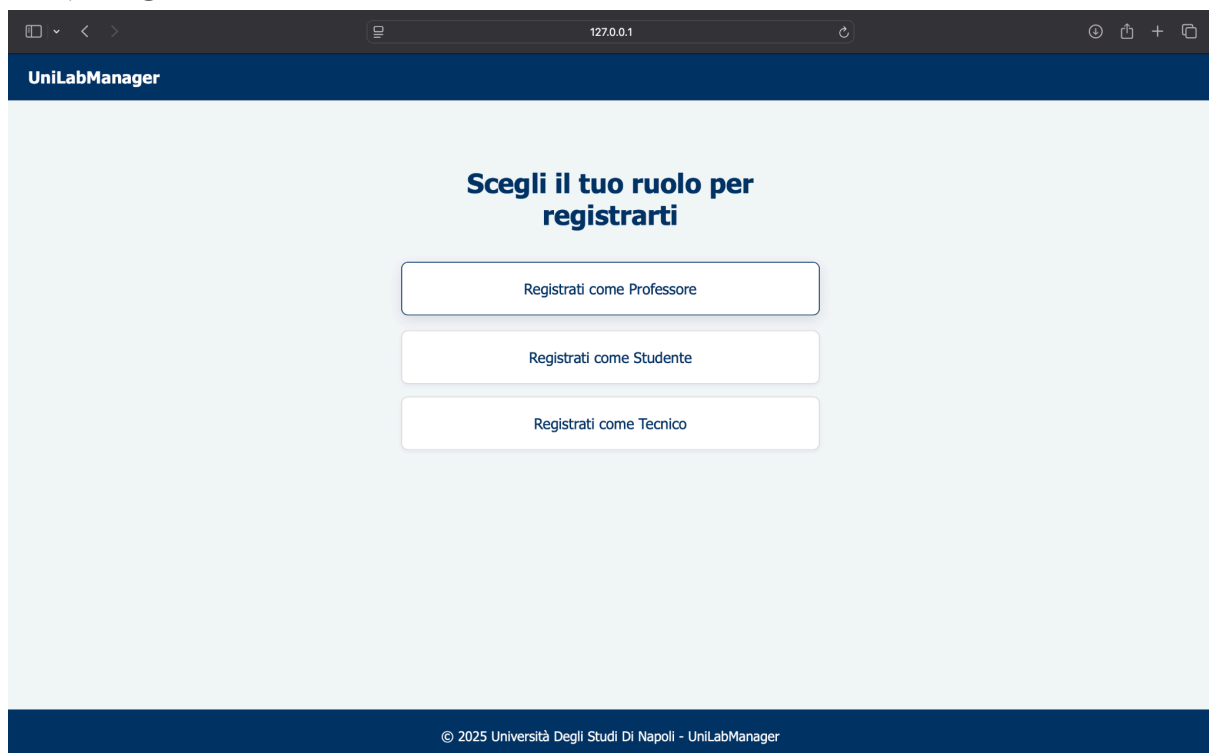
    return render(request, 'login.html')
```

## Interfacce utente

### 1) Homepage



### 2) Registrazione



### 3) Registrazione professore

The screenshot shows a web browser window with the address bar displaying '127.0.0.1'. The page title is 'UniLabManager'. The main content area is titled 'Registrazione Professore' and contains a registration form. The form fields are: Matricola (P34567), Nome (Mario), Cognome (Rossi), Email (rossimario@gmail.it), Password (masked with dots), Conferma Password (masked with dots), Dipartimento (Ingegneria), and Materia (Fisica). At the bottom of the form are two buttons: 'Registrati' and '← Indietro'. The footer of the page reads '© 2025 Università Degli Studi Di Napoli - UniLabManager'.

UniLabManager

#### Registrazione Professore

Matricola: P34567

Nome: Mario

Cognome: Rossi

Email: rossimario@gmail.it

Password: ••••••••

Conferma Password: ••••••••

Dipartimento: Ingegneria

Materia: Fisica

[Registrati](#) [← Indietro](#)

© 2025 Università Degli Studi Di Napoli - UniLabManager

### 4) Dashboard professore

The screenshot shows a web browser window with the address bar displaying '127.0.0.1'. The page title is 'UniLabManager'. The main content area is titled 'Dashboard Professore' and includes a welcome message 'Benvenuto, Mario Rossi, P34567 !'. There are two buttons at the top right: 'Crea Nuovo Progetto' and 'Logout'. Below this is a section titled 'I Tuoi Progetti Sperimentali' which contains two project entries. Each entry has a title, a period, and the number of available posts. For each project, there are two buttons: 'Gestisci Progetto' and 'Elimina'. The footer of the page reads '© 2025 Università Degli Studi Di Napoli - UniLabManager'.

UniLabManager

#### Dashboard Professore

Benvenuto, Mario Rossi, P34567 !

[Crea Nuovo Progetto](#) [Logout](#)

##### I Tuoi Progetti Sperimentali

Verifica Legge di Ohm	<a href="#">Gestisci Progetto</a> <a href="#">Elimina</a>
Accelerazione gravitazionale	<a href="#">Gestisci Progetto</a> <a href="#">Elimina</a>

© 2025 Università Degli Studi Di Napoli - UniLabManager

## 5) Dashboard studente

UniLabManager

Dashboard Studente

Benvenuto/a, Francesca Rinaldi, S34567 !

Logout

I Miei Progetti

Elenco dei progetti a cui sei attualmente iscritto.

Verifica Legge di Ohm (Docente: Rossi)

Visualizza Progetto

Progetti Disponibili

Sfoglia i progetti disponibili e richiedi la partecipazione.

Idrocarburi (Docente: Tammaro)

Visualizza e Partecipa

Accelerazione gravitazionale (Docente: Rossi)

Visualizza e Partecipa

© 2025 Università Degli Studi Di Napoli - UniLabManager

## 6) Dashboard tecnico

Dashboard Tecnico

Benvenuto/a, Giovanni Breccia, T34567 !

Logout

Gestione Attrezzature - Laboratorio: Laboratorio di Fisica

Aggiungi Attrezzatura

Tipo	Marca	Stato Attuale	Azione
Alimentatore a corrente continua	ScienzaItalia	Funzionante	Modifica Stato
Cronometro	ScienzaItalia	Funzionante	Modifica Stato
Multimetro digitale	ScienzaItalia	In manutenzione	Modifica Stato
Pendolo	ScienzaItalia	Non disponibile	Modifica Stato
Sfera Metallica	ScienzaItalia	Funzionante	Modifica Stato

Prenotazioni Laboratorio Imminenti

Data: 03/07/2025 | Orario: 09:30 - 11:30 | Professore: Mario Rossi | Esperimento: Studio di un Circuito in Serie

Prenotazioni Attrezzature Imminenti

Data: 03/07/2025 | Orario: 09:30 - 11:30 | Professore: Mario Rossi | Attrezzatura: Alimentatore a corrente continua (ScienzaItalia)

## 7) Dettagli progetto professore

UniLabManager

← Torna alla Dashboard Professore

Progetto: Verifica Legge di Ohm

Docente: Mario Rossi

**Descrizione:** L'esperimento si concentra sulla verifica sperimentale delle leggi fondamentali dell'elettricità. Gli studenti utilizzeranno una breadboard, resistori di diverso valore e un alimentatore a corrente continua per costruire semplici circuiti. Mediante l'uso di multimetri digitali, verranno effettuate misurazioni precise di tensione (V), corrente (I) e resistenza (R) per analizzare le relazioni che li legano.

**Obiettivi:** - Verificare sperimentalmente la validità della Legge di Ohm ( $V=IR$ ) per un resistore. - Misurare la resistenza equivalente di due o più resistori collegati in serie. - Misurare la resistenza equivalente di due o più resistori collegati in parallelo. - Acquisire manualità con la strumentazione di base di un laboratorio di fisica (multimetro, alimentatore DC, breadboard).

Aggiungi Esperimento

Esperimenti Associati

Studio di un Circuito in Serie

Questo esperimento si concentra sull'analisi delle proprietà dei circuiti in serie. Gli studenti assembleranno un circuito contenente due o più resistori collegati in serie su una breadboard. Verranno effettuate misurazioni della resistenza totale, della corrente che fluisce nel circuito e della caduta di potenziale ai capi di ogni singolo resistore

Data: 03/07/2025 | Orario: 09:30 - 11:30 | Laboratorio: Laboratorio di Fisica

Elimina

© 2025 Università Degli Studi Di Napoli - UniLabManager

## 8) Dettagli progetto studente

UniLabManager

← Torna alla Dashboard Studente

Progetto: Verifica Legge di Ohm

Docente: Mario Rossi

**Descrizione:** L'esperimento si concentra sulla verifica sperimentale delle leggi fondamentali dell'elettricità. Gli studenti utilizzeranno una breadboard, resistori di diverso valore e un alimentatore a corrente continua per costruire semplici circuiti. Mediante l'uso di multimetri digitali, verranno effettuate misurazioni precise di tensione (V), corrente (I) e resistenza (R) per analizzare le relazioni che li legano.

**Obiettivi:** - Verificare sperimentalmente la validità della Legge di Ohm ( $V=IR$ ) per un resistore. - Misurare la resistenza equivalente di due o più resistori collegati in serie. - Misurare la resistenza equivalente di due o più resistori collegati in parallelo. - Acquisire manualità con la strumentazione di base di un laboratorio di fisica (multimetro, alimentatore DC, breadboard).

Sei già iscritto a questo progetto. Ora puoi partecipare ai singoli esperimenti.

Esperimenti Associati

Studio di un Circuito in Serie

Questo esperimento si concentra sull'analisi delle proprietà dei circuiti in serie. Gli studenti assembleranno un circuito contenente due o più resistori collegati in serie su una breadboard. Verranno effettuate misurazioni della resistenza totale, della corrente che fluisce nel circuito e della caduta di potenziale ai capi di ogni singolo resistore

Data: 03/07/2025 | Orario: 09:30 - 11:30 | Laboratorio: Laboratorio di Fisica

Partecipa

31

## 9) Esperimento e prenotazione laboratorio/attrezzatura

127.0.0.1

Aggiungi Esperimento al Progetto: Verifica Legge di Ohm

Titolo: Studio di un Circuito in s

Questo esperimento si concentra sull'analisi delle proprietà dei circuiti in serie. Gli studenti assembleranno un circuito contenente due o più resistori collegati in serie su una breadboard. Verranno effettuate misurazioni della resistenza totale, della corrente che fluisce nel circuito e della caduta di potenziale ai capi di ogni singolo resistore

Descrizione:

- Costruire un circuito con due resistori in serie.  
- Misurare la resistenza totale e verificare sperimentalmente che  $R_{eq} = R_1 + R_2$ .  
- Verificare che la corrente sia la stessa in ogni punto di un circuito in serie.  
- Misurare la caduta di potenziale su ogni resistore e confermare che la loro somma è uguale alla tensione totale

Obiettivi:

Laboratorio: Laboratorio di Fisica

Attrezzature:  
☐ Microscopio  
☐ Sfera Metallica  
☐ Cronometro  
☒ Alimentatore a corrente continua

Data prenotazione: 03/07/2025

Ora inizio: 09:30

Ora fine: 11:30

Salva Esperimento e Prenota

Annulla

32