

---

# SISTEMA DI REGISTRAZIONE E AUTENTICAZIONE

---

GRUPPO 6

2022/2023

RAFFAELE D'AMBROSIO, SOFIA DELLA PENNA, LORENZO PARRACINO, ANGELO FRANCESCO TUORTO  
UNIVERSITÀ DEGLI STUDI DI NAPOLI FEDERICO II – CORSO DI SOFTWARE ARCHITECTURE DESIGN

# Sommario

---

<b>1. INTRODUZIONE .....</b>	<b>3</b>
<b>2. DOCUMENTAZIONE DI ANALISI .....</b>	<b>4</b>
<b>2.1 Documento dei Requisiti .....</b>	<b>4</b>
2.1.1 Requisiti Funzionali .....	4
2.1.2 Requisiti Non Funzionali .....	4
2.1.3 Vincoli .....	5
<b>2.2 Diagramma di Contesto .....</b>	<b>5</b>
<b>2.3 System Domain Model.....</b>	<b>6</b>
<b>2.4 Storie Utente.....</b>	<b>6</b>
<b>2.5 Casi d'Uso .....</b>	<b>7</b>
<b>2.6 Diagramma di Stato .....</b>	<b>10</b>
<b>2.7 Diagramma di Attività.....</b>	<b>12</b>
<b>2.8 Diagrammi di Sequenza .....</b>	<b>15</b>
<b>3. DOCUMENTAZIONE DI PROGETTO .....</b>	<b>18</b>
<b>3.1 Diagramma dei Package .....</b>	<b>20</b>
<b>3.2 Diagramma delle Classi.....</b>	<b>21</b>
<b>3.3 Diagramma di Deploy .....</b>	<b>22</b>
<b>3.4 Diagramma C&amp;C.....</b>	<b>23</b>
<b>3.5 Install View .....</b>	<b>24</b>
<b>3.6 Diagrammi di Sequenza Raffinati .....</b>	<b>25</b>
<b>4. GUIDA ALL'UTILIZZO .....</b>	<b>27</b>
<b>4.1 Dipendenze .....</b>	<b>27</b>
<b>4.2 Installazione .....</b>	<b>27</b>
4.2.1 Installazione di Docker .....	27
4.2.2 Personalizzazione del Docker-compose .....	27
4.2.3 Installazione del sistema di autenticazione .....	27
4.2.3 Inizializzazione del database.....	28
<b>4.3 Configurazione .....</b>	<b>28</b>
4.3.1 Configurazione del reindirizzamento .....	28
4.3.2 Configurazione CORS .....	28
4.3.3 Configurazione del collegamento tra API e database/mail server.....	29
4.3.4 Configurazione della comunicazione con API.....	29
<b>4.4 Esempi d'Uso.....</b>	<b>29</b>
<b>4.5 Rotte API.....</b>	<b>30</b>
4.5.1 RegistraGiocatore .....	30
4.5.2 LogOut .....	30
4.5.3 LoginGiocatore.....	31
4.5.4 Redirect .....	31
4.5.5 ConfermaEmail .....	31
4.5.6 RecuperaAccountViaEmail.....	32

4.5.7 RecuperaAccountCambiaPassword.....	32
4.5.8 OttieniDatiUtente.....	33
<b>5. TESTING .....</b>	<b>34</b>
<b>5.1 Casi di Test .....</b>	<b>34</b>
5.1.1 Registrazione .....	35
5.1.2 Login.....	37
5.1.3 Password Reset .....	38
<b>6. GLOSSARIO.....</b>	<b>39</b>

## **1. INTRODUZIONE**

---

Benvenuti alla documentazione del sistema software " Man vs automated Testing Tools challenges ". Questa documentazione ha lo scopo di fornire una guida completa per comprendere e utilizzare il sistema software, che è progettato per consentire a studenti di Software Testing di fare addestramento su Task di Unit Testing.

La parte qui realizzata del sistema software " Man vs automated Testing Tools challenges " si concentra su due requisiti funzionali fondamentali: la registrazione degli studenti e l'autenticazione dei giocatori. Questi requisiti consentono agli studenti di sfruttare al massimo le funzionalità offerte dall'applicazione, garantendo una migliore esperienza di gioco e facilitando l'accesso alle informazioni riguardanti le attività svolte.

Il requisito funzionale T2, relativo alla registrazione degli studenti, offre la possibilità agli utenti di registrarsi fornendo informazioni personali come nome, cognome, indirizzo e-mail valido e password. Una volta verificata la validità dei dati forniti, il sistema aggiunge lo studente all'elenco dei giocatori registrati e gli assegna un ID univoco. Inoltre, è possibile raccogliere ulteriori informazioni sugli studenti, come il corso di studi a cui sono iscritti, per personalizzare l'esperienza di gioco e fornire requisiti più complessi.

Il requisito funzionale T3, relativo all'autenticazione dei giocatori, consente agli studenti di accedere al sistema utilizzando l'indirizzo e-mail e la password forniti durante la registrazione. Dopo aver verificato la validità dei dati, il sistema autentica il giocatore e fornisce una schermata di accesso alle funzionalità di gioco o alla consultazione delle sessioni di gioco precedenti. Questo permette agli studenti di sfruttare appieno le potenzialità del sistema e di accedere in modo sicuro alle loro informazioni di gioco.

Questa documentazione fornirà dettagli approfonditi su come utilizzare correttamente il sistema software, illustrando i passaggi necessari per registrarsi come studente, effettuare l'autenticazione e sfruttare le funzionalità di gioco disponibili. Saranno fornite istruzioni chiare e dettagliate per l'installazione e configurazione del sistema affinché si possa garantire un'esperienza utente ottimale.

## **2. DOCUMENTAZIONE DI ANALISI**

---

Il **Documento di Analisi** definisce l'approccio e i dettagli dell'analisi per il sistema di autenticazione che sarà sviluppato. Lo scopo di questa analisi è comprendere a fondo le esigenze degli utenti e degli stakeholder, identificare i requisiti funzionali e non funzionali e definire una solida base per la progettazione e lo sviluppo del sistema di autenticazione.

### **2.1 Documento dei Requisiti**

---

Il **Documento dei Requisiti** definisce le specifiche e le esigenze per il sistema di autenticazione che sarà sviluppato. Questo sistema di autenticazione avrà il compito di gestire in modo sicuro e affidabile l'accesso degli utenti al sistema. Il sistema di autenticazione sarà implementato come parte di una piattaforma o di un'applicazione più ampia, che richiede un'adeguata gestione dell'accesso degli utenti. Sarà utilizzato dai giocatori per accedere a risorse protette, quali dati sensibili o funzionalità riservate, garantendo l'integrità, la sicurezza e la privacy dei dati. Inoltre, esso ha lo scopo di fornire una visione chiara e dettagliata dei requisiti funzionali e non funzionali del sistema di autenticazione. È destinato agli sviluppatori, ai progettisti e agli stakeholder coinvolti nel progetto, al fine di garantire una comprensione comune delle funzionalità richieste e delle aspettative nei confronti del sistema.

#### **2.1.1 Requisiti Funzionali**

---

Si elencano i **Requisiti Funzionali** individuati:

1. **REGISTRAZIONE:** L'applicazione deve consentire ai visitatori di registrarsi fornendo nome, cognome, e-mail, password (obbligatoriamente) ed eventualmente il corso di studi a cui sono iscritti.
2. **VALIDAZIONE:** Il sistema deve controllare la validità dei dati forniti e, in caso affermativo, aggiungere il giocatore all'elenco dei giocatori registrati, associandogli un ID univoco.
3. **LOGIN:** Un visitatore registrato, inserendo l'e-mail e la password associate ad un giocatore registrato, può accedere al sistema, diventando giocatore.
4. **CONFERMA E-MAIL:** Il giocatore, prima di avere accesso al gioco, deve confermare la registrazione attraverso un'e-mail di conferma.
5. **LOGOUT:** Il giocatore può effettuare il Logout dal sistema e diventare visitatore.
6. **RECUPERO ACCOUNT:** Un visitatore registrato può modificare la sua password di accesso.

Dal colloquio con lo stakeholder, è emerso che non è necessario visualizzare le attività di gioco del giocatore, in quanto attività designata ad altri team.

#### **2.1.2 Requisiti Non Funzionali**

---

Si elencano i **Requisiti Non Funzionali** individuati:

1. **PRESTAZIONI:** Il sistema deve poter essere utilizzato da più utenti contemporaneamente e deve essere scalabile.

2. **USABILITÀ:** Il sistema deve essere di facile apprendimento, utile e deve fornire feedback all'utente.
3. **AFFIDABILITÀ:** Il sistema deve essere tollerante agli errori e avere una capacità di recupero rapida.
4. **SICUREZZA:** Il sistema deve fornire protezione da accessi non autorizzati, attraverso un sistema di autenticazione e crittografia. Inoltre, deve limitare il traffico che può passare e identificare la fonte di qualsiasi input esterno. Infine, deve garantire l'integrità dei dati che interagiranno con il sistema.
5. **MANUTENIBILITÀ:** Il sistema deve essere modulare ed estensibile. Inoltre, il codice deve essere ben organizzato.
6. **PORTABILITÀ:** Il sistema deve poter essere eseguito su diverse piattaforme o ambienti.
7. **INTEROPERABILITÀ:** Il sistema deve essere capace di interagire con altri sistemi.

### 2.1.3 Vincoli

Si elencano i **vincoli**:

- **VINCOLI TECNICI:** È richiesto che il sistema funzioni su una macchina Windows 11, dotata di 16 GB di RAM, sfruttando un quantitativo di risorse minimo.
- **VINCOLI DI PROGETTO:** Tempistiche di progetto e di sviluppo ridotte (5 mesi).

## 2.2 Diagramma di Contesto

Il **Diagramma di Contesto** rappresenta una panoramica ad alto livello del sistema di autenticazione in analisi e delle sue interazioni con gli attori esterni. Questo diagramma fornisce una visione chiara e concisa del modo in cui il sistema di autenticazione si integra con l'ambiente circostante e interagisce con gli altri componenti del sistema.

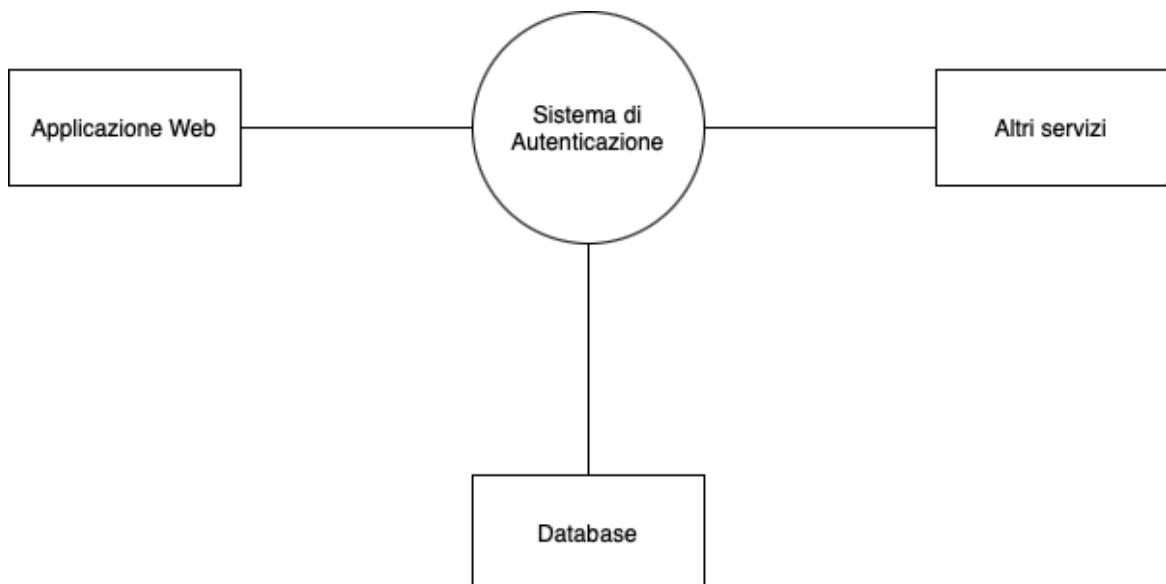


Figura 2.1: Diagramma di Contesto

## 2.3 System Domain Model

Il **System Domain Model** per un sistema di autenticazione rappresenta in modo conciso e chiaro le principali entità e le relazioni all'interno del dominio del sistema. Questo modello fornisce una visione globale delle componenti fondamentali coinvolte nella gestione dell'autenticazione degli utenti, consentendo una migliore comprensione del sistema nel suo complesso.

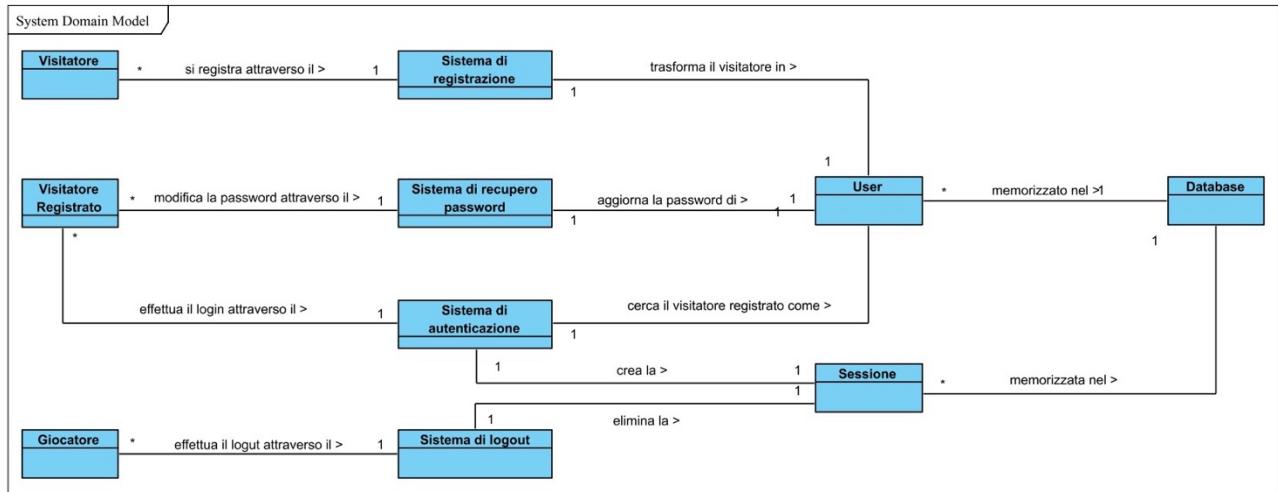


Figura 2.2: System Domain Model

## 2.4 Storie Utente

Le **Storie Utente** per il sistema di autenticazione hanno lo scopo di identificare e documentare in modo dettagliato le esigenze e le aspettative degli utenti finali. Esse descrivono le attività specifiche che gli utenti desiderano svolgere all'interno del sistema di autenticazione e offrono una guida per la progettazione e lo sviluppo delle funzionalità.

As a	VISITATORE,	I want to sign-in	So that I can login
As a	VISITATORE REGISTRATO,	I want to login	So that I can access to the game
As a	VISITATORE REGISTRATO,	I want to reset my password	So that I have the credentials to login again
As a	GIOCATORE,	I want to logout	So that I can came back to the initial page

## 2.5 Casi d'Uso

Il **Diagramma dei Casi d'Uso** rappresenta le interazioni tra gli utenti e il sistema di autenticazione, mettendo in evidenza le funzionalità e le azioni che possono essere eseguite dagli attori coinvolti.

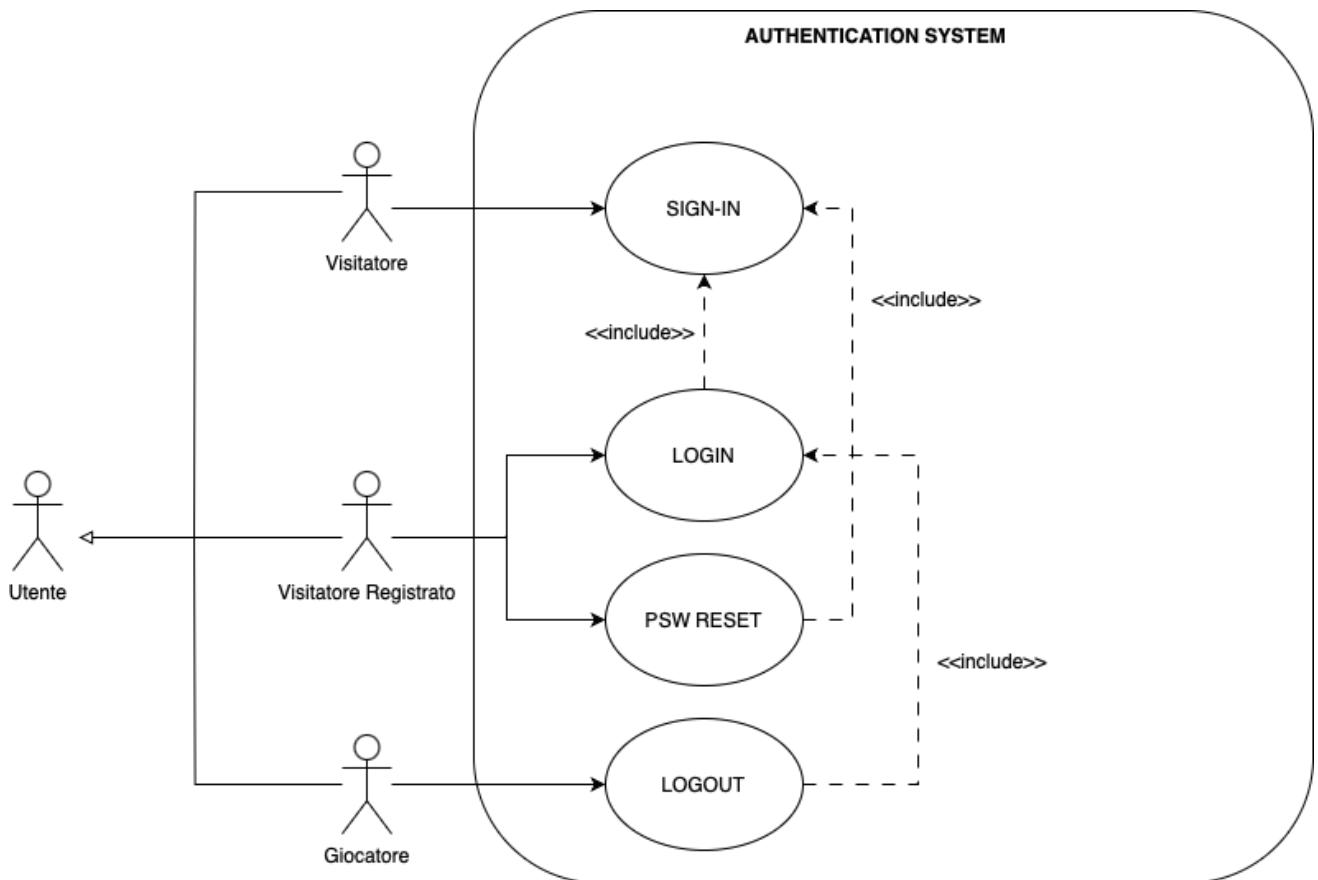


Figura 21.3: Casi d'Uso

Di seguito, sono mostrati gli **Scenari** dei Casi d'Uso:

Caso d'Uso	Registrazione
ID	SIGN-IN
Descrizione	Un visitatore si registra al sistema
Attore Primario	Visitatore
Pre-Condizioni	Nessuno
Sequenza di eventi principali	<ol style="list-style-type: none"> <li>Il caso d'uso inizia quando un visitatore decide di registrarsi al sistema</li> <li>Il sistema predisponde un'interfaccia per poter effettuare la registrazione, per cui sarà necessario compilare alcuni campi</li> <li>Il visitatore compila i campi richiesti (nome, cognome, email, password, eventualmente corso di studi)</li> </ol>

	4. Il sistema verifica la validità delle credenziali inserite dall'utente 5. Il sistema registra l'utente, assegnandogli un ID univoco 6. L'utente è correttamente registrato al sistema
<b>Post-Condizioni</b>	L'utente è correttamente registrato al sistema
<b>Sequenza di eventi alternativi</b>	Se i campi obbligatori non sono compilati o sono compilati in maniera errata, il sistema restituisce un messaggio di errore

<b>Caso d'Uso</b>	<i>Login</i>
<b>ID</b>	LOGIN
<b>Descrizione</b>	Un visitatore registrato effettua il login al sistema
<b>Attore Primario</b>	Visitatore Registrato
<b>Pre-Condizioni</b>	L'utente deve essere correttamente registrato al sistema
<b>Sequenza di eventi principali</b>	1. Il caso d'uso inizia quando un visitatore registrato decide di effettuare il login al sistema 2. Il sistema predisponde un'interfaccia per poter inserire le credenziali 3. Il visitatore registrato inserisce e-mail e password 4. Il sistema verifica la validità delle credenziali inserite dall'utente 5. L'utente è correttamente loggato al sistema
<b>Post-Condizioni</b>	L'utente è correttamente loggato al sistema
<b>Sequenza di eventi alternativi</b>	Se le credenziali non sono valide, il sistema restituisce un messaggio di errore e l'utente risulta non loggato

<b>Caso d'Uso</b>	<i>Reset Password</i>
<b>ID</b>	PSW RST
<b>Descrizione</b>	Un visitatore registrato sceglie di resettare la sua password
<b>Attore Primario</b>	Visitatore Registrato
<b>Pre-Condizioni</b>	L'utente deve essere correttamente registrato al sistema

<b>Sequenza di eventi principali</b>	<ol style="list-style-type: none"> <li>1. Il caso d'uso inizia quando un visitatore registrato decide di resettare la sua password</li> <li>2. Il sistema predispone un'interfaccia per poter inserire l'e-mail</li> <li>3. L'utente inserisce l'e-mail</li> <li>4. Il sistema predispone un'interfaccia per poter inserire la nuova password</li> <li>5. L'utente ha resettato la password con successo</li> </ol>
<b>Post-Condizioni</b>	La password è stata resettata
<b>Sequenza di eventi alternativi</b>	Se l'e-mail inserita dall'utente non è riconosciuta o se la password non soddisfa i pattern richiesti, il sistema restituisce un messaggio di errore

<b>Caso d'Uso</b>	<i>Logout</i>
<b>ID</b>	<i>LOGOUT</i>
<b>Descrizione</b>	Un giocatore decide di effettuare il logout dal sistema
<b>Attore Primario</b>	Giocatore
<b>Pre-Condizioni</b>	L'utente deve essere correttamente registrato e loggato al sistema
<b>Sequenza di eventi principali</b>	<ol style="list-style-type: none"> <li>1. Il caso d'uso inizia quando un giocatore decide di effettuare il logout dal sistema</li> <li>2. Il sistema predispone un bottone per poter effettuare il logout</li> <li>3. L'utente clicca sul bottone di logout</li> <li>4. L'utente ha effettuato il logout con successo</li> </ol>
<b>Post-Condizioni</b>	L'utente ha effettuato il logout con successo
<b>Sequenza di eventi alternativi</b>	N/A

## 2.6 Diagramma di Stato

I **Diagrammi di Stato** rappresentano il comportamento dinamico e le transizioni di stato all'interno del sistema di autenticazione. Questi diagrammi consentono di modellare lo stato del sistema e le condizioni che determinano il passaggio tra i diversi stati durante il processo di autenticazione degli utenti.

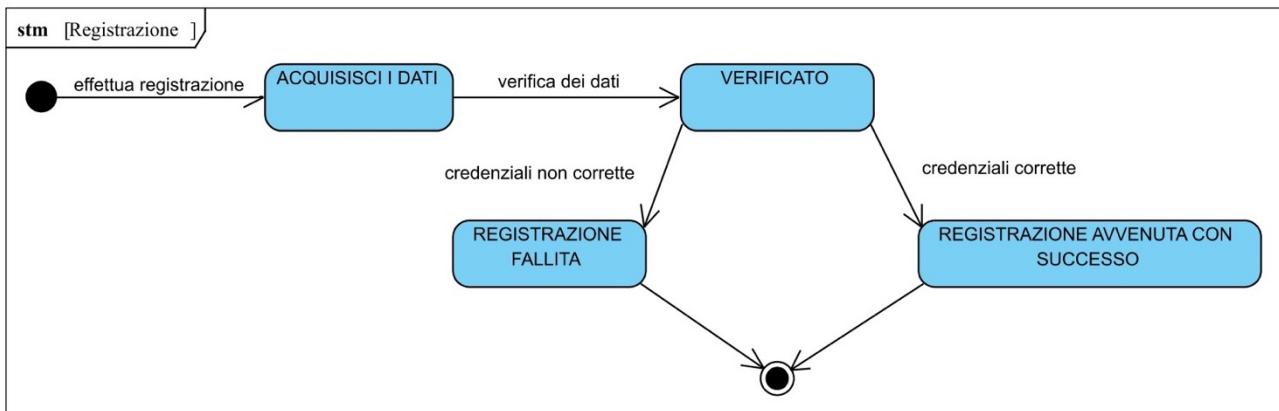


Figura 2.4: Diagramma di stato - Registrazione

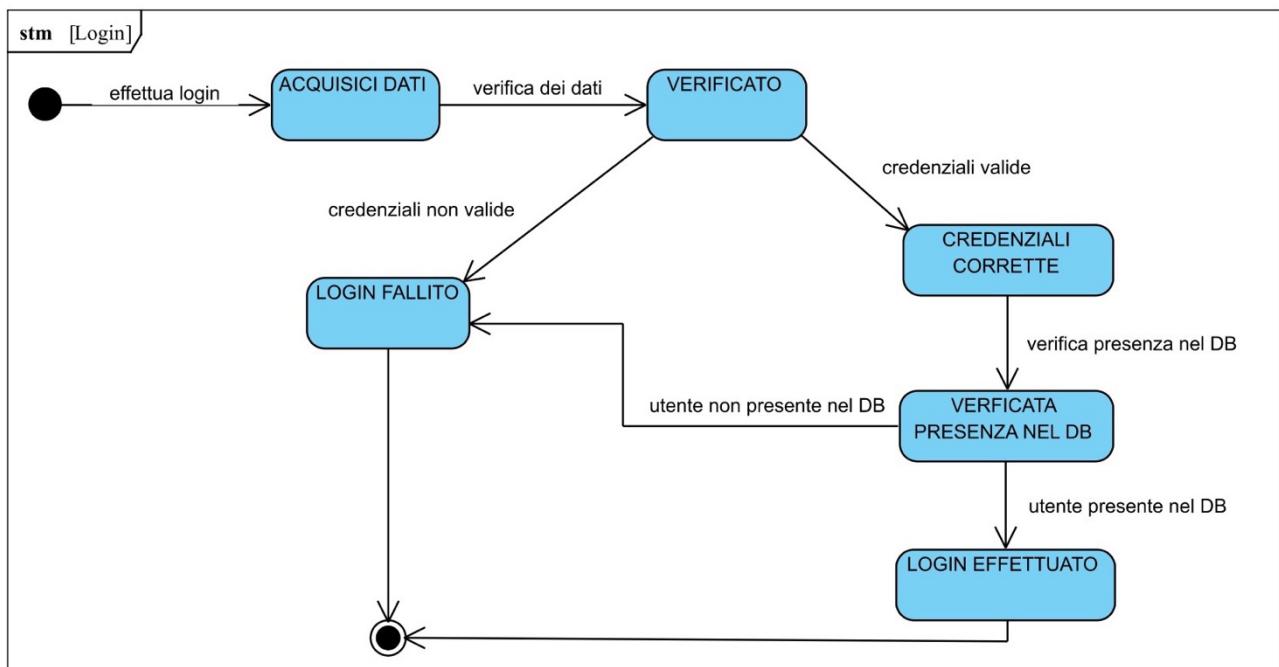


Figura 2.5: Diagramma di Stato - Login

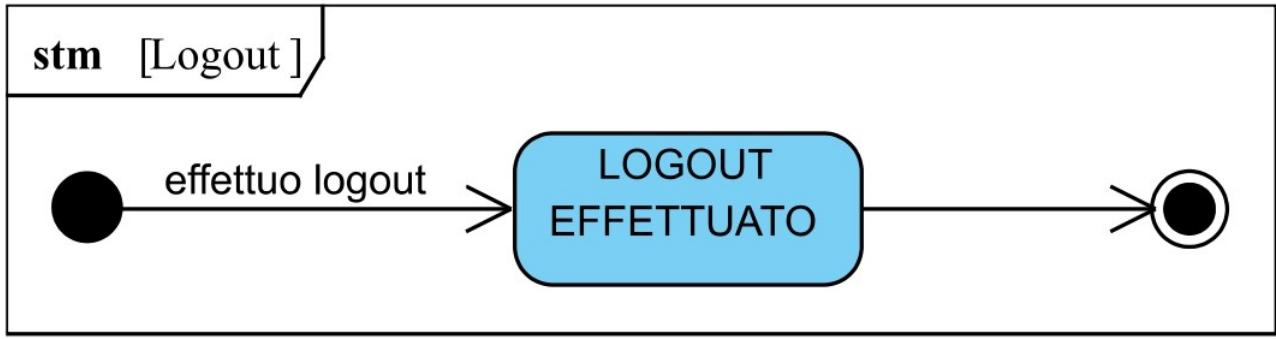


Figura 2.6: Diagramma di Stato - Logout

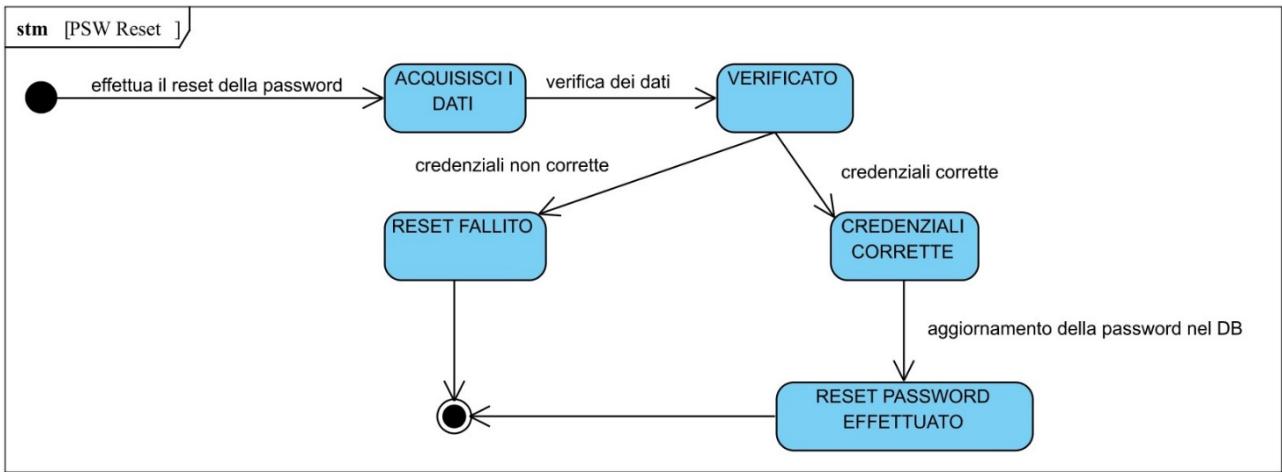


Figura 2.7: Diagramma di Stato - PSW Reset

## 2.7 Diagramma di Attività

I **Diagrammi di Attività** offrono una rappresentazione visuale del flusso di lavoro o dei processi all'interno del sistema di autenticazione. Questi diagrammi consentono di modellare le attività, le decisioni e le scelte durante il processo di autenticazione, fornendo una panoramica chiara e intuitiva delle azioni eseguite dagli utenti e dal sistema.

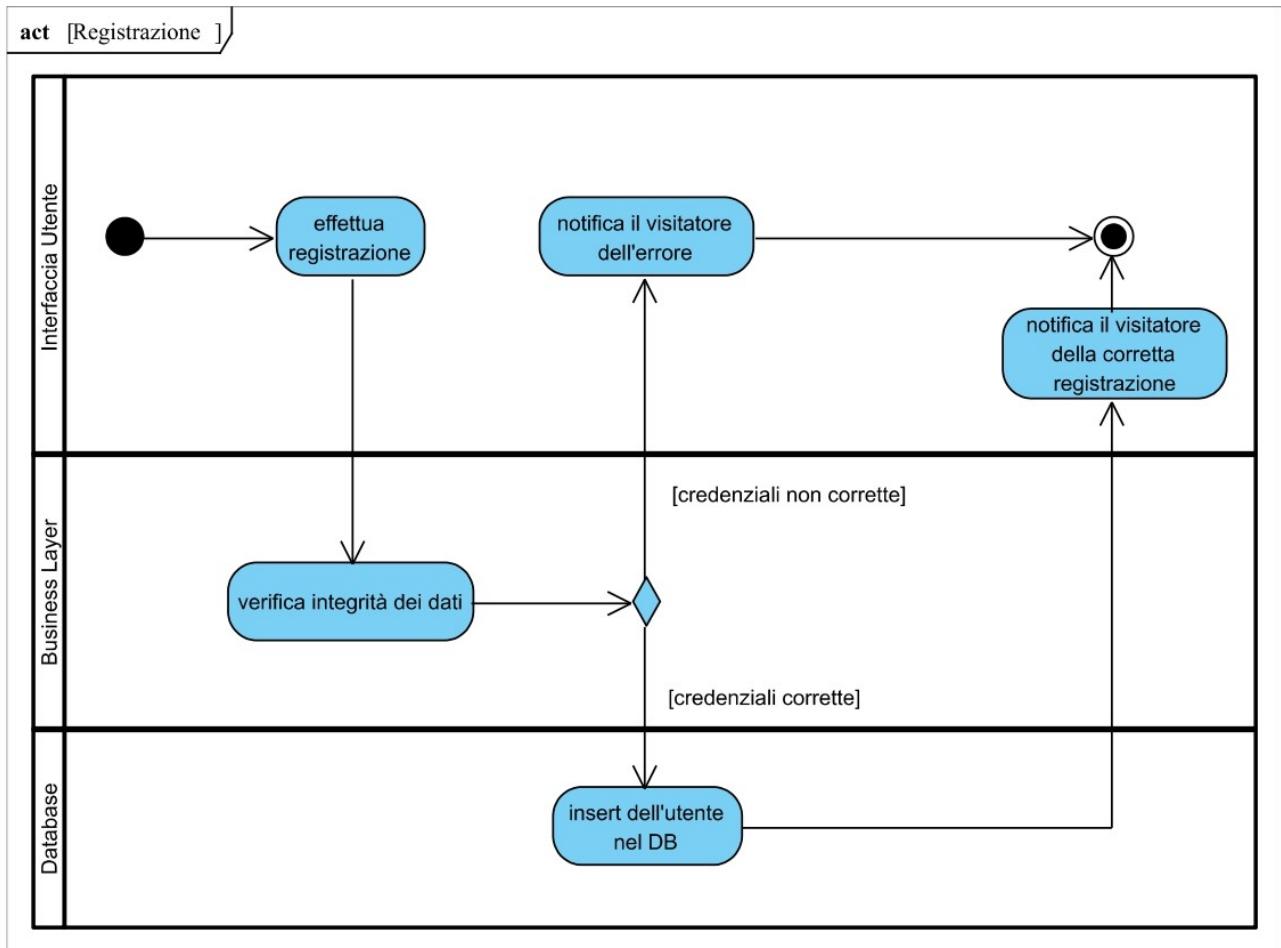


Figura 2.8: Diagramma di Attività - Registrazione

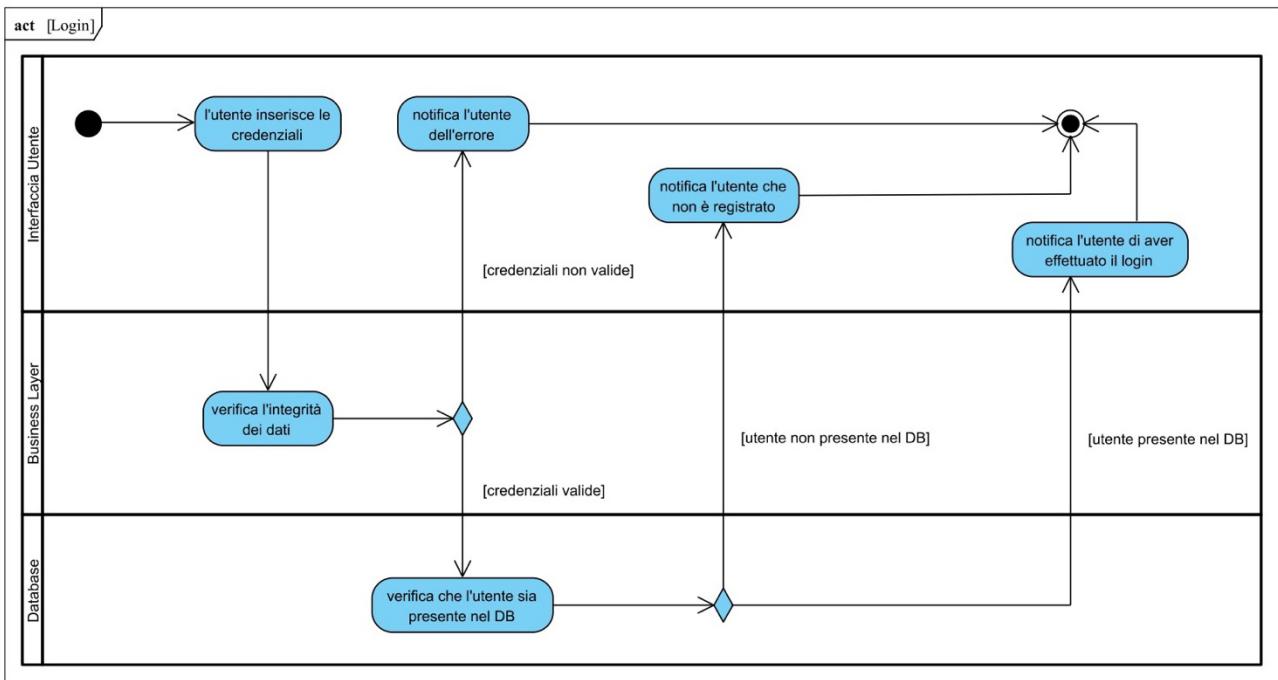


Figura 2.9: Diagramma di Attività - Login

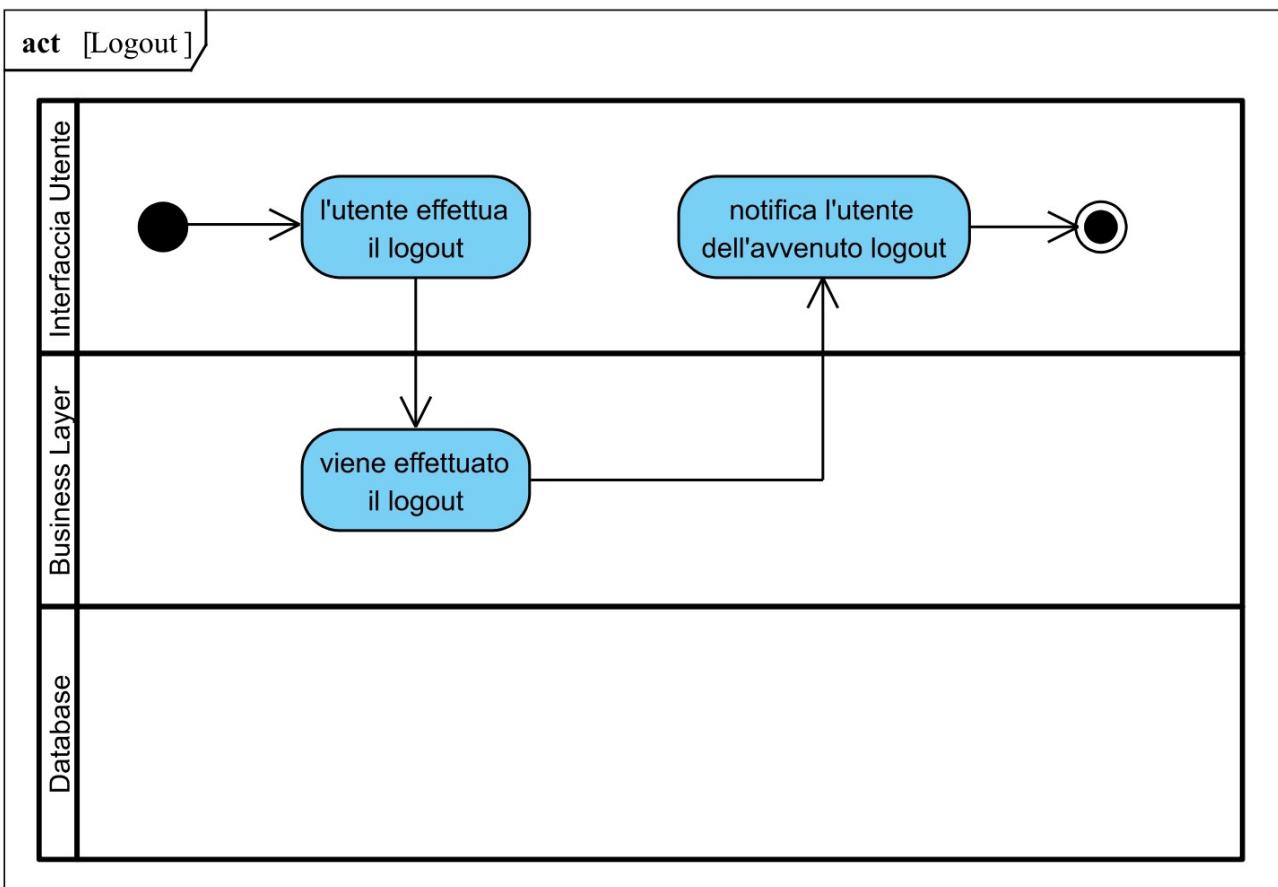


Figura 2.10: Diagramma di Attività - Logout

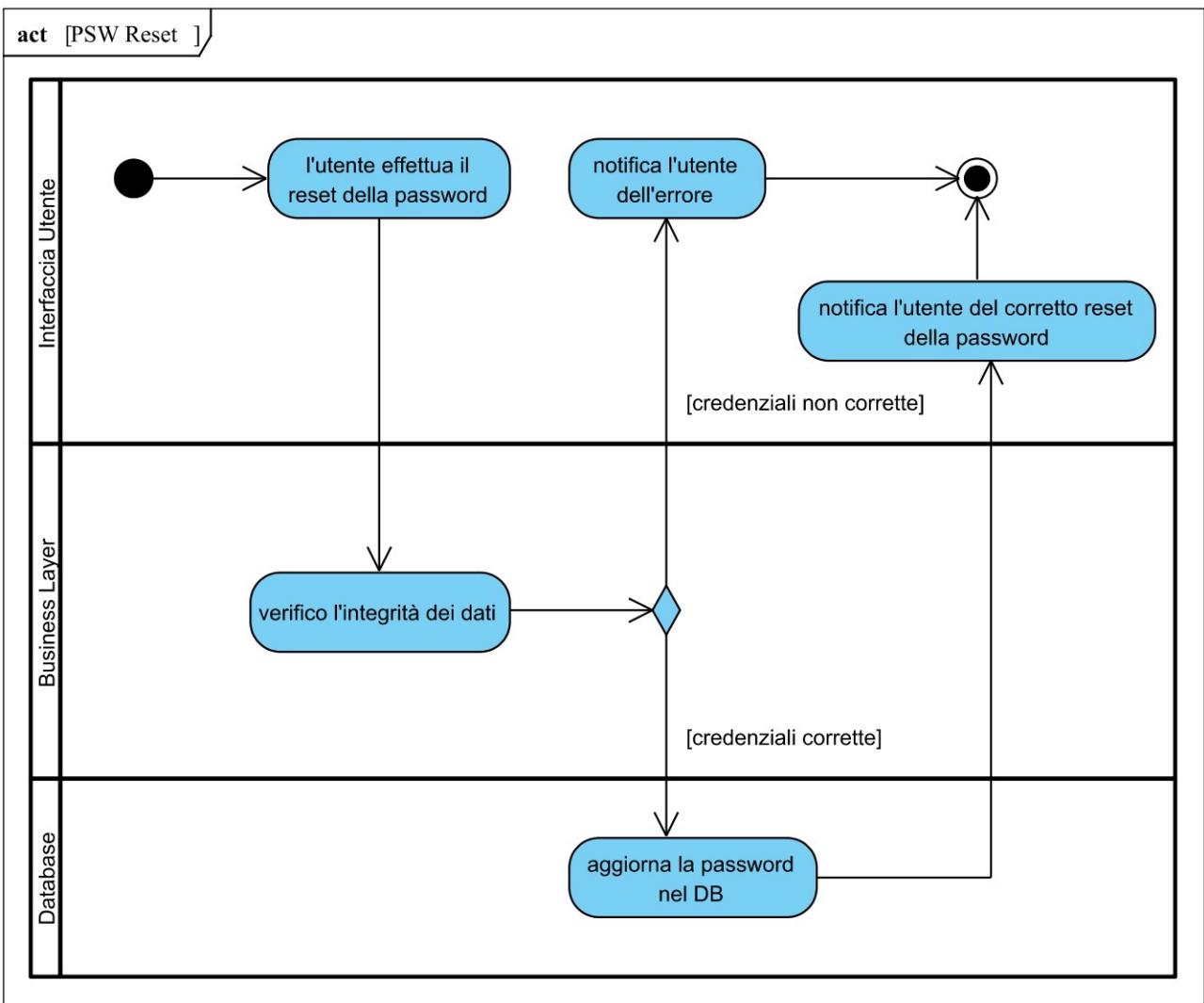


Figura 2.11: Diagramma di Attività - PSW Reset

## 2.8 Diagrammi di Sequenza

I **Diagrammi di Sequenza** rappresenta la sequenza temporale delle interazioni tra gli oggetti principali all'interno del sistema di autenticazione. Questo diagramma offre una visione dettagliata dei passaggi e delle operazioni coinvolte nel processo di autenticazione degli utenti.

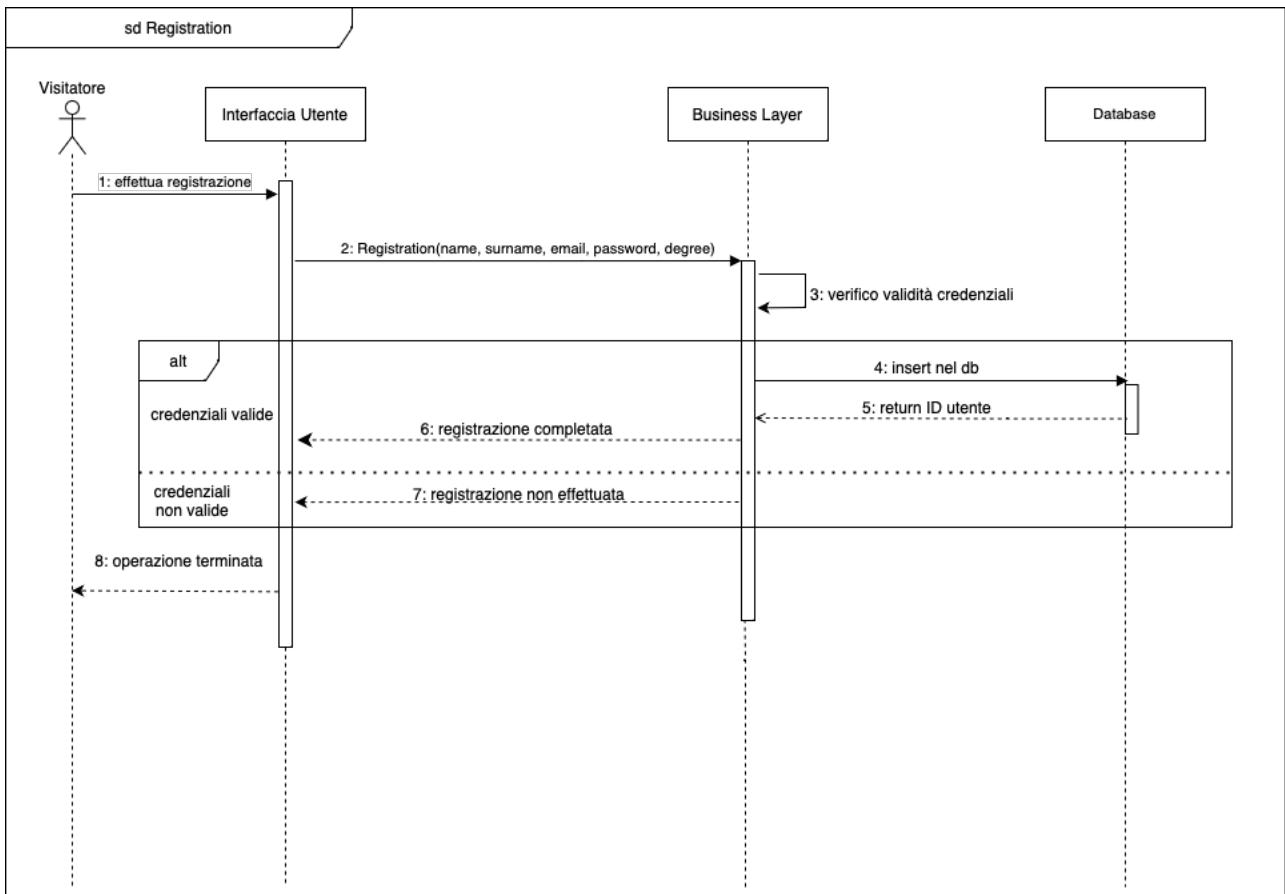


Figura 2.12: Diagramma di Sequenza - Registrazione

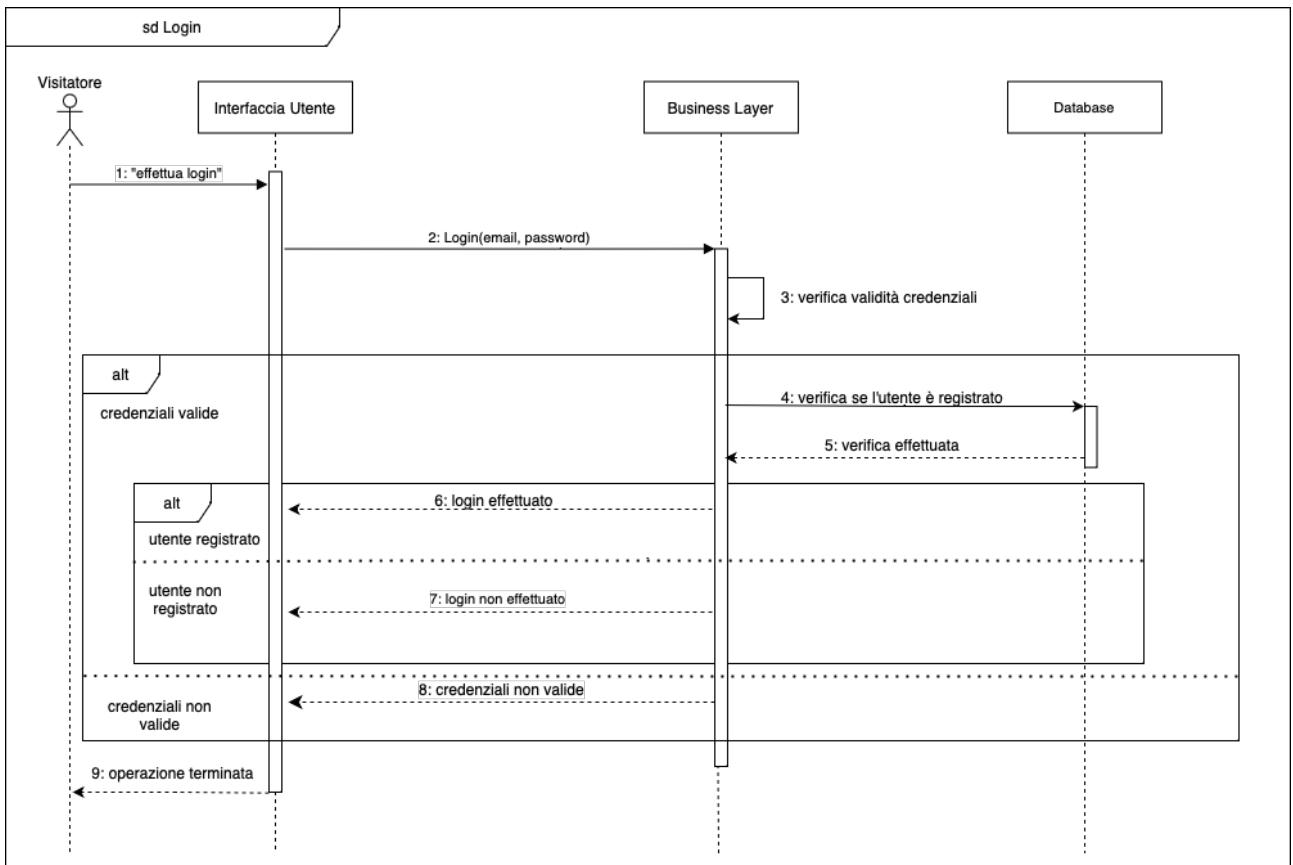


Figura 2.13: Diagramma di Sequenza - Login

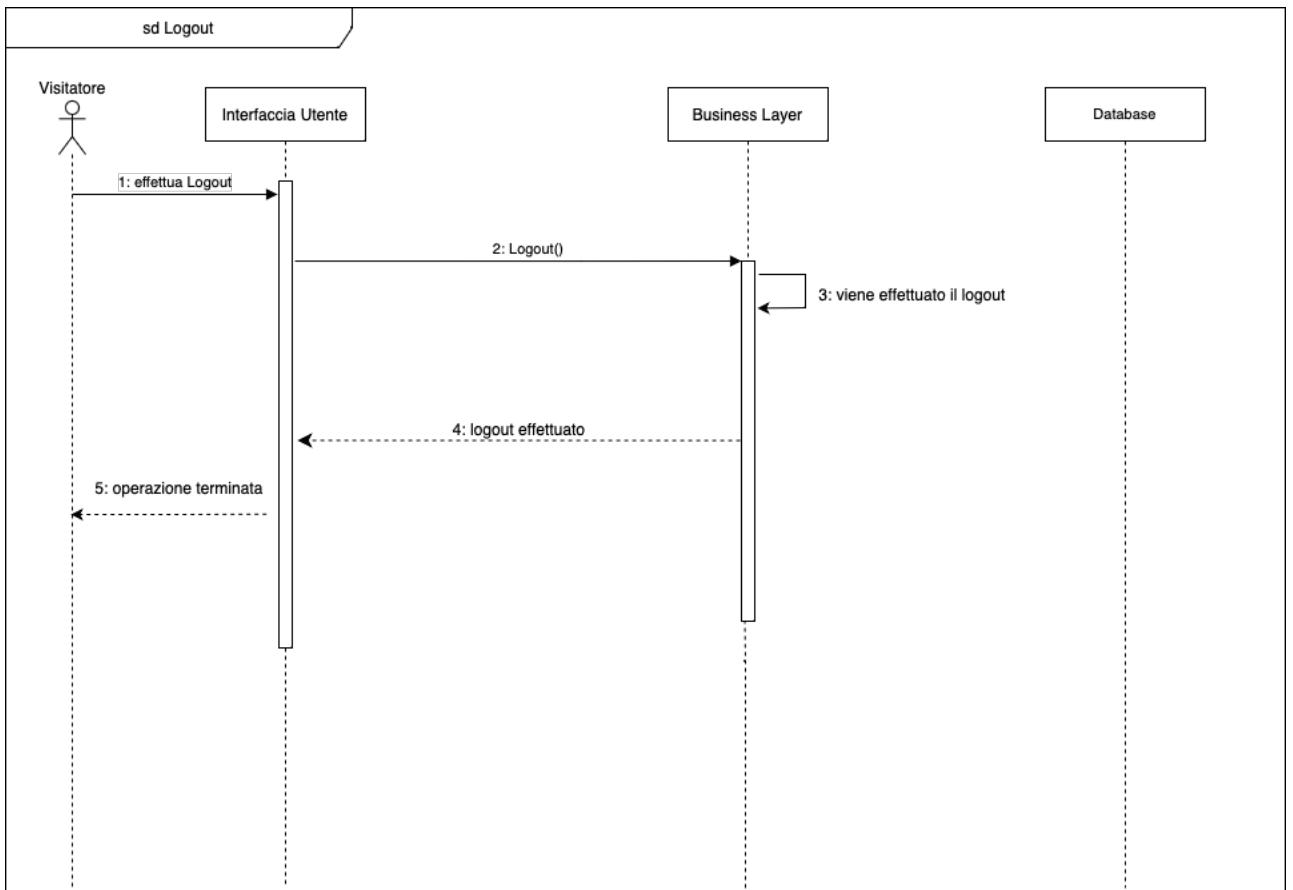


Figura 2.14: Diagramma di Sequenza - Logout

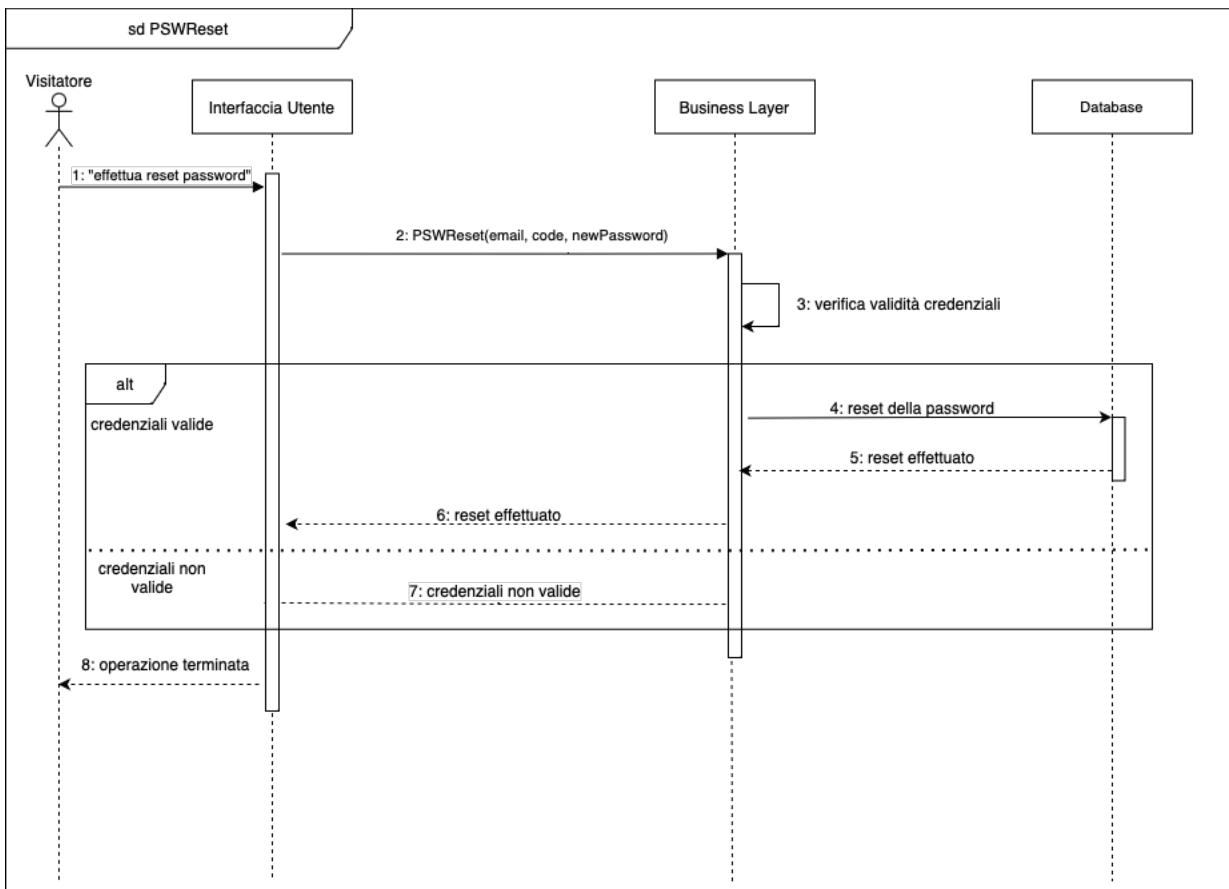


Figura 2.15: Diagramma di Sequenza - PSW Reset

### 3. DOCUMENTAZIONE DI PROGETTO

---

Il **Documento di Progettazione** fornisce una panoramica dettagliata dell'architettura e del design del sistema di autenticazione. Esso descrive in modo completo e approfondito le scelte progettuali, le strutture dei componenti, i modelli di interazione e altre informazioni pertinenti necessarie per la corretta implementazione e comprensione del sistema.

Durante la fase di progettazione per garantire principi come la separazione degli interessi, la modularità e l'astrazione si è deciso di adottare due pattern differenti, per il Front-End e per il Back-End.

Il pattern architettonico scelto per il Back-End è quello a layer, noto anche come "layered architecture" o "architettura a strati". In questa architettura, il sistema viene suddiviso in strati o livelli distinti, ognuno dei quali ha una responsabilità specifica e comunica con gli strati adiacenti secondo regole ben definite. I vantaggi che esso porta sono:

- Separazione delle responsabilità, ogni strato si concentra su un aspetto specifico del sistema, ad esempio la gestione dei dati, la logica di business o l'interfaccia utente. Ciò rende il sistema più modulare, mantenibile e facile da comprendere.
- Riusabilità del codice, i componenti all'interno di ogni strato possono essere progettati in modo indipendente e riutilizzati in altri contesti.
- Testabilità, è possibile creare test specifici per ciascun strato, isolando la logica e le funzionalità specifiche e garantendo che ciascuno di essi funzioni correttamente. Ciò semplifica il processo di test e di individuazione degli errori.
- Scalabilità, è possibile aggiungere o modificare uno specifico strato senza influire sugli altri strati.
- Manutenibilità, è più facile individuare e correggere errori, apportare modifiche o migliorare le funzionalità specifiche di uno strato senza dover toccare gli altri. Semplifica il lavoro di sviluppatori diversi che lavorano su diversi strati del sistema.

Per il Front-End, invece, è stato scelto il pattern MVC, i suoi vantaggi sono:

- Separazione delle responsabilità, favorisce il mantenimento di un codice pulito, modulare e facilmente comprensibile.
- Riuso.
- Basso accoppiamento, migliora la Testabilità dei componenti poiché tutte le classi e gli oggetti sono indipendenti l'uno dall'altro
- Modificabilità e Estendibilità.
- Manutenibilità.
- Sviluppo parallelo, i componenti di MVC possono essere sviluppati contemporaneamente.

Inoltre, si sono valutate diverse opzioni in termini di tecnologie, considerando fattori come la scalabilità, la sicurezza e la facilità di manutenzione. In particolare, le tecnologie analizzate e valutate sono Java Spring Boot e FastAPI.

I punti di forza di Java Spring Boot sono:

- ☺ Ampio supporto dalla comunità di sviluppo e numerosi pacchetti di terze parti disponibili.
- ☺ Architettura modulare e facile da usare.
- ☺ Supporto completo per database relazionali e noSQL.
- ☺ Adatto per applicazioni di grandi dimensioni e complesse.

I contro sono:

- ☹ Scarsa velocità di sviluppo rispetto ad alcuni framework più leggeri.
- ☹ Consuma molta memoria e richiede maggiori risorse di elaborazione rispetto ad altri framework.
- ☹ Requisiti di configurazione complessi, specialmente per progetti di grandi dimensioni.

Per quanto riguarda FastAPI, i punti di forza sono:

- ☺ Altissima velocità di sviluppo grazie alla tipizzazione automatica dei dati e alla documentazione automatica.
- ☺ Minima latenza durante l'esecuzione delle API.
- ☺ Supporto nativo per l'autenticazione e la gestione delle richiesteHTTP.
- ☺ Utilizzo efficiente della memoria per applicazioni di grandi dimensioni.
- ☺ Facilita la gestione di progetti multilingua grazie alla compatibilità con vari linguaggi di programmazione.

I contro sono:

- ☹ Una community limitata.
- ☹ Potrebbe richiedere una maggior esperienza in Python rispetto ad altri linguaggi.

In conclusione, entrambi i framework sono altamente utilizzati per lo sviluppo di applicazioni web, ma presentano differenze significative. Spring Boot offre al programmatore un maggiore supporto per la gestione di progetti complessi e offre un'ampia gamma di funzionalità integrate. Tuttavia, FastAPI risulta essere il framework ideale per gli sviluppatori che lavorano con un progetto a bassa latenza, grazie alla tipizzazione automatica e alle funzionalità di documentazione automatizzata delle API. Inoltre, FastAPI è più efficiente nella gestione della memoria rispetto a Spring Boot ed è ideale per applicazioni che richiedono una grande scalabilità. Per la facilità di gestione e la disponibilità limitata di risorse hardware, si è scelto di sviluppare l'applicazione utilizzando FastAPI e, dunque, Python, in quanto offre una vasta gamma di librerie e framework che consentono di sviluppare in modo efficiente e flessibile.

Si è scelto di utilizzare la tecnologia dei container. I container sono una tecnologia di virtualizzazione che permette di creare applicazioni in modo isolato e portatile. Sono differenti dalle macchine virtuali poiché utilizzano la tecnologia di containerizzazione e condividono il kernel dell'host e questo li rende più veloci e leggeri, più flessibili ed efficienti in termini di risorse di sistema. Inoltre, sono portabili, isolati, leggeri, scalabili e facilmente configurabili tramite script.

Docker è una tecnologia open source che semplifica la creazione, la gestione e la distribuzione di applicazioni in ambienti containerizzati. Grazie alla portabilità dei container, è possibile eseguire l'applicazione su qualsiasi piattaforma supportata da Docker, rendendo il processo di sviluppo e distribuzione estremamente efficiente.

### 3.1 Diagramma dei Package

I **Diagrammi dei Package** per il sistema di autenticazione hanno lo scopo di illustrare la struttura organizzativa delle componenti all'interno del sistema. Essi mostrano come i diversi moduli, le classi e le funzionalità sono raggruppati in package, fornendo una visione ad alto livello dell'architettura e delle relazioni tra le varie parti del sistema.

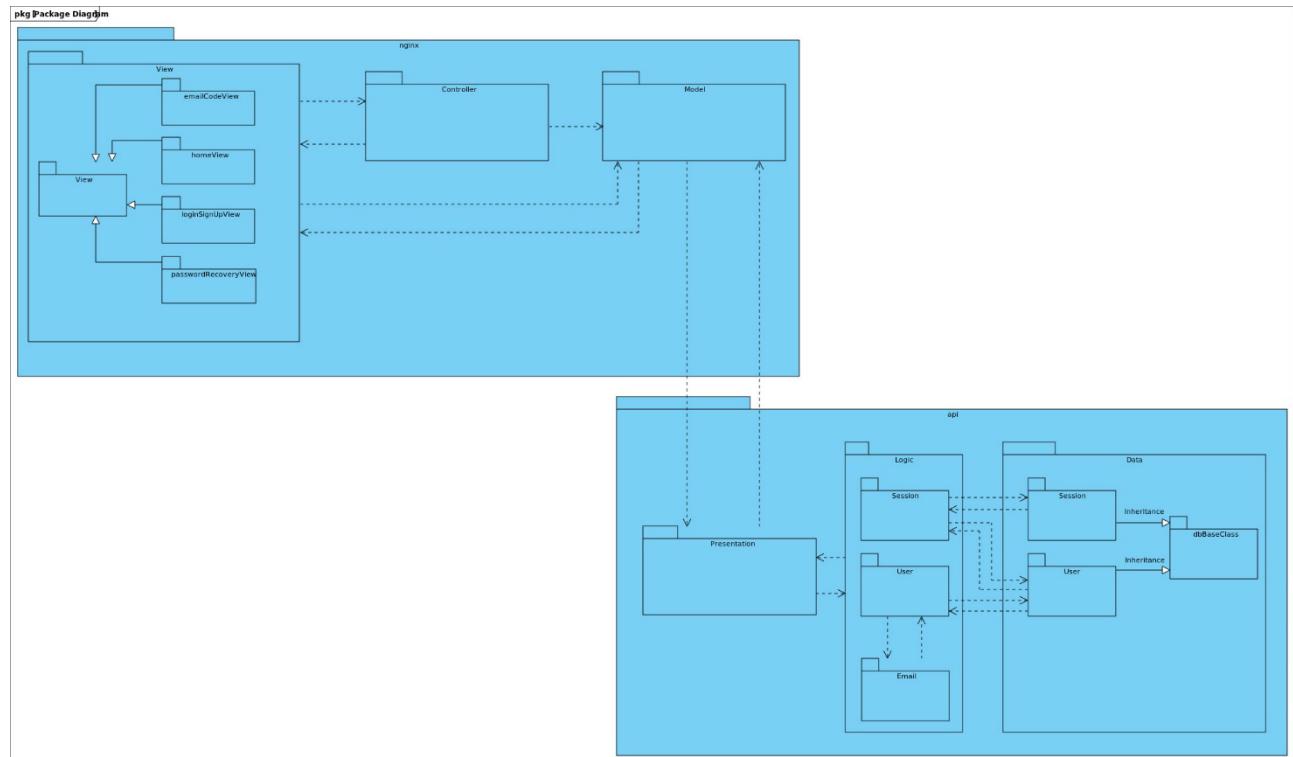


Figura 3.1: Diagramma dei Package

## 3.2 Diagramma delle Classi

Il **Diagramma delle Classi** del sistema di autenticazione fornisce una rappresentazione visuale delle classi che costituiscono il sistema, le loro relazioni e le proprietà e metodi associati a ciascuna classe. Questo diagramma è uno strumento fondamentale per comprendere la struttura e l'organizzazione delle classi all'interno del sistema di autenticazione.

Il seguente diagramma è una rappresentazione delle classi del Back-End.

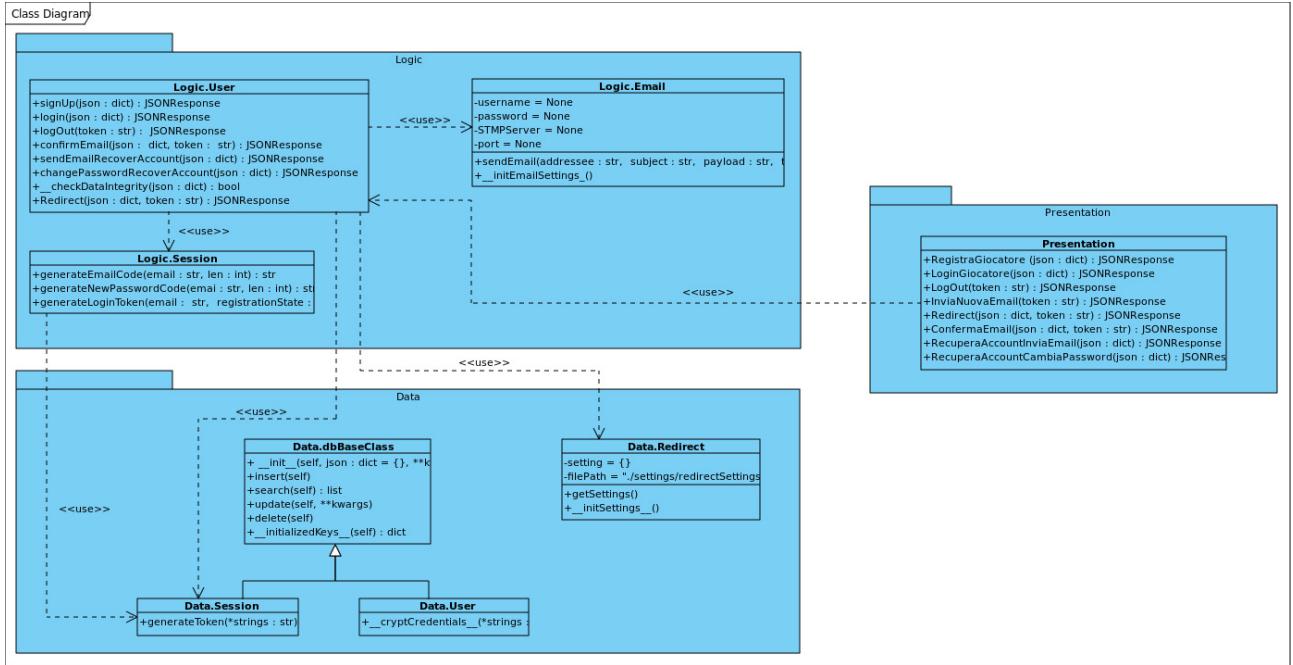


Figura 3.2: Diagramma delle Classi

### 3.3 Diagramma di Deploy

I **Diagrammi di Deploy** forniscono una visualizzazione della distribuzione fisica dei componenti del sistema di autenticazione, illustrando come le diverse parti del sistema sono organizzate e collocate su hardware e infrastrutture specifiche. Questi diagrammi consentono di comprendere l'ambiente di deploy del sistema di autenticazione e le relazioni tra i componenti hardware e software.

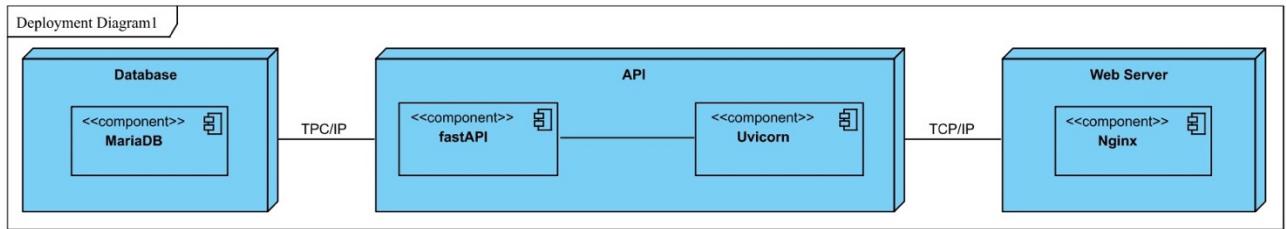


Figura 3.3: Diagramma di Deployment

Il seguente diagramma specifica in maniera più dettagliata l'ambiente e le versioni di deployment del sistema software.

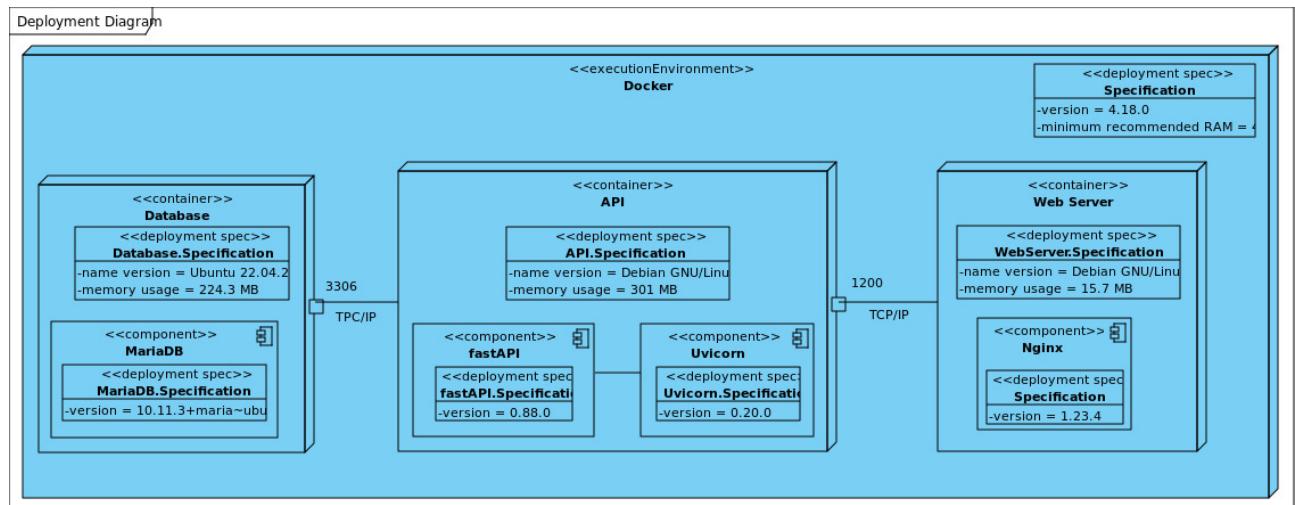


Figura 3.4: Diagramma di Deployment Raffinato

### 3.4 Diagramma C&C

I **Diagrammi C&C** forniscono una visualizzazione dell'architettura del sistema di autenticazione, identificando i componenti software, i connettori e le interazioni tra di essi. Questi diagrammi consentono di comprendere come il sistema di autenticazione si integra con altre parti del sistema più ampio, facilitando la comunicazione tra i diversi componenti e la gestione delle interazioni.

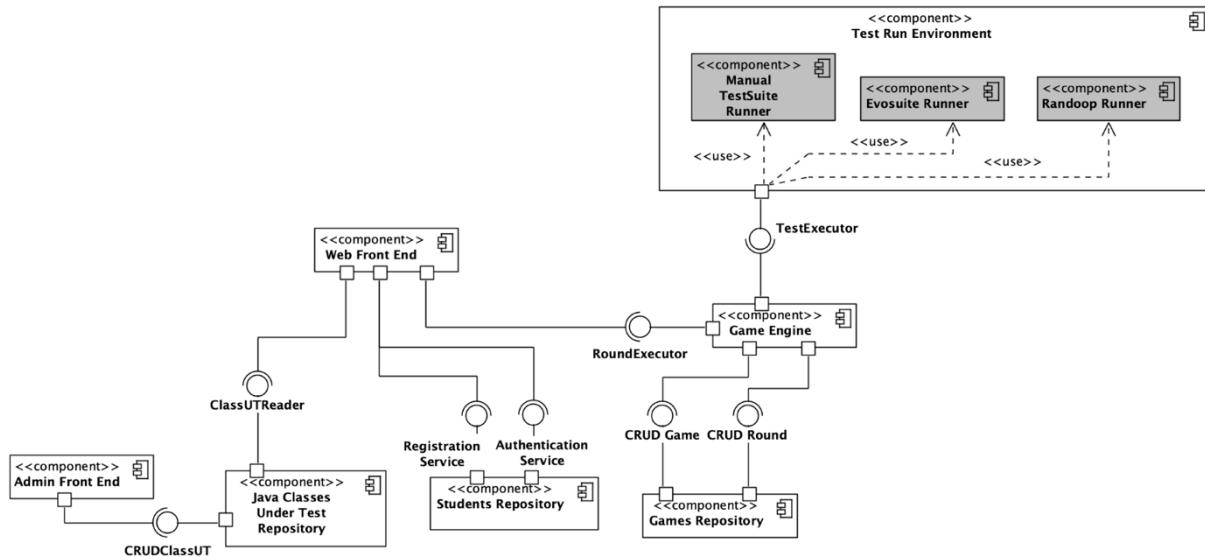


Figura 3.5: Diagramma C&C Generale

Il seguente diagramma specifica in maniera dettagliata componenti e connettori del sistema di autenticazione.

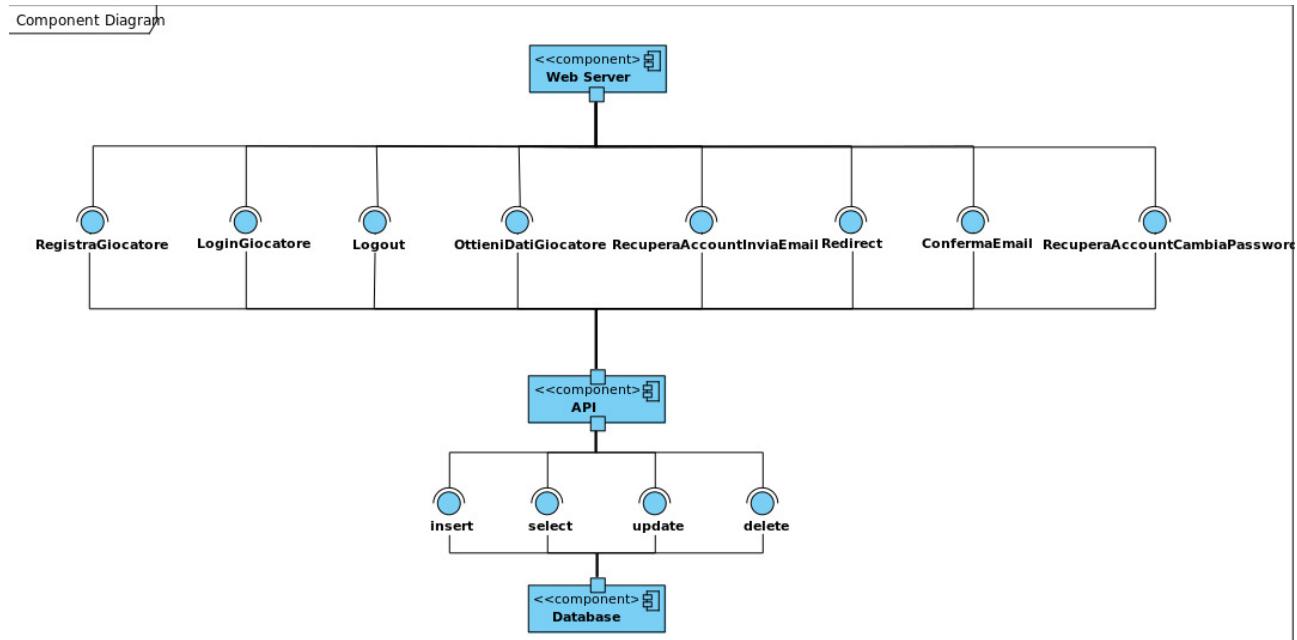


Figura 3.6: Diagramma C&C Specifico

### 3.5 Install View

Il diagramma di **Install View** del sistema di autenticazione fornisce una rappresentazione visuale delle componenti e delle dipendenze necessarie per installare e configurare correttamente il sistema. Questo diagramma è uno strumento fondamentale per comprendere l'ambiente di installazione e i requisiti necessari per il corretto funzionamento del sistema di autenticazione.

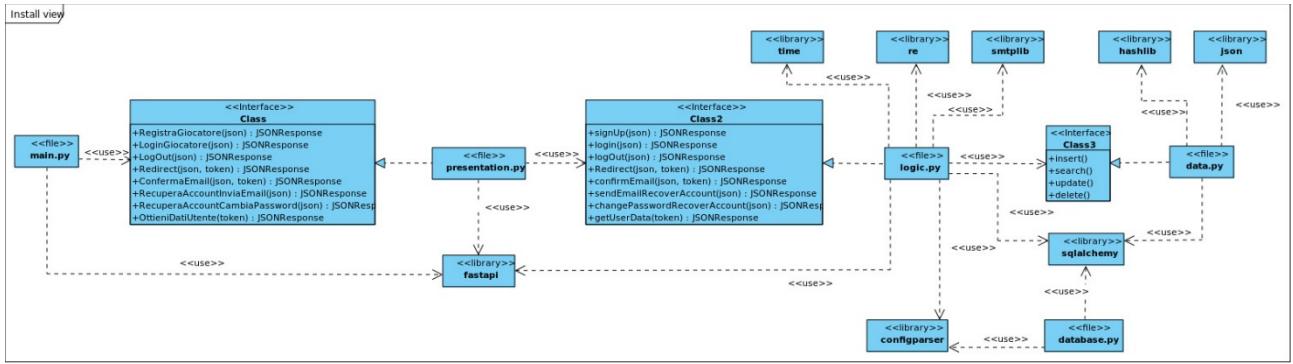


Figura 3.7: *Install View*

## 3.6 Diagrammi di Sequenza Raffinati

I **Diagrammi di Sequenza Raffinati** offrono una rappresentazione dettagliata e completa delle interazioni dinamiche all'interno del sistema di autenticazione. Questi diagrammi permettono di esplorare in modo approfondito i flussi di controllo e di dati durante il processo di autenticazione, consentendo di comprendere i passaggi interni e le interazioni tra gli oggetti coinvolti.

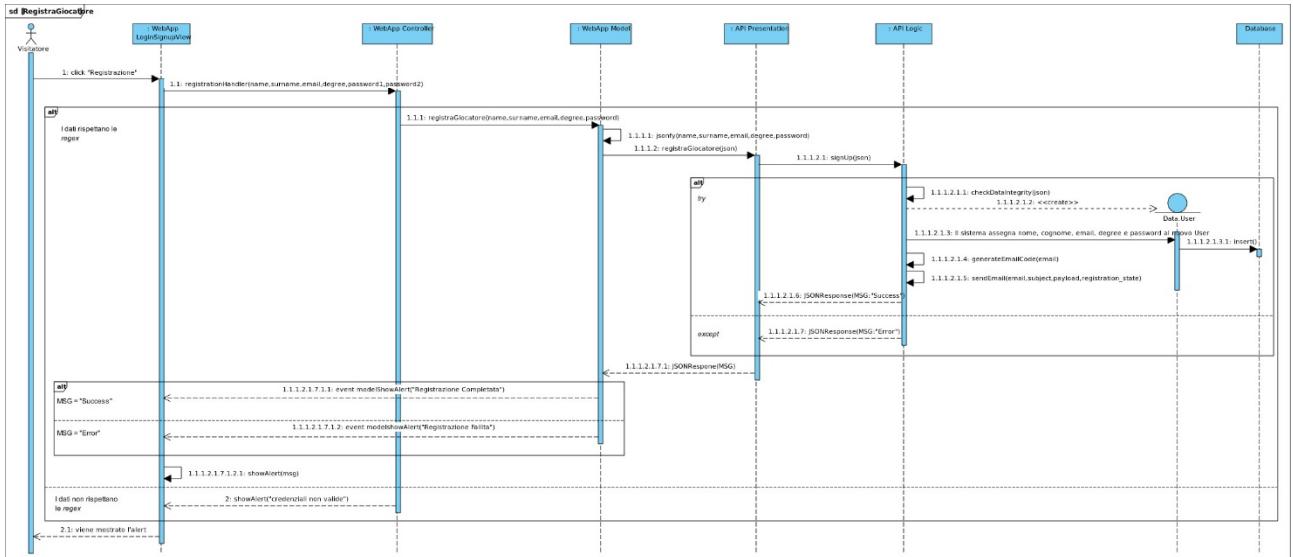


Figura 3.8: Diagramma di Sequenza Raffinato - Registrazione

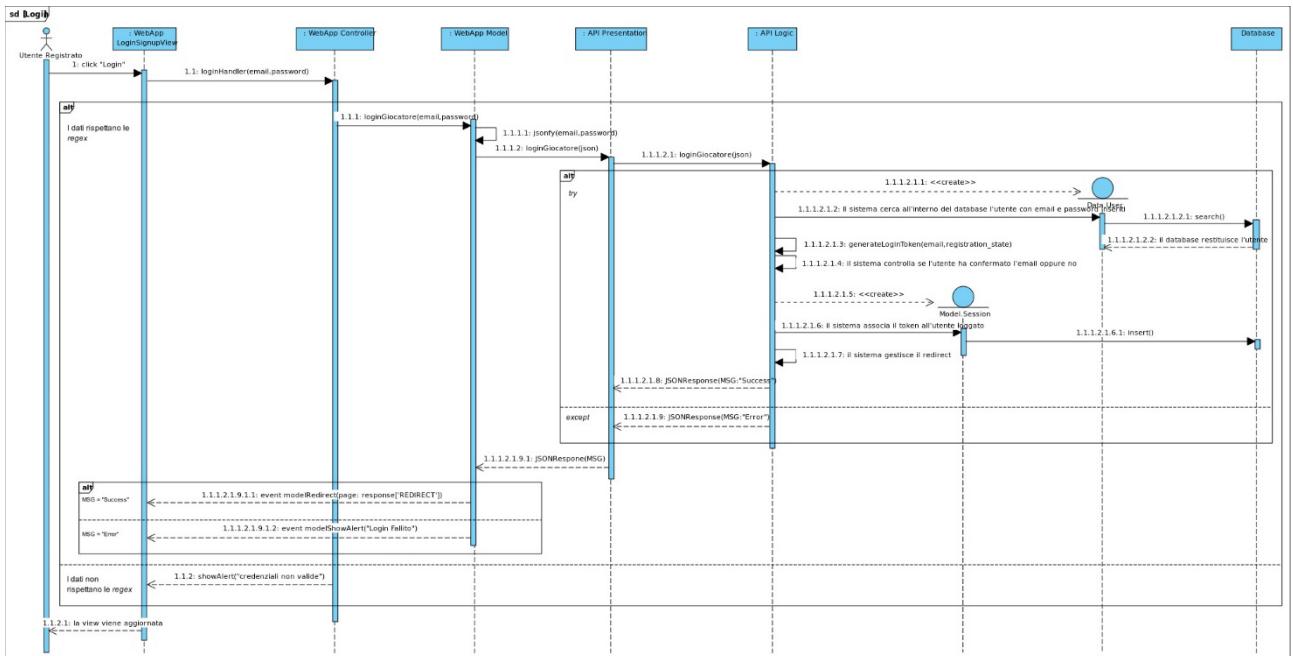


Figura 3.9: Diagramma di Sequenza Raffinato - Login

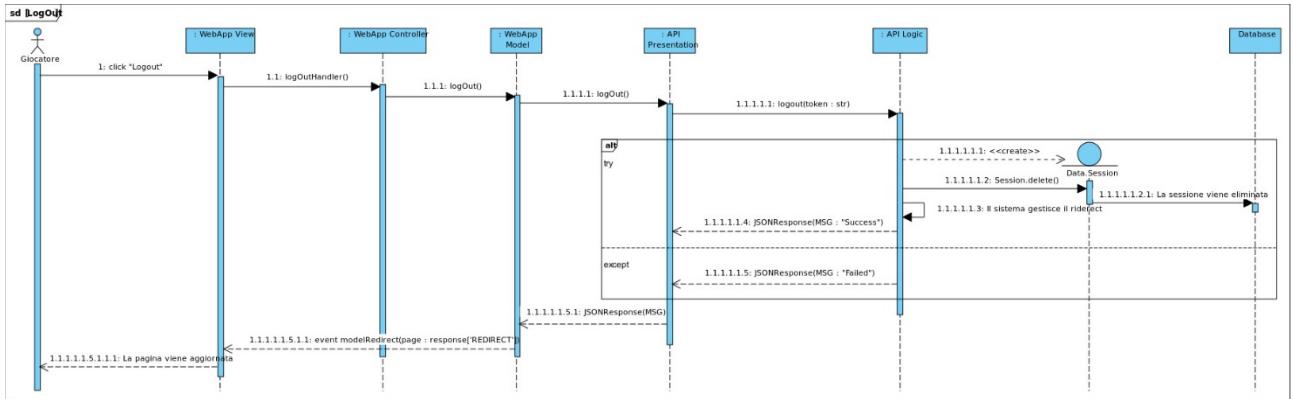


Figura 3.10: Diagramma di Sequenza Raffinato - Logout

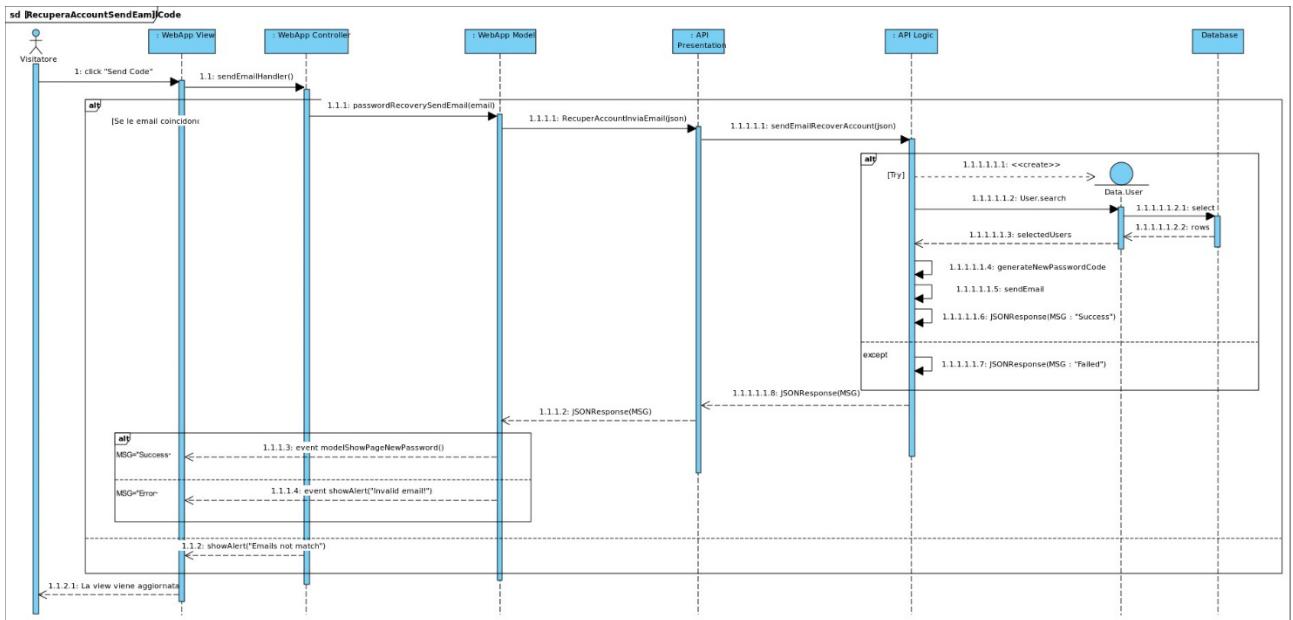


Figura 3.11: Diagramma di Sequenza Raffinato - Recupera Password Invia E-mail

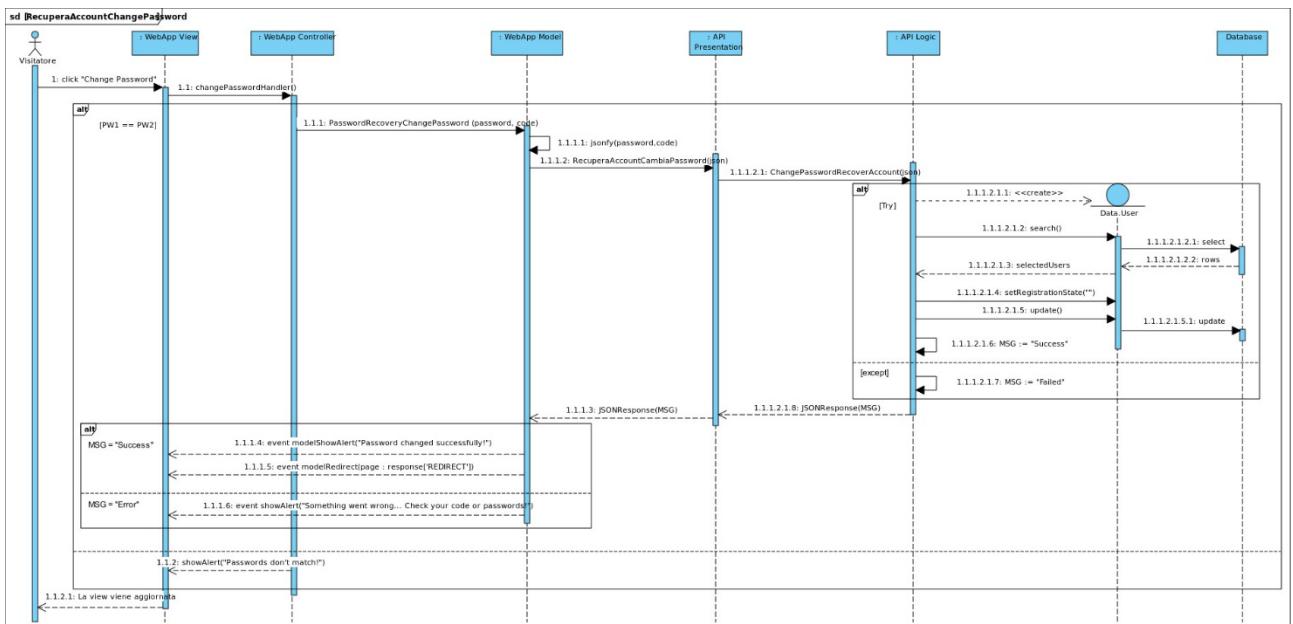


Figura 3.12: Diagramma di Sequenza Raffinato - Recupera Account Cambia Password

## 4. GUIDA ALL'UTILIZZO

---

Questo capitolo fornisce un'esaustiva panoramica sul corretto utilizzo del sistema per garantire un accesso sicuro e affidabile alle risorse digitali. Alla fine di questo capitolo si avrà una comprensione completa su come installare, configurare ed utilizzare il sistema.

### 4.1 Dipendenze

---

Questo paragrafo ha il solo scopo di illustrare i componenti utilizzati per la realizzazione del software e non fornisce ancora una guida all'installazione.

Il sistema di autenticazione è composto da tre macro-blocchi:

- **Web Server:** nginx-1.23.4
- **RestAPI:** FastAPI 0.88.0
- **Database:** MariaDB (10.11.3-MariaDB-1:10.11.3+maria~ubu2204)

In particolare, la RestAPI è stata sviluppata in **Python** 3.9.16 e sono state installate le seguenti librerie esterne:

- **SQLAlchemy** 2.0.15
- **Uvicorn** 0.20.0
- **Pymysql** 1.0.3
- **Pydantic** 1.10.7

Infine, il sistema è containerizzato con **Docker** 4.18.0.

### 4.2 Installazione

---

Questo capitolo fornisce una guida all'installazione del sistema di autenticazione. Seguendo attentamente le istruzioni fornite, sarà possibile configurare correttamente il sistema e iniziare ad utilizzarlo.

#### 4.2.1 Installazione di Docker

---

Un requisito fondamentale per la corretta installazione del sistema è quello di installare Docker sulla propria macchina, a questo proposito si rimanda l'utente alla [documentazione ufficiale](#) fornita da Docker.

#### 4.2.2 Personalizzazione del Docker-compose

---

Una volta installato Docker sarà necessario scaricare la cartella /server dalla seguente [repository github](#). Al percorso ~/server/docker-compose.yml è possibile trovare il file di configurazione con il quale verranno prodotti i container. Il file fornito è già completamente funzionante ma è possibile modificarlo per:

- Modificare i numeri di porto dei container
- Modificare le credenziali di accesso al database
- Aggiungere nuovi container

Per una configurazione più avanzata del Docker Compose si rimanda alla [documentazione ufficiale](#).

#### 4.2.3 Installazione del sistema di autenticazione

---

Una volta ottenuta la cartella sulla propria macchina locale sarà sufficiente eseguire questi due passi:

1. Avviare l'applicazione Docker Desktop ed attendere che questa termini il caricamento
2. Aprire la console, navigare fino a ~/server e digitare il comando  
`docker-compose up`

Per gli utenti Linux/MacOS sarà sufficiente utilizzare il terminale, mentre per gli utenti Windows sarà fondamentale usare il terminale fornito da WSL2.

Una volta terminato il caricamento il sistema è pronto all'utilizzo!

Dopo aver effettuato questo primo avvio, l'applicazione comparirà all'interno di Docker Desktop; quindi, per i successivi avvii, non sarà necessario ripetere tutta la procedura ma basterà fermarsi al passo 1. Al termine dell'installazione si potrà accedere al Web Server all'indirizzo [127.0.0.1](http://127.0.0.1).

#### 4.2.3 Inizializzazione del database

---

Nonostante il Web Server sia già completamente funzionante è necessario inizializzare il database. Per l'inizializzazione è fornita una query in `~/server/database/init.sql`.

### 4.3 Configurazione

---

Il sistema di autenticazione gode di alcune configurazioni che consentono di integrarlo in progetti complessi. In questo paragrafo il lettore apprenderà come configurare il servizio opportunamente.

#### 4.3.1 Configurazione del reindirizzamento

---

In un'applicazione in cui è presente un sistema di autenticazione, vi sono sia pagine pubbliche che pagine private. In generale i giocatori non possono accedere a pagine pubbliche così come i visitatori non possono accedere a pagine private. Il reindirizzamento deve essere sempre configurato qualora si volessero aggiungere nuove pagine pubbliche o private. A questo proposito, all'indirizzo `~/server/api/src/settings/redirectSettings.json` è presente un file di configurazione. Di seguito viene spiegato il significato di ogni singolo campo:

- **publicPages**: Lista delle pagine pubbliche
- **privatePages**: Lista delle pagine private
- **confirmationPage**: Pagina alla quale viene reindirizzato un giocatore che non ha confermato ancora il proprio account
- **notLoggedRedirect**: Pagina alla quale viene reindirizzato un visitatore che cerca di accedere a una pagina privata
- **loggedRedirect**: Pagina alla quale viene reindirizzato un giocatore che cerca di accedere a una pagina pubblica

Una volta modificato il file sarà necessario riavviare il container per rendere la configurazione effettivamente funzionante.

#### 4.3.2 Configurazione CORS

---

Le politiche CORS (Cross-Origin Resource Sharing) sono regole che vengono applicate dai browser web per controllare l'accesso alle risorse da origini diverse. Consentono o limitano le richieste di risorse tra domini diversi, al fine di migliorare la sicurezza e prevenire attacchi di tipo cross-site scripting (XSS) e cross-site request forgery (CSRF). In sostanza, le politiche CORS definiscono quali risorse possono

essere condivise tra domini diversi e quali richieste possono essere effettuate da un'applicazione web verso un'altra origine.

Nel sistema di autenticazione le politiche CORS vengono gestite su due livelli in modo da garantire la massima sicurezza.

Il primo livello di CORS viene gestito dal web server nginx siccome le richieste all'API transitano tutte per esso. La configurazione del web server può essere consultata e modificata attraverso il file `~/server/nginx/site.conf`. Nell'esempio scaricato i CORS consentono soltanto richieste provenienti dall'indirizzo [127.0.0.1](http://127.0.0.1) e bloccano quelle provenienti da tutti gli altri. Per configurare delle richieste da domini personalizzati basterà sostituire <http://127.0.0.1> con l'indirizzo del proprio sito internet.

Per una documentazione più avanzata sulla configurazione del web server si rimanda il lettore alla [documentazione ufficiale di nginx](#).

Il secondo livello di CORS è gestito direttamente dal middleware di fastAPI, questo viene configurato nel file `~/server/api/src/main.py`. Anche in questo caso la configurazione di default consente solo le richieste provenienti da [127.0.0.1](http://127.0.0.1). Per permettere richieste da un altro URL basta semplicemente modificare l'indirizzo ed inserire l'URL del proprio sito internet. Per una configurazione più avanzata si rimanda il lettore alla [documentazione ufficiale di fastAPI](#)

#### [4.3.3 Configurazione del collegamento tra API e database/mail server](#)

---

Nel caso si decida di non utilizzare le impostazioni di default sarà necessario modificare le credenziali con cui il sistema accede a database e-mail server. A questo proposito esiste il file `~/server/api/src/settings/utils.ini`, in cui sarà possibile inserire le proprie credenziali per permettere all'API di connettersi correttamente alle entità esterne.

#### [4.3.4 Configurazione della comunicazione con API](#)

---

Nel caso si decida di utilizzare metodi esposti da un'altra API, diversa da quella di default, sarà necessario modificare il `baseURL` e l'`API address` nel file `~/server/nginx/www/js/model/Model.js` per effettuare la `fetch` verso l'indirizzo desiderato.

### [4.4 Esempi d'Uso](#)

---

Nella [repository github](#) sono presenti dei file DEMO nella cartella `~/server/nginx/www` che utilizzano il sistema di autenticazione. Nella DEMO sono presenti:

- Pagina di login/registrazione
- Pagina di reset della password
- Pagina di conferma dell'account
- Pagina dei termini e delle condizioni fittizia
- Homepage dell'utente
- Pagina di base

La pagina di base fornisce uno scheletro iniziale che supporta il meccanismo di reindirizzamento implementato nell'API.

## 4.5 Rotte API

Questa sezione fornisce una guida dettagliata sulle diverse rotte disponibili all'interno della nostra API, consentendo agli sviluppatori di comprendere e utilizzare correttamente le funzionalità offerte dal nostro sistema.

L'API rappresenta il punto di accesso principale per interagire con il nostro sistema, consentendo agli sviluppatori di inviare richieste e ottenere risposte strutturate. Questa documentazione delle rotte fornisce informazioni cruciali sulle varie rotte disponibili, i parametri richiesti, i formati di input e output, nonché gli eventuali errori o eccezioni che possono essere generati.

### 4.5.1 RegistraGiocatore

<b>Descrizione</b>	Registra un nuovo giocatore nel sistema ed invia un codice di confermata alla e-mail specificata.
<b>URL</b>	/RegistraGiocatore
<b>Metodo</b>	POST
<b>Body</b>	{"NAME": nome del nuovo giocatore, "SURNAME": cognome del nuovo giocatore, "EMAIL": e-mail del nuovo giocatore, "DEGREE": degree del nuovo giocatore, "PW": password del nuovo giocatore }
<b>Risposta</b>	{"MSG": esito dell'operazione} Se non vengono sollevati errori, l'esito dell'operazione è "Success"
<b>Errori</b>	- Invalid User Data (dati in un formato non valido) - Invalid Request (impossibile effettuare l'inserimento) - Could Not Send Email (impossibile inviare l'e-mail di conferma)
<b>Note Aggiuntive</b>	- Assicurarsi nel front-end che i parametri della richiesta siano nel formato corretto

### 4.5.2 LogOut

<b>Descrizione</b>	Permette ad un giocatore di tornare alle pagine iniziale
<b>URL</b>	/ LogOut
<b>Metodo</b>	POST
<b>Body</b>	{ "" }
<b>Risposta</b>	{"MSG": esito dell'operazione, "REDIRECT": pagina da caricare in caso di successo} Se non vengono sollevati errori, l'esito dell'operazione è "Success".
<b>Errori</b>	- "Could not delete the session" (Richiesta invalida)
<b>Note Aggiuntive</b>	- Opzione non possibile per i visitatori. - L'API elabora la richiesta utilizzando il cookie TOKEN.

#### 4.5.3 LoginGiocatore

<b>Descrizione</b>	Permette ad un visitatore registrato nel sistema di accedere alle pagine private
<b>URL</b>	/LoginGiocatore
<b>Metodo</b>	POST
<b>Body</b>	{"EMAIL": e-mail del nuovo giocatore, "PW": password del nuovo giocatore}
<b>Risposta</b>	{"MSG": esito dell'operazione, "REDIRECT": pagina da caricare in caso di successo} Se non vengono sollevati errori, l'esito dell'operazione è "Success", in questo caso la risposta conterrà un cookie identificato con TOKEN.
<b>Errori</b>	- Invalid credentials (Giocatore non trovato). - Invalid Request (Impossibile inserire il token). - Could not login because the user has asked for recovery account (Impossibile loggare un account che ha chiesto il recupero password).
<b>Note Aggiuntive</b>	- Assicurarsi nel front-end che i parametri della richiesta siano nel formato corretto.

#### 4.5.4 Redirect

<b>Descrizione</b>	Fornisce al front-end le pagine a cui reindirizzare l'utente
<b>URL</b>	/Redirect
<b>Metodo</b>	POST
<b>Body</b>	{"LOC": specifica la pagina di origine della richiesta}
<b>Risposta</b>	{"MSG": esito dell'operazione, "REDIRECT": pagina da caricare in caso di successo} Se non vengono sollevati errori, l'esito dell'operazione è "Success"
<b>Errori</b>	- Invalid Request (Errore di accesso di un utente ad una pagina privata con un TOKEN non corretto).
<b>Note Aggiuntive</b>	- In questo caso il front-end dovrà provvedere a reindirizzare l'utente alla pagina specificata in REDIRECT. - L'API calcola il REDIRECT anche in funzione del cookie TOKEN.

#### 4.5.5 ConfermaEmail

<b>Descrizione</b>	L'utente conferma il proprio account attraverso il codice fornитогли in fase di registrazione.
<b>URL</b>	/ConfermaEmail
<b>Metodo</b>	POST

<b>Body</b>	{"CODE": codice inviato alla e-mail dell'utente in fase di registrazione}
<b>Risposta</b>	{"MSG": esito dell'operazione, "REDIRECT": pagina da caricare in caso di successo} Se non vengono sollevati errori, l'esito dell'operazione è "Success"
<b>Errori</b>	- Invalid Request (Errore nel database). - Confirmation failed (Codice errato).
<b>Note Aggiuntive</b>	- Il front-end dovrà provvedere a reindirizzare l'utente alla pagina specificata in REDIRECT.

#### 4.5.6 RecuperaAccountInviaEmail

---

<b>Descrizione</b>	Fornisce all'utente un codice tramite e-mail per effettuare il recupero dell'account.
<b>URL</b>	/RecuperaAccountInviaEmail
<b>Metodo</b>	POST
<b>Body</b>	{"EMAIL": e-mail del giocatore di cui si vuole resettare la password}
<b>Risposta</b>	{"MSG": esito dell'operazione} Se non vengono sollevati errori, l'esito dell'operazione è "Success"
<b>Errori</b>	- Invalid Request (Errore nel database). - Unregistered email - Could not send email (Errore nel mail server)
<b>Note Aggiuntive</b>	- Il front-end deve finalizzare l'operazione di reset della password richiamando la rotta "RecuperaAccountCambiaPassword".

#### 4.5.7 RecuperaAccountCambiaPassword

---

<b>Descrizione</b>	Permette ad un utente che ha richiesto il recupero dell'account di configurare una nuova password.
<b>URL</b>	/RecuperaAccountCambiaPassword
<b>Metodo</b>	POST
<b>Body</b>	{"CODE": codice inviato alla e-mail dell'utente in fase di registrazione, "PW": nuova password immessa dall'utente}
<b>Risposta</b>	{"MSG": esito dell'operazione, "REDIRECT": pagina da caricare in caso di successo} Se non vengono sollevati errori, l'esito dell'operazione è "Success"
<b>Errori</b>	- Invalid Request (Errore nel database). - The code is invalid
<b>Note Aggiuntive</b>	- Il front-end dovrà provvedere a reindirizzare l'utente alla pagina specificata in REDIRECT.

#### [4.5.8 OttieniDatiUtente](#)

---

<b>Descrizione</b>	Fornisce i dati dell'utente associati alla sessione.
<b>URL</b>	/OttieniDatiUtente
<b>Metodo</b>	POST
<b>Body</b>	{""}
<b>Risposta</b>	{"MSG": esito dell'operazione, "NAME", "SURNAME", "EMAIL", "DEGREE"} Se non vengono sollevati errori, l'esito dell'operazione è "Success".
<b>Errori</b>	- Invalid Request (Errore nel database).
<b>Note Aggiuntive</b>	

## 5. TESTING

---

Il **Testing** del software è un'attività fondamentale per garantire la qualità e l'affidabilità dei sistemi software. Tra le diverse strategie di testing disponibili, il testing blackbox si concentra sull'esame del software senza considerare la sua struttura interna. In questa prospettiva, due elementi chiave sono la strategia di testing basata su ECC (Each Choice Coverage) e la libreria Python Unittest.

La strategia di testing basata su ECC mira a coprire tutte le combinazioni possibili delle scelte di input nel software. In altre parole, si concentra sull'esplorazione accurata di tutte le decisioni che il software può prendere durante l'esecuzione. Questo tipo di testing si adatta bene a una prospettiva blackbox, in quanto non richiede la conoscenza dettagliata dell'implementazione interna del software, ma si concentra piuttosto sulla valutazione del suo comportamento esterno.

Unittest, d'altra parte, è specificamente progettato per supportare il processo di testing nel contesto dello sviluppo software. Esso fornisce un framework per creare, eseguire e valutare i test automatizzati. Unittest facilita la scrittura dei test, l'organizzazione dei casi di test e la generazione di report sulle prestazioni e sulla copertura dei test. È particolarmente adatto per il testing blackbox, in quanto consente di verificare il comportamento del software senza necessariamente conoscere i dettagli dell'implementazione interna.

Integrando la strategia di testing basata su ECC e Unittest, è possibile condurre un testing blackbox accurato e completo. Esso rappresenta un approccio solido per valutare il comportamento esterno di un sistema software senza dover conoscere la sua implementazione interna. Questa combinazione permette di eseguire test accurati, di identificare potenziali difetti e di migliorare la qualità e l'affidabilità del software.

### 5.1 Casi di Test

---

Questa sezione fornisce un elenco completo dei **Casi di Test** progettati per verificare l'efficacia, l'affidabilità e la sicurezza del sistema di autenticazione del nostro software. I casi di test presentati coprono una vasta gamma di scenari che possono verificarsi durante il processo di autenticazione. Ogni caso di test è stato sviluppato con attenzione per garantire che tutti gli aspetti cruciali del sistema di autenticazione siano sottoposti a rigorose verifiche.

### 5.1.1 Registrazione

Per poter effettuare la registrazione, un utente deve compilare obbligatoriamente i seguenti campi all'interno di un form, per poi confermare l'operazione di registrazione:

- **Nome:** stringa che non contiene numeri
- **Cognome:** stringa che non contiene numeri
- **E-mail:** stringa che rispetta il pattern di un indirizzo di posta elettronica
- **Password:** stringa con numero di caratteri compreso tra 8 e 31

<b>Titolo</b>	<b>Descrizione</b>	<b>Pre-condizioni</b>	<b>Input</b>	<b>Output</b>	<b>Post-condizioni</b>	<b>Stato</b>
CAMPI VUOTI	I campi compilati dall'utente sono tutti vuoti	L'utente non registrato prova ad effettuare la registrazione non compilando i campi del form di registrazione	Tutti i campi obbligatori sono stringhe nulle	Il sistema mostra a video un messaggio di errore	L'utente non risulta registrato al sistema	PASS
NOME NON VALIDO	I campi compilati dall'utente sono tutti validi, eccetto il campo nome	L'utente non registrato prova ad effettuare la registrazione, compilando correttamente tutti i campi del form di registrazione, eccetto quello del nome	Tutti i campi obbligatori del form sono stringhe valide, eccetto quella del campo nome	Il sistema mostra a video un messaggio di errore	L'utente non risulta registrato al sistema	PASS
COGNOME NON VALIDO	I campi compilati dall'utente sono tutti validi, eccetto il campo cognome	L'utente non registrato prova ad effettuare la registrazione, compilando correttamente tutti i campi del form di registrazione, eccetto quello del cognome	Tutti i campi obbligatori del form sono stringhe valide, eccetto quella del campo cognome	Il sistema mostra a video un messaggio di errore	L'utente non risulta registrato al sistema	PASS
E-MAIL NON VALIDA	I campi compilati dall'utente sono tutti validi, eccetto il	L'utente non registrato prova ad effettuare la registrazione, compilando correttamente tutti i campi del	Tutti i campi obbligatori del form sono stringhe valide,	Il sistema mostra a video un messaggio di errore	L'utente non risulta registrato al sistema	PASS

	campo e-mail	form di registrazione, eccetto quello dell'e-mail	eccetto quella del campo e-mail			
PASSWORD TROPPO LUNGA	I campi compilati dall'utente sono tutti validi, eccetto il campo password	L'utente non registrato prova ad effettuare la registrazione, compilando correttamente tutti i campi del form di registrazione, eccetto quello della password	Tutti i campi obbligatori del form sono stringhe valide, eccetto quella del campo password	Il sistema mostra a video un messaggio di errore	L'utente non risulta registrato al sistema	PASS
UTENTE GIÀ REGISTRATO	L'utente prova a registrarsi con un'e-mail che risulta essere associata ad un altro account	L'utente prova ad effettuare la registrazione, compilando correttamente tutti i campi del form di registrazione, ma l'e-mail inserita è già associata ad un altro account	Tutti i campi obbligatori del form sono stringhe valide, ma l'e-mail è già associata ad un altro utente registrato	Il sistema mostra a video un messaggio di errore	L'utente non risulta registrato al sistema	PASS

### 5.1.2 Login

Per poter effettuare il login, un utente deve compilare obbligatoriamente i seguenti campi all'interno di un form, per poi accedere al sistema:

- *E-mail*: stringa che rispetta il pattern di un indirizzo di posta elettronica, associata ad un account registrato
- *Password*: stringa con numero di caratteri compreso tra 8 e 31, associata ad un account registrato

<b>Titolo</b>	<b>Descrizione</b>	<b>Pre-condizioni</b>	<b>Input</b>	<b>Output</b>	<b>Post-condizioni</b>	<b>Stato</b>
UTENTE NON REGISTRATO LOGIN	I campi compilati dall'utente sono validi, ma e-mail e password inserite non sono associate ad un account registrato	L'utente prova ad effettuare il login compilando correttamente i campi del form, ma e-mail e password inserite non sono associate ad un account registrato	Tutti i campi obbligatori sono stringhe valide, ma non associate ad un account registrato	Il sistema mostra a video un messaggio di errore	L'utente non risulta loggato al sistema	PASS
UTENTE REGISTRATO CON CREDENZIALE SBAGLIATA	I campi compilati dall'utente sono validi, eccetto quello della password, che non è associata all'account relativo all'e-mail inserita	L'utente prova ad effettuare il login compilando correttamente i campi del form, ma la password, non è associata all'account relativo all'e-mail inserita	Tutti i campi obbligatori sono stringhe valide, ma la password non è associata all'account relativo all'e-mail inserita	Il sistema mostra a video un messaggio di errore	L'utente non risulta loggato al sistema	PASS

### 5.1.3 Password Reset

Per poter effettuare il reset della password, un utente deve compilare obbligatoriamente i seguenti campi all'interno di un form, per poi accedere al sistema:

- *E-mail*: stringa che rispetta il pattern di un indirizzo di posta elettronica, associata ad un account registrato
- *Code*: stringa alfanumerica spedita via e-mail
- *Password*: stringa con numero di caratteri compreso tra 8 e 31

<b>Titolo</b>	<b>Descrizione</b>	<b>Pre-condizioni</b>	<b>Input</b>	<b>Output</b>	<b>Post-condizioni</b>	<b>Stato</b>
UTENTE NON REGISTRATO PSW RESET	Il campo e-mail compilato dall'utente è valido, ma non è associato ad un account registrato	L'utente prova ad effettuare il reset della password compilando correttamente il campo e-mail del form, ma essa non è associata ad un account registrato	Il campo e-mail è una stringa valida, ma non è associata ad un account registrato	Il sistema mostra a video un messaggio di errore	L'utente non può modificare la password	PASS
CODICE ERRATO	I campi compilati dall'utente sono validi, eccetto quello del code, che non corrisponde a quello ricevuto	L'utente prova ad effettuare il reset della password compilando correttamente i campi del form, ad eccezione del code, che non corrisponde a quello ricevuto	Tutti i campi obbligatori sono stringhe valide, ma il code non corrisponde a quello ricevuto	Il sistema mostra a video un messaggio di errore	L'utente non può modificare la password	PASS

## 6. GLOSSARIO

---

<b>Termini</b>	<b>Significato</b>
Sistema	Insieme delle risorse hardware e software che offrono il servizio
Utente	Attore umano che interagisce con il sistema
Visitatore	Utente che interagisce con le pagine pubbliche
Visitatore Registrato	Visitatore che ha effettuato la registrazione al sistema
Giocatore	Utente che interagisce con le pagine private
Pagine Pubbliche	Pagine a cui è possibile accedere solo se non è stato effettuato il Login
Pagine Private	Pagine a cui è possibile accedere solo se è stato effettuato il Login