Università degli Studi di Federico Secondo

Corso di Laurea in NomeCorso

Titolo della Relazione

Autore: Nome Cognome

Matricola: 123456

Anno Accademico: 2024/2025

Indice

Capitolo 1

Analisi e specifica dei requisiti

1.1 Analisi nomi-verbi

Il sistema consente la registrazione di utenti, che devono fornire nome, cognome, indirizzo e-mail e password. Ogni cliente dispone di un profilo personale, accessibile tramite autenticazione, dove può visualizzare e gestire le informazioni del proprio account, modificare i dati personali, e consultare lo storico dei biglietti acquistati ed opzionalmente la propria immagine del profilo. Ogni profilo mostra opzionalmente anche il numero totale di eventi a cui l'cliente ha partecipato.

Gli amministratori della piattaforma possono creare nuovi eventi, specificando per ciascuno titolo, descrizione, data, orario, luogo e numero massimo di partecipanti. Gli eventi pubblicati sono consultabili dagli utenti registrati tramite un catalogo eventi, filtrabile opzionalmente per data o località.

Durante il processo di acquisto, l'cliente seleziona un evento e riceve un biglietto elettronico, identificato da un codice univoco. Il biglietto contiene: nome dell'evento, data, orario, nome del partecipante e codice identificativo. I biglietti possono essere scaricati o visualizzati direttamente dal profilo cliente.

Nel giorno dell'evento, l'cliente può accedere a una apposita interfaccia grafica pensata per il controllo degli accessi. In questa interfaccia gli viene presentato l'elenco di tutti gli eventi previsti per la data odierna. L'cliente seleziona l'evento a cui intende partecipare e inserisce il codice del biglietto precedentemente ricevuto. Il sistema, a questo punto, effettua una serie di verifiche: controlla che il codice del biglietto esista e sia effettivamente associato all'evento selezionato, che la data indicata sul biglietto coincida con quella odierna e che il biglietto non sia già stato utilizzato. Se tutte le condizioni risultano verificate, il sistema autorizza l'accesso e marca il biglietto come "consumato". In caso contrario, viene restituito un messaggio di errore esplicativo che impedisce l'ingresso.

Il sistema mantiene traccia in tempo reale delle persone presenti a ciascun evento, aggiornando dinamicamente il numero di ingressi effettuati. Gli amministratori possono accedere a un pannello di gestione per ogni evento. Per gli eventi odierni, il sistema consente di visualizzare non solo il numero di utenti registrati, ma anche l'elenco aggiornato degli utenti effettivamente presenti in quel momento. Per gli eventi passati, invece, l'amministratore potrà accedere anche al numero totale di partecipanti che hanno avuto accesso, senza possibilità di consultarne i nomi.

L'applicazione deve essere accessibile via web da dispositivi desktop e mobili, offrire un'interfaccia grafica chiara e intuitiva, e implementare meccanismi di sicurezza per la protezione dei dati personali, l'autenticazione degli utenti e l'integrità dei biglietti elettronici.

LEGENDA

Classe
Attributo
Funzionalità
Attore
Classe-Attore

1.2 Revisione dei Requisiti

1. Il sistema deve consentire ad un cliente di registrarsi
2. La registrazione consiste nell'inserire nome, cognome, indirizzo e-mail e password
3. Il sistema deve offrire una funzionalità di autenticazione
4. il sistema deve gestire per ogni cliente lo storico dei biglietti acquistati, numero totali di eventi a cui l'cliente ha partecipato e un ImmagineProfilo
5. Il sistema consente di visualizzare lo storico dei biglietti acquisati dall'cliente
6. Il sistema offre una funzionalità di modifica dei dati personali
7. Il sistema deve consentire agli amministratori la creazione di eventi
8. Degli eventi si vuole memorizzare titolo, data, orario, luogo e numero massimo di partecipanti
9. Il sistema deve offrire un catalogo eventi consultabile da un utente registrato
10. Il sistema deve fornire una funzionalità di ricerca di un evento per nome, data o località.
11. Il sistema deve consentire l'acquisto di biglietto per un evento
12. Ogni biglietto elettronico deve avere un codice identificativo univoco
13. il sistema deve offrirre la possibilità all'cliente di visualizzare un biglietto acquistato
14. Il sistema deve offrire la possibilit all'cliente di scaricare un biglietto acquistato
15. Il sistema deve consentire al cliente la partecipazione ad un evento
16. Un biglietto marcato come consumato non può essere più essere riutilizzato
17. Un cliente durante la fase di acquisto può comprare un solo biglietto ad esso associato
18. Il sistema deve tener traccia dei clienti presenti a ciascun evento
19. Il sistema deve fornire all'amministratore la possibilità di consultare informazioni aggiuntive per i suoi eventi pubblicati
20. Il sistema deve implementare meccanismi di sicurezza per la protezione dei dati personali e per l'autenticazione degli utenti
21. Il sistema deve offrire un'interfaccia grafica chiara e intuitiva
22. Il sistema deve garantire l'integrità dei biglietti elettronici
23. Il sistema deve essere accessibile da dispositivi mobili e desktop

1.3 Glossario dei termini

Termine	Descrizione	Sinonimi
Amministratore	Amministratore della piattaforma che si occupa della gestione degli eventi	
Biglietto elettronico	Biglietto acquistabile e utilizzare per partecipare all'evento a cui è associato	
Catalogo eventi	Catalogo che contiene tutti gli eventi pubblicati dagli amministratori	
UtenteNonRegistrato	Una persona che intende registrarsi presso il sistema	
UtenteRegistrato	Un Utente che si è registrata presso il sistema	
Cliente	Utente registrato che acquista o partecipa a eventi. Nei diagrammi dei casi d'uso è rappresentato dall'attore "Utente"	

1.4 Classificazione dei Requisiti

1.4.1 Requisiti Funzionali

ID	Requisito	Origine
RF_{01}	Il sistema offre la possibilità all'cliente di registrarsi	1
RF_{02}	Il sistema deve offrire una funzionalità di autenticazione	3
RF_{03}	il sistema deve gestire per ogni cliente lo storico dei biglietti acquistati e un ImmagineProfilo	4
RF_{04}	Il sistema consente di visualizzare lo storico dei biglietti acquistati dall'cliente	5
RF_{05}	Il sistema offre una funzionalità di modifica dei dati personali	6
RF_{06}	Il sistema deve consentire agli amministratori la creazione di eventi	7
RF_{07}	Il sistema deve offrire un catalogo eventi consultabile da un utente registrato	9
RF_{08}	Il sistema deve fornire una funzionalità di ricerca di un evento per nome, data o località.	10
RF_{09}	Il sistema deve consentire l'acquisto di biglietto per un evento	11
RF_{11}	il sistema deve offrirre la possibilit'a all'cliente di visualizzare un biglietto acquistato	13
RF_{12}	Il sistema deve offrire la possibilit all'cliente di scaricare un biglietto acquistato	14
RF_{14}	Il sistema deve consentire al cliente la partecipazione ad un evento	15
RF_{15}	Il sistema deve tener traccia dei clienti presenti a ciascun evento	18
RF_{16}	Il sistema deve fornire all'amministratore la possibilità di consultare informazioni aggiuntive per i suoi eventi pubblicati	19

1.4.2 Requisiti sui Dati

ID	Requisito	Origine
RD_{01}	La registrazione consiste nell'inserire nome, cognome, indirizzo e-mail e password	2
RD_{03}	Degli eventi si vuole memorizzare titolo, data, orario, luogo e numero massimo di partecipanti	8
RD_{04}	Ogni biglietto elettronico deve avere un codice identificativo univoco	12

1.4.3 Vincoli/Altri Requisiti

ID	Requisito	Origine
V_{01}	Un biglietto marcato come consumato non può essere più riutilizzato	16
V_{02}	Un cliente durante la fase di acquisto può comprare un solo biglietto ad esso associato	17
V_{03}	Il sistema deve implementare meccanismi di sicurezza per la protezione dei dati personali e per l'autenti- cazione degli utenti	21
V_{04}	Il sistema deve offrire un'interfaccia grafica chiara e intuitiva	22
V_{05}	Il sistema deve garantire l'integrità dei biglietti elettronici	23
V_{06}	Il sistema deve essere accessibile da dispositivi mobili e desktop	24

Capitolo 2

Modellazione dei Casi d'Uso

2.1 Attori e Casi d'Uso

Attori primari
- UtenteNonRegistrato
- UtenteRegistrato
- Utente
- Amministratore

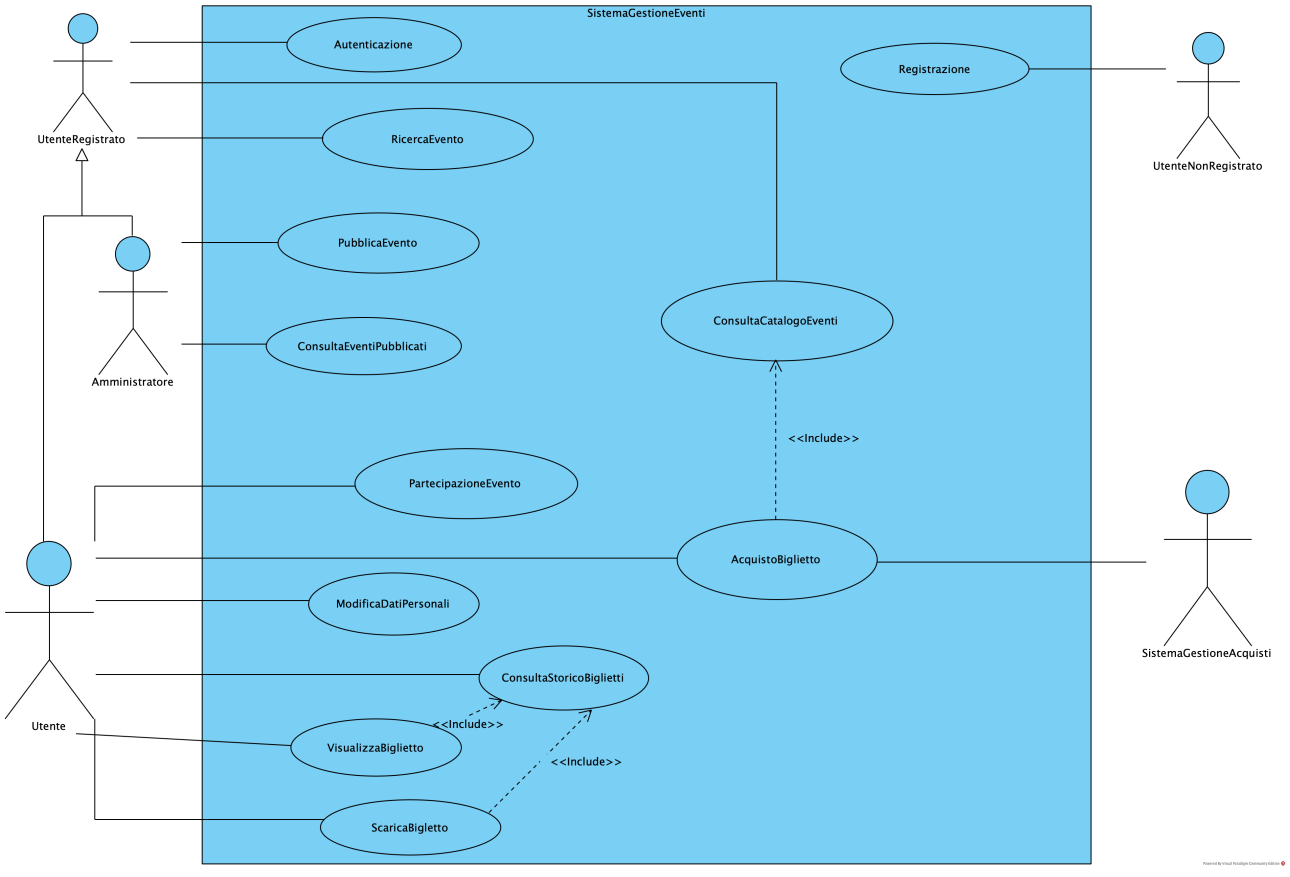
Attori secondari
- SistemaGestioneAcquisti

Casi d'uso
1. Registrazione
2. Autenticazione
3. RicercaEvento
4. PubblicaEvento
5. ConsultaEventiPubblicati
6. PartecipazioneEvento
7. AcquistoBiglietto
8. ModificaDatiPersonali
9. VisualizzaBiglietto
10. ScaricaBiglietto

Casi d'uso di inclusione
1. ConsultaCatalogoEventi
2. ConsultaStoricoBiglietti

Caso d'uso	Attori Primari	Attori Secondari	Incl. / Ext.	Requisiti corrispondenti
Registrazione	UtenteNonRegistrato		–	RF_{01}
Autenticazione	UtenteRegistrato	–	–	RF_{02}
RicercaEvento	UtenteRegistrato	–	–	RF_{08}
PubblicaEvento	Amministratore	–	–	RF_{06}
ConsultaEventiPubblicati	Amministratore	–	Include: Consulta Catalogo Eventi	RF_{19}, RF_{20}, RF_{21}
PartecipazioneEvento	Utente	–	–	RF_{12}
AcquistaBiglietto	Utente	SistemaGestioneAcquisti	Include: Consulta-CatalogoEventi	RF_{09}
ModificaDatiPersonali	Utente	–	–	RF_{05}
VisualizzaBiglietto	Utente	–	–	RF_{10}
ScaricaBiglietto	Utente	–	–	RF_{11}
ConsultaStoricoBiglietti	Utente	–	–	RF_{04}
ConsultaCatalogoEventi	UtenteRegistrato	–	–	RF_{07}

2.2 Diagramma dei Casi d'Uso



2.3 Scenari

Caso d'uso:	**Registrazione**
Attore primario	UtenteNonRegistrato
Attore secondario	–
Descrizione	Un cliente si registra inserendo le proprie credenziali
Pre-condizioni	–
Sequenza di eventi principale	1. Il caso d'uso inizia quando l'UtenteNonRegistrato richiede al sistema di registrarsi 2. Il Sistema richiede le informazioni necessarie per la registrazione: nome, cognome, e-mail, password 3. L'UtenteNonRegistrato inserisce i dati 4. Il Sistema esegue un controllo di validità dei dati inseriti 5. Se il controllo ha successo: 5.1 Il sistema crea un nuovo UtenteRegistrato 6. Altrimenti: 5.1 Il sistema genera un messaggio di errore
Post-condizioni	–
Casi d'uso correlati	–
Sequenza di eventi alternativa	5.1, 6.1

Caso d'uso:	**Autenticazione**
Attore primario	UtenteRegistrato
Attore secondario	–
Descrizione	Un UtenteRegistrato si autentica presso il Sistema
Pre-condizioni	–
Sequenza di eventi principale	1. Il caso d'uso inizia quando l'UtenteRegistrato richiede al Sistema di autenticarsi 2. Il Sistema richiede all'utente di inserire le credenziali per l'autenticazione 3. Il sistema verifica l'esistenza dell'UtenteRegistrato 4. Se il controllo ha successo: 4.1. Il sistema consente l'accesso all'UtenteRegistrato 5. Altrimenti: 5.1. Il sistema mostra un messaggio di errore
Post-condizioni	–
Casi d'uso correlati	–
Sequenza di eventi alternativa	4.1, 5.1

Caso d'uso:	**Consulta Storico Biglietti**
Attore primario	Utente
Attore secondario	-
Descrizione	Un utente visualizza l'elenco di tutti i biglietti acquistati in passato
Pre-condizioni	L'utente deve essersi autenticato
Sequenza di eventi principale	1. L'utente richiede di visualizzare il proprio storico dei biglietti 2. Per ogni biglietto disponibile: 2.1. Il sistema mostra il titolo dell'evento e orario 3. Se non sono disponibili biglietti: 3.1. Il sistema mostra un messaggio di errore
Post-condizioni	-
Casi d'uso correlati	UC10, UC11
Sequenza di eventi alternativa	2.1, 3.1

Caso d'uso:	**Acquista Biglietto**
Attore primario	Utente
Attore secondario	Sistema Gestione Acquisti
Descrizione	L'utente acquista un biglietto digitale per un evento
Pre-condizioni	L'utente ha effettuato l'accesso
Sequenza di eventi principale	1. Il caso d'uso inizia quando l'utente vuole acquistare un biglietto elettronico 2. Include (Consulta Catalogo Eventi) 3. L'utente seleziona un evento e richiede l'acquisto del biglietto 4. Il sistema controlla i posti disponibili per l'evento 5. Se ci sono posti disponibili 5.1. L'utente invia i dati per il pagamento 5.2. Il Sistema invia una richiesta di pagamento al Sistema Gestione Acquisti e aspetta la conferma 5.3. Il sistema crea un biglietto dell'evento e lo associa al profilo personale dell'utente 6. Altrimenti: 6.1. Viene annullato il pagamento
Post-condizioni	Il biglietto è stato generato ed è associato al profilo utente
Casi d'uso correlati	UC12
Sequenza di eventi alternativa	1. Al punto 5.2 se il sistema non riceve conferma viene annullato il pagamento

Caso d'uso:	**PubblicaEvento**
Attore primario	Amministratore
Attore secondario	–
Descrizione	Un amministratore pubblica un nuovo evento
Pre-condizioni	L'amministratore deve essersi autenticato
Sequenza di eventi principale	1. Il caso d'uso inizia quando l'amministratore richiede al sistema di inserire un nuovo evento 2. L'amministratore inserisce i dati dell'evento: titolo,data,orario,luogo,numero massimo di partecipanti 3. Il sistema valida il contenuto dei dati inseriti 4. Se il controllo ha successo: 4.1. Il sistema aggiunge l'evento al catalogo eventi 5. Altrimenti: 5.1. Il sistema mostra un messaggio di errore
Post-condizioni	Il sistema aggiunge l'evento al catalogo
Casi d'uso correlati	-
Sequenza di eventi alternativa	4.1, 5.1

Caso d'uso:	**Partecipazione Evento**
Attore primario	Utente
Attore secondario	–
Descrizione	L'utente partecipa ad un evento specifico
Pre-condizioni	L'utente deve essersi autenticato
Sequenza di eventi principale	1. Il caso d'uso inizia quando l'utente desidera partecipare ad un evento 2. L'utente seleziona l'evento a cui intende partecipare 3. Il sistema richiede il codice del biglietto 4. L'utente inserisce il codice del biglietto 5. Il Sistema esegue delle verifiche sul biglietto inserito 6. Se le verifiche sono soddisfatte: 6.1. Il sistema marca il biglietto come consumato 6.2. Il sistema aggiorna il numero di partecipanti all'evento 6.3. La partecipazione all'evento è consentita all'utente 7. Altrimenti: 7.1. Il sistema mostra un messaggio di errore
Post-condizioni	Lo stato del biglietto viene modificato; viene aggiornato il numero di partecipanti all'evento
Casi d'uso correlati	-
Sequenza di eventi alternativa	6.1, 6.2, 6.3, 7.1

Caso d'uso:	**ConsultaEventiPubblicati**
Attore primario	Amministratore
Attore secondario	–
Descrizione	L'amministratore consulta le informazioni degli eventi da lui pubblicati
Pre-condizioni	L'amministratore deve essersi autenticato
Sequenza di eventi principale	1. Il caso d'uso inizia quando l'amministratore chiede di visualizzare gli eventi pubblicati 2. Per ogni evento pubblicato 2.1. Il sistema mostra il numero degli utenti registrati all'evento. 2.2. Se l'evento è tenuto in data odierna: 2.2.1. Il sistema mostra l'elenco degli utenti che haanno partecipato all'evento 2.3. Altrimenti: 2.3.1. Il sistema mostra il numero degli utenti che hanno partecipato all'evento, senza poter consultare i dati di questi ultimi
Post-condizioni	–
Casi d'uso correlati	–
Sequenza di eventi alternativa	5.1, 6.1

2.4 Diagramma delle Classi

Di seguito riportiamo il diagramma delle classi di analisi.

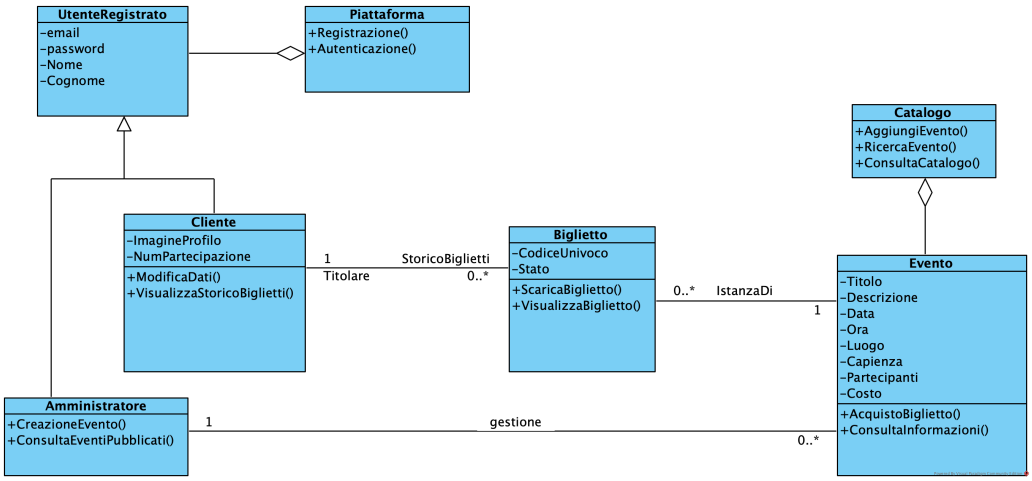


Figura 2.1: Diagramma delle classi di analisi

RESPONSABILITÀ	CLASSE
Registrazione	SistemaGestioneEventi
Autenticazione	SistemaGestioneEventi
ModificaDati	Cliente
VisualizzaStoricoBiglietti	Cliente
CreazioneEvento	Amministratore
ScaricaBiglietto	Biglietto
VisualizzaBiglietto	Biglietto
AcquistoBiglietto	Evento
PartecipazioneEvento	Evento
ConsultaInformazioni	Evento
AggiungiEvento	Catalogo
RicercaEvento	Catalogo

Registrazione e autenticazione sono responsabilità di **SistemaGestioneEventi** in quanto <<information expert >> di *UtenteRegistrato*.

ModificaDati e VisualizzaStoricoBiglietti sono responsabilità di **Cliente** in quanto agiscono su suoi attributi e classi ad esso associate.

AcquistoBiglietto è una responsabilità di **Evento** in quanto <<creator >> di biglietti.

PartecipazioneEvento è una responsabilità di **Evento** seguendo il pattern <<LOW COUPLING >>.

AggiungiEvento è una responsabilità di **Catalogo** poiché, dopo la creazione, l'evento verrà aggiunto al catalogo.

RicercaEvento è una responsabilità di **Catalogo** essendo il contenitore degli eventi.

CreazioneEvento è una responsabilità di **Amministratore** essendo il <<creator >> di Eventi.

2.5 Diagrammi di sequenza

2.5.1 Registrazione

La creazione del seguente diagramma di sequenza, sviluppato a partire dalla descrizione dello scenario del caso d'uso *Registrazione*, ha fatto emergere la necessità di definire un metodo privato specifico della classe **SistemaGestioneEventi**:

controlloEmail(Email)

Tale metodo consente a **SistemaGestioneEventi** di verificare che l'indirizzo email non sia già utilizzato da un altro utente.

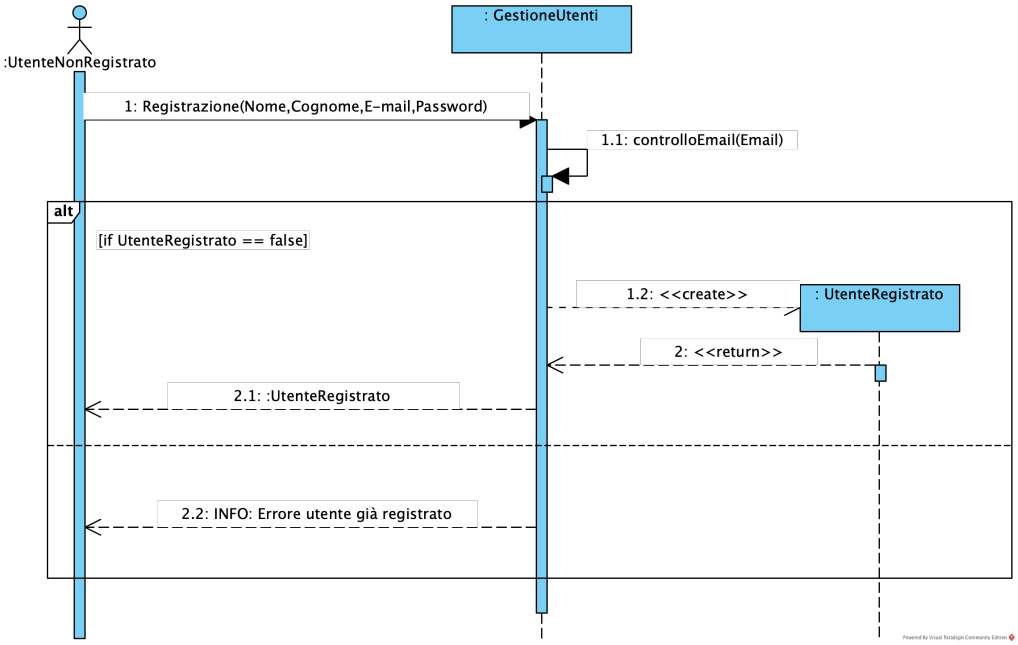


Figura: Diagramma di sequenza per il caso d'uso *Registrazione*

2.5.2 Autenticazione

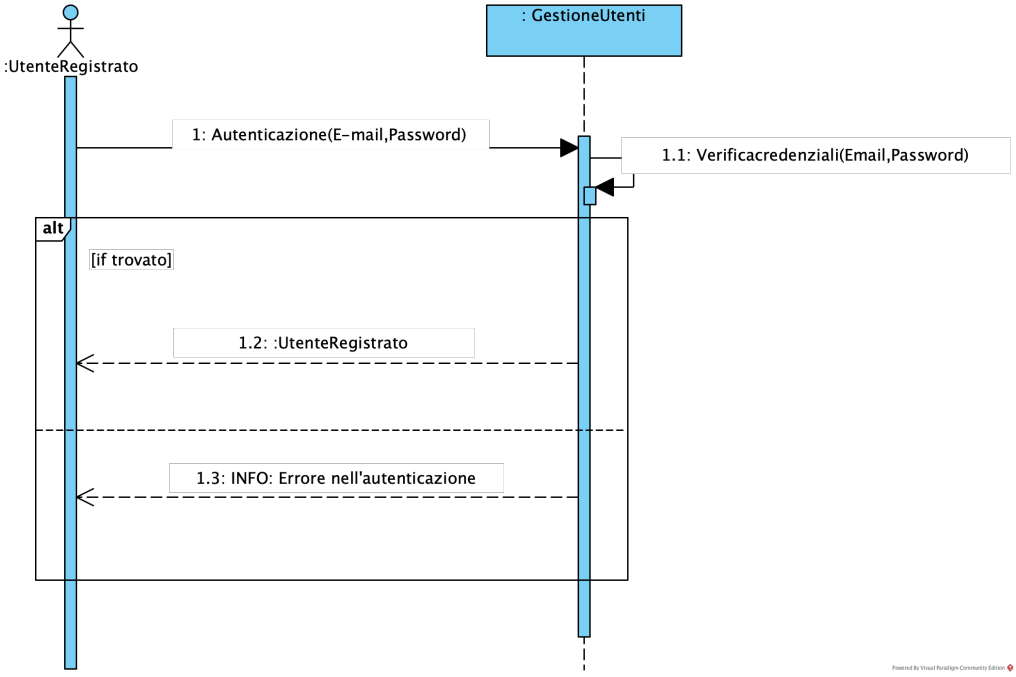


Figura: Diagramma di sequenza per il caso d'uso *Autenticazione*

2.5.3 PubblicaEvento

La creazione del seguente diagramma di sequenza, sviluppato a partire dalla descrizione dello scenario del caso d'uso *PubblicaEvento*, ha fatto emergere la necessità di definire un metodo privato, specifico della classe **Amministratore**:

verificaValidità(Titolo, Descrizione, Data, Orario, Luogo, NumMassimoPartecipanti)

Tale metodo consente all'amministratore di verificare che i dati inseriti siano validi prima della pubblicazione di un evento.

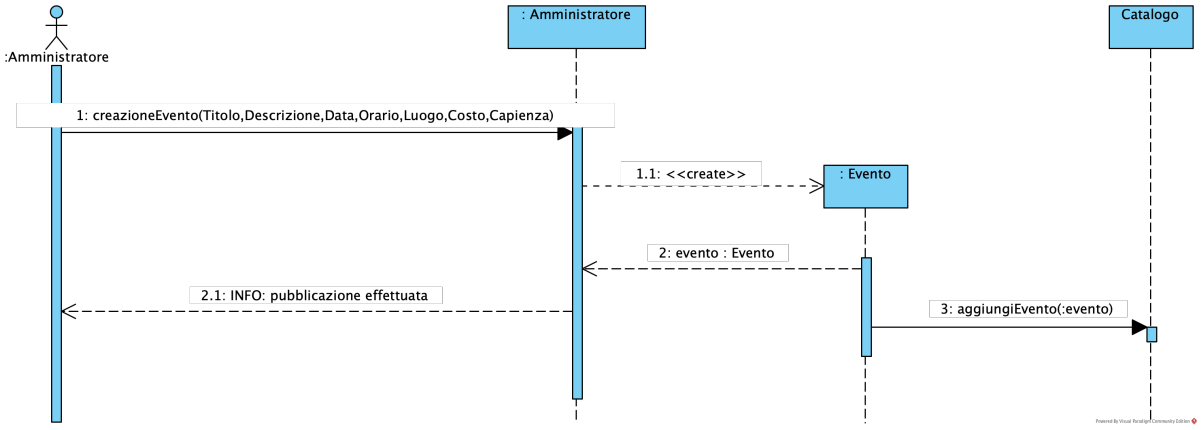


Figura: Diagramma di sequenza per il caso d'uso *PubblicaEvento*

2.5.4 ConsultaEventiPubblicati

Il diagramma di sequenza del caso d'uso *ConsultaEventiPubblicati* mostra la necessità di una responsabilità **getListaPartecipanti()** da parte dell'evento

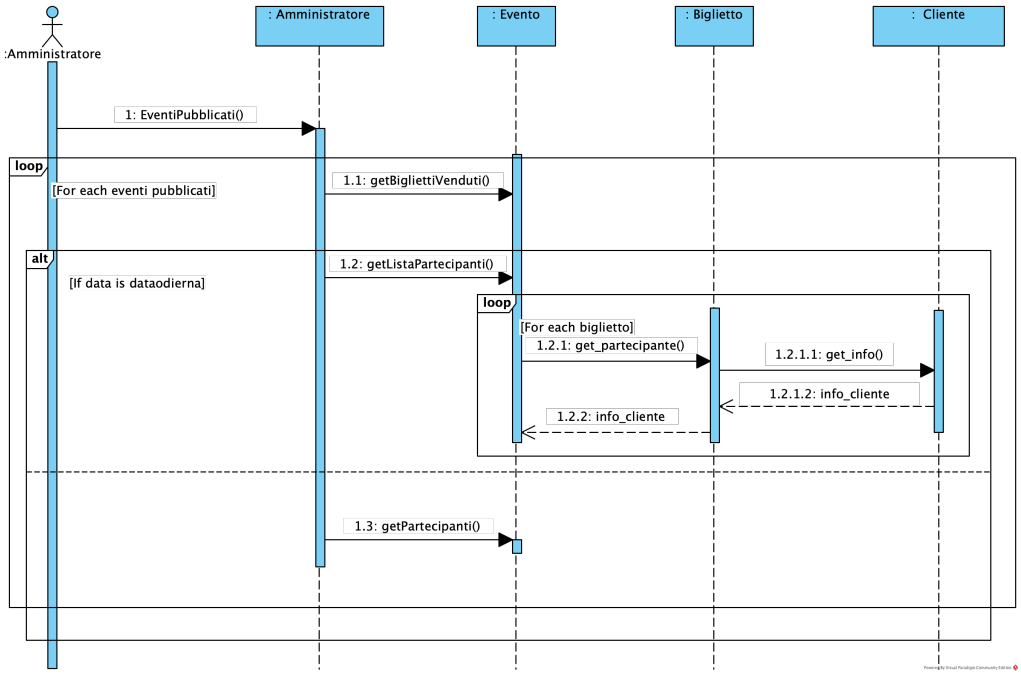


Figura: Diagramma di sequenza per il caso d'uso *ConsultaEventiPubblicati*

2.5.5 AcquistoBiglietti

Il diagramma di sequenza del caso d'uso *AcquistoBiglietto* mostra il processo di acquisto da parte dell'utente. Durante questo flusso emergono tre nuove responsablità per la classe Evento: **verificaDisponibilità()**, **creazioneIdUnivoco()** e **InviaDatiPagamento(Nome, Cognome, Email, informazioniCarta)**.

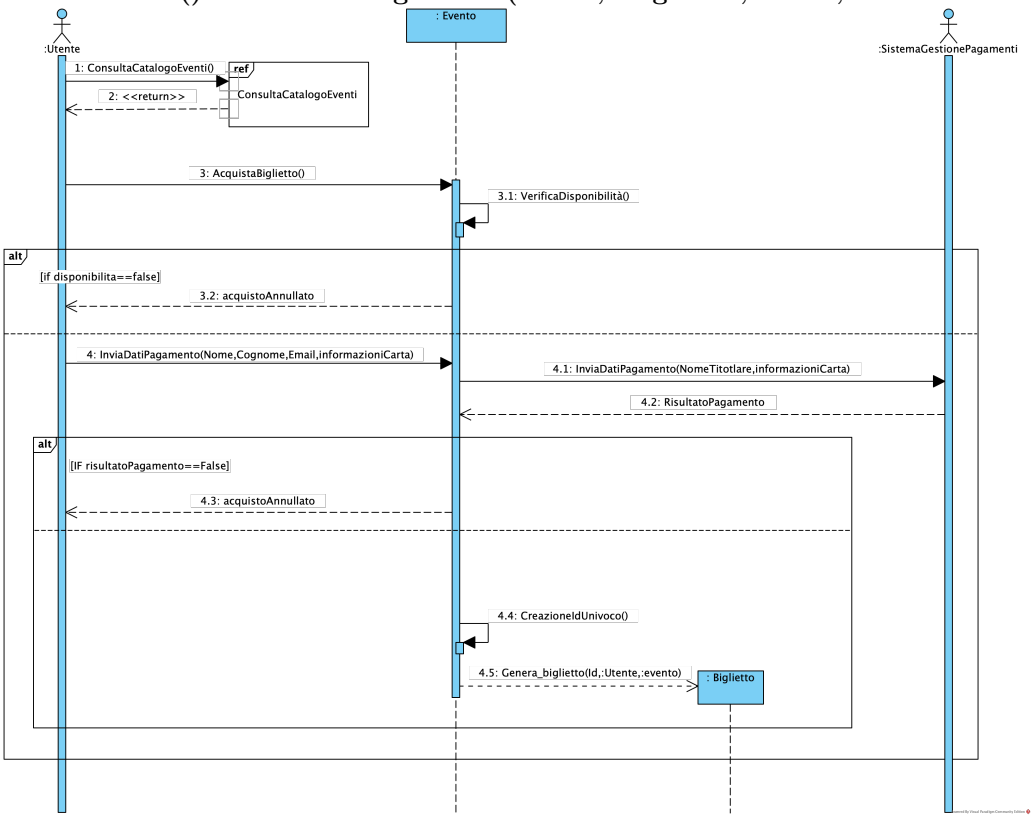
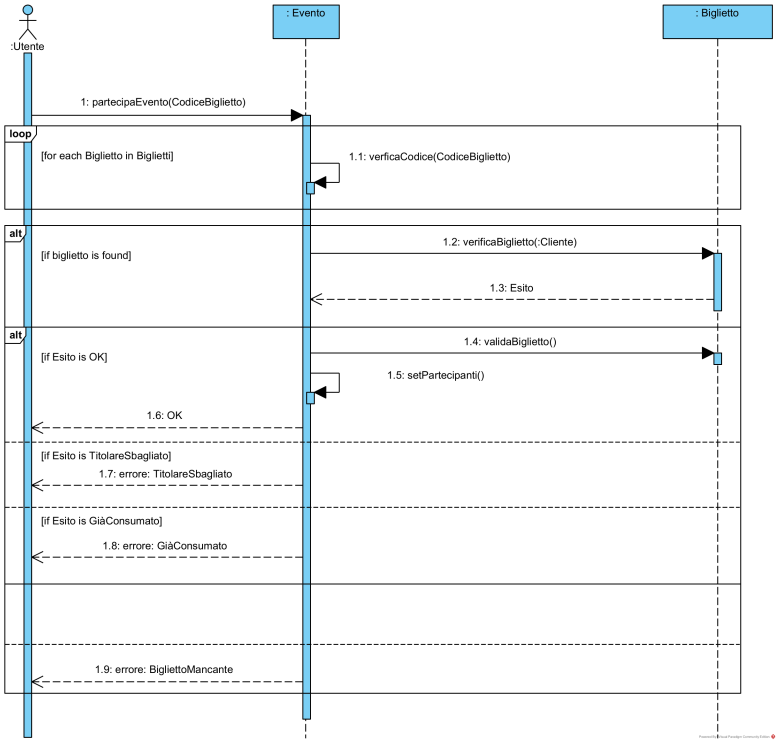


Figura: Diagramma di sequenza per il caso d'uso *AcquistoBiglietto*

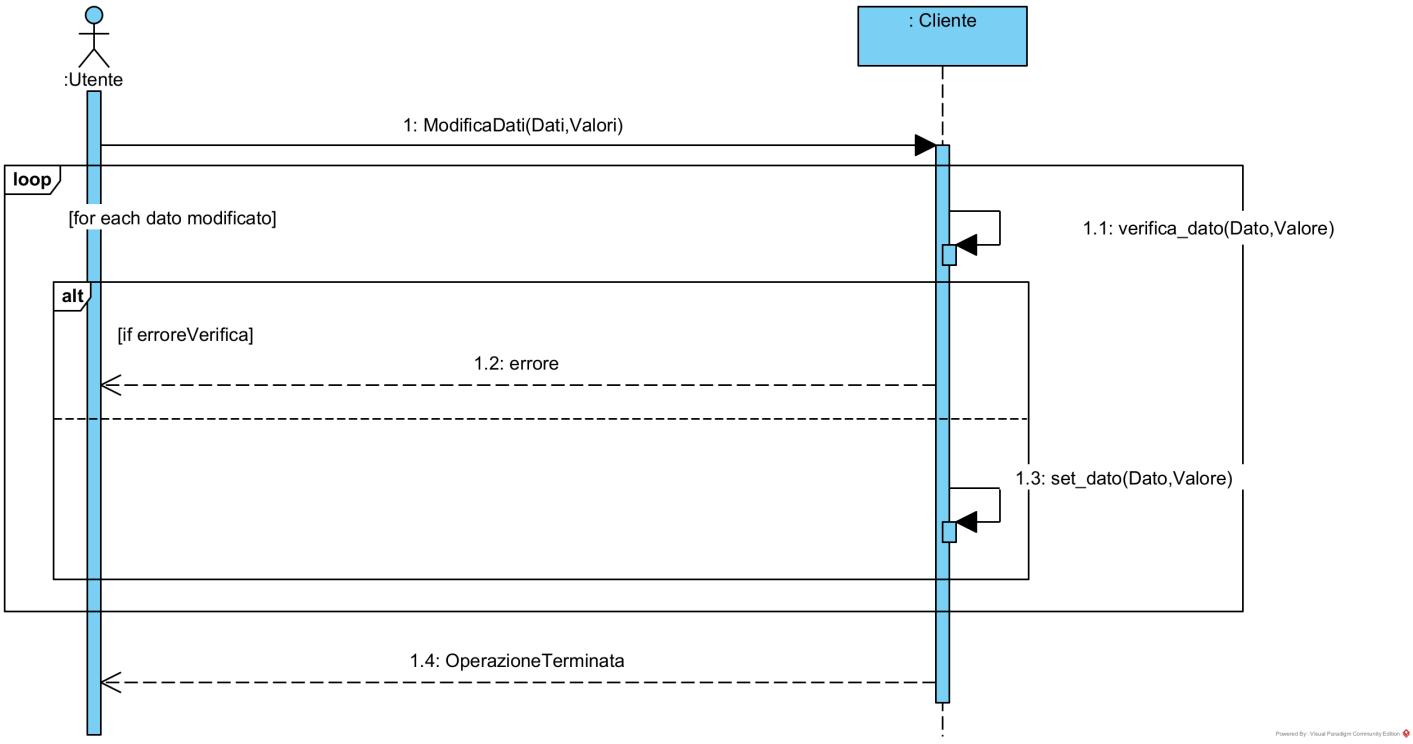
2.5.6 PartecipazioneEvento

Il diagramma di sequenza del caso d'uso *PartecipazioneEvento* evidenzia i passi necessari durante la partecipazione ad un evento da parte di un utente. E' stato necessario dunque aggiungere, come visibile dal diagramma, un nuovo metodo nella classe Evento: **verificaCodice()**, ed un nuovo metodo nella classe Biglietto: **validaBiglietto()**

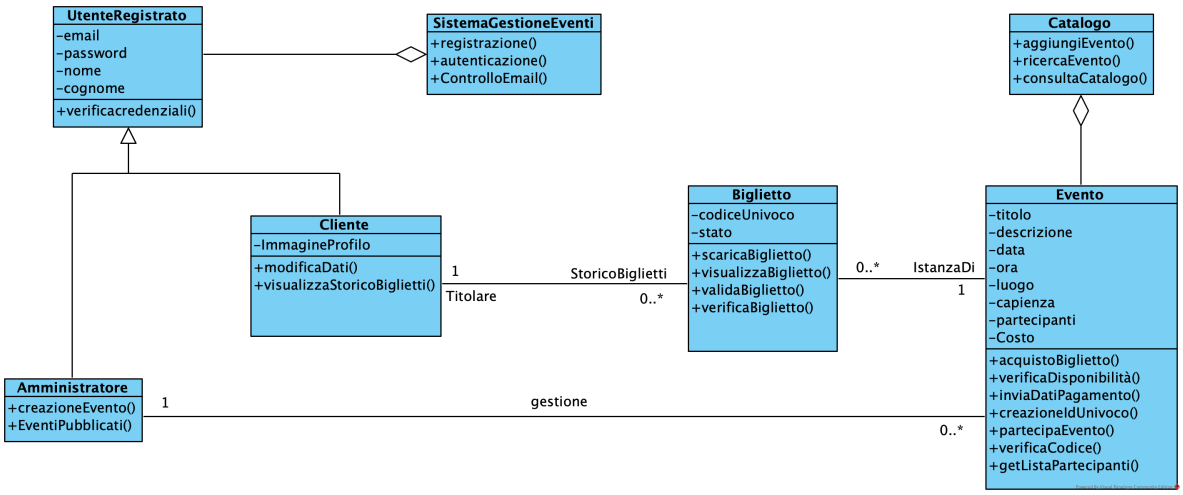


2.5.7 ModificaDatiPersonali

Come visibile dal sequence diagram del caso d'uso *ModificaDatiPersonali* è stato necessario fornire alla classe **Cliente** un metodo privato, di istanza **VerificaDato(Dato,Valore)**



2.6 Diagramma delle classi raffinato



Capitolo 3

Piano di test funzionale

Si intende progettare i casi di test funzionale con la tecnica del Category Partition Testing.

3.1 Registrazione

Nome	Cognome	Email	Password
• Stringa di lunghezza $\leq$ 20 • Stringa di lunghezza $>$ 20 `[ERROR]` • Stringa vuota `[ERROR]`	• Stringa di lunghezza $\leq$ 30 • Stringa di lunghezza $>$ 30 `[ERROR]` • Stringa vuota `[ERROR]`	• Stringa di lunghezza $\leq$ 50 • Stringa con formato diverso da [esempio@dominio.estensione] `[ERROR]` • Stringa di lunghezza $>$ 50 `[ERROR]` • Stringa vuota `[ERROR]` • Stringa già memorizzata `[ERROR]`	• Stringa di lunghezza $\leq$ 40 • Stringa di lunghezza $>$ 40 `[ERROR]` • Stringa vuota `[ERROR]` • Stringa senza caratteri speciali `[ERROR]`

Il numero di test da effettuarsi senza particolari vincoli è: $3 \cdot 3 \cdot 5 \cdot 4 = 180$.

Introduciamo i vincoli `[ERROR]`. Il numero di test da eseguire per testare singolarmente i vincoli è 11 (2 per Nome, 2 per Cognome, 4 per Email, 3 per Password).

Il numero di test risultante è 11: $(1 \cdot 1 \cdot 1 \cdot 1) + 11 = 12$.

ì

Tabella 3.1: Casi di test registrazione

Test Case ID	Descrizione	Classi di Equivalenza Coperte	Pre-condizioni	Input	Output Attesi	Post-condizioni Attese
1	Tutti gli input validi	Nome, Cognome, Email, Password validi	L'utente non deve essere registrato	Nome: Mario, Cognome: Rossi, Email: mario.rossi@email.com, Password: miaPwd£123	Registrazione completata	Utente correttamente registrato nel sistema
2	Nome $>$ 20 caratteri	Nome $>$ 20 caratteri [ERROR], altri validi	L'utente non deve essere registrato	Nome: ..., Cognome: Rossi, Email: mario.rossi@email.com, Password: miaPwd£123	Nome troppo lungo	–

Continued on next page

Tabella 3.1: Casi di test registrazione (Continued)

Test Case ID	Descrizione	Classi di Equivalenza Coperte	Pre-condizioni	Input	Output Attesi	Post-condizioni Attese
3	Nome vuoto	Nome vuoto [ERROR], altri validi	L'utente non deve essere registrato	Nome: (vuoto), Cognome: Rossi, Email: mario.rossi@email.com, Password: miaPwd£123	Inserire un nome	–
4	Cognome > 30 caratteri	Cognome > 30 caratteri [ERROR], altri validi	L'utente non deve essere registrato	Nome: Mario, Cognome: ..., Email: mario.rossi@email.com, Password: miaPwd£123	Cognome troppo lungo	–
5	Cognome vuoto	Cognome vuoto [ERROR], altri validi	L'utente non deve essere registrato	Nome: Mario, Cognome: (vuoto), Email: mario.rossi@email.com, Password: miaPwd£123	Inserire un cognome	–
6	Email > 50 caratteri	Email > 50 caratteri [ERROR], altri validi	L'utente non deve essere registrato	Nome: Mario, Cognome: Rossi, Email: ..., Password: miaPwd£123	Email troppo lunga	–
7	Email vuota	Email vuota [ERROR], altri validi	L'utente non deve essere registrato	Nome: Mario, Cognome: Rossi, Email: (vuoto), Password: miaPwd£123	Inserire un indirizzo email	–
8	Email già registrata	Email già memorizzata [ERROR], altri validi	Email presente nel sistema	Nome: Mario, Cognome: Rossi, Email: cliente.esistente@email.com, Password: miaPwd£123	Email già registrata	–
9	Password > 40 caratteri	Password > 40 caratteri [ERROR], altri validi	L'utente non deve essere registrato	Nome: Mario, Cognome: Rossi, Email: mario.rossi@email.com, Password: ...	Password troppo lunga	–
10	Password vuota	Password vuota [ERROR], altri validi	L'utente non deve essere registrato	Nome: Mario, Cognome: Rossi, Email: mario.rossi@email.com, Password: (vuoto)	Inserire una password	–
11	Password senza caratteri speciali	Password senza caratteri speciali [ERROR], altri validi	L'utente non deve essere registrato	Nome: Mario, Cognome: Rossi, Email: mario.rossi@email.com, Password: miapassword123	Password deve contenere caratteri speciali	–

3.2 Autenticazione

Email	Password
Stringa di lunghezza ≤ 50 Stringa con formato diverso da [esempio@dominio.estensione] `[ERROR]` Stringa di lunghezza > 50 `[ERROR]` Stringa di lunghezza < 0 `[ERROR]` Stringa non memorizzata `[ERROR]`	Stringa di lunghezza ≤ 40 Stringa di lunghezza > 40 `[ERROR]` Stringa di lunghezza $= 0$ `[ERROR]` Stringa senza caratteri speciali `[ERROR]`

Tabella 3.2: Category Partition Testing - Autenticazione

Il numero di test da effettuarsi senza particolari vincoli è: $5 \cdot 4 = 20$.

Con i vincoli [ERROR], invece, il numero di test da eseguire per testare singolarmente i vincoli è 7 (4 per Email, 3 per Password).

Il numero di test risultante è 7: $(1 \cdot 1 \cdot 1 \cdot 1) + 7 = 8$.

Tabella 3.3: Test Suite - Autenticazione

ID	Descrizione	Classi di Equivalenza Coperte	Pre-condizioni	Input	Output Attesi	Post-condizioni Attese
1	Tutti gli input validi	Email, Password validi	L'utente deve essere registrato	Email: marco bianchi@ gmail.com, Password: miapwd£	Autenticazione effettuata	L'cliente è entrato correttamente nel sistema
2	Email > 50 caratteri	Email > 50 caratteri `[ERROR]`, Password	L'utente deve essere registrato	Email molto lunga (¿50 char), Password: miapwd£	Email troppo lunga	–
3	Email senza caratteri	Email con 0 caratteri `[ERROR]`, Password	L'utente deve essere registrato	Email: "", Password: miapwd£	Inserire un indirizzo Email	–
4	Email non memorizzata	Email non presente nel sistema `[ERROR]`, Password	–	Email: mario rosssi@ gmail.com, Password: miaPWD£	Account non registrato	–
5	Password > 40 caratteri	Email, Password > 40 caratteri `[ERROR]`	–	Email: marco bianchi@ gmail.com, Password molto lunga (¿40 char)	Password troppo lunga	–
6	Password senza caratteri	Email, Password senza caratteri `[ERROR]`	–	Email: marco bianchi@ gmail.com, Password: ""	Inserire una password	–
7	Password senza caratteri speciali	Email, Password senza caratteri speciali `[ERROR]`	–	Email: marco bianchi@ gmail.com, Password: miapwd	Inserire una password valida	–

3.3 PubblicaEvento

Titolo	Descrizione	Data	Orario	Luogo	Capienza
Stringa di lunghezza ≤ 50 Stringa di lunghezza > 50 `[ERROR]` Stringa di lunghezza < 0 `[ERROR]` Stringa già memorizzata `[ERROR]`	Stringa di lunghezza ≤ 150 Stringa di lunghezza > 150 `[ERROR]` Stringa di lunghezza < 0 `[ERROR]`	Data con formato valido (gg-mm-aaaa) Data con formato non valido `[ERROR]`	Orario con formato valido (oo-mm) Orario con formato non valido `[ERROR]`	Stringa non contenente caratteri speciali Stringa contenente caratteri speciali `[ERROR]` Stringa contenente numeri `[ERROR]`	Intero di valore massimo 100 Intero di valore > 100 `[ERROR]`

Tabella 3.4: Category Partition Testing - PubblicaEvento

Il numero di test da effettuarsi senza particolari vincoli è: $4 \cdot 3 \cdot 2 \cdot 2 \cdot 3 \cdot 2 = 288$.

Con i vincoli [ERROR], invece, il numero di test da eseguire per testare singolarmente i vincoli è 11 (3 per Titolo, 2 per Descrizione, 1 per Data, 1 per Orario, 2 per Luogo, 1 per NumMassimoPartecipanti).

Il numero di test risultante è 12: $(1 \cdot 1 \cdot 1 \cdot 1 \cdot 1 \cdot 1) + 11 = 12$.

Tabella 3.5: Test Suite - PubblicaEvento

ID	Descrizione	Classi di Equivalenza Coperte	Pre-condizioni	Input	Output Attesi	Post-condizioni Attese
1	Tutti gli input validi	Tutti i parametri validi	Utente autenticato	Titolo: Concerto Rock, Descrizione: Evento musicale serale, Data: 15-06-2025, Orario: 20-30, Luogo: Teatro Comunale, NumMax: 50	Evento pubblicato con successo	L'evento viene correttamente aggiunto al catalogo
2	Titolo > 50 caratteri	Titolo > 50 caratteri `[ERROR]`	Utente autenticato	Titolo molto lungo (¿50 char), altri parametri validi	Titolo troppo lungo	–
3	Titolo vuoto	Titolo con 0 caratteri `[ERROR]`	Utente autenticato	Titolo: "", altri parametri validi	Inserire un titolo	–
4	Titolo già esistente	Titolo già memorizzato `[ERROR]`	Utente autenticato	Titolo: Concerto Rock (esistente), altri parametri validi	Titolo già utilizzato	–
5	Descrizione > 150 caratteri	Descrizione > 150 caratteri `[ERROR]`	Utente autenticato	Descrizione molto lunga (¿150 char), altri parametri validi	Descrizione troppo lunga	–
6	Descrizione vuota	Descrizione con 0 caratteri `[ERROR]`	Utente autenticato	Descrizione: "", altri parametri validi	Inserire una descrizione	–
7	Data formato non valido	Data con formato non valido `[ERROR]`	Utente autenticato	Data: 2025/06/15, altri parametri validi	Formato data non valido	–
8	Orario formato non valido	Orario con formato non valido `[ERROR]`	Utente autenticato	Orario: 20:30, altri parametri validi	Formato orario non valido	–
9	Luogo con caratteri speciali	Luogo con caratteri speciali `[ERROR]`	Utente autenticato	Luogo: Teatro#Comunale, altri parametri validi	Caratteri non consentiti nel luogo	–
10	Luogo con numeri	Luogo con numeri `[ERROR]`	Utente autenticato	Luogo: Teatro123, altri parametri validi	Numeri non consentiti nel luogo	–
11	NumMassimo > 100	NumMassimo > 100 `[ERROR]`	Utente autenticato	NumMax: 150, altri parametri validi	Numero massimo troppo alto	–
12	Utente non autenticato	Utente non autenticato	Utente non loggato	Tutti i parametri validi	Accesso negato	–

3.4 RicercaEvento

Titolo	Data	Luogo
Stringa di lunghezza ≤ 50 Stringa di lunghezza > 50 `[ERROR]` Stringa di lunghezza < 0 `[ERROR]` Stringa già memorizzata `[ERROR]`	Data con formato valido (gg-mm-aaaa) Data con formato non valido `[ERROR]`	Stringa non contenente caratteri speciali Stringa contenente caratteri speciali `[ERROR]` Stringa contenente numeri `[ERROR]`

Il numero di test da effettuarsi senza particolari vincoli è: $4 \cdot 2 \cdot 3 = 24$.

Con i vincoli `[ERROR]`, invece, il numero di test da eseguire per testare singolarmente i vincoli è 6 (3 per Titolo, 1 per Data, 2 per Luogo).

Il numero di test risultante è 7: $(1 \cdot 1 \cdot 1) + 6 = 7$.

Tabella 3.6: Casi di test ricerca evento

Test Case ID	Descrizione	Classi di Equivalenza Coperte	Pre-condizioni	Input	Output Attesi	Post-condizioni Attese
1	Tutti gli input validi	Titolo, Data, Luogo validi	Esiste almeno un evento che corrisponde alla ricerca	Titolo: Concerto, Data: 15-06-2025, Luogo: Teatro Comunale	Eventi trovati	–
2	Titolo > 50 caratteri	Titolo > 50 caratteri [ERROR], altri validi	–	Titolo: Festival internazionale della musica contemporanea elettronica, Data: 15-06-2025, Luogo: Teatro Comunale	Titolo troppo lungo	–
3	Titolo vuoto	Titolo vuoto [ERROR], altri validi	–	Titolo: (vuoto), Data: 15-06-2025, Luogo: Teatro Comunale	Inserire un titolo	–
4	Titolo già memorizzato	Titolo già memorizzato [ERROR], altri validi	Titolo già presente nel DB	Titolo: Concerto Rock, Data: 15-06-2025, Luogo: Teatro Comunale	Titolo già utilizzato	–
5	Data formato non valido	Data con formato non valido [ERROR], altri validi	–	Titolo: Concerto, Data: 2025/06/15, Luogo: Teatro Comunale	Formato data non valido	–
6	Luogo con caratteri speciali	Luogo con caratteri speciali [ERROR], altri validi	–	Titolo: Concerto, Data: 15-06-2025, Luogo: Teatro#Comunale	Caratteri non consentiti nel luogo	–
7	Luogo con numeri	Luogo con numeri [ERROR], altri validi	–	Titolo: Concerto, Data: 15-06-2025, Luogo: Teatro123	Numeri non consentiti nel luogo	–

3.5 Acquista Biglietto

Posti Disponibili	Dati Pagamento
Posti Disponibili Posti Esauriti `[ERROR]`	Dati completi e validi Dati errati (es. carta scaduta) `[ERROR]` Errore conferma dal sistema gestione acquisti `[ERROR]`

Il numero di test da effettuarsi senza particolari vincoli è: $2 \cdot 3 \cdot 2 = 12$.

Con i vincoli `[ERROR]`, invece, il numero di test da eseguire per testare singolarmente i vincoli è 4 (1 per Posti disponibili, 2 per Dati pagamento, 1 per Sistema gestione acquisti).

Il numero di test risultante è 5: $(1 \cdot 1 \cdot 1) + 4 = 5$.

Tabella 3.7: Casi di test Acquista Biglietto

Test Case ID	**Descrizione**	**Classi di Equivalenza Coperte**	**Precondizioni**	**Input**	**Output Attesi**	**Postcondizioni Attese**
1	Posti disponibili, dati pagamento validi	Posti disponibili > 0, dati pagamento validi	Utente autenticato	Evento con posti disponibili, dati pagamento corretti	Biglietto creato	Il biglietto è associato al cliente
2	Posti esauriti	Posti esauriti $= 0$	Utente autenticato, evento esistente	Evento senza posti disponibili, dati pagamento validi	Messaggio di posti esauriti, acquisto annullato	Nessun biglietto creato
3	Dati pagamento incompleti	Posti disponibili > 0, dati pagamento incompleti `[ERROR]`	Utente autenticato, evento esistente	Evento con posti disponibili, dati pagamento incompleti	Errore di pagamento: dati incompleti	Nessun biglietto creato
4	Dati pagamento errati (es. carta scaduta)	Posti disponibili > 0, dati pagamento errati `[ERROR]`	Utente autenticato, evento esistente	Evento con posti disponibili, dati pagamento errati	Errore di pagamento: dati errati	Nessun biglietto creato
5	Errore sistema gestione acquisti	Posti disponibili > 0, errore conferma pagamento `[ERROR]`	Utente autenticato, evento esistente	Evento con posti disponibili, pagamento inviato	Errore di sistema: pagamento non confermato	Nessun biglietto creato

3.6 PartecipaEvento

Codice Biglietto	**Stato Biglietto**
Codice biglietto = xxxx-123-ABC Codice biglietto $\neq$ xxxx-123-ABC `[ERRORE]`	Stato biglietto = [Non consumato] Stato biglietto = [Consumato] `[ERROR]`

Il numero di test da effettuarsi senza particolari vincoli è: $2 \cdot 2 = 4$.
Con i vincoli `[ERROR]`, invece, il numero di test da eseguire per testare singolarmente i vincoli è 2 (1 per Codice Biglietto, 1 per Stato biglietto).
Il numero di test risultante è 5: $(1 \cdot 1 \cdot 1) + 2 = 3$.

Tabella 3.8: Test Suite - PartecipaEvento

Test Case ID	**Descrizione**	**Classi di Equivalenza Coperte**	**Precondizioni**	**Input**	**Output Attesi**	**Postcondizioni Attese**
1	Partecipazione valida	Codice corretto e biglietto non consumato	Utente autenticato, biglietto esistente	Codice: xxx-123-ABC, Stato: Non consumato	Partecipazione registrata con successo	Biglietto marcato come consumato
2	Codice biglietto non valido	Codice biglietto $\neq$ xxxx-123-ABC `[ERROR]`	Utente autenticato	Codice: yyy-456-DEF, Stato: Non consumato	Codice biglietto non valido	–

Continued on next page

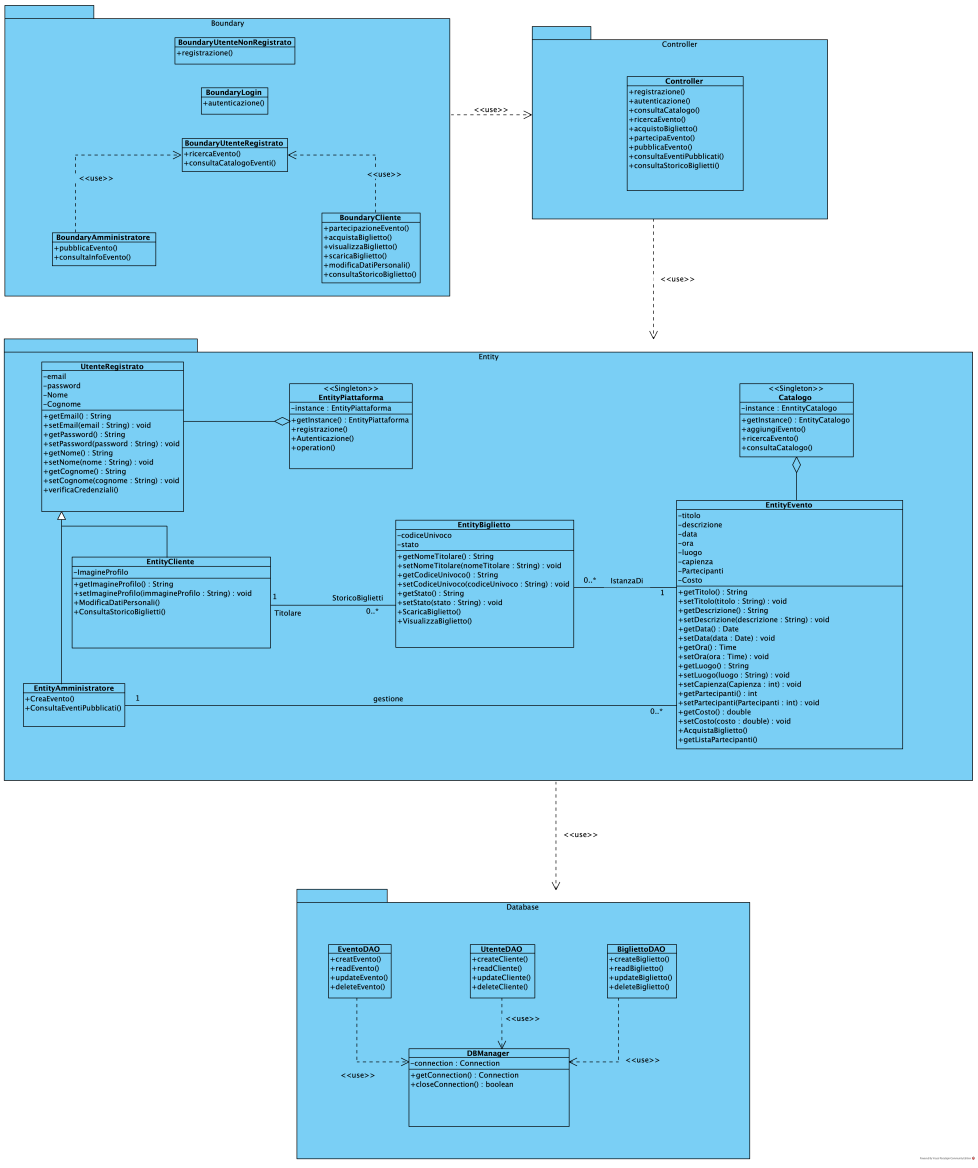
Tabella 3.8: Test Suite - PartecipaEvento (Continued)

Test Case ID	**Descrizione**	**Classi di Equivalenza Coperte**	**Precondizioni**	**Input**	**Output Attesi**	**Postcondizioni Attese**
3	Biglietto già consumato	Stato biglietto = Consumato `[ERROR]`	Utente autenticato, biglietto esistente	Codice: xxx-123-ABC, Stato: Consumato	Biglietto già utilizzato	–

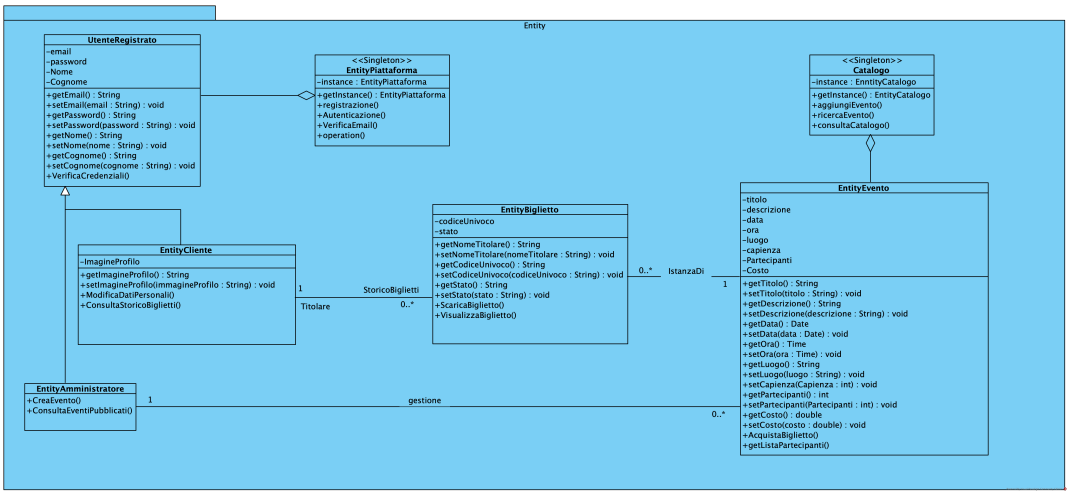
Capitolo 4

Progettazione

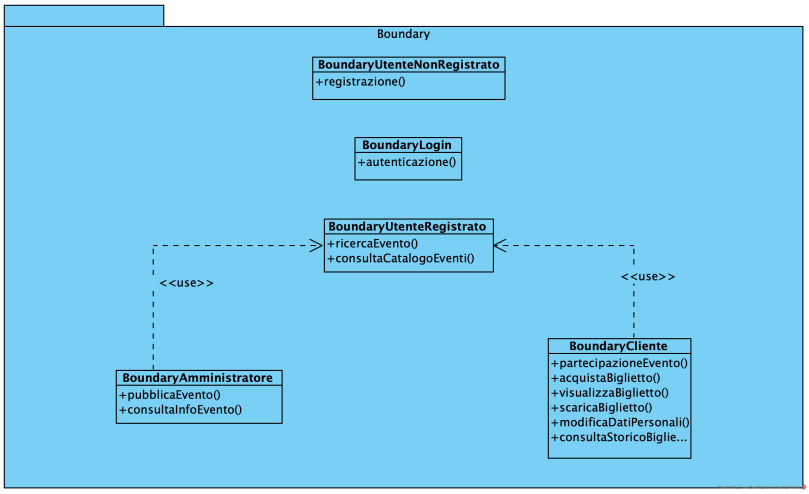
4.1 Diagramma delle classi



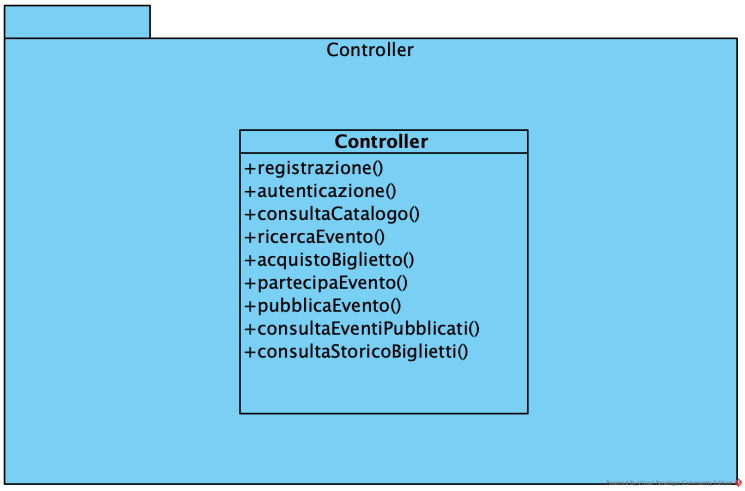
4.1.1 Package Entity



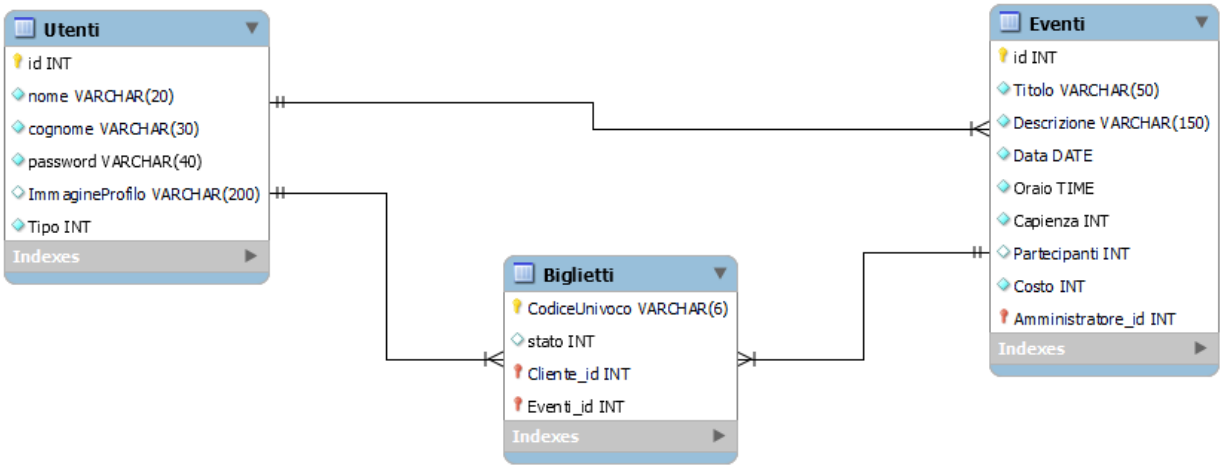
4.1.2 Package Boundary

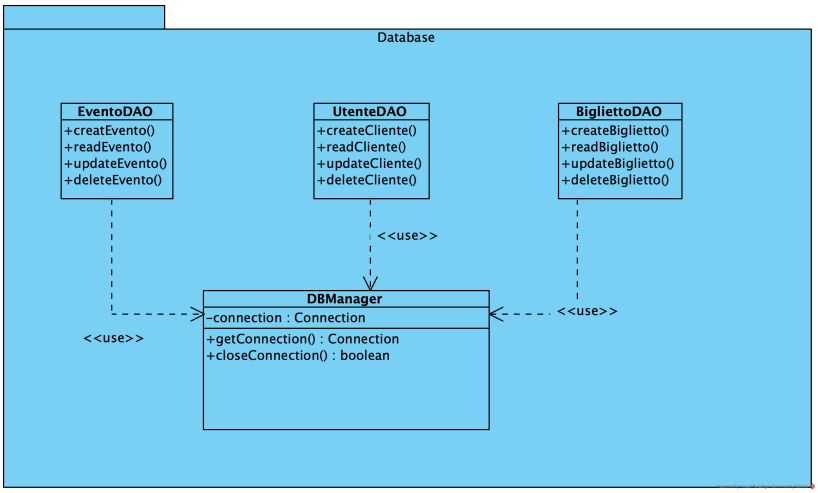


4.1.3 Package Controller



4.1.4 Package Database





Il modello E-R riportato rappresenta le principali entità e relazioni del sistema, progettato utilizzando MySQL Workbench.

In fase di progettazione si è scelto di rappresentare tutte e due i ruoli degli utenti all'interno di un'unica tabella denominata 'Utente'. All'interno di questa tabella è stato inserito un attributo aggiuntivo, chiamato 'Ruolo', che consente di discriminare i due tipi di utente.

4.2 Diagrammi di sequenza

aspetto

Capitolo 5

Implementazione

5.1 Package Boundary

```
boundary/
├── BoundaryAmministratore/
│   ├── FormEvento
│   └── HomeAmministratore
├── BoundaryCliente/
│   ├── FormAcquistoBiglietto
│   ├── FormStoricoBiglietti
│   └── HomeCliente
├── BoundaryUtente/
│   ├── BoundaryRegistrazione
│   └── HomePage
├── BoundaryUtenteRegistrato/
│   ├── BoundaryAutenticazione
│   ├── CatalogoEvento
│   ├── FormRicercaEvento
│   └── HomeUtenteRegistrazione
└── Sessione
```

5.2 Package Database

```
database/
├── BigliettoDAO.java
├── DBConnectionManager.java
├── EventoDAO.java
└── UtenteDAO.java
```

5.3 Package Entity

```
entity/
├── EntityAmministratore.java
├── EntityBiglietto.java
├── EntityCatalogo.java
├── EntityCliente.java
├── EntityEvento.java
├── EntityPiattaforma.java
├── EntitySistemaGestioneAcquisti.java
└── EntityUtenteRegistrato.java
```

5.4 Package Exception

```
exceptions/
├── AcquistoException.java
├── BigliettoConsumatoException.java
├── BigliettoNotFoundException.java
├── DBException.java
├── EventoNotFoundException.java
├── LoadingException.java
├── LoginFailedException.java
├── RedundancyException.java
├── RegistrationFailedException.java
├── UniqueCodeException.java
├── UpdateException.java
├── UtenteNotFoundException.java
└── WrongUserTypeException.java
```

5.5 Package Dto

```
DTO/
├── DTOBiglietto.java
├── DTOEvento.java
└── DTOutente.java
```

5.6 Dipendenze per l'esecuzione ed il funzionamento dell'applicazione

Per il corretto funzionamento dell'applicazione sono necessarie le seguenti risorse:

1. Oracle OpenJDK 24.0.1
2. Oracle MySQL Server 8.0.33
3. mysql-connector-j-8.0.33.jar
4. swingx-1.6.1

5.7 Documentazione Javadoc

La documentazione Javadoc dell'applicazione è presente nella directory **/javadoc/.**

5.8 Diagramma di Deployment