

Università degli Studi di Salerno
Corso di Ingegneria del Software

BEAT – Booking Events And Tickets

Requirement Analysis Document

Versione 1.0



Data: 23/10/2025

Coordinatore del progetto: Prof. Andrea De Lucia

Partecipanti: Carnevale Luigi[0512119029], Di Manso Carmine[0512119521], Clemente Manuel[0512119395]

Scritto da: Carnevale Luigi, Di Manso Carmine, Clemente Manuel

Repository GitHub: [github.com/Luigi-Carnevale/BEAT-Booking Events And Tickets](https://github.com/Luigi-Carnevale/BEAT-Booking-Events-And-Tickets)

Revision History

Data	Versione	Descrizione	Autore
23-10-2025	1.0	Creazione RAD	Carnevale Luigi, Di Manso Carmine, Clemente Manuel

Indice

Revision History

1	Introduction	1
1.1	purpose of the system	1
1.2	Scope of the system	1
1.3	Objectives and success criteria of the project	3
1.4	Definitions, acronyms, and abbreviations	4
1.5	References	4
1.6	Overview	4
2	Current system	5
3	Proposed system	5
3.1	Overview	5
3.2	Functional requirements	5
3.3	Nonfunctional requirements	6
3.3.1	Usability	6
3.3.2	Reliability	6
3.3.3	Performance	6
3.3.4	Supportability	6
3.3.5	Implementation	6
3.3.6	Interface	6
3.3.7	Security	6
3.3.8	Operation	7
3.3.9	Packaging	7
3.3.10	Legal	7
3.4	System models	8
3.4.1	Scenarios	8
3.4.2	Use case model	11
3.4.3	*Object model*	13
3.4.4	*Dynamic model*	13
3.4.5	User interface-navigational paths and screen mock-ups	13

1 Introduction

1.1 purpose of the system

L'obiettivo del progetto BEAT è sviluppare un sistema con interfaccia web per la gestione e prenotazione di eventi, che consenta agli utenti di consultare, prenotare e gestire biglietti per una varietà di eventi (concerti, seminari, corsi, spettacoli). Il sistema intende fornire una piattaforma intuitiva e dinamica che semplifichi sia la gestione amministrativa degli eventi sia l'esperienza dell'utente finale nella consultazione e prenotazione.

1.2 Scope of the system

BEAT è una piattaforma web progettata per offrire agli utenti un'esperienza semplice e sicura nell'acquisto di biglietti per eventi di vario tipo. L'obiettivo principale del sistema è garantire agli utenti la possibilità di cercare, selezionare e acquistare biglietti in modo facile ed efficiente, gestendo al contempo le informazioni sugli eventi, la disponibilità dei biglietti e le transazioni in modo sicuro. La piattaforma web è sviluppata secondo il modello architettonico **MVC**. La **View** è realizzata con **JSP** per la gestione dell'interfaccia utente dinamica, mentre il **Model** interagisce con il database tramite **DAO** e **JDBC**. La logica di business è gestita da **Servlet**. Per garantire la sicurezza degli utenti, il sistema adotta un sistema di hashing delle password utilizzando un algoritmo sicuro con un output di 60 caratteri, proteggendo così i dati sensibili di tutti gli utenti registrati.

Architettura del Sistema:

Il sistema BEAT è costruito secondo il modello **MVC**, che separa la logica applicativa, la presentazione e il controllo dei dati. Questo approccio garantisce una struttura scalabile, manutenibile e facilmente estendibile.

- **Model:** Il **Model** si occupa della gestione dei dati attraverso **DAO**, che interagiscono con il database per recuperare e memorizzare le informazioni sugli utenti, sugli eventi e sui biglietti.
- **View:** La **View** è gestita tramite **JSP**, che permette la creazione di interfacce utente dinamiche e responsive.
- **Controller:** Il **Controller** è composto da **Servlet** che gestiscono la logica di business e il flusso di dati tra il **Model** e la **View**, rispondendo alle richieste HTTP degli utenti.

Tecnologie Utilizzate:

- **Linguaggi di Programmazione:**

- **(Data) Database:** Il sistema utilizza un database relazionale (**MySQL**) per memorizzare le informazioni relative agli utenti, agli eventi e ai biglietti. L'interazione con il database avviene tramite **JDBC**
- **(Application) Backend:** applicazione **Java** basata su **Spring Boot**, responsabile della logica applicativa, della gestione delle API REST e della comunicazione con il database;

- **(Presentation) Frontend:** interfaccia web realizzata utilizzando **HTML**, **CSS** e **JavaScript**, con l’impiego di **AJAX** per garantire aggiornamenti dinamici e una navigazione fluida.

Il sistema verrà eseguito in ambiente locale mediante **Apache Tomcat** come server di applicazione, Lo sviluppo sarà effettuato in **IntelliJ IDEA** o **Eclipse**, con sistema di versionamento tramite GitHub e gestione delle dipendenze mediante Maven.

- **Sicurezza:**

- **Hashing delle password:** Le password degli utenti vengono protette tramite un algoritmo di hashing sicuro (ad esempio **SHA-256**) per garantire che le credenziali non vengano mai memorizzate in chiaro.
- **HTTPS:** L’intera piattaforma è protetta da **HTTPS** per garantire la cifratura delle comunicazioni tra il client e il server, proteggendo i dati sensibili durante il transito.
- **Prevenzione degli attacchi:** Il sistema è progettato per prevenire attacchi comuni come **SQL Injection**, **XSS** (Cross-Site Scripting) e **CSRF** (Cross-Site Request Forgery).

Funzionalità Principali:

1. Registrazione e Autenticazione Utenti:

- Gli utenti possono creare un account con informazioni personali e gestire le proprie credenziali in modo sicuro. Il sistema garantisce un’autenticazione sicura tramite hashing delle password e l’uso di sessioni per mantenere gli utenti loggati.

2. Ricerca e Navigazione degli Eventi:

- Gli utenti possono cercare eventi in base a vari filtri, come **data**, **località**, **tipo di evento** (concerti, sport, teatro, ecc.) e **categoria**. La piattaforma fornisce un’interfaccia facile da navigare per selezionare gli eventi di interesse.

3. Acquisto dei Biglietti:

- Gli utenti possono visualizzare i dettagli degli eventi selezionati, vedere la disponibilità dei biglietti e acquistare i biglietti tramite una procedura di check-out sicura. Durante il processo di acquisto, l’utente può scegliere il numero di biglietti e procedere al pagamento.

4. Gestione del Carrello:

- Gli utenti possono aggiungere eventi al proprio carrello e modificare il contenuto prima di procedere al pagamento. Il carrello fornisce una visione chiara delle voci selezionate, con il totale aggiornato in tempo reale.

5. Sistema di Pagamento Sicuro:

- Il sistema supporta il pagamento online tramite un’integrazione con un gateway di pagamento sicuro (ad esempio **PayPal**, **Stripe**, o simili). Le transazioni sono criptate e gestite in conformità con gli standard di sicurezza PCI-DSS.

6. Dashboard Amministratori:

- Gli amministratori hanno accesso a una **dashboard** che consente di gestire gli eventi, visualizzare e analizzare le vendite, e monitorare l'attività degli utenti. Possono aggiungere, modificare o rimuovere eventi, aggiornare informazioni sui biglietti e gestire i pagamenti ricevuti. Ogni amministratore avrà una dashboard relativa al suo ruolo e essendo in un account da amministratore, non sarà consentito loro effettuare acquisti tramite tale profilo.

Sicurezza e Privacy:

- **Crittografia delle comunicazioni:** Tutte le comunicazioni tra l'utente e il server sono cifrate tramite **HTTPS** per garantire la protezione dei dati sensibili.
- **Protezione dei Dati Utente:** Le informazioni personali e di pagamento degli utenti sono trattate secondo le normative sulla privacy (GDPR). Le password sono memorizzate in formato **hashato** (60 caratteri) per impedire accessi non autorizzati.
- **Prevenzione degli attacchi:** Il sistema implementa meccanismi di protezione contro **SQL injection**, **XSS**, **CSRF** e altri tipi di vulnerabilità comuni nelle applicazioni web.

Performance e Scalabilità:

- Il sistema è progettato per essere scalabile, in modo da supportare un numero crescente di utenti e richieste senza compromettere le performance.

1.3 Objectives and success criteria of the project

- **Obiettivo 1:** Consentire agli utenti di cercare, visualizzare e acquistare biglietti con facilità.
- **Obiettivo 2:** Ridurre il tempo medio di acquisto dei biglietti a meno di 2 minuti per utente.
- **Obiettivo 3:** Garantire la sicurezza dei dati sensibili degli utenti e delle transazioni, prevenendo accessi non autorizzati.
- **Obiettivo 4:** Fornire una dashboard completa per i vari amministratori e organizzatori per la gestione efficiente degli eventi.
- **Criteri di successo:**
 - Riduzione del numero di errori o abbandoni durante il processo di acquisto.
 - Disponibilità del sistema $\geq 99\%$.
 - Interfaccia utente valutata almeno 4/5 in test di usabilità.
 - Completamento di tutte le funzionalità richieste senza bug critici.

1.4 Definitions, acronyms, and abbreviations

1. **SC** = Scenarios;
2. **UC** = Use Case;
3. **CRUD** = Create Read Update Delete;
4. **DBA** = DataBase Administrator;
5. **MVC** = Model View Controller;
6. **JSP** = JavaServer Pages;
7. **DAO** = Data Access Object;
8. **JDBC** = Java DataBase Connectivity;
9. **XSS** = Cross-Site Scripting;
10. **CSRF** = Cross-Site request forgery;
11. **HTTP** = HyperText Transfer Protocol;
12. **HTTPS** = HyperText Transfer Protocol Secure;

1.5 References

- "BEAT Booking Events And Tickets - Problem Statement", Versione 1.1.
- B. Bruegge , A. H. Dutoit , "Object-Oriented Software Engineering: Using UML, Patterns, and Java", Third Edition.

1.6 Overview

Il documento descrive i requisiti e i modelli di sistema per la piattaforma BEAT.

- **La Sezione 2 (Current System)** analizza il contesto e le limitazioni dei sistemi esistenti.
- **La Sezione 3 (Proposed System)** descrive la soluzione proposta, suddivisa in:
 - **Requisiti Funzionali (3.2)**: Specifica le funzionalità per organizzatori, clienti e amministratori.
 - **Requisiti Non Funzionali (3.3)**: Definisce i vincoli di qualità del sistema (es. performance, implementazione, sicurezza).
 - **Modelli di Sistema (3.4)**: Include Scenari, Modelli di Casi d'Uso e Mock-up.

2 Current system

Attualmente, l'acquisto dei biglietti per eventi avviene principalmente tramite siti web generici di biglietteria, direttamente presso le casse fisiche degli eventi oppure i tabacchi. Questi sistemi permettono di selezionare l'evento, il tipo di biglietto e il metodo di pagamento, ma presentano alcune limitazioni significative: l'interfaccia spesso è poco intuitiva, i passaggi per completare l'acquisto sono numerosi e dispersivi, e la visualizzazione dei posti disponibili può risultare poco chiara. Inoltre, alcuni sistemi non offrono informazioni immediate su disponibilità, prezzi aggiornati o eventuali promozioni.

Queste problematiche possono generare incertezza e frustrazione nell'utente, aumentando il tempo necessario per completare l'acquisto e creare sbavature negative nell'esperienza complessiva. Per questo motivo, il sistema proposto, mira a offrire un'interfaccia più semplice, intuitiva e visivamente accattivante, riducendo il numero di passaggi e migliorando l'usabilità complessiva.

3 Proposed system

3.1 Overview

Questa sezione delineandone le funzionalità, i vincoli di qualità e i modelli di progettazione.

- **La Sottosezione 3.2 (Functional Requirements)** elenca le funzionalità specifiche che il sistema deve eseguire , suddivise per i ruoli di Organizzatore, Cliente e Amministratore.
- **La Sottosezione 3.3 (Nonfunctional Requirements)** definisce i vincoli di qualità del sistema, coprendo aspetti di Usabilità, Performance , Implementazione (architettura three-tier) e Sicurezza .
- **La Sottosezione 3.4 (System Models)** illustra il comportamento e la struttura del sistema attraverso Scenari testuali e il Modello dei Casi d'Uso (Use Case).

3.2 Functional requirements

Il sistema deve permettere:

1. Un utente non registrato (guest) deve poter consultare l'elenco e i dettagli di eventi disponibili;
2. Gestione eventi (CRUD) da parte degli organizzatori;
3. Visualizzazione pubblica degli eventi disponibili, filtrabili per categoria, data o luogo;
4. Registrazione e login per gli utenti finali;
5. Prenotazione e cancellazione dei biglietti;
6. Riepilogo prenotazioni personale per ogni utente;

7. Gestione del DataBase da parte di un amministratore di catalogo eventi;
8. Gestione delle prenotazioni effettuate da parte di un amministratore degli ordini;
9. Gestione dei ruoli di tutte le figure presenti a cura di un amministratore dei ruoli;

3.3 Nonfunctional requirements

3.3.1 Usability

Applicazione web **responsiva**, accessibile sia da dispositivi **desktop** che **mobile**, con un layout adattivo e coerente in ogni contesto di utilizzo.

3.3.2 Reliability

Disponibilità minima del sistema pari al **99%** in ambiente di test locale, garantendo continuità di servizio e corretta esecuzione delle principali funzionalità anche in presenza di errori limitati.

3.3.3 Performance

Buona **scalabilità** del sistema, con capacità di gestire contemporaneamente molti utenti senza perdite significative di prestazioni. Il tempo medio di risposta deve rimanere **inferiore a 2 secondi** per operazioni standard (caricamento di pagine, login, ricerca e prenotazioni).

3.3.4 Supportability

Il sistema dovrà essere facilmente **manutenibile**, **configurabile** e documentato, con codice modulare e separazione chiara tra componenti per facilitare aggiornamenti futuri.

3.3.5 Implementation

Architettura **Three-tier**, suddivisa in **Database**, **Backend** e **Frontend**, al fine di garantire indipendenza logica tra i livelli e semplificare la manutenzione.

3.3.6 Interface

Interfaccia utente **intuitiva e coerente**, con aggiornamenti dinamici e reattivi ottenuti tramite tecniche **AJAX** per migliorare l'esperienza d'uso.

3.3.7 Security

Le credenziali sono **cifrate** e il sistema prevede un rigoroso **controllo degli accessi**, con ruoli distinti per **utente**, **organizzatore** e **amministratore** di ogni tipologia.

3.3.8 Operation

Il sistema viene gestito da diverse figure, ognuna delle quali ha un ruolo:

1. **Web master:** per la gestione generale e operativa;
2. **Web developper:** per gli aspetti tecnici e di codice;
3. **Web designer:** per gli aspetti di progettazione e interfaccia;
4. **Esperti di sicurezza:** per la sicurezza e la prevenzione di attacchi;
5. **DBA:** per la gestione e manutenzione del database sottostante.

3.3.9 Packaging

1. **Chi installa il sistema?** L'installazione dell'applicazione web sarà effettuata da un team IT. Gli utenti finali non sono coinvolti nell'installazione del sistema.
2. **Qual'è il numero di installazioni previste?** L'applicazione è progettata per supportare una distribuzione scalabile, ospitata su un server. Al momento è previsto il deployment di due ambienti (produzione e test), con possibilità di scalare in base alle necessità. Non vi sono limiti sul numero di utenti finali che possono accedere all'applicazione, in quanto la piattaforma è progettata per essere altamente scalabile.
3. **Vi sono vincoli temporali sull'installazione?** L'installazione del sistema avverrà in circa un'ora per l'ambiente di produzione. Ogni aggiornamento dell'applicazione sarà effettuato durante finestre di manutenzione precedentemente accordate con il cliente. Ogni aggiornamento sarà implementato senza tempi di fermo, utilizzando tecniche di blue-green deployment, tecnica che minimizza il downtime ed i rischi eseguendo due ambienti di produzione identici blue(versione attuale), green(nuova versione). Una volta che l'ambiente green sarà ultimato il traffico verrà reindirizzato dall'ambiente blue all'ambiente green utilizzando un router o un bilanciatore di carico. Se dovessero sorgere problemi con la nuova versione, il traffico può essere immediatamente riportato all'ambiente blue originale per un rapido rollback.

3.3.10 Legal

Il sistema deve aderire alle seguenti normative:

- **RNF 10.1 (GDPR):** Conformità al Regolamento UE 2016/679 per il trattamento dei dati personali, includendo informative sulla privacy e gestione del consenso.
- **RNF 10.2 (Cookie Policy):** Implementazione di un banner per il consenso all'uso dei cookie.
- **RNF 10.3 (Sicurezza Pagamenti):** Adozione degli standard PCI-DSS per la gestione sicura delle transazioni.
- **RNF 10.4 (Termini e Condizioni):** Disponibilità di Termini di Servizio chiari per definire l'uso della piattaforma, le responsabilità e le politiche di annullamento

3.4 System models

3.4.1 Scenarios

SC_1 Un organizzatore vuole aggiungere un evento: Vincenzo vuole aggiungere all'elenco degli eventi prenotabili una data di un evento che si terrà prossimamente. Vincenzo tramite un apposito pulsante "login" verrà reindirizzato alla pagina di login, qui selezionerà l'opzione di login per gli organizzatori, inserirà le sue credenziali, email (es. vincenzo123@gmail.com) e password (es. Password@123), una volta inserite potrà cliccare il bottone "login" per accedere al proprio account ed essere reindirizzato sulla pagina della dashboard degli organizzatori. Fra le varie opzioni a sua disposizione (crea evento, modifica evento, elimina evento) Vincenzo selezionerà "crea evento" e gli verrà mostrata a schermo la pagina in cui inserire i dati relativi all'evento. I campi che vanno compilati per l'aggiunta dell'evento sono "titolo dell'evento", "data" (formato gg-mm-aaaa), "orario di inizio" (formato [hh]:[mm]), "protagonista" (chi svolge l'evento, un cantante se si tratta di un concerto, un professore se è un seminario, o chiunque sia). Questo campo può essere lasciato vuoto solo se si tratta di una fiera con tanti ospiti e nessun artista "protagonista", ma in questo caso è presente una casella da flaggare in cui indicare "si tratta di una fiera". Va compilato il campo relativo ai "posti disponibili" in cui andrà inserito il limite massimo di prenotazioni accettabili, le "info sull'evento" in cui potrà essere inserita una breve descrizione dell'evento che si terrà. È previsto un campo in cui sarà possibile inserire un'immagine (ad esempio la locandina dell'evento, se presente). Si inserisce poi il "prezzo" che le persone dovranno pagare per avere la loro prenotazione. È prevista una casella da flaggare che permette di impostare l'evento come gratuito: selezionando "ingresso gratuito" il campo prezzo verrà bloccato e l'evento risulterà "gratis". Una volta inserite tutte le informazioni necessarie, Vincenzo potrà cliccare sul pulsante posto in basso nella pagina "crea evento" per ufficializzare la creazione dell'evento. Prima di completare l'operazione, gli verrà mostrato un riepilogo delle informazioni inserite per controllare eventuali errori. Se tutto è corretto, potrà cliccare "conferma creazione". L'evento viene aggiunto all'elenco di quelli disponibili e l'organizzatore viene reindirizzato alla sua "dashboard eventi", dove potrà vedere nella sezione "ancora da svolgere" l'evento appena aggiunto con tutte le informazioni relative. A questo punto gli utenti potranno prenotare l'evento o acquistare i biglietti.

SC_2 Utente registrato vuole acquistare un biglietto: Mario vuole acquistare un biglietto per il concerto dei JazzBrothers. Mario tramite il pulsante di login viene reindirizzato alla schermata apposita in cui può inserire le proprie credenziali, email (es. mario1234@gmail.com) e password (es. Pssword@456), clicca sul pulsante di login ed accede al proprio account. A questo punto Mario cerca all'interno dell'elenco degli eventi oppure tramite l'apposita barra di ricerca il concerto che gli interessa, procede a selezionare l'evento, vengono quindi mostrate a schermo tutte le informazioni riguardo il concerto dei JazzBrothers (orario di inizio hh:mm, data gg-mm-aaaa, luogo di svolgimento, prezzo del biglietto, posti disponibili). Mario seleziona la data di suo interesse e viene reindirizzato alla schermata di selezione del numero di biglietti o posti da prenotare (massimo 4 per evitare bagarinoaggio e consentire un accesso più equo ai biglietti, specialmente per eventi ad alta richiesta), andando avanti verrà reindirizzato alla pagina relativa al pagamento dove potrà inserire i dati relativi al metodo pagamento (paypal o carta di credito) e

le informazioni necessarie ad effettuarlo (numero di carta, scadenza, cvv oppure email per paypal), successivamente conferma il pagamento e viene reindirizzato alla schermata relativa ai suoi ordini, nella quale, fra gli eventi che devono ancora svolgersi, vedrà il biglietto appena acquistato, che potrà essere consultato sia tramite interfaccia web (anche su dispositivi mobile), sia tramite un pdf che potrà scaricare grazie ad un apposito pulsante posto nella parte inferiore del riquadro riguardante il biglietto.

SC_3 Un organizzatore modifica o cancella un evento: Vincenzo ha bisogno di aggiornare i dettagli di un evento già creato o, in un caso separato, di cancellarlo a causa di un imprevisto. Vincenzo accede al sistema tramite l'apposito pulsante "login", seleziona l'opzione per organizzatori e inserisce le sue credenziali, venendo reindirizzato sulla pagina della "dashboard degli organizzatori".

Caso A: Modifica dell'evento

Dalla sua "dashboard eventi", Vincenzo visualizza l'elenco degli eventi inseriti, in particolare quelli nella sezione "ancora da svolgere". Si accorge di aver inserito un prezzo errato per un seminario. Individua l'evento nell'elenco e seleziona l'opzione "modifica evento". Viene reindirizzato alla pagina di gestione dell'evento, che si presenta pre-compilata con tutte le informazioni inserite durante la creazione (titolo, data, posti, ecc.). Vincenzo individua il campo "prezzo", deseleziona la casella "ingresso gratuito" e inserisce il nuovo importo. Clicca quindi sul pulsante "Salva modifiche". Come per la creazione, gli viene mostrato un riepilogo delle informazioni modificate e clicca su "conferma modifiche". L'evento è immediatamente aggiornato e gli utenti che lo consulteranno vedranno il nuovo prezzo.

Caso B: Eliminazione dell'evento

Per un altro evento (ad esempio un concerto) che è stato annullato, Vincenzo torna alla sua "dashboard eventi". Trova l'evento nell'elenco "ancora da svolgere" e questa volta seleziona l'opzione "elimina evento". Il sistema gli chiederà un'ulteriore conferma per evitare cancellazioni accidentali (es. "Sei sicuro di voler eliminare questo evento?"). Vincenzo clicca su "Conferma eliminazione". L'evento viene rimosso dall'elenco e non sarà più visibile né prenotabile dagli utenti.

SC_4 Visualizzazione e download del biglietto digitale: Mario, che ha già acquistato il suo biglietto per il concerto dei JazzBrothers, vuole ora consultare i dettagli in vista dell'evento. Mario accede alla piattaforma BEAT e utilizza il pulsante di login per inserire le sue credenziali. Una volta autenticato, naviga verso la sua area personale, accedendo alla schermata relativa ai suoi "ordini" o al "Riepilogo prenotazioni personale". In questa sezione, Mario visualizza l'elenco degli eventi per cui ha una prenotazione. Individua il riquadro relativo al concerto dei JazzBrothers, che si trova tra gli eventi "ancora da svolgersi". Clicca sull'evento per aprirne i dettagli. Il sistema gli mostra a schermo il biglietto digitale tramite l'interfaccia web. La schermata mostra chiaramente tutte le informazioni essenziali per l'accesso:

- Il titolo dell'evento (Concerto dei JazzBrothers).
- Il nome del "protagonista" (JazzBrothers).
- La data (formato gg-mm-aaaa) e l'orario di inizio (formato hh:mm).
- Le informazioni sul luogo di svolgimento.
- Il numero di biglietti acquistati da Mario e i posti eventualmente assegnati.

- Un codice univoco di prenotazione (ad esempio un QR code o un codice alfanumerico) da presentare all'ingresso per la validazione.

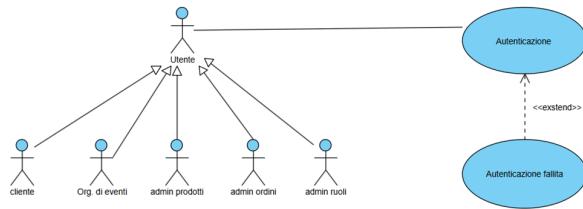
Mario può quindi utilizzare direttamente il suo smartphone all'ingresso dell'evento, mostrando questa schermata web per la scansione e l'ammissione.

SC_5 Un utente registrato vuole annullare una prenotazione: Mario ha acquistato un biglietto per un seminario, ma un impegno improvviso gli impedisce di partecipare. Decide quindi di annullare la sua prenotazione. Mario accede alla piattaforma BEAT e, tramite il pulsante di login , inserisce le sue credenziali (es. mario123@gmail.com e Pssword@456) per accedere al proprio account. Una volta autenticato, naviga verso la sua area personale, accedendo alla schermata "Riepilogo prenotazioni personale" (o la sezione relativa ai suoi ordini). In questa schermata, Mario visualizza l'elenco dei suoi biglietti. Individua il biglietto per il seminario che desidera annullare, che si trova nella sezione "ancora da svolgersi". In corrispondenza di tale prenotazione, individua e clicca sull'apposito pulsante "Annulla prenotazione". Il sistema, per evitare un'azione accidentale, gli mostra un messaggio di conferma (es. "Sei sicuro di voler annullare questa prenotazione?"). Mario clicca sul pulsante "Conferma annullamento". La prenotazione viene così cancellata. *Il biglietto scompare immediatamente dall'elenco "ancora da svolgersi" nel suo riepilogo personale e il posto che occupava viene reso nuovamente disponibile nel conteggio dei "posti disponibili" dell'evento, permettendo ad altri utenti di acquistarla.*

SC_6 Un amministratore dei ruoli gestisce le utenze: Antonio è l'"Amministratore dei Ruoli". Deve promuovere un utente standard, "Vincenzo" (vincenzo123@gmail.com), al ruolo di "Organizzatore", per consentirgli di inserire e gestire i propri eventi. Antonio accede alla piattaforma BEAT tramite la pagina di login, inserendo le sue credenziali (es. admin.ruoli@beat.com e PAdmin!). Essendo un amministratore di questa tipologia , viene reindirizzato a un pannello di amministrazione dedicato alla "Gestione dei ruoli". In questa schermata, Antonio visualizza un elenco di tutti gli utenti registrati nel sistema. Utilizza la funzione di ricerca per trovare l'account "vincenzo123@gmail.com". Una volta trovato, vede che il ruolo attualmente associato a Vincenzo è "Utente Registrato(cliente)". Antonio clicca sul pulsante "Modifica Ruolo" relativo a quell'utente. Gli viene presentata una selezione dei ruoli disponibili che può assegnare, tra cui "Utente Registrato", "Organizzatore", "Amministratore Catalogo Eventi" e "Amministratore Ordini". Antonio seleziona "Organizzatore" e clicca su "Salva Modifiche". *Il sistema aggiorna immediatamente i permessi dell'utente. Al suo prossimo accesso, Vincenzo, dopo aver inserito le sue credenziali , verrà reindirizzato "sulla pagina della schermata degli organizzatori" e avrà accesso alle funzionalità di creazione, modifica ed eliminazione degli eventi.*

3.4.2 Use case model

UC_1 AUTENTICAZIONE



Attore: Utente

Entry Condition: L'utente si trova nella schermata di autentificazione di BEAT

Flusso di eventi:

1. L'utente inserisce username e password
2. L'utente invia i dati al sistema
3. Il sistema controlla le credenziali e il controllo ha esito positivo
4. Il sistema reindirizza l'utente alla sua home page

Exit Condition: L'utente è autenticato e si trova sulla sua home page

Flussi alternativi / Eccezioni:

Se al punto 3 il sistema rileva credenziali non corrette, il sistema mostrerà il messaggio di errore “errore di login riprovare” e ripresenterà la schermata di autenticazione (UC 1.1 Autenticazione Fallita)

Alternativa

Se al punto 3 il sistema rileva credenziali non corrette, verrà eseguito il caso d'uso UC 1.1 Autenticazione Fallita

UC 1.1: Autenticazione fallita

Attore: Utente

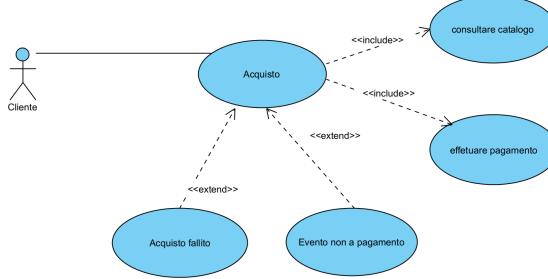
Entry Condition: L'utente ha provato ad autenticarsi e le credenziali non erano corrette

Flusso di eventi:

1. il sistema mostra il messaggio di errore “errore di login riprovare”
2. il sistema ripresenta all'utente la schermata di autenticazione

Exit Condition: L'utente si trova nella schermata di autentificazione di BEAT

UC_2 ACQUISTO



Attore: Utente(cliente)

Entry Condition: L'utente registrato ha effettuato l'accesso al suo account e vuole acquistare un biglietto

Flusso di eventi:

1. L'utente si dirige sul catalogo degli eventi disponibili
2. L'utente seleziona l'evento di suo interesse
3. L'utente procede con la selezione dell'opzione "prenota evento"
4. Il sistema reindirizza l'utente alla schermata di pagamento
5. L'utente invia al sistema i dati per il pagamento
6. Il sistema verifica che il pagamento sia andato a buon fine
7. Il sistema reindirizza l'utente alla schermata "Le mie prenotazioni"

Exit Condition: L'utente ha prenotato il suo biglietto per l'evento

Flussi alternativi / Eccezioni:

Se al punto 3 il sistema rileva che l'evento è gratuito il sistema mostrerà il messaggio "non è necessario inserire metodi di pagamento per gli eventi gratuiti" e presenterà la schermata di prenotazione in cui l'utente dovrà inserire solo le proprie generalità

Alternativa

Se al punto 3 il sistema rileva un evento gratuito, verrà eseguito il caso d'uso UC 2.1 Evento non a pagamento

UC 2.1: Evento non a pagamento

Attore: Utente

Entry Condition: L'utente vuole acquistare un biglietto per un evento ma questo è gratuito

Flusso di eventi:

1. Il sistema mostra il messaggio "non è necessario inserire metodi di pagamento per gli eventi gratuiti"
2. Il sistema presenta all'utente una schermata in cui inserire unicamente le proprie generalità per procedere alla prenotazione, senza dover inserire alcun metodo di pagamento
3. Il sistema reindirizzerà l'utente sulla schermata "Le mie prenotazioni"

3.4.3 *Object model*

3.4.4 *Dynamic model*

3.4.5 User interface-navigational paths and screen mock-ups

- User(cliente)

