|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  | | --- | | ENTE/I: INGSW – GRUPPO 32  DATA EMISSIONE: A.A. 2017/18 | | |  | | --- | |  | | **DOCUMENTO DI DESIGN** | |

|  |  |
| --- | --- |
| |  | | --- | | **OGGETTO:**  **Design del Sistema Informativo “*Event Manager ‘17*”** | |

|  |  |
| --- | --- |
| |  | | --- | | **SINTESI DEI CONTENUTI:**  Il documento contiene le seguenti informazioni riguardanti la struttura del software:  *1. Modello Funzionale.*   * *Modellazione dei casi d’uso.* * *Tabelle di Cockburn per ogni caso d’uso.* * *Mock-up interfaccia utente.* * *Una proposta di impegno risorse e di pianificazione dettagliata dell’attività, con diagrammi di Gantt.* * *Glossario.*   *2. Modelli di Dominio.*   * *f. Classi, oggetti e relazioni di analisi.* * *g. Diagrammi di sequenza di analisi.* * *h. Diagrammi di stato di analisi* * *Diagrammi di attività.* | |

|  |  |
| --- | --- |
| **EMITTENTE** | **DESTINATARIO** |
| Gruppo 32:  *Alessandro Pianese*  Roberto Trinchese | *Sergio Di Martino* |

Sommario

[I. Indice delle figure 2](#_Toc519779111)

[II. Revisioni 3](#_Toc519779112)

[III. Obiettivi 3](#_Toc519779113)

[IV. Modello funzionale 4](#_Toc519779114)

[A. Modellazione dei casi d’uso 4](#_Toc519779115)

[B. Tabelle di Cockburn 5](#_Toc519779116)

[C. Rappresentazione delle schermate tramite Mock-up 11](#_Toc519779117)

[1. Applicazione desktop 11](#_Toc519779118)

[2. Applicazione Android 15](#_Toc519779119)

[V. Modelli di dominio 17](#_Toc519779120)

[A. Classi, oggetti e relazioni di analisi 17](#_Toc519779121)

[B. Diagrammi di sequenza di analisi 22](#_Toc519779122)

# Indice delle figure

[Figura 1: Use Case Diagram sistema complete 5](#_Toc519779052)

[Figura 2: Login da applicazione desktop 12](#_Toc519779053)

[Figura 3: Gestione eventi 13](#_Toc519779054)

[Figura 4: Aggiungi evento 13](#_Toc519779055)

[Figura 5: Aggiungi evento (campi non compilati) 14](#_Toc519779056)

[Figura 6: Modifica evento 14](#_Toc519779057)

[Figura 7: Rimuovi Evento 15](#_Toc519779058)

[Figura 8: Login da Android 16](#_Toc519779059)

[Figura 9: Accettazione evento 16](#_Toc519779060)

[Figura 10: Scansione codice QR 17](#_Toc519779061)

[Figura 11 e 12: QR code valido/non valido 17](#_Toc519779062)

[Figura 12: Class Diagram "Aggiungi Evento" 18](#_Toc519779063)

[Figura 13: Class Diagram "Modifica Evento" 19](#_Toc519779064)

[Figura 14: Classi Diagram "Rimuovi Evento" 20](#_Toc519779065)

[Figura 15: Class Diagram "Visualizza Statistiche" 21](#_Toc519779066)

[Figura 16: Class Diagram "Scansiona QR Code" 22](#_Toc519779067)

[Figura 17: Sequence Diagram "Aggiungi Evento" 23](#_Toc519779068)

[Figura 18: Sequence Diagram "Modifica Evento" 23](#_Toc519779069)

[Figura 19: Sequence Diagram "Rimuovi Evento" 24](#_Toc519779070)

[Figura 20: Sequence Diagram "Visualizza Statistiche" 24](#_Toc519779071)

[Figura 21: Sequence Diagram "Scansiona QR Code" 25](#_Toc519779072)

# Revisioni

|  |  |  |  |
| --- | --- | --- | --- |
| Data | Versione | Autore | Descrizione |
| 18/07/2018 | 0.1 | Roberto Trinchese | Iniziato e Concluso il document di analisi |

# Obiettivi

Questo documento ha lo scopo di analizzare e formalizzare, tramite opportune notazioni, i requisiti del sistema software da sviluppare, relativamente alle funzionalità richieste.

Queste ultime sono state concordate con il committente e riguardano:

* Codice d’accesso da android: il software deve permettere di accedere ad un determinate evento tramite applicazione android.

Nei capitoli successivi sarà descritto in maniera esaustiva tutto ciò che il sistema deve essere in grado di fare. Per fare ciò si useranno delle notazioni formali, come il diagramma dei casi d’uso, corredato da mock-up del sistema e da una descrizione testuale per ogni funzionalità, ed altri diagrammi classici di UML.

# Modello funzionale

## Modellazione dei casi d’uso

Il diagramma dei casi d’uso serve a descrivere le funzionalità di un sistema dal punto di vista degli utenti. Di seguito sono mostrati gli use case diagram per il SuD (System under Developement).

Il seguente Use Case Diagram mostra le funzionalità generali dell’intero sistema:

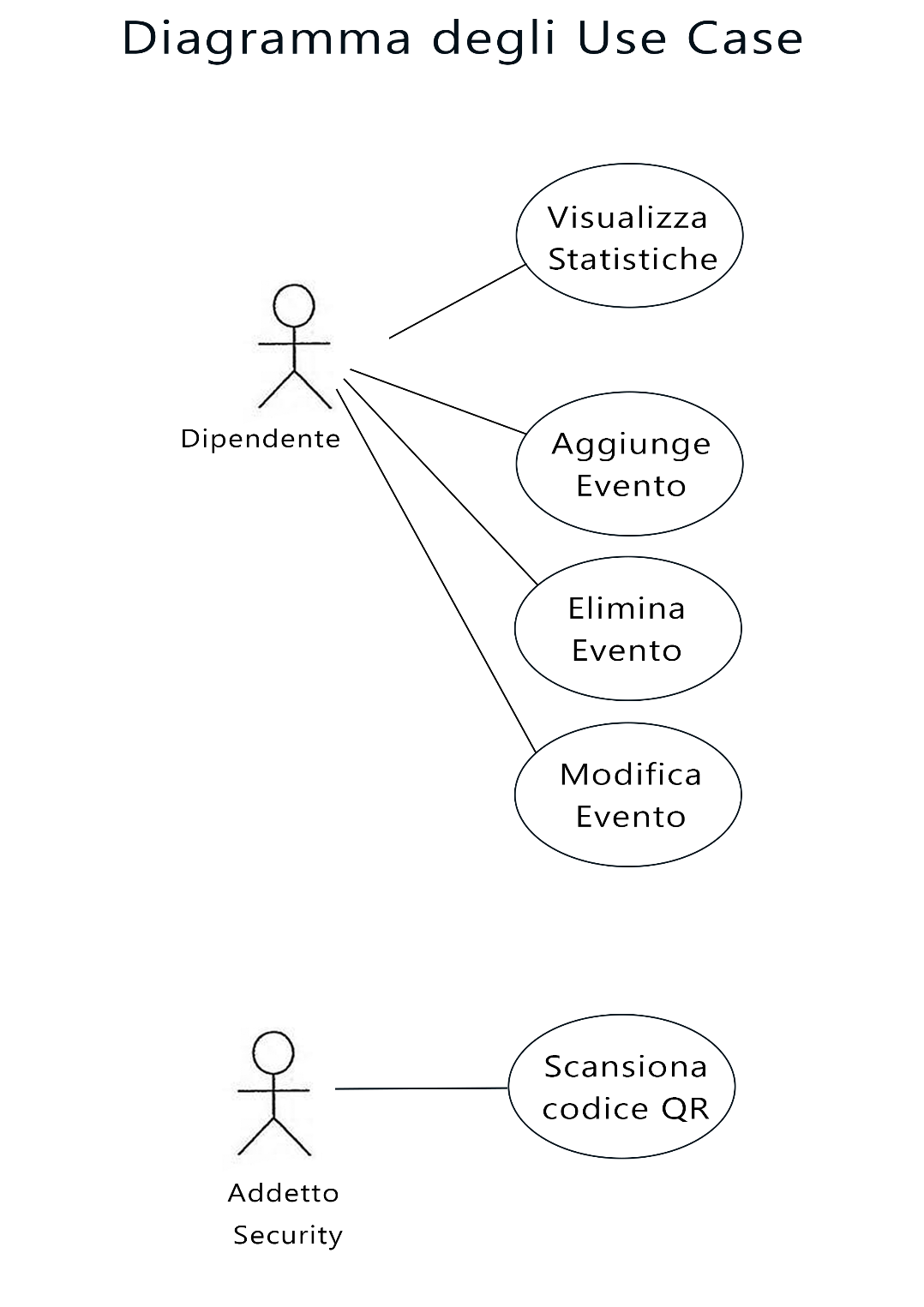


Figura 1: Use Case Diagram sistema complete

## Tabelle di Cockburn

Il modello di Cockburn spiega il funzionamento principale del software (quello relativo ai casi d’uso) tramite delle tabelle che specificano le interazioni tra l’utente e il sistema e i collegamenti tra le varie interfacce utente, chiarendo anche il funzionamento del programma in caso di errori da parte dell’utente.

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE *#1* | *Visualizza Statistiche* | | |
| Goal in Context | Permettere all’utente di visionare le statistiche relative ad un evento | | |
| Scope & Level | EventManager  Primary Task | | |
| Preconditions | Il dipendente deve avere effettuato l’accesso al sistema (tramite la schermata di mockup denominata [*Login*](https://github.com/Wertiz/EventManager/blob/master/Requirements%20Analysis%20Document/Mockups/Login.png)) | | |
| Success End Condition | Il sistema mostra le statistiche relative all’evento scelto | | |
| Failed End Condition | / | | |
| Primary Actor | Dipendente | | |
| Trigger | Il Dipendente clicca sul pulsante “Visualizza Statistiche” raffigurato da un grafico a torta vicino l’evento come mostrato nel mockup [Gestione Evento](https://github.com/Wertiz/EventManager/blob/master/Requirements%20Analysis%20Document/Mockups/Gestione%20Eventi.png) | | |
| DESCRIPTION | **Step n°** | **Dipendente** | **Sistema** |
| *1* | Il dipendente clicca sul pulsante “Visualizza Statistiche” |  |
| *2* |  | Il Sistema mostra la schermata contenente le statistiche dell’evento (Mockup [Visualizza Statistiche](https://github.com/Wertiz/EventManager/blob/master/Requirements%20Analysis%20Document/Mockups/Visualizza%20Statistiche.png)) |

Seguono quindi le tabelle relative al caso d’uso mostrato in precedenza:

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE *#2* | *Aggiungi Evento* | | |
| Goal in Context | Permettere all’utente di aggiungere un evento al sistema | | |
| Scope & Level | EventManager  PrimaryTask | | |
| Preconditions | Il Dipendente deve avere effettuato l’accesso | | |
| Success End Condition | Il Dipendente aggiunge un nuovo evento al sistema | | |
| Failed End Condition | / | | |
| Primary Actor | Dipendente | | |
| Trigger | Il Dipendente clicca sul pulsante di aggiunta di un evento | | |
| DESCRIPTION | **Step n°** | **Dipendente** | **Sistema** |
| *1* | Il Dipendente clicca sul pulsante raffigurante il simbolo ‘+’ posizionato in basso a destra della schermata |  |
| *2* |  | Il Sistema mostra la schermata di [Aggiunta di un Evento](https://github.com/Wertiz/EventManager/blob/master/Requirements%20Analysis%20Document/Mockups/Aggiungi%20Evento.png) |
| *3* | Il Dipendente riempie tutti i campi che la schermata comporta |  |
| *4* | Il Dipendente clicca sul pulsante *“Inserisci Evento”* |  |
| *5* |  | Il Sistema mostra la schermata di [Gestione Eventi](https://github.com/Wertiz/EventManager/blob/master/Requirements%20Analysis%20Document/Mockups/Gestione%20Eventi.png) con l’aggiunta dell’evento precedentemente creato |
| EXTENSION | *Step* | **Dipendente** | **Sistema** |
|  | *3* | Il Dipendente non riempie tutti i campi |  |
|  | *4* | Il Dipendente clicca sul pulsante *“Inserisci Evento”* |  |
|  | *5* |  | Il Sistema mostra un [*alert*](https://github.com/Wertiz/EventManager/blob/master/Requirements%20Analysis%20Document/Mockups/Aggiungi%20Evento%20(Non%20tutti%20i%20campi%20inseriti).png) che invita l’utente a compilare tutti i campi |
|  | *6* | Il Dipendente clicca su *“Ok”* |  |
|  | *7* |  | Si ritorna allo step n°2 del *main flow* |
| SUBVARIATION | *Step* | **Dipendente** | **Sistema** |
|  | *3a* | Il Dipendente clicca sul tasto in alto a desta contenente la parentesi angolare “<” per tornare indietro |  |
|  | *4a* |  | Il Sistema torna indietro mostrando la schermata di [Gestione Eventi](https://github.com/Wertiz/EventManager/blob/master/Requirements%20Analysis%20Document/Mockups/Gestione%20Eventi.png) |

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE *#4* | *Rimuovi Evento* | | |
| Goal in Context | Permettere all’utente di eliminare un evento presente nel sistema | | |
| Scope & Level | EventManager  Primary Task | | |
| Preconditions | Il dipendente deve avere effettuato l’accesso al sistema (tramite la schermata di mockup denominata [*Login*](https://github.com/Wertiz/EventManager/blob/master/Requirements%20Analysis%20Document/Mockups/Login.png)) | | |
| Success End Condition | L’evento viene eliminato dal sistema | | |
| Failed End Condition | / | | |
| Primary Actor | Dipendente | | |
| Trigger | Il Dipendente clicca sul pulsante “Cancella Evento” raffigurato da una “x” vicino l’evento come mostrato nel mockup [Gestione Evento](https://github.com/Wertiz/EventManager/blob/master/Requirements%20Analysis%20Document/Mockups/Gestione%20Eventi.png) | | |
| DESCRIPTION | **Step n°** | **Dipendente** | **Sistema** |
| *1* | Clicca sull’icona a forma di “x” |  |
| *2* |  | Avvisa il Dipendente di confermare l’eliminazione dell’evento come mostrato nel mockup [Elimina Evento](https://github.com/KillTheProcess23/EventManager/blob/master/Requirements%20Analysis%20Document/Mockups/Rimuovi%20Eventi.png) |
|  | *3* | Clicca su “Ok” |  |
|  | *4* |  | Elimina l’evento e mostra [Gestione Evento](https://github.com/Wertiz/EventManager/blob/master/Requirements%20Analysis%20Document/Mockups/Gestione%20Eventi.png) |
| SUBVARIATION | *Step* | **Dipendente** | **Sistema** |
|  | *3a* | Il Dipendente clicca sul tasto annulla |  |
|  | *4°* |  | Il Sistema torna indietro mostrando la schermata di [Gestione Evento](https://github.com/Wertiz/EventManager/blob/master/Requirements%20Analysis%20Document/Mockups/Gestione%20Eventi.png) |

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE *#3* | *Modifica Evento* | | |
| Goal in Context | Permettere all’utente di modificare un evento presente nel sistema | | |
| Scope & Level | EventManager  PrimaryTask | | |
| Preconditions | Il Dipendente deve avere effettuato l’accesso | | |
| Success End Condition | Il Dipendente modifica un evento presente nel sistema | | |
| Failed End Condition | / | | |
| Primary Actor | Dipendente | | |
| Trigger | Il Dipendente preme il pulsante “Modifica Evento” raffigurato dal simbolo di una matita come mostrato nel Mockup [Gestione Evento](https://github.com/Wertiz/EventManager/blob/master/Requirements%20Analysis%20Document/Mockups/Gestione%20Eventi.png) | | |
| DESCRIPTION | **Step n°** | **Dipendente** | **Sistema** |
| *1* | Clicca sul pulsante raffigurante il simbolo della matita posizionato vicino ad un evento |  |
| *2* |  | Mostra la schermata di Modifica Evento |
| *3* | Modifica tutti i campi che desidera modificare |  |
| *4* | Clicca sul pulsante *“Modifica evento”* |  |
| *5* |  | Il Sistema avvisa il Dipendente dell’avvenuta modifica e mostra la schermata [Gestione Eventi](https://github.com/Wertiz/EventManager/blob/master/Requirements%20Analysis%20Document/Mockups/Gestione%20Eventi.png) |

|  |  |  |  |
| --- | --- | --- | --- |
| EXTENSION | *Step* | **Dipendente** | **Sistema** |
|  | *3* | Non riempie tutti i campi |  |
|  | *4* | Clicca sul pulsante *“Modifica evento”* |  |
|  | *5* |  | Mostra un [*alert*](https://github.com/Wertiz/EventManager/blob/master/Requirements%20Analysis%20Document/Mockups/Aggiungi%20Evento%20(Non%20tutti%20i%20campi%20inseriti).png) che invita l’utente a compilare tutti i campi |
|  | *6* | Clicca su *“Ok”* |  |
|  | *7* |  | ritorna allo step n°2 del *main flow* |

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE *#5* | *Scansione Codice QR* | | |
| Goal in Context | Permettere all’addetto alla security di validare i biglietti degli eventi | | |
| Scope & Level | EventManager  PrimaryTask | | |
| Preconditions | L’addetto alla security deve aver inserito il codice dell’evento | | |
| Success End Condition | La scansione accetta il QR Code | | |
| Failed End Condition | / | | |
| Primary Actor | Addetto alla security | | |
| Trigger | L’addetto alla security preme il pulsante “Scansiona Evento” come mostrato nel mockup [Accettazione Evento](https://github.com/KillTheProcess23/EventManager/blob/master/Requirements%20Analysis%20Document/Mockups/Android%20Accettazione%20Evento.png) | | |
| DESCRIPTION | **Step n°** | **Addetto alla security** | **Sistema** |
| *1* | Preme sul pulsante “Scansiona Evento” |  |
| *2* |  | Mostra la schermata con la fotocamera come mostrato nel mockup [Scansione codice QR](https://github.com/KillTheProcess23/EventManager/blob/master/Requirements%20Analysis%20Document/Mockups/Scansione%20codice%20QR.png) |
| *3* | Inquadra il QR Code del cliente |  |
| *5* |  | Mostra la [schermata di successo](https://github.com/KillTheProcess23/EventManager/blob/master/Requirements%20Analysis%20Document/Mockups/Android%20QR%20Scansionato.png) |

|  |  |  |  |
| --- | --- | --- | --- |
| EXTENSION | *Step* | **Dipendente** | **Sistema** |
|  | *5* |  | Mostra la [schermata di fallimento](https://github.com/KillTheProcess23/EventManager/blob/master/Requirements%20Analysis%20Document/Mockups/Android%20QR%20Scansionato%20(QR%20non%20valido).png) |
|  | *6* |  | Torna allo step 3 del Main Flow |

## Rappresentazione delle schermate tramite Mock-up

I Mock-up sono rappresentazioni semplificate di quelle che saranno le vere e proprie schermate del software, e chiariscono visivamente il funzionamento di quest’ultimo, descritto nel capitolo precedente tramite i modelli di Cockburn.

Segue una rappresentazione grafica per ogni schermata del programma:

### Applicazione desktop

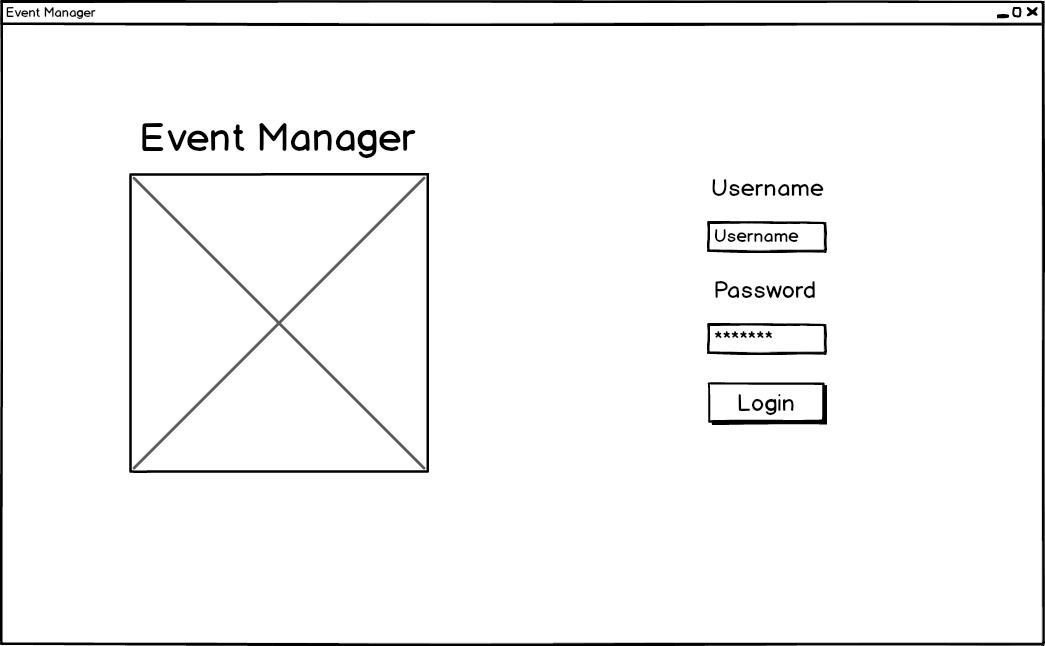


Figura 2: Login da applicazione desktop

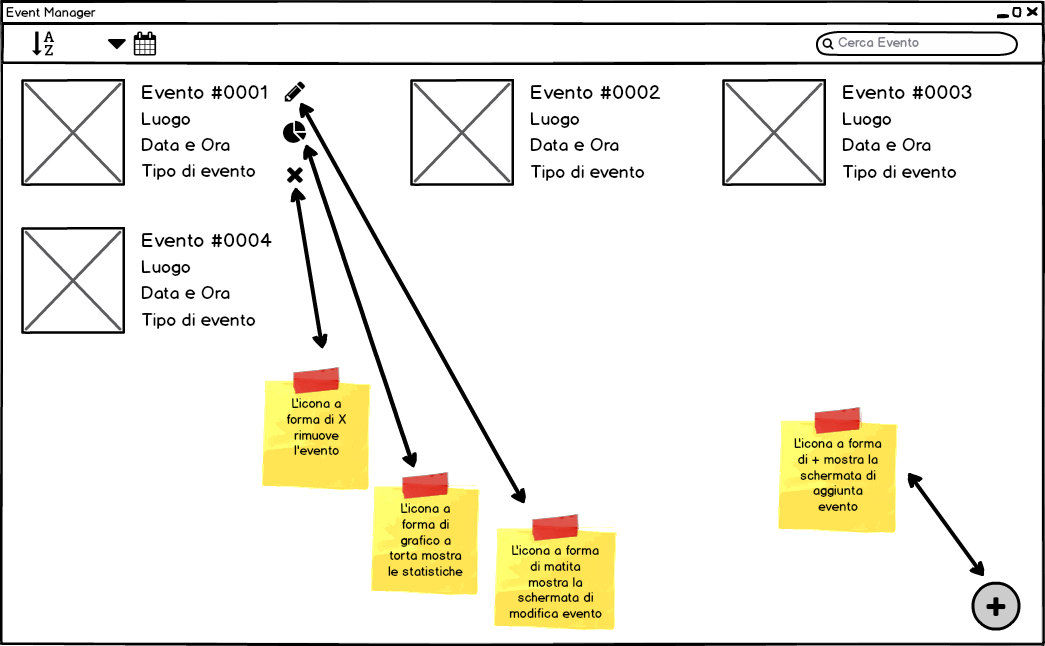


Figura 3: Gestione eventi

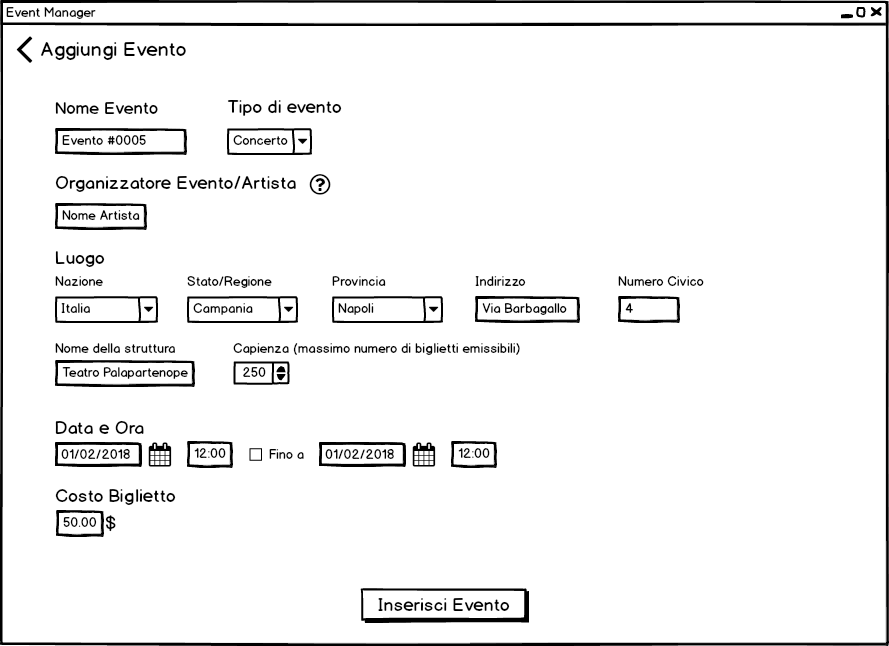


Figura 4: Aggiungi evento

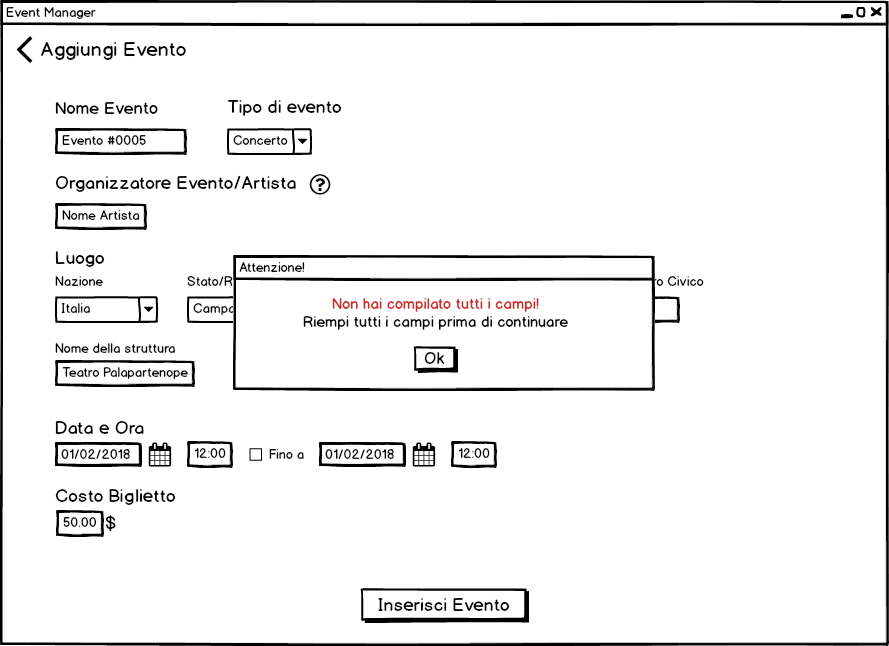


Figura 5: Aggiungi evento (campi non compilati)

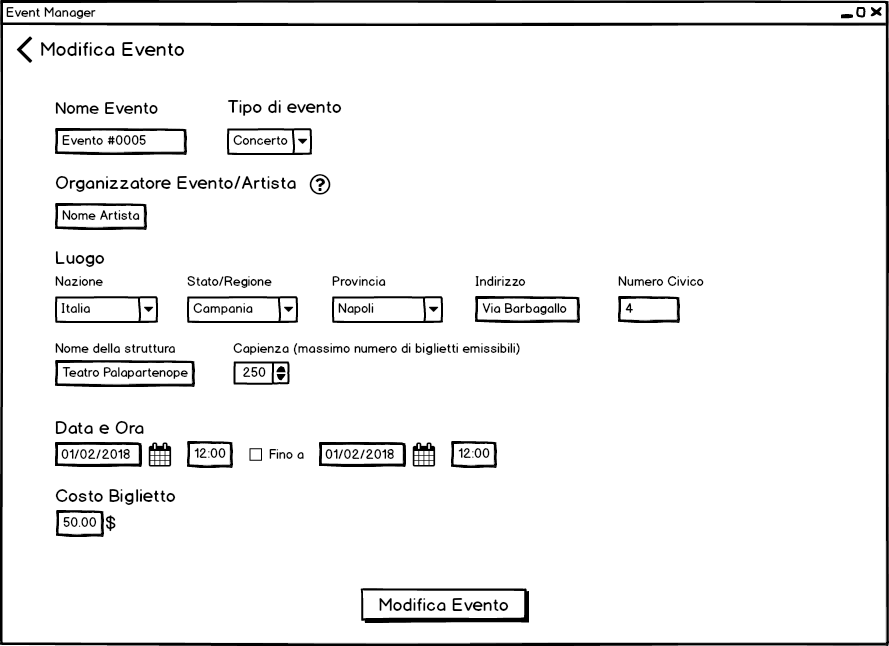


Figura 6: Modifica evento

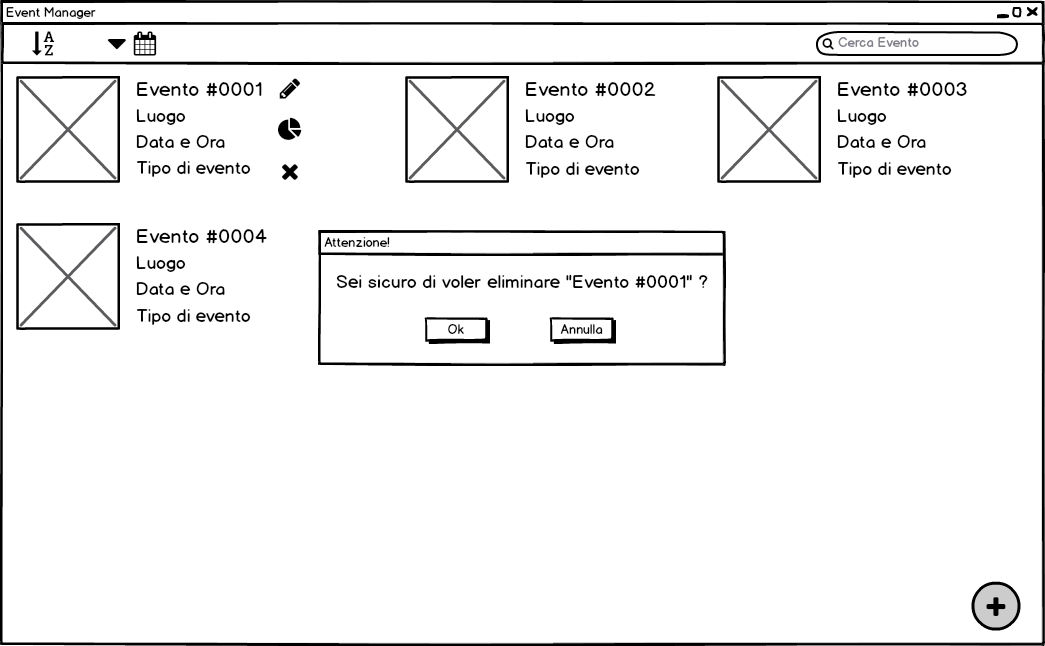
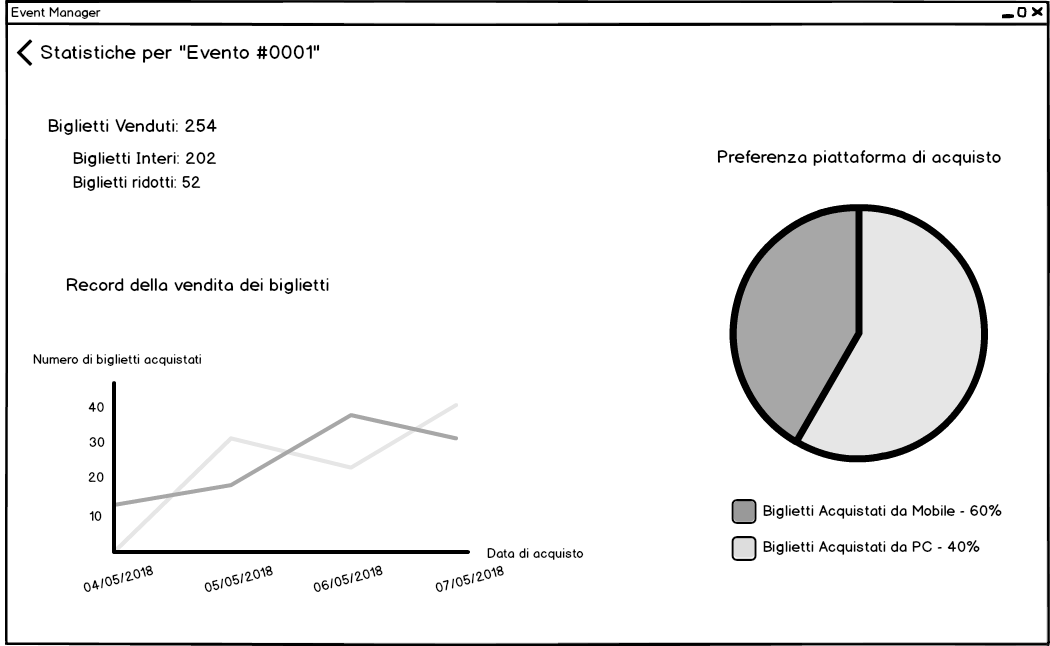


Figura 7: Rimuovi Evento



### Applicazione Android

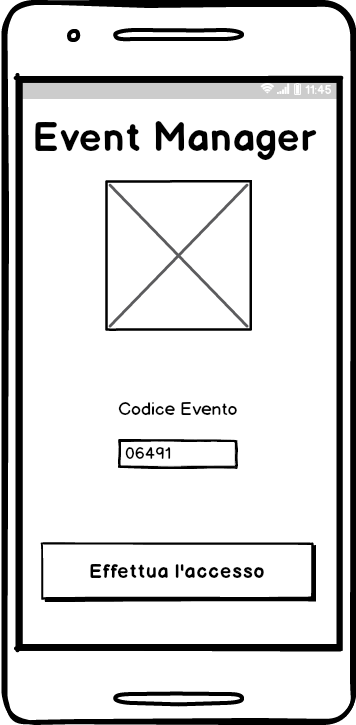


Figura 8: Login da Android

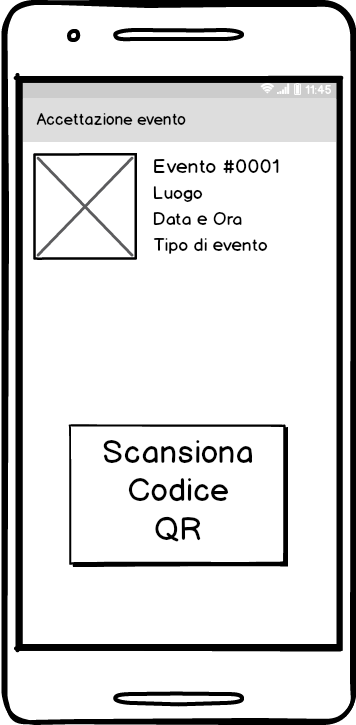


Figura 9: Accettazione evento



Figura 10: Scansione codice QR

Figura 11 e 12: QR code valido/non valido

# Modelli di dominio

## Classi, oggetti e relazioni di analisi

I class diagram mostrano la struttura statica del sistema software. Per individuare le diverse classi si è usata un’euristica three object type, che prevede di identificare tre tipi di oggetti a partire dai diagrammi di Cockburn: oggetti entity, boundary e control.

Gli oggetti boundary rappresentano le interfacce del sistema, quelli control racchiudono la logica di business, mentre quelli entity rappresentano i concetti del dominio che sono persistenti.

Seguono i class diagram del sistema, suddivisi per funzionalità:

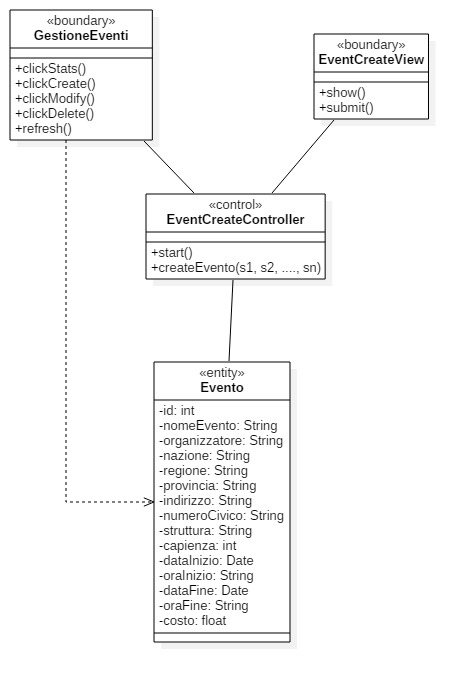


Figura 12: Class Diagram "Aggiungi Evento"

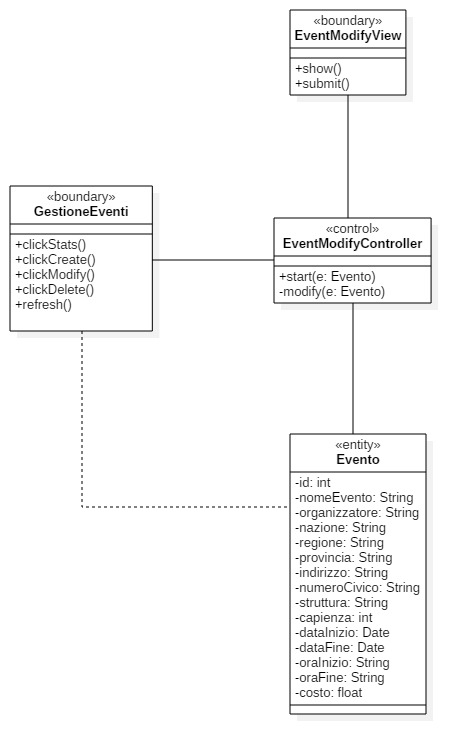


Figura 13: Class Diagram "Modifica Evento"

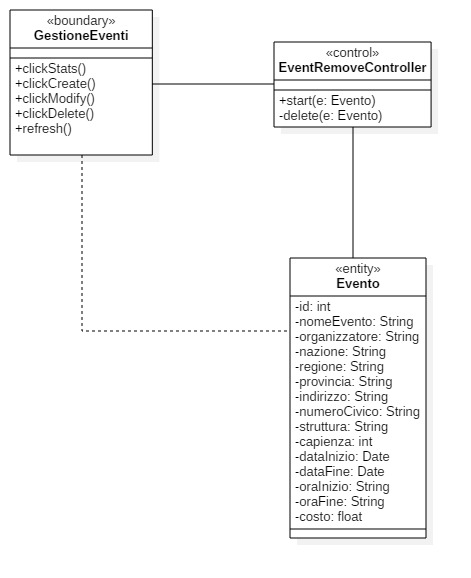


Figura 14: Classi Diagram "Rimuovi Evento"

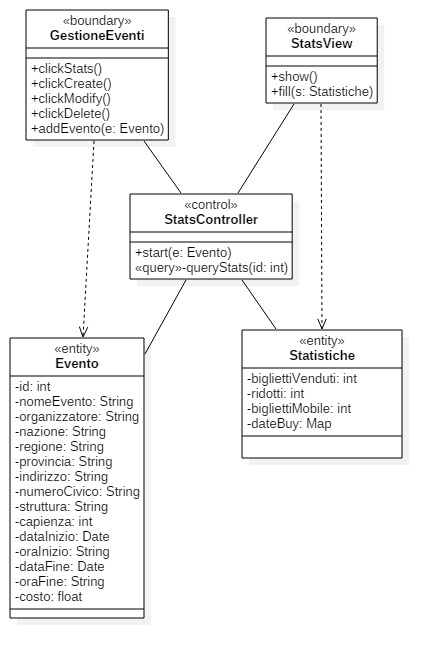


Figura 15: Class Diagram "Visualizza Statistiche"

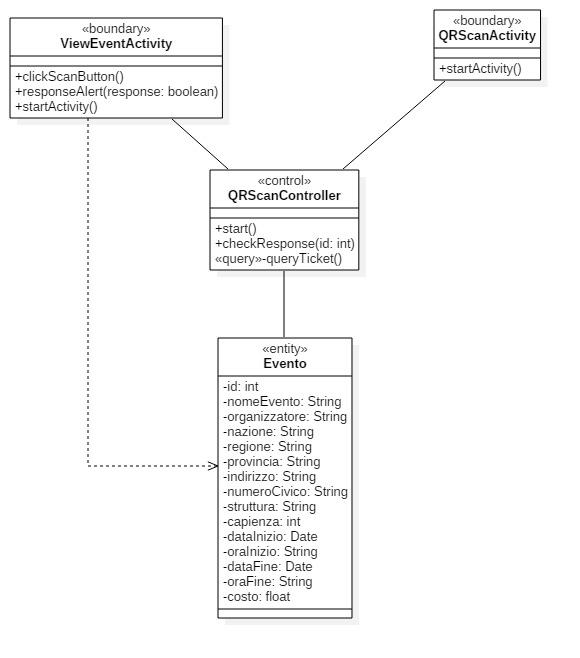


Figura 16: Class Diagram "Scansiona QR Code"

## Diagrammi di sequenza di analisi

I sequence diagram mostrano gli scambi di messaggi necessari tra gli oggetti del sistema per portare a termine delle funzionalità.

Seguono dei sequence diagram che rappresentano gli scenari principali di ogni funzionalità dettagliata tramite il formalismo di Cockburn

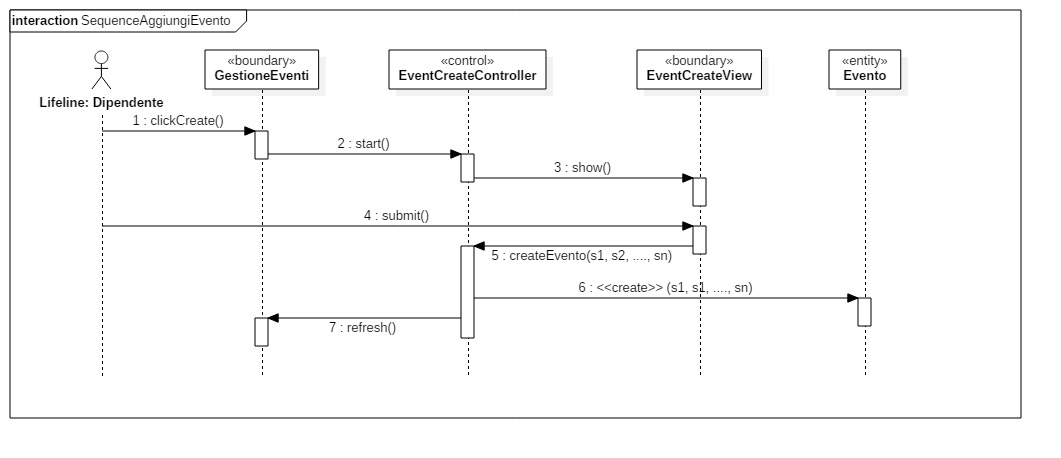


Figura 17: Sequence Diagram "Aggiungi Evento"

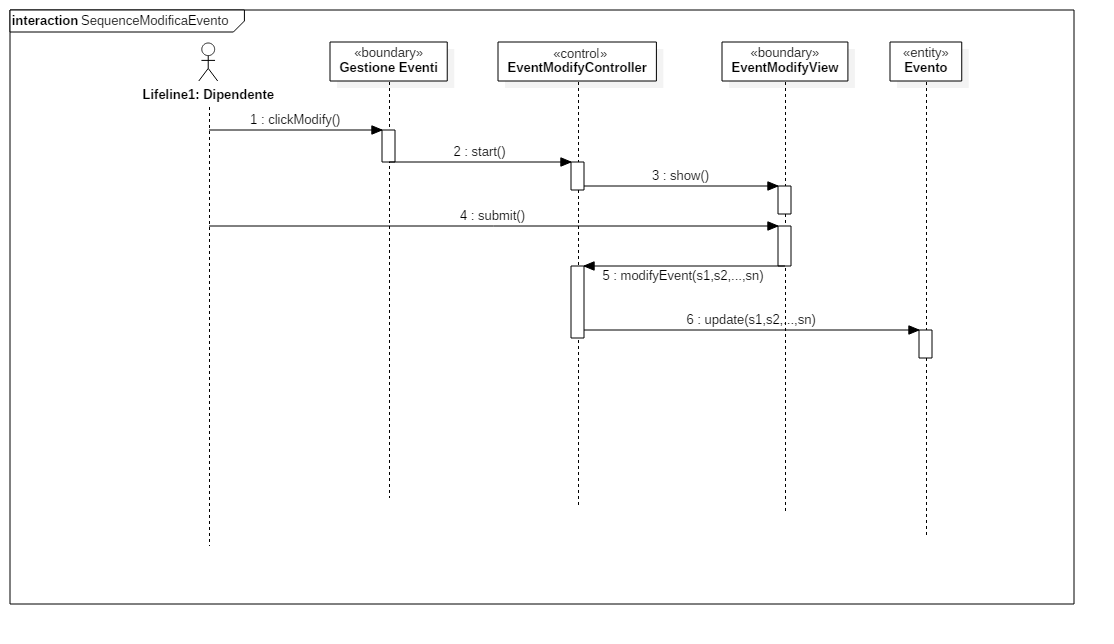


Figura 18: Sequence Diagram "Modifica Evento"

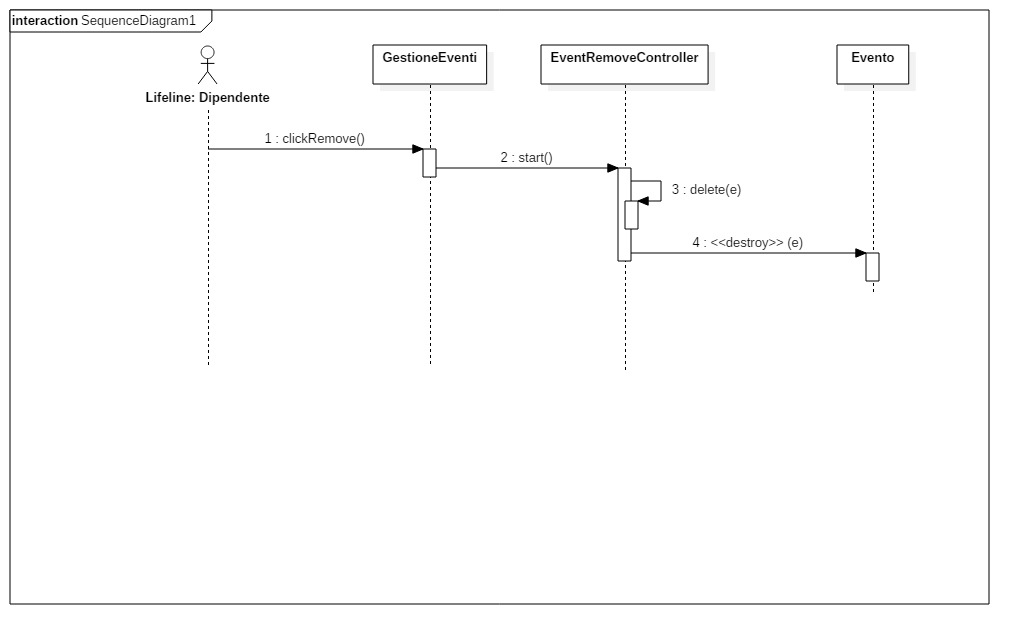


Figura 19: Sequence Diagram "Rimuovi Evento"

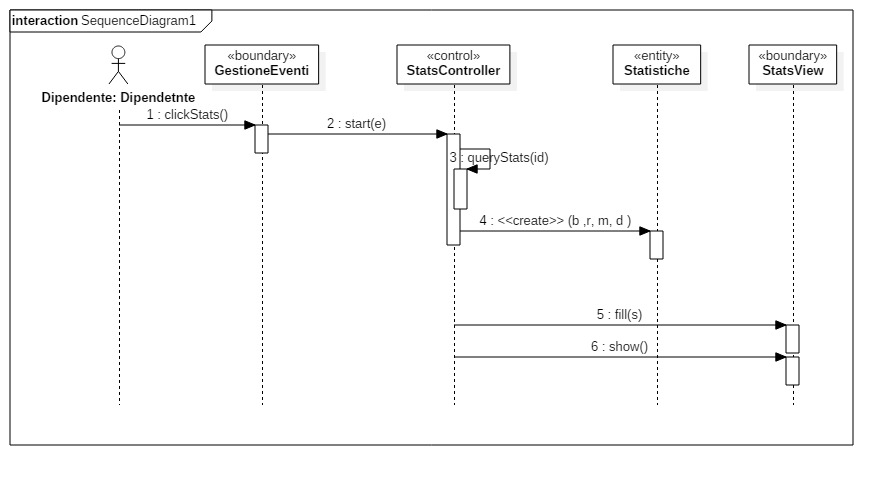


Figura 20: Sequence Diagram "Visualizza Statistiche"

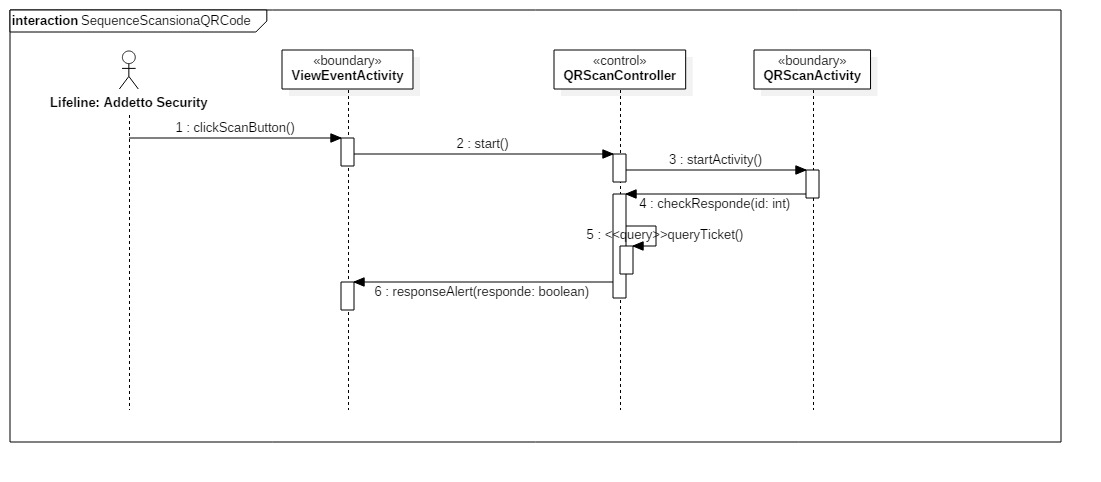


Figura 21: Sequence Diagram "Scansiona QR Code"