****

**Università degli studi di Napoli**

**Federico II**

**anno 2017/2018**

**PROGETTO INGEGNERIA DEL SOFTWARE**

**Progetto em\_17**

**AUTORI:**

**Gruppo 22**

Giuliano Vanesio N86001585

Tommaso Pirozzi

Marcello Quattromani N86001799

Vincenzo Torino

**Capitolo 1: Documento dei requisiti software**

Il sistema è composto da:

* un software per la gestione di eventi, clienti, addetti alla sicurezza, turni di lavoro e visualizzazione statistiche;
* un’applicazione mobile utile per gli addetti alla sicurezza, con cui potranno verificare la validità di un biglietto tramite la lettura del QR-CODE su di esso.

**Requisiti Funzionali**

Per requisiti funzionali intendiamo i requisiti che descrivono i servizi o funzioni offerti dal sistema.

**Caratteristiche generali di un evento**

L’evento avrà le seguenti caratteristiche:

1. il suo ID di identificazione;
2. il titolo;
3. la descrizione;
4. il tipo di evento;
5. il genere di evento in base al tipo;
6. la data;
7. il luogo in cui si svolge.

**Gestione Eventi**

Le operazioni inerenti all’evento sono gestite da parte di un amministratore.

Questo prima di compiere qualsiasi operazione, dovrà autenticarsi tramite il proprio username e la propria password (spiegazione in GESTIONE LOGIN). Dopo avere dunque effettuato l’accesso sarà indirizzato alla schermata principale che avrà delle etichette laterali dalla quale potrà scegliere il campo da gestire.

La prima etichetta è proprio quella inerente agli eventi, dove sarà possibile

* **CERCARE** genericamente un evento tramite qualsiasi suo attributo;
* cercare un evento tramite una **RICERCA AVANZATA**;
* **MODIFICARE** gli attributi di un evento dopo averlo trovato tramite ricerca;
* **CANCELLARE** uno o più eventi dopo averli trovati tramite ricerca;
* **CREARE** un evento.

**Il sistema deve permettere le ricerca di un evento. Tale ricerca può avvenire in due modi:**

1. La parola cercata verrà confrontata con qualsiasi attributo dell’evento così da avere una ricerca generica;
2. Tramite il pulsante ADVANCED SEARCH si può effettuare una ricerca avanzata.

Questa permette dunque all’amministratore di eseguire delle ricerche più mirate e più filtrate.

**Il sistema deve permettere la modifica di un evento:**

* Una volta effettuata una ricerca, se questa va a buon fine mostrerà in una tabella i risultati;
* Gli eventi trovati potranno dunque essere modificati;
* Per apportare una modifica ad un campo di un evento basterà cliccare due volte su di esso e cambiare il suo valore con quello desiderato;
* Fatto ciò basterà premere il pulsante invio tramite il quale si darà conferma della modifica;
* Il campo modificato salverà il nuovo valore.

**Il sistema deve permettere la cancellazione di un evento:**

* Una volta effettuata una ricerca, se questa va a buon fine mostrerà in una tabella i risultati;
* Gli eventi trovati potranno dunque essere cancellati;
* Si può selezionare tramite mouse uno o più eventi che si vogliono cancellare;
* Una volta scelti gli eventi si potranno eliminare tramite il pulsante con il simbolo del cestino.

**Il sistema deve permettere la creazione di un evento:**

* Dalla schermata relativa agli eventi si può cliccare sul pulsante CREATE EVENT il quale aprirà una schermata adatta allo scopo.
* Dalla nuova finestra si dovranno riempire i campi relativi all’evento;
* Dopo aver inserito i dati dell’evento questo potrà essere creato cliccando sul tasto CREATE.

**Caratteristiche generali di un Cliente**

Un cliente dovrà avere le seguenti caratteristiche:

1. ID per identificarlo
2. Username
3. Password
4. Nome
5. Cognome
6. Codice fiscale
7. Anno di nascita
8. Email
9. Numero cellulare

**Gestione Clienti**

I clienti sono gestiti da un amministratore, previa autenticazione, che può CERCARLI, MODIFICARE i loro attributi, CANCELLARLI.

La ricerca può essere effettuata sia ricercando generalmente per tutti gli attributi, sia effettuando una ricerca avanzata.

Analogamente agli eventi, la cancellazione e la modifica degli eventi avviene dopo la ricerca di questi ultimi.

I clienti saranno già presenti nel database in quanto si creeranno in modo autonomo all’iscrizione sul sito apposito.

**Il sistema deve permettere le ricerca di un cliente. Tale ricerca può avvenire in due modi:**

1. La parola cercata verrà confrontata con qualsiasi attributo di un cliente così da ottenere una ricerca generica;
2. Tramite il pulsante ADVANCED SEARCH, invece, si può effettuare una ricerca avanzata. Questa permette dunque all’amministratore di eseguire delle ricerche più mirate e più filtrate.

**Il sistema deve permettere la modifica di un cliente:**

* Una volta effettuata una ricerca, se questa va a buon fine mostrerà in una tabella i risultati;
* I clienti trovati potranno a questo punto essere modificati;
* Per apportare una modifica ad un campo di un cliente basterà cliccare due volte su di esso e cambiare il suo valore con quello desiderato;
* Fatto ciò basterà premere il pulsante invio tramite il quale si darà conferma della modifica;
* Il campo modificato conterrà così il nuovo valore.

**Il sistema deve permettere la cancellazione di un cliente:**

* Una volta effettuata una ricerca, se questa va a buon fine mostrerà in una tabella i risultati;
* I clienti trovati potranno dunque essere cancellati;
* Si può selezionare tramite mouse uno o più clienti che si vogliono cancellare;
* Una volta scelti si potranno eliminare tramite il pulsante con il simbolo del cestino.

**Caratteristiche generali di un addetto alla sicurezza**

L’evento avrà le seguenti caratteristiche:

1. ID per identificarlo
2. Username
3. Password
4. Nome
5. Cognome
6. Codice fiscale
7. Anno di nascita
8. Email
9. Numero cellulare
10. Salario

**Gestione Addetto alla sicurezza**

La gestione di un addetto alla sicurezza avviene in modo totalmente analogo a quella degli eventi.

Si potrà allora:

* **CERCARE** genericamente un addetto tramite qualsiasi suo attributo;
* cercare un addetto tramite una **RICERCA AVANZATA**;
* **MODIFICARE** gli attributi di un addetto dopo averlo trovato tramite ricerca;
* **CANCELLARE** uno o più addetti dopo averli trovati tramite ricerca;
* **CREARE** un addetto.

**Il sistema deve permettere le ricerca di un addetto alla sicurezza. Tale ricerca può avvenire in due modi:**

Analogamente agli eventi

**Il sistema deve permettere la modifica di un addetto alla sicurezza:**

Analogamente agli eventi

**Il sistema deve permettere la cancellazione di un addetto alla sicurezza:**

Analogamente agli eventi

**Il sistema deve permettere la creazione di un addetto alla sicurezza:**

Analogamente agli eventi

**Caratteristiche generali di un turno di lavoro**

Un turno di lavoro avrà le seguenti caratteristiche:

1. ID per identificarlo;
2. ID dell’addetto alla sicurezza;
3. ID dell’evento;
4. Orario inizio turno;
5. Orario fine turno.

**Gestione turno di lavoro**

Un turno di lavoro può essere gestito da un amministratore (previa autenticazione) tramite l’apposita schermata. Si potrà a questo punto gestire i turni di lavoro, ricercandoli sia per un preciso addetto, sia per un preciso evento, sia per entrambi gli attributi. I risultati della ricerca verranno mostrati nella relativa tabella e potranno eventualmente essere rimossi tramite il pulsante del “DELETE” (rappresentato dal cestino) che si attiverà una volta selezionato uno o più turni.

Per creare un nuovo turno di lavoro basterà soltanto compilare ogni campo stando attenti a non superare le 8 ore lavorative (in caso contrario apparirà un messaggio di errore) ed infine cliccare sul pulsante “SET WORK”.

**Il sistema deve permettere la ricerca di un turno di lavoro:**

* La ricerca può essere effettuata riempiendo solo il campo COGNOME relativo all’addetto oppure riempiendo solo il campo TITOLO relativo all’evento o ancora riempiendo entrambi i campi suddetti. Una volta rispettati i precedenti punti bisogna cliccare sul pulsante “CERCA” (rappresentato da una lente)

**Il sistema deve permettere la creazione di un turno di lavoro:**

* Una volta scritto il cognome dell’addetto alla sicurezza, bisogna caricare il codice fiscale tramite il pulsante “ **** ” così da essere sicuri di scegliere il giusto addetto in caso di omonimia;
* Stessa procedura avverrà anche per l’evento da voler selezionare;
* Una volta scelto l’addetto e l’evento, bisogna scegliere l’orario di inizio e fine turno;
* Infine per settare il turno di lavoro basta cliccare sul pulsante “SET WORK”.

**Il sistema deve permettere la cancellazione di un turno di lavoro:**

* Dopo avere effettuato una ricerca, se questa è andata a buone fine, mostrerà i risultati in una tabella;
* Per cancellare uno o più turni di lavoro basta selezionarli nella tabella e cliccare sul pulsante “DELETE” (rappresentato dal cestino).

**Gestione Statistiche:**

Le statistiche potranno essere visualizzate da parte di un amministratore (previa autenticazione) selezionando il tipo da esaminare.

In particolare avremo quattro tipi di statistica:

1. Percentuale di presenze per tipo evento con età media;
2. Biglietti venduti negli anni;
3. Guadagni per anno;
4. Presenze negli anni per un tipo particolare di evento.

Le statistiche verranno mostrate tramite dei grafici che potranno essere o “PIE CHART” o “”LINE CHART”.

**Il sistema deve permettere di mostrare le statistiche in base a quella scelta:**

* Percentuale di presenze per tipo evento con età media: in questo caso verrà generato un grafico cliccando sul pulsante “GENERATE CHART”;
* Biglietti venduti negli anni: bisognerà scegliere il range di anni su cui visualizzare le statistiche;
* Guadagni per anno: analogo al secondo;
* Presenze negli anni per un tipo particolare di evento: oltre alla selezione delle date bisognerà selezionare il tipo di evento sul quale si vuole esaminare le presenze.

**MODELLO FUNZIONALE**

**Use Case diagram**

Con il seguente use case diagram vogliamo descrivere quali sono i servizi e le funzioni offerti dal sistema.

Gli attori rilevati sono:

* Amministratore;
* Addetto alla sicurezza.

I casi d’uso riscontrati per l’amministratore sono:

1. CRUD evento;
2. CRUD addetto alla sicurezza;
3. RUD cliente;
4. Visualizzazione statistiche;
5. Gestione turni lavoro;
6. Effettua il Login.

I casi d’uso rilevati per l’addetto alla sicurezza sono:

1. Login in app;
2. Lettura biglietto tramite QR-code.

Nella pagina successiva sarà mostrato l’use case diagram realizzato.

Immagine che contiene testo, mappa

Descrizione generata con affidabilità molto elevata

**Mock Up**

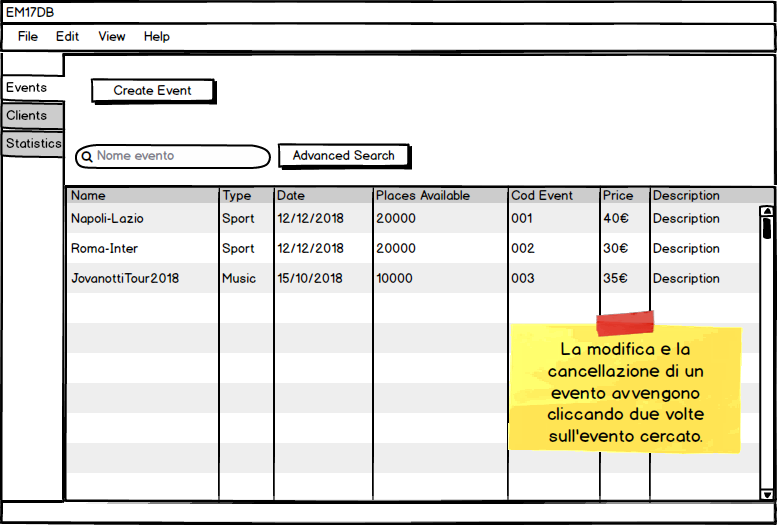
Nella seguente sezione viene illustrato come potrebbe presentarsi l’applicativo desktop e l’app mobile. L’idea grafica di base è stata di rendere il più semplice possibile l’utilizzo dei due software, i mock up sono una rappresentazione di massima dei sistemi sviluppati, il loro reale aspetto potrebbe differire.   
I mock up presentati saranno:

* Event;
* Customer;
* Create Event;
* Advanced Search Event;
* Work Shift Officer;
* Statistics;
* Error message;
* Success message;
* Confirm message.

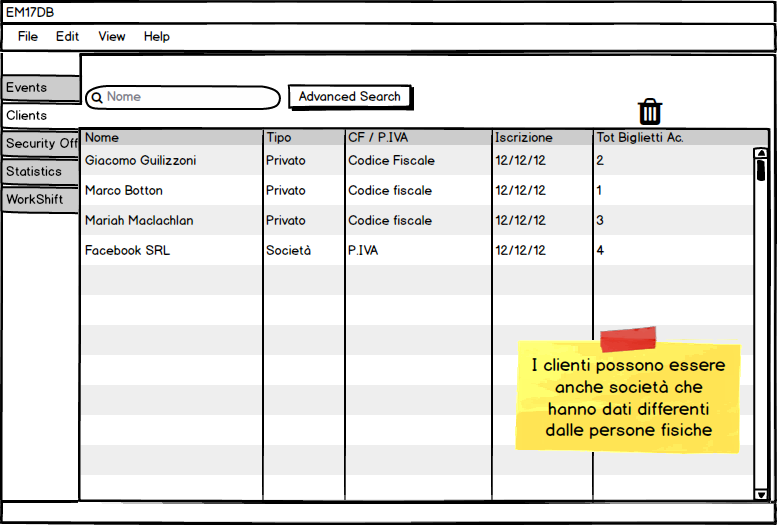
I mock up relativi all’addetto alla sicurezza non sono stati rappresentati in quanto analoghi a quelli dell’evento.

Di seguito verranno mostrati tutti i mock up rappresentati.

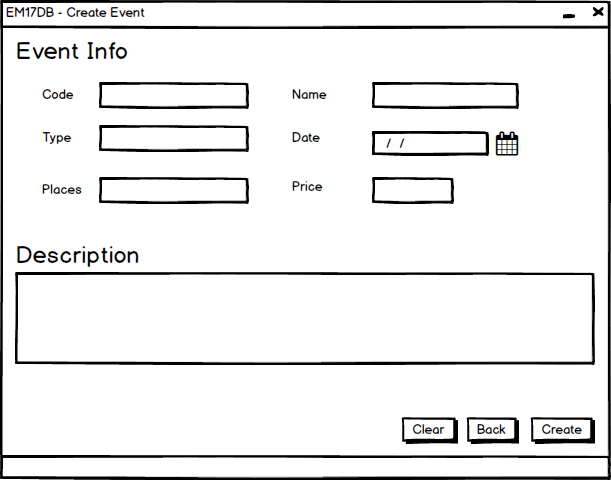
Mock Up EVENT



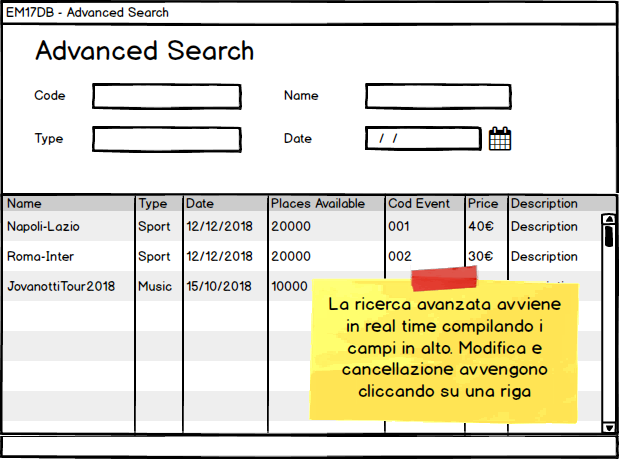
Mock Up CUSTOMER



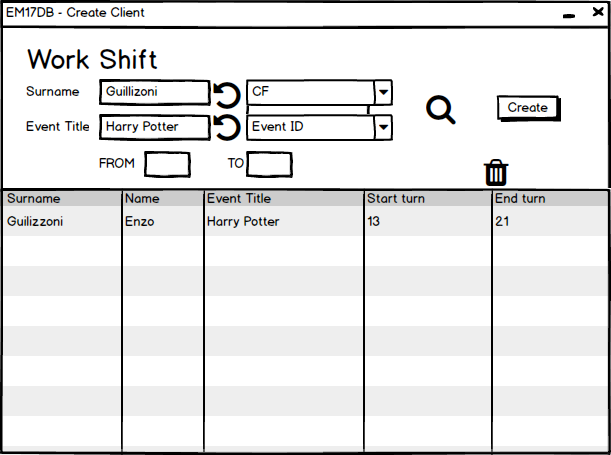
Mock Up CREATE EVENT



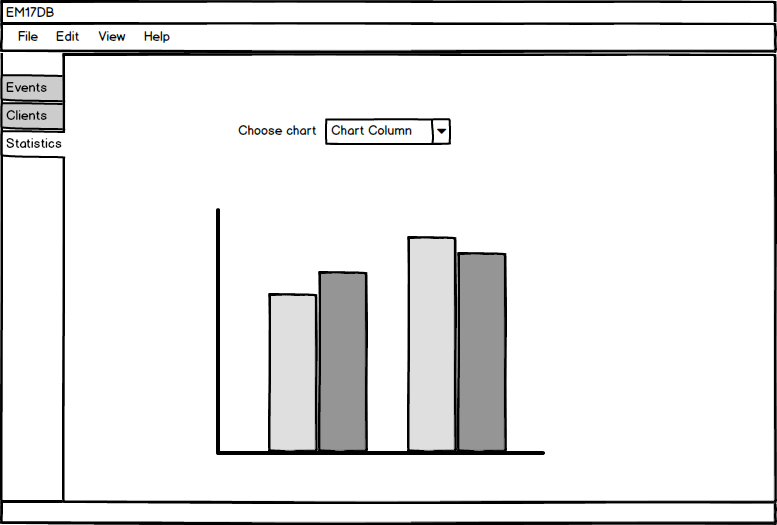
Mock Up ADVANCED SEARCH EVENT



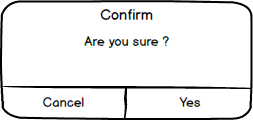
Mock Up WORK SHIFT

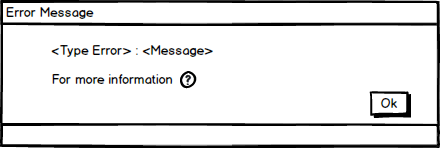


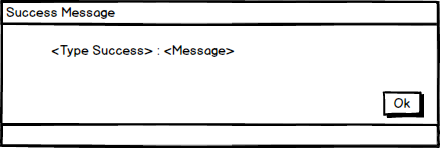
Mock Up STATISTICS



Mock Up MESSAGES







Modelli di dominio

Class diagram database

Di seguito il class diagram, rappresentato tramite UML, inerente alle relazioni delle classi (non dettagliate) presenti nel database.

**CAPITOLO 2: Design del Sistema**

Analisi dell’architettura

Il sistema presenta un’architettura repository: l’applicativo desktop e l’app mobile accedono e modificano il database memorizzato sul server.

I sottosistemi sono indipendenti tra loro, difatti i due non comunicano mai tra di loro direttamente.

Il seguente schema presenterà il tipo di architettura:

**SERVER**

**CLASS DIAGRAM**

Per quanto riguarda lo sviluppo dell’applicativo desktop sono stati classificati tre tipi di oggetti come Entity, Boundery, Control in quanto è stato seguito il tipo di euristica Three-Object-Type.

Specificando i tre tipi di oggetto:

* Entity: oggetti che modellano un’informazione persistente;
* Boundary: oggetti che modellano le interazioni tra attori e sistema;
* Control: oggetti che modellano la logica che si occupa di realizzare gli use case

Nella pagina successiva viene mostrato il class diagram realizzato.

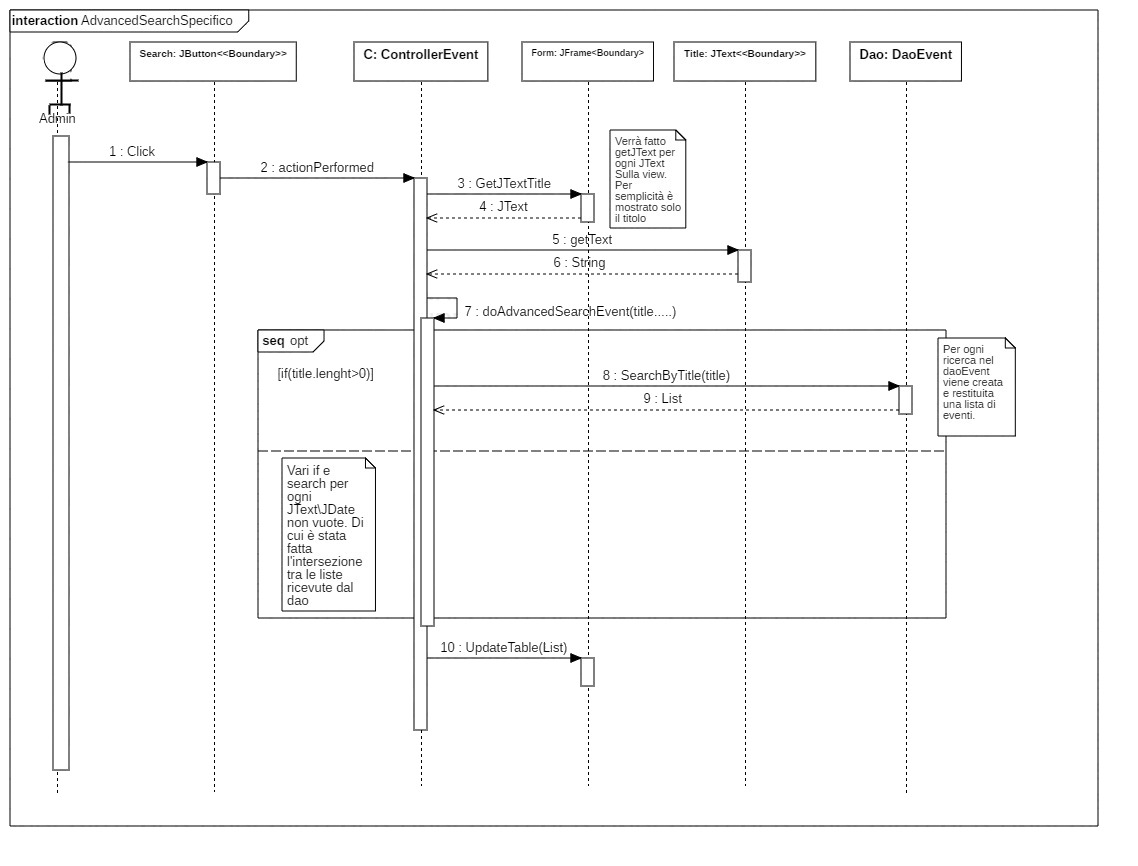
**SEQUENCE DIAGRAM**

In questa sezione verranno analizzati i sequence diagram realizzati per l’applicativo desktop. In particolare verranno dettagliati i sequence diagram relativi al caso d’uso attinente agli eventi.

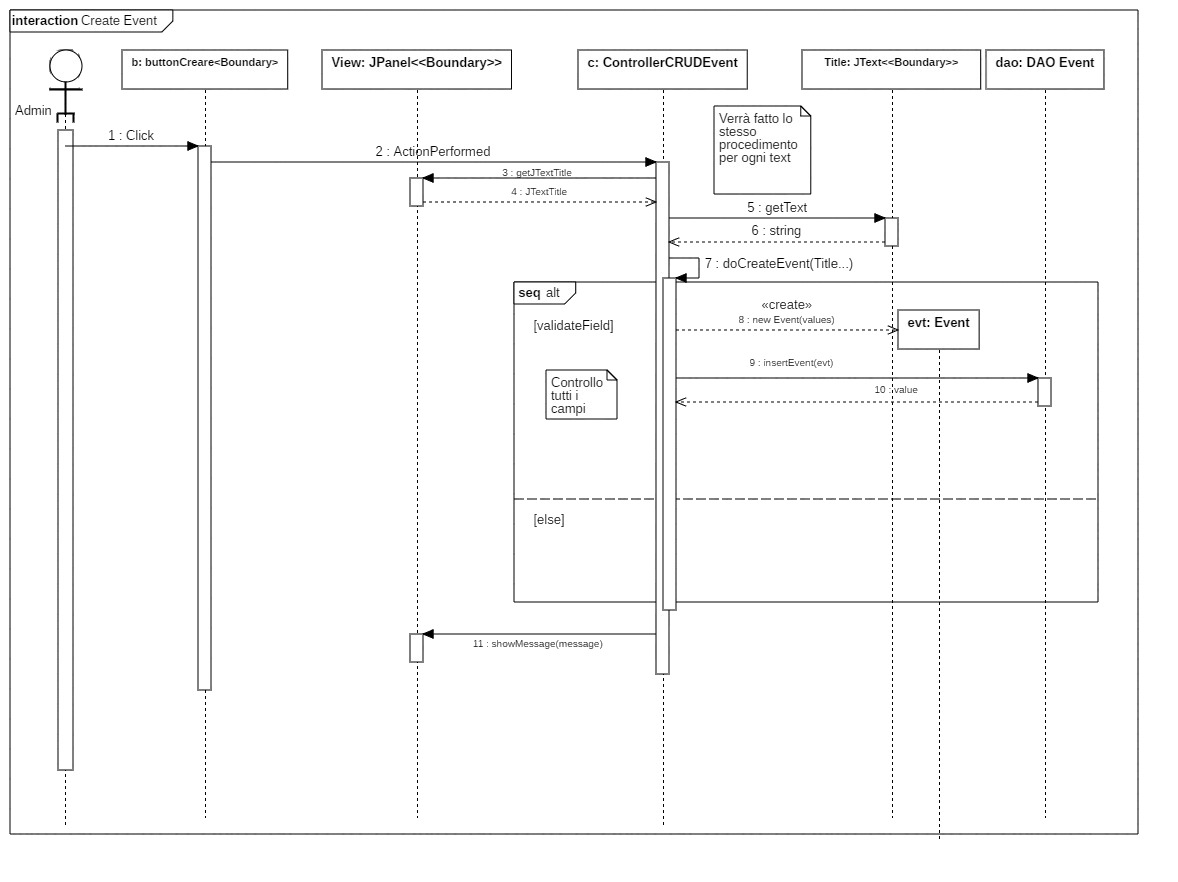
I sequence diagram allegati nelle pagine successive riguardano:

1. Ricerca avanzata;
2. Creazione di un evento;
3. Cancellazione di uno o più eventi;
4. Modifica di un evento;
5. Statistiche.

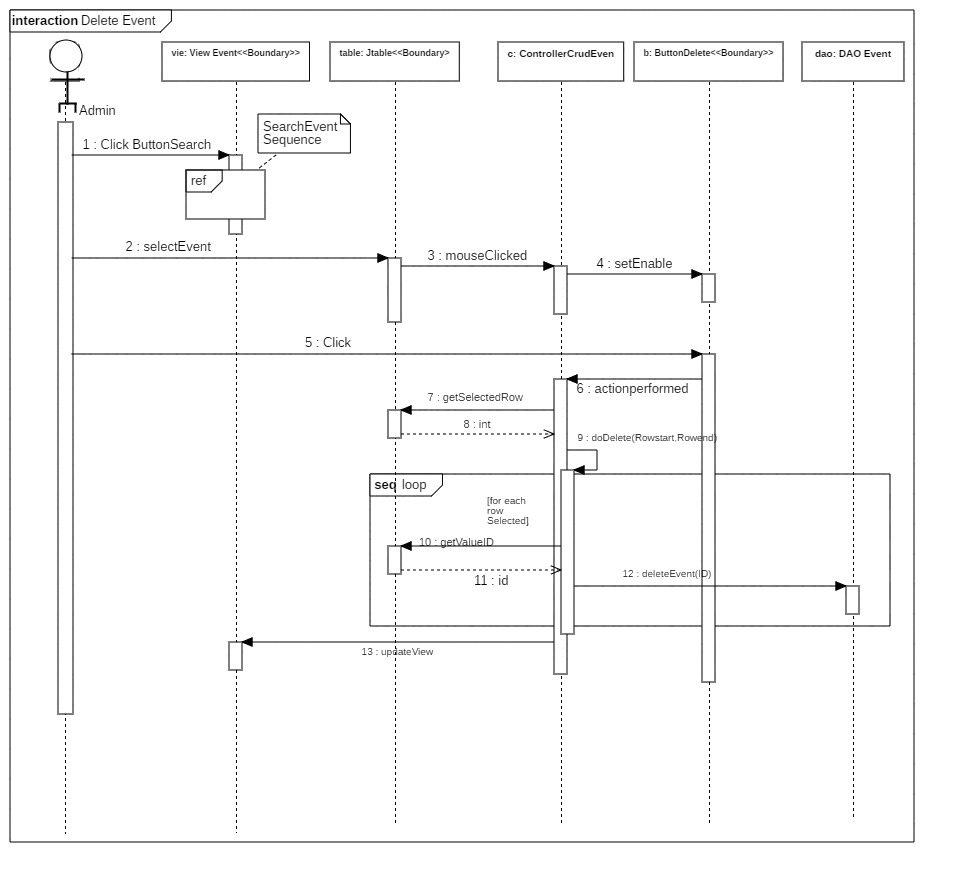
Sequence diagram: Ricerca avanzata



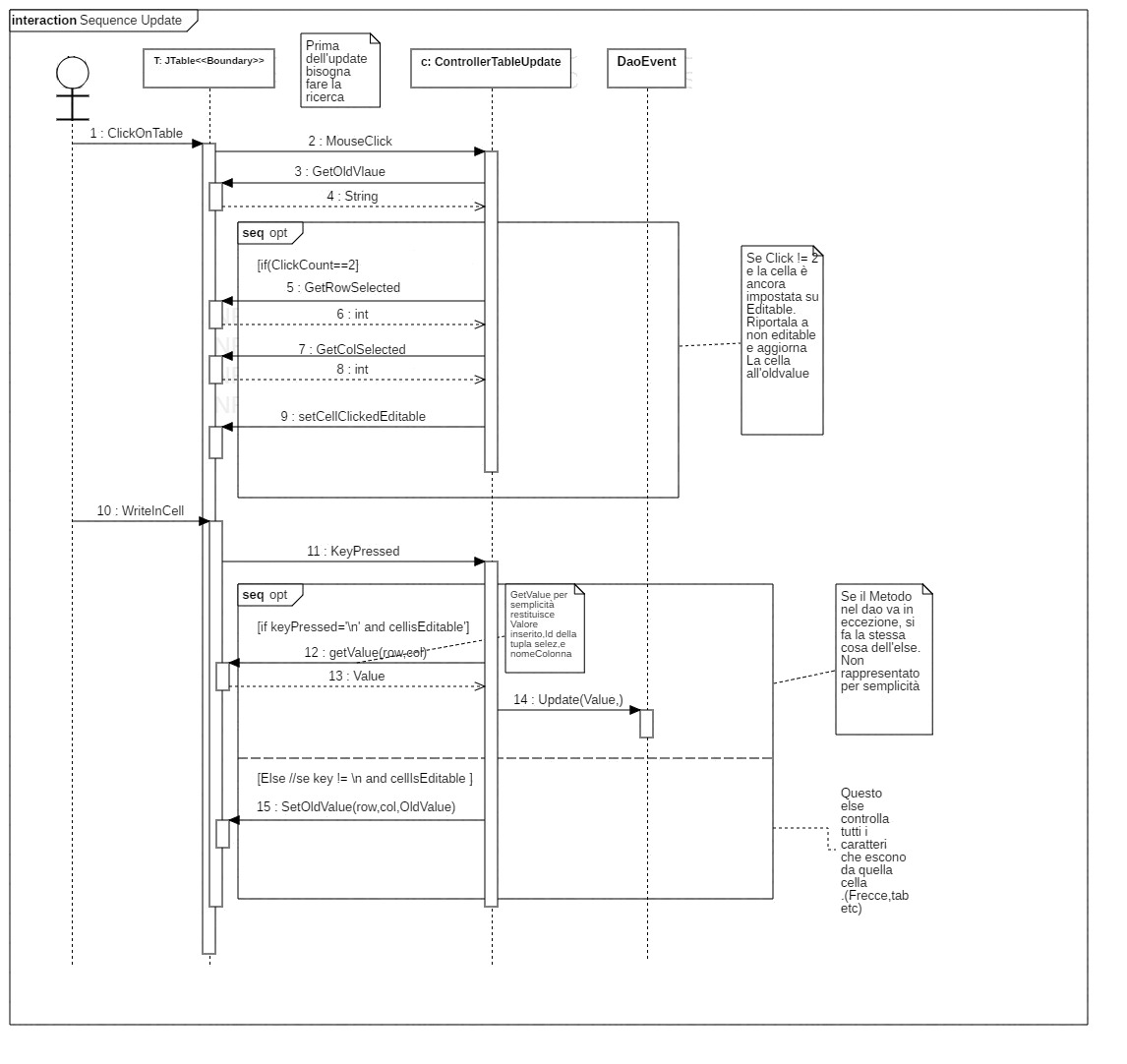
Sequence diagram: Creazione Evento



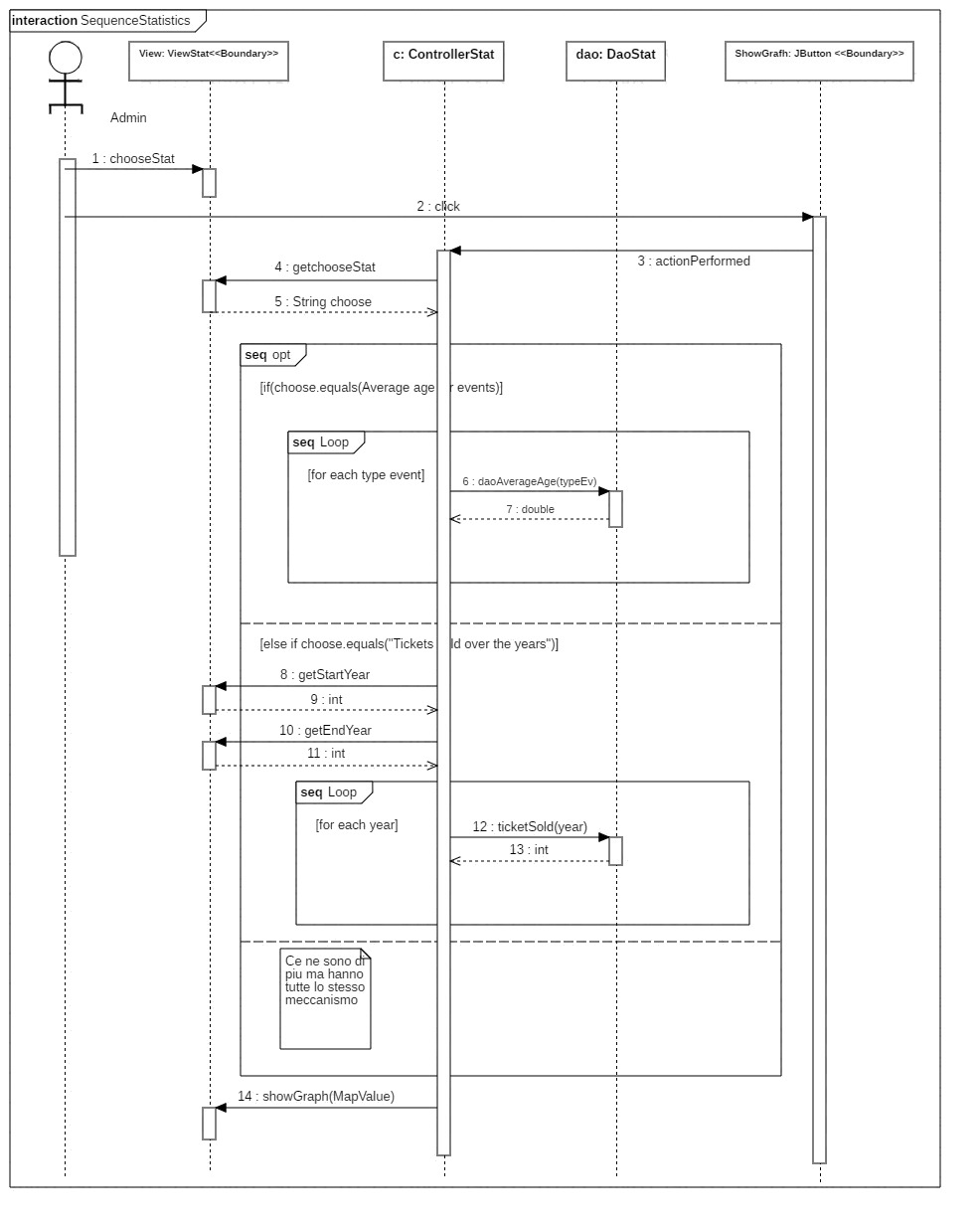
Sequence diagram: Cancellazione evento



Sequence diagram: Modifica evento



Sequence diagram: Statistiche



CRC CARDS APPLICATIVO DESKTOP

Verranno di seguito introdotte le CRC cards (Class Responsability Collaboration) per dettagliare le collaborazioni tra le classi e le loro responsabilità.

|  |  |
| --- | --- |
| **Class name** | ImplMySqlEventDao |
| **Super Class** | Implements EventDao |
| **Sub Class** | - |
| **Responsabilità** | **Collaboratori** |
| La classe ImplMySqlEventDao rappresenta i dati correlati agli eventi presenti sul database. E’ stata creata seguendo le direttive del design pattern: DataAccessObject. | Event |

|  |  |
| --- | --- |
| **Class name** | ImplMySqlAddettiDao |
| **Super Class** | Implements AddettiDao |
| **Sub Class** | - |
| **Responsabilità** | **Collaboratori** |
| La classe implMySqlAddettiDao rappresenta i dati correlati agli addetti alla sicurezza presenti sul database. E’ stata creata seguendo le direttive del design pattern: DataAccessObject. | Addetto |

|  |  |
| --- | --- |
| **Class name** | implMySqlCustomerDao |
| **Super Class** | Implements CustomerDao |
| **Sub Class** | - |
| **Responsabilità** | **Collaboratori** |
| La classe implMySqlCustomer Dao rappresenta i dati correlati ai clienti presenti sul database. E’ stata creata seguendo le direttive del design pattern: DataAccessObject. | Customer |

|  |  |
| --- | --- |
| **Class name** | implMySqlTurnDao |
| **Super Class** | Implements TurnDao |
| **Sub Class** | - |
| **Responsabilità** | **Collaboratori** |
| La classe implMySqlTurnDao rappresenta i dati correlati ai turni degli addetti agli eventi presenti sul database. E’ stata creata seguendo le direttive del design pattern: DataAccessObject. | Turn  Event  Addetto |

|  |  |
| --- | --- |
| **Class name** | ListenerPanel (Abstract) |
| **Super Class** | Implements KEyListener,ActionListener,MouseListener |
| **Sub Class** | - |
| **Responsabilità** | **Collaboratori** |
| La classe si occupa di gestire i vari listener dei Panel.Per esempio attivazione e disattivazione di button etc. E’ stata resa classe in quanto comune a diversi panel. | GeneralPanel. |

|  |  |
| --- | --- |
| **Class name** | ControllerCRDEvent |
| **Super Class** | ListenerPanel |
| **Sub Class** | - |
| **Responsabilità** | **Collaboratori** |
| ControllerCRDEvent è un controllore del caso d’uso “CRUDEvent” descritto nella documentazione e si occupa di creare,eliminare e cercare gli eventi. Le sue responsabilità sono quelle tipiche di un controllore nella rappresentazione “Entity-Boundary-Control” | EventDao  EventPanel  Event |

|  |  |
| --- | --- |
| **Class name** | ListenerTable(Abstract) |
| **Super Class** | Implements MouseListener,KeyListener |
| **Sub Class** | - |
| **Responsabilità** | **Collaboratori** |
| ListenerTable è una classe che permette di gestire i listener di un oggetto JTAble. | JTable |

|  |  |
| --- | --- |
| **Class name** | ControllerUpdateEvent |
| **Super Class** | ListenerTable |
| **Sub Class** | - |
| **Responsabilità** | **Collaboratori** |
| ControllerUpdateEvent è un controllore del caso d’uso “CRUDEvent” descritto nella documentazione e si occupa di effettuare update degli eventi. Le sue responsabilità sono quelle tipiche di un controllore nella rappresentazione “Entity-Boundary-Controll” | EventPanel  Event Dao  Event |

|  |  |
| --- | --- |
| **Class name** | ControllerCRDAddetto |
| **Super Class** | ListenerPanel |
| **Sub Class** | - |
| **Responsabilità** | **Collaboratori** |
| ControlleCRDEvent è un controllore del caso d’uso “CRUDAddetti” descritto nella documentazione e si occupa di creare,eliminare e cercare gli addetti. Le sue responsabilità sono quelle tipiche di un controllore nella rappresentazione “Entity-Boundary-Controll” | AddettiDao  AddettiPanel.  Addetto |

|  |  |
| --- | --- |
| **Class name** | ControllerUpdateAddetto |
| **Super Class** | ListenerTable |
| **Sub Class** | - |
| **Responsabilità** | **Collaboratori** |
| ControlleCRDEvent è un controllore del caso d’uso “CRUDAddetti” descritto nella documentazione e si occupa di fare update degli addetti. Le sue responsabilità sono quelle tipiche di un controllore nella rappresentazione “Entity-Boundary-Controll”. | AddettiDao  AddettiPanel.  Addetto |

|  |  |
| --- | --- |
| **Class name** | ControllerCRUDTurn |
| **Super Class** | ListenerTable |
| **Sub Class** | - |
| **Responsabilità** | **Collaboratori** |
| ControlleCRUDTurn è un controllore del caso d’uso “CRUDTurn” descritto nella documentazione e si occupa di creare,eliminare e ricercare i turni di un evento o di un addetto. Le sue responsabilità sono quelle tipiche di un controllore nella rappresentazione “Entity-Boundary-Controll”. | TurnDao  ManagementTurnView  Addetto  Evento |

|  |  |
| --- | --- |
| **Class name** | ControllerRDCustomer |
| **Super Class** | ListenerPanel |
| **Sub Class** | - |
| **Responsabilità** | **Collaboratori** |
| ControlleRDEvent è un controllore del caso d’uso “RUDCustomer” descritto nella documentazione e si occupa di creare ed eliminare i customer. Le sue responsabilità sono quelle tipiche di un controllore nella rappresentazione “Entity-Boundary-Controll”. | CustomerDao  CustomerPanel  Customer |

|  |  |
| --- | --- |
| **Class name** | ControllerUpdateCustomer |
| **Super Class** | ListenerTable |
| **Sub Class** | - |
| **Responsabilità** | **Collaboratori** |
| ControlleRDEvent è un controllore del caso d’uso “RUDCustomer” descritto nella documentazione e si occupa degli update di un customer. Le sue responsabilità sono quelle tipiche di un  controllore nella rappresentazione “Entity-Boundary-Controll”. | Customer Dao.  CustomerPanel.  Customer |

|  |  |
| --- | --- |
| **Class name** | ControllerStatistics |
| **Super Class** | Implements ActionListener,ItemListener |
| **Sub Class** | - |
| **Responsabilità** | **Collaboratori** |
| ControlleStatistics è un controllore del caso d’uso “Check Statistics” descritto nella documentazione . | DaoStatistics  PanelStats |

|  |  |
| --- | --- |
| **Class name** | ControllerLogin |
| **Super Class** | ListenerPanel |
| **Sub Class** | - |
| **Responsabilità** | **Collaboratori** |
| ControlleLogin è un controllore del caso d’uso “Login” descritto nella documentazione e si occupa di far effettuare il login al database. Le sue responsabilità sono quelle tipiche di un controllore nella rappresentazione “Entity-Boundary-Controll”. | LoginView |

|  |  |
| --- | --- |
| **Class name** | DBConnect |
| **Super Class** | - |
| **Sub Class** | - |
| **Responsabilità** | **Collaboratori** |
| La classe si occupa di creare la connessione all’database e, quando richiesto, fornire la connessione creata. | - |

|  |  |
| --- | --- |
| **Class name** | MainView |
| **Super Class** |  |
| **Sub Class** | - |
| **Responsabilità** | **Collaboratori** |
| Frame che contiene un JTabbedPane, la quale conterrà tutti i panel presenti nel software. |  |

|  |  |
| --- | --- |
| **Class name** | GeneralPanel(Abstract) |
| **Super Class** | - |
| **Sub Class** | - |
| **Responsabilità** | **Collaboratori** |
| Classe astratta che contiene elementi di grafica. |  |

|  |  |
| --- | --- |
| **Class name** | EventPanel |
| **Super Class** | GeneralPanel |
| **Sub Class** | - |
| **Responsabilità** | **Collaboratori** |
| Interfaccia grafica per gestire gli eventi. |  |

|  |  |
| --- | --- |
| **Class name** | AddettiPanel |
| **Super Class** | GeneralPanel |
| **Sub Class** | - |
| **Responsabilità** | **Collaboratori** |
| Interfaccia grafica per gestire gli addetti. |  |

|  |  |
| --- | --- |
| **Class name** | CustomerPanel |
| **Super Class** | GeneralPanel |
| **Sub Class** | - |
| **Responsabilità** | **Collaboratori** |
| Interfaccia grafica per gestire i clienti. |  |

|  |  |
| --- | --- |
| **Class name** | ManagementTurnView |
| **Super Class** | GeneralPanel |
| **Sub Class** | - |
| **Responsabilità** | **Collaboratori** |
| Interfaccia grafica per gestire i turni degli addetti. |  |

|  |  |
| --- | --- |
| **Class name** | LoginView |
| **Super Class** | - |
| **Sub Class** | - |
| **Responsabilità** | **Collaboratori** |
| Interfaccia grafica che permette il login all’applicativo. |  |

|  |  |
| --- | --- |
| **Class name** | StatsPanel |
| **Super Class** | - |
| **Sub Class** | - |
| **Responsabilità** | **Collaboratori** |
| Interfaccia che permette di visualizzare le statistiche. |  |

|  |  |
| --- | --- |
| **Class name** | Customer |
| **Super Class** | - |
| **Sub Class** | - |
| **Responsabilità** | **Collaboratori** |
| Contiene Tutti i dati del singolo customer. |  |

|  |  |
| --- | --- |
| **Class name** | Addetto |
| **Super Class** | - |
| **Sub Class** | - |
| **Responsabilità** | **Collaboratori** |
| Contiene Tutti i dati del singolo Addetto. |  |

|  |  |
| --- | --- |
| **Class name** | Event |
| **Super Class** | - |
| **Sub Class** | - |
| **Responsabilità** | **Collaboratori** |
| Contiene Tutti i dati del singolo evento. |  |

|  |  |
| --- | --- |
| **Class name** | Turn |
| **Super Class** | - |
| **Sub Class** | - |
| **Responsabilità** | **Collaboratori** |
| Contiene Tutti i dati del singolo turno |  |