|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  | | --- | | ENTE/I: INGSW – GRUPPO 32  DATA EMISSIONE: A.A. 2017/18  PAG. 1/12 | | |  | | --- | | **Documento di testing** | |  | |

|  |  |
| --- | --- |
| |  | | --- | | **OGGETTO:**  **Piano di test del Sistema Informativo “Event Manager ‘17”** | |

|  |  |
| --- | --- |
| |  | | --- | | **SINTESI DEI CONTENUTI:**  Il documento contiene le seguenti informazioni e linee di attività riguardanti il test delle funzionalità del sistema Event Manager’17:   1. Introduzione 2. Caratteristiche da testare/escludere 3. Approcci 4. System testing 5. Unit testing | |

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

# Sommario

1. Sommario 2

2. Revisioni 3

3. Introduzione 4

A. Relazioni con altri documenti 4

4. Caratteristiche da testare/escludere 5

5. Approcci 6

6. System Testing 7

7. Unit Testing …………………………………………………………………………………………………………………………………………11

# Revisioni

|  |  |  |  |
| --- | --- | --- | --- |
| Data | Versione | Autore | Descrizione |
| 06/06/2018 | 0.1 | Alessandro Pianese | Creato il document e stilato il piano di testing |
| 07/06/2018 | 0.2 | Alessandro Pianese | Aggiunto codice JUnit |

# Introduzione

Questo documento ha lo scopo di definire un piano di testing per tutte le funzionalità concordate con il committente e mostrare una serie di test automatici (sviluppati con jUnit) relativa a due specifici metodi di una classe del software Event Manager’17.

## **Relazioni con altri documenti**

I requisiti del software sono specificati nel “*Documento dei requisiti software”*, mentre la struttura del sistema è trattata nel “*Documento di design”.*

# Caratteristiche da Testare o escludere

Saranno escluse dal piano di testing le funzionalità non concordate con il committente, quali:

* Visualizzazione, su un sito web, degli eventi disponibili.
* Acquisto biglietto per un evento, dopo la visualizzazione del punto di cui sopra.
* Gestione dei clienti
  + Visulizzazione dati relativi ad un cliente
  + Cancellazione di un cliente
* Gestione Addetti

Le funzionalità prese in considerazione saranno:

* Controllo Accesso all’evento da parte di un addetto alla security
  + Accesso alla schermata di accettazione evento
  + Scansione codice QR
* Gestione Eventi
  + Login Dipendenti
  + Inserimento nuovo evento
  + Modifica evento esistente
  + Cancellazione evento
* Generazione Statistiche relative ad uno o più eventi

La classe presa in considerazione per lo Unit Testing è *ViewEventActivity* presente nella componente android del progetto e, più nello specifico, i metodi *checkEvento* e *checkTicket*.

# Approcci

Sia per il System Testign sia per lo Unit Testign verrò usato un approccio *Black Box*, ovvero non considereremo il codice sorgente ma ci concentreremo sul funzionamento esterno del software, quindi dal punto di vista dell’utente.

Nella scelta dei casi di test, per il System Testing, sarà utilizzata l’euristica WECT (Weak Equivalence Class Testing), la quale prevede che per ogni classe di equivalenza ci sia un Test Case che usi un valore nominale che la rappresenti.

# System Testing

Qui sono elencati tutti i casi di test per l’intero sistema:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TEST ID** | 1 | | | |
| **TEST NAME** | Test Accesso Evento | | | |
| **TEST** **DESCRIPTION** | Verifica della funzionalità di accesso all’accetazione dell’evento | | | |
| **INPUT** | **RISULTATO DESIDERATO** | | **RISULTATO OTTENUTO** | |
| Inserire “19466” nel campo Codice | | Codice riconosciuto dal Sistema e caricamento della schermata di accettazione | |  |
| Inserire “10101” nel campo Codice | | Codice non riconosciuto dal Sistema, si mostra un messagio di errore | |  |
| Non inserire niente nel campo Codice | | Codice non riconosciuto dal Sistema, si mostra un messaggio di errore | |  |
| **NOTE** | Il database deve contenere l’evento con codice “19466” il database non deve contenere l’evento con codice “10101” | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **TEST ID** | 2 | | |
| **TEST NAME** | Test Scansione QR Code | | |
| **TEST** **DESCRIPTION** | Verifica il corretto scansionamento dei codici QR | | |
| **INPUT** | **RISULTATO DESIDERATO** | **RISULTATO OTTENUTO** | |
| Si inquadra il codice QR contenente la stringa “45671234” | Il Sistema mostra una finestra di dialogo con il messaggio “Biglietto valido” | |  |
| Si inquadra il codice QR contenente la stringa “67892345” | Il Sistema mostra una finestra di dialogo con il messaggio “Biglietto non valido” | |  |
| Si Inquadra il codice QR contenente la stringa “12345678” | Il Sistema mostra una finestra di dialogo con il messaggio “Biglietto non valido” | |  |
| **NOTE** | Nel database è presente il biglietto “45671234” ed è associato all’evento che stiamo scansionando.  Nel database è presente il biglietto “67892346” e non è associato all’evento che stiamo scansionando. Nel database non è presente il biglietto “12345678” | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TEST ID** | 3 | | | |
| **TEST NAME** | Test Login Dipendenti | | | |
| **TEST** **DESCRIPTION** | Verifica della funzionalità di login per I dipendenti | | | |
| **INPUT** | **RISULTATO DESIDERATO** | | **RISULTATO OTTENUTO** | |
| Inserire “Roberto” nel campoUser, “password” nel campo Password | | Login effettuato, visualizza la schermata di gestione eventi | |  |
| Inserire “utente” nel campo User, “password” nel campo Password | | Login fallito, si visualizza un messaggio di errore | |  |
| Inserire “Roberto” nel campo User, “altraPassword” nel campo Password | | Login fallito, si visualizza un messaggio di errore | |  |
| **NOTE** | Il database deve contenere l’utente con username “Roberto” e password “password” | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TEST ID** | 4 | | | |
| **TEST NAME** | Test Aggiunta di un Evento | | | |
| **TEST** **DESCRIPTION** | Verifica della funzionalità di aggiungere un evento | | | |
| **INPUT** | **RISULTATO DESIDERATO** | | **RISULTATO OTTENUTO** | |
| Inserire i valori desiderati in tutti I campi nella finestra di aggiunta di un evento | | Il Sistema ritorna alla schermata di gestione eventi mostrando anche l’evento appena creato | |  |
| Non inserire tutti I valori nei campi della finestra di aggiunta di un evento | | Viene mostrato un alert che invita l’utente a compilare tutti I campi rimanendo nella schermata di aggiunta di un evento | |  |
| **NOTE** |  | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TEST ID** | 5 | | | |
| **TEST NAME** | Test Modifica Evento | | | |
| **TEST** **DESCRIPTION** | Verifica della funzionalità di modifica di un evento | | | |
| **INPUT** | **RISULTATO DESIDERATO** | | **RISULTATO OTTENUTO** | |
| Modificare tutti I campi presenti nella finestra | | Il Sistema ritorna alla schermata di gestione evento mostrando l’evento appena modificato con I relativi dati aggiornati | |  |
| Modificare solo alcuni campi presenti nella finestra | | Il Sistema ritorna alla schermata di gestione evento mostrando l’evento appena modificato con I relativi dati aggiornati | |  |
| Non modificare niente | | Il Sistema ritorna alla schermata di gestione evento mostrando tutti gli eventi senza modifiche | |  |
| **NOTE** |  | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TEST ID** | 6 | | | |
| **TEST NAME** | Test Rimozione Evento | | | |
| **TEST** **DESCRIPTION** | Verifica della funzionalità di rimozione di un evento | | | |
| **INPUT** | **RISULTATO DESIDERATO** | | **RISULTATO OTTENUTO** | |
| Cliccare “OK” sull’alert che compare per la notifica della rimozione | | Il Sistema ritorna alla schermata di gestione evento mostrando tutti gli eventi tranne quello appena rimosso | |  |
| Cliccare “Annulla” sull’alert che compare per la notifica della rimozione | | Il Sistema ritorna alla schermata di gestione eventogli eventi | |  |
| **NOTE** |  | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TEST ID** | 7 | | | |
| **TEST NAME** | Test Rimozione Evento | | | |
| **TEST** **DESCRIPTION** | Verifica della funzionalità di rimozione di un evento | | | |
| **INPUT** | **RISULTATO DESIDERATO** | | **RISULTATO OTTENUTO** | |
| Click sul bottone delle statistiche relative all’evento “19466” | | Il Sistema mostra la schermata delle statistiche con I relative dati mostrati sia come testo sia come grafici | |  |
| Click sul bottone delle statistiche relative all’evento “10101” | | Il Sistema mostra la schermata delle statistiche mostrando dati e grafici vuoti | |  |
| **NOTE** | L’evento 19466 presenta svariati bliglietti associati nel database  L’evento 10101 non presenta biglietti associate nel database | | | |

# Unit testing

Per il test automatico effettuato con JUnit si sono scelti i metodi *checkTicket* e *checkEvent* della classe *ViewEventActivity* presente nella componente android del progetto.

Entrambi i metodi svolgono funzioni analoghe, cioè verificare se l’oggetto passato, rispettivamente Ticket nel primo caso e Event nel secondo, sia nullo, vuoto oppure regolarmente compilato.

Le classi di equivalenza per questi metodi sono le seguenti:

* L’oggetto passato è nullo
* L’oggetto passato ha i campi impostati a *null*
* L’oggetto passato è correttamente compilato

Avremo quindi 3 classi di equivalenza analoghe per ogni metodo.

I metodi sono di tipo booleano e restituiranno *false* nei primi due casi oppure *true* nell’ultimo.

Codice JUnit per il test dei metodi:

