Università degli studi di Salerno  
Corso di Ingegneria del Software

**RooManageR**

Requirement Analysis Document

Versione 1.1



05/11/2015

Coordinatore del progetto:

|  |  |
| --- | --- |
| **Nome** | **Matricola** |
| Andrea De Lucia |  |

Partecipanti:

|  |  |
| --- | --- |
| **Nome** | **Matricola** |
| Gargiulo Emanuele | 0512102244 |
| Izzo Giandomenico | 0512102292 |
| Malinconico Francesco | 0512103309 |

|  |
| --- |
| Scritto da: Gargiulo Emanuele, Izzo Giandomenico e Malinconico Francesco |

Sommario

[Introduzione 4](#_Toc434501513)

[Scopo del sistema 4](#_Toc434501514)

[Ambito del sistema 4](#_Toc434501515)

[Obbiettivi e criteri di accettazione del progetto 4](#_Toc434501516)

[Situazione Corrente 5](#_Toc434501517)

[Il nostro sistema 5](#_Toc434501518)

[Requisiti funzionali 5](#_Toc434501519)

[Requisiti non funzionali 5](#_Toc434501520)

[Scenari d’utilizzo 6](#_Toc434501521)

[Use Cases 7](#_Toc434501522)

[Login 7](#_Toc434501523)

[Nuova visita 8](#_Toc434501524)

[Registrazione stanza 9](#_Toc434501525)

[Conclusione visita 10](#_Toc434501526)

[Conclusione permanenza 11](#_Toc434501527)

[Modifica anagrafica 12](#_Toc434501528)

[Rendi agibile/inagibile una stanza 12](#_Toc434501529)

[Rendi agibile/inagibile una struttura 13](#_Toc434501530)

[Nuova struttura 13](#_Toc434501531)

[Nuova Stanza 14](#_Toc434501532)

[Rimuovi Stanza 15](#_Toc434501533)

[Rimuovi Struttura 16](#_Toc434501534)

# Introduzione

## Scopo del sistema

Il nostro software si propone come unica soluzione al problema della gestione di una struttura che offre un servizio di tipo ricettivo. Attualmente non esiste un’unica suite software in grado di gestire tutte le problematiche derivanti da ciò.

## Ambito del sistema

**RooManageR** è rivolto a tutte le persone coinvolte nell’amministrazione di una struttura:

* **Proprietario**
* **Dirigente**
* **Portinaio**

## Obbiettivi e criteri di accettazione del progetto

I principali obbiettivi del nostro software sono:

* Riunire, sotto un unico tetto, tutta la gestione della struttura
* Avere una curva di apprendimento molto bassa
* Essere multi-piattaforma
* Avere un avvio in meno di 5 secondi

I criteri di accettazione del progetto sono:

* Superare una fase di beta-testing della durata di 30gg con un bacino d’utenza di almeno 500 persone.

# Situazione Corrente

Le soluzioni software attualmente sul mercato risultano essere poco intuitive e molto frammentate e presentano frequenti crash ed errori di sistema dovuti alle eccessive interazioni tra questi.

In questo si differenzia **RooManageR** in quanto punta ad essere immediato, rapido e affidabile garantendo all’utente finale un incremento della sua produttività.

# Il nostro sistema

## Requisiti funzionali

Il **Proprietario** deve poter:

* Creare le proprie strutture
* Creare le stanze locate all’interno delle proprie strutture
* Gestire le anagrafiche
* Creare e gestire le visite all’interno delle proprie strutture
* Gestire gli inquilini all’interno delle proprie strutture
* Poter nominare un’anagrafica come dirigente di una o più delle sue struttura
* Poter nominare un’anagrafica come portinaio per una o più delle sue strutture

Il **Dirigente** deve poter:

* Gestire le anagrafiche
* Creare e gestire le visite all’interno delle proprie strutture
* Gestire gli inquilini all’interno delle proprie strutture
* Poter nominare un’anagrafica come portinaio per una o più delle sue strutture

Il **Portinaio** deve poter:

* Creare un’anagrafica
* Creare e gestire le visite all’interno della propria struttura
* Gestire gli inquilini all’interno della propria struttura

In generale:

* Deve avere un tempo di avvio minore di 5 secondi
* Deve essere multipiattaforma

## Requisiti non funzionali

* Il tempo di risposta del server deve essere minore di 2 secondi
* Deve poter garantire un “up-time” del 98% annuo
* L’applicazione non deve avere “freeze” dell’interfaccia
* L’applicazione non deve avere alcun “**ANR**” (**A**pplication **N**ot **R**esponding)
* L’applicazione deve essere eseguibile dalla JDK 1.7 in poi e dalla versione 4.0 di Android
* L’applicazione Android deve seguire i canoni dell’interfaccia “**Material Design**” mentre l’applicazione desktop dovrà ereditare lo stile grafico del sistema operativo su cui è in esecuzione.

# Scenari d’utilizzo

|  |  |
| --- | --- |
| **Nome dello scenario** | Aggiunta di una visita |
| **Attori interessati** | Nicola:**Portinaio**, Giuseppe**:Proprietario**, Gianmarco:**Visitatore** |
| **Flusso degli eventi** | 1. Nicola è un portinaio presso la struttura di **Giuseppe** 2. Gianmarco arriva alla struttura, e deve essere registrato 3. Nicola avvia il software, inserisce le proprie credenziali (codice fiscale e password) e viene portato alla schermata di **RooManageR** ideata per i portinai. 4. Nicola ha a disposizione un elenco delle camere e le relative informazioni 5. Nicola seleziona una camera singola libera e inserisce le informazioni relative a Gianmarco e alla sua registrazione, inclusi naturalmente i dati anagrafici e la data di ingresso e di uscita dalla struttura. 6. Nel caso in cui tutto vada a buon fine, i dati vengono aggiunti al sistema e la visita viene registrata. 7. A visita terminata, Gianmarco si reca da Nicola, che registra l’avvenuta fine della visita. |

|  |  |
| --- | --- |
| **Nome dello scenario** | Aggiunta di una struttura |
| **Attori interessati** | Giuseppe**:Proprietario** |
| **Flusso degli eventi** | 1. Giuseppe ha acquistato una nuova struttura e vuole aggiungerla a **RooManageR** 2. Avvia il software, inserisce le sue credenziali e viene portato alla schermata ideata per i proprietari. 3. Clicca nel menu in alto e seleziona la voce per creare una nuova struttura 4. Inserisce ora i dati relativi alla nuova struttura 5. Al successo dell’operazione viene visualizzato un messaggio di conferma. |

|  |  |
| --- | --- |
| **Nome dello scenario** | Ricezione di una lamentela |
| **Attori interessati** | Giandomenico:**Dirigente**, Gianmarco:**Visitatore** |
| **Flusso degli eventi** | 1. Giandomenico riceve una lamentela riguardo il comportamento di Gianmarco 2. Giandomenico verifica l’esattezza di tale segnalazione e vuole prendere provvedimenti al riguardo 3. Giandomenico apre l’applicazione, inserisce le proprie credenziali, seleziona la visita di Gianmarco e la termina in anticipo. |

# Use Cases

## Login

|  |  |
| --- | --- |
| **Nome dello use case** | Login |
| **ID dello use case** | UC-1 |
| **UML** |  |
| **Flusso di eventi** | In questo use case l’utente vuole effettuare il login su **RooManageR.**  L’utente deve inserire il suo codice fiscale e la sua password.  Il sistema può genera due eccezioni:   1. Il server risulta irraggiungibile. 2. I dati inseriti dall’utente non sono validi.   In entrambi i casi, si riporta l’utente alla schermata di Login, altrimenti lo si porta alla schermata dedicata alla tipologia di utente. |
| **Entry Condition** | L’utente si trova nella schermata di autenticazione |
| **Exit Condition** | Il programma visualizzerà la home relativa all’utente loggato |

## Nuova visita

|  |  |
| --- | --- |
| **Nome dello use case** | Inserimento di una visita |
| **ID dello use case** | UC-2 |
| **UML** |  |
| **Flusso di eventi** | In questo use case l’utente vuole inserire una nuova visita.  Per fare ciò segue il seguente protocollo:   1. Clicca sul bottone “Inserisci visita”. 2. Sceglie la stanza a cui assegnare la visita. 3. Se la stanza risulta essere vuota, il sistema genera un’eccezione “Stanza Vuota”, mostra un messaggio di errore e torna alla schermata precedente. 4. Se la stanza ha già raggiunto il massimo di visitatori, il sistema genera un’eccezione “Numero Massimo di Visitatori”, mostra un messaggio di errore e torna alla schermata precedente. 5. L’utente inserisce il codice fiscale del visitatore. Se il sistema lo riconosce, importa i dati correlati e popola i campi con i rispettivi valori, altrimenti l’utente inserisce manualmente i campi richiesti (codice fiscale, nome, cognome, indirizzo, telefono, cellulare, email, data di nascita, tipo di documento, numero di documento e la nazionalità) e clicca sul bottone “Conferma”. 6. Se l’operazione è andata a buon fine, l’applicazione visualizza a schermo un messaggio di notifica e torna alla Home Page. |
| **Entry Condition** | L’utente deve essere loggato al sistema, deve trovarsi sulla schermata per aggiungere una visita e deve avere i permessi necessari per effettuare l’operazione |
| **Exit Condition** | La visita viene inserita correttamente nel database e l’utente viene riportato alla schermata principale del programma |

## Registrazione stanza

|  |  |
| --- | --- |
| **Nome dello use case** | Registrazione di un nuovo ospite |
| **ID dello use case** | UC-3 |
| **UML** |  |
| **Flusso di eventi** | In questo use case l’utente vuole registrare un ospite e, per farlo, segue il seguente protocollo:   1. Clicca sul bottone “Registra ospite”. 2. Sceglie la stanza da assegnare all’ospite. 3. Se la stanza risulta essere occupata, il sistema genera un’eccezione “Stanza Occupata”, mostra un messaggio di errore e torna alla schermata precedente. 4. L’utente inserisce il codice fiscale dell’ospite. Se il sistema lo riconosce, importa i dati correlati e popola i campi con i rispettivi valori, altrimenti l’utente inserisce manualmente i campi richiesti (codice fiscale, nome, cognome, indirizzo, telefono, cellulare, email, data di nascita, tipo di documento, numero di documento e la nazionalità) e clicca sul bottone “Conferma”. 5. Se l’operazione è andata a buon fine, l’applicazione visualizza a schermo un messaggio di notifica e torna alla Home Page. |
| **Entry Condition** | L’utente deve essere loggato al sistema, deve trovarsi sulla schermata per registrare un ospite e deve avere i permessi necessari per effettuare l’operazione |
| **Exit Condition** | L’ospite viene inserito correttamente nel database e l’utente viene riportato alla schermata principale del programma |

## Conclusione visita

|  |  |
| --- | --- |
| **Nome dello use case** | Conclusione di una visita |
| **ID dello use case** | UC-4 |
| **UML** |  |
| **Flusso di eventi** | In questo use case l’utente vuole inserire le informazioni della fine di una visita:   1. L’utente seleziona la visita dall’elenco. 2. L’utente clicca sul bottone “Fine Visita”. 3. Il sistema automaticamente salva la data e l’ora e cancella la voce dall’elenco delle visite in atto. 4. Il sistema segnala a schermo la fine dell’operazione e torna alla Home Page. |
| **Entry Condition** | L’utente deve essere loggato al sistema, deve trovarsi sulla schermata per concludere una visita e deve avere i permessi necessari per effettuare l’operazione |
| **Exit Condition** | La conclusione della visita viene inserita correttamente nel database e l’utente viene riportato alla schermata principale del programma |

## Conclusione permanenza

|  |  |
| --- | --- |
| **Nome dello use case** | Conclusione di una permanenza |
| **ID dello use case** | UC-5 |
| **UML** |  |
| **Flusso di eventi** | In questo use case l’utente vuole inserire le informazioni della fine di una permanenza:   1. L’utente seleziona la stanza occupata. 2. L’utente clicca sul bottone “Libera stanza”. 3. L’utente inserisce il prezzo pagato dall’ospite. 4. Il sistema automaticamente salva la data e l’ora, aggiorna l’elenco delle stanze, segnala a schermo la fine dell’operazione e torna alla Home Page. |
| **Entry Condition** | L’utente deve essere loggato al sistema, deve trovarsi sulla schermata per concludere una permanenza e deve avere i permessi necessari per effettuare l’operazione |
| **Exit Condition** | La conclusione della permanenza viene inserita correttamente nel database e l’utente viene riportato alla schermata principale del programma |

## Modifica anagrafica

|  |  |
| --- | --- |
| **Nome dello use case** | Modifica di un’anagrafica |
| **ID dello use case** | UC-6 |
| **UML** |  |
| **Flusso di eventi** | In questo use case l’utente vuole modificare le informazioni di una anagrafica:   1. L’utente clicca sul bottone “Modifica anagrafica”. 2. L’utente modifica il/i valore/i errato/i. 3. Il sistema salva le informazioni aggiornate, segnala a schermo la fine dell’operazione e torna alla Home Page. |
| **Entry Condition** | L’utente deve essere loggato al sistema, deve trovarsi sulla schermata per modificare un’anagrafica e deve avere i permessi necessari per effettuare l’operazione |
| **Exit Condition** | L’anagrafica modificata viene scritta correttamente nel database e l’utente viene riportato alla schermata principale del programma |

## Rendi agibile/inagibile una stanza

|  |  |
| --- | --- |
| **Nome dello use case** | Rendi agibile/inagibile una stanza |
| **ID dello use case** | UC-7 |
| **UML** |  |
| **Flusso di eventi** | In questo use case l’utente vuole rendere agibile/inagibile una stanza della struttura:   1. L’utente seleziona la stanza in questione. 2. L’utente clicca sul bottone “Rendi inagibile/Rendi agibile”. 3. Se la stanza è occupata, il sistema automaticamente indica una stanza disponibile dello stesso tipo per trasferire l’ospite. Se non ci sono altre stanze dello stesso tipo, il sistema lancia l’eccezione “No Altre Stanze”. 4. Il sistema segnala la fine dell’operazione e torna alla Home Page. |
| **Entry Condition** | L’utente deve essere loggato al sistema, deve trovarsi sulla schermata per modificare l’agibilità di una stanza e deve avere i permessi necessari per effettuare l’operazione |
| **Exit Condition** | Lo stato modificato della stanza viene registrato correttamente nel database e l’utente viene riportato alla schermata principale del programma |

## Rendi agibile/inagibile una struttura

|  |  |
| --- | --- |
| **Nome dello use case** | Rendi agibile/inagibile una struttura |
| **ID dello use case** | UC-8 |
| **UML** |  |
| **Flusso di eventi** | In questo use case l’utente vuole rendere agibile/inagibile una struttura   1. L’utente seleziona la struttura in questione. 2. L’utente clicca sul bottone “Rendi inagibile/Rendi agibile”. 3. Se ci sono camere occupate, verrà lanciata l’eccezione “Camere Occupate”, e verrà data la possibilità di rendere inagibili le stanze in questione (UC-7) 4. Il sistema segnala la fine dell’operazione e torna alla Home Page. |

## Nuova struttura

|  |  |
| --- | --- |
| **Nome dello use case** | Inserimento di una struttura |
| **ID dello use case** | UC-9 |
| **UML** |  |
| **Flusso di eventi** | In questo use case il proprietario vuole inserire una nuova struttura collegata a se nel sistema.  Per farlo, segue il seguente protocollo:   1. Preme il bottone “Inserisci Struttura” 2. Inserisce i dati della struttura (indirizzo e descrizione) 3. Preme il bottone “Conferma”, viene mostrato un messaggio di successo e l’utente viene portato alla Home Page. |
| **Entry Condition** | L’utente deve essere loggato al sistema, deve trovarsi sulla schermata per inserire una struttura e deve avere i permessi necessari per effettuare l’operazione |
| **Exit Condition** | La struttura viene inserita correttamente nel database e l’utente viene riportato alla schermata principale del programma |

## Nuova Stanza

|  |  |
| --- | --- |
| **Nome dello use case** | Inserimento di una stanza |
| **ID dello use case** | UC-10 |
| **UML** |  |
| **Flusso di eventi** | Questo use case fa riferimento allo stesso protocollo dello use case UC-8, con l’unica modifica che riguarda le stanze anziché le strutture. |
| **Entry Condition** | L’utente deve essere loggato al sistema, deve trovarsi sulla schermata per inserire una stanza e deve avere i permessi necessari per effettuare l’operazione |
| **Exit Condition** | La stanza viene inserita correttamente nel database e l’utente viene riportato alla schermata principale del programma |

## Rimuovi Stanza

|  |  |
| --- | --- |
| **Nome dello use case** | Rimozione di una stanza |
| **ID dello use case** | UC-11 |
| **UML** |  |
| **Flusso di eventi** | In questo use case si descrive il protocollo seguito dal proprietario di una struttura quando vuole eliminare una stanza.   1. Visualizza l’elenco delle stanze presenti 2. Seleziona la stanza che desidera eliminare e preme il bottone “Elimina Stanza”. 3. Se la stanza risulta essere occupata il sistema lancia un eccezione del tipo “Stanza Occupata”, mostra un messaggio di errore e ritorna alla schermata del punto 1, altrimenti elimina la stanza selezionata dall’archivio e ritorna alla schermata del punto 1. |
| **Entry Condition** | L’utente deve essere loggato al sistema, deve trovarsi sulla schermata per rimuovere una stanza e deve avere i permessi necessari per effettuare l’operazione |
| **Exit Condition** | La stanza viene rimossa correttamente dal database e l’utente viene riportato alla schermata principale del programma |

## Rimuovi Struttura

|  |  |
| --- | --- |
| **Nome dello use case** | Rimozione di una struttura |
| **ID dello use case** | UC-12 |
| **UML** |  |
| **Flusso di eventi** | In questo use case si descrive il protocollo seguito dal proprietario di una struttura quando vuole eliminare una sua struttura.   1. Visualizza l’elenco delle strutture presenti 2. Seleziona la struttura che desidera eliminare e preme il bottone “Elimina Struttura”. 3. Se nella struttura ci sono ancora delle camere, il sistema lancia un eccezione del tipo “Struttura Occupata”, mostra un messaggio di errore e ritorna alla schermata del punto 1, altrimenti elimina la struttura selezionata dall’archivio e ritorna alla schermata del punto 1. |
| **Entry Condition** | L’utente deve essere loggato al sistema, deve trovarsi sulla schermata per rimuovere una struttura e deve avere i permessi necessari per effettuare l’operazione |
| **Exit Condition** | La struttura viene rimossa correttamente dal database e l’utente viene riportato alla schermata principale del programma |