Università degli studi di Salerno  
Corso di Ingegneria del Software

**RooManageR**

System Design Document

Versione 1.1



09/04/2016

Coordinatore del progetto:

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

Partecipanti:

|  |  |
| --- | --- |
| **Nome** | **Matricola** |
| Gargiulo Emanuele | 0512102244 |
| Izzo Giandomenico | 0512102292 |

|  |
| --- |
| Scritto da: Gargiulo Emanuele, Izzo Giandomenico |

Sommario

[Introduzione 4](#_Toc447979945)

[Scopo del sistema 4](#_Toc447979946)

[Architettura Software Proposta 4](#_Toc447979947)

[Panoramica del sistema 4](#_Toc447979948)

[Il nostro sistema 4](#_Toc447979949)

[Le possibilità offerte dal nostro sistema alle varie tipologie di utenti 4](#_Toc447979950)

[Vincoli di Design 5](#_Toc447979951)

[Referenze del Progetto 5](#_Toc447979952)

[Architettura del sistema 5](#_Toc447979953)

[Hardware 5](#_Toc447979954)

[Decomposizione in sottosistemi 5](#_Toc447979955)

[Architettura del Database 6](#_Toc447979956)

[Sicurezza e controllo degli accessi 7](#_Toc447979957)

[Matrice di controllo degli Accessi 7](#_Toc447979958)

[Flusso di controllo globale 7](#_Toc447979959)

[Boundary Conditions 8](#_Toc447979960)

[Configuration 8](#_Toc447979961)

[Start up and Shutdown use cases 8](#_Toc447979962)

[Exception use cases 9](#_Toc447979963)

# Introduzione

Questo System Design Document è stato creato per delineare il sistema proposto atto a rivoluzionare il modo in cui vengono gestite le strutture.

## Scopo del sistema

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

# Architettura Software Proposta

## Panoramica del sistema

Chi gestisce una o più strutture come alberghi o dormitori universitari sa quanto sia complesso. RooManageR potrà funzionare con praticamente ogni computer in circolazione, senza distinzioni di sistema operativo e senza quindi richiedere spese extra per l’aggiornamento dei sistemi.   
Un database conterrà tutte le informazioni relative alle strutture, con le relative camere ed ospiti. RooManageR consentirà l’accesso rapido ed intuitivo al database, con la possibilità di aggiungere e modificare facilmente i dati in esso contenuti da un qualunque computer compatibile con Java.

# Il nostro sistema

## Le possibilità offerte dal nostro sistema alle varie tipologie di utenti

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
* Nominare un’anagrafica come portinaio per una 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

## Vincoli di Design

* RooManageR dev’essere compatibile con la maggioranza dei computer esistenti (purché supportino Java), non rendendo necessario l’acquisto di nuove macchine.
* RooManageR non deve pesare in modo eccessivo sulla rete, in modo da essere utilizzabile con qualunque tipo di connessione internet.

## Referenze del Progetto

* Documento di Analisi dei Requisiti

## Architettura del sistema

### Hardware

RooManageR girerà su qualunque tipo di computer connesso ad internet e con supporto a Java

### Decomposizione in sottosistemi

Il software è sviluppato in tre moduli principali:

* Un’ applicazione java che interagisce con l’utente attraverso un’interfaccia grafica semplice ed intuitiva e invia le informazioni inserite al server;
* Un’ applicazione server, implementata in PHP, che riceve le informazioni inviate dall’applicazione sovrastante e le salva sul database, oppure che preleva le informazioni dal database e le invia all’applicazione suddetta;
* Un database relazionale che contiene le informazioni necessarie al funzionamento del programma.

L’applicazione Java è divisa principalmente quattro sottosistemi:

* Il sottosistema web\_services si occupa di comunicare con il server PHP
* Il sottosistema User Interface che consente all’utente di visualizzare tramite interfaccia grafica le informazioni già presenti e di interagire con esse, e acquisisce le nuove informazioni inserite dall’utente
* Il sottosistema Utils, che raggruppa per classi necessarie a mantenere una copia in locale di alcuni dati
* Il sottosistema entities, che contiene le entità necessarie al funzionamento del sistema

## Architettura del Database

Il database contiene una tabella per ogni entità del programma.

Una tabella “Nazionalità” che serve come registro delle nazionalità delle varie anagrafiche che vengono registrate. L’elenco completo delle nazionalità deve essere inserito da database in quanto non sarà possibile aggiungere nuove nazionalità da RooManager.

Una tabella “Anagrafica” che contiene i seguenti campi: codice fiscale, nome, cognome, indirizzo, data di nascita, tipo e numero di documento, cellulare, telefono, email e nazionalità. La chiave primaria è il campo codice fiscale.

Una tabella “Struttura” che contiene i seguenti campi: nome, indirizzo, numero di stanze della struttura, descrizione, stato di agibilità e codice fiscale del proprietario. Ha una chiave esterna col campo codice fiscale della tabella Anagrafica.

Una tabella “Stanza” che contiene i seguenti campi: nome della struttura, codice fiscale proprietario della struttura, numero della stanza, tipo, descrizione, metri quadri della stanza, agibile, permanenza in corso, visita in corso.

Una tabella “AnagraficaMansione” che contiene i seguenti campi: codice fiscale dell’anagrafica, password, nome struttura in cui l’anagrafica svolge la mansione, codice fiscale proprietario della struttura, il tipo di mansione.

Una tabella “AnagraficaStanza” che contiene i campi: codice fiscale anagrafica assegnata alla stanza, numero della stanza, nome della struttura, codice fiscale proprietario della struttura, tipo, data ed ora d’ingresso, data ed ora d’uscita, costo.

## Sicurezza e controllo degli accessi

L’avvio del programma richiede un login in cui l’utente dovrà inserire il suo codice fiscale e la sua password.   
L’accesso alle funzionalità del programma è correlato al tipo d’utente:

* Il proprietario può accedere a tutte le funzioni del programma
* Il portinaio può accedere a un sottoinsieme delle funzioni del proprietario

### Matrice di controllo degli Accessi

#### Parte1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Utente** | **Gestione Permanenze** | **Gestione Visite** | **Gestione Strutture** | **Gestione Stanze** |
| **Proprietario** | Si | Si | Si | Si |
| **Portinaio** | SI | Si | No | No |

#### Parte2

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Utente** | **Visualizzazione Panoramica** | **Visualizzazione Registro Movimenti** | **Gestione Portinaio** | **Modifica Password** | **Gestione Anagrafiche** |
| **Proprietario** | Si | Si, per tutte le sue strutture | Si | Si | Si |
| **Portinaio** | Si | Solo per la struttura dove è impiegato | No | Si | Si |

## Flusso di controllo globale

RooManager ha un flusso guidato dagli eventi (event-driven): non esiste una sequenza di operazioni prestabilita, è l’utente a scegliere di volta in volta l’operazione da eseguire. L’architettura software comprende un web server che rimane in ascolto in attesa di una richiesta da parte dell’applicazione.   
Per evitare freeze dell’interfaccia grafica, quando sarà necessario inviare ed attendere risposte dal database sul server, saranno utilizzati dei thread.

## Boundary Conditions

All’avvio il programma presenta una schermata di login.   
In caso di errore (esempio: server offline), viene presentata una finestra di dialogo che avvisa l’utente dell’errore e il database resta inalterato.   
La chiusura del programma avviene normalmente chiudendo l’interfaccia tramite la X sulla barra superiore o tramite il bottone exit raggiungibile dalla barra del menu’.

## Configuration

Per ogni oggetto persistente dobbiamo esaminare in quale caso d’uso è creato o distrutto

* Anagrafica: Viene creata dall’utente durante l’aggiunta di una nuova visita o una nuova permanenza. Quando l’utente inserisce il codice fiscale, se esso non è già presente nel database potrà compilare i restanti campi e l’anagrafica verrà creata.
* AnagraficaMansione: L’anagraficaMansione per il portinaio viene creata dall’utente di tipo Proprietario quando aggiunge un portinaio a una delle sue strutture.
* AnagraficaStanza: Viene creata dall’utente quando da inizio a una nuova visita o permanenza. Ai fini del mantenimento del registro clienti, non viene mai eliminata neanche dopo la sua terminazione, ma viene semplicemente contrassegnata come “terminata”.
* Stanza: Viene creta dall’utente di tipologia Proprietario tramite la schermata Gestione Strutture.   
  Può essere eliminata dal proprietario dalla schermata Gestione Strutture se non ci sono visite o permanenze in corso nella stanza in questione.
* Struttura: Viene creata dall’utente di tipologia Proprietario tramite la schermata Gestione Strutture.  
  Può essere eliminata dal proprietario dalla schermata Gestione Strutture se sono state preventivamente eliminate tutte le stanze.

## Start up and Shutdown use cases

RooManager prevede tre componenti: Un’applicazione client java, un server e un database.

Il server e il database su esso presente sono componenti off-the-shelf, e sono avviati e spenti indipendentemente e sono pensati per essere in funzione h24, a meno di guasti tecnici

Il client Java RooManager invece verrà avviato dall’utente a sua discrezione sul proprio computer.

## Exception use cases

L’errore principale che può affliggere RooManager è relativo alla comunicazione con il server e con il database, che questo avvenga per problemi di connessione o per problemi tecnici del server.   
Tale errore verrà opportunamente segnalato dall’applicazione client, che manterrà in locale una copia dei dati interessati dall’ultima operazione ai fini di effettuare ulteriori tentativi e avviserà l’utente che l’operazione da lui richiesta non è andata a buon fine, evitando la possibilità di lavorare senza rendersi conto della perdita di dati.

|  |
| --- |
| Extension Use Case: Errore di connessione |
| ID: Errore di connessione |
| Il sistema non riesce a comunicare con il server e con il database Il sistema memorizza in locale i dati relativi all’ultima operazione richiesta  Il sistema avvisa l’utente del problema di connessione e chiede se vuole riprovare l’operazione o uscire dal programma |