



Summer Center

**Test Planning**

# Top manager

| **Professore** | **Tutor** |
| --- | --- |
| De Lucia Andrea | Pecorelli Fabiano |

# Partecipanti

| **Cognome** | **Nome** | **Matricola** |
| --- | --- | --- |
| **Dalia** | Antonio | 0512106118 |
| **Donnarumma** | Gerardo | 0512106064 |
| **Esposito** | Vincenzo | 0512106070 |

# Storico versioni

| **Data** | **Versione** | **Descrizione** | **Autori** |
| --- | --- | --- | --- |
| **03/02/2021** | 1.0 | Creazione del documento, introduzione | Dalia |
| **05/02/2021** | 1.1 | Completamento e impaginazione | Esposito |
| **06/02/2021** | 1.2 | Revisione del documento | Esposito |

Sommario

[Top manager 2](#_Toc63499426)

[Partecipanti 2](#_Toc63499427)

[Storico versioni 2](#_Toc63499428)

[Introduzione 4](#_Toc63499429)

[Introduzione 4](#_Toc63499430)

[Relazione con altri documenti 4](#_Toc63499431)

[Panoramica del sistema 4](#_Toc63499432)

[Definizioni, acronimi e abbreviazioni 4](#_Toc63499433)

[Funzionalità da Testare / Non testare 5](#_Toc63499434)

[**Funzionalità da testare** 5](#_Toc63499435)

[**Funzionalità da non testare** 5](#_Toc63499440)

[Criteri di successo /fallimento 6](#_Toc63499473)

[Approccio 6](#_Toc63499474)

[**Test di unità** 6](#_Toc63499475)

[**Test di integrazione** 6](#_Toc63499476)

[**Test di sistema** 6](#_Toc63499477)

# Introduzione

## Introduzione

Lo scopo di questa fase è quello di evidenziare difetti nel sistema prima che esso stesso sia rilasciato al pubblico.

L’estensione dei test dovrà coprire tutte le funzionalità ad alta priorità e dovrà seguire un approccio formale, in modo da individuare un numero sufficiente di input da testare per assicurare il corretto funzionamento del sistema e che i casi di test siano ripetibili.

Il documento quindi presenterà l’approccio, i criteri di successo ed il test plan.

## Relazione con altri documenti

* Requirement Analysis Document: necessario per determinare i requisiti funzionali e non funzionali da convalidare in fase di testing.
* System Design Document: viene definita l’architettura del sistema client-server multi-tier basato sul modello Model-Control-View e i servizi che vengono esposti all’utente.
* Object Design Document: vengono definite le intefacce delle componenti del sistema, assieme ai vincoli e i contratti da rispettare che saranno inclusi nella fase di testing.

## Panoramica del sistema

Il sistema segue lo stile architetturale MVC, dove i componenti Model gestiranno la logica e i dati del dominio applicativo, i Control invieranno messaggi ai Model e ricevono gli input dell’Utente che interagiscono con le pagine generate dalle View. Sarà importante dunque testare tutte e tre le tipologie di componenti in modo da assicurare il corretto funzionamento del sistema.

## Definizioni, acronimi e abbreviazioni

|  |  |
| --- | --- |
| Denominazione | Formato |
| Random Control | Le classi Control gestiscono le richieste HTTP e inviano le risposte al client |
| Random manager | Le classi Manager gestiscono la Business Logic e l’interazione con i file e il database. |
| Jsp | [JavaServer Pages](https://www.oracle.com/java/technologies/jspt.html) |

## Funzionalità da Testare / Non testare

Saranno testate le funzionalità che partecipano ai requisiti funzionali con priorità alta.

### **Funzionalità da testare**

### Gestione Autenticazione

### Login

### Gestione Iscrizione

### Iscrizione

### **Funzionalità da non testare**

### Gestione Registrazione

### Registrazione

### Verifica account

### Gestione Autenticazione

### Logout

### Gestione account

### Inserimento dati genitore

### Modifica dati genitore

### Visualizza dati personali

### Visualizza bambini

### Visualizza dettagli bambino

### Cambia password

### Nuova password (il Segretario che effettua il primo accesso cambia la propria password)

### Cambia ruolo Responsabile

### Cambia ruolo Segretario

### Registra segretario

### Visualizza andamento centro

### Gestione iscrizione:

### Selezione iscrizioni da pagare

### Modalità di pagamento

### Paypal

### Pagamento con carta

### Modifica iscrizione bambino

### Visualizza iscrizioni

### Elimina iscrizione bambino

### Visualizza dettagli iscrizione

### Disdetta iscrizione

### Approva disdetta

### Visualizza iscrizioni pagate

### Ricerca iscrizioni

### Visualizza dettagli iscrizione pagata

### Controllo prenotazione QR-code

## Criteri di successo /fallimento

L’esito dei test sarà determinato dalla loro equivalenza con l’oracolo: se il risultato ottenuto dal test è quello atteso, allora avrà avuto successo; in caso contrario sarà considerata come una failure, che, verrà individuata e corretta per poi ripetere il test.

## Approccio

Il testing sarà diviso in 3 fasi:

### **Test di unità**

Sarà adottata la tecnica di testing “black-box”. L’attenzione cadrà principalmente sul comportamento dell’ I/O senza badare alla struttura interna del codice. Se ogni test rispetterà l’output previsto dall’oracolo, il test avrà avuto esito positivo.

Non essendo possibile testare tutti i possibili output, utilizzeremo una partizione delle condizioni di input in classi di equivalenza con le quali saranno scelti i casi di test.

### **Test di integrazione**

----

### **Test di sistema**

Essendo un sistema web-based, per questa fase di test si utilizzerà Selenium. Saranno simulate le varie funzioni in questo caso relative all’iscrizione di un bambino ex-novo e il login.