Test Plan

1 Introduzione

1.1 Scopo del documento

Lo scopo del presente documento è quello di definire i casi di test (test case) per le funzionalità del sistema software. Ogni test case definisce una serie di input ed un output atteso, sulla base dei quali si potrà decretare la correttezza della funzionalità testata.

Overview del Sistema

Il sistema è stato diviso in diversi sottosistemi in fase di ristrutturazione e di conseguenza verranno testati:

- Sottosistema utente.
- Sottosistema gestione locale
- Sottosistema gestione dati

2 Funzionalità testate

Verranno testate le funzionalità di

- Prenotazione online
- Gestione di un'ordinazione
- Gestione di un tavolo
- Gestione di un utente

3 Criteri Pass/Failed

All'esecuzione di un metodo verranno testati se in base ad un determinato input venga restituito l'output desiderato. Ovviamente non sarà possibile effettuare test su ogni singolo input in quanto essi sono infiniti.

4 Approccio

Durante questa fase verranno ricercate condizioni di fallimento isolando le classi dal sistema integrando ed test driver/stub (implementazioni parziali di componenti che dipendono o da cui dipendono le componenti da testare).
Verrà utilizzato per effettuare i test la tecnica Black-box che si basa non sulla variante del testing tramite interfaccia utente ma tramite comunicazione tramite interfacce.
Utilizzeremo il framework PHPUnit per effettuare i test, integrato nella suite PHPStorm utilizzata per sviluppare il sistema.

5 Sospensione

Il test avrà fine quando tutte le classi saranno state testate e per ogni input il risultato sarà quello atteso.
In caso di test negativo il test inizierà da capo onde evitare che eventuali modifiche siano incompatibili con classi che precedentemente avevano superato il test.

6 Materiale per il testing

Per eseguire il test è necessario un pc/mac con il software PHPStorm installato.
All'interno della suite PHPStorm, ricordiamo, è presente il framework PHPUnit.

7 Classi di equivalenza

Si sono individuate varie classi di equivalenza per ogni input che possa essere immesso per l'utilizzo di una o più componenti. Di seguito sono elencate le classi di equivalenza che saranno prese in considerazione durante i successivi documenti di Testing per sviluppare i test case.

- Classe Utente

#CE	**Input**	**Condizione valida**	**#CE**	**Condizione non valida**
CE01	Cliente (anagrafica)	Nome, password, email, tipologia, cognome $\neq$ ‘ ’	CE02	Nome, password, email, tipologia, cognome = ‘ ’

- Datasource

#CE	**Input**	**Condizione valida**	**#CE**	**Condizione non valida**
CE01	Connessione mysql	Conn = mysqlConnection	CE02	Conn = ‘ ’ o connessione mysql non valida

- Ordinazione

#CE	Input	Condizione valida	#CE	Condizione non valida
CE01	Prezzo Totale	PrezzoTotale >= 0	CE02	PrezzoTotale < 0
CE03	Tavolo	Instanceof(Tavolo) sia di tipo Tavolo	CE04	Instanceof(Tavolo) restituisce false
CE05	UserSala	Instanceof(UserSala) restitusce true	CE06	Instanceof(UserSala) restitusce false

- Portata

#CE	Input	Condizione valida	#CE	Condizione non valida
CE01	Descrizione	Descrizione ≠''	CE02	Descrizione = ''
CE03	Prezzo	Prezzo > 0	CE04	Prezzo < 0
CE05	Categoria	Categoria = (Primo || Secondo || Frutta || Contorno || Dolci || Bevande || Sfizi || Pizze)	CE06	Categoria ≠ (Primo && Secondo && Frutta && Contorno && Dolci && Bevande && Sfizi && Pizze)

- Prenotazioni

#CE	Input	Condizione valida	#CE	Condizione non valida
CE01	Utente	Utente ≠"" ed Instanceof(Cliente)	CE02	Utente = ''
CE03	Tavolo	Instanceof(Tavolo)	CE04	Tavolo = ''
CE05	Data	Data >= DataOdierna	CE05	Data < DataOdierna

- Tavolo

#CE	Input	Condizione valida	#CE	Condizione non valida
CE01	numTavolo	numTavolo ≠''	CE02	numTavolo non valida o gia' assegnato
CE03	numPosti	numPosti > 0	CE04	numPosti < 0