ITPD – Integration Test Plan Document

Versione	1.1
Data	22/01/2022
Destinatario	Prof.ssa F. Ferrucci, Prof. Fabio Palomba
Presentato da	Tutti i team member
Approvato da	Salvatore Amideo Alice Vidoni

Revision History

Data	Versione	Descrizione	Autori
18/01/2022	0.1	Prima stesura	tutti
21/01/2022	1.0	Aggiunta del Glossario	tutti
22/01/2022	1.1	Correzione formattazione documento	tutti

Sommario

1. Introduzione

Il testing di integrazione rappresenta una delle fasi di testing più importanti, in quanto consiste nella verifica delle interazioni tra due o più componenti. L'obiettivo del testing è la verifica della corretta interazione tra le componenti nonché il rispetto delle interfacce, secondo quanto stabilito nelle specifiche di Integrazione. Questo documento ha il compito di identificare la strategia di testing di integrazione per il sistema UnisaEat(UE).

1.1 Definizioni, Acronimi e Abbreviazioni

1.1.1 Definizioni

- **Branch Converage**: Misura utilizzata per descrivere il grado di copertura del codice sorgente di un programma quando viene eseguita una particolare suite di test;
- **Bottom-up**: Parti individuali del sistema sono specificate in dettaglio, e poi connesse tra loro in modo da formare componenti più grandi, a loro volta interconnesse fino a realizzare un sistema completo;
- **Testing di Integrazione**: Fase di test del software in cui i singoli moduli software vengono combinati e testati come gruppo;
- **Category Partition**: Una tecnica di test Black Box che formalizza le specifiche del dominio di input del sistema in prova. Una specifica CP è guidata dall'esperienza del tester e comprende parametri, categorie (caratteristiche dei parametri) e scelte (valori accettabili per le categorie) richiesti;
- **Test Case Specification**: descrive lo scopo di uno specifico test, identifica gli input richiesti e i risultati previsti, fornisce procedure dettagliate per l'esecuzione del test e delinea i criteri di pass/fail per determinare l'accettazione;
- **Test Plan**: documento che dettaglia gli obiettivi, le risorse e i processi per un test specifico;
- **System Design Document**: descrive il sistema a livello di architettura, inclusi i sottosistemi e i loro servizi, la mappatura hardware, la gestione dei dati, il controllo degli accessi, la struttura di controllo del software globale e le boundary condition;

1.1.2 Acronimi e Abbreviazioni

- **UE_ITPD_Vers1.1**: Acronimo utilizzato per indicare il documento Integration Test Plan;
- **UE**: Acronimo utilizzato per indicare il progetto UnisaEAT;
- **TP**: Acronimo utilizzato per indicare il Test Plan;
- **TCS**: Acronimo utilizzato per indicare il Test Case Specification;
- **CP**: Acronimo utilizzato per indicare la Category Partition;
- **SDD**: Acronimo utilizzato per indicare il System Design Document;

1.2 Riferimenti

- Kathy Schwalbe, "Information Technology Project Management", International Edition 7E, Cengage Learning, 2014;
- Bernd Bruegge, Allen H. Dutoit, "Object-Oriented Software Engineering Using UML, Patterns and Java", Third Ed., Pearson, 2010; Sommerville, "Software Engineering", Addison Wesley;
- PMBOK® Guide and Software Extention to the PMBOK® Guide, Fifth Ed., Project Management Institute, 2013;
- Documentazione di Progetto:
  - UE_TP_Vers1.1;
  - UE_TCS_Vers1.1;
  - UE_CP_Vers1.1;
  - UE_SDD_Vers1.1;

2. Test di Integrazione

2.1 Approccio di Integrazione Testing

È stata scelta la strategia bottom-up, che permette di poter iniziare l'attività di testing non appena il primo modulo è stato specificato. Per eseguire il testing di integrazione sarà utilizzato Mocha il quale sfrutta il tool NYC per il continuous integration.

2.1.1 Componenti da testare

Per quanto riguarda il layer Model, le componenti da testare sono:

- Cliente
- Admin
- Conversazione
- Messaggio
- Faq
- Menu
- Personale
- Tesserino
- Ticket
- Notifica
- Statistica
- Ordine
- Pasto

Per quanto riguarda il layer Controller, le componenti da testare sono:

- ControllerConversazione
- ControllerFaq
- ControllerLogin
- ControllerMessaggio
- ControllerPersonale
- ControllerProfilo
- ControllerTesserino

- ControllerTicket
- ControllerOrdine
- ControllerStatistiche
- ControllerAdmin
- ControllerMenu

Per quanto riguarda il layer View, le componenti da testare sono:

GENERIC:

- Login.js
- Logout.js
- Homepage.js

CLIENTE:

- Chat. .js
- VisualizzaSaldo.js
- VisualizzazioneFAQ.js
- RicaricaTesserino.js
- RinnovoTesserino.js
- RichiestaTesserino.js
- CompilazioneTicket.js
- VisualizzazioneMenu.js
- VisualizzazioneNotifiche.js
- RimozioneNotifiche.js
- DettagliOrdine.js
- ListaOrdini.js
- PagamentoPasto.js
- SceltaPasti.js
- Profilo.js

PERSONALE ADISU

- Chat.js
- VisualizzazioneFAQ.js
- InserimentoFAQ.js

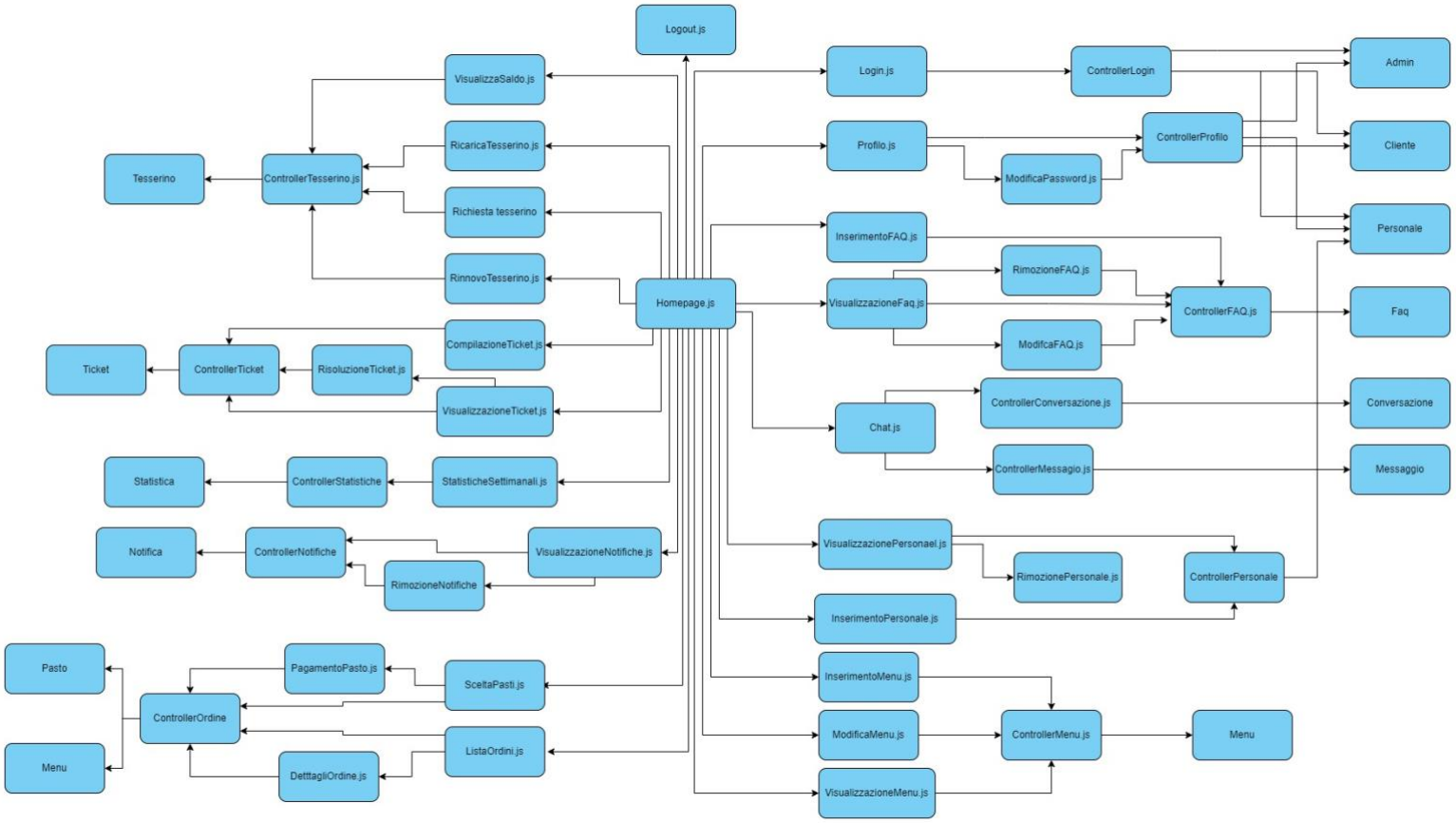
- ModificaFAQ.js
- InserimentoPersonale.js
- VisualizzazionePersonale
- VisualizzazioneNotifiche
- RimozioneNotifiche
- RimozionePersonale.js
- ModificaPassword.js
- CompilazioneTicket.js
- Profilo.js

OPERATORE MENSA

- InserimentoMenu.js
- VisualizzazioneMenu.js
- ModificaMenu.js
- ModificaPassword.js
- CompilazioneTicket.js
- Profilo.js
- StatisticheSettimanali.js

ADMIN:

- InserimentoPersonale.js
- RimozionePersonale.js
- VisualizzazionePersonale.js
- ModificaPassword.js
- RisoluzioneTicket.js
- VisualizzazioneTicket.js
- Profilo.js



3. Pass/Fail criteria

Il testing avrà SUCCESSO se:

- risulterà una Branch coverage maggiore o uguale al 75%;
- la build di NYC avrà successo;

Avremo un INSUCCESSO se:

- risulterà una Branch coverage minore del 75%;
- fallisce anche un solo dei casi di test, invalidando la build di NYC;

4. Glossario

- **Build:** Versione precedente in merito al rilascio sul mercato, identificata da un numero;
- **Chat:** conversazione fra più interlocutori costituita da uno scambio di messaggi scritti che appaiono in tempo reale nel monitori di ciascun partecipante;
- **Componente:** un blocco di programma riutilizzabile anche in combinazione con altre

componenti;

- **Criterio:** Norma su cui si fondano le distinzioni, i giudizi, le diverse linee d'azione o di condotta;
- **Errore:** Una misura della differenza stimata tra il valore osservato o calcolato e il suo valore reale;
- **Fail:** Fallimento in una o più fasi di test;
- **Insuccesso:** Esito negativo;
- **Pass:** Successo in una o più fasi di test;
- **Software:** Insieme delle procedure e delle istruzioni in un sistema di elaborazione dati;
- **Specifica:** Nota distinta in cui sono specificati uno per uno elementi utili a un determinato fine;
- **Successo:** Esito positivo;
- **Test:** Esperimento espletato allo scopo di valutare, mediante determinate reazioni, la correttezza di un software;
- **Verifica:** Operazione di controllo per mezzo della quale si procede all'accertamento di determinati risultati.