BARBER APP

Test Plan (TP)

Autori
Francesco Pio Guerriero matr. 0512114790
Luca Vincenzo Lambiase Fontana matr. 0512113338
Ivan Prota matr. 0512106063

Sommario

INTRODUZIONE

Il presente documento è stato redatto al fine di fornire una panoramica completa delle attività di testing svolte nel contesto del progetto BarberApp. Il testing svolge un ruolo fondamentale nel garantire la qualità e l'affidabilità del software, contribuendo a identificare e risolvere eventuali difetti prima del rilascio in produzione.

In questa documentazione, verranno descritte le strategie di testing adottate e i risultati ottenuti attraverso le diverse fasi di testing. Attraverso un approccio metodico e sistematico, ci siamo impegnati a garantire che il software soddisfi i requisiti funzionali e non funzionali concordati, assicurando al contempo un'esperienza utente ottimale.

RIFERIMENTI AD ALTRI DOCUMENTI

PANORAMICA DEL SISTEMA

In questo documento verranno eseguiti test di unità per ogni classe che si occupa della logica di business, ad eccezione dei metodi getter, setter e costruttori. Verranno quindi prese in considerazioni le seguenti classi: AppuntamentoController, AppuntamentoDAO, AppuntamentoRepository, ClientsController, ClienteDAO, ClienteRepository, DipendenteController, DipendenteDAO, DipendenteRepository, ServizioController, ServizioDAO, ServizioRepository, TitolareController, TitolareDAO, TitolareRepository. In seguito, verrà effettuato il test di integrazione e il test di sistema.

Tenendo conto della stratificazione del sistema, verranno testate prima le classi Repository, poi le classi DAO ed infine le classi Controller.

Dato l'utilizzo del framework Springboot abbiamo deciso di non testare le funzionalità auto-implementate.

Ogni tipo di controllo dei parametri è stato eseguito all'interno delle pagine dell'interfaccia grafica, motivo per cui verranno testate all'interno del testing di sistema.

FUNZIONALITA' DA TESTARE

Saranno soggette a test le seguenti funzionalità:

Funzionalità Appuntamento:

1. Funzionalità AppuntamentoRepository
    - countAppointmentsByCliente_IdAndDate
    - findByClienteIdOrderById
    - findByDipendenteId
    - findAllByOrderByDate
2. Funzionalità AppuntamentoDAO
    - checkAppuntamento
    - getAppuntamentiByCliente
    - getAppuntamentiByDipendente
    - getAppuntamentiOrdered
3. Funzionalità AppuntamentoController
    - saveAppuntamento
    - getAppointment
    - getAppointmentByDipendente
    - getAllAppuntamentiOrdered

Funzionalità Cliente:

1. ClienteRepository
    - getClienteByEmail
    - getClienteByEmailAndPassword
2. ClienteDAO
    - checkCliente
    - loginCliente
    3. ClienteController
    - save

- check
- update
- login

Funzionalità Dipendente:

1. DipendenteRepository
   - findByEmail
   - findAvailableEmployeesByAppuntamentiDateAndAppuntamentiTime
   - getDipendenteByEmailAndPassword
2. DipendenteDAO
   - checkDipendente
   - getEmployeeByDate
   - loginDipendente
3. DipendenteController
   - check
   - saveEmployee
   - update
   - getDipendentiByDate
   - login

Funzionalità Servizio:

1. ServizioController
   - updateServizio

Funzionalità Titolare:

1. TitolareRepository
   - getTitolareByEmail
   - getTitolareByEmailAndPassword
2. TitolareDAO

- checkTitolare
- loginTitolare

3. TitolareController
   - saveTitolare
   - check
   - update
   - login

CRITERIO PASSED/FAILED

Un test viene considerato "Passed" quando non si sono riscontrate failure durante l'esecuzione, mentre viene considerato "Failed" quando sono state riscontrare una o più failure.

APPROCCIO

L'approccio utilizzato per il testing è il seguente:

1. **Test di unità.**

   Per il test di unità verrà utilizzata la tecnica di "Black-Box testing". Ci concentreremo, quindi, sul comportamento di I/O, senza considerare la struttura interna dell'unità. Verranno generate classi di equivalenza da cui verranno scelti i test case.

2. **Test di integrazione.**

   Per il test di integrazione verrà utilizzata la strategia "bottom-up". Verranno, quindi, testanti i sottosistemi che si trovano più in basso nella gerarchia (test di unità) per poi arrivare a testare i sottosistemi che utilizzano quelli testati in precedenza. Si ripete quest'ultimo passo finché non verranno integrati tutti i sottosistemi.

3. **Test di sistema**

STRUMENTI UTILIZZATI PER IL TESTING

- JUnit
- Mockito
- IntelliJIDEA

TEST CASES

1. AppuntamentoRepository

1.1 countAppointmentsByCliente_IdAndDate

TC 1.1.1	PASSED
TC 1.1.2	PASSED

1.1 findByClienteIdOrderById

TC 1.2.1	PASSED
TC 1.2.2	PASSED

1.2 findByDipendenteid

TC 1.3.1	PASSED
TC 1.3.2	PASSED

1.3 findAllByOrderByDate

TC 1.4.1	PASSED
TC 1.4.2	PASSED

2. ClienteRepository

2.1 getClienteByEmail

TC 2.1.1	PASSED
TC 2.1.2	PASSED

2.2 getClienteByEmailAndPassword

TC 2.2.1	PASSED
TC 2.2.2	PASSED

3. DipendenteRepository

3.1 findByEmail

TC 3.1.1	PASSED
TC 3.1.2	PASSED

3.2 findAvaiableEmployeesByAppuntamentiDateAndAppuntamentiTime

TC 3.2.1	PASSED
TC 3.2.2	PASSED

3.3 getDipendenteByEmailAndPassword

TC 3.3.1	PASSED
TC 3.3.2	PASSED

4. TitolareRepository

4.1 getTitolareByEmail

TC 4.1.1	PASSED
TC 4.1.2	PASSED

4.2 getTitolareByEmailAndPassword

TC 4.1.1	PASSED
TC 4.1.2	PASSED

5. AppuntamentoDAO

5.1 checkAppuntamento

Parametro	Appuntamento
Presenza [P]	Presente Non presente

Identificativo	**Tipologia**	**Esito**
TC 5.1.1	P.1	PASSED
TC 5.1.2	P.2	PASSED

5.2 getAppuntamentiByCliente

TC 5.2.1	PASSED
TC 5.2.2	PASSED

5.3 getAppuntamentiByDipendente

TC 5.3.1	PASSED
TC 5.3.2	PASSED

5.4 getAppuntamentiOrdered

TC 5.4.1	PASSED
TC 5.4.2	PASSED

6. ClienteDAO

6.1 checkCliente

Parametro	email
Presenza [P]	Presente Non presente

Identificativo	**Tipologia**	**Esito**
TC 6.1.1	P.1	PASSED
TC 6.1.2	P.2	PASSED

6.2 loginCliente

TC 6.2.1	PASSED

TC 6.2.2	PASSED

7. DipendenteDAO

7.1 checkDipendente

Parametro	email
Presenza [P]	Presente Non presente

Identificativo	**Tipologia**	**Esito**
TC 7.1.1	P.1	PASSED
TC 7.1.2	P.2	PASSED

7.2 getEmployeeByDate

TC 7.2.1	PASSED
TC 7.2.2	PASSED

7.3 loginDipendente

TC 7.3.1	PASSED
TC 7.3.2	PASSED

8. TitolareDAO

8.1 checkTitolare

Parametro	email
Presenza [P]	Presente Non presente

Identificativo	Tipologia	Esito
TC 8.1.1	P.1	PASSED
TC 8.1.2	P.2	PASSED

8.2 loginTitolare

TC 8.2.1	PASSED
TC 8.2.2	PASSED

9. AppuntamentoController

9.1 saveAppuntamento

Parametro	Appuntamento
Presenza [P]	Presente Non presente

Identificativo	Tipologia	Esito
TC 9.1.1	P.1	PASSED
TC 9.1.2	P.2	PASSED

9.2 getAppointment

TC 9.2.1	PASSED
TC 9.2.2	PASSED

9.3 getAppointmentByDipendente

TC 9.3.1	PASSED

TC 9.3.2	PASSED

9.4 getAllAppuntamentiOrdered

TC 9.4.1	PASSED
TC 9.4.2	PASSED

10. ClienteController

10.1 save

Parametro	Cliente
Presenza [P]	Presente Non presente

Identificativo	Tipologia	Esito
TC 10.1.1	P.1	PASSED
TC 10.1.2	P.2	PASSED

10.2 check

Parametro	email
Presenza [P]	Presente Non presente

Identificativo	Tipologia	Esito
TC 10.2.1	P.1	PASSED

TC 10.2.2	P.2	PASSED

10.3 update

Parametro	Cliente
Presenza [P]	Presente Non presente

Identificativo	Tipologia	Esito
TC 10.3.1	P.1	PASSED
TC 10.3.2	P.2	PASSED

10.4 login

TC 10.4.1	PASSED
TC 10.4.2	PASSED

11. DipendenteController

11.1 check

Parametro	email
Presenza [P]	Presente Non presente

Identificativo	Tipologia	Esito
TC 11.1.1	P.1	PASSED
TC 11.1.2	P.2	PASSED

11.2 saveEmployee

Parametro	Dipendente
Presenza [P]	Presente Non presente

Identificativo	**Tipologia**	**Esito**
TC 11.2.1	P.1	PASSED
TC 11.2.2	P.2	PASSED

11.3 update

Parametro	Dipendente
Presenza [P]	Presente Non presente

Identificativo	**Tipologia**	**Esito**
TC 11.3.1	P.1	PASSED
TC 11.3.2	P.2	PASSED

11.4 getDipendenteByDate

TC 11.4.1	PASSED
TC 11.4.2	PASSED

11.5 login

TC 11.5.1	PASSED

TC 11.5.2	PASSED

12. ServizioController

12.1 update

Parametro	Servizio
Presenza [P]	Presente Non presente

Identificativo	Tipologia	Esito
TC 12.1.1	P.1	PASSED
TC 12.1.2	P.2	PASSED

13. TitolareController

13.1 saveTitolare

Parametro	Titolare
Presenza [P]	1. Presente 2. Non presente

Identificativo	Tipologia	Esito
TC 13.1.1	P.1	PASSED
TC 13.1.2	P.2	PASSED

13.2 check

Parametro	email
Presenza [P]	Presente Non presente

Identificativo	Tipologia	Esito
TC 13.2.1	P.1	PASSED
TC 13.2.2	P.2	PASSED

13.3 update

Parametro	Titolare
Presenza [P]	Presente Non presente

Identificativo	Tipologia	Esito
TC 13.3.1	P.1	PASSED
TC 13.3.2	P.2	PASSED

13.4 login

TC 13.4.1	PASSED
TC 13.4.2	PASSED

Test schedule

Abbiamo optato per concentrare i nostri sforzi iniziali sui test delle singole componenti a livello dibusiness, poiché gran parte della logica critica risiede in quest'area. Di conseguenza, abbiamo avviato i test sulle singole entità e successivamente sulle classi DAO, utilizzando simulazioni (mock) per gestire eventuali dipendenze esterne.

Successivamente, ci siamo dedicati ai test di integrazione, riusando gli stessi casi di test utilizzatiper i test di unità, ma questa volta senza l'uso di mock, seguendo un approccio bottom-up. Infine, abbiamo eseguito il testing di sistema per garantire che il sistema soddisfi le aspettativedegli utenti finali.

Tutti i risultati dei test, insieme a eventuali modifiche apportate al sistema a seguito di erroririscontrati, sono stati accuratamente documentati nel Report di Esecuzione dei Test (TER).