Università degli Studi di Salerno

Corso di Ingegneria del Software

Auto Shop
Unit Test Plan
Versione 1.0

Data: 01/02/2018

Cognome Nome	Matricola
Corrado Mancino Alfredo	0512102506
Carbè Daniele	0512102326
Caloia Gennaro	0512102332

Indice

1. Introduzione
Il testing di unità rappresenta la fase di testing in cui si assicura che le componenti sviluppate funzionino singolarmente nel modo corretto.
Questo documento ha il compito di identificare la strategia di testing di unità per il sistema. In particolare, saranno specificate le componenti da testare e il modo in cui il testing dovrà essere eseguito.

2. Relazione con gli altri documenti
Per verificare la corretta implementazione delle varie componenti del sistema sono stati predisposti dei test cases basati sui requisiti funzionali identificati nella fase di analisi e specifica dei requisiti. I documenti ai quali si fa riferimento sono:
· *Test Plan;*
· *RAD;*
· *SDD ;*
· *DBD;*
· *ODD .*

3. Dettagli del Level Testing Design
3.1 Approccio di Unit Testing
Il primo modo di valutare le funzionalità di un sistema è quello di verificare che le componenti sviluppate funzionino singolarmente così come dovrebbero.
Quindi, all'interno del sistema dovranno essere testati innanzitutto gli oggetti Entity.
Inoltre, per il testing di unità è necessario utilizzare un approccio di tipo White-Box. In particolare, lo sviluppo dei casi di test sarà effettuato utilizzando il framework JUnit.

3.2 Componenti da testare
Di seguito vengono elencati le componenti del sistema da testare. Per quanto riguarda gli oggetti Entity, le componenti da testare sono:
● Auto.java;
● RicambiAuto.java;
● Client.java;
● Dipendente.java;
● Preventivo.java;
● Ordine.java;
● Rifornimento.java;

● Carrello.java;

4. Pass/Fail Criteria

Il testing ha successo se l’output osservato è diverso dall’output atteso: ciò significa che la fase di testing avrà successo se individuerà una failure. In tal caso questa verrà analizzata e, se legata ad un fault, si procederà alla sua correzione. Sarà infine iterata la fase di testing per verificare che la modifica non abbia impattato su altri componenti del sistema.
Al contrario, il testing fallirà se l’output osservato sarà uguale all’oracolo