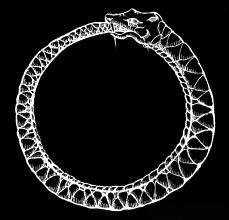
|  |
| --- |
| **Università degli Studi di Salerno Corso di Ingegneria del Software** |

**Uroboros  
Test Plan  
Versione 1.0**



Data: 10/01/2021

**Coordinatore del progetto:**

|  |  |
| --- | --- |
| Nome | Matricola |
| Andrea Amorosini | 005714 |
|  |  |

**Partecipanti:**

|  |  |
| --- | --- |
| Nome | Matricola |
| Andrea Amorosini | 005714 |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

|  |  |
| --- | --- |
| **Scritto da:** | Andrea Amorosini |

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Data | Versione | Descrizione | Autore |
| 11/01/2021 | 1.0 |  | Andrea Amorosini |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Indice

1. INTRODUZIONE 6

2. RELAZIONE CON ALTRI DOCUMENTI 6

3. OVERVIEW DEL SISTEMA 6

4. CARATTERISTICHE DA TESTARE E DA NON TESTARE 6

4.1. Da Testare 6

4.2. Da Non testare 6

5. CRITERI PER APPROVAZIONE E FALLIMENTO 7

6. APPROCCIO 7

7. MATERIALE USATO PER IL TESTING 7

8. CASI DI TEST 7

8.1. TestAccount 7

8.1.1. Oggetti del test 7

8.1.2. Specifiche di input 7

8.1.3. Specifiche di output 8

8.1.4. Requisiti d’ambiente 8

8.1.5. Procedura di testing 8

8.2. TestCarrello 9

8.2.1. Oggetti del test 9

8.2.2. Specifiche di input 9

8.2.3. Specifiche di output 9

8.2.4. Requisiti d’ambiente 9

8.2.5. Procedura di testing 9

8.3. TestCarta 10

8.3.1. Oggetti del test 10

8.3.2. Specifiche di input 10

8.3.3. Specifiche di output 10

8.3.4. Requisiti d’ambiente 10

8.3.5. Procedura di testing 10

8.4. TestCategoria 11

8.4.1. Oggetti del test 11

8.4.2. Specifiche di input 11

8.4.3. Specifiche di output 11

8.4.4. Requisiti d’ambiente 11

8.4.5. Procedura di testing 11

8.5. TestContact 12

8.5.1. Oggetti del test 12

8.5.2. Specifiche di input 12

8.5.3. Specifiche di output 12

8.5.4. Requisiti d’ambiente 12

8.5.5. Procedura di testing 12

8.6. TestIndirizzo 13

8.6.1. Oggetti del test 13

8.6.2. Specifiche di input 13

8.6.3. Specifiche di output 13

8.6.4. Requisiti d’ambiente 13

8.6.5. Procedura di testing 14

8.7. TestProdotto 14

8.7.1. Oggetti del test 14

8.7.2. Specifiche di input 14

8.7.3. Specifiche di output 14

8.7.4. Requisiti d’ambiente 15

8.7.5. Procedura di testing 15

8.8. TestTransazione 15

8.8.1. Oggetti del test 15

8.8.2. Specifiche di input 15

8.8.3. Specifiche di output 15

8.8.4. Requisiti d’ambiente 16

8.8.5. Procedura di testing 16

8.9. TestAccountAcquistoOperation 16

8.9.1. Oggetti del test 16

8.9.2. Specifiche di input 17

8.9.3. Specifiche di output 17

8.9.4. Requisiti D’Ambiente 17

8.9.5. Procedura di Testing 17

8.10. TestCategorieAndRimozioneCarrello 18

8.10.1.Oggetti del test 18

8.10.2. Specifiche di input 18

8.10.3. Specifiche di output 18

8.10.4. Requisiti d’ambiente 18

8.10.5. Procedura di Testing 18

8.11. TestFunzioniGestioneAccount 19

8.11.1. Oggetti del test 19

8.11.2. Specifiche di input 19

8.11.3. Specifiche di output 19

8.11.4. Requisiti d’Ambiente 19

8.11.5. Procedura di testing 20

8.12. TestFunzioniGestioneCategorie 20

8.12.1. Oggetti del test 20

8.12.2. Specifiche di input 20

8.12.3. Specifiche di output 21

8.12.4. Requisiti d’Ambiente 21

8.12.5. Procedura di testing 21

8.13. TestFunzioniGestioneProdotti 21

8.13.1. Oggetti del Test 21

8.13.2. Specifiche di input 22

8.13.3. Specifiche di output 22

8.13.4. Requisiti d’ambiente 22

8.13.5. Procedura di testing 22

8.14. TestNavigazionePagine 23

8.14.1. Oggetti del Test 23

8.14.2. Specifiche di input 23

8.14.3. Specifiche di output 23

8.14.4. Requisiti d’ambiente 23

8.14.5. Procedura di testing 23

8.15. TestRicercaProdotto 24

8.15.1. Oggetti del test 24

8.15.2. Specifiche di input 24

8.15.3. Specifiche di output 24

8.15.4. Requisiti d’ambiente 24

8.15.5. Procedura di testing 24

8.16. TestRegistrazione 24

8.16.1. Oggetti del test 24

8.16.2. Specifiche di input 25

8.16.3. Specifiche di output 25

8.16.4. Requisiti d’ambiente 26

8.16.5. Procedura di testing 26

1. INTRODUZIONE

Gli obiettivi del testing sono:

* L’assicurarsi del corretto funzionamento dei singoli sottosistemi
* L’assicurarsi della corretta interazione tra più sottosistemi
* L’assicurarsi di una corretta fruizione del sito
* L’assicurarsi che il sistema lavori in toto

1. RELAZIONE CON ALTRI DOCUMENTI

Tutti i casi di test sono stati ideati a partire dai requisiti funzionali e non, così come dai casi d’uso presentati nel [Requirement Analysis Document Uroboros](https://unisalerno-my.sharepoint.com/:w:/g/personal/a_amorosini_studenti_unisa_it/EblNRhfh6qtMjielr3oy0VcBhMvqCg__FZ9E682tBvgK9w), inoltre il metodo del testing è stato dato dall’organizzazione in sottosistemi del sistema così come viene presentato nel [SystemDesignDocument](https://unisalerno-my.sharepoint.com/:w:/g/personal/a_amorosini_studenti_unisa_it/ESMM6QiVxX9Eng6Kd9xqMycB87suPLJNXg0LmtbkBVsV2g).

1. OVERVIEW DEL SISTEMA

Il sistema parte dall’unico sottosistema indipendente dagli altri, JavaBeans, andando quindi a testare prima i vari Bean degli oggetti del sistema, poi prosegue con il testare il sottosistema Connection che utilizza alcuni dei servizi del sottosistema JavaBeans, dopodichè si passa al testing dei sottosistemi contenuti nel Presentation Layer e nel Bussiness Layer i quali dipendono dai sottosistemi Connection e JavaBeans.

1. CARATTERISTICHE DA TESTARE E DA NON TESTARE

## Da Testare

Risultano da testare:

* Metodi dei Bean per l’impostazione ed il recupero dei dati dagli oggetti
* Metodi dei DAO per il recupero e l’impostazione dei dati col database
* Metodi delle Servlet per la logica dietro al sito
* Visualizzazione delle jsp
* Combinazione delle quattro caratteristiche precedenti per la navigazione e l’utilizzo del sito

## Da Non testare

Non risultano da testare:

1. CRITERI PER APPROVAZIONE E FALLIMENTO

Il sistema verrà accettato con un percentuale di successo dei test dell’oltre 80%, se la percentuale non dovrebbe raggiungere l’80% il testing si dichiarerà fallito e si dovrà tornare al lavoro sul codice.

1. APPROCCIO

Per il testing del sistema viene utilizzato una tecnica Bottom-Up partendo dai sottosistemi indipendenti dagli altri sottosistemi per poi risalire a mano a mano aggiungendo altri sottosistemi al testing.

Per il testing specifico viene utilizzata la strategia Black-Box dove si verifica se l’output del sistema sia lo stesso dell’output previsto.

Per ridurre la ridondanza dei casi di test si è effettuato un partizionamento di equivalenza , aggregando così più casi di test sotto un unico caso di test.

1. MATERIALE USATO PER IL TESTING

Per il testing sono stati utilizzati:

* JUnit per il testing delle classi Java
* Selenium per il testing dell’interazione utente-sistema
* Selenium IDE per l’automatizzazione dei casi di test
* IntelliJ Idea per l’esecuzione dei test

1. CASI DI TEST

## TestAccount

### Oggetti del test

Questo test mette sotto test il componente Account del package JavaBeans

### Specifiche di input

Nessun input è richiesto dal tester visto che vengono dati di default per il testing dei setter.

I dati di default ai setter sono:

* Username = “Username”
* Password = “Password”
* NumCarta = “12345”
* NomeAzienda = “Azienda”
* Indirizzo = “Indirizzo”
* CF = “12345”
* isAdmin = “true”

### Specifiche di output

L’output sperato dato dall’esecuzione dei getter della classe è:

Username

Password

12345

Indirizzo

Azienda

12345

True

True

### Requisiti d’ambiente

Per poter eseguire il test c’è bisogno di:

* JUnit, deve essere specificato tra le librerie del sistema
* IntelliJ Idea , è l’ambiente da cui si avvia il test

### Procedura di testing

Il Test prova ad eseguire tutti i metodi della classe Account nel package JavaBeans eseguendo con ordine:

* + setUsername(String Username)
  + setPassword(String Password)
  + setNumCarta(int numCarta)
  + setIndirizzo(String indirizzo)
  + setCf(int cf)
  + setAdmin(boolean admin)
  + getUsername()
  + getPassword()
  + getCF()
  + getIndirizzo()
  + getNomeAzienda()
  + getNumCarta()
  + isAdmin()
  + equals(Account account)

## TestCarrello

### Oggetti del test

Questo test mette sotto test il componente Carrello del package JavaBeans

### Specifiche di input

Nessun input è richiesto dal tester visto che vengono dati di default per il testing dei setter

I dati di default ai setter sono:

* Quantita = “3”
* nomeProdotto = “prodotto2”
* descrizioneProdotto = “descrizione del prodotto”
* IDProdotto = “143”
* PrezzoProdotto = “23.34”
* Prodotto = “prodotto2”

### Specifiche di output

L’output sperato dopo l’esecuzione del test è :

1

143

0

### Requisiti d’ambiente

Per poter eseguire il test c’è bisogno di:

* JUnit, deve essere specificato tra le librerie del sistema
* IntelliJ Idea , è l’ambiente da cui si avvia il test

### Procedura di testing

Il test inizializza un oggetto di tipo Prodotto che servirà per il testing successivo

Poi il test prova ad eseguire tutti i metodi della classe Carrello nel package JavaBeans eseguendo con ordine:

put(Prodotto, int quantità)

getProdotti()

get(int IDProdotto)

remove(int IDProdottto)

getProdotti()

## TestCarta

### Oggetti del test

Questo test mette sotto test il componente Carta del package JavaBeans

### Specifiche di input

Nessun input è richiesto dal tester visto che vengono dati di default per il testing dei setter

I dati di default del setter sono:

* NumCarta = “12345”
* CSV = “123”
* NomeTitolare = “Nome”
* CognomeTitolare = “Cognome”
* DataDiScadenza = “24/03/2030”

### Specifiche di output

L’output sperato dopo l’esecuzione del test è:

12345

123

Nome

Cognome

24/03/2030

### Requisiti d’ambiente

Per poter eseguire il test c’è bisogno di:

* JUnit, deve essere specificato tra le librerie del sistema
* IntelliJ Idea , è l’ambiente da cui si avvia il test

### Procedura di testing

Il test prova ad eseguire tutti i metodi della classe Carta nel package JavaBeans eseguendo con ordine:

setNumCarta(int numCarta)

setCSV(int CSV)

setNomeTitolare(String nome)

setCognomeTitolare(String cognome)

setDataDiScadenza(String data)

getNumCarta()

getCSV()

getNomeTitolare()

getCognomeTitolare()

getDataDiScadenza()

## TestCategoria

### Oggetti del test

Questo test mette sotto test il componente Categoria del package JavaBeans

### Specifiche di input

Nessun input è richiesto dal tester visto che vengono dati di default per il testing del setter

I dati di default al setter sono:

* IDCategoria = “12”
* NomeCategoria = “nome”
* DescrizioneCategoria = “descrizione”

### Specifiche di output

L’output sperato dopo l’esecuzione del test è:

12

nome

descrizione

### Requisiti d’ambiente

Per poter eseguire il test c’è bisogno di:

* JUnit, deve essere specificato tra le librerie del sistema
* IntelliJ Idea , è l’ambiente da cui si avvia il test

### Procedura di testing

Il test prova ad eseguire tutti i metodi della classe Categoria nel package JavaBeans eseguendo con ordine:

setIdCategoria(int ID)

setNomeCategoria(String nome)

setDescrizioneCategoria(String descrizione)

getIDCategoria()

getNomeCategoria()

getDescrizioneCategoria()

## TestContact

### Oggetti del test

Questo test mette sotto test il componente Contact del package JavaBeans

### Specifiche di input

Nessun input è richiesto dal tester visto che vengono dati di default dal testing per i setter

I dati di default sono:

* CF = “12345”
* Nome = “nome”
* Cognome = “cognome”
* Mail = [mail@mail.it](mailto:mail@mail.it)
* DataDiNascita = “24/03/2030”
* Telefono = “333333333”

### Specifiche di output

L’output sperato dopo l’esecuzione del test è:

12345

Nome

Cognome

[mail@mail.it](mailto:mail@mail.it)

24/03/20000

333333333

### Requisiti d’ambiente

Per poter eseguire il test c’è bisogno di:

* JUnit, deve essere specificato tra le librerie del sistema
* IntelliJ Idea , è l’ambiente da cui si avvia il test

### Procedura di testing

Il test prova ad eseguire tutti i metodi della classe Contact nel package JavaBeans eseguendo in ordine:

setCF(int CF)

setNome(String nome)

setCognome(String cognome)

setEmail(String mail)

setDataDiNascita(String data)

setTelefono(String telefono)

getCF()

getNome()

getCognome()

getEmail()

getTelefono()

getDataDiNascita()

## TestIndirizzo

### Oggetti del test

Questo test prevede il testing del componente Indirizzo del package JavaBeans

### Specifiche di input

Nessun input è richiesto dal tester visto che vengono dati di default per il testing del setter

I dati di default al setter sono:

* Via = “via”
* Via2 = “via2”
* Nazione = “nazione”
* Regione = “regione”
* Città = “città”
* CAP = “12345”

### Specifiche di output

L’output sperato dopo l’esecuzione del test è:

via

via2

nazione

regione

città

12345

### Requisiti d’ambiente

Per poter eseguire il test c’è bisogno di:

* JUnit, deve essere specificato tra le librerie del sistema
* IntelliJ Idea , è l’ambiente da cui si avvia il test

### Procedura di testing

Il test prova ad eseguire tutti i metodi della classe Indirizzo nel package JavaBeans eseguendo con ordine:

setVia(String via)

setVia2(String via2)

setNazione(String nazione)

setRegione(String regione)

setCitta(String città)

setCAP(int CAP)

getVia()

getVia2()

getNazione()

getRegione()

getCitta()

getCAP()

## TestProdotto

### Oggetti del test

Questo test prevede il testing del componente Prodotto del package JavaBeans

### Specifiche di input

Nessun input è richiesto dal tester visto che vengono dati di default per il testing del setter

I dati di default al setter sono:

* NomeProdotto = “prodotto”
* DescrizioneProdotto = “descrizione del prodotto”
* IDProdotto = “134”
* PrezzoProdotto = “23.34”
* Quantita = “3”

### Specifiche di output

L’output sperato dopo l’esecuzione del test è:

prodotto

descrizione del prodotto

134

3

23.34

### Requisiti d’ambiente

Per poter eseguire il test c’è bisogno di:

* JUnit, deve essere specificato tra le librerie del sistema
* IntelliJ Idea , è l’ambiente da cui si avvia il test

### Procedura di testing

Il test prova ad eseguire tutti i metodi della classe Prodotto nel package JavaBeans eseguendo con ordine:

setNomeProdotto(String nome)

setDescrizioneProdotto(String descrizione)

setIDprodotto(int ID)

setPrezzoProdotto(double prezzo)

setQuantita(int quantita)

getNomeProdotto()

getDescrizioneProdotto()

getIDprodotto()

getQuantita()

getPrezzoProdotto()

## TestTransazione

### Oggetti del test

Questo test prevede il testing del componente Transazione nel package JavaBeans

### Specifiche di input

Nessun input è richiesto dal tester visto che vengono dati di default per il testing del setter

I dati di default al setter sono:

* IDtransazione = “12”
* Username = “Username”
* Data = “valueOf(“2000-03-24”)”
* Spesa = “35.40”

### Specifiche di output

L’output sperato dopo l’esecuzione del test è:

12

Username

24/03/2000

35.40

### Requisiti d’ambiente

Per poter eseguire il test c’è bisogno di:

* JUnit, deve essere specificato tra le librerie del sistema
* IntelliJ Idea , è l’ambiente da cui si avvia il test

### Procedura di testing

Il test prova ad eseguire tutti i metodi della classe Transazione nel package JavaBeans eseguendo con ordine:

setIdTransazione(int ID)

setUsername(String username)

setData(Date data)

setSpesa(double spesa)

getIdtransazione()

getUsername()

getData()

getSpesa()

## TestAccountAcquistoOperation

### Oggetti del test

Questo test prevede il testing dei componenti:

* CatalogoServlet del package ControlloProdotti
* ProdottoServlet del package ControlloProdotti
* CarrelloServlet del package ControlloAcquisto
* LoginServlet del package ControlloAccount
* AcquistoServlet del package COntrolloAcquisto
* StoricoServlet del package ControlloOrdini
* ShowAccountDetail del package ControlloAccount
* ModificaDatiAccountServlet del package ControlloAccount
* LogoutServlet del package ControlloAccount
* AccountDAO del package Connection
* CartaDAO del package Connection
* CategoriaDAO del package Connection
* ConPool del package Connection
* IndirizzoDAO del package Connection
* ProdottoDAO del package Connection
* TransazioneDAO del package Connection
* Account del package JavaBeans
* Carrello del package JavaBeans
* Carta del package JavaBeans
* Categoria del package JavaBeans
* Contact del package JavaBeans
* Indirizzo del package JavaBeans
* Prodotto del package JavaBeans
* Transazione del package JavaBeans

### Specifiche di input

Nessun input è richiesto dal tester visto che vengono dati di default

I dati di default sono:

* Username = “Utente4”
* Password = “Utente40”
* newUsername = “Utente40”
* newPassword = “Utente4000”

### Specifiche di output

L’output sperato dopo l’esecuzione del test è:

Visualizzazione Catalogo OK

Apertura prodotto OK

Aggiunta al Carrello OK

Acquisto senza Login OK

Acquisto OK

Visualizzazione Storico Ordini OK

Modifica Dati Utente OK

Logout OK

### Requisiti D’Ambiente

Per poter eseguire il test c’è bisogno di:

* Aver Caricato il file.war del sito sulla piattaforma web di Tomcat
* Aver avviato il database collegato al sito
* JUnit, deve essere specificato tra le librerie del sistema
* IntelliJ Idea , è l’ambiente da cui si avvia il test

### Procedura di Testing

Il test esegue le seguenti procedure, con ordine:

1. Seleziona l’opzione per la visualizzazione del catalogo completo
2. Seleziona un prodotto
3. Aggiunge il prodotto al carrello
4. Prova ad eseguire l’acquisto senza aver fatto il login a priori
5. Esegue il login
6. Esegue l’acquisto
7. Seleziona l’opzione per la visualizzazione dello storico degli ordini
8. Seleziona i dettagli di una transazione
9. Seleziona l’opzione per la visualizzazione dei propri dati utente
10. Seleziona l’opzione per modificare i propri dati utenti
11. Inserisce un nuovo username e password
12. Esegue il Logout

## TestCategorieAndRimozioneCarrello

### 8.10.1.Oggetti del test

Questo test prevede il testing dei seguenti componenti:

* CategoriaServlet del package ControlloCategorie
* ProdottoServlet del package ControlloProdotti
* CarrelloServlet del package ControlloAcquisto
* CarrelloRemoveServlet del package ControlloAcquisto
* CategoriaDAO del package Connection
* ProdottoDAO del package Connection
* Categoria del package JavaBeans
* Prodotto del package JavaBeans
* Carrello del package JavaBeans

### Specifiche di input

Nessun input è richiesto dal tester

### Specifiche di output

L’output sperato dopo l’esecuzione del test è:

Visualizzazione Categoria OK

Selezione Prodotto OK

Aggiunta prodotto al carrello OK

Rimozione Prodotto dal carrello OK

### Requisiti d’ambiente

Per poter eseguire il test c’è bisogno di:

* Aver Caricato il file.war del sito sulla piattaforma web di Tomcat
* Aver avviato il database collegato al sito
* JUnit, deve essere specificato tra le librerie del sistema
* IntelliJ Idea , è l’ambiente da cui si avvia il test

### Procedura di Testing

Il test esegue le seguenti procedure, in ordine:

1. Seleziona una delle categorie dall’header
2. Seleziona uno dei prodotti della categoria selezionata
3. Aggiunge il prodotto al carrello
4. Rimuove il prodotto dal carrello

## TestFunzioniGestioneAccount

### Oggetti del test

Questo test prevede il testing dei seguenti componenti:

* LoginServlet del package ControlloAccount
* CambioPaginaServlet del package HomePage
* GestioneAccount del package ControlloAccount
* ShowStoricoAdminServlet del package ControlloOrdini
* ShowAccountDetailServlet del package ControlloAccount
* DeleteAccountServlet del package ControlloAccount
* AccountDAO del package Connection
* ProdottoDAO del package Connection
* TransazioneDAO del package Connection
* Account del package JavaBeans
* Prodotto del package JavaBeans
* Transazione del package JavaBeans

### Specifiche di input

Nessun input è richiesto dal tester visto che vengono dati di default

I dati di default sono:

* Username = “admin1”
* Password = “admin1”

### Specifiche di output

L’output sperato dopo l’esecuzione del test è:

Accesso come Gestore OK

Visualizzazione Gestione Account OK

Visualizzazione Storico Ordini OK

Visualizzazione Dettagli Account OK

Eliminazione Account OK

### Requisiti d’Ambiente

Per poter eseguire il test c’è bisogno di:

* Aver Caricato il file.war del sito sulla piattaforma web di Tomcat
* Aver avviato il database collegato al sito
* JUnit, deve essere specificato tra le librerie del sistema
* IntelliJ Idea , è l’ambiente da cui si avvia il test

### Procedura di testing

Il test esegue le seguenti procedure, in ordine:

1. Effettua l’accesso come gestore
2. Seleziona la DashBoard dalle opzioni account
3. Seleziona la gestione degli account
4. Seleziona l’opzione di visualizzazione dello storico ordini in corrispondenza di un account
5. Torna indietro e seleziona l’opzione di visualizzazione dei dettagli di un account
6. Torna indietro e seleziona l’opzione per l’eliminazione di un account

## TestFunzioniGestioneCategorie

### Oggetti del test

Questo test prevede il testing dei seguenti componenti:

* LoginServlet del package ControlloAccount
* CambioPaginaServlet del package HomePage
* GestioneCategorie del package ControlloCategorie
* AggiungiCategoriaServlet del package ControlloCategorie
* AggiornaCategoriaSevrlet del package ControlloCategorie
* DeleteCategoriaServlet del package ControlloCategorie
* AccountDAO del package Connection
* CategoriaDAO del package Connection
* Account del package JavaBeans
* Categoria del package JavaBeans

### Specifiche di input

Nessun input è richiesto dal tester visto che vengono dati di default

I dati di default sono:

* Username = “admin1”
* Password = “admin1”
* IDCategoria = “4”
* nomeCategoria = “categoria4”
* descrizione = “categoria4”
* newDescrizione = “Descrizione della categoria 4”

### Specifiche di output

L’output sperato dopo l’esecuzione del test è:

Accesso come Gestore OK

Visualizzazione Gestione Categorie OK

Aggiunta di una Categoria OK

Modifica Dati Categoria OK

Rimozione Categoria OK

### Requisiti d’Ambiente

Per poter eseguire il test c’è bisogno di:

* Aver Caricato il file.war del sito sulla piattaforma web di Tomcat
* Aver avviato il database collegato al sito
* JUnit, deve essere specificato tra le librerie del sistema
* IntelliJ Idea , è l’ambiente da cui si avvia il test

### Procedura di testing

Il test esegue le seguenti procedure , in ordine, :

1. Effettua il login come gestore
2. Seleziona l’opzione DashBoard
3. Seleziona l’opzione “Gestione Categorie”
4. Seleziona “Aggiungi una Categoria”
5. Inserisce i dati richiesti e seleziona “Salva”
6. Seleziona l’opzione “Modifica Dati”
7. Inserisce i dati aggiornati e seleziona “Modifica”
8. Seleziona l’opzione “Elimina Categoria”

## TestFunzioniGestioneProdotti

### Oggetti del Test

Questo test prevede il testing dei seguenti componenti:

* LoginServlet del package ControlloAccount
* CambioPaginaServlet del package HomePage
* GestioneProdottiServlet del package ControlloCategorie
* AggiungiProdottoServlet del package ControlloCategorie
* ModificaDatiProdottoSevrlet del package ControlloCategorie
* CancellaProdottoServlet del package ControlloCategorie
* AccountDAO del package Connection
* ProdottoDAO del package Connection
* CategoriaDAO del package Connection
* Account del package JavaBeans
* Prodotto del package JavaBeans
* Categoria del package JavaBeans

### Specifiche di input

Nessun input è richiesto dal tester visto che vengono dati di default

I dati di default sono:

* Username = “admin1”
* Password = “admin1”
* IDProdotto = “12”
* nomeProdotto = “prodotto12”
* descrizioneProdotto = “loremi ipsu descriptus”
* prezzo = “20.00”
* quantita = “3”
* Categoria = “Hardware”
* newNome = “prodotto13”

### Specifiche di output

L’output sperato dopo l’esecuzione del test è:

Accesso come Gestore OK

Visualizzazione gestione Prodotti OK

Aggiunta di un prodotto OK

Modifica Dati Prodotto OK

Rimozione Prodotto OK

### Requisiti d’ambiente

Per poter eseguire il test c’è bisogno di:

* Aver Caricato il file.war del sito sulla piattaforma web di Tomcat
* Aver avviato il database collegato al sito
* JUnit, deve essere specificato tra le librerie del sistema
* IntelliJ Idea , è l’ambiente da cui si avvia il test

### Procedura di testing

Il test esegue le seguenti procedure, in ordine :

1. Effettua il login come gestore
2. Seleziona l’opzione DashBoard
3. Seleziona l’opzione “Gestione Prodotti”
4. Seleziona l’opzione “Aggiungi un Prodotto”
5. Inserisce i dati richiesti e seleziona “Salva”
6. Seleziona l’opzione Modifica Dati Prodotto
7. Inserisce i dati aggiornati e seleziona “Modifica”
8. Seleziona l’opzione “Elimina Prodotto”

## TestNavigazionePagine

### Oggetti del Test

Questo test prevede il testing dei seguenti componenti:

* CambioPaginaServlet del package HomePage
* ProdottoDAO del package Connection
* Prodotto del package JavaBeans

### Specifiche di input

Nessun input è richiesto dal tester

### Specifiche di output

L’Output sperato dopo l’esecuzione del test è:

Apertura pagina Contact OK

Apertura pagina About Us OK

Ritorno alla Home OK

### Requisiti d’ambiente

Per poter eseguire il test c’è bisogno di:

* Aver Caricato il file.war del sito sulla piattaforma web di Tomcat
* Aver avviato il database collegato al sito
* JUnit, deve essere specificato tra le librerie del sistema
* IntelliJ Idea , è l’ambiente da cui si avvia il test

### Procedura di testing

Il test esegue le seguenti procedure, in ordine:

1. Seleziona il pulsante “Contact” nella navBar
2. Seleziona il pulsante “About Us” nella navBar
3. Seleziona il logo del sito

## TestRicercaProdotto

### Oggetti del test

Questo test prevede il testing dei seguenti componenti:

* SearchServlet del package ControlloProdotti
* ProdottoDAO del package Connection
* Prodotto del package JavaBeans

### Specifiche di input

Nessun input è richiesto dal tester visto che vengono dati di default

I dati di default sono:

* searchTerm = “hardware”

### Specifiche di output

L’output sperato dopo l’esecuzione del test è:

Ricerca Prodotto OK

### Requisiti d’ambiente

Per poter eseguire il test c’è bisogno di:

* Aver Caricato il file.war del sito sulla piattaforma web di Tomcat
* Aver avviato il database collegato al sito
* JUnit, deve essere specificato tra le librerie del sistema
* IntelliJ Idea , è l’ambiente da cui si avvia il test

### Procedura di testing

Il test esegue le seguenti procedure, in ordine:

1. Muove il mouse sull’icona della ricerca
2. Inserisce il termine di ricerca nella casella che compare
3. Seleziona l’icona di ricerca

## TestRegistrazione

### Oggetti del test

Questo test prevede il testing dei seguenti componenti:

* CambioPagina del package HomePage
* RegistrazioneServlet del package ControlloAccount
* AccountDAO del package Connection
* ContactDAO del package Connection
* CartaDAO del package Connection
* IndirizzoDAO del package Connection
* Account del package JavaBeans
* Contact del package JavaBeans
* Carta del package JavaBeans
* Indirizzo del package JavaBeans

### Specifiche di input

Nessun input è richiesto dal tester visto che vengono dati di default

I dati di default sono:

* Username = “utente4”
* Password = “Utente40”
* Verifica Password = “Utente40”
* Nome = “Utente”
* Cognome = “2”
* CF = “8476847694”
* dataDiNascita = “2000-03-24”
* telefono = “947864”
* mail = “mail2@mail.it”
* #carta = “84769376”
* tipoCarta = “visa”
* dataDiScadenza = “2030-01-20”
* CSV = “234”
* nomeTitolare = “utente”
* cognomeTitolare = “2”
* via = “via10”
* via2 = “2”
* nazione = “nazione2”
* regione = “regione2”
* CAP = “479468”
* Città = “citta2”

### Specifiche di output

L’output sperato dopo l’esecuzione del test è:

Registrazione OK

### Requisiti d’ambiente

Per poter eseguire il test c’è bisogno di:

* Aver Caricato il file.war del sito sulla piattaforma web di Tomcat
* Aver avviato il database collegato al sito
* JUnit, deve essere specificato tra le librerie del sistema
* IntelliJ Idea , è l’ambiente da cui si avvia il test

### Procedura di testing

Il test esegue le seguenti procedure, in ordine:

1. Seleziona l’opzione “Registrati” dalla navBar
2. Inserisce i dati necessari
3. Seleziona il pulsante “Registrati”