*Test Plan*

*Di Unità*

*Progetto*

*OctoPlus*



**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Data | Versione | Descrizione | Autore |
| 22/01/2024 | **0.1** | **Prima stesura** | **Tutto il team** |
| 25/01/2024 | **0.2** | **Aggiunti CartaUnitDao e ProdottoUnitDao** | **Orlando Tomeo** |
| 28/01/2024 | **0.3** | **Aggiunti PhotoUnitDao e CarrelloUnitDao** | **Donnarumma Salvatore** |
| 06/02/2024 | **0.4** | **Aggiunto sommario + OrdineUnitDao** | **Donnarumma Salvatore** |

Sommario

[1. Introduzione 3](#_Toc158151121)

[2. Relazioni con altri documenti 3](#_Toc158151122)

[3. Componenti da testare 3](#_Toc158151123)

[4. Test case 3](#_Toc158151124)

[4.1 TCU1\_1: UtenteUnitDaoData 3](#_Toc158151125)

[4.2 TCU2\_1: CartaUnitDaoData 8](#_Toc158151126)

[4.3 TCU3\_1: ProdottoUnitDaoData 9](#_Toc158151127)

[4.3 TCU3\_2: PhotoUnitDaoData 14](#_Toc158151128)

[4.4 TCU4\_1: OrdiniUnitDaoData 15](#_Toc158151129)

[4.5 TCU5\_1: CarrelloUnitDaoData 17](#_Toc158151130)

# Introduzione

L’obbiettivo di questo documento è quello di pianificare e progettare i test case di riferimento per l’implementazione dei test di unità.

# Relazioni con altri documenti

* Test Plan: le informazioni riguardanti l’approccio che verrà utilizzato per il testing sono presenti nel Test Plan
* O.D.D: faremo riferimento alle interfacce definite nell’ODD.

# Componenti da testare

- Package Gestione Utenti

* IUserDao
* ICardDao
* IGestoreDao

- Package Gestione Prodotti

* IProdottoDao
* IPhotoDao

- Package Gestione Ordini

* OrdineDaoDataSource

- Package Carrello

* ICarrelloDao

# Test case

## TCU1\_1: UtenteUnitDaoData

**TCU1\_1\_1 doRetrieveByEmail(User user):**

|  |  |
| --- | --- |
| **Parametri** | user |
| **Oggetti dell’ambiente** | Database |
| **Categorie** | **Scelte** |
| **user** | US1: user null  US2: user non null |
| **user.email** | ER1: formato valido [property email\_corretta]  ER2: formato null or vuoto or non valido  EP1: email già presente nel DB [if email\_corretta]  EP2: email non presente nel DB if[email\_corretta] |

**Test Frame**

|  |  |
| --- | --- |
| **Combinazione** | **Oracolo** |
| US1-ER1-EP1 | Restituisce Utente user relativo all’email inserita |
| US1-ER1-EP2 | Restituisce null |
| US1-ER2 | Lancia una CheckException |
| US2 | Lancia una CheckException |

**TCU1\_1\_2 doRetrieveByKey(User user):**

|  |  |
| --- | --- |
| **Parametri** | user |
| **Oggetti dell’ambiente** | Database |
| **Categorie** | **Scelte** |
| **user** | US1: user null  US2: user non null |
| **user.idutente** | ID1: formato valido [property: id\_corretto]  ID2: non valido  IP1: idutente già presente nel DB [if id\_corretto]  IP2: idutente non presente nel DB [if id\_corretto] |

**Test Frame**

|  |  |
| --- | --- |
| **Combinazione** | **Oracolo** |
| ID1-IP1 | Restituisce Utente user relativo all’email inserita |
| ID1-IP2 | Restituisce null |
| ID2 | Lancia una CheckException |

**TCU1\_1\_2 doRetrieveAllUser(String order):**

|  |  |
| --- | --- |
| **Parametri** | order |
| **Oggetti dell’ambiente** | Database |
| **Categorie** | **Scelte** |
| order | OR1: order = “email”  OR2: order = “nome”  OR3: order = null  OR4: order = “”  OR5 : order non “mail” e non “nome” |

**Test Frame**

|  |  |
| --- | --- |
| **Combinazione** | **Oracolo** |
| OR1 | Restituisce un ArrayList<Utente> di tutti gli utenti presenti nel database in ordine di email (crescente) |
| OR2 | Restituisce un ArrayList<Utente> di tutti gli utenti presenti nel database in ordine di  nome (crescente) |
| OR3 | Restituisce un ArrayList<Utente> di tutti gli utenti presenti nel database in ordine di  idutente (crescente) |
| OR4 | Restituisce un ArrayList<Utente> di tutti gli utenti presenti nel database in ordine di  idutente (crescente) |
| OR5 | Lancia una CheckException |

**TCU1\_1\_3 doSaveUser (User user):**

|  |  |
| --- | --- |
| **Parametri** | user |
| **Oggetti dell’ambiente** | Database |
| **Categorie** | **Scelte** |
| User | US1: user non vuoto  US2: user vuoto |
| user.email | FE1: formato valido [property email\_corretta]  FE2: formato non valido  EP1: email non presente nel DB [if email\_corretta]  EP2: email presente nel DB [if email\_corretta] |
| user.nome | NU1: formato valido  NU2: formato non valido |
| user.cognome | CU1: formato valido  CU2: formato valido |
| user.password | LP1: lunghezza password >= 12  LP2: lunghezza password < 12 |
| user.numero\_telefono | FT1: formato numero di telefono  FT2: formato numero di telefono non valido |
| user.ruolo | FR1: formato valido  FR2: formato non valido |

**Test Frame**

|  |  |
| --- | --- |
| **Combinazione** | **Oracolo** |
| US1-FE1-EP1-NU1-CU1-LP1-FT1-FR1 | Salva l’utente nel DB |
| US1-FE2-EP1-NU1-CU1-LP1-FT1-FR1 | Lancia una CheckException |
| US1-FE1-EP2-NU1-CU1-LP1-FT1-FR1 | Lancia un SQLIntegrityException |
| US1-FE1-EP1-NU2-CU1-LP1-FT1-FR1 | Lancia una CheckException |
| US1-FE1-EP1-NU1-CU2-LP1-FT1-FR1 | Lancia una CheckException |
| US1-FE1-EP1-NU1-CU1-LP2-FT1-FR1 | Lancia una CheckException |
| US1-FE1-EP1-NU1-CU1-LP1-FT2-FR1 | Lancia una CheckException |
| US1-FE1-EP1-NU1-CU1-LP1-FT1-FR2 | Lancia una CheckException |
| US2 | Lancia una CheckException |

**TCU1\_1\_4 changePass(User user, String pass):**

|  |  |
| --- | --- |
| **Parametri** | user, pass |
| **Oggetti dell’ambiente** | Database |
| **Categorie** | **Scelte** |
| user | US1: user non vuoto  US2: user vuoto |
| user.email | FE1: formato valido [property email\_corretta]  FE2: formato non valido  EM1: email già presente [if email\_corretta]  EM2: email non presente [if email\_corretta] |
| pass | FP1: formato valido  FP2: formato non valido |

**Test Frame**

|  |  |
| --- | --- |
| **Combinazione** | **Oracolo** |
| US1-FE1-EM1-FP1 | Cambia il campo password relativo all’utente con email corrispondente all’email passata |
| US1-FE2-EM1-FP1 | Lancia una CheckException |
| US1-FE1-EM2-FP1 | Non viene effettuata alcuna modifica alle tuple esistenti sul db |
| US1-FE1-EM1-FP2 | Lancia una CheckException |
| US2 | Lancia una CheckException |

**TCU1\_1\_5 doSaveGestore(User user):**

|  |  |
| --- | --- |
| **Parametri** | user |
| **Oggetti dell’ambiente** | Database |
| **Categorie** | **Scelte** |
| User | US1: user non vuoto  US2: user vuoto |
| user.email | FE1: formato valido [property email\_corretta]  FE2: formato non valido  EP1: email non presente nel DB [if email\_corretta]  EP2: email presente nel DB [if email\_corretta] |
| user.cognome | CU1: formato valido  CU2: formato valido |
| user.password | LP1: lunghezza password >= 12  LP2: lunghezza password < 12 |
| user.numero\_telefono | FT1: formato numero di telefono  FT2: formato numero di telefono non valido |
| user.ruolo | FR1: formato valido  FR2: formato non valido |

**Test Frame**

|  |  |
| --- | --- |
| **Combinazione** | **Oracolo** |
| US1-FE1-EP1-CU1-LP1-FT1-FR1 | Salva l’utente nel DB |
| US1-FE2-EP1 -CU1-LP1-FT1-FR1 | Lancia una CheckException |
| US1-FE1-EP2-CU1-LP1-FT1-FR1 | Lancia un SQLIntegrityException |
| US1-FE1-EP1 -CU2-LP1-FT1-FR1 | Lancia una CheckException |
| US1-FE1-EP1-CU1-LP2-FT1-FR1 | Lancia una CheckException |
| US1-FE1-EP1-CU1-LP1-FT2-FR1 | Lancia una CheckException |
| US1-FE1-EP1-CU1-LP1-FT1-FR2 | Lancia una CheckException |
| US2 | Lancia una CheckException |

**TCU1\_1\_6 changePassGestore(User user):**

|  |  |
| --- | --- |
| **Parametri** | user |
| **Oggetti dell’ambiente** | Database |
| **Categorie** | **Scelte** |
| user | US1: user non vuoto  US2: user vuoto |
| user.email | FE1: formato valido [property email\_corretta]  FE2: formato non valido  EM1: email già presente [if email\_corretta]  EM2: email non presente [if email\_corretta] |
| user.pass | FP1: formato valido  FP2: formato non valido |

**Test Frame**

|  |  |
| --- | --- |
| **Combinazione** | **Oracolo** |
| US1-FE1-EM1-FP1 | Cambia il campo password relativo all’utente con email corrispondente all’email passata |
| US1-FE2-EM1-FP1 | Lancia una CheckException |
| US1-FE1-EM2-FP1 | Non viene effettuata alcuna modifica alle tuple esistenti sul db |
| US1-FE1-EM1-FP2 | Lancia una CheckException |
| US2 | Lancia una CheckException |

**TCU1\_1\_7 doDeleteGestore(User user):**

|  |  |
| --- | --- |
| **Parametri** | user |
| **Oggetti dell’ambiente** | Database |
| **Categorie** | **Scelte** |
| user | US1: user non vuoto  US2: user vuoto |
| user.email | FE1: formato valido [property email\_corretta]  FE2: formato non valido  EP1: email presente nel DB [if email\_corretta]  EP2: email non presente nel DB [if email\_corretta] |

**Test Frame**

|  |  |
| --- | --- |
| **Combinazione** | **Oracolo** |
| US1-FE1-EP1 | Cancella l’utente nel DB |
| US1-FE2-EP1 | Lancia una CheckException |
| US1-FE1-EP2 | Non viene effettuata alcuna modifica alle tuple esistenti sul db |
| US2 | Lancia una CheckException |

## TCU2\_1: CartaUnitDaoData

**TCU2\_1\_1 cancellaCarta():**

|  |  |
| --- | --- |
| **Parametri** | Carta |
| **Oggetti dell’ambiente** | Database |
| **Categorie** | **Scelte** |
| carta | CS1: carta non vuoto  CS2: carta vuoto |
| Carta.idCarta | CF1: formato valido [property id\_corretto]  CF2: formato non valido  CV1: id già presente [if id\_corretto]  CV2: id non presente [if id\_corretto] |

**Test Frame**

|  |  |
| --- | --- |
| **Combinazione** | **Oracolo** |
| CS1-CF1-CV1 | Cancella la carta dal database |
| CS1-CF2-CV1 | Lancia una CheckException |
| CS1-CF1-CV2 | Non viene effettuata alcuna modifica alle tuple esistenti sul db |
| CS2 | Lancia una CheckException |

**TCU2\_1\_2 salvaCarta():**

|  |  |
| --- | --- |
| **Parametri** | Carta |
| **Oggetti dell’ambiente** | Database |
| **Categorie** | **Scelte** |
| carta | CS1: carta non vuota  CS2: carta vuota |
| Carta.idCarta | CF1: formato valido [property id\_corretto]  CF2: formato non valido  CV1: id non presente [if id\_corretto]  CV2: id già presente [if id\_corretto] |
| Carta.proprietario | CP1: formato valido  CP2: formato non valido |
| Carta.numero\_carta | CN1: formato valido  CN2: formato non valido |
| Carta.data\_scadenza | CD1: formato valido  CD2: formato non valido |

**Test Frame**

|  |  |
| --- | --- |
| **Combinazione** | **Oracolo** |
| CS1-CF1-CV1-CP1-CN1-CD1 | Salva la carta nel DB |
| CS1-CF2-CV1-CP1-CN1-CD1 | Lancia una CheckException |
| CS1-CF1-CV2-CP1-CN1-CD1 | Sovrascrivere una carta già esistente |
| CS1-CF1-CV1-CP2-CN1-CD1 | Lancia una CheckException |
| CS1-CF1-CV1-CP2-CN2-CD1 | Lancia una CheckException |
| CS1-CF1-CV1-CP2-CN1-CD2 | Lancia una CheckException |
| CS2 | Lancia una CheckException |

**TCU2\_1\_3 recuperaCarta():**

|  |  |
| --- | --- |
| **Parametri** | Carta |
| **Oggetti dell’ambiente** | Database |
| **Categorie** | **Scelte** |
| carta | CS1: carta non vuoto  CS2: carta vuoto |
| Carta.idCarta | CF1: formato valido [property id\_corretto]  CF2: formato non valido  CV1: id non presente [if id\_corretto]  CV2: id già presente [if id\_corretto] |

**Test Frame**

|  |  |
| --- | --- |
| **Combinazione** | **Oracolo** |
| CS1-CF1-CV2 | Recupera la carta dal DB |
| CS1-CF2-CV2 | Lancia una CheckException |
| CS1-CF1-CV1 | Restituisce null |
| CS2 | Lancia una CheckException |

## TCU3\_1: ProdottoUnitDaoData

**TCU3\_1\_1 doSave(Product product):**

|  |  |
| --- | --- |
| **Parametri** | product |
| **Oggetti dell’ambiente** | Database |
| **Categorie** | **Scelte** |
| product | PR1: product non vuoto  PR2: product vuoto |
| product.nome | NP1: formato valido  NP2: formato non valido |
| product.categoria | CAT1: formato valido  CAT2: formato non valido |
| product.prezzo | P1: prezzo>0  P2: prezzo<=0 |
| product.descrizione | DP1: formato valido  DP2: formato valido |
| product.image | IMG1: formato valido  IMG: formato non valido |
| product.statistiche | ST1: formato valido  ST2: formato non valido |
| product.taglia | TF1: formato valido [property taglie\_corrette]  TF2: formato non valido  TP1: Taglie.S +Taglie.M + Taglie.L + Taglie.XL >=0 [if taglie\_corrette]  TP2: Taglie.S +Taglie.M + Taglie.L + Taglie.XL<0 [if taglie\_corrette] |

**Test Frame**

|  |  |
| --- | --- |
| **Combinazione** | **Oracolo** |
| PR1-NP1-CAT1-P1-DP1-IMG1-ST1-TF1-TP1 | Salva il prodotto nel DB |
| PR1-NP2-CAT1-P1-DP1-IMG1-ST1-TF1-TP1 | Lancia una CheckException |
| PR1-NP1-CAT2-P1-DP1-IMG1-ST1-TF1-TP1 | Lancia una CheckException |
| PR1-NP1-CAT1-P2-DP1-IMG1-ST1-TF1-TP1 | Lancia una CheckException |
| PR1-NP1-CAT1-P1-DP2-IMG1-ST1-TF1-TP1 | Lancia una CheckException |
| PR1-NP1-CAT1-P1-DP1-IMG2-ST1-TF1-TP1 | Lancia una CheckException |
| PR1-NP1-CAT1-P1-DP1-IMG1-ST2-TF1-TP1 | Lancia una CheckException |
| PR1-NP1-CAT1-P1-DP1-IMG1-ST1-TF2-TP1 | Lancia una CheckException |
| PR1-NP1-CAT1-P1-DP1-IMG1-ST1-TF1-TP2 | Lancia una CheckException |
| PR2 | Lancia una CheckException |

**TCU3\_1\_2 doRetrieveByKey(Prodotto product):**

|  |  |
| --- | --- |
| **Parametri** | prodotto |
| **Oggetti dell’ambiente** | Database |
| **Categorie** | **Scelte** |
| product.idProdotto | ID1: formato valido [property: id\_corretto]  ID2: formato non valido  IP1: idProdotto già presente nel DB [if id\_corretto]  IP2: idProdotto non presente nel DB [if id\_corretto] |

**Test Frame**

|  |  |
| --- | --- |
| **Combinazione** | **Oracolo** |
| ID1-IP1 | Restituisce Prodotto product relativo all’id inserito |
| ID1-IP2 | Restituisce null |
| ID2 | Lancia una CheckException |

**TCU3\_1\_3 doRetrieveAllProducts(String order):**

|  |  |
| --- | --- |
| **Parametri** | order |
| **Oggetti dell’ambiente** | Database |
| **Categorie** | **Scelte** |
| Order | OR1: order = “categoria”  OR2: order = “nome”  OR3: order = null  OR4: order = “”  OR5: order non “categoria” e non “nome” |

**Test Frame**

|  |  |
| --- | --- |
| **Combinazione** | **Oracolo** |
| OR1 | Restituisce un ArrayList<Prodotto> di i prodotti presenti nel database in ordine di categoria |
| OR2 | Restituisce un ArrayList<Prodotto> di i prodotti presenti nel database in ordine di prezzo (crescente) |
| OR3 | Restituisce un ArrayList<Prodotto> di i prodotti presenti nel database in ordine di idProdotto (crescente) |
| OR4 | Restituisce un ArrayList<Prodotto> di i prodotti presenti nel database in ordine di idProdotto (crescente) |
| OR5 | Lancia una CheckException |

**TCU3\_1\_4 doDelete(Product product):**

|  |  |
| --- | --- |
| **Parametri** | product |
| **Oggetti dell’ambiente** | Database |
| **Categorie** | **Scelte** |
| **product** | P1: prodotto non vuoto  P2: prodotto vuoto |
| **product.idProdotto** | IPF1: formato valido [property: id\_corretto]  IPF2: formato non valido  IP1: idProdotto già presente nel DB [if id\_corretto]  IP2: idProdotto non presente nel DB [if id\_corretto] |

**Test Frame**

|  |  |
| --- | --- |
| **Combinazione** | **Oracolo** |
| P1-IPF1-IP1 | Il prodotto viene rimosso dal DB |
| P1-IPF2-IP1 | Lancia una CheckException |
| P1-IPF1-IP2 | Non viene effettuata alcuna modifica alle tuple esistenti sul db |
| P2 | Lancia una CheckException |

**TCU3\_1\_5 doUpdate(Product product):**

|  |  |
| --- | --- |
| **Parametri** | product |
| **Oggetti dell’ambiente** | Database |
| **Categorie** | **Scelte** |
| **product** | PR1: prodotto non vuoto  PR2: prodotto vuoto |
| **product.id** | ID1: formato valido [property: id\_corretto]  ID2: formato non valido  IP1: idProdotto presente nel DB [if id\_corretto]  IP2: idProdotto non presente nel DB [if id\_corretto] |
| **product.nome** | NP1: formato valido  NP2: formato non valido |
| **product.categoria** | CAT1: formato valido  CAT2: formato non valido |
| **product.prezzo** | P1: prezzo>=0  P2: prezzo<0 |
| **product.descrizione** | DP1: formato valido  DP2: formato valido |
| **product.image** | IMG1: formato valido  IMG: formato non valido |
| **product.statistiche** | ST1: formato valido  ST2: formato non valido |
| **product.taglia** | TF1: formato valido [property taglie\_corrette]  TF2: formato non valido  TP1: Taglie.S +Taglie.M + Taglie.L + Taglie.XL >=0 [if taglie\_corrette]  TP2: Taglie.S +Taglie.M + Taglie.L + Taglie.XL<0 [if taglie\_corrette] |

**Test Frame**

|  |  |
| --- | --- |
| **Combinazione** | **Oracolo** |
| PR1-ID1-IP1-NP1-CAT1-P1-DP1-IMG1-ST1-TF1-TP1 | Salva le modifiche del prodotto nel DB |
| PR1-ID2-IP1-NP1-CAT1-P1-DP1-IMG1-ST1-TF1-TP1 | Lancia una CheckException |
| PR1-ID1-IP2-NP1-CAT1-P1-DP1-IMG1-ST1-TF1-TP1 | Non viene effettuata alcuna modifica alle tuple esistenti sul db |
| PR1-ID1-IP1-NP2-CAT1-P1-DP1-IMG1-ST1-TF1-TP1 | Lancia una CheckException |
| PR1-ID1-IP1-NP1-CAT2-P1-DP1-IMG1-ST1-TF1-TP1 | Lancia una CheckException |
| PR1-ID1-IP1-NP1-CAT1-P2-DP1-IMG1-ST1-TF1-TP1 | Lancia una CheckException |
| PR1-ID1-IP1-NP1-CAT1-P1-DP2-IMG1-ST1-TF1-TP1 | Lancia una CheckException |
| PR1-ID1-IP1-NP1-CAT1-P1-DP1-IMG2-ST1-TF1-TP1 | Lancia una CheckException |
| PR1-ID1-IP1-NP1-CAT1-P1-DP1-IMG1-ST2-TF1-TP1 | Lancia una CheckException |
| PR1-ID1-IP1-NP1-CAT1-P1-DP1-IMG1-ST1-TF2-TP1 | Lancia una CheckException |
| PR1-ID1-IP1-NP1-CAT1-P1-DP1-IMG1-ST1-TF1-TP2 | Lancia una CheckException |
| PR2 | Lancia una CheckException |

**TCU3\_1\_6 setTaglieByKey(Taglie taglie):**

|  |  |
| --- | --- |
| **Parametri** | taglie |
| **Oggetti dell’ambiente** | Database |
| **Categorie** | **Scelte** |
| taglie | PR1: taglie non null  PR2: taglie null |
| taglie.id | ID1: formato valido [property: id\_corretto]  ID2: formato non valido  IP1: idProdotto presente nel DB [if id\_corretto]  IP2: idProdotto non presente nel DB [if id\_corretto] |
| taglie.tagliaM | TM1: tagliaM >= 0 [property taglie\_corrette]  TM2: tagliaM < 0 |
| taglie.tagliaL | TL1: tagliaL >= 0 [property taglie\_corrette]  TL2: tagliaL < 0 |
| taglie.tagliaXL | TXL1: tagliaXL >= 0 [property taglie\_corrette]  TXL2: tagliaXL < 0 |
| taglie.tagliaXXL | TXXL1: tagliaXXL >= 0 [property taglie\_corrette]  TXXL2: taglia XXL < |

**Test Frame**

|  |  |
| --- | --- |
| **Combinazione** | **Oracolo** |
| PR1-ID1-IP1-TM1-TL1-TXL1-TXXL1 | Le taglie del prodotto vengono aggiornate |
| PR1-ID2-IP1-TM1-TL1-TXL1-TXXL1 | Lancia una CheckException |
| PR1-ID1-IP2-TM1-TL1-TXL1-TXXL1 | Non vengono effettuate modifiche |
| PR1-ID1-IP1-TM2-TL1-TXL1-TXXL1 | Lancia una CheckException |
| PR1-ID1-IP1-TM1-TL2-TXL1-TXXL1 | Lancia una CheckException |
| PR1-ID1-IP1-TM1-TL1-TXL2-TXXL1 | Lancia una CheckException |
| PR1-ID1-IP1-TM1-TL1-TXL1-TXXL2 | Lancia una CheckException |
| PR2 | Lancia una CheckException |

**TCU3\_1\_7 getTaglieByKey(Product product):**

|  |  |
| --- | --- |
| **Parametri** | product |
| **Oggetti dell’ambiente** | Database |
| **Categorie** | **Scelte** |
| product | PR1: prodotto non vuoto  PR2: prodotto vuoto |
| product.id | ID1: formato valido [property: id\_corretto]  ID2: formato non valido  IP1: idProdotto presente nel DB [if id\_corretto]  IP2: idProdotto non presente nel DB [if id\_corretto] |

**Test Frame**

|  |  |
| --- | --- |
| **Combinazione** | **Oracolo** |
| PR1-ID1-IP1 | Restituisce un ArrayList<Taglia> di taglie presenti nel database per il prodotto specificato |
| PR1-ID2-IP1 | Lancia una CheckException |
| PR1-ID1-IP2 | Restituisce null |
| PR2 | Lancia una CheckException |

## 4.3 TCU3\_1: PhotoUnitDaoData

**TCU3\_2\_1 updatePhoto(Product product):**

|  |  |
| --- | --- |
| **Parametri** | product |
| **Oggetti dell’ambiente** | Database |
| **Categorie** | **Scelte** |
| product | PR1: product non vuoto  PR2: product vuoto |
| Product.id | ID1: formato valido [property formato\_corretto]  ID2: formato non valido  IS1: id presente nel DB [if formato\_corretto]  IS2: id non presente nel DB [if formato\_corretto] |
| product.imagePath | IP1: formato valido  IP2: formato non valido |

**Test Frame**

|  |  |
| --- | --- |
| **Combinazione** | **Oracolo** |
| PR1-ID1-IS1-IP1 | Modifica l’attributo “imagePath” di una tupla prodotto arbitraria |
| PR1-ID2-IS1-IP1 | Lancia una CheckException |
| PR1-ID1-IS2-IP1 | Non viene effettuata alcuna modifica alle tuple esistenti sul db |
| PR1-ID1-IS1-IP2 | Lancia una CheckException |
| PR2 | Lancia una CheckException |

## TCU4\_1: OrdiniUnitDaoData

**TCU4\_1\_1 doSaveOrdine(Order order):**

|  |  |
| --- | --- |
| **Parametri** | order |
| **Oggetti dell’ambiente** | Database |
| **Categorie** | **Scelte** |
| order | OS1: order non vuoto  OS2: order vuoto |
| order.idUtente | OM1: formato valido  OM2: formato non valido |
| order.idOrdine | IO1: formato valido [property: ordine\_corretto]  IO2: formato non valido  IS1: id non presente nel db [if ordine\_corretto]  IS2: id presente nel db [if ordine\_corretto] |
| order.data | ID1: formato valido  ID2: formato non valido |
| order.stato | SO1: formato valido  SO2: formato non valido |
| order.prezzo | PO1: formato valido  PO2: formato non valido |

**Test Frame**

|  |  |
| --- | --- |
| **Combinazione** | **Oracolo** |
| OS1-OM1-IO1-IS1-ID1-SO1-PO1 | Inserisce un nuovo ordine nel db |
| OS1-OM2-IO1-IS1-ID1-SO1-PO1 | Lancia una CheckException |
| OS1-OM2-IO2-IS1-ID1-SO1-PO1 | Lancia una CheckException |
| OS1-OM2-IO1-IS2-ID1-SO1-PO1 | Lancia una SQLIntegrityException |
| OS1-OM2-IO1-IS1-ID2-SO1-PO1 | Lancia una CheckException |
| OS1-OM2-IO1-IS1-ID1-SO2-PO1 | Lancia una CheckException |
| OS1-OM2-IO1-IS1-ID1-SO1-PO2 | Lancia una CheckException |
| OS2 | Lancia una CheckException |

**TCU4\_1\_2 changeStatoOrdine(Order order):**

|  |  |
| --- | --- |
| **Parametri** | order |
| **Oggetti dell’ambiente** | Database |
| **Categorie** | **Scelte** |
| order | OS1: order non vuoto  OS2: order vuoto |
| order.idOrdine | IO1: formato valido [property: ordine\_corretto]  IO2: formato non valido  IS1: id presente nel db [if ordine\_corretto]  IS2: id non presente nel db [if ordine\_corretto] |
| order.stato | SO1: formato valido  SO2: formato non valido |

**Test Frame**

|  |  |
| --- | --- |
| **Combinazione** | **Oracolo** |
| OS1-IO1-IS1-SO1 | Cambia lo stato di un ordine nel db |
| OS1-IO2-IS1-SO1 | Lancia una CheckException |
| OS1-IO1-IS2-SO1 | Non viene effettuata alcuna modifica alle tuple esistenti sul db |
| OS1-IO1-IS1-SO2 | Lancia una CheckException |

**TCU4\_1\_3 doRetrieveAllOrders ():**

|  |  |
| --- | --- |
| **Parametri** | order |
| **Oggetti dell’ambiente** | Database |
| **Categorie** | **Scelte** |
| order | OR1: order = null or order = “” |

**Test Frame**

|  |  |
| --- | --- |
| **Combinazione** | **Oracolo** |
| OR1 | Restituisce un ArrayList<Ordine> di ordini presenti nel database in ordine crescente per id |

**TCU4\_1\_4 doRetrieveOrderById (Order order):**

|  |  |
| --- | --- |
| **Parametri** | order |
| **Oggetti dell’ambiente** | Database |
| **Categorie** | **Scelte** |
| order | ON1: order non vuoto  ON2: order vuoto |
| order.idOrdine | OI1: formato valido [property formato\_valido]  OI2: formato non valido  IS1: id presente nel db [if formato\_valido]  IS2: id non presente nel db [if formato\_valido] |

**Test Frame**

|  |  |
| --- | --- |
| **Combinazione** | **Oracolo** |
| ON1-OI1-IS1 | Restituisce una tupla Ordine con idOrdine corrispondente a quello cercato |
| ON1-OI2-IS1 | Lancia una CheckException |
| ON1-OI1-IS2 | Lancia una CheckException |
| ON2 | Lancia una CheckException |

## TCU5\_1: CarrelloUnitDaoData

**TCU5\_1\_1 cancellaCarrello(Carrello carrello):**

|  |  |
| --- | --- |
| **Parametri** | carrello |
| **Oggetti dell’ambiente** | Database |
| **Categorie** | **Scelte** |
| carrello | CS1: carrello non vuoto  CS2: carrello vuoto |
| Carrello.idcarrello | CF1: formato valido [property id\_corretto]  CF2: formato non valido  CV1: id già presente [if id\_corretto]  CV2: id non presente [if id\_corretto] |

**Test Frame**

|  |  |
| --- | --- |
| **Combinazione** | **Oracolo** |
| CS1-CF1-CV1 | Cancella il carrello dal database |
| CS1-CF2-CV1 | Lancia una CheckException |
| CS1-CF1-CV2 | Non viene effettuata alcuna modifica alle tuple esistenti sul db |
| CS2 | Lancia una CheckException |

**TCU5\_1\_2 salvaCarrello(Carrello carrello):**

|  |  |
| --- | --- |
| **Parametri** | carrello |
| **Oggetti dell’ambiente** | Database |
| **Categorie** | **Scelte** |
| carrello | CS1: carrello non vuota  CS2: carrello vuota |
| carrello.idcarrello | CF1: formato valido [property id\_corretto]  CF2: formato non valido  CV1: id non presente [if id\_corretto]  CV2: id già presente [if id\_corretto] |
| carrello.idprodottoc | CP1: formato valido  CP2: formato non valido |

**Test Frame**

|  |  |
| --- | --- |
| **Combinazione** | **Oracolo** |
| CS1-CF1-CV1-CP1 | Salva il carrello nel DB |
| CS1-CF2-CV1-CP1 | Lancia una CheckException |
| CS1-CF1-CV2-CP1 | Viene sovrascritto un carrello già esistente |
| CS1-CF1-CV1-CP2 | Lancia una CheckException |
| CS2 | Lancia una CheckException |

**TCU5\_1\_3 recuperaCarrello(Utente utente):**

|  |  |
| --- | --- |
| **Parametri** | utente |
| **Oggetti dell’ambiente** | Database |
| **Categorie** | **Scelte** |
| utente | US1: utente non vuoto  US2: utente vuoto |
| utente.idUtente | CF1: formato valido [property id\_corretto]  CF2: formato non valido  CV1: id non presente [if id\_corretto]  CV2: id già presente [if id\_corretto] |

**Test Frame**

|  |  |
| --- | --- |
| **Combinazione** | **Oracolo** |
| US1-CF1-CV2 | Recupera il carrello dal DB |
| US1-CF2-CV2 | Lancia una CheckException |
| US1-CF1-CV1 | Restituisce un carrello vuoto |
| US2 | Lancia una CheckException |