**SE 318**

**SOFTWARE VERIFICATION AND VALIDATION**

**SPRING 2020**

*bus management*

*ısmaıl semıh pehlıvan*

*yağız gültekin ünal*

*ziyad ege uçar*

*orhan eryiğit*

UNIT TEST DOCUMENT

Version *<3.0>*

*<05/21/2020>*

VERSION HISTORY

|  |
| --- |
| **VERVERSION 1.0 (date)** |
| -Select seat  -Select Destination  -Buy ticket  -Ticket details |
| **VERVERSION 2.0 (date)** |
| -10 test cases added  -Database tables created  -Database connection |
| **VERVERSION 3.0 (date)** |
| -Login  -Register  -Select which type account to forward  -Discount for subscribers |

# INTRODUCTION

## PURPOSE OF THE TEST CASE DOCUMENT

Choose seat. Choose time. Users database. Login screen. Register account. One account cannot be associated with multiple users.

## CONSTRAINTS

JAVA-JUNIT5

# UNIT TEST FRAMEWORK: *<THE NAME OF THE FRAMEWORK>*

JUnit 5 is most widely used testing framework for java applications. For very long time, JUnit has been doing its job perfectly. In between, JDK 8 brought very exciting features in java and most notably lambda expressions. JUnit 5 aims to adapt java 8 style of coding and several other features as well, that’s why java 8 is required to create and execute tests in JUnit 5 (though it is possible to execute tests written with JUnit 3 or JUnit 4 for backward compatibility).

# TEST CASES

|  |  |
| --- | --- |
| **Test Case 1** | |
| **Test Definition** | |
| **Checking if the destination is same** | |
| **Input Value** | |
| Destination | |
| **Expected Value** | **Actual Value** |
| 2,”izmir”,”ankara”,150.0,475,null | 1,”ankara”,”izmir”,150.0,475,null |
| **Result of Test Case** | ***Successful*** |
| **Test Script** |  |
| Destinations dest1 = new Destinations(1,"ankara","izmir",150.0,475,null);  Destinations dest2 = new Destinations(2,"izmir","ankara",150.0,475,null);  *assertNotSame*(dest1,dest2); |  |

|  |  |
| --- | --- |
| **Test Case 2** | |
| **Test Definition** | |
| **Checks that if the booked seat is effect the overall seat count** | |
| **Input Value** | |
| Destination | |
| **Expected Value** | **Actual Value** |
| arraylist | arraylist |
| **Result of Test Case** | ***Successful*** |
| **Test Script** |  |
| ArrayList<Integer> seats = new ArrayList<>();  for(int i=0;i<40;i++){  seats.add(i);  }  Destinations dest1 = new Destinations();  dest1.setSeat(new ArrayList<>(seats));  ArrayList<Integer> dest1seats = dest1.getSeat();  Destinations dest2 = new Destinations();  dest2.setSeat(new ArrayList<>(seats));  ArrayList<Integer> dest2seats = dest2.getSeat();  dest2seats.remove(1);  int destseatsize = dest1seats.size();  int destseat2size = dest2seats.size();  *assertNotEquals*(destseatsize,destseat2size); |  |
| **Test Case 3** | |
| **Test Definition** | |
| Checks that if the id of Destination is valid | |
| **Input Value** | |
| **id** | |
| **Expected Value** | **Actual Value** |
| **id** | **id** |
| **Result of Test Case** | ***Successful*** |
| **Test Script** |  |
| Destinations dest = new Destinations();  dest.setId(1);  int result =dest.getId();  *assertNotEquals*(1.0,result); |  |

|  |  |
| --- | --- |
| **Test Case 4** | |
| **Test Definition** | |
| **Checks if the where value is valid** | |
| **Input Value** | |
| **String** | |
| **Expected Value** | **Actual Value** |
| **String** | **String** |
| **Result of Test Case** | ***Successful*** |
| **Test Script** |  |
| Destinations dest = new Destinations();  dest.setWhere("ANKARA");  String where = dest.getWhere();  *assertEquals*("ANKARA",where); |  |
| **Test Case 5** | |
| **Test Definition** | |
| **Checking the two destinations are the same** | |
| **Input Value** | |
| **Destinations** | |
| **Expected Value** | **Actual Value** |
| **Destinations** | **Destinations** |
| **Result of Test Case** | ***Successful*** |
| **Test Script** |  |
| Destinations dest1 = new Destinations(4,"ankara","istanbul",190.0,420,null);  Seat seat1 = new Seat();  Destinations dest2 = new Destinations();  Seat seat2 = new Seat();  seat1.setDestination(dest1);  seat2.setDestination(dest2);  Destinations route1=seat1.getDestination();  Destinations route2 = seat2.getDestination();  *assertNotSame*(route1,route2); |  |

|  |  |
| --- | --- |
| **Test Case 6** | |
| **Test Definition** | |
| **Checking the price type is the same** | |
| **Input Value** | |
| **price** | |
| **Expected Value** | **Actual Value** |
| 150 | 150.0 |
| **Result of Test Case** | ***Successful*** |
| **Test Script** |  |
| Seat seat = new Seat();  seat.setPrice(150.0);  double price =seat.getPrice();  *assertNotEquals*(150,150.0); |  |
| **Test Case 7** | |
| **Test Definition** | |
| **Checks if the seat is valid** | |
| **Input Value** | |
| **Seat** | |
| **Expected Value** | **Actual Value** |
| Seat | Seat |
| **Result of Test Case** | ***<successful OR fail>*** |
| **Test Script** |  |
| Seat seat = new Seat();  *assertNotNull*(seat); |  |

|  |  |
| --- | --- |
| **Test Case 8** | |
| **Test Definition** | |
| Checking the property to is the same | |
| **Input Value** | |
| **Antalya** | |
| **Expected Value** | **Actual Value** |
| **Antalya** | **antalya** |
| **Result of Test Case** | ***Successful*** |
| **Test Script** |  |
| Destinations dest = new Destinations();  dest.setTo("ANTALYA");  String to = dest.getTo();  *assertNotEquals*("antalya",to); |  |
| **Test Case 9** | |
| **Test Definition** | |
| **Checking price types are the same or not** | |
| **Input Value** | |
| **price** | |
| **Expected Value** | **Actual Value** |
| price | price |
| **Result of Test Case** | ***Successful*** |
| **Test Script** |  |
| Destinations dest = new Destinations();  dest.setPrice(122.0);  Double price = dest.getPrice();  *assertNotEquals*(122,price); |  |

|  |  |
| --- | --- |
| **Test Case 10** | |
| **Test Definition** | |
| **Checking the duration is same or not** | |
| **Input Value** | |
| **int** | |
| **Expected Value** | **Actual Value** |
| int | int |
| **Result of Test Case** | ***Successful*** |
| **Test Script** |  |
| Destinations dest = new Destinations();  dest.setDuration(180);  int duration = dest.getDuration();  *assertNotEquals*(180.0,duration); |  |
| **Test Case 11** | |
| **Test Definition** | |
| Checks if the user id is valid | |
| **Input Value** | |
| **int** | |
| **Expected Value** | **Actual Value** |
| **int** | **int** |
| **Result of Test Case** | ***Successful*** |
| **Test Script** |  |
| User usr1 = new User();  usr1.setId(123);  int finalid = usr1.getId();  *assertEquals*(123,finalid); |  |

|  |  |
| --- | --- |
| **Test Case 12** | |
| **Test Definition** | |
| Checks if the user password is valid | |
| **Input Value** | |
| **String** | |
| **Expected Value** | **Actual Value** |
| **String** | **String** |
| **Result of Test Case** | ***Successful*** |
| **Test Script** |  |
| User usr1 = new User();  usr1.setPassword("123");  String finalpw = usr1.getPassword();  *assertEquals*("123",finalpw); |  |
| **Test Case 13** | |
| **Test Definition** | |
| Checks if the vip seat has higher price. | |
| **Input Value** | |
| **Seat** | |
| **Expected Value** | **Actual Value** |
| **Seat** | **Seat** |
| **Result of Test Case** | ***Successful*** |
| **Test Script** |  |
| Seat seat1= new Seat(1,8,"vip",new Destinations(1,"ankara","izmir",150.0,475,null));  double x = seat1.getPrice();  *assertEquals*(x, 187.5,0); |  |

|  |  |
| --- | --- |
| **Test Case 14** | |
| **Test Definition** | |
| Checks if the normal seat price is valid. | |
| **Input Value** | |
| **Seat** | |
| **Expected Value** | **Actual Value** |
| **Seat** | **Seat** |
| **Result of Test Case** | ***Successful*** |
| **Test Script** |  |
| Seat seat1= new Seat(5,10,"regular",new Destinations(1,"antalya","izmir",220.0,360,null));  double price = seat1.getPrice();  *assertEquals*(price,220,0); |  |
| **Test Case 15** | |
| **Test Definition** | |
| **Checks if the subscriber’s price is discounted.** | |
| **Input Value** | |
| **Destination.price** | |
| **Expected Value** | **Actual Value** |
| **Destination.price** | **Destination.price** |
| **Result of Test Case** | ***Successful*** |
| **Test Script** |  |
| Destinations dest = new Destinations(0,"ankara","izmir",150.0,130,new ArrayList<>());  User usr = new User(0,null,null);  usr.setDestinations(dest);  Double usrprice = usr.getDestinations().getPrice();  *assertEquals*(usrprice,112.5,0); |  |

|  |  |
| --- | --- |
| **Test Case 16** | |
| **Test Definition** | |
| **Checks if the regular and vip seat prices are not the same.** | |
| **Input Value** | |
| **Seat** | |
| **Expected Value** | **Actual Value** |
| **Seat** | **Seat** |
| **Result of Test Case** | ***Successful*** |
| **Test Script** |  |
| Seat seat1= new Seat(5,10,"regular",new Destinations(1,"antalya","izmir",220.0,360,null));  Seat seat2= new Seat(1,8,"vip",new Destinations(1,"ankara","izmir",150.0,475,null));  double regprice = seat1.getPrice();  double x = seat2.getPrice();  *assertNotEquals*(x,regprice); |  |
| **Test Case 17** | |
| **Test Definition** | |
| Checks if the user discount applied to vip seat. | |
| **Input Value** | |
| **Destination.price** | |
| **Expected Value** | **Actual Value** |
| **Destination.price** | **Destination.price** |
| **Result of Test Case** | ***Successful*** |
| **Test Script** |  |
| Destinations dest = new Destinations(1,"ankara","izmir",150.0,475,new ArrayList<>());  User usr = new User(0,null,null);  usr.setDestinations(dest);  Double usrprice = usr.getDestinations().getPrice();  Seat seat2= new Seat(1,8,"vip",new Destinations(1,"ankara","izmir",150.0,475,null));  double x = seat2.getPrice();  *assertNotEquals*(x,usrprice); |  |

|  |  |
| --- | --- |
| **Test Case 18** | |
| **Test Definition** | |
| Checks if the user discount applied to regular seat. | |
| **Input Value** | |
| **User.destinations.price** | |
| **Expected Value** | **Actual Value** |
| **price** | **price** |
| **Result of Test Case** | ***Successful*** |
| **Test Script** |  |
| User usr = new User();  Destinations dest = new Destinations(5,"antalya","izmir",220.0,360,null);  usr.setDestinations(dest);  double price = usr.getDestinations().getPrice();  Seat seat1= new Seat(5,10,"regular",new Destinations(5,"antalya","izmir",220.0,360,null));  double regprice = seat1.getPrice();  *assertNotEquals*(regprice,price); |  |
| **Test Case 19** | |
| **Test Definition** | |
| **Checks if the user provide correct type of id** | |
| **Input Value** | |
| **String** | |
| **Expected Value** | **Actual Value** |
| **int** | **String** |
| **Result of Test Case** | ***Successful*** |
| **Test Script** |  |
| User usr =new User();  usr.setId(123123);  int usrid = usr.getId();  String id1 ="123123";  *assertNotEquals*(usrid,id1); |  |

|  |  |
| --- | --- |
| **Test Case 20** | |
| **Test Definition** | |
| Checks if the user provide correct type of password | |
| **Input Value** | |
| **String** | |
| **Expected Value** | **Actual Value** |
| **int** | **int** |
| **Result of Test Case** | ***Successful*** |
| **Test Script** |  |
| User usr = new User();  usr.setPassword("123");  String pw=usr.getPassword();  int pw1 = 123;  *assertNotEquals*(pw,pw1); |  |
| **Test Case 21** | |
| **Test Definition** | |
| Checks if the vip seat has a higher price | |
| **Input Value** | |
| **Seat** | |
| **Expected Value** | **Actual Value** |
| Seat | Seat |
| **Result of Test Case** | ***Successful*** |
| **Test Script** |  |
| **Seat seat1= new Seat(1,8,"vip",new Destinations(1,"ankara","izmir",150.0,475,null)); double x = seat1.getPrice(); assertNotEquals(x, 177.5,0);** |  |

|  |  |
| --- | --- |
| **Test Case 22** | |
| **Test Definition** | |
| Checks if the regular price is valid | |
| **Input Value** | |
| **Seat** | |
| **Expected Value** | **Actual Value** |
| **Seat** | **Seat** |
| **Result of Test Case** | ***Successful*** |
| **Test Script** |  |
| **Seat seat1= new Seat(5,10,"regular",new Destinations(1,"antalya","izmir",220.0,360,null));**  **double price = seat1.getPrice();**  **assertEquals(price,220,0);** |  |
| **Test Case 23** | |
| **Test Definition** | |
| Checks if the regular price is not valid | |
| **Input Value** | |
| Destination.seat.price | |
| **Expected Value** | **Actual Value** |
| Destination.seat.price | Destination.seat.price |
| **Result of Test Case** | ***Successful*** |
| **Test Script** |  |
| **Seat seat1= new Seat(5,10,"regular",new Destinations(1,"antalya","izmir",220.0,360,null));**  **double price = seat1.getPrice(); assertNotEquals(price,230,0);** |  |

|  |  |
| --- | --- |
| **Test Case 24** | |
| **Test Definition** | |
| Checks if the registered users price is valid | |
| **Input Value** | |
| **Destination.price** | |
| **Expected Value** | **Actual Value** |
| **Destination.price** | **Destination.price** |
| **Result of Test Case** | ***<successful OR fail>*** |
| **Test Script** |  |
| **Destinations dest = new Destinations(0,"ankara","izmir",150.0,130,new ArrayList<>());**  **User usr = new User(0,null,null);**  **usr.setDestinations(dest);**  **Double usrprice = usr.getDestinations().getPrice();**  **assertEquals(usrprice,112.5,0);** |  |
| **Test Case 25** | |
| **Test Definition** | |
| **Checks if the registeration price is not valid** | |
| **Input Value** | |
| **Destination.price** | |
| **Expected Value** | **Actual Value** |
| **Destination.price** | **Destination.price** |
| **Result of Test Case** | ***Successful*** |
| **Test Script** |  |
| **Destinations dest = new Destinations(0,"ankara","izmir",150.0,130,new ArrayList<>());**  **User usr = new User(0,null,null);**  **usr.setDestinations(dest);**  **Double usrprice = usr.getDestinations().getPrice();**  **assertNotEquals(usrprice,112.0,0);** |  |

|  |  |
| --- | --- |
| **Test Case 26** | |
| **Test Definition** | |
| Compare vip and regular seat prices. | |
| **Input Value** | |
| **Seat** | |
| **Expected Value** | **Actual Value** |
| **Seat** | **Seat** |
| **Result of Test Case** | ***Succesful*** |
| **Test Script** |  |
| **Seat seat1= new Seat(5,10,"regular",new Destinations(1,"ankara","izmir",150.0,475,null));**  **double regprice = seat1.getPrice();**  **Seat seat2= new Seat(1,8,"vip",new Destinations(1,"ankara","izmir",150.0,475,null));**    **double x = seat2.getPrice();**  **assertNotEquals(x,regprice);** |  |
| **Test Case 27** | |
| **Test Definition** | |
| **Checks if the subscribers vip seat and regular seat is the same price.** | |
| **Input Value** | |
| **Seat** | |
| **Expected Value** | **Actual Value** |
| **Seat** | **Seat** |
| **Result of Test Case** | ***Successful*** |
| **Test Script** |  |
| **Destinations dest = new Destinations(1,"ankara","izmir",150.0,475,new ArrayList<>());**  **User usr = new User(0,null,null);**  **usr.setDestinations(dest);**    **Double usrprice = usr.getDestinations().getPrice();**  **Seat seat2= new Seat(1,8,"vip",new Destinations(1,"ankara","izmir",150.0,475,null));**  **double x = seat2.getPrice();**    **assertNotEquals(x,usrprice);** |  |

|  |  |
| --- | --- |
| **Test Case 28** | |
| **Test Definition** | |
| Control if subscriber has a regular or vip seat | |
| **Input Value** | |
| **user.destination.seat** | |
| **Expected Value** | **Actual Value** |
| **user.destination.seat** | **user.destination.seat** |
| **Result of Test Case** | ***Successful*** |
| **Test Script** |  |
| **User usr = new User();**  **Destinations dest = new Destinations(5,"antalya","izmir",220.0,360,null);**  **usr.setDestinations(dest);**  **double price = usr.getDestinations().getPrice();**  **Seat seat1= new Seat(5,10,"regular",new Destinations(5,"antalya","izmir",220.0,360,null));**  **double regprice = seat1.getPrice();**  **assertNotEquals(regprice,price);** |  |
| **Test Case 29** | |
| **Test Definition** | |
| **Checks if the user has proper id** | |
| **Input Value** | |
| **Integer** | |
| **Expected Value** | **Actual Value** |
| **Integer** | **String** |
| **Result of Test Case** | ***<successful OR fail>*** |
| **Test Script** |  |
| **User usr = new User();**  **usr.setId(123123);**  **int usrid = usr.getId();**  **String id1= "123";**  **assertNotEquals(usrid,id1);** |  |

|  |  |
| --- | --- |
| **Test Case 30** | |
| **Test Definition** | |
| **Checks if the user has proper password** | |
| **Input Value** | |
| **String** | |
| **Expected Value** | **Actual Value** |
| **String** | **int** |
| **Result of Test Case** | ***Successful*** |
| **Test Script** |  |
| **User usrpw = new User();**  **usrpw.setPassword("asd123");**  **String usrpw1 = usrpw.getPassword();**  **Double pw1= 150.0;**  **assertNotEquals(usrpw1,pw1);** |  |

**4. CONCLUSION**