

Assignment 3 Part 1

CSE2115: Software Engineering Methods

Group 06b

Assignment 3 Part 1

by

Group 06b

Instructor:	Dr. A. Panichella & F. Mulder
Teaching Assistant:	M. Segers
Institution:	Delft University of Technology
Place:	Faculty of Electrical Engineering, Mathematics and Computer Science
Date:	20-01-2023

Contents

1 Automated Mutation Testing	1
List of Figures	9

1

Automated Mutation Testing

Class Name - OrderCouponImpl

Before the changes, this class had **0%** mutation coverage. After the changes, it has **70%** mutation coverage.

Commit link: fcc856a020939f500c39f2e737ec5e0f18f43f0a

Mutation Test Coverage Report Before:

Pit Test Coverage Report

Package Summary

nl.tudelft.sem.group06b.order.service.coupon

Number of Classes	Line Coverage	Mutation Coverage	Test Strength
1	68% <div><div>27/40</div></div>	0% <div><div>0/10</div></div>	0% <div><div>0/7</div></div>

Breakdown by Class

Name	Line Coverage	Mutation Coverage	Test Strength
OrderCouponImpl.java	68% <div><div>27/40</div></div>	0% <div><div>0/10</div></div>	0% <div><div>0/7</div></div>

Figure 1.1: OrderCouponImpl mutation score before refactoring

Mutation Test Coverage Report After:

Pit Test Coverage Report

Package Summary

nl.tudelft.sem.group06b.order.service.coupon

Number of Classes	Line Coverage	Mutation Coverage	Test Strength
1	68% <div><div>27/40</div></div>	70% <div><div>7/10</div></div>	100% <div><div>7/7</div></div>

Breakdown by Class

Name	Line Coverage	Mutation Coverage	Test Strength
OrderCouponImpl.java	68% <div><div>27/40</div></div>	70% <div><div>7/10</div></div>	100% <div><div>7/7</div></div>

Figure 1.2: OrderCouponImpl mutation score after refactoring

Test additions/edits code snippets:

```

assertThrows(Exception.class, () -> {
    orderCoupon.addCoupon( token: null, orderId: 11L, coupon);
});
assertThrows(Exception.class, () -> {
    orderCoupon.addCoupon(token, orderId: 12L, coupon);
});
assertThrows(Exception.class, () -> {
    orderCoupon.addCoupon(token, orderId: 11L, coupon: null);
});
assertThrows(Exception.class, () -> {
    orderCoupon.addCoupon(token, orderId: 11L, coupon);
});
assertThrows(Exception.class, () -> {
    orderCoupon.addCoupon(token, orderId: null, coupon);
});

order1.setStatus(Status.ORDER_PLACED);
assertThrows(Exception.class, () -> {
    orderCoupon.addCoupon( token: "token", orderId: 11L, coupon: "extr10");
});

when(couponCommunication.validateCoupon(coupon, token)).thenReturn(false);
assertThrows(Exception.class, () -> {
    orderCoupon.addCoupon(token, orderId: 11L, coupon);
});

```

Figure 1.3: OrderCouponImpl code before refactoring

```

Exception exception1 = assertThrows(Exception.class, () -> {
    orderCoupon.addCoupon( token: null, orderId: 11L, coupon);
});
assertEquals( expected: "Invalid token", exception1.getMessage());

Exception exception2 = assertThrows(Exception.class, () -> {
    orderCoupon.addCoupon(token, orderId: 12L, coupon);
});

Exception exception3 = assertThrows(Exception.class, () -> {
    orderCoupon.addCoupon(token, orderId: 11L, coupon: null);
});
assertEquals( expected: "Please enter valid coupon", exception3.getMessage());

Exception exception4 = assertThrows(Exception.class, () -> {
    orderCoupon.addCoupon(token, orderId: 11L, coupon);
});

Exception exception5 = assertThrows(Exception.class, () -> {
    orderCoupon.addCoupon(token, orderId: null, coupon);
});
assertEquals( expected: "Invalid order ID", exception5.getMessage());

order1.setStatus(Status.ORDER_PLACED);
Exception exception6 = assertThrows(Exception.class, () -> {
    orderCoupon.addCoupon( token: "token", orderId: 11L, coupon: "extr10");
});
assertEquals( expected: "No active order with this ID", exception6.getMessage());

when(couponCommunication.validateCoupon(coupon, token)).thenReturn(false);

```

Figure 1.4: OrderCouponImpl code after refactoring

Class Name - MenuPizzaService

Before the changes, this class had **39%** mutation coverage. After the changes, it has **59%** mutation coverage.

Commit link: 7decbc23938e69cfa6606f6a82fa539cf8e96d37

Mutation Test Coverage Report Before:

Pit Test Coverage Report

Package Summary

nl.tudelft.sem.group06b.menu.service

Number of Classes	Line Coverage	Mutation Coverage	Test Strength
3	73% <div><div>157/214</div></div>	46% <div><div>48/105</div></div>	70% <div><div>48/69</div></div>

Breakdown by Class

Name	Line Coverage	Mutation Coverage	Test Strength
MenuAllergyService.java	68% <div><div>59/87</div></div>	43% <div><div>16/37</div></div>	73% <div><div>16/22</div></div>
MenuPizzaService.java	72% <div><div>59/82</div></div>	39% <div><div>17/44</div></div>	61% <div><div>17/28</div></div>
MenuToppingService.java	87% <div><div>39/45</div></div>	63% <div><div>15/24</div></div>	79% <div><div>15/19</div></div>

Figure 1.5: MenuPizzaService mutation score before refactoring

Mutation Test Coverage Report After:

Pit Test Coverage Report

Package Summary

nl.tudelft.sem.group06b.menu.service

Number of Classes	Line Coverage	Mutation Coverage	Test Strength
3	80% <div><div>171/214</div></div>	54% <div><div>57/105</div></div>	72% <div><div>57/79</div></div>

Breakdown by Class

Name	Line Coverage	Mutation Coverage	Test Strength
MenuAllergyService.java	68% <div><div>59/87</div></div>	43% <div><div>16/37</div></div>	73% <div><div>16/22</div></div>
MenuPizzaService.java	89% <div><div>73/82</div></div>	59% <div><div>26/44</div></div>	68% <div><div>26/38</div></div>
MenuToppingService.java	87% <div><div>39/45</div></div>	63% <div><div>15/24</div></div>	79% <div><div>15/19</div></div>

Figure 1.6: MenuPizzaService mutation score after refactoring

Test additions/edits code snippets:

```
@Test
public void addPizzaTest() {
    Assertions.assertThat(this.menuPizzaService.addPizza(this.p1)).isFalse();
    Pizza p = new Pizza(id: 43L, List.of(t1, t2, t3), name: "Depperoni", new BigDecimal(val: "78.99"));
    Assertions.assertThat(this.menuPizzaService.addPizza(p)).isTrue();
    Pizza p1 = new Pizza(id: 44L, List.of(t1, t2, t3), name: "Fepperoni", new BigDecimal(val: "78.99"));
    when(this.toppingRepository.findToppingById(10L)).thenReturn(Optional.empty());
    Assertions.assertThat(this.menuPizzaService.addPizza(p1)).isFalse();
    when(authManager.getRole()).thenReturn("customer");
    when(this.toppingRepository.findToppingById(10L)).thenReturn(Optional.ofNullable(t1));
    Assertions.assertThat(this.menuPizzaService.addPizza(p)).isFalse();
    when(authManager.getRole()).thenReturn("regional_manager");
    when(this.toppingRepository.findToppingById(10L)).thenReturn(Optional.ofNullable(t2));
    Assertions.assertThat(this.menuPizzaService.addPizza(p)).isFalse();
}
```

Figure 1.7: MenuPizzaService added test to kill mutants

Class Name - MenuAllergyService

Before the changes, this class had **43%** mutation coverage. After the changes, it has **63%** mutation coverage.

Commit link: 502ddd032c5a2db35a9d7a8c5975de9c40b4f07b

Mutation Test Coverage Report Before:

Pit Test Coverage Report

Package Summary

nl.tudelft.sem.group06b.menu.service

Number of Classes	Line Coverage	Mutation Coverage	Test Strength
3	73% 157/214	46% 48/105	70% 48/69

Breakdown by Class

Name	Line Coverage	Mutation Coverage	Test Strength
MenuAllergyService.java	68% 59/87	43% 16/37	73% 16/22
MenuPizzaService.java	72% 59/82	39% 17/44	61% 17/28
MenuToppingService.java	87% 39/45	63% 15/24	79% 15/19

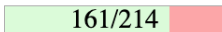
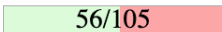
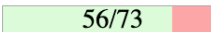
Figure 1.8: MenuAllergyService mutation score before refactoring

Mutation Test Coverage Report After:

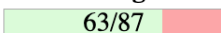
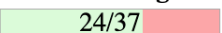
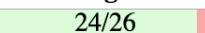
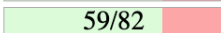
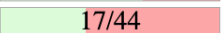
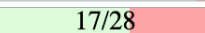
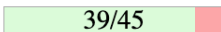
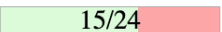
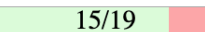
Pit Test Coverage Report

Package Summary

nl.tudelft.sem.group06b.menu.service

Number of Classes	Line Coverage	Mutation Coverage	Test Strength
3	75% 	53% 	77% 

Breakdown by Class

Name	Line Coverage	Mutation Coverage	Test Strength
MenuAllergyService.java	72% 	65% 	92% 
MenuPizzaService.java	72% 	39% 	61% 
MenuToppingService.java	87% 	63% 	79% 

Report generated by [PIT](#) 1.9.5

Figure 1.9: MenuAllergyService mutation score after refactoring

Test additions/edits code snippets:

```
@Test
public void addAllergyTest_wrongRole_returnsFalse() {
    when(authManager.getRole()).thenReturn("customer");
    Allergy allergy = new Allergy(id: 2L, allergen: "allergy");
    when(allergyRepository.findAllById(3L)).thenReturn(Optional.of(allergy));
    Assertions.assertThat(menuAllergyService.addAllergy(allergy)).isFalse();
}

@Test
public void addAllergyTest_notPresent_returnsFalse() {
    when(authManager.getRole()).thenReturn("regional_manager");
    when(allergyRepository.findAllById(3L)).thenReturn(Optional.empty());
    Assertions.assertThat(menuAllergyService.addAllergy(new Allergy(id: 2L, allergen: "allergy"))).isFalse();
}
```

Figure 1.10: MenuAllergyService code snippet 1


```

@Test
public void filterAllergies() {
    //simple test for filtering out one
    Assertions.assertThat(
        this.menuAllergyService.filterPizzasByAllergens(List.of("Peanuts")))
        .hasSameElementsAs(List.of(p2));
    //test for filtering out no pizzas
    Assertions.assertThat(
        this.menuAllergyService.filterPizzasByAllergens(List.of("Sugar")))
        .hasSameElementsAs(List.of(p1, p2, p3));

    ArrayList<Topping> toppings = new ArrayList<>();
    toppings.addAll(p3.getToppings());
    toppings.add(new Topping( id: 6L, name: "SaLami", List.of(bbb), new BigDecimal( val: "32.99")));
    this.p3.setToppings(toppings);

    Assertions.assertThat(
        this.menuAllergyService.filterPizzasByAllergens(List.of("Hawaii")))
        .hasSameElementsAs(List.of(p1, p2));

    Assertions.assertThat(
        this.menuAllergyService.filterPizzasByAllergens(
            List.of("Peanuts", "Hawaii", "Food")))
        .hasSameElementsAs(List.of(p2));
    verify(allergyRepository, times( wantedNumberOfInvocations: 15)).flush();
}

```

Figure 1.11: MenuAllergyService code snippet 2

```

@Test
public void filterToppings() {
    Assertions.assertThat(
        this.menuAllergyService.filterToppingsByAllergens(List.of("Peanuts")))
        .hasSameElementsAs(List.of(t3));
    Assertions.assertThat(
        this.menuAllergyService.filterToppingsByAllergens(List.of("Sugar")))
        .hasSameElementsAs(List.of(t1, t2, t3));
    verify(allergyRepository, times( wantedNumberOfInvocations: 5)).flush();
}

```

Figure 1.12: MenuAllergyService code snippet 3

```

@Test
public void getAllergyById() {
    Assertions.assertThat(this.menuAllergyService.getAllergyById(null)).isEmpty();
    verify(allergyRepository, times( wantedNumberOfInvocations: 1)).flush();
}

```

Figure 1.13: MenuAllergyService code snippet 4

Figure 1.14: MenuAllergyService code snippet 5

Class Name - MenuToppingService

Before the changes, this class had **63%** mutation coverage. After the changes, it has **83%** mutation coverage. For this addition, some production code has changed, mostly adding exceptions that the database might throw to method signatures and the according try catch parts.

Commit links:

92687911238a56fe94372564c5c11545ad003dad, c2cb7150dbf97672c0d0197e12428eaa842d93f3

Mutation Test Coverage Report Before:

Pit Test Coverage Report

Package Summary

nl.tudelft.sem.group06b.menu.service

Number of Classes	Line Coverage	Mutation Coverage	Test Strength
3	73% 157/214	46% 48/105	70% 48/69

Breakdown by Class

Name	Line Coverage	Mutation Coverage	Test Strength
MenuAllergyService.java	68% 59/87	43% 16/37	73% 16/22
MenuPizzaService.java	72% 59/82	39% 17/44	61% 17/28
MenuToppingService.java	87% 39/45	63% 15/24	79% 15/19

Figure 1.15: MenuToppingService mutation score before refactoring

Mutation Test Coverage Report After:

Pit Test Coverage Report

Package Summary

nl.tudelft.sem.group06b.menu.service

Number of Classes	Line Coverage	Mutation Coverage	Test Strength
3	75% 160/214	50% 53/105	75% 53/71

Breakdown by Class

Name	Line Coverage	Mutation Coverage	Test Strength
MenuAllergyService.java	68% 59/87	43% 16/37	73% 16/22
MenuPizzaService.java	72% 59/82	39% 17/44	61% 17/28
MenuToppingService.java	93% 42/45	83% 20/24	95% 20/21

Report generated by [PIT](#) 1.9.5

Figure 1.16: MenuToppingService mutation score after refactoring

Test additions/edits code snippets:

```
@Test
public void getAllToppingsTest_emptyList() {
    when(toppingRepository.findAll()).thenReturn(Collections.emptyList());
    Assertions.assertThat(menuToppingService.getAllToppings()).isEqualTo(Collections.emptyList());
    verify(toppingRepository, times(wantedNumberOfInvocations: 1)).flush();
}

@Test
public void getAllToppingsTest_listWithElements() {
    Topping t1 = new Topping(id: 18L, name: "Potato", List.of(this.aaa, this.bbb), new BigDecimal(val: "10.99"));
    Topping t2 = new Topping(id: 19L, name: "Potato2", List.of(this.aaa, this.ccc), new BigDecimal(val: "10.99"));

    when(toppingRepository.findAll()).thenReturn(List.of(t1, t2));
    Assertions.assertThat(menuToppingService.getAllToppings()).containsExactlyInAnyOrder(t2, t1);
    verify(toppingRepository, times(wantedNumberOfInvocations: 1)).flush();
}
```

Figure 1.17: MenuToppingService code snippet 1

```
@Test
public void addTopping() {
    Allergy ddd = new Allergy(id: 988L, allergen: "Something made up");
    Topping t = new Topping(id: 18L, name: "Potato", List.of(this.aaa, ddd), new BigDecimal(val: "10.99"));
    Assertions.assertThat(this.menuToppingService.addTopping(t)).isFalse();
    Topping t2 = new Topping(id: 19L, name: "Potato", List.of(this.aaa, this.ccc), new BigDecimal(val: "10.99"));
    Assertions.assertThat(this.menuToppingService.addTopping(t2)).isTrue();
    Assertions.assertThat(this.menuToppingService.addTopping(t: null)).isFalse();
    verify(toppingRepository, times(wantedNumberOfInvocations: 1)).flush();
    verify(toppingRepository, times(wantedNumberOfInvocations: 1)).save(any());
}

@Test
public void addTopping_throws_returnsFalse() throws Exception {
    when(toppingRepository.findToppingById(any(Long.class))).thenReturn(new Exception("Database error"));
    Assertions.assertThat(this.menuToppingService.addTopping(
        new Topping(id: 2L, name: "topping", allergies: null, price: null))).isFalse();
}
```

Figure 1.18: MenuToppingService code snippet 2

```
@Test
public void removeToppingById() throws Exception {
    Assertions.assertThat(this.menuToppingService.removeToppingById(null)).isFalse();
    Assertions.assertThat(this.menuToppingService.removeToppingById(3948793L)).isFalse();
    Assertions.assertThat(this.menuToppingService.removeToppingById(10L)).isTrue();
    verify(toppingRepository, times(wantedNumberOfInvocations: 1)).flush();
}
```

Figure 1.19: MenuToppingService code snippet 3

List of Figures

1.1	OrderCouponImpl mutation score before refactoring	1
1.2	OrderCouponImpl mutation score after refactoring	1
1.3	OrderCouponImpl code before refactoring	2
1.4	OrderCouponImpl code after refactoring	2
1.5	MenuPizzaService mutation score before refactoring	3
1.6	MenuPizzaService mutation score after refactoring	3
1.7	MenuPizzaService added test to kill mutants	4
1.8	MenuAllergyService mutation score before refactoring	4
1.9	MenuAllergyService mutation score after refactoring	5
1.10	MenuAllergyService code snippet 1	5
1.11	MenuAllergyService code snippet 2	6
1.12	MenuAllergyService code snippet 3	6
1.13	MenuAllergyService code snippet 4	6
1.14	MenuAllergyService code snippet 5	6
1.15	MenuToppingService mutation score before refactoring	7
1.16	MenuToppingService mutation score after refactoring	7
1.17	MenuToppingService code snippet 1	8
1.18	MenuToppingService code snippet 2	8
1.19	MenuToppingService code snippet 3	8