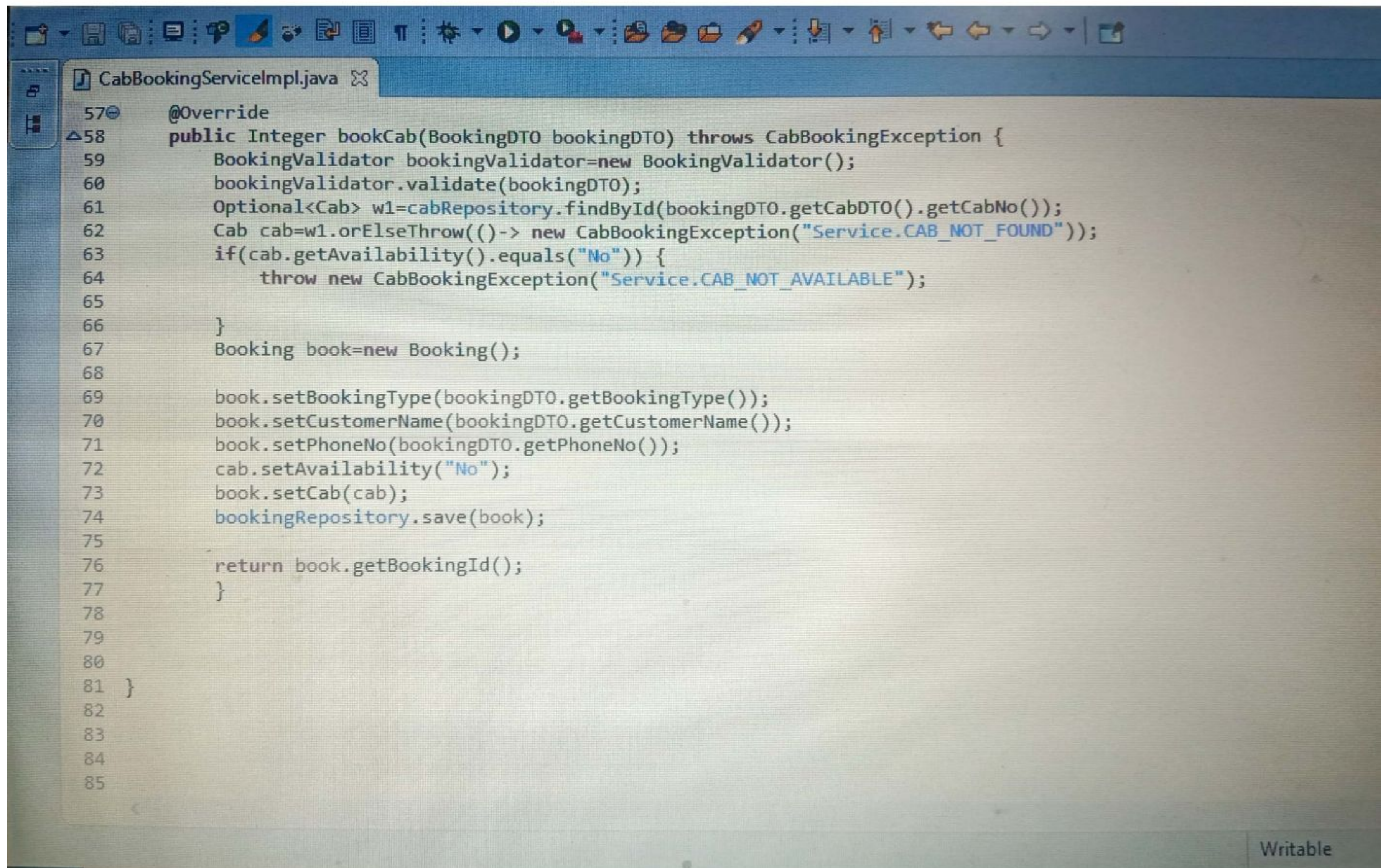


CabBookingServiceImpl.java

```
1 package com.infy.cabbooking.service;
2
3 import java.util.ArrayList;
4
5
6
7
8
9
10
11
12
13 @Service(value="cabBookingService")
14 @Transactional
15 public class CabBookingServiceImpl implements CabBookingService {
16     @Autowired
17     private BookingRepository bookingRepository;
18     @Autowired
19     private CabRepository cabRepository;
20
21
22
23 @Override
24 public List<BookingDTO> getDetailsByBookingType(String bookingType) throws CabBookingException {
25     List<Booking> w1=bookingRepository.findByBookingType(bookingType);
26     List<BookingDTO> bookingDTOs=new ArrayList<>();
27     if(w1.isEmpty()) {
28         throw new CabBookingException("Service.NO_DETAILS_FOUND");
29     }
30     for(Booking booking:w1) {
31         BookingDTO b=new BookingDTO();
32         b.setBookingId(booking.getBookingId());
33         b.setCustomerName(booking.getCustomerName());
34         b.setBookingType(booking.getBookingType());
35         b.setPhoneNo(booking.getPhoneNo());
36         CabDTO c=new CabDTO();
37     }
38 }
```

Writab

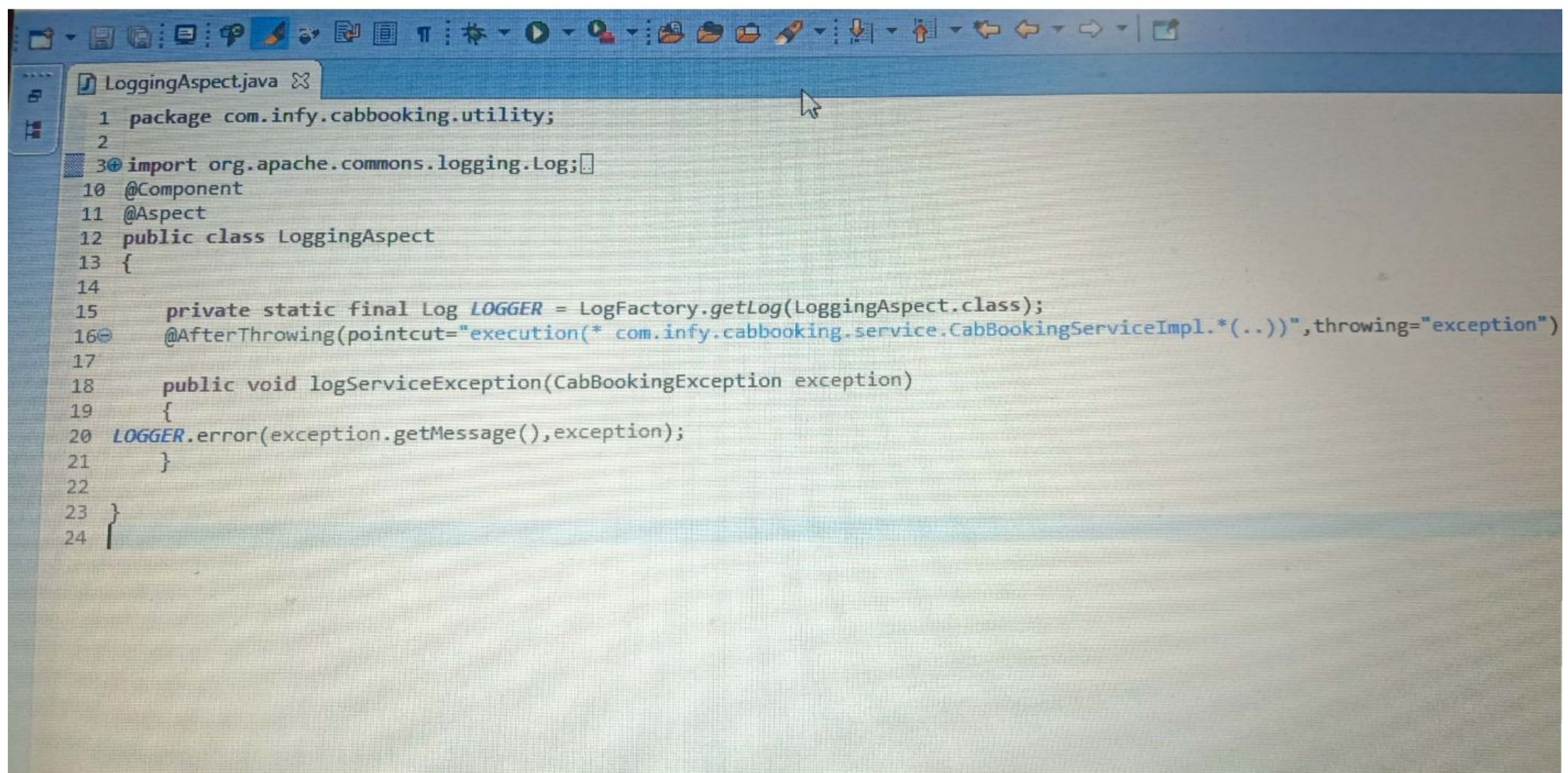

```
CabBookingServiceImpl.java
40      bookingDTO b=new bookingDTO();
41      b.setBookingId(booking.getBookingId());
42      b.setCustomerName(booking.getCustomerName());
43      b.setBookingType(booking.getBookingType());
44      b.setPhoneNo(booking.getPhoneNo());
45      CabDTO c=new CabDTO();
46      c.setCabNo(booking.getCab().getCabNo());
47      c.setModelName(booking.getCab().getModelName());
48      c.setAvailability(booking.getCab().getAvailability());
49      c.setDriverPhoneNo(booking.getCab().getDriverPhoneNo());
50      b.setCabDTO(c);
51      bookingDTOs.add(b);
52  }
53      return bookingDTOs;
54  }
55
56
57  @Override
58  public Integer bookCab(BookingDTO bookingDTO) throws CabBookingException {
59      BookingValidator bookingValidator=new BookingValidator();
60      bookingValidator.validate(bookingDTO);
61      Optional<Cab> w1=cabRepository.findById(bookingDTO.getCabDTO().getCabNo());
62      Cab cab=w1.orElseThrow(()-> new CabBookingException("Service.CAB_NOT_FOUND"));
63      if(cab.getAvailability().equals("No")) {
64          throw new CabBookingException("Service.CAB_NOT_AVAILABLE");
65      }
66  }
67      Booking book=new Booking();
68
```

The image shows a screenshot of an IDE window titled "CabBookingServiceImpl.java". The code is as follows:

```
57 @Override
58 public Integer bookCab(BookingDTO bookingDTO) throws CabBookingException {
59     BookingValidator bookingValidator=new BookingValidator();
60     bookingValidator.validate(bookingDTO);
61     Optional<Cab> w1=cabRepository.findById(bookingDTO.getCabDTO().getCabNo());
62     Cab cab=w1.orElseThrow(()-> new CabBookingException("Service.CAB_NOT_FOUND"));
63     if(cab.getAvailability().equals("No")) {
64         throw new CabBookingException("Service.CAB_NOT_AVAILABLE");
65     }
66     Booking book=new Booking();
67     book.setBookingType(bookingDTO.getBookingType());
68     book.setCustomerName(bookingDTO.getCustomerName());
69     book.setPhoneNo(bookingDTO.getPhoneNo());
70     cab.setAvailability("No");
71     book.setCab(cab);
72     bookingRepository.save(book);
73     return book.getBookingId();
74 }
75
76
77
78
79
80
81 }
82
83
84
85
```

At the bottom right of the IDE window, the word "Writable" is visible.



```
1 package com.infy.cabbooking.utility;
2
3 import org.apache.commons.logging.Log;
4
10 @Component
11 @Aspect
12 public class LoggingAspect
13 {
14
15     private static final Log LOGGER = LoggerFactory.getLog(LoggingAspect.class);
16     @AfterThrowing(pointcut="execution(* com.infy.cabbooking.service.CabBookingServiceImpl.*(..))",throwing="exception")
17
18     public void logServiceException(CabBookingException exception)
19     {
20         LOGGER.error(exception.getMessage(),exception);
21     }
22
23 }
24 }
```



```
CabBookingApplicationTests.java
1 package com.infy.cabbooking;
2
3 import java.util.ArrayList;
24
25 @SpringBootTest
26 public class CabBookingApplicationTests {
27
28
29     @Mock
30     private CabRepository cabRepository;
31
32
33     @Mock
34     private BookingRepository bookingRepository;
35
36     @InjectMocks
37     private CabBookingService cabBookingService = new CabBookingServiceImpl();
38
39
40     @Test
41     public void bookCabInvalidCabNoTest() throws Exception{
42         BookingDTO bookingDTO=new BookingDTO();
43         bookingDTO.setCustomerName("Robert");
44         bookingDTO.setBookingId(1001);
45         bookingDTO.setPhoneNo(9867542341L);
46         bookingDTO.setBookingType("Personal");
47         CabDTO cabDTO=new CabDTO();
48         cabDTO.setCabNo(451678);
49         cabDTO.setDriverPhoneNo(9947835654L);
```



```
File Edit Source Refactor Navigate Search Project Run Window Help
CabBookingApplicationTests.java
40 @test
41 public void bookCabInvalidCabNoTest() throws Exception{
42     BookingDTO bookingDTO=new BookingDTO();
43     bookingDTO.setCustomerName("Robert");
44     bookingDTO.setBookingId(1001);
45     bookingDTO.setPhoneNo(9867542341L);
46     bookingDTO.setBookingType("Personal");
47     CabDTO cabDTO=new CabDTO();
48     cabDTO.setCabNo(451678);
49     cabDTO.setDriverPhoneNo(9947835654L);
50     cabDTO.setModelName("Toyota");
51     cabDTO.setAvailability("Yes");
52     bookingDTO.setCabDTO(cabDTO);
53     CabDTO cab=new CabDTO();
54     cab.setCabNo(159721);
55     cab.setDriverPhoneNo(9947835654L);
56     cab.setAvailability("Yes");
57     cab.setModelName("Toyota");
58     Mockito.<Optional<Cab>>when(cabRepository.findById(cabDTO.getCabNo())).thenReturn(Optional.empty());
59     CabBookingException cBE=Assertions.assertThrows(CabBookingException.class,()->cabBookingService.bookCab(bookingDTO));
60
61     Assertions.assertEquals("Service.CAB_NOT_FOUND",cBE.getMessage());
62 }
63
64 @Test
65 public void getDetailsByBookingTypeNoDetailsFound() throws Exception{
66     BookingDTO bookingDTO=new BookingDTO();
67     bookingDTO.setCustomerName("Robert");
68     bookingDTO.setBookingId(1001);
69     ...
```


CabBookingApplicationTests.java

```

61 Assertions.assertEquals("Service.CAB_NOT_FOUND", cBE.getMessage());
62 }
63
64 @Test
65 public void getDetailsByBookingTypeNoDetailsFound() throws Exception{
66     BookingDTO bookingDTO=new BookingDTO();
67     bookingDTO.setCustomerName("Robert");
68     bookingDTO.setBookingId(1001);
69     bookingDTO.setPhoneNo(9867542341L);
70     bookingDTO.setBookingType("Shared");
71     CabDTO cabDTO=new CabDTO();
72     cabDTO.setDriverPhoneNo(9947835654L);
73     cabDTO.setModelName("Toyota");
74     cabDTO.setAvailability("Yes");
75     bookingDTO.setCabDTO(cabDTO);
76     List<Booking> bookings=new ArrayList<>();
77     Booking b=new Booking();
78     b.setCustomerName("Robert");
79     b.setBookingId(1001);
80     b.setPhoneNo(9867542341L);
81     b.setBookingType("Share");
82     Cab cab=new Cab();
83     cab.setDriverPhoneNo(9947835654L);
84     cab.setAvailability("Yes");
85     cab.setModelName("Toyota");
86     b.setCab(cab);
87     bookings.add(b);
88     BookingDTO booking1=new BookingDTO();
89     booking1.setCustomerName("Robert");
90     booking1.setBookingId(1001);
91     booking1.setPhoneNo(9867542341L);
92     booking1.setBookingType("Share");
93     booking1.setCabDTO(cab);
94     List<Booking> bookings1=new ArrayList<>();
95     bookings1.add(booking1);
96     BookingDTO booking2=new BookingDTO();
97     booking2.setCustomerName("Robert");
98     booking2.setBookingId(1001);
99     booking2.setPhoneNo(9867542341L);
100    booking2.setBookingType("Share");
101    booking2.setCabDTO(cab);
102    List<Booking> bookings2=new ArrayList<>();
103    bookings2.add(booking2);
104    BookingDTO booking3=new BookingDTO();
105    booking3.setCustomerName("Robert");
106    booking3.setBookingId(1001);
107    booking3.setPhoneNo(9867542341L);
108    booking3.setBookingType("Share");
109    booking3.setCabDTO(cab);
110    List<Booking> bookings3=new ArrayList<>();
111    bookings3.add(booking3);
112    BookingDTO booking4=new BookingDTO();
113    booking4.setCustomerName("Robert");
114    booking4.setBookingId(1001);
115    booking4.setPhoneNo(9867542341L);
116    booking4.setBookingType("Share");
117    booking4.setCabDTO(cab);
118    List<Booking> bookings4=new ArrayList<>();
119    bookings4.add(booking4);
120    BookingDTO booking5=new BookingDTO();
121    booking5.setCustomerName("Robert");
122    booking5.setBookingId(1001);
123    booking5.setPhoneNo(9867542341L);
124    booking5.setBookingType("Share");
125    booking5.setCabDTO(cab);
126    List<Booking> bookings5=new ArrayList<>();
127    bookings5.add(booking5);
128    BookingDTO booking6=new BookingDTO();
129    booking6.setCustomerName("Robert");
130    booking6.setBookingId(1001);
131    booking6.setPhoneNo(9867542341L);
132    booking6.setBookingType("Share");
133    booking6.setCabDTO(cab);
134    List<Booking> bookings6=new ArrayList<>();
135    bookings6.add(booking6);
136    BookingDTO booking7=new BookingDTO();
137    booking7.setCustomerName("Robert");
138    booking7.setBookingId(1001);
139    booking7.setPhoneNo(9867542341L);
140    booking7.setBookingType("Share");
141    booking7.setCabDTO(cab);
142    List<Booking> bookings7=new ArrayList<>();
143    bookings7.add(booking7);
144    BookingDTO booking8=new BookingDTO();
145    booking8.setCustomerName("Robert");
146    booking8.setBookingId(1001);
147    booking8.setPhoneNo(9867542341L);
148    booking8.setBookingType("Share");
149    booking8.setCabDTO(cab);
150    List<Booking> bookings8=new ArrayList<>();
151    bookings8.add(booking8);
152    BookingDTO booking9=new BookingDTO();
153    booking9.setCustomerName("Robert");
154    booking9.setBookingId(1001);
155    booking9.setPhoneNo(9867542341L);
156    booking9.setBookingType("Share");
157    booking9.setCabDTO(cab);
158    List<Booking> bookings9=new ArrayList<>();
159    bookings9.add(booking9);
160    BookingDTO booking10=new BookingDTO();
161    booking10.setCustomerName("Robert");
162    booking10.setBookingId(1001);
163    booking10.setPhoneNo(9867542341L);
164    booking10.setBookingType("Share");
165    booking10.setCabDTO(cab);
166    List<Booking> bookings10=new ArrayList<>();
167    bookings10.add(booking10);
168    BookingDTO booking11=new BookingDTO();
169    booking11.setCustomerName("Robert");
170    booking11.setBookingId(1001);
171    booking11.setPhoneNo(9867542341L);
172    booking11.setBookingType("Share");
173    booking11.setCabDTO(cab);
174    List<Booking> bookings11=new ArrayList<>();
175    bookings11.add(booking11);
176    BookingDTO booking12=new BookingDTO();
177    booking12.setCustomerName("Robert");
178    booking12.setBookingId(1001);
179    booking12.setPhoneNo(9867542341L);
180    booking12.setBookingType("Share");
181    booking12.setCabDTO(cab);
182    List<Booking> bookings12=new ArrayList<>();
183    bookings12.add(booking12);
184    BookingDTO booking13=new BookingDTO();
185    booking13.setCustomerName("Robert");
186    booking13.setBookingId(1001);
187    booking13.setPhoneNo(9867542341L);
188    booking13.setBookingType("Share");
189    booking13.setCabDTO(cab);
190    List<Booking> bookings13=new ArrayList<>();
191    bookings13.add(booking13);
192    BookingDTO booking14=new BookingDTO();
193    booking14.setCustomerName("Robert");
194    booking14.setBookingId(1001);
195    booking14.setPhoneNo(9867542341L);
196    booking14.setBookingType("Share");
197    booking14.setCabDTO(cab);
198    List<Booking> bookings14=new ArrayList<>();
199    bookings14.add(booking14);
200    BookingDTO booking15=new BookingDTO();
201    booking15.setCustomerName("Robert");
202    booking15.setBookingId(1001);
203    booking15.setPhoneNo(9867542341L);
204    booking15.setBookingType("Share");
205    booking15.setCabDTO(cab);
206    List<Booking> bookings15=new ArrayList<>();
207    bookings15.add(booking15);
208    BookingDTO booking16=new BookingDTO();
209    booking16.setCustomerName("Robert");
210    booking16.setBookingId(1001);
211    booking16.setPhoneNo(9867542341L);
212    booking16.setBookingType("Share");
213    booking16.setCabDTO(cab);
214    List<Booking> bookings16=new ArrayList<>();
215    bookings16.add(booking16);
216    BookingDTO booking17=new BookingDTO();
217    booking17.setCustomerName("Robert");
218    booking17.setBookingId(1001);
219    booking17.setPhoneNo(9867542341L);
220    booking17.setBookingType("Share");
221    booking17.setCabDTO(cab);
222    List<Booking> bookings17=new ArrayList<>();
223    bookings17.add(booking17);
224    BookingDTO booking18=new BookingDTO();
225    booking18.setCustomerName("Robert");
226    booking18.setBookingId(1001);
227    booking18.setPhoneNo(9867542341L);
228    booking18.setBookingType("Share");
229    booking18.setCabDTO(cab);
230    List<Booking> bookings18=new ArrayList<>();
231    bookings18.add(booking18);
232    BookingDTO booking19=new BookingDTO();
233    booking19.setCustomerName("Robert");
234    booking19.setBookingId(1001);
235    booking19.setPhoneNo(9867542341L);
236    booking19.setBookingType("Share");
237    booking19.setCabDTO(cab);
238    List<Booking> bookings19=new ArrayList<>();
239    bookings19.add(booking19);
240    BookingDTO booking20=new BookingDTO();
241    booking20.setCustomerName("Robert");
242    booking20.setBookingId(1001);
243    booking20.setPhoneNo(9867542341L);
244    booking20.setBookingType("Share");
245    booking20.setCabDTO(cab);
246    List<Booking> bookings20=new ArrayList<>();
247    bookings20.add(booking20);
248    BookingDTO booking21=new BookingDTO();
249    booking21.setCustomerName("Robert");
250    booking21.setBookingId(1001);
251    booking21.setPhoneNo(9867542341L);
252    booking21.setBookingType("Share");
253    booking21.setCabDTO(cab);
254    List<Booking> bookings21=new ArrayList<>();
255    bookings21.add(booking21);
256    BookingDTO booking22=new BookingDTO();
257    booking22.setCustomerName("Robert");
258    booking22.setBookingId(1001);
259    booking22.setPhoneNo(9867542341L);
260    booking22.setBookingType("Share");
261    booking22.setCabDTO(cab);
262    List<Booking> bookings22=new ArrayList<>();
263    bookings22.add(booking22);
264    BookingDTO booking23=new BookingDTO();
265    booking23.setCustomerName("Robert");
266    booking23.setBookingId(1001);
267    booking23.setPhoneNo(9867542341L);
268    booking23.setBookingType("Share");
269    booking23.setCabDTO(cab);
270    List<Booking> bookings23=new ArrayList<>();
271    bookings23.add(booking23);
272    BookingDTO booking24=new BookingDTO();
273    booking24.setCustomerName("Robert");
274    booking24.setBookingId(1001);
275    booking24.setPhoneNo(9867542341L);
276    booking24.setBookingType("Share");
277    booking24.setCabDTO(cab);
278    List<Booking> bookings24=new ArrayList<>();
279    bookings24.add(booking24);
280    BookingDTO booking25=new BookingDTO();
281    booking25.setCustomerName("Robert");
282    booking25.setBookingId(1001);
283    booking25.setPhoneNo(9867542341L);
284    booking25.setBookingType("Share");
285    booking25.setCabDTO(cab);
286    List<Booking> bookings25=new ArrayList<>();
287    bookings25.add(booking25);
288    BookingDTO booking26=new BookingDTO();
289    booking26.setCustomerName("Robert");
290    booking26.setBookingId(1001);
291    booking26.setPhoneNo(9867542341L);
292    booking26.setBookingType("Share");
293    booking26.setCabDTO(cab);
294    List<Booking> bookings26=new ArrayList<>();
295    bookings26.add(booking26);
296    BookingDTO booking27=new BookingDTO();
297    booking27.setCustomerName("Robert");
298    booking27.setBookingId(1001);
299    booking27.setPhoneNo(9867542341L);
300    booking27.setBookingType("Share");
301    booking27.setCabDTO(cab);
302    List<Booking> bookings27=new ArrayList<>();
303    bookings27.add(booking27);
304    BookingDTO booking28=new BookingDTO();
305    booking28.setCustomerName("Robert");
306    booking28.setBookingId(1001);
307    booking28.setPhoneNo(9867542341L);
308    booking28.setBookingType("Share");
309    booking28.setCabDTO(cab);
310    List<Booking> bookings28=new ArrayList<>();
311    bookings28.add(booking28);
312    BookingDTO booking29=new BookingDTO();
313    booking29.setCustomerName("Robert");
314    booking29.setBookingId(1001);
315    booking29.setPhoneNo(9867542341L);
316    booking29.setBookingType("Share");
317    booking29.setCabDTO(cab);
318    List<Booking> bookings29=new ArrayList<>();
319    bookings29.add(booking29);
320    BookingDTO booking30=new BookingDTO();
321
```

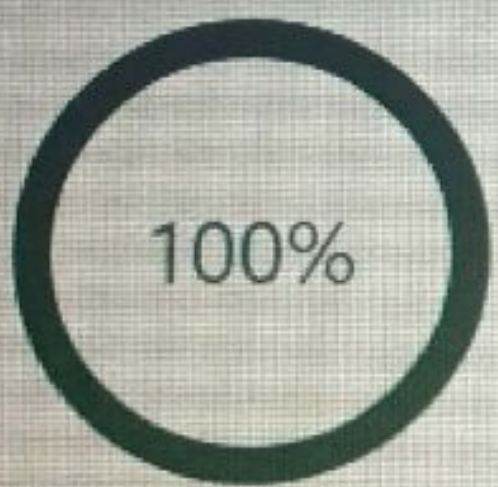


```
bookingApplicationTests.java
b.setPhoneNo(9867542341L);
b.setBookingType("Share");
Cab cab=new Cab();
cab.setDriverPhoneNo(9947835654L);
cab.setAvailability("Yes");
cab.setModelName("Toyota");
b.setCab(cab);
bookings.add(b);
BookingDTO booking1=new BookingDTO();
booking1.setCustomerName("Robert");
booking1.setBookingId(1001);
booking1.setPhoneNo(9867542341L);
b.setBookingType("Share");
CabDTO cab1=new CabDTO();
cab1.setDriverPhoneNo(9947835654L);
cab1.setAvailability("Yes");
cab1.setModelName("Toyota");
booking1.setCabDTO(cab1);

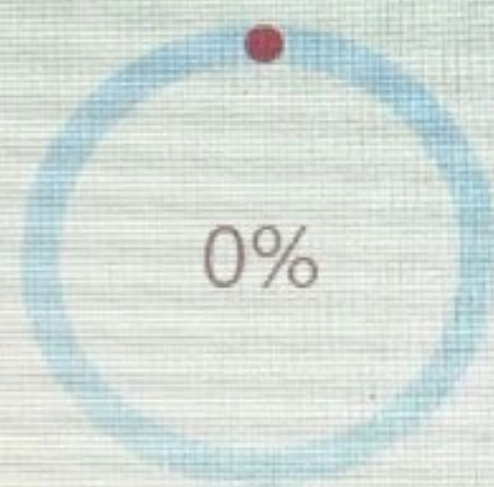
Mockito.<List<Booking>>when(bookingRepository.findByBookingType(bookingDTO.getBookingType())).thenReturn(bookings);
CabBookingException cBE=Assertions.assertThrows(CabBookingException.class,()->cabBookingService.getDetailsByBookingType(booking1.getBookingType()));

Assertions.assertEquals("Service.NO_DETAILS_FOUND",cBE.getMessage());
}
```


Verifying Your Code



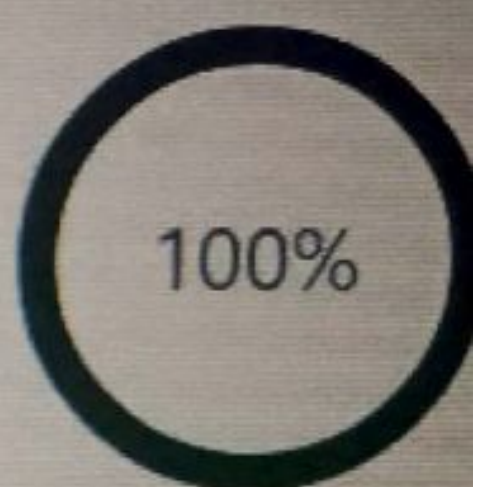
Total Passed Test Cases



Total Failed Test Cases



Structural Correctness



Logical Correctness

Overall Report

Detailed Report