Booking.java | CabBookingAPI.java

```java
package com.infy.cabbooking.api;

import java.util.List;
@RestController
@RequestMapping(value="api")
public class CabBookingAPI
{
    @Autowired
    private CabBookingService cabBookingService;
    @Autowired
    private Environment environment;
    @GetMapping(value="cab/{bookingType}")
    public ResponseEntity<List<BookingDTO>> getDetailsByBookingType(@PathVariable  String bookingType) throws CabBookingExceptio
    {
    return new ResponseEntity<>(cabBookingService.getDetailsByBookingType(bookingType),HttpStatus.OK);
    }
    @PostMapping(value="cab")
    public ResponseEntity<String> bookCab(@RequestBody BookingDTO bookingDTO) throws CabBookingException
    {
        Integer bookId=cabBookingService.bookCab(bookingDTO);

        return new ResponseEntity<>(environment.getProperty("API.BOOKING_SUCCESS")+":"+bookId,HttpStatus.CREATED);
    }

}
```

Scanned with CamScanner

Booking.java

```java
package com.infy.cabbooking.entity;

import javax.persistence.CascadeType;

@Entity
@Table(name="booking")
public class Booking {

    @Id
    @GeneratedValue(strategy=GenerationType.IDENTITY)
    private Integer bookingId;
    private String customerName;
    private Long phoneNo;
    private String bookingType;

    @OneToOne(cascade=CascadeType.ALL)
    @JoinColumn(name="cab_no",unique=true)
    private Cab cab;


    public Integer getBookingId() {
        return bookingId;
    }
    public void setBookingId(Integer bookingId) {
        this.bookingId = bookingId;
    }
    public String getCustomerName() {
        return customerName;
    }
```

Scanned with CamScanner

Booking.java

```java
    public void setBookingId(Integer bookingId) {
        this.bookingId = bookingId;
    }
    public String getCustomerName() {
        return customerName;
    }
    public void setCustomerName(String customerName) {
        this.customerName = customerName;
    }
    public Long getPhoneNo() {
        return phoneNo;
    }
    public void setPhoneNo(Long phoneNo) {
        this.phoneNo = phoneNo;
    }
    public String getBookingType() {
        return bookingType;
    }
    public void setBookingType(String bookingType) {
        this.bookingType = bookingType;
    }
    public Cab getCab() {
        return cab;
    }
    public void setCab(Cab cab) {
        this.cab = cab;
    }

}
```

Writab

Scanned with CamScanner

Cab.java

```java
package com.infy.cabbooking.entity;

import javax.persistence.Entity;

@Entity
@Table(name="cab")
public class Cab {
    @Id
    private Integer cabNo;
    private String modelName;
    private Long driverPhoneNo;
    private String availability;


    public Integer getCabNo() {
        return cabNo;
    }
    public void setCabNo(Integer cabNo) {
        this.cabNo = cabNo;
    }
    public String getModelName() {
        return modelName;
    }
    public void setModelName(String modelName) {
        this.modelName = modelName;
    }

    public Long getDriverPhoneNo() {
        return driverPhoneNo;
```

Scanned with CamScanner

Cab.java

```java


    public Integer getCabNo() {
        return cabNo;
    }
    public void setCabNo(Integer cabNo) {
        this.cabNo = cabNo;
    }
    public String getModelName() {
        return modelName;
    }
    public void setModelName(String modelName) {
        this.modelName = modelName;
    }

    public Long getDriverPhoneNo() {
        return driverPhoneNo;
    }
    public void setDriverPhoneNo(Long driverPhoneNo) {
        this.driverPhoneNo = driverPhoneNo;
    }
    public String getAvailability() {
        return availability;
    }
    public void setAvailability(String availability) {
        this.availability = availability;
    }

}
```

Scanned with CamScanner

BookingRepository.java

```java
package com.infy.cabbooking.repository;

import java.util.List;

public interface BookingRepository extends CrudRepository<Booking,Integer> {
    List<Booking> findByBookingType(String BookingType);

}
```

Scanned with CamScanner

CabRepository.java

```java
package com.infy.cabbooking.repository;

import org.springframework.data.repository.CrudRepository;

public interface CabRepository extends CrudRepository<Cab,Integer> {

}
```

Scanned with CamScanner

CabBookingServiceImpl.java

```java
package com.infy.cabbooking.service;

import java.util.ArrayList;



@Service(value="cabBookingService")
@Transactional
public class CabBookingServiceImpl implements CabBookingService {
    @Autowired
    private BookingRepository bookingRepository;
    @Autowired
    private CabRepository cabRepository;


    @Override
    public List<BookingDTO> getDetailsByBookingType(String bookingType) throws CabBookingException {
        List<Booking> w1=bookingRepository.findByBookingType(bookingType);
        List<BookingDTO> bookingDTOs=new ArrayList<>();
        if(w1.isEmpty()) {
            throw new CabBookingException("Service.NO_DETAILS_FOUND");
        }
        for(Booking booking:w1) {
            BookingDTO b=new BookingDTO();
            b.setBookingId(booking.getBookingId());
            b.setCustomerName(booking.getCustomerName());
            b.setBookingType(booking.getBookingType());
            b.setPhoneNo(booking.getPhoneNo());
            CabDTO c=new CabDTO();
```

Writabl

Scanned with CamScanner

CabBookingServiceImpl.java

```
            BookingDTO b=new BookingDTO();
            b.setBookingId(booking.getBookingId());
            b.setCustomerName(booking.getCustomerName());
            b.setBookingType(booking.getBookingType());
            b.setPhoneNo(booking.getPhoneNo());
            CabDTO c=new CabDTO();
            c.setCabNo(booking.getCab().getCabNo());
            c.setModelName(booking.getCab().getModelName());
            c.setAvailability(booking.getCab().getAvailability());
            c.setDriverPhoneNo(booking.getCab().getDriverPhoneNo());
            b.setCabDTO(c);
            bookingDTOs.add(b);
        }
            return bookingDTOs;
        }


    @Override
    public Integer bookCab(BookingDTO bookingDTO) throws CabBookingException {
        BookingValidator bookingValidator=new BookingValidator();
        bookingValidator.validate(bookingDTO);
        Optional<Cab> w1=cabRepository.findById(bookingDTO.getCabDTO().getCabNo());
        Cab cab=w1.orElseThrow(()-> new CabBookingException("Service.CAB_NOT_FOUND"));
        if(cab.getAvailability().equals("No")) {
            throw new CabBookingException("Service.CAB_NOT_AVAILABLE");

        }
        Booking book=new Booking();
```

Scanned with CamScanner

CabBookingServiceImpl.java

```java
    @Override
    public Integer bookCab(BookingDTO bookingDTO) throws CabBookingException {
        BookingValidator bookingValidator=new BookingValidator();
        bookingValidator.validate(bookingDTO);
        Optional<Cab> w1=cabRepository.findById(bookingDTO.getCabDTO().getCabNo());
        Cab cab=w1.orElseThrow(()-> new CabBookingException("Service.CAB_NOT_FOUND"));
        if(cab.getAvailability().equals("No")) {
            throw new CabBookingException("Service.CAB_NOT_AVAILABLE");

        }
        Booking book=new Booking();

        book.setBookingType(bookingDTO.getBookingType());
        book.setCustomerName(bookingDTO.getCustomerName());
        book.setPhoneNo(bookingDTO.getPhoneNo());
        cab.setAvailability("No");
        book.setCab(cab);
        bookingRepository.save(book);

        return book.getBookingId();
        }



}
```

Writable

Scanned with CamScanner

LoggingAspect.java

```java
package com.infy.cabbooking.utility;

import org.apache.commons.logging.Log;
@Component
@Aspect
public class LoggingAspect
{

    private static final Log LOGGER = LogFactory.getLog(LoggingAspect.class);
    @AfterThrowing(pointcut="execution(* com.infy.cabbooking.service.CabBookingServiceImpl.*(..))",throwing="exception")

    public void logServiceException(CabBookingException exception)
    {
LOGGER.error(exception.getMessage(),exception);
    }

}
```

Scanned with CamScanner

CabBookingApplicationTests.java

```java
package com.infy.cabbooking;

import java.util.ArrayList;...

@SpringBootTest
public class CabBookingApplicationTests {


    @Mock
    private CabRepository cabRepository;


    @Mock
    private BookingRepository bookingRepository;

    @InjectMocks
    private CabBookingService cabBookingService = new CabBookingServiceImpl();


    @Test
    public void bookCabInvalidCabNoTest() throws Exception{
        BookingDTO bookingDTO=new BookingDTO();
    bookingDTO.setCustomerName("Robert");
    bookingDTO.setBookingId(1001);
    bookingDTO.setPhoneNo(9867542341L);
    bookingDTO.setBookingType("Personal");
    CabDTO cabDTO=new CabDTO();
    cabDTO.setCabNo(451678);
    cabDTO.setDriverPhoneNo(9947835654L);
```

Writab

Scanned with CamScanner

File Edit Source Refactor Navigate Search Project Run Window Help

CabBookingApplicationTests.java

```java
    @Test
    public void bookCabInvalidCabNoTest() throws Exception{
        BookingDTO bookingDTO=new BookingDTO();
    bookingDTO.setCustomerName("Robert");
    bookingDTO.setBookingId(1001);
    bookingDTO.setPhoneNo(9867542341L);
    bookingDTO.setBookingType("Personal");
    CabDTO cabDTO=new CabDTO();
    cabDTO.setCabNo(451678);
    cabDTO.setDriverPhoneNo(9947835654L);
    cabDTO.setModelName("Toyota");
    cabDTO.setAvailability("Yes");
    bookingDTO.setCabDTO(cabDTO);
    CabDTO cab=new CabDTO();
    cab.setCabNo(159721);
    cab.setDriverPhoneNo(9947835654L);
    cab.setAvailability("Yes");
    cab.setModelName("Toyota");
    Mockito.<Optional<Cab>>when(cabRepository.findById(cabDTO.getCabNo())).thenReturn(Optional.empty());
    CabBookingException cBE=Assertions.assertThrows(CabBookingException.class,()->cabBookingService.bookCab(bookingDTO));

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

    @Test
    public void getDetailsByBookingTypeNoDetailsFound() throws Exception{
        BookingDTO bookingDTO=new BookingDTO();
        bookingDTO.setCustomerName("Robert");
        bookingDTO.setBookingId(1001);
```

Scanned with CamScanner

CabBookingApplicationTests.java

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

    @Test
    public void getDetailsByBookingTypeNoDetailsFound() throws Exception{
        BookingDTO bookingDTO=new BookingDTO();
        bookingDTO.setCustomerName("Robert");
        bookingDTO.setBookingId(1001);
        bookingDTO.setPhoneNo(9867542341L);
        bookingDTO.setBookingType("Shared");
        CabDTO cabDTO=new CabDTO();
        cabDTO.setDriverPhoneNo(9947835654L);
        cabDTO.setModelName("Toyota");
        cabDTO.setAvailability("Yes");
        bookingDTO.setCabDTO(cabDTO);
        List<Booking> bookings=new ArrayList<>();
        Booking b=new Booking();
        b.setCustomerName("Robert");
        b.setBookingId(1001);
        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();
```

Scanned with CamScanner

ookingApplicationTests.java

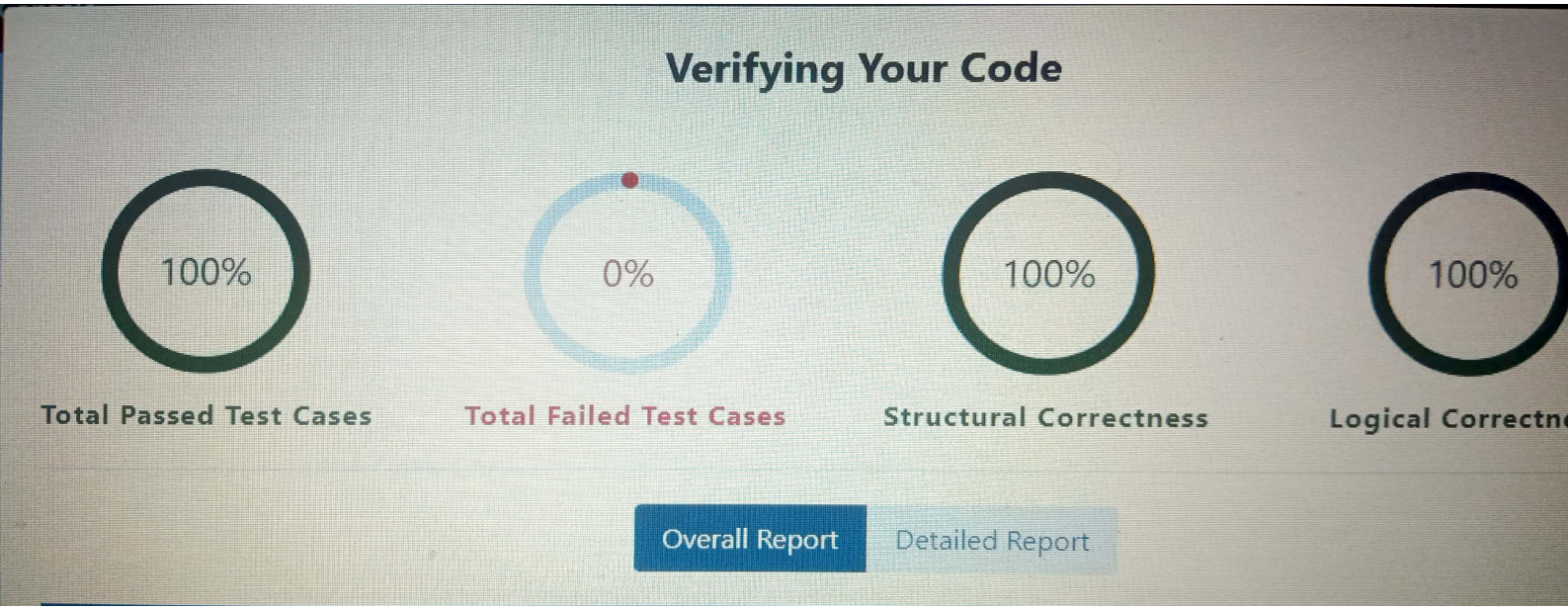
```java
b.setPhoneNo(9867542341L);
b.setBookingType("Share");
Cab cab=new Cab();
cab.setDriverPhoneNo(9947835654L);
cab.setAvailability("Yes");
cab.setModelName("Toyota");
b.setCab(cab);
bookings.add(b);
BookingDTO booking1=new BookingDTO();
booking1.setCustomerName("Robert");
booking1.setBookingId(1001);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

}
```

Scanned with CamScanner



Scanned with CamScanner