MovieBooking.java | MovieBookingServiceImpl.java

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

import java.time.LocalDate;

@Entity
@Table(name="movie_booking")
public class MovieBooking {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Integer bookingId;
    private String movieName;
    private String screenName;
    private LocalDate showDate;
    private Integer noOfSeats;
    private Double bookingAmount;
    private String paymentType;
    private Long customerPhoneNo;

    public Integer getBookingId() {
        return bookingId;
    }
    public String getMovieName() {
        return movieName;
    }
    public String getScreenName() {
        return screenName;
    }
    public LocalDate getShowDate() {
        return showDate;
    }
    public Integer getNoOfSeats() {
        return noOfSeats;
```

MovieBooking.java | MovieBookingServiceImpl.java | MovieBookingDTO.java

```java
package com.infy.moviebooking.dto;
import java.time.LocalDate;
public class MovieBookingDTO {
private Integer bookingid;

    @NotNull(message="{movie.name.absent}")
    private String movieName;

    @NotNull(message="{movie.screen.absent}")
    @Pattern(regexp = "(Sapphire|Turquoise|Rhombus)", message = "{movie.screen.invalid}")
    private String screenName;

    @NotNull(message="{movie.show.absent}")
    @FutureOrPresent(message = "{movie.show.invalid}")
    private LocalDate showDate;


    @NotNull(message = "{movie.noOfSeats.absent}")
    @Min(value = 1,message = "{movie.noOfSeats.invalid}")
    @Max(value = 5,message = "{movie.noOfSeats.invalid}")
    private Integer noOfSeats;


    private Double bookingAmount;
    private String paymentType;
    private Long customerPhoneNo;
    public Integer getBookingid() {
        return bookingid;
    }
    public void setBookingId(Integer bookingid) {
        this.bookingid = bookingid;
    }
    public String getMovieName() {
        return movieName;
    }
    public void setMovieName(String movieName) {
```

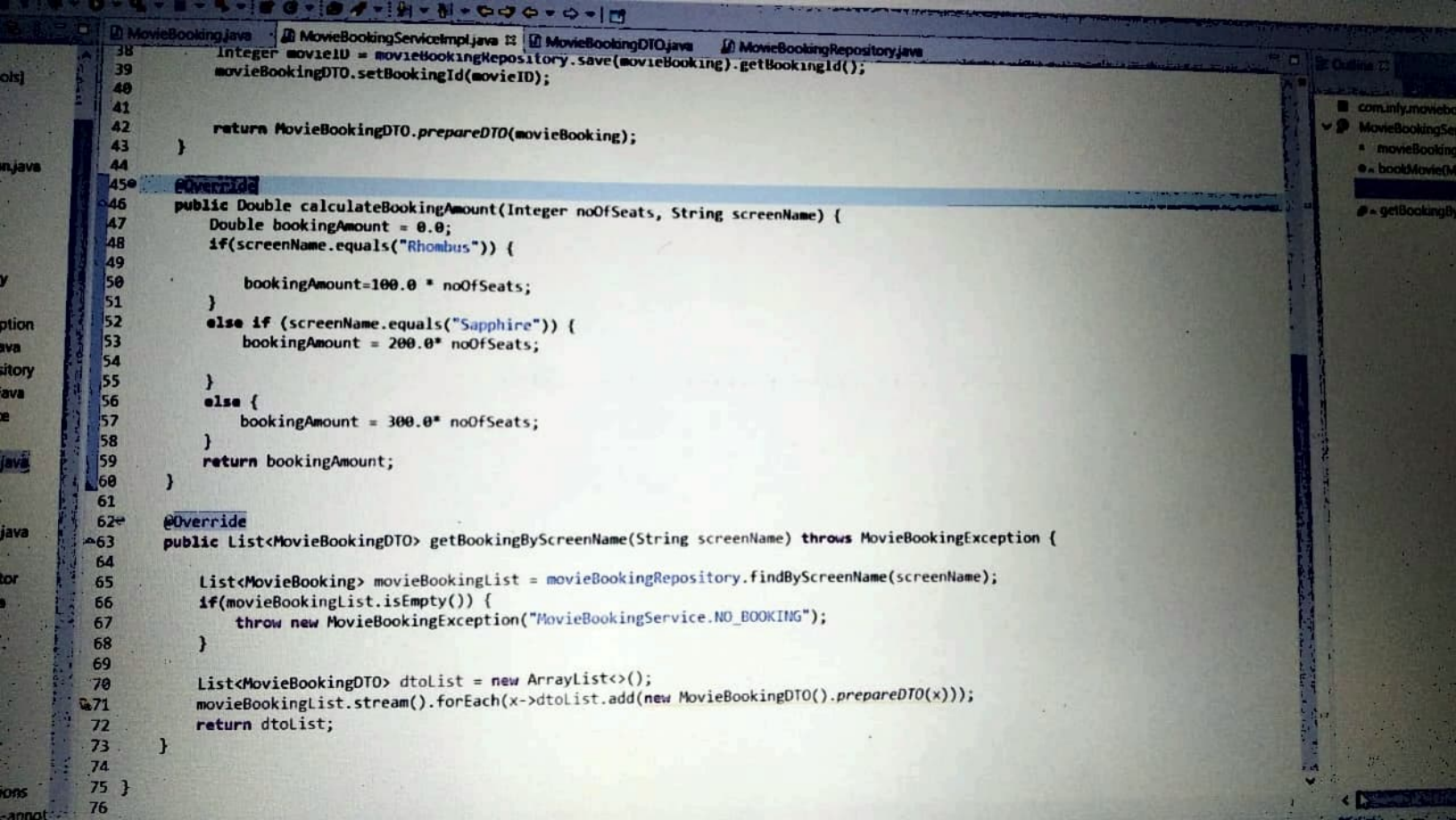
```
    public void setCustomerPhoneNo(Long customerPhoneNo) {
        this.customerPhoneNo = customerPhoneNo;
    }

    public static MovieBooking prepareEntity(MovieBookingDTO movieBookingDTO) {

        MovieBooking movieBooking = new MovieBooking();
        movieBooking.setMovieName(movieBookingDTO.getMovieName());
        movieBooking.setScreenName(movieBookingDTO.getScreenName());
        movieBooking.setShowDate(movieBookingDTO.getShowDate());
        movieBooking.setNoOfSeats(movieBookingDTO.getNoOfSeats());
        movieBooking.setBookingAmount(movieBookingDTO.getBookingAmount());
        movieBooking.setPaymentType(movieBookingDTO.getPaymentType());
        movieBooking.setCustomerPhoneNo(movieBookingDTO.getCustomerPhoneNo());
        return movieBooking;

    }

    public static MovieBookingDTO prepareDTO( MovieBooking movieBooking) {

        MovieBookingDTO movieBookingDTO = new MovieBookingDTO();
        movieBookingDTO.setBookingId(movieBooking.getBookingId());
        movieBookingDTO.setMovieName(movieBooking.getMovieName());
        movieBookingDTO.setScreenName(movieBooking.getScreenName());
        movieBookingDTO.setShowDate(movieBooking.getShowDate());
        movieBookingDTO.setNoOfSeats(movieBooking.getNoOfSeats());
        movieBookingDTO.setBookingAmount(movieBooking.getBookingAmount());
        movieBookingDTO.setPaymentType(movieBooking.getPaymentType());
        movieBookingDTO.setCustomerPhoneNo(movieBooking.getCustomerPhoneNo());
        return movieBookingDTO;

    }

}
```

MovieBooking.java | MovieBookingServiceImpl.java | MovieBookingDTO.java | MovieBookingRepository.java

```
1  package com.infy.moviebooking.repository;
2
3  import java.time.LocalDate;
11
12 public interface MovieBookingRepository extends CrudRepository<MovieBooking, Integer> {
13     @Query("select m from MovieBooking m where m.customerPhoneNo=?1 and m.showDate=?2")
14     List<MovieBooking> getBookingDetails(Long custmoerPhoneNo, LocalDate showDate );
15     List<MovieBooking> findByScreenName(String screenName);
16 }
17
```

MovieBooking.java | MovieBookingServiceImpl.java | MovieBookingDTO.java | MovieBookingRepository.java

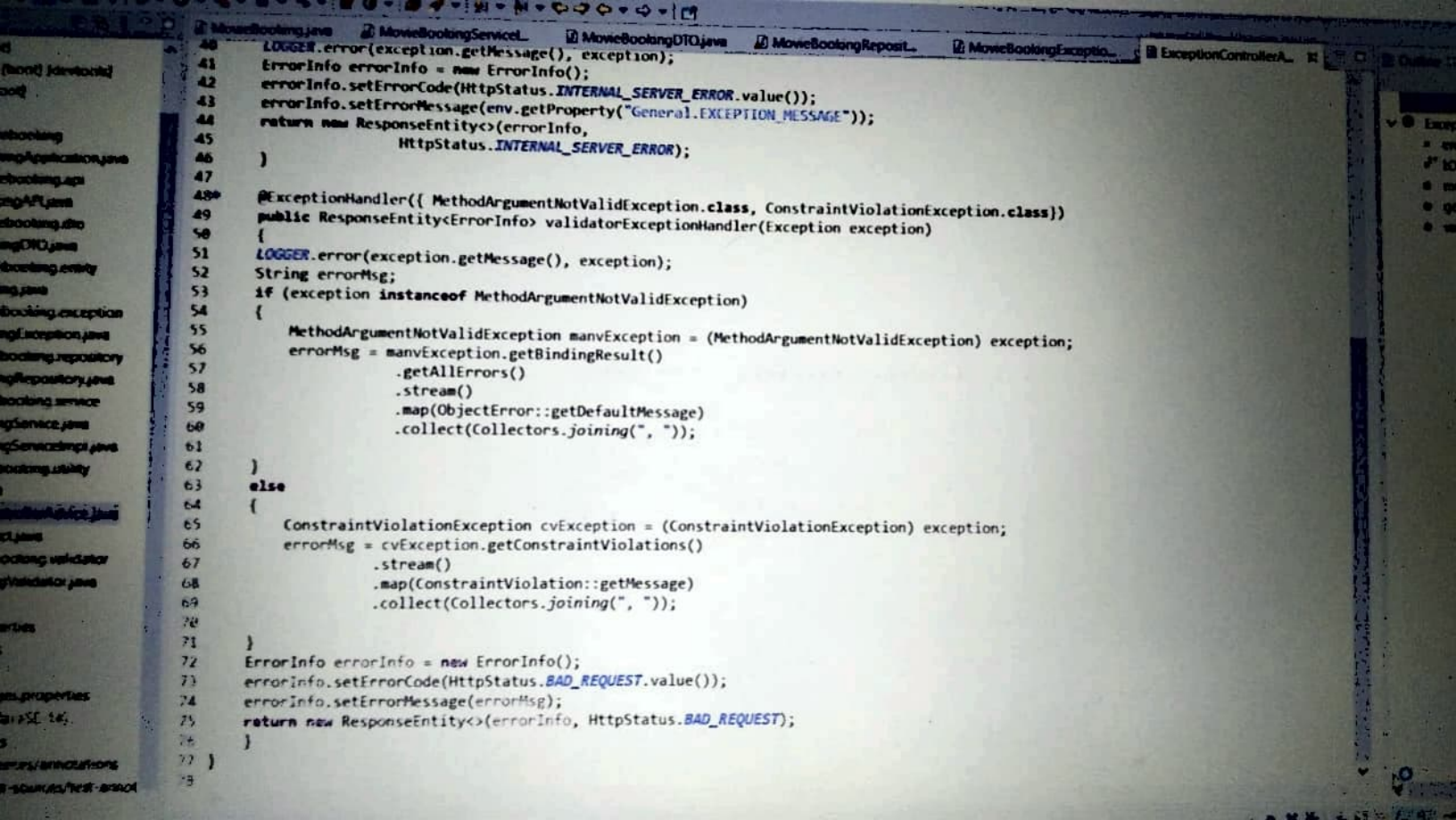
```java
package com.infy.moviebooking.service;

import java.util.ArrayList;

@Service(value="movieBookingService")
@Transactional
public class MovieBookingServiceImpl implements MovieBookingService {

    @Autowired
    private MovieBookingRepository movieBookingRepository;

    @Override
    public MovieBookingDTO bookMovie(MovieBookingDTO movieBookingDTO) throws MovieBookingException {

        MovieBookingValidator.validate(movieBookingDTO);
        List<MovieBooking> movieBookingList = movieBookingRepository.getBookingDetails(movieBookingDTO.getCustomerPhoneNo(),
                movieBookingDTO.getShowDate());

        if(Boolean.FALSE.equals(movieBookingList.isEmpty())){

            throw new MovieBookingException("MovieBookingService.BOOKING_EXISTS");
        }

        MovieBooking movieBooking = MovieBookingDTO.prepareEntity(movieBookingDTO);
        movieBooking.setBookingAmount(calculateBookingAmount(movieBooking.getNoOfSeats(), movieBooking.getScreenName()));
        Integer movieID = movieBookingRepository.save(movieBooking).getBookingId();
        movieBookingDTO.setBookingId(movieID);


        return MovieBookingDTO.prepareDTO(movieBooking);
    }

    @Override
    public Double calculateBookingAmount(Integer noOfSeats, String screenName) {
```



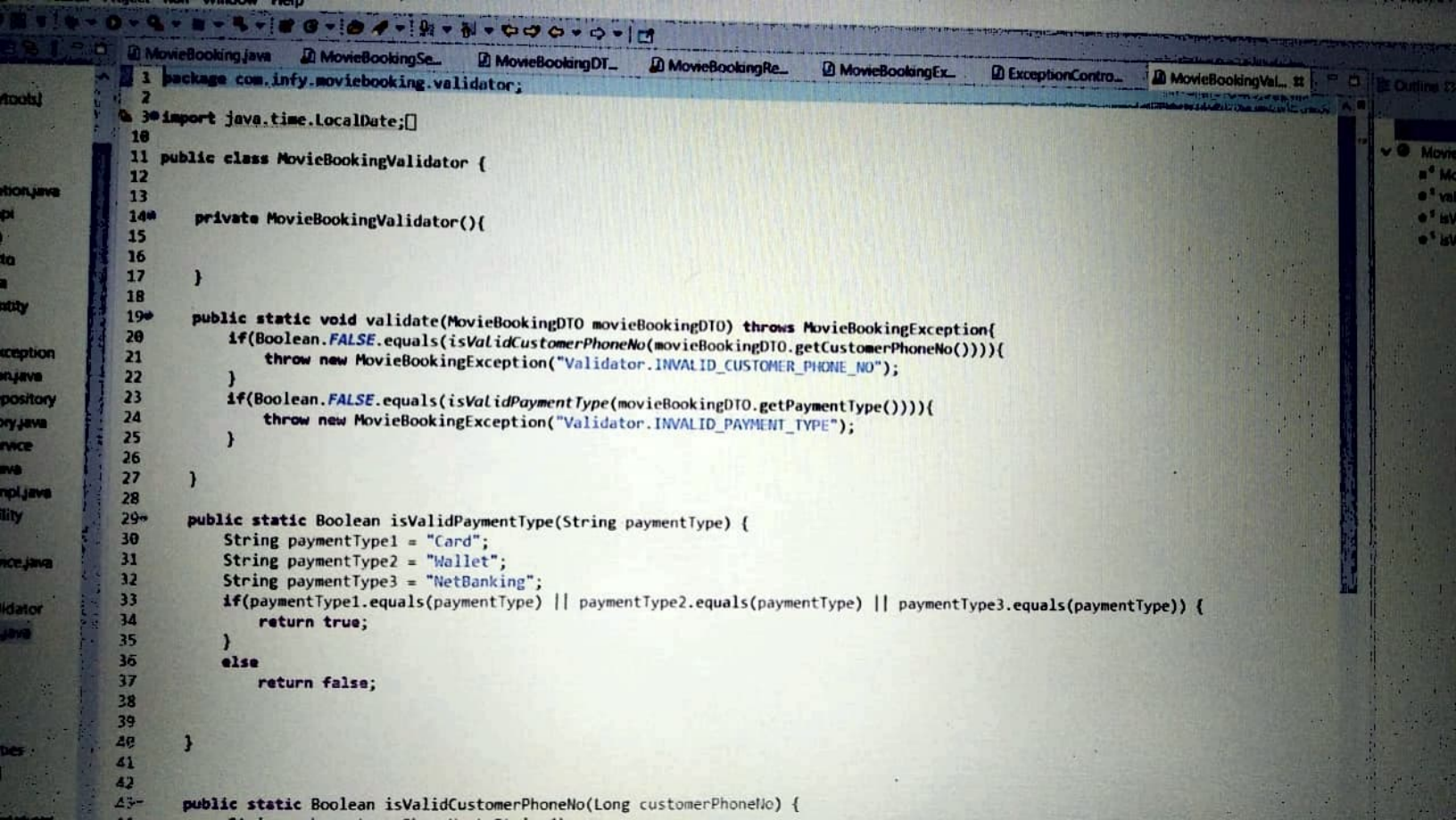
```
        Integer movieID = movieBookingRepository.save(movieBooking).getBookingId();
        movieBookingDTO.setBookingId(movieID);


        return MovieBookingDTO.prepareDTO(movieBooking);
    }

    @Override
    public Double calculateBookingAmount(Integer noOfSeats, String screenName) {
        Double bookingAmount = 0.0;
        if(screenName.equals("Rhombus")) {

            bookingAmount=100.0 * noOfSeats;
        }
        else if (screenName.equals("Sapphire")) {
            bookingAmount = 200.0* noOfSeats;

        }
        else {
            bookingAmount = 300.0* noOfSeats;
        }
        return bookingAmount;
    }

    @Override
    public List<MovieBookingDTO> getBookingByScreenName(String screenName) throws MovieBookingException {

        List<MovieBooking> movieBookingList = movieBookingRepository.findByScreenName(screenName);
        if(movieBookingList.isEmpty()) {
            throw new MovieBookingException("MovieBookingService.NO_BOOKING");
        }

        List<MovieBookingDTO> dtoList = new ArrayList<>();
        movieBookingList.stream().forEach(x->dtoList.add(new MovieBookingDTO().prepareDTO(x)));
        return dtoList;
    }

}
```

MovieBooking.java | MovieBookingServiceI... | MovieBookingDTO.java | MovieBookingReposit... | MovieBookingExceptio... | ExceptionControllerA...

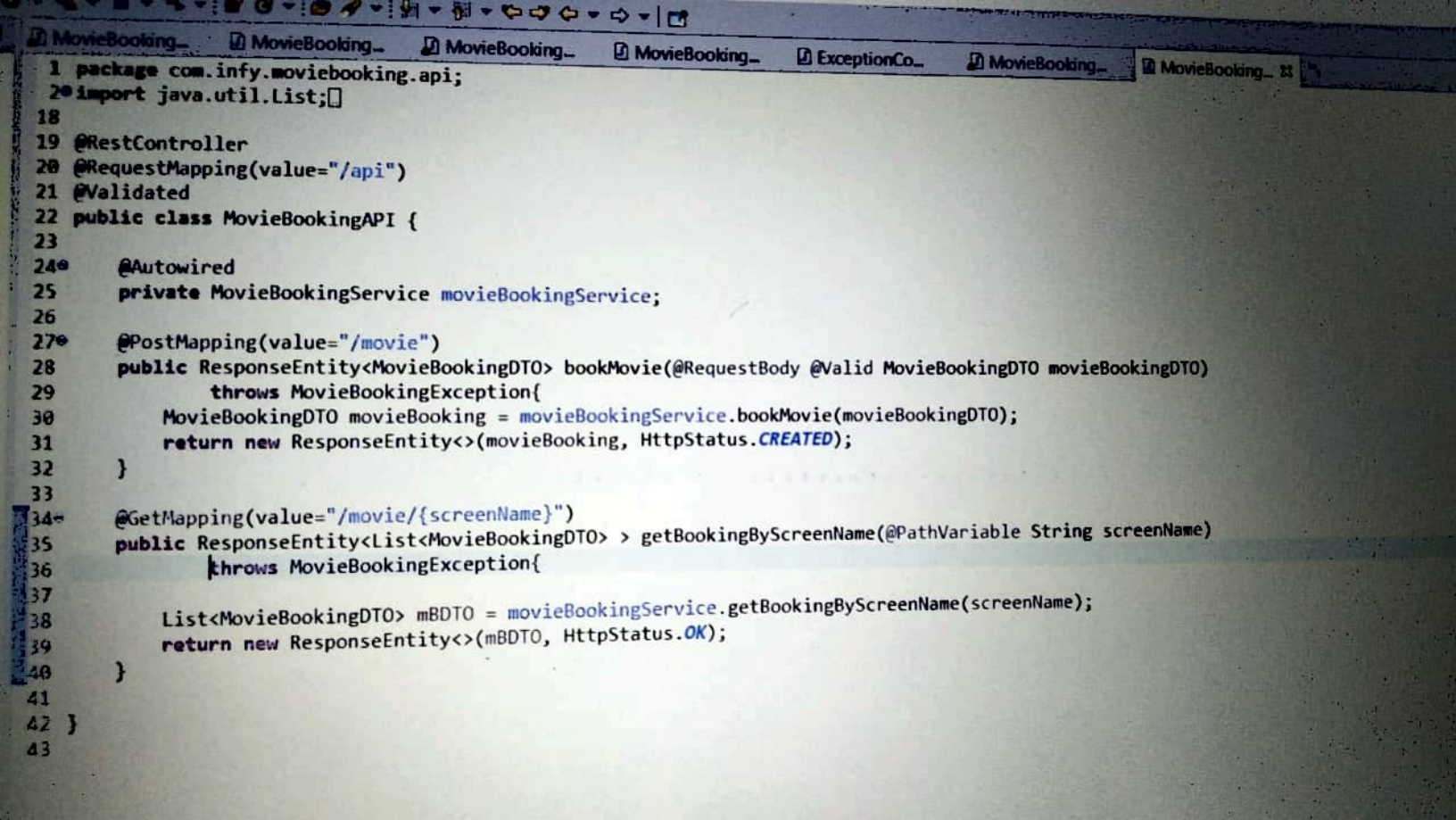
```java
package com.infy.moviebooking.utility;

import java.util.stream.Collectors;

@RestControllerAdvice
public class ExceptionControllerAdvice {
    @Autowired
    private Environment env;

    private static final Log LOGGER = LogFactory.getLog(ExceptionControllerAdvice.class);

    @ExceptionHandler(MovieBookingException.class)
    public ResponseEntity<ErrorInfo> movieBookingExceptionHandler(MovieBookingException exception){
        LOGGER.error(exception.getMessage(), exception);
        ErrorInfo errorInfo = new ErrorInfo();
        errorInfo.setErrorCode(HttpStatus.BAD_REQUEST.value());
        errorInfo.setErrorMessage(env.getProperty(exception.getMessage()));
        return new ResponseEntity<>(errorInfo, HttpStatus.BAD_REQUEST);
    }

    @ExceptionHandler(Exception.class)
    public ResponseEntity<ErrorInfo> generalExceptionHandler(Exception exception)
    {
    LOGGER.error(exception.getMessage(), exception);
    ErrorInfo errorInfo = new ErrorInfo();
    errorInfo.setErrorCode(HttpStatus.INTERNAL_SERVER_ERROR.value());
    errorInfo.setErrorMessage(env.getProperty("General.EXCEPTION_MESSAGE"));
    return new ResponseEntity<>(errorInfo,
                    HttpStatus.INTERNAL_SERVER_ERROR);
    }

    @ExceptionHandler({ MethodArgumentNotValidException.class, ConstraintViolationException.class})
    public ResponseEntity<ErrorInfo> validatorExceptionHandler(Exception exception)
    {
    LOGGER.error(exception.getMessage(), exception);
    String errorMsg;
    if (exception instanceof MethodArgumentNotValidException)
```


LOGGER.error(exception.getMessage(), exception);
ErrorInfo errorInfo = new ErrorInfo();
errorInfo.setErrorCode(HttpStatus.INTERNAL_SERVER_ERROR.value());
errorInfo.setErrorMessage(env.getProperty("General.EXCEPTION_MESSAGE"));
return new ResponseEntity<>(errorInfo,
HttpStatus.INTERNAL_SERVER_ERROR);
}
@ExceptionHandler({ MethodArgumentNotValidException.class, ConstraintViolationException.class})
public ResponseEntity<ErrorInfo> validatorExceptionHandler(Exception exception)
{
LOGGER.error(exception.getMessage(), exception);
String errorMsg;
if (exception instanceof MethodArgumentNotValidException)
{
MethodArgumentNotValidException manvException = (MethodArgumentNotValidException) exception;
errorMsg = manvException.getBindingResult()
.getAllErrors()
.stream()
.map(ObjectError::getDefaultMessage)
.collect(Collectors.joining(", "));
}
else
{
ConstraintViolationException cvException = (ConstraintViolationException) exception;
errorMsg = cvException.getConstraintViolations()
.stream()
.map(ConstraintViolation::getMessage)
.collect(Collectors.joining(", "));
}
ErrorInfo errorInfo = new ErrorInfo();
errorInfo.setErrorCode(HttpStatus.BAD_REQUEST.value());
errorInfo.setErrorMessage(errorMsg);
return new ResponseEntity<>(errorInfo, HttpStatus.BAD_REQUEST);
}
}

```java
        }

    }

    public static Boolean isValidPaymentType(String paymentType) {
        String paymentType1 = "Card";
        String paymentType2 = "Wallet";
        String paymentType3 = "NetBanking";
        if(paymentType1.equals(paymentType) || paymentType2.equals(paymentType) || paymentType3.equals(paymentType)) {
            return true;
        }
        else
            return false;


    }


    public static Boolean isValidCustomerPhoneNo(Long customerPhoneNo) {
        String mob=customerPhoneNo.toString();
        Boolean b=true;
        if(mob.length()==10) {
            char c=mob.charAt(0);
            Integer a = Character.getNumericValue(c);
            if((a>=3)&&(a<=9))

                b=true;

                else
                   b=false;
                   return b;


            }
        else
            return false;
    }

}

```



```java
package com.infy.moviebooking.validator;

import java.time.LocalDate;

public class MovieBookingValidator {


    private MovieBookingValidator(){


    }

    public static void validate(MovieBookingDTO movieBookingDTO) throws MovieBookingException{
        if(Boolean.FALSE.equals(isValidCustomerPhoneNo(movieBookingDTO.getCustomerPhoneNo()))){
            throw new MovieBookingException("Validator.INVALID_CUSTOMER_PHONE_NO");
        }
        if(Boolean.FALSE.equals(isValidPaymentType(movieBookingDTO.getPaymentType()))){
            throw new MovieBookingException("Validator.INVALID_PAYMENT_TYPE");
        }

    }

    public static Boolean isValidPaymentType(String paymentType) {
        String paymentType1 = "Card";
        String paymentType2 = "Wallet";
        String paymentType3 = "NetBanking";
        if(paymentType1.equals(paymentType) || paymentType2.equals(paymentType) || paymentType3.equals(paymentType)) {
            return true;
        }
        else
            return false;


    }


    public static Boolean isValidCustomerPhoneNo(Long customerPhoneNo) {
```


MovieBooking_
MovieBooking_
MovieBooking_
MovieBooking_
ExceptionCo_
MovieBooking_
MovieBooking_
1 package com.infy.moviebooking.api;
2 import java.util.List;
18
19 @RestController
20 @RequestMapping(value="/api")
21 @Validated
22 public class MovieBookingAPI {
23
24 @Autowired
25 private MovieBookingService movieBookingService;
26
27 @PostMapping(value="/movie")
28 public ResponseEntity<MovieBookingDTO> bookMovie(@RequestBody @Valid MovieBookingDTO movieBookingDTO)
29 throws MovieBookingException{
30 MovieBookingDTO movieBooking = movieBookingService.bookMovie(movieBookingDTO);
31 return new ResponseEntity<>(movieBooking, HttpStatus.CREATED);
32 }
33
34 @GetMapping(value="/movie/{screenName}")
35 public ResponseEntity<List<MovieBookingDTO> > getBookingByScreenName(@PathVariable String screenName)
36 throws MovieBookingException{
37
38 List<MovieBookingDTO> mBDTO = movieBookingService.getBookingByScreenName(screenName);
39 return new ResponseEntity<>(mBDTO, HttpStatus.OK);
40 }
41
42 }
43