







user-controller



PUT	/user/edit	 
POST	/user/add	
GET	/user/{id}	
DELETE	/user/{id}	
GET	/user/all	

```
curl -X 'GET' \
  'http://localhost:8081/user/all' \
  -H 'accept: */*'

```



Request URL

http://localhost:8081/user/all

Server response

Code

Details

200

Response body

```
{
  "userId": 1,
  "userName": "Rohit Sharma",
  "email": "ro@gmail.com",
  "password": "12345678",
  "role": "customer",
  "mobile": "9632587410",
  "address": "asdfghjkl"
},
{
  "userId": 3,
  "userName": "Virat",
  "email": "v@gmail.com",
  "password": "125896347",
  "role": "customer",
  "mobile": "3698521470",
  "address": "asdfghjklasdf"
},
{
  "userId": 21,
  "userName": "Sanju Samson",
  "email": "sanju@gmail.com",
  "password": "12345678",
  "role": "customer",
  "mobile": "9632587410",
  "address": "vhv,mvmv m v "
}
}
```



Download

room-controller



PUT /rooms/update



POST /rooms/add



GET /rooms/{id}



DELETE /rooms/{id}



GET /rooms/all



hotel-controller



PUT /hotels/update



POST /hotels/add



GET /hotels/{id}



DELETE /hotels/{id}



GET /hotels/all



```
curl -X 'GET' \
  'http://localhost:8090/hotels/1' \
  -H 'accept: */*'

```



Request URL

```
http://localhost:8090/hotels/1
```

Server response

Code Details

200

Response body

```
{
  "hotelId": 1,
  "city": "chennai",
  "hotelName": "Taj",
  "address": "street 1",
  "description": "jhjvjh",
  "avgRatePerDay": 800,
  "email": "string@gmail.com",
  "phone1": "5262932503",
  "phone2": "4598950733",
  "website": "strings",
  "hotelRooms": [
    {
      "roomId": 1,
      "roomNo": "1",
      "roomType": "AC",
      "ratePerDay": 900,
      "isavailable": false
    },
    {
      "roomId": 2,
      "roomNo": "2",
      "roomType": "AC",
      "ratePerDay": 900,
      "isavailable": true
    },
  ],
  "hotelId": 2
}
```



Download

booking-controller



PUT /booking/update



PUT /booking/cancel/{id}



POST /booking/add



GET /booking/{id}



GET /booking/all



POST /booking/add

Parameters

Cancel

Reset

No parameters

Request body required

application/json

```
{
  "hotelId": 1,
  "roomId": 2,
  "userId": 3,
  "bookingDate": "2024-05-06",
  "bookedFromDate": "2024-05-08",
  "bookedToDate": "2024-05-09",
  "noOfAdults": 1,
  "noOfChildren": 0,
  "amount": 700,
  "status": "Active"
}
```

Request URL

http://localhost:9000/booking/add

Server response

Code	Details
------	---------

201

Undocumented

Response body

```
{
  "bookingId": 21,
  "hotelId": 1,
  "roomId": 2,
  "userId": 3,
  "bookingDate": "2024-05-06",
  "bookedFromDate": "2024-05-08",
  "bookedToDate": "2024-05-09",
  "noOfAdults": 1,
  "noOfChildren": 0,
  "amount": 700,
  "status": "Active"
}
```



Download

Curl

```
curl -X 'GET' \  
  'http://localhost:9000/booking/21' \  
  -H 'accept: */*'
```



Request URL

http://localhost:9000/booking/21

Server response

Code

Details

200

Response body

```
{  
  "customerName": "Virat",  
  "mobile": "3698521470",  
  "hotelName": "Taj",  
  "roomNo": "2",  
  "roomType": "AC",  
  "amount": 700,  
  "hotelPhone1": "5262932503",  
  "hotelPhone2": "4598950733",  
  "bookingDate": "2024-05-06",  
  "bookedFromDate": "2024-05-08",  
  "bookedToDate": "2024-05-09",  
  "noOfAdults": 1,  
  "noOfChildren": 0,  
  "status": "Active"  
}
```



Download

payment-controller



POST /payment/make



GET /payment/{id}



GET /payment/all



POST /payment/make



Parameters

Cancel

Reset

No parameters

Request body required

application/json



```
{  
  "bookingId": 21,  
  "pamount": 700  
}
```

Package Explorer JUnit x

finished after 10.138 seconds

Runs: 5/5 Errors: 0 Failures: 0

UserServiceTest [Runner: JUnit 5] (0.168 s)

- getUserDetailsTest() (0.100 s)
- deleteUserExceptionTest() (0.025 s)
- testGetUserDetailsException() (0.007 s)
- deleteUserTest() (0.012 s)
- getAllUserDetailsTest() (0.010 s)

Failure Trace

Boot Dashboard x

UserServiceTest.java x

```
1 package com.hotelbooking.userservice.service;
2
3 import static org.junit.jupiter.api.Assertions.assertEquals;
22
23 @SpringBootTest(properties = "eureka.client.enabled=false")
24 public class UserServiceTest {
25
26     @InjectMocks
27     private UserService userService = new UserServiceImpl();
28
29     @Mock
30     private UserDao userDao;
31
32     @Test
33     public void getUserDetailsTest() {
34
35         User user = new User();
36
37         user.setUserName("Rohit");
38         user.setMobile("9856231470");
39         user.setUserId(100);
40         user.setEmail("gdcgf@gmail.com");
41         user.setAddress("ehgfchkuyjh");
42         user.setRole("customer");
43
44         when(userDao.findById(100)).thenReturn(Optional.of(user));
45
46         User testuser = userService.getUserById(100);
47
48         assertEquals("Rohit", testuser.getUserName());
49
50     }
```

Package Explorer JUnit x

Finished after 9.919 seconds

Runs: 8/8 Errors: 0 Failures: 0

HotelServiceTest [Runner: JUnit 5] (0.183 s)

- deleteHotelNotFoundTest() (0.076 s)
- createHotelTest() (0.032 s)
- getHotelByIdTest() (0.005 s)
- getHotelByIdNotFoundTest() (0.006 s)
- getAllHotelsTest() (0.004 s)
- deleteHotelTest() (0.008 s)
- updateHotelTest() (0.008 s)
- updateHotelNotFoundTest() (0.011 s)

Failure Trace

Root Dashboard x

UserServiceTest.java HotelServiceTest.java x

```
1 package com.hotelbooking.hotel.service;
2
3 import static org.junit.jupiter.api.Assertions.assertEquals;
23
24 @SpringBootTest(properties = "eureka.client.enabled=false")
25 public class HotelServiceTest {
26
27     @InjectMocks
28     private HotelService hotelService = new HotelServiceImpl();
29
30     @Mock
31     private HotelDao hotelDao;
32
33     @Test
34     public void getAllHotelsTest() {
35         List<Hotel> hotels = new ArrayList<>();
36         hotels.add(new Hotel(1, "New York", "Hotel California", "Some address", "Nice place", 200.0,
37             "contact@hotelcal.com", "1234567890", "0987654321", "www.hotelcal.com", new ArrayList<Room>()));
38         hotels.add(new Hotel(2, "Los Angeles", "Hotel New York", "Another address", "Great stay", 150.0,
39             "info@hotelnyc.com", "1112223330", "4445556661", "www.hotelnyc.com", new ArrayList<Room>()));
40
41         when(hotelDao.findAll()).thenReturn(hotels);
42
43         List<Hotel> result = hotelService.getAllHotels();
44
45         assertEquals(2, result.size());
46         assertEquals("New York", result.get(0).getCity());
47         verify(hotelDao).findAll();
48     }
49 }
```

Package Explorer JUnit x

nished after 10.278 seconds

Runs: 8/8 Errors: 0 Failures: 0

RoomServiceTest [Runner: JUnit 5] (0.280 s)

- getRoomByIdTest() (0.136 s)
- createRoomHotelNotFoundTest() (0.014 s)
- getRoomByIdNotFoundTest() (0.011 s)
- updateRoomNotFoundTest() (0.009 s)
- updateRoomTest() (0.013 s)
- createRoomTest() (0.010 s)
- getAllRoomsTest() (0.036 s)
- deleteRoomTest() (0.021 s)

Failure Trace

Boot Dashboard x

```
UserServiceTest.java HotelServiceTest.java RoomServiceTest.java x
1 package com.hotelbooking.hotel.service;
2
3 import static org.junit.jupiter.api.Assertions.assertEquals;
4
5 @SpringBootTest(properties = "eureka.client.enabled=false")
6 public class RoomServiceTest {
7
8     @InjectMocks
9     private RoomService roomService = new RoomServiceImpl();
10
11     @Mock
12     private HotelDao hotelDao;
13
14     @Mock
15     private RoomDao roomDao;
16
17     @Test
18     public void getAllRoomsTest() {
19         List<Room> rooms = new ArrayList<>();
20         rooms.add(new Room(1, "101", "Single", 100.0, true, new Hotel()));
21         rooms.add(new Room(2, "102", "Double", 150.0, true, new Hotel()));
22
23         when(roomDao.findAll()).thenReturn(rooms);
24
25         List<Room> result = roomService.getAllRooms();
26
27         assertEquals(2, result.size());
28         verify(roomDao).findAll();
29     }
30
31     @Test
32     public void getRoomByIdTest() {
33         // ...
34     }
35 }
```

Package Explorer JUnit ×

Finished after 10.441 seconds

Runs: 4/4 Errors: 0 Failures: 0

BookingServiceTest [Runner: JUnit 5] (0.178 s)

- addBookingRoomNotAvailableTest() (0.120 s)
- getBookingDetailsByIdTest() (0.023 s)
- cancelBookingDeadlinePassedTest() (0.006 s)
- testAddBooking() (0.013 s)

Failure Trace

Boot Dashboard ×

Type tags, projects, or working set names to match (incl. * and ? wild)

> local

```
UserServiceTest.java HotelServiceTest.java RoomServiceTest.java BookingServiceTest.java ×
1 package com.hotelbooking.bookingservice.service;
2
3 import static org.junit.jupiter.api.Assertions.assertEquals;
26
27 @SpringBootTest(properties = "eureka.client.enabled=false")
28 public class BookingServiceTest {
29
30     @InjectMocks
31     private BookingServiceImpl bookingService;
32
33     @Mock
34     private BookingDao bookingDao;
35
36     @Mock
37     private UserServiceConsumer userService;
38
39     @Mock
40     private HotelServiceConsumer hotelService;
41
42     @Test
43     public void testAddBooking() {
44
45         User user = new User(1, "Rohit", "1234567890@gmail", "6563133", "user", "84121211", "ghgvvnvn");
46         Room room = new Room(1, "101", "Deluxe", 100.0, true, null);
47         Booking booking = new Booking(1, 1, 1, 1, LocalDate.now(), LocalDate.now().plusDays(2),
48             LocalDate.now().plusDays(5), 2, 1, 200.0, "Booked");
49
50         when(userService.getUserById(1)).thenReturn(user);
51         when(hotelService.getRoomById(1)).thenReturn(room);
52
53         // used to return the updated room
54         when(hotelService.updateRoom(any(Room.class))).thenAnswer(invocation -> invocation.getArgument(0));
55         when(bookingDao.save(any(Booking.class))).thenReturn(booking);
56
57         Booking savedBooking = bookingService.addBooking(booking);
58
59         assertEquals("Booked", savedBooking.getStatus());
60         assertFalse(room.isIsavailable());
61     }
62
63     @Test
64     void addBookingRoomNotAvailableTest() {
65         Booking booking = new Booking(1, 1, 1, 1, LocalDate.now(), LocalDate.now().plusDays(5),
```

Package Explorer JUnit x

Finished after 9.591 seconds

Runs: 6/6 Errors: 0 Failures: 0

PaymentServiceTest [Runner: JUnit 5] (0.167 s)

- testMakePaymentPaymentAlreadyExistException() (0.091 s)
- testMakePayment() (0.015 s)
- testViewPaymentsByIdPaymentNotFoundException() (0.008 s)
- testMakePaymentAmountMismatchException() (0.012 s)
- testViewAllPayments() (0.009 s)
- testViewPaymentsById() (0.006 s)

Failure Trace

Boot Dashboard x

Type tags, projects, or working set names to match (incl. * and ? wild)

> local

```
UserServiceTest.java HotelServiceTest.java RoomServiceTest.java BookingServiceTest.java Payr
1 package com.hotelbooking.paymentservice.service;
2
3 import static org.junit.jupiter.api.Assertions.assertEquals;
4
22
23 @SpringBootTest(properties = "eureka.client.enabled=false")
24 public class PaymentServiceTest {
25
26     @Mock
27     private PaymentDao paymentDao;
28
29     @Mock
30     private BookingServiceConsumer bookingService;
31
32     @InjectMocks
33     private PaymentServiceImpl paymentService;
34
35     @Test
36     public void testMakePayment() {
37         Payments payment = new Payments();
38         payment.setBookingId(1);
39         payment.setPAmount(100.0);
40
41         BookingResponse bookingResponse = new BookingResponse();
42         bookingResponse.setAmount(100.0);
43
44         when(bookingService.getBookingDetailsById(1)).thenReturn(bookingResponse);
45         when(paymentDao.existsByBookingId(1)).thenReturn(false);
46         when(paymentDao.save(payment)).thenReturn(payment);
47
48         PaymentResponse response = paymentService.makePayment(payment);
49
50         assertEquals("Payment Successful", response.getMessage());
51         assertEquals(bookingResponse, response.getBookingResponse());
52     }
53
54     @Test
55     public void testMakePaymentPaymentAlreadyExistException() {
56         Payments payment = new Payments();
57         payment.setBookingId(1);
58
59         when(paymentDao.existsByBookingId(1)).thenReturn(true);
60
61         assertThrows(PaymentFailedException.class, () -> {
```

user-service - UserServiceApplication [Spring Boot App] D:\Work\Software\sts-4.22.0.RELEASE

2024-05-06T13:58:15.641+05:30 INFO 18416 --- [user-service] [tr

2024-05-06T14:02:27.656+05:30 INFO 18416 --- [user-service] [n:

Hibernate: select u1_0.user_id,u1_0.address,u1_0.email,u1_0.mob:

2024-05-06T14:03:15.646+05:30 INFO 18416 --- [user-service] [tr

2024-05-06T14:08:15.651+05:30 INFO 18416 --- [user-service] [tr

2024-05-06T14:13:15.675+05:30 INFO 18416 --- [user-service] [tr

2024-05-06T14:18:15.677+05:30 INFO 18416 --- [user-service] [tr

Hibernate: select u1_0.user_id,u1_0.address,u1_0.email,u1_0.mob:

2024-05-06T14:23:15.685+05:30 INFO 18416 --- [user-service] [tr

2024-05-06T14:28:15.694+05:30 INFO 18416 --- [user-service] [tr

2024-05-06T14:33:15.706+05:30 INFO 18416 --- [user-service] [tr

Hibernate: select u1_0.user_id,u1_0.address,u1_0.email,u1_0.mob:

2024-05-06T14:38:15.712+05:30 INFO 18416 --- [user-service] [tr

2024-05-06T14:43:15.742+05:30 INFO 18416 --- [user-service] [tr