

CAPSTONE PROJECT

ONLINE SALOON SERVICE APP

TEAM-C

BATCH-8 JAVA-J2EE

Name:RohithkumarSurisetty

E-mail:rohithsurisetty@gmail.com

Date: 2September,2024

Batch Mentor: Mr.Ramakrishna Sir

Problem Statement:

Develop a distributed Online Salon Service using Spring Microservices where:

- Admin can manage services, staff, bookings, and customers.
- Users can browse available services, book appointments, and manage their bookings.

Admin should be able to:

1.Admin Dashboard:

- A consolidated view of all operations handled by the Admin.
- Provides access to manage services, staff, bookings, and customers

2.Service Management should be able to:

- Admin can perform CRUD operations on salon services through the Service Management Microservice.

3.Staff Management should be able to:

- Admin can manage staff information and their service assignments through the Staff Management Microservice.

4.Booking Management should be able to:

- Admin can monitor and modify bookings through the Booking Management Microservice.
- Admin can reassign staff to bookings and update booking statuses.

5.Customer Management should be able to:

- Admin can view and manage customer profiles and booking history.

6.User should be able to:

1.User Registration & Authentication:

- Users can register and log in using the Authentication Service.
- JWT tokens are used to secure API access.

2.Service Browsing:

- Users can browse available salon services through the Service Management Microservice.

3.Appointment Booking:

- Users can book appointments, specifying their preferred service, date, time, and staff.
- The Booking Management Microservice ensures that the booking is valid and staff are available.

4.Booking Management:

- Users can view, modify, or cancel their appointments via the Booking Management Microservice.
- Notifications are sent for booking confirmations and reminders.

Use Case Diagram for Online Salon service:

Online Salon Service:

Customer | -----> (Book Appointment)
-----> (View Services)

Staff | -----> (Manage Schedule)
-----> (Update Availability)

Admin -----> (Manage Users)
-----> (Manage Staff)
 (Booking)
 (Manage Services)

UML Flowchart for Online Salon Service

Customer -----> (Book Appointment) -----> BookingService

Staff -----> (Manage Schedule) -----> StaffService

Admin -----> (Manage Users) -----> UserService

SpringBootApp-Backend:

Service Registry:

Running the Application as SpringBoot Application:

```

2024-09-01T20:28:29, 518-07:00 INFO 14852 --- [Service-registry] [main] o.a.c.c.C.[Tomcat].[localhost].[]: [Service-registry]
2024-09-01T20:28:29, 519-07:00 INFO 14852 --- [Service-registry] [main] w.s.c.ServletWebServerApplicationContext
2024-09-01T20:28:30, 573-07:00 INFO 14852 --- [Service-registry] [main] c.n.d.provider.DiscoveryJerseyProvider
2024-09-01T20:28:30, 575-07:00 INFO 14852 --- [Service-registry] [main] c.n.d.provider.DiscoveryJerseyProvider
2024-09-01T20:28:30, 787-07:00 INFO 14852 --- [Service-registry] [main] c.n.d.provider.DiscoveryJerseyProvider
2024-09-01T20:28:31, 986-07:00 WARN 14852 --- [Service-registry] [main] c.n.d.provider.DiscoveryJerseyProvider
2024-09-01T20:28:32, 031-07:00 INFO 14852 --- [Service-registry] [main] c.n.d.provider.DiscoveryJerseyProvider
2024-09-01T20:28:32, 032-07:00 INFO 14852 --- [Service-registry] [main] c.n.d.provider.DiscoveryJerseyProvider
2024-09-01T20:28:32, 034-07:00 INFO 14852 --- [Service-registry] [main] c.n.d.provider.DiscoveryJerseyProvider
2024-09-01T20:28:32, 133-07:00 INFO 14852 --- [Service-registry] [main] c.n.eureka.DefaultEurekaServerContext
2024-09-01T20:28:32, 136-07:00 INFO 14852 --- [Service-registry] [main] c.n.eureka.cluster.PeerEurekaNodes
2024-09-01T20:28:32, 290-07:00 INFO 14852 --- [Service-registry] [main] c.n.d.provider.DiscoveryJerseyProvider
2024-09-01T20:28:32, 291-07:00 INFO 14852 --- [Service-registry] [main] c.n.d.provider.DiscoveryJerseyProvider
2024-09-01T20:28:32, 291-07:00 INFO 14852 --- [Service-registry] [main] c.n.d.provider.DiscoveryJerseyProvider
2024-09-01T20:28:32, 325-07:00 INFO 14852 --- [Service-registry] [main] c.n.eureka.cluster.PeerEurekaNodes
2024-09-01T20:28:32, 338-07:00 INFO 14852 --- [Service-registry] [main] c.n.e.registry.AbstractInstanceRegistry
2024-09-01T20:28:32, 338-07:00 INFO 14852 --- [Service-registry] [main] c.n.eureka.DefaultEurekaServerContext
2024-09-01T20:28:32, 349-07:00 INFO 14852 --- [Service-registry] [main] o.s.b.a.e.web.EndpointLinksResolver
2024-09-01T20:28:32, 412-07:00 INFO 14852 --- [Service-registry] [main] s.c.n.e.s.EurekaServiceRegistry
2024-09-01T20:28:32, 426-07:00 INFO 14852 --- [Service-registry] [Thread-9] o.s.c.n.e.server.EurekaServerBootstrap
2024-09-01T20:28:32, 427-07:00 INFO 14852 --- [Service-registry] [Thread-9] o.s.c.n.e.server.EurekaServerBootstrap
2024-09-01T20:28:32, 427-07:00 INFO 14852 --- [Service-registry] [Thread-9] c.n.e.r.PeerAwareInstanceRegistryImpl
2024-09-01T20:28:32, 427-07:00 INFO 14852 --- [Service-registry] [Thread-9] c.n.e.r.PeerAwareInstanceRegistryImpl
2024-09-01T20:28:32, 431-07:00 INFO 14852 --- [Service-registry] [Thread-9] e.s.EurekaServerInitializerConfiguratio
2024-09-01T20:28:32, 437-07:00 INFO 14852 --- [Service-registry] [main] o.s.b.w.embedded.tomcat.TomcatWebServer
2024-09-01T20:28:32, 438-07:00 INFO 14852 --- [Service-registry] [main] com.example.ServiceRegistryApplication
2024-09-01T20:28:32, 458-07:00 INFO 14852 --- [Service-registry] [on(4)-127.0.0.1] o.a.c.c.C.[Tomcat].[localhost].[]: [on(4)-127.0.0.1] o.s.web.servlet.DispatcherServlet
2024-09-01T20:28:33, 167-07:00 INFO 14852 --- [Service-registry] [on(4)-127.0.0.1] o.s.web.servlet.DispatcherServlet
2024-09-01T20:28:33, 169-07:00 INFO 14852 --- [Service-registry] [on(4)-127.0.0.1] o.s.web.servlet.DispatcherServlet

```

Eureka server:

System Status

Environment	test	Current time	2024-09-01T20:42:02 -0700
Data center	default	Uptime	00:21
		Lease expiration enabled	true
		Renews threshold	10
		Renews (last min)	20

DS Replicas

localhost

Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
APICATEGWY	(1)	(1)	UP (1) DESKTOP-FK4390F APIGateway:8004

All microservices are existed with Eureka server:

The screenshot shows a web browser window with the URL localhost:8761. At the top, there are several tabs: (8) W, Eurek, Until, JAVA, Swag, Swag, Swag, Chat, local, Meet. Below the tabs, the page header includes 'Renews threshold' (10) and 'Renews (last min)' (20). The main content area is titled 'DS Replicas' and shows a table of registered instances:

Application	AMIs	Availability Zones	Status
APIGATEWAY	n/a (1)	(1)	UP (1) - DESKTOP-5KL380E:APIGateway:9094
AUTHENTICATIONSECURITY	n/a (1)	(1)	UP (1) - DESKTOP-5KL380E:authenticationSecurity:9095
SERVICEMANAGEMENTMICROSERVICE	n/a (1)	(1)	UP (1) - DESKTOP-5KL380E:ServiceManagementMicroservice:9091
STAFFMANAGEMENTMICROSERVICE	n/a (1)	(1)	UP (1) - DESKTOP-5KL380E:StaffManagementMicroservice:9092
USERMANAGEMENTMICROSERVICE	n/a (1)	(1)	UP (1) - DESKTOP-5KL380E:UserManagementMicroservice:9093

Below this, a section titled 'General Info' shows a table of system properties:

Name	Value
Type here to search	

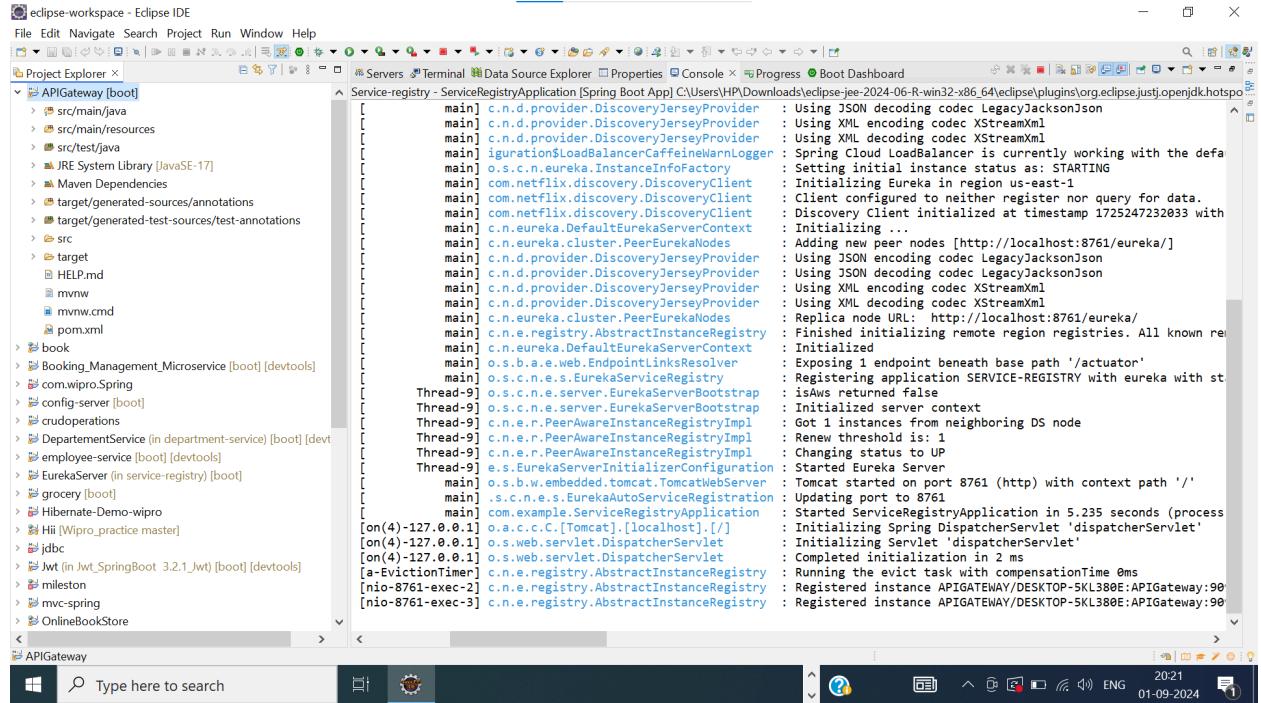
The status bar at the bottom indicates the date and time as 01-09-2024 12:17.

All microservices are registered in eclipse BootDashboard :

The screenshot shows the Eclipse IDE interface with the title 'eclipse-workspace - Spring/src/com/wipro/spring/Welcome.java - Eclipse IDE'. The 'Boot Dashboard' view is active in the center. The left side shows the 'Project Explorer' with multiple projects like APIGateway, Booking_Management_Microservice, config-server, DepartementService, employee-service, EurekaServer, Jwt, grocery, project-salon, salon-api-gateway, salon-booking-service, salon-service, salon-service-registry, salon-user-service, Service_Management_Microservice, Service-registry, springboot-demo, Staff_Management_Microservice, and User_Management_Microservice. The right side shows the 'Servers' view with a single entry for 'local'. The status bar at the bottom indicates the date and time as 01-09-2024 20:37.

APIGateway:

Running as the spring boot application:



The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows the project structure for "APIGateway [boot]".
- Servers View:** Displays logs for the "Service-registry - ServiceRegistryApplication [Spring Boot App]". The logs include:

```
[main] c.n.d.provider.DiscoveryJerseyProvider : Using JSON decoding codec LegacyJacksonJson
[main] c.n.d.provider.DiscoveryJerseyProvider : Using XML encoding codec XStreamXml
[main] c.n.d.provider.DiscoveryJerseyProvider : Using XML decoding codec XStreamXml
[main] igureationsLoadBalancerCaffeineWarnLogger : Spring Cloud LoadBalancer is currently working with the defa
[main] o.s.c.n.eureka.InstanceInfoFactory : Setting initial instance status as: STARTING
[main] com.netflix.discovery.DiscoveryClient : Initializing Eureka in region us-east-1
[main] com.netflix.discovery.DiscoveryClient : Client configured to neither register nor query for data.
[main] c.n.eureka.DefaultEurekaServerContext : Discovery Client initialized at timestamp 1725247232033 with
[main] c.n.eureka.cluster.PeerEurekaNodes : Initializing ...
[main] c.n.d.provider.DiscoveryJerseyProvider : Adding new peer nodes [http://localhost:8761/eureka/]
[main] o.s.c.n.eureka.InstanceInfoFactory : Using JSON encoding codec LegacyJacksonJson
[main] c.n.d.provider.DiscoveryJerseyProvider : Using XML decoding codec XStreamXml
[main] o.s.c.n.eureka.InstanceInfoFactory : Using XML encoding codec XStreamXml
[main] c.n.d.provider.DiscoveryJerseyProvider : Replica node URL: http://localhost:8761/eureka/
[main] c.n.e.registry.AbstractInstanceRegistry : Finished initializing remote region registries. All known re
[main] c.n.eureka.DefaultEurekaServerContext : Initialized
[main] c.n.e.registry.ExposingEndpointLinksResolver : Exposing 1 endpoint beneath base path '/actuator'
[main] o.s.c.n.e.web.EndpointLinksResolver : Registering application SERVICE-REGISTRY with eureka with st
[main] o.s.c.n.e.server.EurekaServerBootstrap : isAws returned false
[Thread-9] o.s.c.n.e.server.EurekaServerBootstrap : Initialized server context
[Thread-9] c.n.e.r.PeerAwareInstanceRegistryImpl : Got 1 instances from neighboring DS node
[Thread-9] c.n.e.r.PeerAwareInstanceRegistryImpl : Renew threshold is: 1
[Thread-9] c.n.e.r.PeerAwareInstanceRegistryImpl : Changing status to UP
[Thread-9] e.s.EurekaServerInitializerConfiguration : Started Eureka Server
[main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port 8761 (http) with context path ''
[main] s.c.n.e.s.EurekaAutoServiceRegistration : Updating port to 8761
[main] com.example.ServiceRegistryApplication : Started ServiceRegistryApplication in 5.235 seconds (process
[on(4)-127.0.0.1] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring DispatcherServlet 'dispatcherServlet'
[on(4)-127.0.0.1] o.s.web.servlet.DispatcherServlet : Initializing Servlet 'dispatcherServlet'
[on(4)-127.0.0.1] o.s.web.servlet.DispatcherServlet : Completed initialization in 2 ms
[a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 0ms
[nio-8761-exec-2] c.n.e.registry.AbstractInstanceRegistry : Registered instance APIGATEWAY/DESKTOP-5KL380E:APIGateway:90
[nio-8761-exec-3] c.n.e.registry.AbstractInstanceRegistry : Registered instance APIGATEWAY/DESKTOP-5KL380E:APIGateway:90
```

User_Management_Microservice:

Running as the spring boot application in eclipse:

```

eclipse-workspace - Eclipse IDE
File Edit Navigate Search Project Run Window Help
Project Explorer X Servers Terminal Data Source Explorer Properties Console Progress Boot Dashboard
Service-registry - ServiceRegistryApplication [Spring Boot App] C:\Users\Wipro\Downloads\edpse-eje-2024-04-Bwv32-x86_64\plugins\org.eclipse.just.jopenjdk.hotspot.jdk17
main] i guration$LoadBalancerCaffeineWarnLogger : Spring Cloud LoadBalancer is currently working with the default caci
main] o.s.c.n.eureka.InstanceInfoFactory : Setting initial instance status as: STARTING
main] com.netflix.discovery.DiscoveryClient : Initializing Eureka in region us-east-1
main] com.netflix.discovery.DiscoveryClient : Client configured to neither register nor query for data.
main] c.n.eureka.DefaultEurekaServerContext : Discovery Client initialized at timestamp 1725247232033 with initia
main] c.n.d.provider.DiscoveryJerseyProvider : Initializing ...
main] c.n.d.provider.DiscoveryJerseyProvider : Adding new peer nodes [http://localhost:8761/eureka/]
main] c.n.d.provider.DiscoveryJerseyProvider : Using JSON encoding codec LegacyJacksonJson
main] c.n.d.provider.DiscoveryJerseyProvider : Using XML encoding codec XStreamXml
main] c.n.d.provider.DiscoveryJerseyProvider : Using XML decoding codec XStreamXml
main] c.n.eureka.cluster.PeerEurekaNodes : Replica node URL: http://localhost:8761/eureka/
main] c.n.e.registry.AbstractInstanceRegistry : Finished initializing remote region registries. All known remote re
main] c.n.eureka.DefaultEurekaServerContext : Initialized
main] o.s.b.a.e.web.EndpointLinksResolver : Exposing 1 endpoint beneath base path '/actuator'
main] o.s.c.n.e.EurekaServiceRegistry : Registering application SERVICE-REGISTRY with eureka with status UP
main] o.s.c.n.e.server.EurekaServerBootstrap : isAws returned false
Thread-9 : Initialized server context
Thread-9 : Got 1 instances from neighboring DS node
Thread-9 : Renew threshold is: 1
Thread-9 : Changing status to UP
Thread-9 : Started Eureka Server
main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port 8761 (http) with context path '/'
main] com.example.ServiceRegistryApplication : Upgrading port to 8761
-127.0.0.1 : Started ServiceRegistryApplication in 5.235 seconds (process running)
o.a.c.c.C.[Tomcat].[localhost].[] : Initializing Spring DispatcherServlet 'dispatcherServlet'
-127.0.0.1 : Initializing Servlet 'dispatcherServlet'
-127.0.0.1 : Completed initialization in 2 ms
c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 0ms
761-exec-2 : Registered instance APIGATEWAY/DESKTOP-5KL380E:APIGateway:9094 with
761-exec-3 : Registered instance APIGATEWAY/DESKTOP-5KL380E:APIGateway:9094 with
761-exec-8 : Registered instance USERMANAGEMENTMICROSERVICE/DESKTOP-5KL380E:UserM
761-exec-9 : Registered instance USERMANAGEMENTMICROSERVICE/DESKTOP-5KL380E:UserM
ctionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 0ms

```

User_Management_Microservice 2022 01-09-2024 ENG

Data base for user_management_microservice in sql workbench:

ID	Email	Mobile No	Name	Password	Role	Sex
1	rohit.sursetti@gmail.com	12	Rohith	Rohith123	USER	MALE
2	sursetti@gmail.com	88999776	Ganesh	Rohit123	USER	MALE
4	sursetti45@gmail.com	889997745	Ramakrishna	Ro123	USER	MALE
5	wipro5642@gmail.com	7893966136	Balu	Balu@123	ADMIN	MALE
6	virat122@gmail.com	6454646464	virat	HULL	HULL	MALE

MySQL Workbench 2022 01-09-2024 ENG

Crud operations using post for user management microservice:

The screenshot shows the Postman application interface. At the top, there are navigation links for Home, Workspaces, and Explore, along with a search bar labeled "Search Postman". On the right, there are buttons for "Sign In" and "Create Account". Below the header, a message states: "You are using the Lightweight API Client, sign in or create an account to work with collections, environments and unlock all free features in Postman." A sidebar on the left shows a list of collections: "POST http://localhost:9093/api/users", "HTTP http://localhost:9093/api/users", and "Params Authorization Headers (8) Body Pre-request Script Tests Settings". The main area displays a POST request to "http://localhost:9093/api/users". The "Body" tab is selected, showing a JSON payload:

```

1 {
2   "id":1876,
3   "name":"virat",
4   "email":"virat122@gmail.com",
5   "mobileNo":6454646464,
6   "password":"jjjj",
7   "sex":"MALE",

```

The "Send" button is highlighted in blue. Below the request, the response status is shown as "Status: 200 OK Time: 283 ms Size: 283 B". The response body is also displayed in JSON format:

```

1 {
2   "id": 6,
3   "name": "virat",
4   "email": "virat122@gmail.com",
5   "mobileNo": "6454646464",
6   "password": "jjjj",

```

The bottom of the screen shows a taskbar with various icons and the system clock indicating "11:57 30-08-2024".

By using Swagger:

The screenshot shows the Swagger UI interface for the "User Management Microservice API". The title is "User Management Microservice API 1.0 OAS 3.0". Below the title, it says "API for managing user profiles and authentication in the Online Salon Service" and "Contact Rohithkumar Surisetti".

The "Servers" section shows a dropdown menu with the URL "http://localhost:9093 - Generated server url".

The main content area is titled "User Management APIs for managing user profiles and authentication". It lists four endpoints:

- GET /api/users/{id}** Get user by ID
- PUT /api/users/{id}** Update an existing user
- DELETE /api/users/{id}** Delete a user by ID
- GET /api/users** Get all users

The bottom of the screen shows a taskbar with various icons and the system clock indicating "21:20 01-09-2024".

Curl

```
curl -X 'GET' \
'http://localhost:9093/api/users/2' \
-H 'accept: */*'
```

Request URL

```
http://localhost:9093/api/users/2
```

Server response

Code	Details
200	Response body <pre>{ "id": 2, "name": "Ganesh", "email": "surisetti@gmail.com", "mobileNo": "88999776", "password": "Rohil123", "sex": "MALE", "role": "USER" }</pre> <div style="text-align: right;"> Copy Download </div> Response headers <pre>connection: keep-alive content-type: application/json date: Mon, 02 Sep 2024 03:59:31 GMT keep-alive: timeout=60 transfer-encoding: chunked</pre>

Responses



localhost:9093/swagger-ui/index.html#/User%20Management/getUserById

Request URL

```
http://localhost:9093/api/users
```

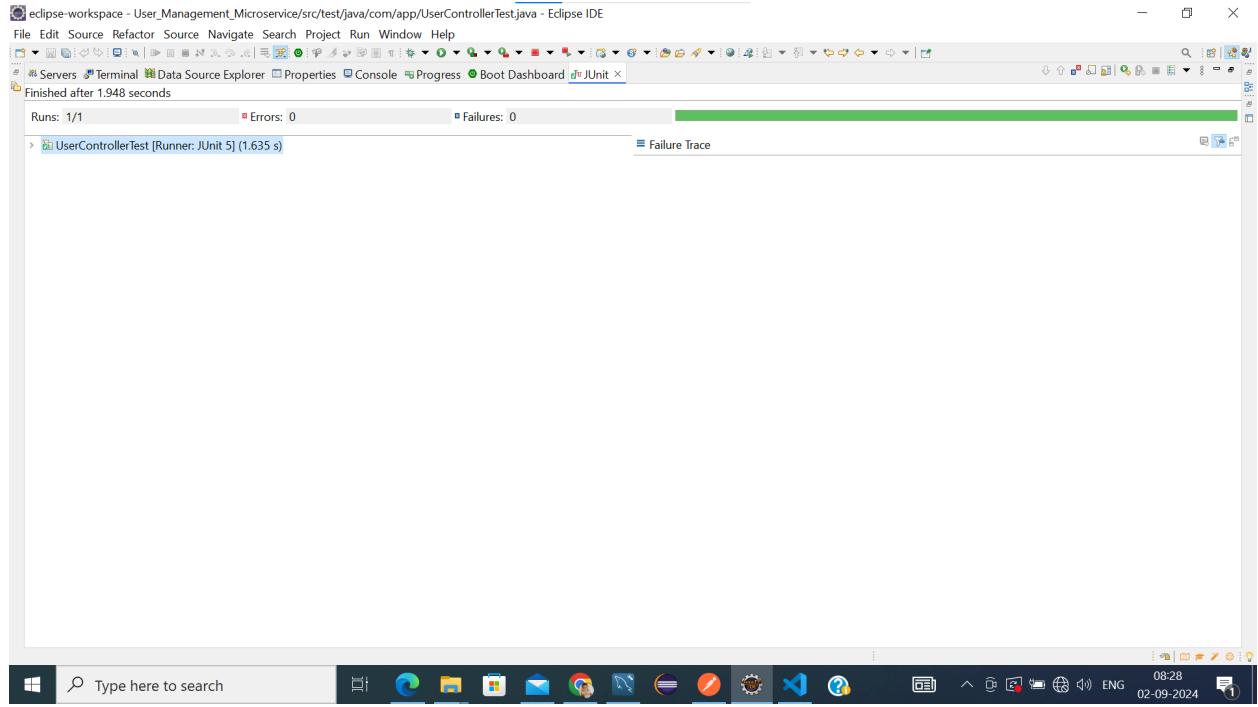
Server response

Code	Details
200	Response body <pre>[{ "id": 1, "name": "Rohith", "email": "rohit.surisetti@gmail.com", "mobileNo": "123", "password": "Rohith123", "sex": "MALE", "role": "USER" }, { "id": 2, "name": "Ganesh", "email": "surisetti@gmail.com", "mobileNo": "88999776", "password": "Rohil123", "sex": "MALE", "role": "USER" }, { "id": 4, "name": "Ramakrishna", "email": "surisetti1453@gmail.com", "mobileNo": "8899977645", "password": "Rohi23", "sex": "MALE", "role": "USER" }]</pre> <div style="text-align: right;"> Copy Download </div> Response headers <pre>connection: keep-alive content-type: application/json date: Mon, 02 Sep 2024 03:59:31 GMT keep-alive: timeout=60 transfer-encoding: chunked</pre>



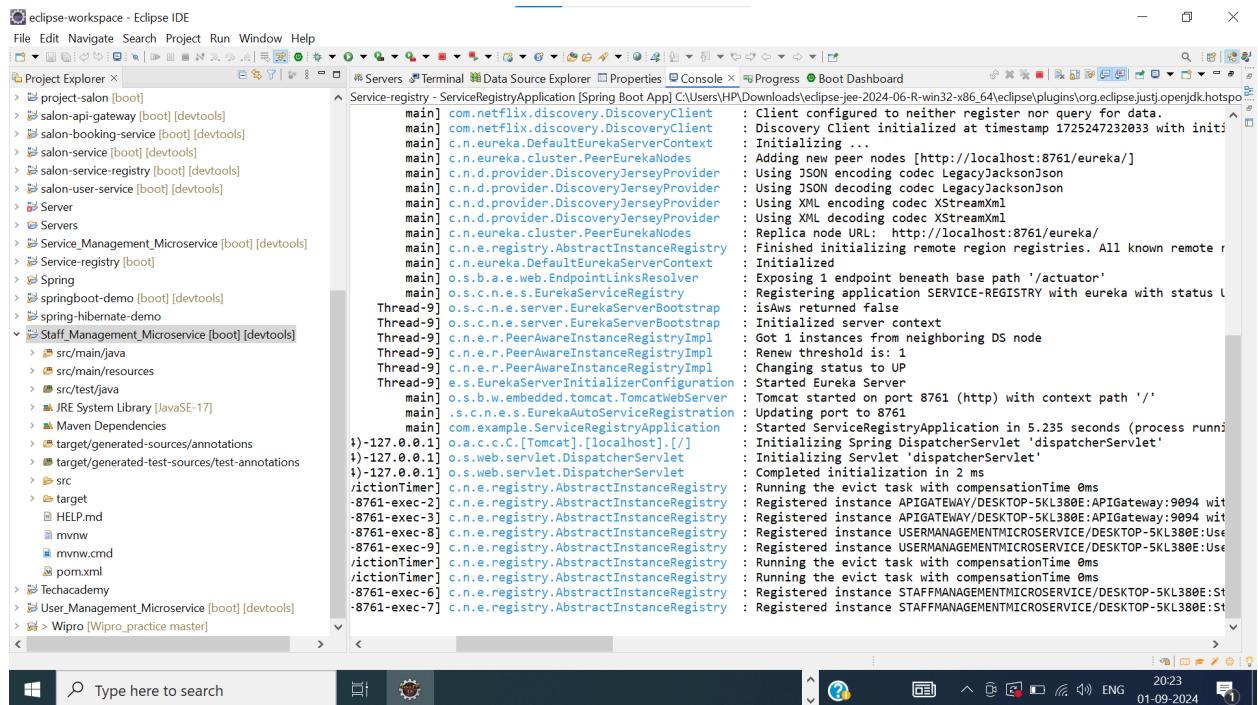
localhost:9093/swagger-ui/index.html#/User%20Management/getAllUsers

JUNIT Testing for User_Management_MicroService:



Staff_Management_Microservice:

Running as the SpringBoot Application in the Eclipse IDE:



Staff_Management_Microservice Database in the Sql Work bench:

The screenshot shows the MySQL Workbench interface. In the Navigator pane, under the schema 'staff_management', the 'Tables' section is expanded, showing the 'staff' table. A query window displays the following SQL statement and its results:

```
1 • SELECT * FROM staff_management.staff;
```

ID	details	gender	name	price
1	Senior stylist with 10 years of experience	MALE	John Doe	50
2	Vitzag	MALE	Rohith	1120.65
3	Hyderabad	MALE	virat	1234.87
4	Senior stylist with 10 years of experience	MALE	John Doe	50
*	HULL	HULL	HULL	HULL

The Action Output pane shows the history of database actions:

#	Time	Action	Message	Duration / Fetch
18	12:39:29	SELECT * FROM booking_management.booking	4 row(s) returned	0.000 sec / 0.000 sec
19	10:32:57	Apply changes to salon_db	Selected name conflicts with existing schema 'salon_db'.	
20	10:33:17	Apply changes to salon_project	Changes applied	
21	12:22:50	SELECT * FROM salon_project.user_info	4 row(s) returned	0.000 sec / 0.000 sec
22	20:53:30	SELECT * FROM user_management.users	5 row(s) returned	0.016 sec / 0.000 sec
23	21:05:11	SELECT * FROM staff_management.staff	4 row(s) returned	0.000 sec / 0.000 sec

Crud operations for staff management by using POSTMAN:

The screenshot shows the Postman interface. A POST request is being made to <http://localhost:9092/api/staff>. The request body is set to JSON and contains the following data:

```
1 {  
2   "id": 1876,  
3   "name": "virat",  
4   "details": "Hyderabad",  
5   "price": 1234.87,  
6   "gender": "MALE"  
7 }
```

The response status is 201 Created, with a time of 344 ms and a size of 246 B. The response body is identical to the request body:

```
1 {  
2   "id": 3,  
3   "name": "virat",  
4   "details": "Hyderabad",  
5   "price": 1234.87,  
6   "gender": "MALE"  
7 }
```

CRUD operations by using SWAGGER:

The screenshot shows the Swagger UI interface for the "Staff Management Microservice API". The title bar indicates "1.0 OAS 3.0". Below the title, it says "API for managing staff members and their assignments to services in the Online Salon Service". A note from "Rohithkumar - Website" suggests sending an email to Rohithkumar. The "Servers" dropdown is set to "http://localhost:9092 - Generated server url". The main section, "Staff Management APIs for managing staff members", lists four operations: GET /api/staff/{id}, PUT /api/staff/{id}, DELETE /api/staff/{id}, and GET /api/staff. The browser taskbar at the bottom shows various open tabs and system icons.

The screenshot shows the "Responses" section for the "createStaff" endpoint. It includes a "Curl" command for POSTing data to "http://localhost:9092/api/staff". The "Request URL" is "http://localhost:9092/api/staff". The "Server response" section shows a successful 201 response with a "Response body" containing JSON data for a new staff member named "John Doe" with ID 1. The browser taskbar at the bottom shows various open tabs and system icons.

The screenshot shows a web browser window with a pink header bar containing various icons. The main content area displays a Swagger UI interface. At the top left is a "Request URL" input field containing "http://localhost:9092/api/staff". Below it is a "Server response" section with tabs for "Code" and "Details", currently showing "Code". The status code is 200, and the "Response body" tab is selected, displaying the following JSON array:

```
[{"id": 1, "name": "John Doe", "details": "Senior stylist with 10 years of experience", "price": 50, "gender": "MALE"}, {"id": 2, "name": "Rohith", "details": "Vizag", "price": 1120.65, "gender": "MALE"}, {"id": 3, "name": "virat", "details": "Hyderabad", "price": 1234.87, "gender": "MALE"}, {"id": 4, "name": "John Doe", "details": "Senior stylist with 10 years of experience", "price": 50},
```

At the bottom right of the response body area are "Copy" and "Download" buttons. The browser's taskbar at the bottom shows various pinned icons and the date/time "01-09-2024 21:11".

JUNIT Testing for Staff_Management_Microservice:

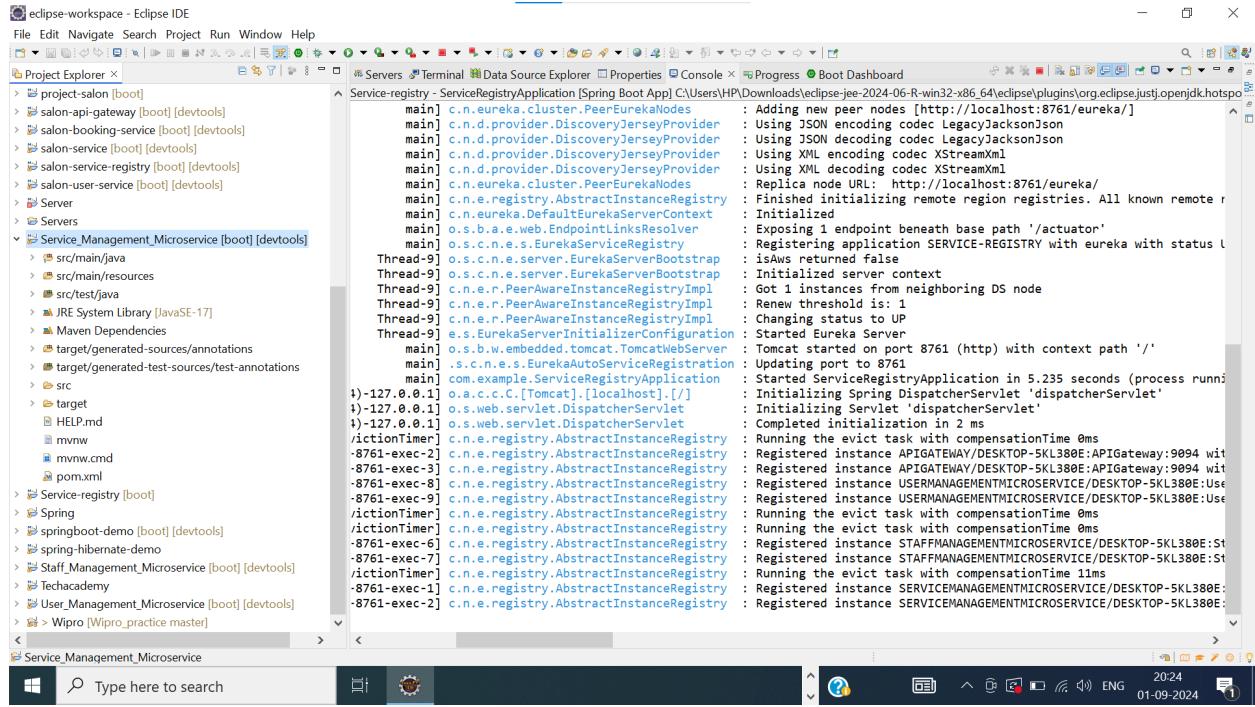
The screenshot shows the Eclipse IDE interface with the title "eclipse-workspace - Staff_Management_Microservice/src/test/java/com/app/StaffControllerTest.java - Eclipse IDE". The central workspace shows a JUnit test runner window with the following details:

- Runs: 1/1
- Errors: 0
- Failures: 0

Below this, a list shows "StaffControllerTest [Runner: JUnit 5] (1.607 s)". The bottom of the screen shows the Windows taskbar with pinned icons and the date/time "02-09-2024 08:30".

Service_Management_Microservice:

Running the Service_Management_Microservice as SpringBoot Application in Eclipse IDE:



Creating Database for Service_Management_Microservice in Sql workbench:

The screenshot shows the MySQL Workbench interface. In the top navigation bar, 'File', 'Edit', 'View', 'Query', 'Database', 'Server', 'Tools', 'Scripting', and 'Help' are visible. The 'Query' tab is active. The 'Navigator' pane on the left lists databases like 'grocery-db', 'grocery_categories', 'milestone4', 'rohitdb', 'salon', 'salon-db', 'salon_booking_detail', 'salon_detail', 'salon_project', 'salon_user_detail', and 'service_management'. Under 'service_management', the 'Tables' section shows 'services' with columns 'id', 'detail', 'name', 'price', and 'type'. A query window displays the result of the SQL command: 'SELECT * FROM service_management.services;'. The result grid shows three rows of data:

	id	detail	name	price	type
1	1	A professional haircut service	Haircut	25.5	Grooming
2	2	hyderabad	balu	1120	short
3	3	Hyderabad	virat	1234.87	fade
	NULL	NULL	NULL	NULL	NULL

The 'Output' pane at the bottom shows the action history with details like time, action, message, and duration.

Crud operations by using the POSTMAN:

The screenshot shows the Postman interface. The top navigation bar includes 'Home', 'Workspaces', 'Explore', 'Search Postman', 'Sign In', and 'Create Account'. The main area shows a POST request to 'http://localhost:9091/api/services'. The 'Body' tab is selected, showing a JSON payload:

```

1 {
2   "id": 1876,
3   "name": "virat",
4   "detail": "Hyderabad",
5   "price": 1234.87,
6   "type": "fade"
7 }
  
```

The response status is '201 Created' with a time of '587 ms' and a size of '243 B'. The response body is also displayed in JSON format:

```

1 {
2   "id": 3,
3   "name": "virat",
4   "detail": "Hyderabad",
5   "price": 1234.87,
6   "type": "fade"
7 }
  
```

CRUD Operations by using SWAGGER:

The screenshot shows the Spring Boot REST API Documentation page. At the top, there's a navigation bar with tabs like Eureka, Untitled, JAVA, Swag, and another Swag tab. Below the navigation bar, the title "Spring Boot REST API Documentation" is displayed, followed by a sub-link "/v3/api-docs". A brief description of the API follows, mentioning Rohithkumar - Website, Send email to Rohithkumar, Apache 2.0, and Detailed Spring Boot service management documentation.

Servers
http://localhost:9091 - Generated server url

Service Resource CRUD REST APIs for Service Resource

Operations

- GET** /api/services/{id} Get Service by ID
- PUT** /api/services/{id} Update Service
- DELETE** /api/services/{id} Delete Service

Windows taskbar at the bottom: Type here to search, File, Internet Explorer, File Explorer, Mail, Google Chrome, Task View, Taskbar settings, Network, Power, System, ENG, 21:22, 01-09-2024.

The screenshot shows the details for the "Service Resource" endpoint. The URL is localhost:9091/swagger-ui/index.html#/Service%20Resource/getServiceById. The "Responses" section is expanded, showing a "Curl" command:

```
curl -X 'GET' \
'http://localhost:9091/api/services/1' \
-H 'accept: */*'
```

The "Request URL" is http://localhost:9091/api/services/1. The "Server response" section shows a "Code" table with a single row for "200". The "Details" column for "200" contains the "Response body" and "Response headers".

Code

Code	Details
200	Response body { "id": 1, "name": "Haircut", "detail": "A professional haircut service", "price": 25.5, "type": "Grooming" } Response headers connection: keep-alive content-type: application/json date: Mon, 02 Sep 2024 04:31:02 GMT keep-alive: timeout=60 transfer-encoding: chunked

Windows taskbar at the bottom: Type here to search, File, Internet Explorer, File Explorer, Mail, Google Chrome, Task View, Taskbar settings, Network, Power, System, ENG, 21:31, 01-09-2024.

Request URL
http://localhost:9091/api/services

Server response

Code Details

200 Response body

```
[{"id": 1, "name": "Haircut", "detail": "A professional haircut service", "price": 25.5, "type": "Grooming"}, {"id": 2, "name": "balu", "detail": "hyderabad", "price": 1120, "type": "short"}, {"id": 3, "name": "virat", "detail": "hyderabad", "price": 1234.87, "type": "fade"}]
```

Download

Response headers

```
connection: keep-alive
content-type: application/json
```

Type here to search 21:31 01-09-2024

JUNIT Testing For Service_Management_MicroService:

eclipse-workspace - Service_Management_Microservice/src/test/java/com/app/ServiceControllerTest.java - Eclipse IDE

File Edit Source Refactor Source Navigate Search Project Run Window Help

Servers Terminal Data Source Explorer Properties Console Progress Boot Dashboard JUnit

Finished after 4.834 seconds

Runs: 1/1 Errors: 0 Failures: 0

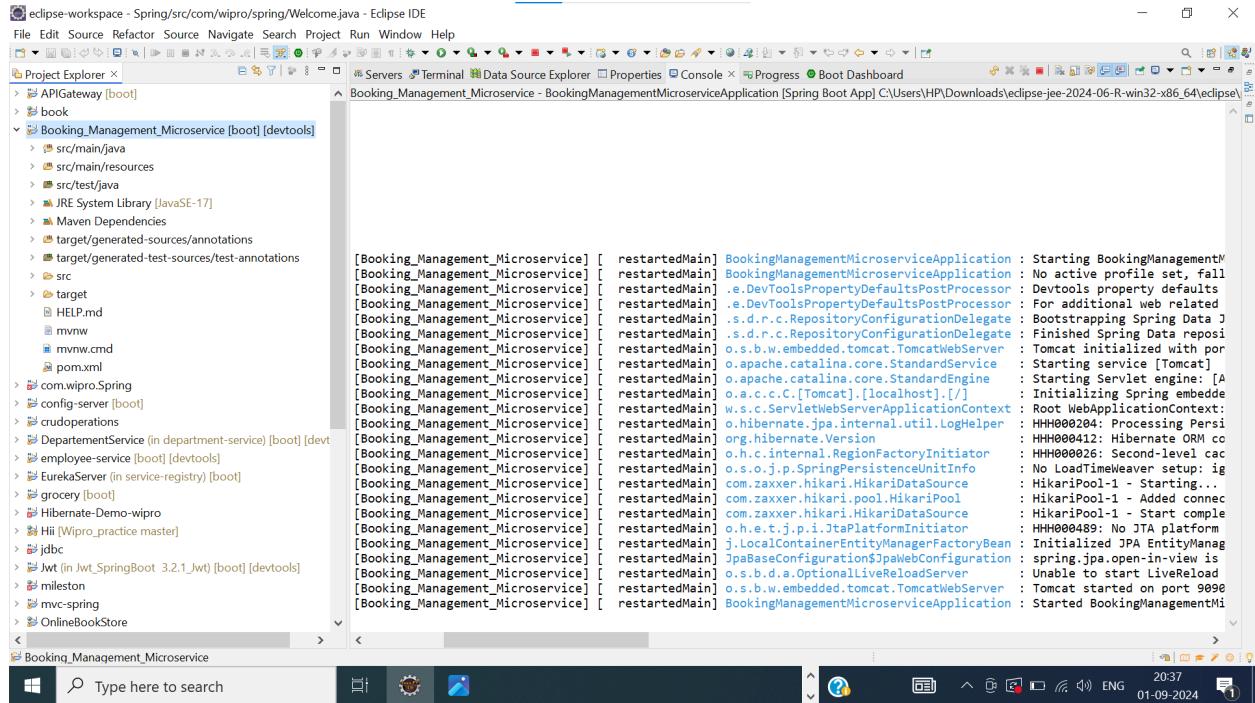
ServiceControllerTest [Runner: JUnit 5] (0.492 s)

Failure Trace

Type here to search 08:31 02-09-2024

Booking_Management_Microservice:

Booking_Management_Microservice Running as the SpringBootApplication in Eclipse IDE:



Creating DataBase for Booking_Management_Microservice in sql WorkBench:

MySQL Workbench - Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator: staff services services booking booking users staff - Table staff services booking

SQLAdditions: Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Result Grid: Filter Rows: Edit: Export/Import: Wrap Cell Content: Result Grid Form Editor

Output: Action Output

#	Time	Action	Message	Duration / Fetch
13	23:18:26	SELECT * FROM booking_management.booking	0 row(s) returned	0.000 sec / 0.000 sec
14	23:20:50	SELECT * FROM booking_management.booking	3 row(s) returned	0.000 sec / 0.000 sec
15	11:48:35	SELECT * FROM user_management.users	4 row(s) returned	0.015 sec / 0.000 sec
16	12:27:05	SELECT * FROM staff_management.staff	3 row(s) returned	0.016 sec / 0.000 sec
17	12:32:06	SELECT * FROM service_management.services	3 row(s) returned	0.000 sec / 0.000 sec
18	12:39:29	SELECT * FROM booking_management.booking	4 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

Type here to search

12:39 30-08-2024 ENG

CRUD operations by using POSTMAN:

Home Workspaces Explore Search Postman Sign In Create Account

You are using the Lightweight API Client, sign in or create an account to work with collections, environments and unlock all free features in Postman.

POST http://localhost:9090/api/bookings

HTTP: http://localhost:9090/api/bookings Save </>

POST http://localhost:9090/api/bookings Send

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies Beautify

Body: none form-data x-www-form-urlencoded raw binary JSON

```

1 {
2   "id": 1876,
3   "price": 1234.87,
4   "status": "PENDING"
5
6
7
  
```

Status: 200 OK Time: 858 ms Size: 207 B Save Response

Pretty Raw Preview Visualize JSON

```

1 {
2   "id": 4,
3   "price": 1234.87,
4   "status": "PENDING"
5
  
```

Console Not connected to a Postman account

Type here to search

12:37 30-08-2024 ENG

CRUD Operations by using SWAGGER:

The screenshot shows the Swagger UI interface for an OpenAPI definition. At the top, it displays "OpenAPI definition v0 OAS 3.0" and the URL "/v3/api-docs". Below this, there's a "Servers" section with a dropdown menu set to "http://localhost:9090 - Generated server url". The main content area is titled "booking-controller" and lists several API endpoints:

- GET** /api/bookings/{id}
- PUT** /api/bookings/{id}
- DELETE** /api/bookings/{id}
- GET** /api/bookings
- POST** /api/bookings

At the bottom of the screen, the Windows taskbar is visible, showing various pinned icons and the system tray.

The screenshot shows the Swagger UI interface for the "getBookingById" endpoint of the booking-controller. The URL in the address bar is "localhost:9090/swagger-ui/index.html#/booking-controller/getBookingById".

The "Responses" section contains a "Curl" block with the following command:

```
curl -X 'GET' \
'http://localhost:9090/api/bookings/1' \
-H 'accept: */*'
```

Below the curl block are sections for "Request URL" (set to "http://localhost:9090/api/bookings/1") and "Server response". Under "Server response", there's a "Code" table with one row for "200". The "Details" column for "200" shows the "Response body" as:

```
{  
    "id": 1,  
    "price": 110,  
    "status": "PENDING"  
}
```

At the bottom of the screen, the Windows taskbar is visible, showing various pinned icons and the system tray.

```
curl -X 'GET' \
  'http://localhost:9090/api/bookings' \
  -H 'accept: */*'
```

Request URL

```
http://localhost:9090/api/bookings
```

Server response

Code	Details
200	Response body

```
[  
  {  
    "id": 1,  
    "price": 1120,  
    "status": "PENDING"  
  },  
  {  
    "id": 2,  
    "price": 1500,  
    "status": "CONFIRMED"  
  },  
  {  
    "id": 3,  
    "price": 15000,  
    "status": "CANCELED"  
  },  
  {  
    "id": 4,  
    "price": 1234.87,  
    "status": "PENDING"  
  }]
```

Download

```
curl -X 'POST' \
  'http://localhost:9090/api/bookings' \
  -H 'accept: */*' \
  -H 'Content-Type: application/json' \
  -d '{
    "id": 98,
    "price": 8876,
    "status": "PENDING"
}'
```

Request URL

```
http://localhost:9090/api/bookings
```

Server response

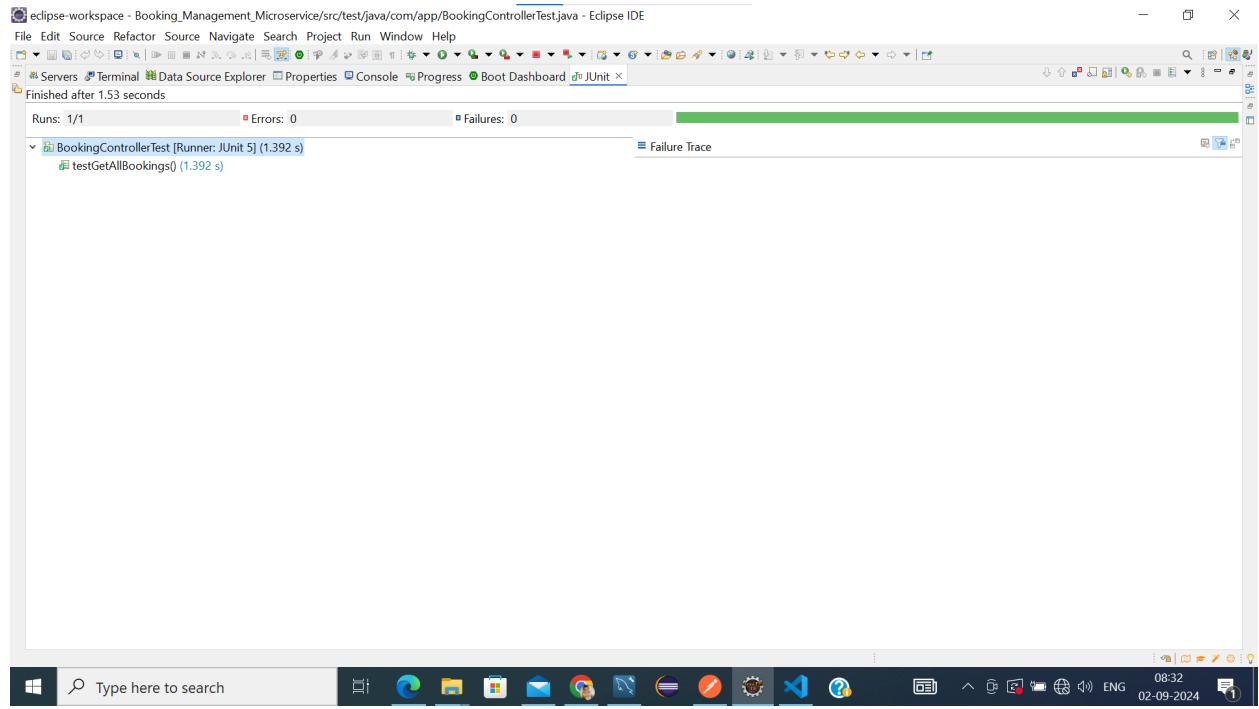
Code	Details
200	Response body

```
{  
  "id": 6,  
  "price": 8876,  
  "status": "PENDING"  
}
```

Response headers

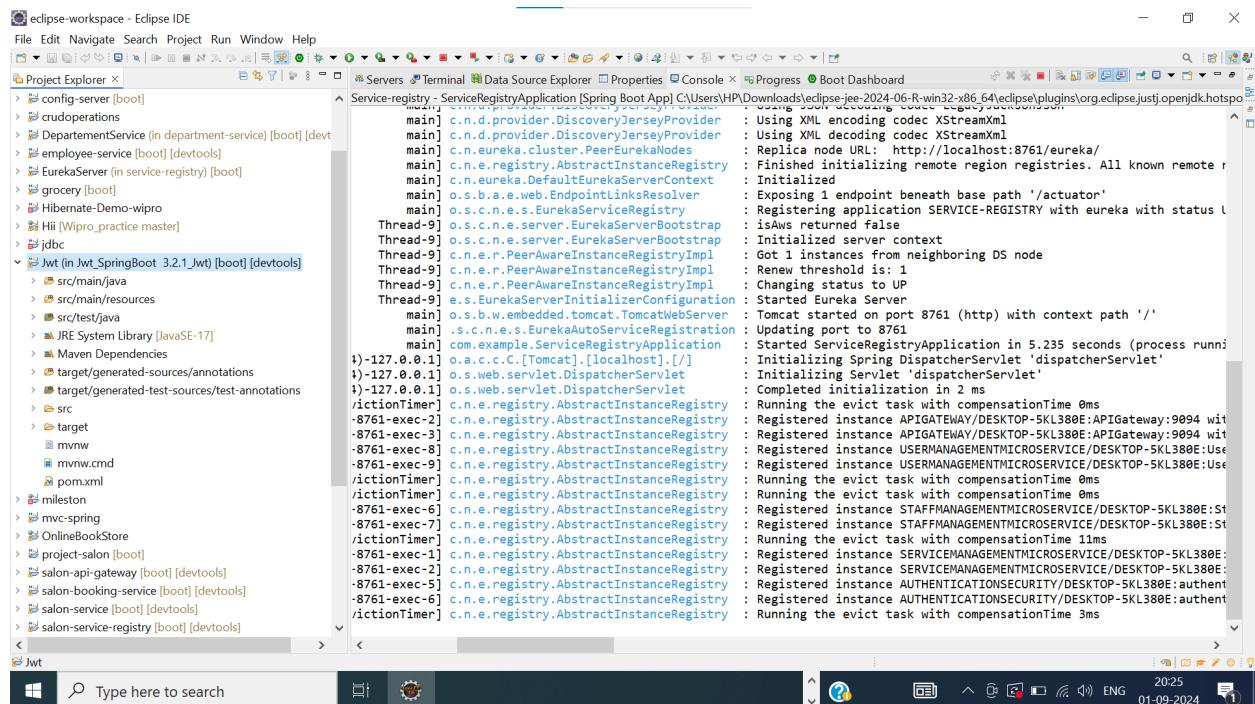
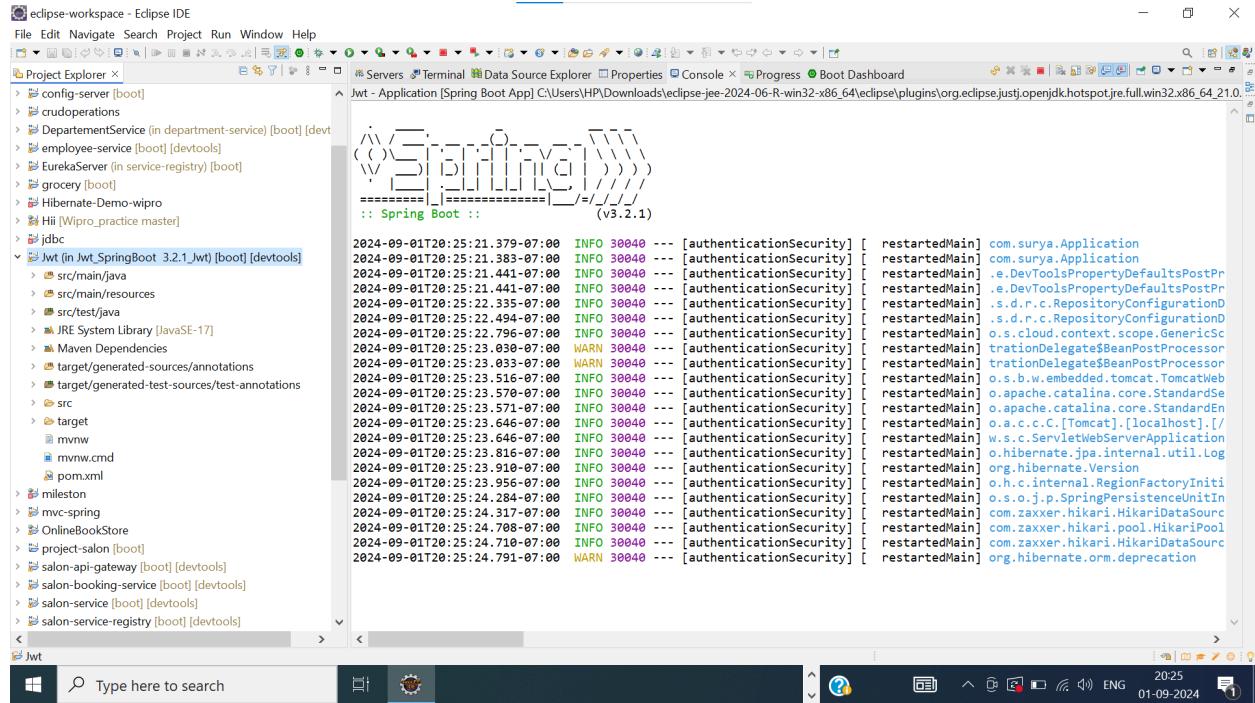
```
connection: keep-alive  
content-type: application/json  
date: Mon, 02 Sep 2024 04:41:52 GMT  
keep-alive: timeout=60  
transfer-encoding: chunked
```

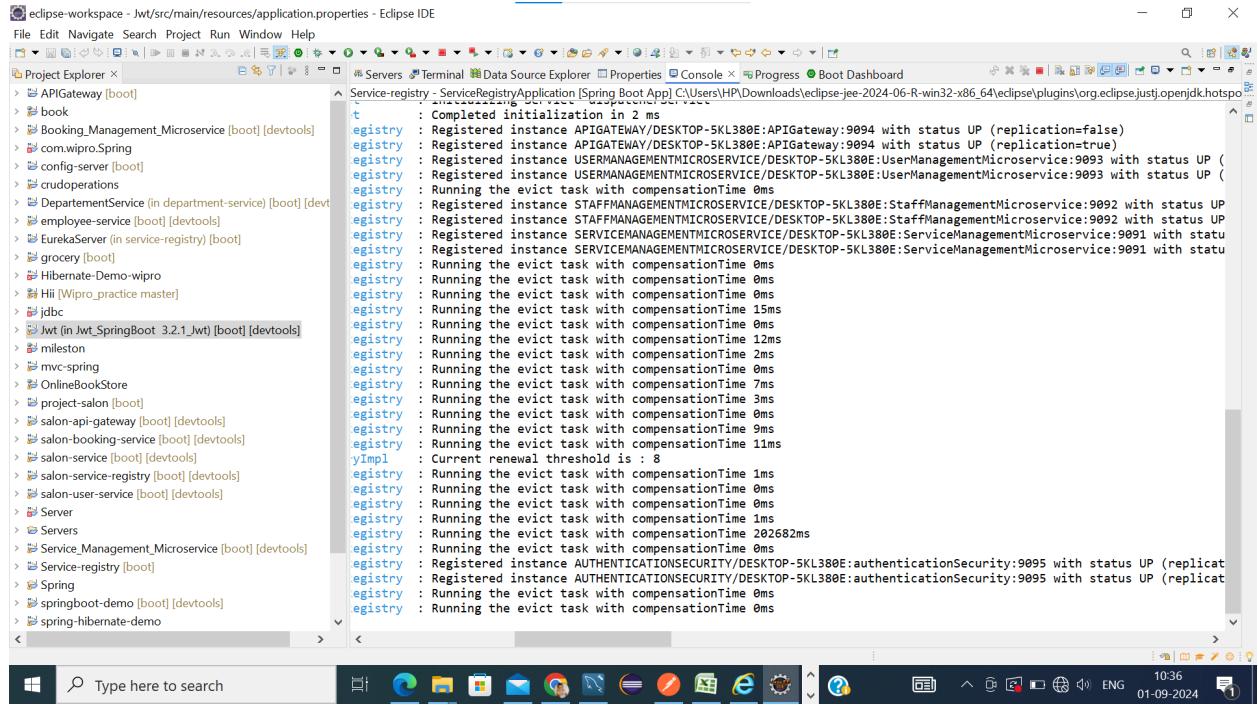
JUNIT Testing For Booking_Management_Microservice:



JWT AUTHENTICATION:

JWT Authentication Running as the SpringBootApplication in the Eclipse IDE:





Creating the JWT Database in SQL WorkBench:

#	Time	Action	Message	Duration / Fetch
23	21:05:11	SELECT * FROM staff_management.staff	4 row(s) returned	0.000 sec / 0.000 sec
24	21:17:21	SELECT * FROM service_management.services	3 row(s) returned	0.000 sec / 0.000 sec
25	21:36:24	SELECT * FROM booking_management.booking	4 row(s) returned	0.000 sec / 0.000 sec
26	21:52:08	SELECT * FROM salon_project.user_info	4 row(s) returned	0.000 sec / 0.000 sec
27	21:52:10	SELECT * FROM salon_project.user_info_seq	1 row(s) returned	0.000 sec / 0.000 sec
28	21:52:16	SELECT * FROM salon_project.user_info	4 row(s) returned	0.000 sec / 0.000 sec

The screenshot shows the MySQL Workbench interface. In the Navigator pane, under the 'Schemas' section, the 'salon_project' schema is selected, showing its tables: 'info', 'users', 'staff', 'services', 'booking', 'user_info', 'user_info_seq', and 'user_info_Table'. The 'user_info' table is currently selected. The SQL Editor pane contains the query: `SELECT * FROM salon_project.user_info;`. The Result Grid pane displays the following data:

	id	password	roles	username
1	\$2a\$10\$hxTmML.HcVjN7TAQNMF.Ib482V0Hg...	ADMIN	Rohith	
2	\$2a\$10\$HL05gOydbjZ1NNoknjo.97IRZV10n...	USER	Rohith	
3	\$2a\$10\$laMseEW2gpfy1mQuO9pPoQmU0nT...	USER	Rohith	
4	\$2a\$10\$6xdkcCQ13KRaNSwPRCG4nuQf57y.6r...	ADMIN	Ganesh	
52	\$2a\$10\$xvr3vXlUpJy5hnXna/486.YhJgwXy64...	ADMIN	Balu	
	NULL	NULL	NULL	

The Output pane shows the execution history:

#	Time	Action	Message	Duration / Fetch
24	21:17:21	SELECT * FROM service_management.services	3 row(s) returned	0.000 sec / 0.000 sec
25	21:36:24	SELECT * FROM booking_management.booking	4 row(s) returned	0.000 sec / 0.000 sec
26	21:52:08	SELECT * FROM salon_project.user_info	4 row(s) returned	0.000 sec / 0.000 sec
27	21:52:10	SELECT * FROM salon_project.user_info_seq	1 row(s) returned	0.000 sec / 0.000 sec
28	21:52:16	SELECT * FROM salon_project.user_info	4 row(s) returned	0.000 sec / 0.000 sec
29	21:57:27	SELECT * FROM salon_project.user_info	5 row(s) returned	0.000 sec / 0.000 sec

Generating the JWT Token in the POSTMAN:

The screenshot shows the Postman interface. The URL in the header is `POST http://localhost:9095/auth/authenticate`. The Body tab is selected, showing the following JSON payload:

```

1
2   "username": "Balu",
3   "password": "Rohi"
4
5
6
7
    
```

The Response tab shows the following details:

- Status: 200 OK
- Time: 151 ms
- Size: 553 B
- Save Response

The response body is a long JWT token: `eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJCYWxIiwiWF0IjoxNzI1MjUyOTQwLCJleHAiOjE3MjUyNTY1NDB9.igN3FmwzvaBxyZvwDa1Ve01nSZ30fFkEu8DEvI4m2wY`.

The screenshot shows the Postman interface. At the top, there are navigation links for Home, Workspaces, and Explore, along with a search bar and account options for Sign In and Create Account. A message at the top states: "You are using the Lightweight API Client, sign in or create an account to work with collections, environments and unlock all free features in Postman." Below this, a request card is displayed for a POST request to <http://localhost:9095/user/save>. The request method is set to POST, and the URL is <http://localhost:9095/user/save>. The "Body" tab is selected, showing a JSON payload:

```
1  {
2     "username": "Balu",
3     "password": "Rohi",
4     "roles": "ADMIN"
5
6
7 }
```

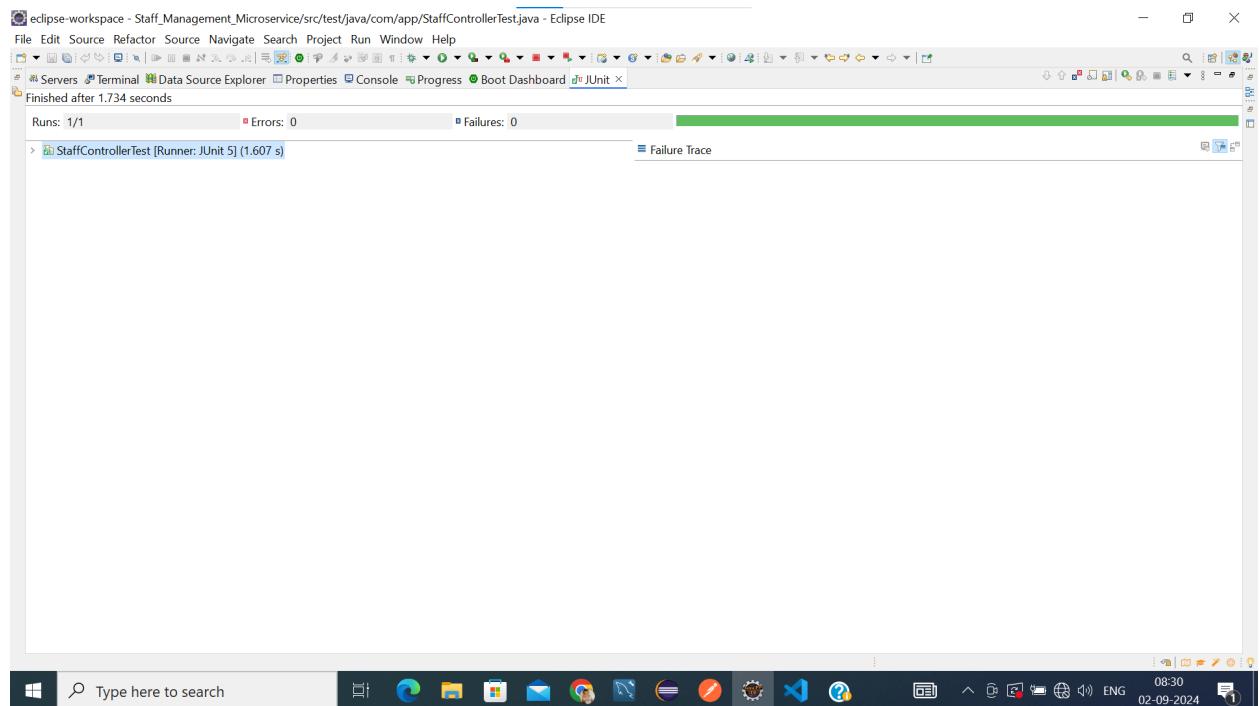
The "Params", "Authorization", "Headers", and "Tests" tabs are also visible. The "Body" dropdown shows options: none, form-data, x-www-form-urlencoded, raw, binary, and JSON (which is selected). Buttons for "Send", "Cookies", and "Beautify" are present. Below the request card, the response status is shown as 200 OK with a time of 223 ms and a size of 460 B. The response body contains the message: "User Created Successfully saved in db". The bottom of the window shows the Windows taskbar with various pinned icons.

JUNIT TEST CASES:

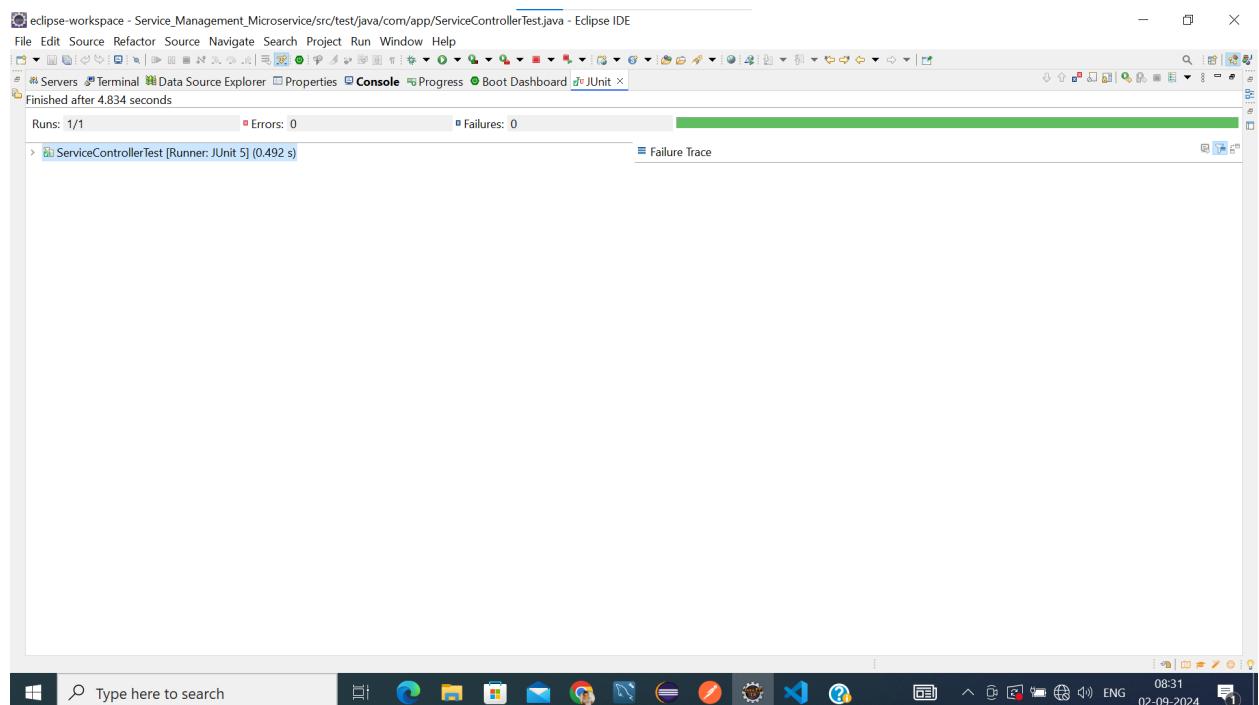
1. User_Management_Microservice:

The screenshot shows the Eclipse IDE interface. The title bar indicates the project is "eclipse-workspace - User_Management_Microservice/src/test/java/com/app/UserControllerTest.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Source, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations. The central workspace shows a JUnit test runner titled "UserControllerTest [Runner: JUnit 5] (1.635 s)". The status bar at the bottom shows "Runs: 1/1", "Errors: 0", and "Failures: 0". The bottom of the window shows the Windows taskbar with pinned icons.

2.Satff_Management_Microservice:



3.Service_Management_Microservice:



4.Booking_Management_Microservice:

