

BookMyConsultation

Upgrad Capstone Project

Initial Analysis:

Understood the overall design and architecture of book my consultation application and the services involved by going through the videos and documents provided.

I have made few changes in terms of response fields and values for few requests so that its more meaningful and also to handle the subsequent tasks easily.

The BookMyConsultation folder will contain the following services and files apart from this documentation file:

- doctor-service
- appointment-service
- user-service
- rating-service
- notification-service
- payment-service
- bmc-gateway
- docker-compose.yml

Note: I have decided to use externalized Kafka and MongoDB by setting it up on ec2 instance and created an RDS instance for MySQL DB in this capstone project.

I have also not used eureka-server for this project as the reason being that when we deploy these services on the docker and then eventually on k8s, they have their own DNS service discovery available as service component which will render eureka useless and additional overhead.

Hence, I have only used gateway service here to route requests to different services.

The BookMyConsultation application has many services and the logic and configuration changes for each service will be explained individually below and the Kafka, MongoDB, MySQL and ubuntu instance setup's will be available at the end of this documentation.

General Components (All Services):

I have implemented two components which are same across all services important for monitoring and securing all of them. They are **Aspect Oriented Logging** and **Security**.

Aspect Oriented Logging: I have added an spring aop starter and implemented Aspect Logging in all services to reduce the manual effort of logging in every class unless required, which combined with spring cloud sleuth helps in tracing.

General Log Format used for every service method:

While Entering: *log.info("Entering CLASS_NAME:{} METHOD_NAME:{} with input parameters {}", className, methodName, inputArgs);*

While Exiting: *log.info("Exiting CLASS_NAME:{} METHOD_NAME:{} with return value [{}]) and execution time of {}", className, methodName,*

result,getTimeInMinutesAndSeconds(stopWatch.getTotalTimeMillis()));

This can be seen in logs of all services for any request made.

Security: I have used spring security to secure all the endpoints for all services and hence before accessing any service endpoint you need to get an auth jwt token from any service, and then this same token can be used for all requests to all services for a day as I have set the expiration date for the tokens post 24hours from creation for any user or admin.

For the purpose of this project, I have created two admin user's and 4 normal users (patient/doctor) as below:

```
private List<ApplicationUser> loadAllUsers(){
    return Arrays.asList(
        ApplicationUser
            .builder()
            .username("Elon Musk")
            .password(encoder.encode("bmc-password-1"))
            .authorities(ApplicationRole.USER.getAuthorities())
            .build(),
        ApplicationUser
            .builder()
            .username("Mukesh Ambani")
            .password(encoder.encode("bmc-password-2"))
            .authorities(ApplicationRole.USER.getAuthorities())
            .build(),
        ApplicationUser
            .builder()
```

```

        .authorities(ApplicationRole.USER.getAuthorities())
        .build(),
    ApplicationUser
        .builder()
        .username("Sundar Pichai")
        .password(encoder.encode("bmc-password-6"))
        .authorities(ApplicationRole.ADMIN.getAuthorities())
        .build(),
    ApplicationUser
        .builder()
        .username("Syed Ruhan")
        .password(encoder.encode("bmc-password-7"))
        .authorities(ApplicationRole.ADMIN.getAuthorities())
        .build()
);

```

You can get the user or admin token for hitting the individual endpoints based on roles allowed on the endpoint from below uri:

`{environment}{doctor-service-context-path}/login`

`localhost:9191/doctor-service/login`

Here environment refers to the host where your running, if its local you hit the gateway on localhost:9191 or if its on ec2 then ec2-url:9191, and for doctor-service-context-path, this needs to be added to every doctor-service uri so that bmc-gateway can route the requests to services correctly.

Note: I have added a `/verify-email` under doctor-service so that an admin can have his email verified and use that same email in notification-service for default `from-email` for all types of events.

BMC Gateway Logic:

Eg: In case of doctor-service endpoint you need to hit the endpoint on gateway with context path doctor-service before the actual doctor-service uri:

`GET: localhost:9191/doctor-service/doctors/{doctorId}`

Note: bmc-gateway has a filter in application.yml to strip the context path before routing the requests, that way I didn't have to add a context path to the individual services for the sole purpose of routing.

Context-paths for all services:

doctor-service: /doctor-service

appointment-service: /appointment-service

user-service: /user-service

rating-service: /rating-service

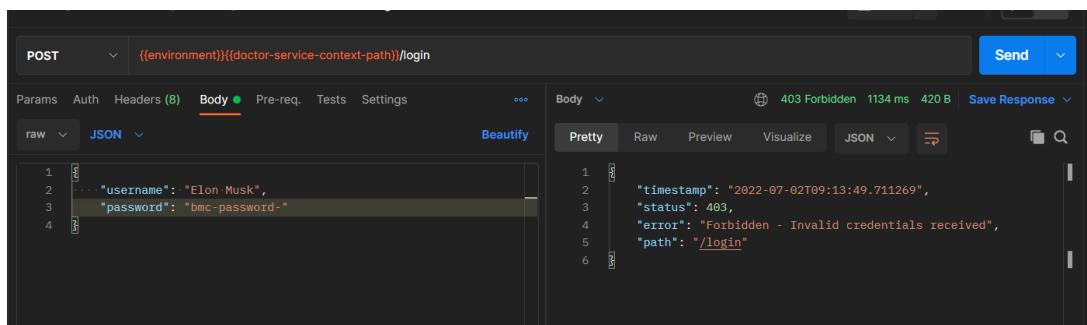
payment-service: /payment-service

Doctor Service (doctor-service):

Logic and Design: In doctor-service we are storing the details of doctor in MongoDB and sending an email verification mail to doctor email Id provided using Amazon SES. We are also sending a message to notification-service for all doctor related events.

For doctor sorted list logic I have used required tag as false for speciality param so that we can either pass status, or both status and speciality and internally used a dynamic query with mongo template to get the result.

Changes on Responses: For the error responses I have attached the default messages along with the error fields so that user can understand the right format to send details. In case of forbidden errors, I have modified the response message to alert if it's an authorization issue or invalid credentials issue so that we can understand the issue. Please find the responses below:



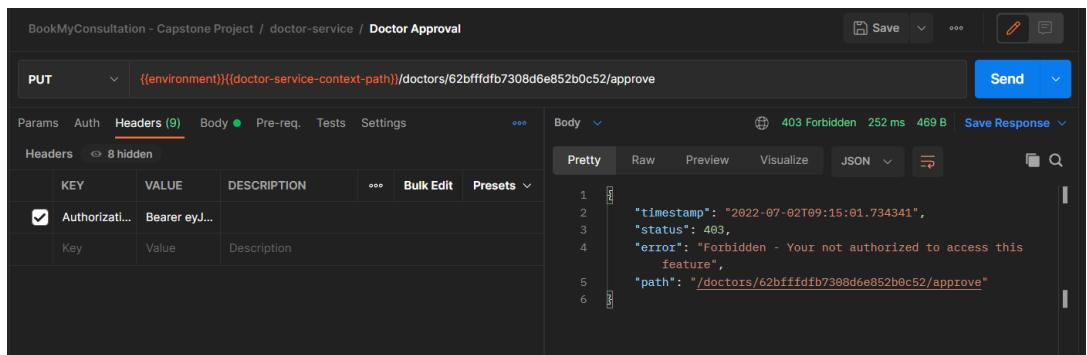
A screenshot of the Postman application interface. The request method is POST, the URL is `POST {{environment}}{{(doctor-service-context-path)}/login`, and the status code is 403 Forbidden. The request body is JSON with the following content:

```
1 {  
2   "username": "Elon Musk",  
3   "password": "bmc-password"  
4 }
```

The response body is JSON with the following content:

```
1 {  
2   "timestamp": "2022-07-02T09:13:49.711269",  
3   "status": 403,  
4   "error": "Forbidden - Invalid credentials received",  
5   "path": "/login"  
6 }
```

Invalid password response



A screenshot of the Postman application interface. The request method is PUT, the URL is `PUT {{environment}}{{(doctor-service-context-path)}/doctors/62bffffdb7308d6e852b0c52/approve`, and the status code is 403 Forbidden. The request headers include an Authorization header with the value "Bearer eyJ...". The response body is JSON with the following content:

```
1 {  
2   "timestamp": "2022-07-02T09:15:01.734341",  
3   "status": 403,  
4   "error": "Forbidden - You not authorized to access this feature",  
5   "path": "/doctors/62bffffdb7308d6e852b0c52/approve"  
6 }
```

Using USER token for approve request

Once the tokens are received, they can be used for a day as its expiration is for current day + 1. The same token can be used for all service requests all of the logic and key to decode the token and allow access.

One important logic for rating I have implemented is that the rating service will calculate the average doctor rating and doctor-service will only receive the average rating from rating-service and update in its table, this way once its updated users can

see the doctors rating and also the sorting becomes easier at doctor side because of which I have added average rating field in doctor schema.

I used Json non-null property to not send any field which are null which handles the null being shown until the rating is received.

All other request responses are same as mentioned in the postman collection and service designs.

Configuration Change:

For the purpose of simplicity, I have added a comment **Configure before run** comment on every line where changes are to be made before run based on your setup.

You can find the screenshots for doctor-service config changes below:

```
data:  
  mongodb:  
    host: ec2-3-232-57-4.compute-1.amazonaws.com # Configure before run  
    port: 27017  
    database: doctor-service|  
  
server:  
  port: 8081  
  
kafka:  
  producer:  
    bootstrap-servers: ec2-44-207-79-237.compute-1.amazonaws.com:9092 # Configure before run  
    topic: bmc-notification  
  consumer:  
    bootstrap-servers: ec2-44-207-79-237.compute-1.amazonaws.com:9092 # Configure before run  
    topic: bmc-doctor  
    group-id: doctor-service
```

application.yml

```

public class S3Repository {
    private AmazonS3 s3Client;
    private String BUCKET_NAME = "ruhan.bmc.doctor.documents"; // Configure before run
    // This needs to be a unique bucket name across all the regions.

    ObjectMetadata metadata;

    @PostConstruct
    public void init(){
        // Update the amazon s3 admin access key and secret key here once generated for your account
        String accessKey = ""; // Configure before run
        String secretKey = ""; // Configure before run
        AWSCredentials credentials = new BasicAWSCredentials(accessKey,secretKey);
        s3Client = AmazonS3ClientBuilder
            .standard()
            .withCredentials(new AWSStaticCredentialsProvider(credentials))
            .withRegion(Regions.US_EAST_1)
            .build();
    }
}

```

S3Repository

```

@PostConstruct
public void init(){
    // When you hit the endpoint to verify the email this needs to be the ses access key for your AWS account
    String accessKey = ""; // Configure before run
    // When you hit the endpoint to verify the email this needs to be the ses secret key for your AWS account
    String secretKey = ""; // Configure before run

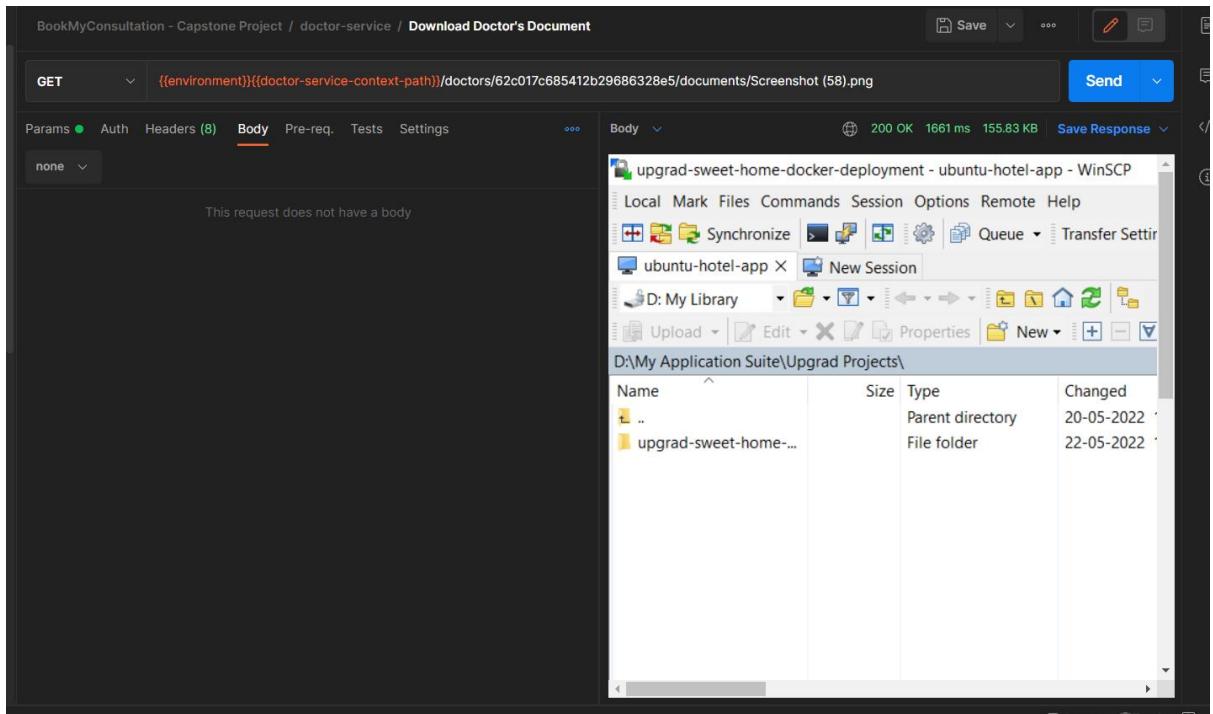
    //This is required for email verification
    StaticCredentialsProvider staticCredentialsProvider = StaticCredentialsProvider
        .create(AwsBasicCredentials.create(accessKey,secretKey));
    sesClient = SesClient.builder()
        .credentialsProvider(staticCredentialsProvider)
        .region(Region.US_EAST_1)
        .build();
}

```

EmailService

I have also implemented the optional endpoints for getting the list of documents uploaded and also downloading a document based on its full name including the extension. I referred the online docs from amazon to understand how to handle such requests. Please enter the full name with extension (.png/.pdf/.txt) while retrieving the document.

Doctor service listens to bmc-doctor topic to update the rating of doctor received from rating-service.



For example: Here I have uploaded a screenshot of my previous assignment for demo.

User Service (user-service):

- Logic and Design:** In case of user-service most of the logic and response are same as in doctor-service. It has all the operations such as register, upload, download and get individual user details.

I have used comments wherever there is complex logic but usage of understandable variable names will make the steps easier to understand. A kafka producer also is used to send a user registration event to notification-service.

- Configuration Changes:**

For the purpose of simplicity, I have added a comment **Configure before run** on every line where changes are to be made before run based on your setup. You can find the screenshots for user-service config changes below:

```
service
  data:
    mongodb:
      host: ec2-3-232-57-4.compute-1.amazonaws.com # Configure before run
      port: 27017
      database: user-service

  server:
    port: 8083

  kafka:
    producer:
      bootstrap-servers: ec2-44-207-79-237.compute-1.amazonaws.com:9092 # Configure before run
      topic: bmc-notification
```

application.yml

```
@PostConstruct
public void init(){
    // When you hit the endpoint to verify the email this needs to be the ses access key for your AWS account
    String accessKey = ""; // Configure before run
    // When you hit the endpoint to verify the email this needs to be the ses secret key for your AWS account
    String secretKey = ""; // Configure before run

    //This is required for email verification
    StaticCredentialsProvider staticCredentialsProvider = StaticCredentialsProvider
        .create(AwsBasicCredentials.create(accessKey,secretKey));
    sesClient = SesClient.builder()
        .credentialsProvider(staticCredentialsProvider)
        .region(Region.US_EAST_1)
        .build();
}
```

EmailService

```

public class S3Repository {
    private AmazonS3 s3Client;
    private String BUCKET_NAME = "ruhan.bmc.user.documents"; // Configure before run
    // This needs to be a unique bucket name across all the regions.

    ObjectMetadata metadata;

    @PostConstruct
    public void init(){
        // Update the s3 admin access credentials here after creating it in the AWS console
        String accessKey = ""; // Configure before run
        String secretKey = ""; // Configure before run
        AWSCredentials credentials = new BasicAWSCredentials(accessKey,secretKey);
        s3Client = AmazonS3ClientBuilder
            .standard()
            .withCredentials(new AWSStaticCredentialsProvider(credentials))
            .withRegion(Regions.US_EAST_1)
            .build();
    }
}

```

S3Repository

Appointment Service (appointment-service):

- **Logic and Design:** In appointment service I have used mapping method to take the input for doctor availability as map and then mapped them to Availability Entities as defined in schema.

Appointment service has synchronous calls being made to doctor-service and user-service using open-feign during appointment booking call. I have added comments for the same in code. Considering we only receive the id's I make calls to these two services in order to fetch user and doctor details for notification-service and DB update.

Request response for most of the endpoints are as expected apart from the few exception response changes mentioned in the doctor-service.

- **Configuration Change:**

For the purpose of simplicity, I have added a comment **Configure before run** on every line where changes are to be made before run based on your setup. You can find the screenshots for appointment-service config changes below:

```

data:
  mongodb:
    host: ec2-3-232-57-4.compute-1.amazonaws.com # Configure before run
    port: 27017
    database: appointment-service
  datasource:
    url: jdbc:mysql://bookmyconsultation.cqelf5ahntp.us-east-1.rds.amazonaws.com/appointment_service # Configure before run
    username: admin # Configure before run
    password: bmc-1234 # Configure before run
  jpa:
    hibernate:
      ddl-auto: none
      naming:
        physical-strategy: org.hibernate.boot.model.naming.PhysicalNamingStrategyStandardImpl
      show-sql: true
  server:
    port: 8082
  kafka:
    producer:
      bootstrap-servers: ec2-44-207-79-237.compute-1.amazonaws.com:9092 # Configure before run
      topic: bmc-notification

```

application.yml

Payment Service (payment-service):

- **Logic and Design:** In payment service we have only one endpoint for making a payment for appointment. Here I am doing a synchronous call to first update the payment status in appointment-service and then getting that same appointment object to send payment update notification to user by sending it to notification-service via kafka topic (Payment Event).

Here I am also storing the transaction details at payment repo first in order to have a backup of transaction and also if there occurs an error, I have used transaction roll back to reverse any DB updates.

- **Configuration Changes:**

For the purpose of simplicity, I have added a comment **Configure before run** on every line where changes are to be made before run based on your setup.
You can find the screenshots for payment-service config changes below:

```

3   name: payment-service
4   data:
5     mongodb:
6       host: ec2-3-232-57-4.compute-1.amazonaws.com # Configure before run
7       port: 27017
8       database: payment-service
9
10  server:
11    port: 8086
12
13  kafka:
14    producer:
15      topic: bmc-notification
16      bootstrap-servers: ec2-44-207-79-237.compute-1.amazonaws.com:9092 # Configure before run
17

```

application.yml

Notification Service (notification-service):

- Logic and Design:** In notification-service I have used a single topic bmc-notification to receive notifications from all services for all event types, but I have used the key of a record to identify the event type and process them accordingly.

The kafka consumer only consumes and forwards the record on to the event processor service where the actual logic for handling different events resides.

Used user, doctor, appointment event types to process them separately before handing it over to email service for notification.

In notification service it's important that you set a from-email in application.yml which is verified in ses to send emails to other verified user's else mail will not be sent.

There are 5 email templates present under templates folder of src/main/resources. Each template is used for notification of user-appointment, doctor-approval, doctor-rejection, payment-confirmation and prescription-upload.

- Configuration Change:**

For the purpose of simplicity, I have added a comment **Configure before run** on every line where changes are to be made before run based on your setup.

You can find the screenshots for notification-service config changes below:

```

@PostConstruct
public void init(){
    // When you hit the endpoint to send an email this value needs to be updated to the Smtp username that you generated
    smtpUserName(""); // Configure before run
    // When you hit the endpoint to send an email this value needs to be updated to the Smtp password that you generated
    smtpPassword(""); // Configure before run
}

```

EmailService

```

ts\Boot 2   application:
ts\Boo 3     name: notification-service
Project:
4
5       server:
6         port: 8085
7
8       kafka:
9         consumer:
10           topic: bmc-notification
11           bootstrap-servers: ec2-44-207-79-237.compute-1.amazonaws.com:9092 # Configure before run
12           group-id: notification-service
13
14   from-email: ruhan008@gmail.com # Configure before run

```

application.yml

Rating Service (rating-service):

- Logic and Design:** In rating service we have a single endpoint to accept the rating for any doctor and I have modified the response to send a string confirming the acceptance of rating.

I have added comments in the code for logic. What I am doing is that I have two separate tables, one for just storing the rating received (Rating) and other for storing the count with average doctor rating (DoctorRating). The logic adds the rating to individual ratings (1-5) counts and the calculates the average doctor rating for the doctor and updates the DoctorRating repository, which is then sent to kafka topic bmc-doctor which is being listened by doctor-service so that it can update the doctors average rating in its repository whenever a rating event happens.

I have also added validation on the rating received in case the user inputs a value for rating outside the range of 1-5 or if doctorId provided is null.

The screenshot shows a Postman interface with the following details:

- Request Method: POST
- URL: {{environment}}{{(rating-service-context-path)}/ratings}
- Body Type: JSON
- Body Content:

```

1 {
2   ...
3   "rating": 10
4 }

```
- Response Status: 400 Bad Request
- Response Body (Pretty):

```

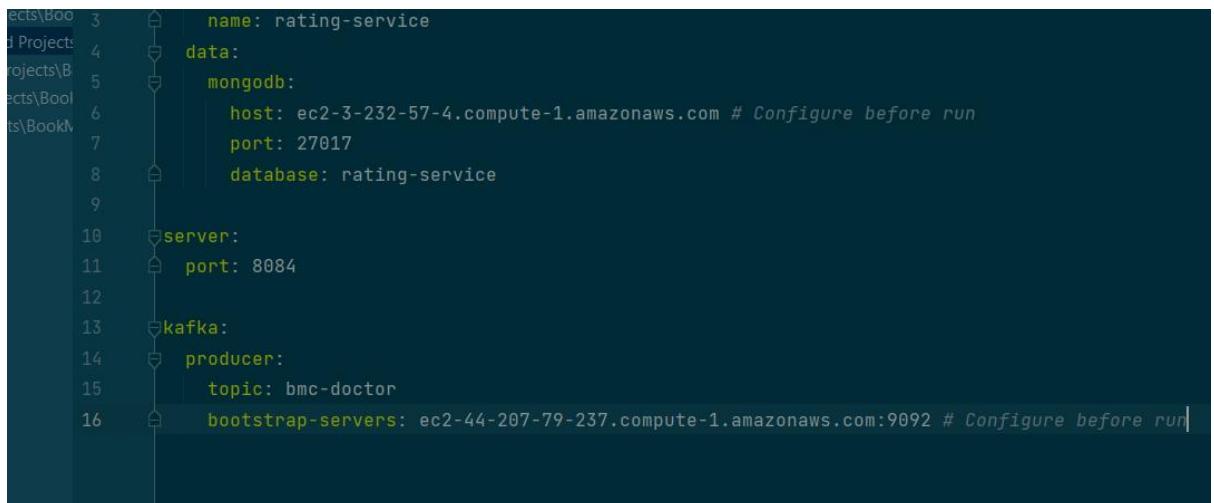
1 {
2   "errorCode": "ERR_INVALID_INPUT",
3   "errorMessage": "Invalid input. Parameter name: ",
4   "errorFields": [
5     "doctorId - Doctor Id field cannot be null",
6     "rating - Please enter a value between 1-5 (inclusive
       of 1 and 5"
7   ]
8 }

```

Error response for doctorId - null and rating - 10

- **Configuration Change:**

For the purpose of simplicity, I have added a comment **Configure before run** on every line where changes are to be made before run based on your setup. You can find the screenshots for rating-service config changes below:



```
ects\Boo 3     name: rating-service
d Projects\ 4     data:
objects\Boo 5       mongodb:
ects\Boo 6         host: ec2-3-232-57-4.compute-1.amazonaws.com # Configure before run
ts\BookN 7         port: 27017
ts\BookN 8         database: rating-service
ts\BookN 9
ts\BookN 10      server:
ts\BookN 11        port: 8084
ts\BookN 12
ts\BookN 13      kafka:
ts\BookN 14        producer:
ts\BookN 15          topic: bmc-doctor
ts\BookN 16        bootstrap-servers: ec2-44-207-79-237.compute-1.amazonaws.com:9092 # Configure before run
```

application.yml

BMC Gateway (bmc-gateway):

- **Logic and Design:** For bmc-gateway the routes are already configured to route the incoming requests to different services based on a predicate (context-path).
- **Configuration Change:**
There is no configuration change required for bmc-gateway.

Docker Compose (docker-compose):

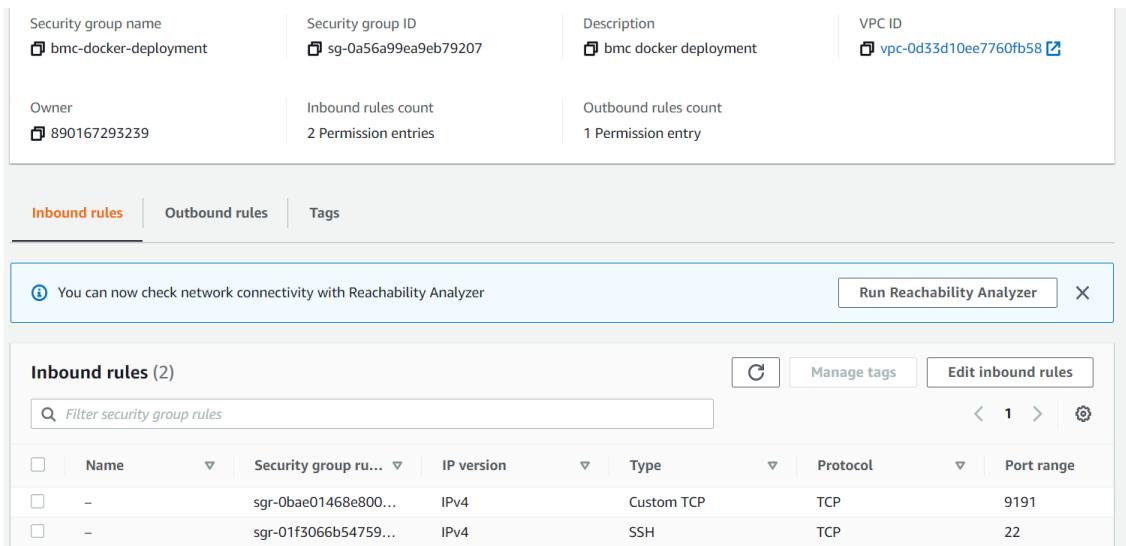
- The docker-compose.yml file is configured to build and run all the services once all the config changes are done as explained above and other DB and Kafka setup is also done.

Command: **docker-compose up or sudo docker-compose up** can be used depending on whether you're running locally or on an ubuntu instance.

Note: All the service folders have Dockerfile's written for them.
 Also, in case you're running it on aws ubuntu or ec2 instance and the build isn't able to find the folder then please replace build tag value under services with eg. `./notification-service` with `notification-service` and try (remove `./` and try).

Screenshots for request and response on all services

I have deployed all services on an ubuntu instance using docker-compose and accessing all of them from local by only exposing bmc-gateway port 9191.



The screenshot shows the AWS Security Groups console. The main table displays the following details for the security group 'bmc-docker-deployment':

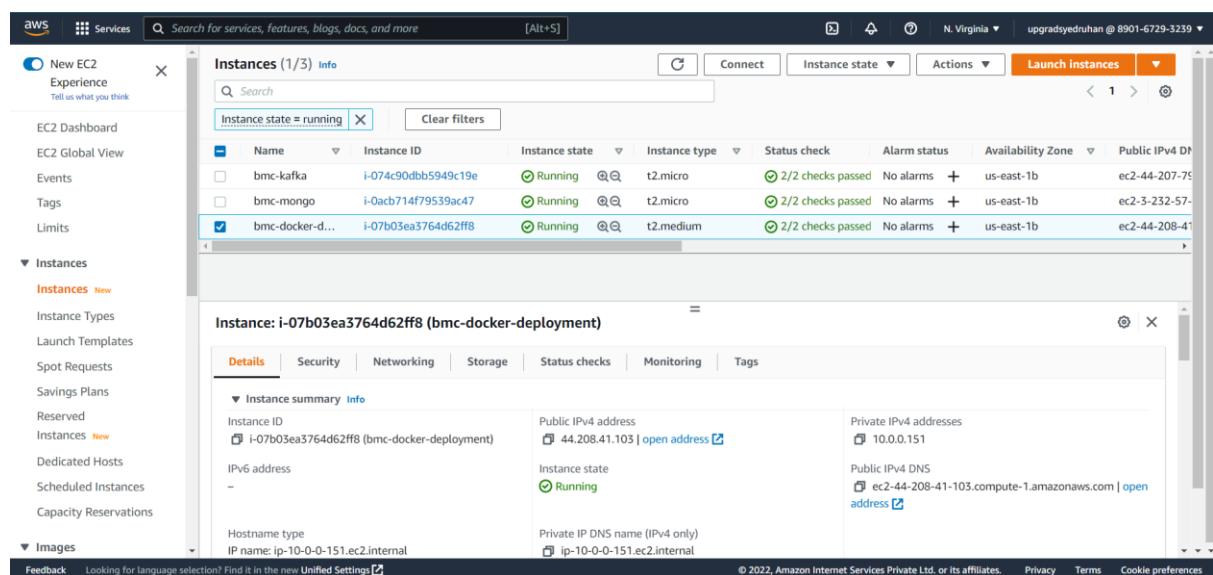
Security group name	Security group ID	Description	VPC ID
bmc-docker-deployment	sg-0a56a99ea9eb79207	bmc docker deployment	vpc-0d33d10ee7760fb58

Below the table, there are tabs for 'Inbound rules', 'Outbound rules', and 'Tags'. The 'Inbound rules' tab is selected, showing two entries:

Name	Security group rule ID	IP version	Type	Protocol	Port range
-	sgr-0bae01468e800...	IPv4	Custom TCP	TCP	9191
-	sgr-01f3066b54759...	IPv4	SSH	TCP	22

A message at the top indicates: "You can now check network connectivity with Reachability Analyzer" with buttons for "Run Reachability Analyzer" and "X".

ubuntu instance security group



The screenshot shows the AWS EC2 Instances console. The main table lists three instances:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
bmc-kafka	i-074c90dbb5949c19e	Running	t2.micro	2/2 checks passed	No alarms	us-east-1b	ec2-44-207-75-
bmc-mongo	i-0acb714f79539ac47	Running	t2.micro	2/2 checks passed	No alarms	us-east-1b	ec2-3-232-57-
bmc-docker-deployment	i-07b03ea3764d62ff8	Running	t2.medium	2/2 checks passed	No alarms	us-east-1b	ec2-44-208-41-

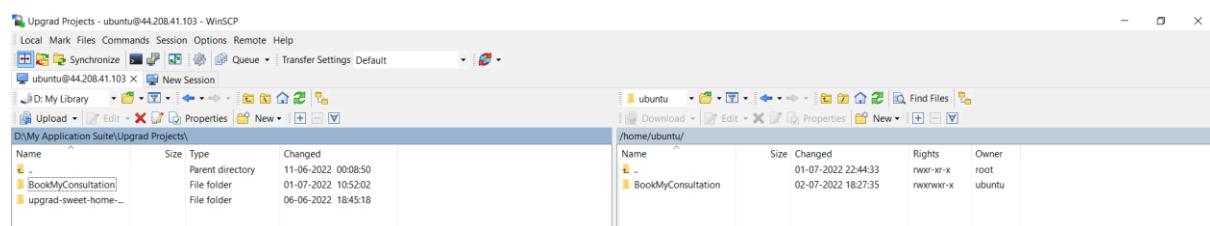
The detailed view for the 'bmc-docker-deployment' instance shows the following information:

Details	Security	Networking	Storage	Status checks	Monitoring	Tags
Instance summary	Info					
Instance ID: i-07b03ea3764d62ff8 (bmc-docker-deployment)	Public IPv4 address: 44.208.41.103 open address	Private IPv4 addresses: 10.0.0.151				
IPv6 address: -	Instance state: Running	Public IPv4 DNS: ec2-44-208-41-103.compute-1.amazonaws.com open address				
Hostname type: IP name: ip-10-0-0-151.ec2.internal	Private IP DNS name (IPv4 only): ip-10-0-0-151.ec2.internal					

ubuntu ec2-instance

bmc-gateway				
	VARIABLE	TYPE	INITIAL VALUE	CURRENT VALUE
appointment-service	environment	default	44.208.41.103:9191	44.208.41.103:9191
bmc-gateway	doctor-service-context-path	default	/doctor-service	/doctor-service
doctor-service	appointment-service-context-path	default	/appointment-service	/appointment-service
EC2-54.145.89.93	user-service-context-path	default	/user-service	/user-service
local	payment-service-context-path	default	/payment-service	/payment-service
payment-service	rating-service-context-path	default	/rating-service	/rating-service
rating-service				
user-service				

Postman environment setup: The environment variable has the ubuntu ec2 public ip with bmc-gateway port.



Uploaded via WinScp to ubuntu

```
ubuntu@ip-10-0-0-151:~/BookMyConsultation
[doctor-service] | 2022-07-02 13:11:35.986 INFO [doctor-service,,] 1 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data MongoDB repositories in DEFAULT mode.
[doctor-service] | 2022-07-02 13:11:36.159 INFO [rating-service,,] 1 --- [main] o.s.cloud.context.scope.GenericScope : BeanFactory id=0ae39acc-6622-3326-90a3-c6463ecc
[doctor-service] | 2022-07-02 13:11:36.712 INFO [doctor-service,,] 1 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 680 ms. Found 2 MongoDB repository interfaces.
[payment-service] | 2022-07-02 13:11:37.112 INFO [payment-service,,] 1 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data MongoDB repositories in DEFAULT mode.
[user-service] | 2022-07-02 13:11:37.312 INFO [user-service,,] 1 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data MongoDB repositories in DEFAULT mode.
[payment-service] | 2022-07-02 13:11:37.673 INFO [payment-service,,] 1 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 50 ms. Found 1 MongoDB repository interface.
[user-service] | 2022-07-02 13:11:38.006 INFO [user-service,,] 1 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 630 ms. Found 1 MongoDB repository interface.
[appointment-service] | 2022-07-02 13:11:38.984 INFO [appointment-service,,] 1 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Multiple Spring Data modules found, entering strict repository configuration mode!
[appointment-service] | 2022-07-02 13:11:39.015 INFO [appointment-service,,] 1 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFAULT mode.
[appointment-service] | 2022-07-02 13:11:39.595 INFO [appointment-service,,] 1 --- [main] .RepositoryConfigurationExtensionsSupport : Spring Data JPA - Could not safely identify store assignment for repository candidate interface com.bookmyconsultation.appointmentservice.repository.PrescriptionRepository. If you want this repository to be a JPA repository, consider annotating your entities with one of these annotations: javax.persistence.Entity, javax.persistence.MappedSuperclass (preferred), or consider extending one of the following types with your repository: org.springframework.data.jpa.repository.JpaRepository.
[notification-service] | 2022-07-02 13:11:39.652 INFO [notification-service,,] 1 --- [main] .s.o.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port(s): 8085 (http)
[modification-service] | 2022-07-02 13:11:39.722 INFO [modification-service,,] 1 --- [main] o.apache.catalina.core.StandardService : Starting service [Tomcat]
[modification-service] | 2022-07-02 13:11:39.722 INFO [modification-service,,] 1 --- [main] org.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/9.0.63]
[appointment-service] | 2022-07-02 13:11:39.847 INFO [appointment-service,,] 1 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 763 ms. Found 2 JPA repository interfaces.
[appointment-service] | 2022-07-02 13:11:39.969 INFO [appointment-service,,] 1 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Multiple Spring Data modules found, entering strict repository configuration mode!
[appointment-service] | 2022-07-02 13:11:39.989 INFO [appointment-service,,] 1 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data MongoDB repositories in DEFAULT mode.
[appointment-service] | 2022-07-02 13:11:40.077 INFO [appointment-service,,] 1 --- [main] .RepositoryConfigurationExtensionsSupport : Spring Data MongoDB - Could not safely identify store assignment for repository candidate interface com.bookmyconsultation.appointmentservice.repository.NotificationRepository. If you want this repository to be a MongoDB repository, consider annotating your entities with one of these annotations: org.springframework.data.mongodb.core.mapping.Document (preferred), or consider extending one of the following types with your repository: org.springframework.data.mongodb.repository.MongoRepository.
[appointment-service] | 2022-07-02 13:11:40.087 INFO [appointment-service,,] 1 --- [main] .RepositoryConfigurationExtensionsSupport : Spring Data MongoDB - Could not safely identify store assignment for repository candidate interface com.bookmyconsultation.appointmentservice.repository.AppointmentRepository. If you want this repository to be a MongoDB repository, consider annotating your entities with one of these annotations: org.springframework.data.mongodb.core.mapping.Document (preferred), or consider extending one of the following types with your repository: org.springframework.data.mongodb.repository.MongoRepository.
[appointment-service] | 2022-07-02 13:11:40.137 INFO [appointment-service,,] 1 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 147 ms. Found 1 MongoDB repository interface.
[modification-service] | 2022-07-02 13:11:40.707 INFO [modification-service,,] 1 --- [main] o.a.c.c.C.[localhost].[/] : Initializing Spring embedded WebApplicationContext
[modification-service] | 2022-07-02 13:11:40.733 INFO [modification-service,,] 1 --- [main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialized in 23321 ms
[doctor-service] | 2022-07-02 13:11:40.925 INFO [doctor-service,,] 1 --- [main] o.s.cloud.context.scope.GenericScope : BeanFactory id=413b7f8d-7b4c-36f5-9076-543238ba
[dd06]
```

Services Starting up

The screenshot shows the Postman application interface. A POST request is being made to the URL `POST {{environment}}{{doctor-service-context-path}}/login`. The request body is JSON, containing the following data:

```
1 | { "username": "Elon Musk", "password": "bmc-password-1"}
```

The response status is 200 OK, with a response time of 2.13 s and a size of 1.41 KB. The response headers are:

KEY	VALUE
authorization	Bearer eyJhbGciOiJIUzI1NiJ9eyJzdWIiOiJ... ... (redacted)
X-Content-Type-Options	nosniff
X-XSS-Protection	1; mode=block
Cache-Control	no-cache, no-store, max-age=0, must-re...
Pragma	no-cache
Expires	0
X-Frame-Options	DENY
Date	Sat, 02 Jul 2022 13:14:39 GMT
content-length	0

Getting user role token (Request & Response)

The screenshot shows the Postman application interface. A POST request is being made to the URL `POST {{environment}}{{doctor-service-context-path}}/login`. The request body is JSON, containing the following data:

```
1 | { "username": "Sundar Pichai", "password": "bmc-password-6"}
```

The response status is 200 OK, with a response time of 995 ms and a size of 1.32 KB. The response headers are:

KEY	VALUE
authorization	Bearer eyJhbGciOiJIUzI1NiJ9eyJzdWIiOiJ... ... (redacted)
X-Content-Type-Options	nosniff
X-XSS-Protection	1; mode=block
Cache-Control	no-cache, no-store, max-age=0, must-re...
Pragma	no-cache
Expires	0
X-Frame-Options	DENY
Date	Sat, 02 Jul 2022 13:16:04 GMT
content-length	0

Getting admin role token (Request & Response)

Doctor Service Request-Response:

The screenshot shows the Postman interface with the following details:

- Request Method:** POST
- URL:** {{environment}}{{doctor-service-context-path}}/doctors
- Body (JSON):**

```
1
2   "firstName": "Sadiq",
3   "lastName": "Umar",
4   "dob": "1965-10-25",
5   "mobile": "9900352601",
6   "emailId": "ruhan008@gmail.com",
7   "pan": "KKAPS8132M"
```
- Response Status:** 200 OK
- Response Time:** 2.52 s
- Response Size:** 544 B
- Response Body (Pretty JSON):**

```
1
2   "id": "62c0476d1948a6022c16be3b",
3   "firstName": "Sadiq",
4   "lastName": "Umar",
5   "speciality": "GENERAL_PHYSICIAN",
6   "dob": "1965-10-25",
7   "mobile": "9900352601",
8   "emailId": "ruhan008@gmail.com",
9   "pan": "KKAPS8132M",
10  "status": "Pending",
11  "registrationDate": "2022-07-02"
```

Register Doctor: Success (Request & Response)

The email from Amazon Web Services is as follows:

Amazon Web Services <no-reply-aws@amazon.com>
to me ▾
6:56 PM (1 minute ago) ⚡ ⓘ

Dear Amazon Web Services Customer,

We have received a request to authorize this email address for use with Amazon SES and Amazon Pinpoint in region US East (N. Virginia). If you requested this verification, please go to the following URL to confirm that you are authorized to use this email address:

https://email-verification.us-east-1.amazonaws.com/?Context=890167293239&X-Amz-Date=20220702T132605Z&Identity.IdentityName=ruhan008%40gmail.com&X-Amz-Algorithm=AWS4-HMAC-SHA256&Identity.IdentityType=EmailAddress&X-Amz-SignedHeaders=host&X-Amz-Credential=AKIAVM67ZIEFRDECB3HF%2F20220702%2Fus-east-1%2Fses%2Faws4_request&Operation=ConfirmVerification&Namespace=Bacon&X-Amz-Signature=28f3439353181a42cbd83255fb7e0fbde802a5b74769f81e2ca3fe0ec46a2ca

Reply Forward



Email Verification

The screenshot shows the Postman interface with a request to 'Register Doctor'. The request method is POST, the URL is `http://{{environment}}{{(doctor-service-context-path)}}/doctors`, and the status code is 400 Bad Request. The response body is a JSON object with an 'errorCode' of 'ERR_INVALID_INPUT', an 'errorMessage' of 'Invalid input. Parameter name: ', and an 'errorFields' array containing several validation errors for fields like 'pan', 'mobile', 'emailId', 'dob', 'firstName', and 'lastName'.

```
1 "errorCode": "ERR_INVALID_INPUT",
2 "errorMessage": "Invalid input. Parameter name: ",
3 "errorFields": [
4     "pan - Pan card should be a mix of 10 alphanumeric
5         characters",
6     "mobile - Mobile number is invalid",
7     "emailId - Please enter a valid email id",
8     "dob - Please enter date in format - YYYY-MM-DD",
9     "firstName - First Name field cant be null",
10    "lastName - Last Name field cant be null"
11 ]
12 ]
```

Doctor Registration: Error response for invalid field values.

The screenshot shows the Postman interface with a request to 'Upload doctor documents'. The request method is POST, the URL is `http://{{environment}}{{(doctor-service-context-path)}}/doctors/62c0476d1948a6022c16be3b/document`, and the status code is 200 OK. The response body is a simple text message: '1 File(s) uploaded successfully'.

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> files	Screen...	
<input checked="" type="checkbox"/> files	Hina S...	

Upload doctor documents: Success

The screenshot shows the Postman interface with the following details:

- Request Method:** PUT
- Request URL:** `({{environment}}){{(doctor-service-context-path)}}/doctors/62c0476d1948a6022c16be3b/approve`
- Body (JSON):**

```
1  "approvedBy": "Ruhan",
2  "approverComments": "Documents look good"
```
- Response Status:** 200 OK
- Response Time:** 962 ms
- Response Size:** 637 B
- Response Body (Pretty JSON):**

```
1  {
2    "id": "62c0476d1948a6022c16be3b",
3    "firstName": "Sadiq",
4    "lastName": "Umar",
5    "speciality": "GENERAL_PHYSICIAN",
6    "dob": "1965-10-25",
7    "mobile": "9900352601",
8    "emailId": "ruhan008@gmail.com",
9    "pan": "KKAPS8132M",
10   "status": "Active",
11   "approvedBy": "Ruhan",
12   "approverComments": "Documents look good",
13   "registrationDate": "2022-07-02",
14   "verificationDate": "2022-07-02"
15 }
```

Approve Doctor Registration with Admin token in authorization header

The screenshot shows an email in the inbox with the following details:

- Subject:** BMC - Application Approved
- From:** ruhan008@gmail.com via amazonses.com
- To:** me
- Date:** 7:05 PM (1 minute ago)
- Content:**

Hi Dr Sadiq Umar,
Congratulations ! Your Registration is approved.
Welcome aboard, we are pleased to have you with us.
Please follow the next steps to upload your availability slots and start taking appointments.

Thanks & Regards,
BookMyConsultation
- Warning:** Be careful with this message. This may be a spoofed message. The message claims to have been sent from your account, but Gmail couldn't verify the actual source. Avoid clicking links or replying with sensitive information, unless you are sure you actually sent this message. (No need to reset your password, the real sender does not actually have access to your account!)
- Buttons:** Report spam, Looks safe

Approval Email Confirmation.

The screenshot shows the Postman interface with a request to 'Doctor Approval'. The status bar indicates a 403 Forbidden error with a timestamp of 2022-07-02T13:38:19.961353, a duration of 262 ms, and a size of 469 B. The response body is a JSON object:

```
1 "timestamp": "2022-07-02T13:38:19.961353",
2 "status": 403,
3 "error": "Forbidden - You're not authorized to access this
4         feature",
5 "path": "/doctors/62c0476d1948a6022c16be3b/approve"
```

Forbidden error response for accessing approval with user token

The screenshot shows the Postman interface with a request to 'Doctor Rejection'. The status bar indicates a 200 OK response with a timestamp of 2022-07-02T13:38:19.961353, a duration of 306 ms, and a size of 643 B. The response body is a JSON object:

```
1 {
2     "id": "62c0476d1948a6022c16be3b",
3     "firstName": "Sadiq",
4     "lastName": "Umar",
5     "speciality": "GENERAL_PHYSICIAN",
6     "dob": "1965-10-25",
7     "mobile": "9900352501",
8     "emailId": "ruhan000@gmail.com",
9     "pan": "KKAPS8132M",
10    "status": "Rejected",
11    "approvedBy": "Ruhan",
12    "approverComments": "Application is rejected",
13    "registrationDate": "2022-07-02",
14    "verificationDate": "2022-07-02"
15 }
```

Doctor Rejection with admin token



BMC - Application Rejected ➔ Inbox



ruhan008@gmail.com via amazonses.com
to me ▾



Be careful with this message

This may be a spoofed message. The message claims to have been sent from you. Do not reply with sensitive information, unless you are sure you actually sent this message. (This message was sent from an unverified address and we do not have access to your account!)

[Report spam](#)

[Looks safe](#)

Hi Dr Sadiq Umar,

We are sorry to inform you but your application has been rejected.

We wish you the best of luck for future.

Note: If comments suggest there is additional information required from your side then please do the needful so that we can have you onboarded as soon as possible.

Kindly find the comments below.

Comments: Application is rejected

Thanks & Regards,

BookMyConsultation

Doctor Rejection Email

The screenshot shows the Postman interface with a failed API call. The URL is `PUT {{(environment)}{{(doctor-service-context-path)}}}/doctors/62c0476d19416be3b/reject`. The response status is 404 Not Found, with the message "Requested resource is not available".

```
1
2     "approvedBy": "Ruhan",
3     "approverComments": "Application is rejected"
4 
```

```
1
2     "errorCode": "ERR_RESOURCE_NOT_FOUND",
3     "errorMessage": "Requested resource is not available"
4 
```

When invalid doctorId is sent. (Error fields is null and hence json property doesn't send it in response.

The screenshot shows the Postman interface with a successful API call. The URL is `GET {{(environment)}{{(doctor-service-context-path)}}}/doctors?status=Pending`. The response status is 200 OK, with a total of 278 ms and 2.35 KB. The response body contains two doctor objects.

KEY	VALUE	DESCRIPTION
speciality	Dentist	
<input checked="" type="checkbox"/> status	Pending	

```
1
2
3     "id": "62bd41ff3cff4736eb321724",
4     "firstName": "Hina",
5     "lastName": "Shaheen",
6     "speciality": "GENERAL_PHYSICIAN",
7     "dob": "Sun Jun 02 18:30:00 GMT 1974",
8     "mobile": "9449426549",
9     "emailId": "ruhan008@gmail.com",
10    "pan": "KKAPS1032M",
11    "status": "Pending",
12    "registrationDate": "2022-06-29",
13    "averageDoctorRating": 5.0
14 },
15 {
16     "id": "62bb3a9380f7c9222490b561",
17     "firstName": "Syed",
18     "lastName": "Farhan",
19     "speciality": "GENERAL_PHYSICIAN",
20     "dob": "Sun Oct 13 18:30:00 GMT 1996",
21     "mobile": "99009352601",
22     "emailId": "ruhan008@gmail.com",
23     "pan": "KKAPS1032M",
24     "status": "Pending",
25     "registrationDate": "2022-06-27",
26     "averageDoctorRating": 4.5
27 }
```

Get doctors list for status=Pending

BookMyConsultation - Capstone Project / doctor-service / Get Doctors List - Sorted on Rating and Criteria

GET {{(environment)}{{(doctor-service-context-path)}}}/doctors?status=Active

Params Auth Headers (7) Body Pre-req. Tests Settings ...

Query Params

KEY	VALUE	DESCRIPTION
speciality	Dentist	
<input checked="" type="checkbox"/> status	Active	

Body

Pretty Raw Preview Visualize JSON ...

```
1 {
2   "id": "62bd3ce8933921119481c5bd",
3     "firstName": "Syed",
4     "lastName": "Ruhan",
5     "speciality": "Dentist",
6     "dob": "Sun Oct 13 00:00:00 GMT 1996",
7     "mobile": "9900352601",
8     "emailId": "ruhan008@gmail.com",
9     "pan": "KKAPS8132M",
10    "status": "Active",
11    "approvedBy": "Ruhan",
12    "approverComments": "Application is approved",
13    "registrationDate": "2022-06-29",
14    "verificationDate": "2022-06-29",
15    "averageDoctorRating": 5.0
16  },
17  {
18    "id": "62b9a265bbe2af58b19ffa5c",
19      "firstName": "Syed",
20      "lastName": "Ruhan",
21      "speciality": "GENERAL_PHYSICIAN",
22      "dob": "Thu Oct 13 18:30:00 GMT 2022",
23      "mobile": "9900352601",
24      "emailId": "ruhan008@gmail.com",
25      "pan": "KKAPS1032M",
26    ...
27  }
```

Runner Trash ⓘ 07:22 PM 02-07-2022 8

Get doctors list for status=Active

POST Login with User Token fr POST Login with Admin Token GET Get Doctors List - Sort POST Register Doctor ...

bmc-gateway

BookMyConsultation - Capstone Project / doctor-service / Register Doctor

POST {{(environment)}{{(doctor-service-context-path)}}}/doctors

Params Auth Headers (9) Body Pre-req. Tests Settings ...

Body

Pretty Raw Preview Visualize JSON ...

```
1 {
2   "firstName": "Syed",
3   "lastName": "Fazhan",
4   "dob": "2000-08-24",
5   "speciality": "Dentist",
6   "mobile": "9900352601",
7   "emailId": "ruhan008@gmail.com",
8   "pan": "KKAPS8132M"
9 }
```

```
1 {
2   "id": "62c04ebd1948a6022c16be3e",
3     "firstName": "Syed",
4     "lastName": "Fazhan",
5     "speciality": "Dentist",
6     "dob": "2000-08-24",
7     "mobile": "9900352601",
8     "emailId": "ruhan008@gmail.com",
9     "pan": "KKAPS8132M",
10    "status": "Pending",
11    "registrationDate": "2022-07-02"
12  }
```

Runner Trash ⓘ 07:27 PM 02-07-2022 8

Upload a doctor with speciality Dentist

POST Login with User Token | POST Login with Admin Token | GET Get Doctors List - Sort | POST Register Doctor | + ... | bmc-gateway | BookMyConsultation - Capstone Project / doctor-service / Get Doctors List - Sorted on Rating and Criteria | Save | ... | ⚙️ | 🖊️ | 📄 | ↻

GET {{environment}}{{doctor-service-context-path}}/doctors?speciality=Dentist&status=Pending | Send | ↻

Params Auth Headers (7) Body Pre-req. Tests Settings | Body | 200 OK 252 ms 1011 B | Save Response | ↻

Query Params

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> speciality	Dentist	
<input checked="" type="checkbox"/> status	Pending	

Body | Pretty | Raw | Preview | Visualize | JSON | ↻ | 🗂️ | 🔎 | ⓘ

```
13 },
14 {
15     "id": "62c04e8d1948a6022c16be3d",
16     "firstName": "Syed",
17     "lastName": "Ruhan",
18     "speciality": "Dentist",
19     "dob": "2000-08-24",
20     "mobile": "9900352601",
21     "emailId": "ruhan008@gmail.com",
22     "pan": "KKAPS8132M",
23     "status": "Pending",
24     "registrationDate": "2022-07-02"
25 },
26 {
27     "id": "62c04ebd1948a6022c16be3e",
28     "firstName": "Syed",
29     "lastName": "Farhan",
30     "speciality": "Dentist",
31     "dob": "2000-08-24",
32     "mobile": "9900352601",
33     "emailId": "ruhan008@gmail.com",
34     "pan": "KKAPS8132M",
35     "status": "Pending",
36     "registrationDate": "2022-07-02"
37 }
38 }
```

Runner | Trash | ⓘ | 25°C | ENG | 07:27 PM | 02-07-2022 | 📌

Get doctor with status=Pending and speciality=Dentist

POST Login with User Token | POST Login with Admin Token | GET Get Doctor based on ID | + ... | bmc-gateway | BookMyConsultation - Capstone Project / doctor-service / Get Doctor based on ID | Save | ... | ⚙️ | 🖊️ | 📄 | ↻

GET {{environment}}{{doctor-service-context-path}}/doctors/62c0476d1948a6022c16be3b | Send | ↻

Params Auth Headers (7) Body Pre-req. Tests Settings | Body | 200 OK 263 ms 643 B | Save Response | ↻

Headers | 6 hidden

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> Authorization	Bearer eyJ...	

Body | Pretty | Raw | Preview | Visualize | JSON | ↻ | 🗂️ | 🔎 | ⓘ

```
1,
2     "id": "62c0476d1948a6022c16be3b",
3     "firstName": "Sadiq",
4     "lastName": "Umar",
5     "speciality": "GENERAL_PHYSICIAN",
6     "dob": "1965-10-25",
7     "mobile": "9900352601",
8     "emailId": "ruhan008@gmail.com",
9     "pan": "KKAPS8132M",
10    "status": "Rejected",
11    "approvedBy": "Ruhan",
12    "approverComments": "Application is rejected",
13    "registrationDate": "2022-07-02",
14    "verificationDate": "2022-07-02"
15 }
```

Runner | Trash | ⓘ | 25°C | ENG | 07:30 PM | 02-07-2022 | 📌

Get doctor by id

The screenshot shows the Postman application interface. The request URL is `({{environment}}{{doctor-service-context-path}}/doctors/62c0476d1948a6022c16be3b`. The response status is 404 Not Found, with a response body containing:

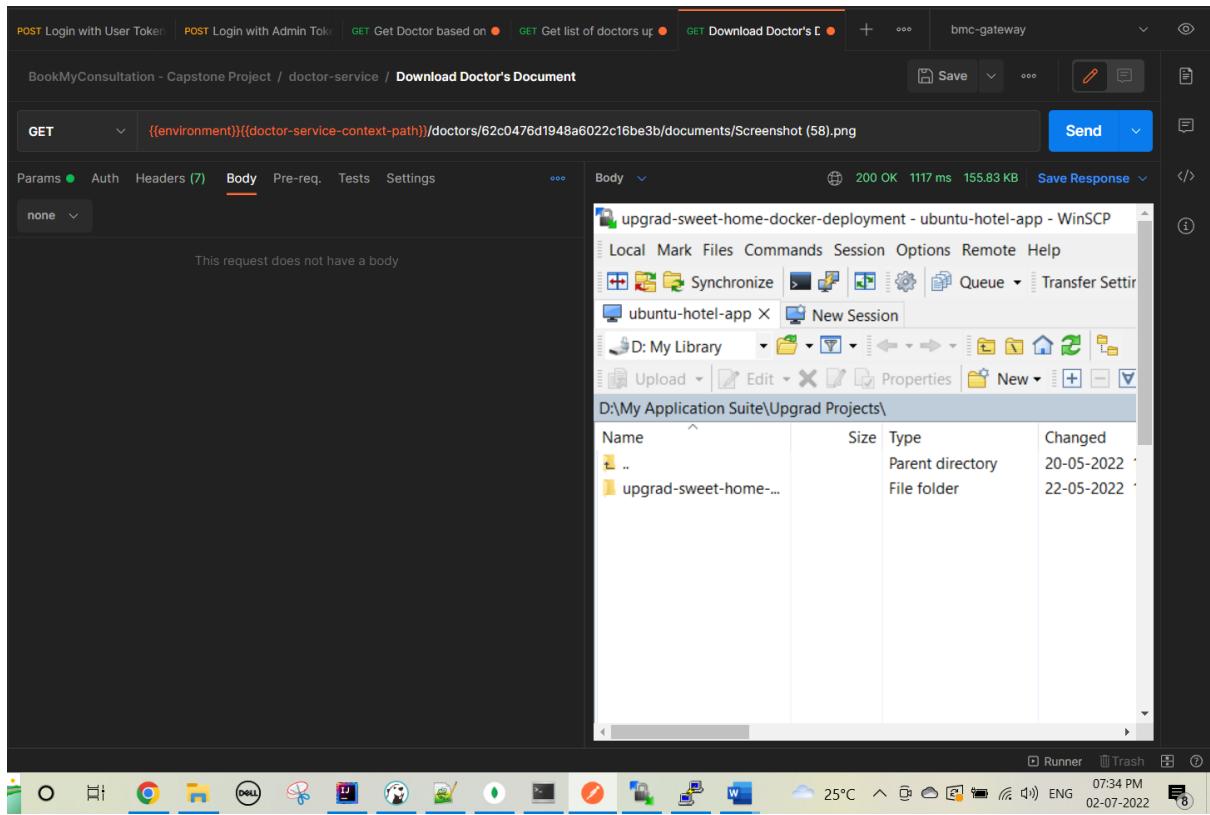
```
1 "errorCode": "ERR_RESOURCE_NOT_FOUND",
2 "errorMessage": "Requested resource is not available"
```

Invalid Doctor Id

The screenshot shows the Postman application interface. The request URL is `({{environment}}{{doctor-service-context-path}}/doctors/62c0476d1948a6022c16be3b/documents/metadata`. The response status is 200 OK, with a response body containing:

```
1 [
2   {
3     "name": "Hina Shaheen - Aadhaar Back.pdf",
4     "path": "Screenshot (58).png"
5   }
6 ]
```

Get a list of uploaded documents by a doctor



Download a document uploaded by a doctor.

User Service Request-Response:

The screenshot shows the Postman interface with a successful user registration request. The request URL is `POST {{(environment)}{{(user-service-context-path)}}}/users`. The response status is 200 OK with a response time of 3.31s and a size of 470 B. The response body is:

```
1 "id": "62c08098be6dea305b12b013",
2 "firstName": "Saumya",
3 "lastName": "Verma",
4 "dob": "1996-01-16",
5 "emailId": "ruhan008@gmail.com",
6 "mobile": "7017614371"
7
8
9 "createdDate": "2022-07-02"
```

Register User: Success

The screenshot shows the Postman interface with an invalid input error response. The request URL is `POST {{(environment)}{{(user-service-context-path)}}}/users`. The response status is 400 Bad Request with a response time of 285 ms and a size of 611 B. The response body is:

```
1 "errorCode": "ERR_INVALID_INPUT",
2 "errorMessage": "Invalid input. Parameter name: ",
3 "errorFields": [
4     "firstName - Please enter a valid name",
5     "mobile - Mobile number is invalid",
6     "lastName - Please enter a valid name",
7     "emailId - Please enter a valid email id",
8     "dob - Please enter date in format - YYYY-MM-DD"
9 ]
10
11 ]
```

Register User: Invalid input error message

The screenshot shows the Postman application interface. At the top, there are three tabs: "POST Login with User Token", "POST Login with Admin Token", and "GET Get User By ID". The "GET Get User By ID" tab is active. Below the tabs, the URL is set to `http://{{environment}}{{(user-service-context-path)}/users/62c08098be6dea305b12b013}`. The "Headers" section contains a single header: "Authorization: Bearer eyJ...". The response status is 200 OK, with a response time of 324 ms and a body size of 470 B. The response body is displayed in JSON format:

```
1 "id": "62c08098be6dea305b12b013",
2   "firstName": "Saumya",
3   "lastName": "Verma",
4   "dob": "1996-01-16",
5   "emailId": "ruhan008@gmail.com",
6   "mobile": "7817614371",
7   "createdDate": "2022-07-02"
```

Get user details based on ID

The screenshot shows the Postman application interface. The "GET Get User By ID" tab is active. The URL is set to `http://{{environment}}{{(user-service-context-path)}/users/62c08098be6dea312b013}`. The "Headers" section contains a single header: "Authorization: Bearer eyJ...". The response status is 404 Not Found, with a response time of 254 ms and a body size of 392 B. The response body is displayed in JSON format:

```
1   "errorCode": "ERR_RESOURCE_NOT_FOUND",
2   "errorMessage": "Requested resource is not valid"
```

Invalid user id request

The screenshot shows the Postman application interface. At the top, there are several tabs: "POST Login with User Token", "POST Login with Admin Token", "GET Get User By ID", "POST User Upload Document", and "bmc-gateway". The "POST User Upload Document" tab is active. Below the tabs, the URL is specified as `POST {{environment}}{{user-service-context-path}}/users/62c08098be6dea305b12b013/documents`. The "Body" tab is selected, showing a "form-data" section with three entries: "files" (value: "Screenshot (39).png"), "files" (value: "Screenshot (44).png"), and "files" (value: "Upgrad Docker.pdf"). The response status is 200 OK, with a time of 2.34 s and a size of 328 B. The response body is: "1 File(s) uploaded successfully."

Upload user documents to s3: Success

The screenshot shows the Postman application interface. At the top, there are several tabs: "POST Login with User Token", "POST Login with Admin Token", "GET Get User By ID", "POST User Upload Document", "GET Get list of user uploaded documents", and "bmc-gateway". The "GET Get list of user uploaded documents" tab is active. Below the tabs, the URL is specified as `GET {{environment}}{{user-service-context-path}}/users/62c08098be6dea305b12b013/documents/metadata`. The "Headers" tab is selected, showing a single entry: "Authorization" (value: "Bearer eyJ..."). The response status is 200 OK, with a time of 343 ms and a size of 363 B. The response body is displayed in JSON format, showing a list of file names: ["Screenshot (39).png", "Screenshot (44).png", "Upgrad Docker.pdf"].

Get list of documents uploaded by user

The screenshot shows a Postman collection named "bmc-gateway" with several API endpoints listed at the top. The current request is a GET operation to "Download document uploaded by user". The URL is specified as `{(environment)}{{user-service-context-path}}/users/62c08098be6dea305b12b013/documents/Upgrad Docker.pdf`. The response status is 200 OK, with a duration of 1283 ms and a size of 328.35 KB. The response body contains the text "upGrad" and a section titled "Dockerfile instructions" which provides basic Dockerfile commands like FROM, MAINTAINER, ADD, WORKDIR, ENV, and ENTRYPOINT. Below this, it lists other Docker instructions like RUN, CMD, EXPOSE, and VOLUME. The response also includes a "Docker Images" section with some common commands.

Download the document uploaded by user based on document name. (Make sure to give the full name including (.) extension.

Appointment Service Request-Response:

The screenshot shows a Postman collection named "bmc-gateway" with several API endpoints listed at the top. The current request is a POST operation to "Upload Doctor Availability". The URL is specified as `{(environment)}{{appointment-service-context-path}}/doctor/62c0476d1948a6022c16be3b/availability`. The response status is 200 OK, with a duration of 1227 ms and a size of 257 B. The response body is empty, indicated by a single digit "1". The request body is shown in JSON format, containing a nested object "availabilityMap" with two entries for dates "2022-07-04" and "2022-07-05", each mapping to a list of time intervals like "11AM-12PM" and "11PM-11:30PM".

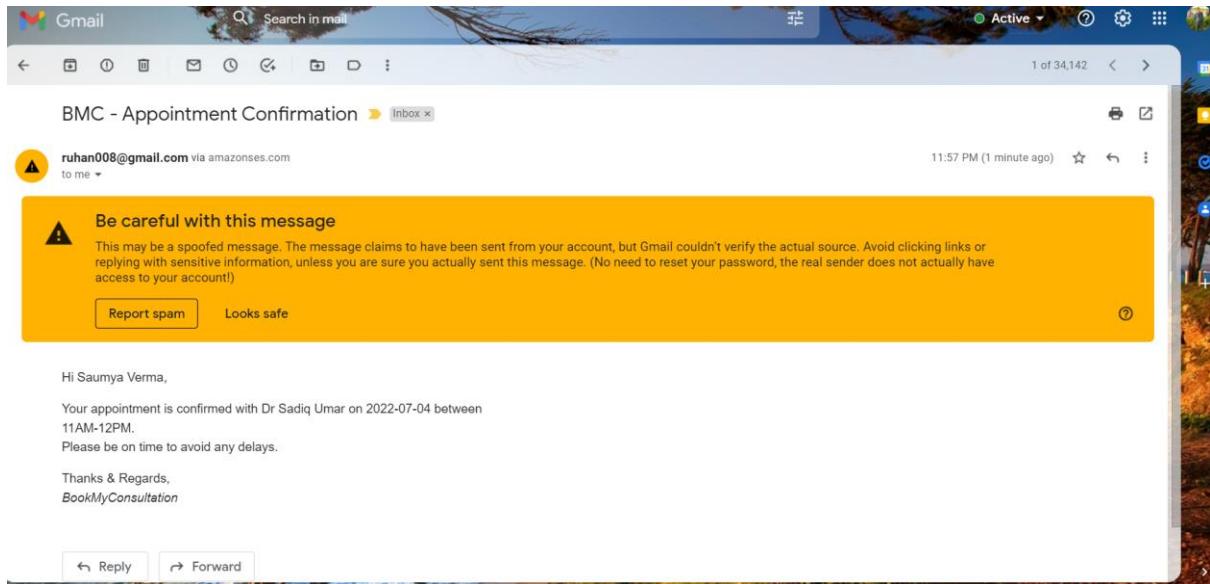
Update Doctor availability: Success

The screenshot shows the Postman application interface. At the top, there are several tabs: POST Login with User Token, POST Login with Admin Token, POST Upload Doctor Availability, and GET Get Doctor Availability (which is currently selected). The URL in the header bar is `BookMyConsultation - Capstone Project / appointment-service / Get Doctor Availability`. Below the header, a search bar contains the placeholder `({{environment}}{{{appointment-service-context-path}}}/doctor/62c0476d1948a6022c16be3b/availability`. On the left, a sidebar shows the request method (GET), the URL, and various settings like Headers (7), Body, Pre-req., Tests, and Settings. The Headers section includes a table with one row: "Authorization" set to "Bearer eyJ...". The Body section shows a JSON response with a "doctorId" field and an "availabilityMap" object containing two entries for dates (2022-07-05 and 2022-07-04) with their respective time slots. The response status is 200 OK with 290 ms duration and 441 B size. A "Send" button is visible at the top right. The bottom of the screen shows the Windows taskbar with icons for various applications.

Get doctor availability based on ID

The screenshot shows the Postman application interface. At the top, there are several tabs: POST Login with User Token, POST Login with Admin Token, POST Book appointment with..., and POST Book appointment with doctor (which is currently selected). The URL in the header bar is `BookMyConsultation - Capstone Project / appointment-service / Book appointment with doctor`. Below the header, a search bar contains the placeholder `({{environment}}{{{appointment-service-context-path}}}/appointments`. On the left, a sidebar shows the request method (POST), the URL, and various settings like Headers (9), Body (selected), Pre-req., Tests, and Settings. The Body section shows a JSON payload with fields: "doctorId" (set to "62c0476d1948a6022c16be3b"), "userId" (set to "62c08098be6dea305b12b013"), "appointmentDate" (set to "2022-07-04"), and "timeSlot" (set to "11AM-12PM"). The response status is 200 OK with 2.86 s duration and 334 B size. A "Send" button is visible at the top right. The bottom of the screen shows the Windows taskbar with icons for various applications.

Book appointment with doctor: Success



Email notification to user regarding appointment confirmation.

```
1 "appointmentId": "1267f34d-4383-4a83-be38-333c37e114d1",
2 "appointmentDate": "2022-07-04",
3 "createdDate": "2022-07-02T18:27:28",
4 "doctorId": "62c0476d1948a6022c16be3b",
5 "status": "PendingPayment",
6 "timeSlot": "11AM-12PM",
7 "userId": "62c08098be6dea305b12b013",
8 "userEmailId": "ruhan008@gmail.com",
9 "userName": "Saumya Verma",
10 "doctorName": "Sadiq Umar",
11 "doctorEmailId": "ruhan008@gmail.com"
```

Get appointment details based on appointment id.

POST Login with User Token | POST Login with Admin Token | GET Get List of Appointments for a User

BookMyConsultation - Capstone Project / appointment-service / Get List of Appointments for a User

GET {{environment}}{{appointment-service-context-path}}/users/62c08098be6dea305b12b013/appointments

Params Auth Headers (7) Body Pre-req. Tests Settings

Headers (6 hidden)

KEY	VALUE	DESCRIPTION	...	Bulk Edit	Presets
<input checked="" type="checkbox"/> Authorization	Bearer eyJ...				
Key	Value	Description			

Body

Pretty Raw Preview Visualize JSON

```
1 {  
2     "appointmentId": "1267f34d-4383-4a83-be38-333c37e114d1",  
3     "appointmentDate": "2022-07-04",  
4     "createdDate": "2022-07-02T18:27:28",  
5     "doctorId": "62c0476d1948a6022c16be3b",  
6     "status": "PendingPayment",  
7     "timeSlot": "11AM-12PM",  
8     "userId": "62c08098be6dea305b12b013",  
9     "userEmailId": "ruhan008@gmail.com",  
10    "userName": "Saumya Verma",  
11    "doctorName": "Sadiq Umar",  
12    "doctorEmailId": "ruhan008@gmail.com"  
13 }  
14  
15
```

200 OK 254 ms 670 B Save Response

Runner Trash 12:05 AM 03-07-2022

Get list of appointments booked by a user based on userId

BookMyConsultation - Capstone Project / appointment-service / Upload Prescription by Doctor

POST {{environment}}{{appointment-service-context-path}}/prescriptions

Params Auth Headers (9) Body Pre-req. Tests Settings

Body

raw JSON Beautify

Pretty Raw Preview Visualize JSON

```
1 {  
2     "appointmentId": "1267f34d-4383-4a83-be38-333c37e114d1",  
3     "doctorId": "62c0476d1948a6022c16be3b",  
4     "userId": "62c08098be6dea305b12b013",  
5     "diagnosis": "Teeth Cavity",  
6     "medicineList": [  
7         {  
8             "name": "Calpol",  
9             "dosage": "1 week",  
10            "frequency": "3 times a day",  
11            "remarks": "after food"  
12        },  
13        {  
14            "name": "Painkill",  
15            "dosage": "1 week",  
16            "frequency": "3 times a day",  
17            "remarks": "after food"  
18        }  
19    ]  
20 }
```

400 Bad Request 320 ms 425 B Save Response

Runner Trash 12:08 AM 03-07-2022

Upload Prescription: Failure, error due to payment pending from patient side.

The screenshot shows the Postman interface with a successful API call. The URL is `POST {{(environment)}{{(appointment-service-context-path)}}/prescriptions`. The response status is `200 OK` with `459 ms` and `257 B` size. The JSON response body is:

```
1 {
2     "appointmentId": "267f34d-4383-4a83-be38-333c37e114d1",
3     "doctorId": "62c0476d1948a6022c16be3b",
4     "userId": "62c08098be6dea305b12b013",
5     "diagnosis": "Teeth cavity",
6     "medicineList": [
7         {
8             "name": "Calpol",
9             "dosage": "1 week",
10            "frequency": "3 times a day",
11            "remarks": "after food"
12        },
13        {
14            "name": "Painkill",
15            "dosage": "1 week",
16            "frequency": "3 times a day",
17            "remarks": "after food"
18        }
19    ]
20 }
```

Upload prescription succeeds once user makes the payment and status is updated.

The screenshot shows an email from `ruhan008@gmail.com via amazoneses.com` to the user. The subject is `BMC - Prescription Uploaded`. The email body contains:

Hi Saumya Verma,
A prescription has been uploaded by Dr. Sadiq Umar on the portal with regard to your appointment.
We wish you a speedy recovery and also thank you for choosing us to serve you.
Please find the prescription details below.
Diagnosis: Teeth Cavity
Medicine List :
Medicine(name=Calpol, dosage=1 week, frequency=3 times a day, remarks=after food)
Medicine(name=Painkill, dosage=1 week, frequency=3 times a day, remarks=after food)
Thanks & Regards,
BookMyConsultation

Prescription email to user after successful upload from doctor.

Payment Service Request-Response:

The screenshot shows the Postman interface with the following details:

- Request URL:** {{environment}}{{payment-service-context-path}}/payments?appointmentId=1267f34d-4383-4a83-be38-333c37e114d1
- Method:** POST
- Headers:** Authorization: Bearer eyJ...
- Body (Pretty):**

```
1 "id": "62c09119570aaf5a0250a59b",
2 "appointmentId": "1267f34d-4383-4a83-be38-333c37e114d1",
3 "createdDate": "2022-07-02T18:40:25.456491",
4 "userName": "Saumya Verma",
5 "userEmailId": "ruhan008@gmail.com"
```
- Response Status:** 200 OK
- Time taken:** 2.83 s
- Size:** 490 B

Payment for appointment – Success

The screenshot shows a Gmail inbox with the following details:

- Subject:** BMC - Payment Confirmation for Appointment
- From:** ruhan008@gmail.com via amazonses.com
- Time:** 12:10 AM (1 minute ago)
- Message Content:**

Be careful with this message
This may be a spoofed message. The message claims to have been sent from your account, but Gmail couldn't verify the actual source. Avoid clicking links or replying with sensitive information, unless you are sure you actually sent this message. (No need to reset your password, the real sender does not actually have access to your account!)

Hi Saumya Verma,
This is to inform you that your payment has been received for appointment-id 1267f34d-4383-4a83-be38-333c37e114d1.
Your unique payment-id generated is 62c09119570aaf5a0250a59b.

Thanks & Regards,
BookMyConsultation
- Buttons:** Report spam, Looks safe

Payment email notification to user/patient

Rating Service Request-Response:

The screenshot shows the Postman interface with a successful POST request to the rating-service. The request URL is `POST {{environment}}{{(rating-service-context-path)}}/ratings`. The response status is 200 OK with a response body containing the message: "Thanks for the feedback! We have accepted your rating".

Submit rating for a doctor: Success

The screenshot shows the Postman interface with a successful GET request to the doctor-service. The request URL is `GET {{environment}}{{(doctor-service-context-path)}}/doctors/62c0476d1948a6022c16be3b`. The response status is 200 OK with a response body containing detailed information about a doctor, including their ID, first name, last name, speciality, and status.

KEY	VALUE	DESCRIPTION
id	62c0476d1948a6022c16be3b	
firstName	Sadiq	
lastName	Umar	
speciality	GENERAL_PHYSICIAN	
dob	1965-10-25	
mobile	9900352601	
emailId	ruhan008@gmail.com	
pan	KKAPS8132M	
status	Rejected	
approvedBy	Ruhan	
approverComments	Application is rejected	
registrationDate	2022-07-02	
verificationDate	2022-07-02	
averageDoctorRating	5.0	

Doctor rating updated in response after processing the message received from bmc-doctor topic sent by rating-service.

Docker-Compose Logs:

rating-service | 2022-07-02 18:50:32.375 INFO [rating-service,fefeb7cf5fcf78d,fefeb7cf5fcf78d] 1 --- [nio-8084-exec-1] o.a.kafka.common.utils.AppInfoParser : Kafka startTimeMs: 165678732369 rating-service | 2022-07-02 18:50:33.033 INFO [rating-service,] 1 --- [ad | producer-1] org.apache.kafka.clients.Metadata : [Producer clientId=producer-1] Cluster ID: D66tXgkQx0Yam9lcvahQ rating-service | 2022-07-02 18:50:33.044 INFO [rating-service,] 1 --- [ad | producer-1] o.a.k.c.p.internals.TransactionManager : [Producer clientId=producer-1] ProducerId set to 0 with epoch 0 rating-service | 2022-07-02 18:50:33.098 INFO [rating-service,fefeb7cf5fcf78d,fefeb7cf5fcf78d] 1 --- [nio-8084-exec-1] c.b.ratingservice.aspect.LoggingAspect : Exiting CLASS_N AMB:RatingServiceImpl METHOD NAME:sendRatingEventWith return value [null] and execution time of 0 minutes and 1 seconds rating-service | 2022-07-02 18:50:33.099 INFO [rating-service,fefeb7cf5fcf78d,fefeb7cf5fcf78d] 1 --- [nio-8084-exec-1] c.b.ratingservice.aspect.LoggingAspect : Exiting CLASS_N AMB:RatingServiceImpl METHOD NAME:processRatingReceivedWith return value [null] and execution time of 0 minutes and 1 seconds rating-service | 2022-07-02 18:50:33.103 INFO [rating-service,fefeb7cf5fcf78d,fefeb7cf5fcf78d] 1 --- [nio-8084-exec-1] c.b.ratingservice.aspect.LoggingAspect : Exiting CLASS_N AMB:RatingController METHOD NAME:acceptDoctorRatingWith return value [<200 OK, Thanks for the feedback! We have accepted your rating, !>] and execution time of 0 minutes and 1 seconds rating-service | 2022-07-02 18:50:33.155 INFO [rating-service,fefeb7cf5fcf78d,38b9d8624ecc40c1] 1 --- [ad | producer-1] c.b.product.KafkaProducer : Message sent successfully to kafka for key: DoctorRatingEvent value: DoctorRating(doctorId=d2c0476d1948a6022c16be3b, oneStarRatingsCount=0, twoStarRatingsCount=0, threeStarRatingsCount=0, fourStarRatingsCount=0, fiveStarRatingsCount=1, averageDoctorRating=5.0) and the partition is 0 doctor-service | 2022-07-02 18:50:33.170 INFO [doctor-service,fefeb7cf5fcf78d,cldoe59a685c5a7] 1 --- [ntainer#0-0-C-1] c.b.doctorservice.aspect.LoggingAspect : Entering CLASS_NAME:DoctorServiceImpl METHOD NAME:onMessage with input parameters [ConsumerRecord[topic = bmc-doctor, partition = 0, leaderEpoch = 0, offset = 24, CreateTime = 165678733033, serializedKeySize = 166, serializedValueSize = 198, recordHeaders = [RecordHeader[key = _TypId, value = {99, 111, 109, 46, 98, 111, 111, 107, 109, 121}], recordHeaders[key = _isDoc, value = {100, 111, 109, 46, 111, 111, 107, 109, 121}], recordHeaders[key = _isReadonly, value = {false}], key = DoctorRatingEvent, value = {doctorId=d2c0476d1948a6022c16be3b}, oneStarRatingsCount=0, twoStarRatingsCount=0, threeStarRatingsCount=0, fourStarRatingsCount=0, fiveStarRatingsCount=1, averageDoctorRating=5.0]] doctor-service | 2022-07-02 18:50:33.179 INFO [doctor-service,fefeb7cf5fcf78d,cldoe59a685c5a7] 1 --- [ntainer#0-0-C-1] c.b.doctor.consumer.KafkaConsumer : Received event: DoctorRatingEvent with message: [doctorId=d2c0476d1948a6022c16be3b, oneStarRatingsCount=0, twoStarRatingsCount=0, threeStarRatingsCount=0, fourStarRatingsCount=0, fiveStarRatingsCount=1, averageDoctorRating=5.0]] doctor-service | 2022-07-02 18:50:33.181 INFO [doctor-service,fefeb7cf5fcf78d,cldoe59a685c5a7] 1 --- [ntainer#0-0-C-1] c.b.doctorservice.aspect.LoggingAspect : Entering CLASS_NAME:DoctorServiceImpl METHOD NAME:processDoctorRatingReceivedWith input parameters [ConsumerRecord[topic = bmc-doctor, partition = 0, leaderEpoch = 0, offset = 24, CreateTime = 16567873833, serializedKeySize = 17, serializedValueSize = 198, headers = RecordHeaders[headers = [Recordheader[key = _TypId, value = {99, 111, 109, 46, 98, 111, 111, 107, 109, 121}], recordHeaders[key = _isDoc, value = {100, 111, 109, 46, 111, 111, 107, 109, 121}], recordHeaders[key = _isReadonly, value = {false}], key = DoctorRatingEvent, value = {doctorId=d2c0476d1948a6022c16be3b}, oneStarRatingsCount=0, twoStarRatingsCount=0, threeStarRatingsCount=0, fourStarRatingsCount=0, fiveStarRatingsCount=1, averageDoctorRating=5.0]] doctor-service | 2022-07-02 18:50:33.194 INFO [doctor-service,fefeb7cf5fcf78d,cldoe59a685c5a7] 1 --- [ntainer#0-0-C-1] c.b.doctorservice.aspect.LoggingAspect : Entering CLASS_NAME:CrudRepository METHOD NAME:findByIdWith input parameters [doctorId=d2c0476d1948a6022c16be3b] doctor-service | 2022-07-02 18:50:33.208 INFO [doctor-service,fefeb7cf5fcf78d,cldoe59a685c5a7] 1 --- [ntainer#0-0-C-1] c.b.doctorservice.aspect.LoggingAspect : Exiting CLASS_N AMB:CrudRepository METHOD NAME:findByIdWith return value [Optional<Doctor>(id=d2c0476d1948a6022c16be3b, firstName=Sadiq, lastName=Umar, specialty=GENERAL PHYSICIAN, dob=1965-10-25, mobile=9900352601, email=ruhan008@gmail.com, pan=KRAF88132M, status=Rejected, approvedBy=Ruhan, approverComments=Application is rejected, registrationDate=2022-07-02, verificationDate=2022-07-02, averageDoctorRating=5.0)] and execution time of 0 minutes and 0 seconds doctor-service | 2022-07-02 18:50:33.211 INFO [doctor-service,fefeb7cf5fcf78d,cldoe59a685c5a7] 1 --- [ntainer#0-0-C-1] c.b.doctorservice.aspect.LoggingAspect : Entering CLASS_NAME:CrudRepository METHOD NAME:save with input parameters [Doctor{id=d2c0476d1948a6022c16be3b, firstName=Sadiq, lastName=Umar, specialty=GENERAL PHYSICIAN, dob=1965-10-25, mobile=9900352601, email=ruhan008@gmail.com, pan=KRAF88132M, status=Rejected, approvedBy=Ruhan, approverComments=Application is rejected, registrationDate=2022-07-02, verificationDate=2022-07-02, averageDoctorRating=5.0}] and execution time of 0 minutes and 0 seconds doctor-service | 2022-07-02 18:50:33.305 INFO [doctor-service,fefeb7cf5fcf78d,cldoe59a685c5a7] 1 --- [ntainer#0-0-C-1] c.b.doctorservice.aspect.LoggingAspect : Exiting CLASS_N AMB:CrudRepository METHOD NAME:save with return value [Doctor{id=d2c0476d1948a6022c16be3b, firstName=Sadiq, lastName=Umar, specialty=GENERAL PHYSICIAN, dob=1965-10-25, mobile=9900352601, email=ruhan008@gmail.com, pan=KRAF88132M, status=Rejected, approvedBy=Ruhan, approverComments=Application is rejected, registrationDate=2022-07-02, verificationDate=2022-07-02, averageDoctorRating=5.0}] and execution time of 0 minutes and 0 seconds doctor-service | 2022-07-02 18:50:33.306 INFO [doctor-service,fefeb7cf5fcf78d,cldoe59a685c5a7] 1 --- [ntainer#0-0-C-1] c.b.doctorservice.aspect.LoggingAspect : Exiting CLASS_N AMB:DoctorServiceImpl METHOD NAME:processDoctorRatingReceivedWith return value [null] and execution time of 0 minutes and 0 seconds doctor-service | 2022-07-02 18:50:33.306 INFO [doctor-service,fefeb7cf5fcf78d,cldoe59a685c5a7] 1 --- [ntainer#0-0-C-1] c.b.doctorservice.aspect.LoggingAspect : Exiting CLASS_N AMB:KafkaConsumer METHOD NAME:OnMessage with return value [null] and execution time of 0 minutes and 0 seconds

As you can see from the logs the doctor-service consumes the doctor rating message received from rating-service on bmc-doctor topic and updates the same in its repository.

Similar events happen with notification service when it receives different events for email notifications. Below are few log screenshots for the same:

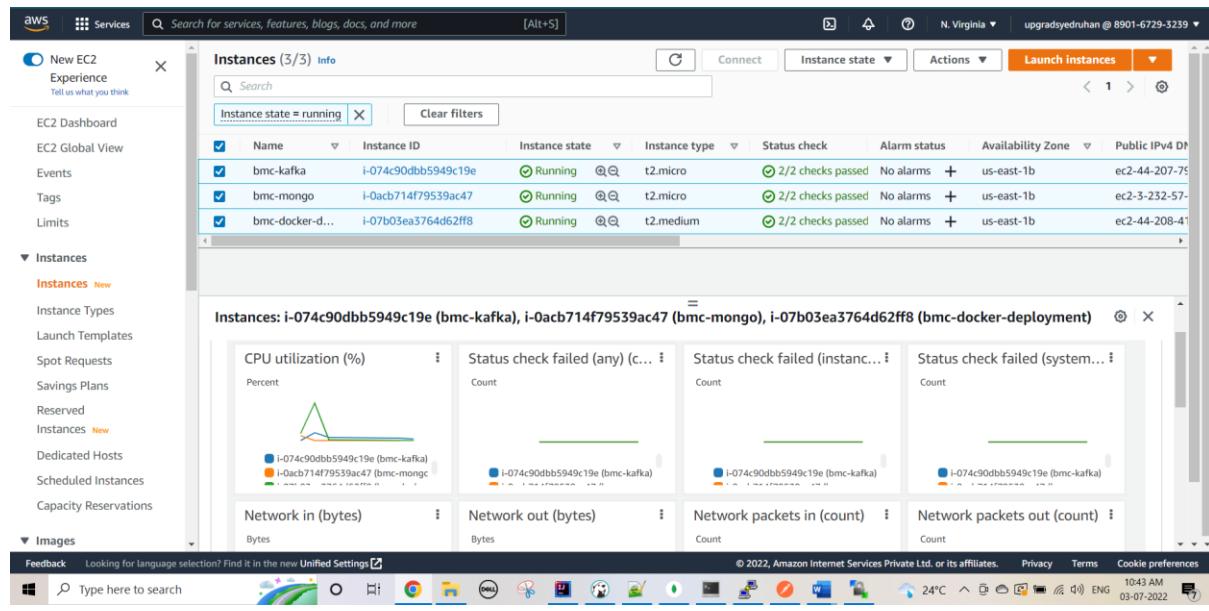
```
user,userName='Samya Verma',userId=ruhan008@gmail.com,appointmentId='1267f1d4-4403-a483-b3e8-33c7e114d1',diagnosis='Teeth Cavity',medicineList='[{"name":"Calpol","dosage":1,"week":1,"frequency":3,"times a day","remarks":"after food"}]' and execution time of 0 minutes and 0 seconds
appointment-service | 2022-07-02 18:43:04.844 INFO [appointment-service,200ade632da38091,200ade632da38091] 1 --- [nio-8082-exec-8] c.b.a.aspect.LoggingAspect : Entering C
LASS NAME:KafkaProducer METHOD NAME:sendEvent with input parameters [PrescriptionUploadEvent, {"id": "62c09b1833bce79ff28c47c", "userId": "62c08098b6de30a5b12013", "doctorId": "62c0476d1948a6022c16be3b", "doctorName": "Sadiq Umar", "userName": "Samya Verma", "userEmailId": "ruhan008@gmail.com", "appointmentId": "1267f1d4-4403-a483-b3e8-33c7e114d1"}, "diagnosis": "Teeth Cavity", "medicineList": [{"name": "Calpol", "dosage": "1 week", "frequency": "3 times a day", "remarks": "after food"}]] and execution time of 0 minutes and 0 seconds
appointment-service | 2022-07-02 18:43:04.849 INFO [appointment-service,200ade632da38091,200ade632da38091] 1 --- [nio-8082-exec-8] c.b.a.aspect.LoggingAspect : Exiting CL
ASS NAME:KafkaProducer METHOD NAME:sendEvent with return value [null] and execution time of 0 minutes and 0 seconds
appointment-service | 2022-07-02 18:43:04.850 INFO [appointment-service,200ade632da38091,200ade632da38091] 1 --- [nio-8082-exec-8] c.b.a.aspect.LoggingAspect : Exiting CL
ASS NAME:AppointmentService METHOD NAME:prescriptionForUser with input parameters [null] and execution time of 0 minutes and 0 seconds
appointment-service | 2022-07-02 18:43:04.853 INFO [notification-service,200ade632da38091,200ade632da38091] 1 --- [nio-8082-exec-8] c.b.a.aspect.LoggingAspect : Exiting CL
ASS NAME:AppointmentController METHOD NAME:uploadPrescriptionsForUser with input parameters [null] and execution time of 0 minutes and 0 seconds
appointment-service | 2022-07-02 18:43:04.855 INFO [notification-service,200ade632da38091,200ade632da38091] 1 --- [nio-8082-exec-8] c.b.a.aspect.LoggingAspect : Entering C
CLASS NAME:KafkaConsumer METHOD NAME:onMessage with input parameters [ConsumerRecord(topic = bmc-notification, partition = 0, leaderEpoch = 0, offset = 87, CreateTime = 1656787384040, serialized key size = 23, serialized value size = 469, headers = RecordHeaders(headers = [], isReadonly = false), key = PrescriptionUploadEvent, value = {"id": "62c09b1833bce79ff28c47c", "userId": "62c08098b6de30a5b12013", "doctorId": "62c0476d1948a6022c16be3b", "doctorName": "Sadiq Umar", "userName": "Samya Verma", "userEmailId": "ruhan008@gmail.com", "appointmentId": "1267f1d4-4403-a483-b3e8-33c7e114d1"}, "diagnosis": "Teeth Cavity", "medicineList": [{"name": "Calpol", "dosage": "1 week", "frequency": "3 times a day", "remarks": "after food"}], "name": "Painkill", "dosage": "1 week", "frequency": "3 times a day", "remarks": "after food"}]]
notification-service | 2022-07-02 18:43:04.855 INFO [notification-service,200ade632da38091,ac37ab029fe1fe4f] 1 --- [inainer#0-0-C-1] c.b.n.consumer.KafkaConsumer : Received event:PrescriptionUploadEvent with message: {"id": "62c09b1833bce79ff28c47c", "userId": "62c08098b6de30a5b12013", "doctorId": "62c0476d1948a6022c16be3b", "doctorName": "Sadiq Umar", "userName": "Samya Verma", "userEmailId": "ruhan008@gmail.com", "appointmentId": "1267f1d4-4403-a483-b3e8-33c7e114d1"}, "diagnosis": "Teeth Cavity", "medicineList": [{"name": "Calpol", "dosage": "1 week", "frequency": "3 times a day", "remarks": "after food"}], "name": "Painkill", "dosage": "1 week", "frequency": "3 times a day", "remarks": "after food"}]
appointment-service | 2022-07-02 18:43:04.856 INFO [notification-service,200ade632da38091,ac37ab029fe1fe4f] 1 --- [inainer#0-0-C-1] c.b.n.consumer.KafkaConsumer : Received event:PrescriptionUploadEvent with message: {"id": "62c09b1833bce79ff28c47c", "userId": "62c08098b6de30a5b12013", "doctorId": "62c0476d1948a6022c16be3b", "doctorName": "Sadiq Umar", "userName": "Samya Verma", "userEmailId": "ruhan008@gmail.com", "appointmentId": "1267f1d4-4403-a483-b3e8-33c7e114d1"}, "diagnosis": "Teeth Cavity", "medicineList": [{"name": "Calpol", "dosage": "1 week", "frequency": "3 times a day", "remarks": "after food"}], "name": "Painkill", "dosage": "1 week", "frequency": "3 times a day", "remarks": "after food"}]
notification-service | 2022-07-02 18:43:04.856 INFO [notification-service,200ade632da38091,ac37ab029fe1fe4f] 1 --- [inainer#0-0-C-1] c.b.n.consumer.KafkaConsumer : Received event:PrescriptionUploadEvent with message: {"id": "62c09b1833bce79ff28c47c", "userId": "62c08098b6de30a5b12013", "doctorId": "62c0476d1948a6022c16be3b", "doctorName": "Sadiq Umar", "userName": "Samya Verma", "userEmailId": "ruhan008@gmail.com", "appointmentId": "1267f1d4-4403-a483-b3e8-33c7e114d1"}, "diagnosis": "Teeth Cavity", "medicineList": [{"name": "Calpol", "dosage": "1 week", "frequency": "3 times a day", "remarks": "after food"}], "name": "Painkill", "dosage": "1 week", "frequency": "3 times a day", "remarks": "after food"}]
appointment-service | 2022-07-02 18:43:04.857 INFO [notification-service,200ade632da38091,ac37ab029fe1fe4f] 1 --- [ad | producer-1] c.b.a.producer.KafkaProducer : Message sent successfully to kafka for key: PrescriptionUploadEvent value: {"id": "62c09b1833bce79ff28c47c", "userId": "62c08098b6de30a5b12013", "doctorId": "62c0476d1948a6022c16be3b", "doctorName": "Sadiq Umar", "userName": "Samya Verma", "userEmailId": "ruhan008@gmail.com", "appointmentId": "1267f1d4-4403-a483-b3e8-33c7e114d1"}, "diagnosis": "Teeth Cavity", "medicineList": [{"name": "Calpol", "dosage": "1 week", "frequency": "3 times a day", "remarks": "after food"}]] and the partition is 0
notification-service | 2022-07-02 18:43:04.874 INFO [notification-service,200ade632da38091,ac37ab029fe1fe4f] 1 --- [inainer#0-0-C-1] c.b.n.consumer.KafkaConsumer : Received event:PrescriptionUploadEvent with input parameters [PrescriptionUploadEvent, {"id": "62c09b1833bce79ff28c47c", "userId": "62c08098b6de30a5b12013", "doctorId": "62c0476d1948a6022c16be3b", "doctorName": "Sadiq Umar", "userName": "Samya Verma", "userEmailId": "ruhan008@gmail.com", "appointmentId": "1267f1d4-4403-a483-b3e8-33c7e114d1"}, "diagnosis": "Teeth Cavity", "medicineList": [{"name": "Calpol", "dosage": "1 week", "frequency": "3 times a day", "remarks": "after food"}]] and execution time of 0 minutes and 0 seconds
notification-service | 2022-07-02 18:43:05.005 INFO [notification-service,200ade632da38091,ac37ab029fe1fe4f] 1 --- [inainer#0-0-C-1] c.b.n.consumer.KafkaConsumer : Received event:EmailService with input parameters [EmailService, {"id": "62c09b1833bce79ff28c47c", "userId": "62c08098b6de30a5b12013", "doctorId": "62c0476d1948a6022c16be3b", "doctorName": "Sadiq Umar", "userName": "Samya Verma", "userEmailId": "ruhan008@gmail.com", "appointmentId": "1267f1d4-4403-a483-b3e8-33c7e114d1"}, "diagnosis": "Teeth Cavity", "medicineList": [{"name": "Calpol", "dosage": "1 week", "frequency": "3 times a day", "remarks": "after food"}]] and execution time of 0 minutes and 0 seconds
notification-service | 2022-07-02 18:43:05.005 INFO [notification-service,200ade632da38091,ac37ab029fe1fe4f] 1 --- [inainer#0-0-C-1] c.b.n.consumer.KafkaConsumer : Email sent to id: ruhan008@gmail.com successfully
notification-service | 2022-07-02 18:43:05.305 INFO [notification-service,200ade632da38091,ac37ab029fe1fe4f] 1 --- [inainer#0-0-C-1] c.b.n.consumer.KafkaConsumer : Received event:EmailService with return value [null] and execution time of 0 minutes and 0 seconds
notification-service | 2022-07-02 18:43:05.305 INFO [notification-service,200ade632da38091,ac37ab029fe1fe4f] 1 --- [inainer#0-0-C-1] c.b.n.consumer.KafkaConsumer : Received event:EmailService with input parameters [EmailService, {"id": "62c09b1833bce79ff28c47c", "userId": "62c08098b6de30a5b12013", "doctorId": "62c0476d1948a6022c16be3b", "doctorName": "Sadiq Umar", "userName": "Samya Verma", "userEmailId": "ruhan008@gmail.com", "appointmentId": "1267f1d4-4403-a483-b3e8-33c7e114d1"}, "diagnosis": "Teeth Cavity", "medicineList": [{"name": "Calpol", "dosage": "1 week", "frequency": "3 times a day", "remarks": "after food"}]] and execution time of 0 minutes and 0 seconds
notification-service | 2022-07-02 18:43:05.306 INFO [notification-service,200ade632da38091,ac37ab029fe1fe4f] 1 --- [inainer#0-0-C-1] c.b.n.consumer.KafkaConsumer : Received event:EmailService with return value [null] and execution time of 0 minutes and 0 seconds
notification-service | 2022-07-02 18:43:05.307 INFO [notification-service,200ade632da38091,ac37ab029fe1fe4f] 1 --- [inainer#0-0-C-1] c.b.n.consumer.KafkaConsumer : Received event:EmailService with input parameters [EmailService, {"id": "62c09b1833bce79ff28c47c", "userId": "62c08098b6de30a5b12013", "doctorId": "62c0476d1948a6022c16be3b", "doctorName": "Sadiq Umar", "userName": "Samya Verma", "userEmailId": "ruhan008@gmail.com", "appointmentId": "1267f1d4-4403-a483-b3e8-33c7e114d1"}, "diagnosis": "Teeth Cavity", "medicineList": [{"name": "Calpol", "dosage": "1 week", "frequency": "3 times a day", "remarks": "after food"}]] and execution time of 0 minutes and 0 seconds
notification-service | 2022-07-02 18:43:05.307 INFO [notification-service,200ade632da38091,ac37ab029fe1fe4f] 1 --- [inainer#0-0-C-1] c.b.n.consumer.KafkaConsumer : Received event:EmailService with return value [null] and execution time of 0 minutes and 0 seconds
notification-service | 2022-07-02 18:43:05.307 INFO [notification-service,200ade632da38091,ac37ab029fe1fe4f] 1 --- [inainer#0-0-C-1] c.b.n.consumer.KafkaConsumer : Received event:EmailService with input parameters [EmailService, {"id": "62c09b1833bce79ff28c47c", "userId": "62c08098b6de30a5b12013", "doctorId": "62c0476d1948a6022c16be3b", "doctorName": "Sadiq Umar", "userName": "Samya Verma", "userEmailId": "ruhan008@gmail.com", "appointmentId": "1267f1d4-4403-a483-b3e8-33c7e114d1"}, "diagnosis": "Teeth Cavity", "medicineList": [{"name": "Calpol", "dosage": "1 week", "frequency": "3 times a day", "remarks": "after food"}]] and execution time of 0 minutes and 0 seconds
rating-service | 2022-07-02 18:50:31.245 INFO [rating-service,feferbf7cf5ff78d,fefefbf7cf5ff78d] 1 --- [nio-8084-exec-1] o.a.c.c.C.[localhost].[] : Initializing Sp
ring DispatcherServlet 'dispatcherServlet'
rating-service | 2022-07-02 18:50:31.245 INFO [rating-service,feferbf7cf5ff78d,fefefbf7cf5ff78d] 1 --- [nio-8084-exec-1] o.s.web.servlet.DispatcherServlet : Initializing Se
rvlet 'dispatcherServlet'
```

Prescription upload event

Payment Event

Kafka, MongoDB, MySQL and Ubuntu Setup Screenshots:

For Kafka, MongoDB and ubuntu I have created ec2 instances and installed kafka, MongoDB, Docker and Docker-Compose on them manually. As for MySQL I have used a RDS instance for the project. I did create a bookmyconsultation VPC to deploy all my resources under one VPC for project specifically. Please find the screenshots below:



EC-2 Instances bmc-kafka, bmc-mongo and bmc-docker-deployment

Amazon RDS

RDS > Databases > bookmyconsultation

bookmyconsultation

Summary

DB identifier	CPU	Status	Class
bookmyconsultation	2.71%	Starting	db.t3.micro
Role	Current activity	Engine	Region & AZ
Instance	0 Connections	MySQL Community	us-east-1b

Connectivity & security | Monitoring | Logs & events | Configuration | Maintenance & backups | Tags

Connectivity & security

Endpoint & port	Networking	Security
Endpoint bookmyconsultation.cqelf5aheptp.us-east-1.rds.amazonaws.com	Availability Zone us-east-1b	VPC VPC security groups bmc-mysql (sg-0b8f4aea1a20f4004)

Feedback Looking for language selection? Find it in the new Unified Settings [?](#)

© 2022, Amazon Internet Services Private Ltd. or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)

RDS MySQL Database instance

DBeaver 22.0.4 - Appointment

File Edit Navigate Search SQL Editor Database Window Help

Database Navigator > Projects > Appointment

Properties Data ER Diagram

Table Name: Appointment

Engine: InnoDB

Auto Increment: 0

CharSet: utf8mb4

Collation: utf8mb4_0900_ai_ci

Description:

```

CREATE TABLE `Appointment` (
    `appointment_id` varchar(64) NOT NULL,
    `appointment_date` date DEFAULT NULL,
    `created_date` datetime DEFAULT NULL,
    `doctor_id` varchar(255) DEFAULT NULL,
    `prior_medical_history` varchar(255) DEFAULT NULL,
    `status` varchar(255) DEFAULT NULL,
    `symptoms` varchar(255) DEFAULT NULL,
    `time_slot` varchar(255) DEFAULT NULL,
    `user_id` varchar(255) DEFAULT NULL,
    `user_email_id` varchar(255) DEFAULT NULL,
    `user_name` varchar(255) DEFAULT NULL,
    `doctor_name` varchar(255) DEFAULT NULL,
    `doctor_email_id` varchar(255) DEFAULT NULL,
    PRIMARY KEY (`appointment_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

```

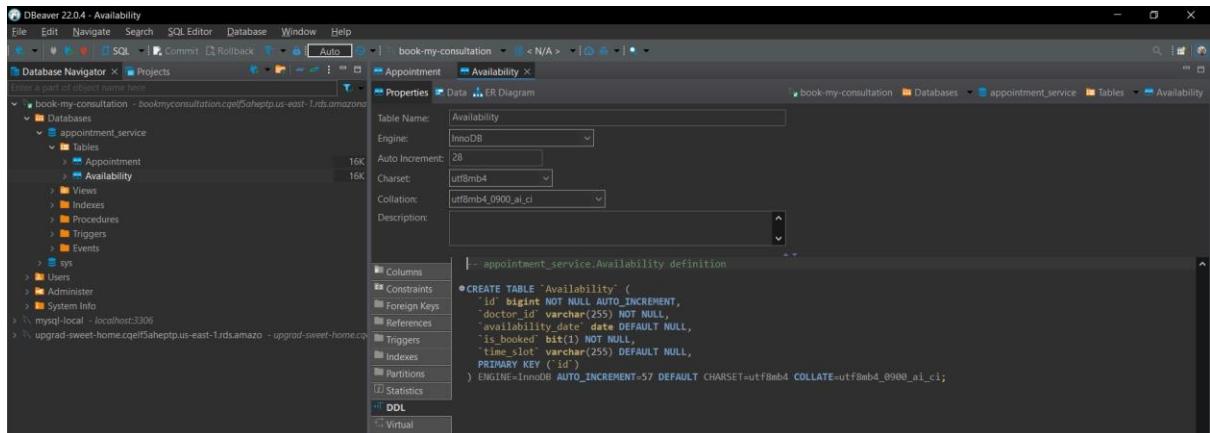
Appointment Table in MySQL

DDL:

```

CREATE TABLE `Appointment` (
    `appointment_id` varchar(64) NOT NULL,
    `appointment_date` date DEFAULT NULL,
    `created_date` datetime DEFAULT NULL,
    `doctor_id` varchar(255) DEFAULT NULL,
    `prior_medical_history` varchar(255) DEFAULT NULL,
    `status` varchar(255) DEFAULT NULL,
    `symptoms` varchar(255) DEFAULT NULL,
    `time_slot` varchar(255) DEFAULT NULL,
    `user_id` varchar(255) DEFAULT NULL,
    `user_email_id` varchar(255) DEFAULT NULL,
    `user_name` varchar(255) DEFAULT NULL,
    `doctor_name` varchar(255) DEFAULT NULL,
    `doctor_email_id` varchar(255) DEFAULT NULL,
    PRIMARY KEY (`appointment_id`)
)

```



Availability Table in MySQL

DDL:

```
CREATE TABLE `Availability` (
  `id` bigint NOT NULL AUTO_INCREMENT,
  `doctor_id` varchar(255) NOT NULL,
  `availability_date` date DEFAULT NULL,
  `is_booked` bit(1) NOT NULL,
  `time_slot` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`id`)
)
```

Inbound rules	Type	Protocol	Port range	Source	Description - optional
sgr-004b41651e58d55ee	HTTP	TCP	80	Custom	0.0.0.0/0
sgr-0fc8c856dedced6e5	Custom TCP	TCP	9092	Custom	0.0.0.0/0
sgr-078ca006c84aa2fa9	HTTPS	TCP	443	Custom	0.0.0.0/0
sgr-029b9bcc24af7d598	SSH	TCP	22	Custom	124.123.80.124/32
sgr-063d68146606c300f	Custom TCP	TCP	2181	Custom	0.0.0.0/0

bmc-kafka security group

AWS Services Search for services, features, blogs, docs, and more [Alt+S] N. Virginia upgradedyedruhan @ 8901-6729-3239

EC2 > Security Groups > sg-04886a75ee2a2f540 - bmc-mongoDB > Edit inbound rules

Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the instance.

Security group rule ID	Type <small>Info</small>	Protocol <small>Info</small>	Port range <small>Info</small>	Source <small>Info</small>	Description - optional <small>Info</small>
sgr-0b5517a1ea6a85e87	SSH	TCP	22	Custom	<input type="text" value="Q_"/> 124.123.80.124/32 <input type="button" value="Delete"/>
sgr-032c4b48e4e72794c	All TCP	TCP	0 - 65535	Custom	<input type="text" value="Q_"/> sg-0a56a99ea9eb79207 <input type="button" value="Delete"/>
sgr-0273bad368b8f7454	Custom TCP	TCP	27017	Custom	<input type="text" value="Q_"/> 124.123.80.124/32 <input type="button" value="Delete"/>

Cancel

Feedback Looking for language selection? Find it in the new Unified Settings 

Type here to search  © 2022, Amazon Internet Services Private Ltd. or its affiliates. Privacy Terms Cookie preferences 24°C 1054 AM 03-07-2022

bmc-mongo security group

AWS Services Search for services, features, blogs, docs, and more [Alt+S] N. Virginia upgradedyedruhan @ 8901-6729-3239

EC2 > Security Groups > sg-0a56a99ea9eb79207 - bmc-docker-deployment > Edit inbound rules

Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the instance.

Security group rule ID	Type <small>Info</small>	Protocol <small>Info</small>	Port range <small>Info</small>	Source <small>Info</small>	Description - optional <small>Info</small>
sgr-0bae01468e8004ed0	Custom TCP	TCP	9191	Custom	<input type="text" value="Q_"/> 124.123.80.124/32 <input type="button" value="Delete"/>
sgr-01f3066b54759491d	SSH	TCP	22	Custom	<input type="text" value="Q_"/> 124.123.80.124/32 <input type="button" value="Delete"/>

Cancel

Feedback Looking for language selection? Find it in the new Unified Settings 

Type here to search  © 2022, Amazon Internet Services Private Ltd. or its affiliates. Privacy Terms Cookie preferences 24°C 1055 AM 03-07-2022

bmc-docker-deployment security group

Screenshot of the AWS EC2 Security Groups page showing inbound rules for a MySQL instance.

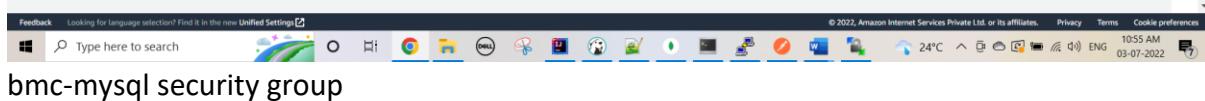
Inbound rules

Inbound rules control the incoming traffic that's allowed to reach the instance.

Security group rule ID	Type	Protocol	Port range	Source	Description - optional
sgr-079beaa4fd27a6734	All TCP	TCP	0 - 65535	Custom	Q, sg-0a56a99ea9eb79207 X
sgr-081922baed55480b6	MySQL/Aurora	TCP	3306	Custom	Q, 124.123.80.124/32 X

Add rule

Buttons: Cancel, Preview changes, Save rules



MongoDB Compass - 3.232.57.4:27017/doctor-service.Doctor

Connect View Collection Help

3.232.57.4:27017

8 DBS 10 COLLECTIONS

HOST: 3.232.57.4:27017 CLUSTER: Standalone EDITION: MongoDB 5.0.9 Community

My Queries Databases Filter your data admin appointment-service config doctor-service Doctor local payment-service rating-service user-service User

doctor-service.Doctor

4 DOCUMENTS 1 INDEXES

Documents Aggregations Schema Explain Plan Indexes Validation

ADD DATA FILTER { field: 'value' } OPTIONS FIND RESET ...

Displaying documents 1 - 16 of 16

```
_id: ObjectId('62c017c685412b29606328e5')
firstName: "Syed"
lastName: "Suhani"
speciality: "GENERAL_PHYSICIAN"
dob: "1996-10-14"
mobile: "9900352601"
emailId: "syedsuhani@gmail.com"
pan: "KXAP88132M"
status: "Pending"
registrationDate: 2022-07-01T18:30:00.000+00:00
__class: "com.bookmyconsultation.doctorservice.model.Doctor"
```

ObjectID String String String String String String String String Date String

>_MONGOSH

MongoDB connected locally from MongoDBCompass

```

ubuntu@ip-10-0-0-151: ~
attach      Attach local standard input, output, and error streams to a running container
build       Build an image from a Dockerfile
commit      Create a new image from a container's changes
cp          Copy files/folders between a container and the local filesystem
create      Create a new container
diff        Inspect changes to files or directories on a container's filesystem
events     Get real time events from the server
exec        Run a command in a running container
export      Export a container's filesystem as a tar archive
history    Show the history of an image
images     List images
import     Import the contents from a tarball to create a filesystem image
info        Display system-wide information
inspect    Inspect a container's configuration on Docker objects
kill        Kill one or more running containers
load        Load an image from a tar archive or STDIN
login      Log in to a Docker registry
logout    Log out from a Docker registry
logs       Fetch the logs of a container
pause      Pause all processes within one or more containers
port       List port mappings or a specific mapping for the container
ps         List containers
pull       Pull an image or a repository from a registry
push       Push an image or a repository to a registry
rename    Rename a container
restart   Restart one or more containers
rm        Remove one or more containers
rmi      Remove one or more images
run        Run a command in a new container
save      Save one or more images to a tar archive (streamed to STDOUT by default)
search    Search the Docker Hub for images
start     Start one or more stopped containers
stats     Display a stream of container(s) resource usage statistics
stop      Stop one or more running containers
tag       Create a tag TARGET_IMAGE that refers to SOURCE_IMAGE
top       Display the running processes of a container
unpause  Unpause all processes within one or more containers
update   Update configuration of one or more containers
version  Show the Docker version information
wait     Block until one or more containers stop, then print their exit codes

Run 'docker COMMAND --help' for more information on a command.

To get more help with docker, check out our guides at https://docs.docker.com/go/guides/
ubuntu@ip-10-0-0-151: ~$ sudo docker -v
Docker version 20.10.17, build 100c701
ubuntu@ip-10-0-0-151: ~$ docker-compose -v
docker-compose version 1.29.2, build unknown
ubuntu@ip-10-0-0-151: ~$ 
```

Docker and Docker-Compose setup on ubuntu

Future Enhancements: As part of future enhancements I have thought of the below implementation's for this project:

Monitoring Tools: As part of monitoring I will try to implement Prometheus, Grafana and Jaeger/Zipkin as part of the 3 pillars of monitoring, namely, Logging, Metrics Monitoring and distributed tracing.

Deployment: Deploy these services to on Kubernetes with all the previously mentioned monitoring tools.

Security: Implement Oauth2 as part of security to learn a new security implementation.

Final Note: I had a great time working on this project and learnt a lot of new things while working on these services and it was a fun journey to implement all the things I learnt through the course of this program.

I have tried to cover all the aspects of this project in terms of logic, design and config changes to be evaluated as mentioned in the evaluation guidelines, but in case of any issue or doubts while setting up the project please reach out to me on the below email ID or mobile number.

Email Id: ruhan008@gmail.com

Mobile: 9900352601

THANK YOU