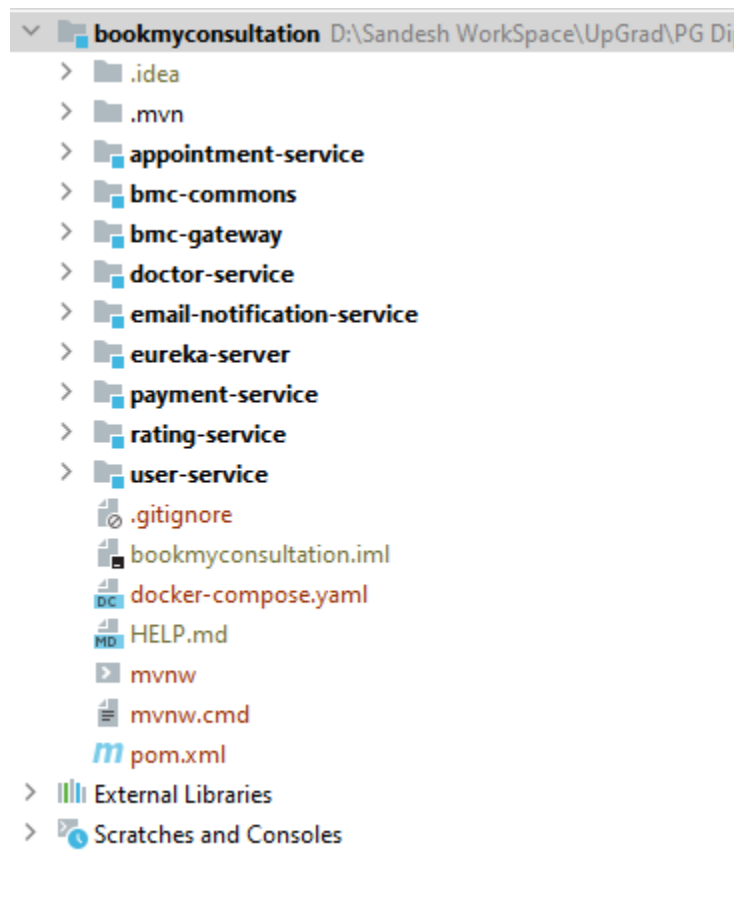


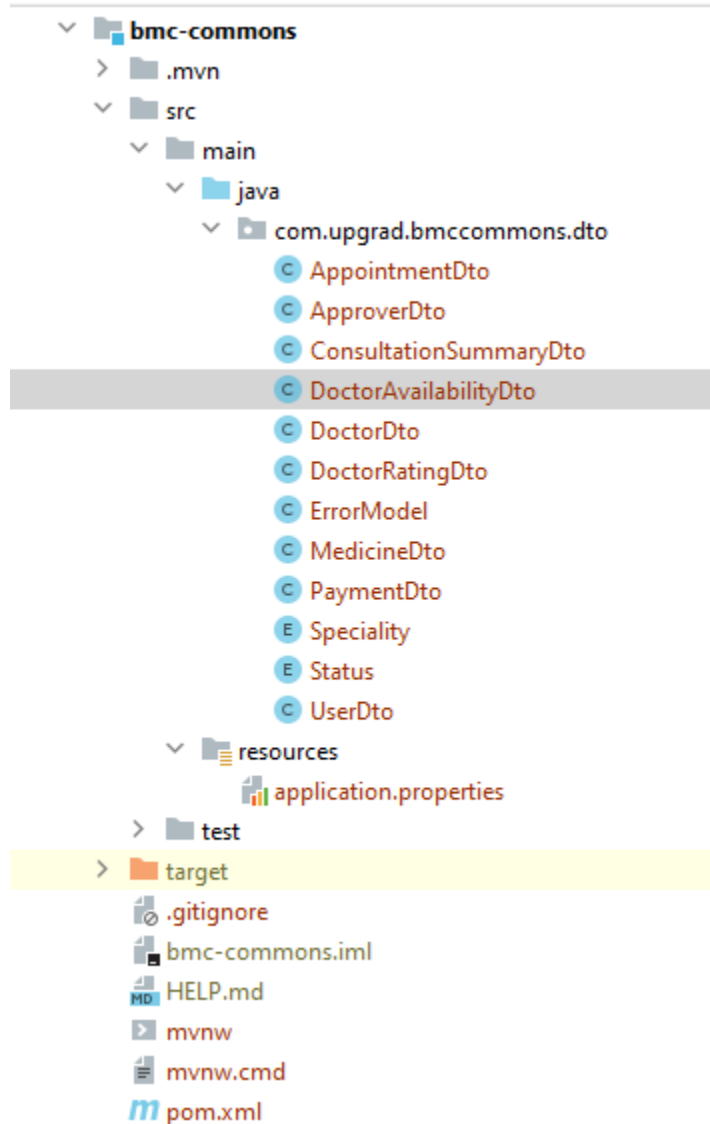
1. [Project Structure](#)
2. [bmc-commons:](#)
3. [doctor-service](#)
4. [docker-compose:](#)
5. [user-service](#)
6. [docker-compose](#)
7. [appointment-service](#)
8. [docker-compose](#)
9. [payment-service](#)
10. [docker-compose](#)
11. [rating-service](#)
12. [docker-compose](#)
13. [Email-notification-service](#)
14. [docker-compose](#)
15. [bmc-gateway](#)
16. [docker-compose](#)
17. [eureka-server](#)
18. [docker-compose](#)
19. [Database](#)
20. [S3 Bucket & SES](#)
21. [Postman screen shots](#)
 - a. [Register doctor](#)
 - b. [Fetch doctor](#)
 - c. [Upload document](#)
 - d. [Approve doctor](#)
 - e. [Reject doctor](#)
 - f. [Get document metadata](#)
 - g. [Download uploaded document](#)
 - h. [Payment-service](#)
 - i. [Rating service](#)
22. [Security](#)
23. [Things to update to run the project](#)

24. Project Structure



A parent project BookMyConsultation consisting for child modules for each of the microservice
The parent module has the docker-compose file

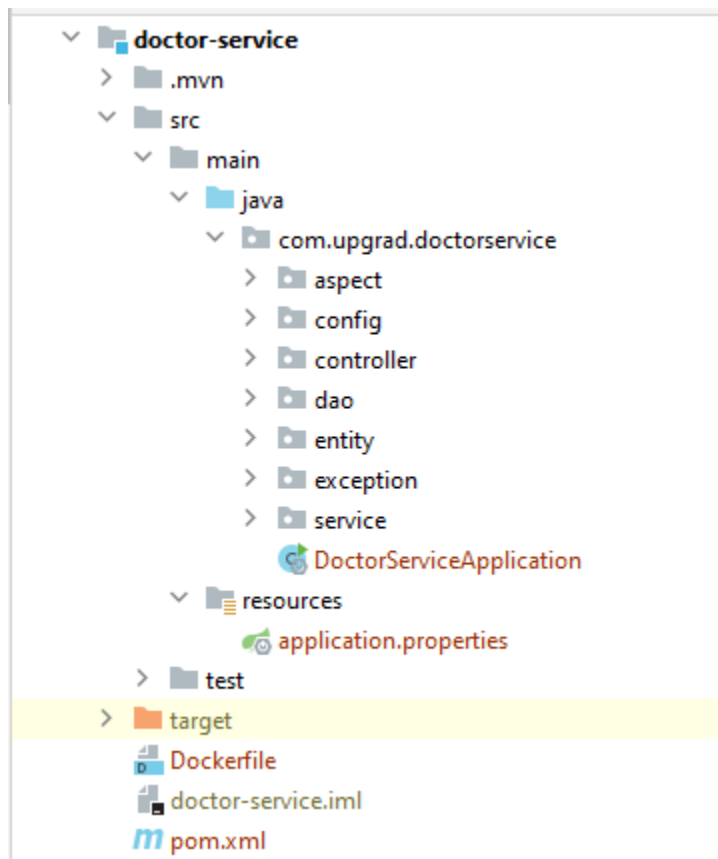
25. bmc-commons:



This module contains all the dto's that are used commonly across the project. By externalizing this we can reduce the duplication and also maintain the same version of dto throughout the project.

This artifact will be used by all the modules to use the dtos

26. doctor-service



This module is a microservice responsible for doctor onboarding and related functions. The service is registered as doctor-service in eureka and hosted on 8080

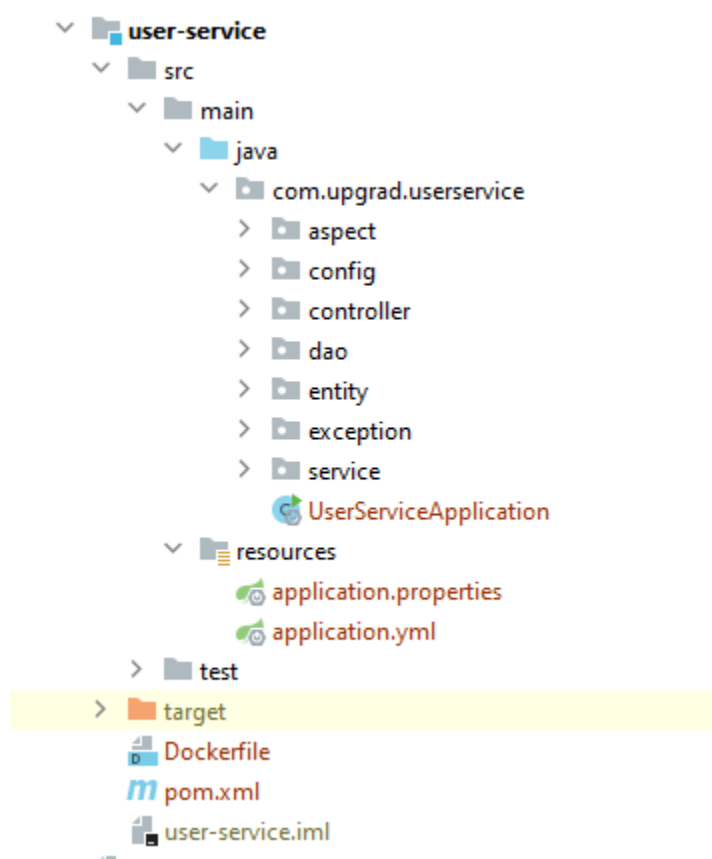
DOCTOR-SERVICE	n/a (1)	(1)	UP (1) - 172.18.0.6:8080
----------------	---------	-----	--------------------------

- docker-compose:

```
doctor-service:
  build: doctor-service
  container_name: doctor-service
  image: bookmyconsultation/doctor-service:1.0.0
  ports:
    - "8080:8080"
  networks:
    - bookmyconsultationnet
  environment:
    EUREKA_HOST_NAME: eureka-server
    MONGODB_HOST_NAME: ec2-44-201-170-28.compute-1.amazonaws.com
    KAFKA_HOST_NAME: kafka
    S3_ACCESS_KEY: AKIAUPFTNDWLKA5S34X4
    S3_SECRET_KEY: oBEPBolmnc0V3JCUd9LAKpHuTzTaJCJK+xaMUVa
```

```
depends_on:
  - eureka-server
  - kafka
```

27. user-service



This module is a microservice responsible for user onboarding and related functions. The service is registered as user-service in eureka and hosted on 8083

USER-SERVICE	n/a (1)	(1)	UP (1) - 172.18.0.7:8083
--------------	---------	-----	--------------------------

- docker-compose

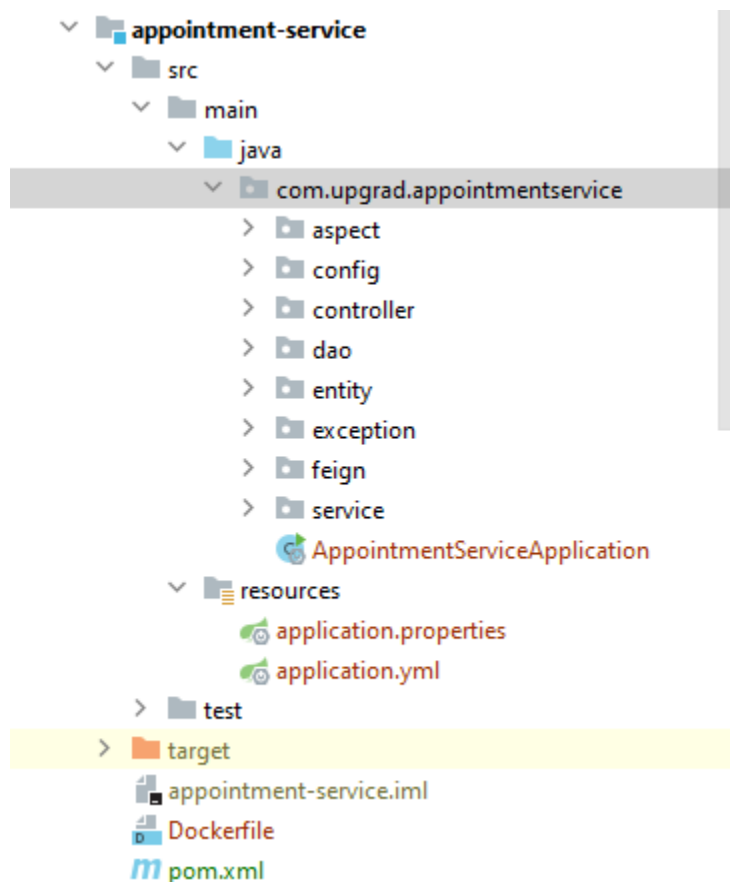
```
user-service:
  build: user-service
  container_name: user-service
  image: bookmyconsultation/user-service:1.0.0
  ports:
    - "8083:8083"
  networks:
    - bookmyconsultationnet
  environment:
```

```

EUREKA_HOST_NAME: eureka-server
MONGODB_HOST_NAME: ec2-44-201-170-28.compute-1.amazonaws.com
KAFKA_HOST_NAME: kafka
S3_ACCESS_KEY: AKIAUPFTNDWLKA5S34X4
S3_SECRET_KEY: oBEPBolmnc0V3JCUd9LAKpHuTzTaJCJK+xaMUVa
depends_on:
  - eureka-server
  - kafka

```

28. appointment-service



This module is a microservice responsible for appointment related functions. The service is registered as appointment-service in eureka and hosted on 8083

APPOINTMENT-SERVICE	n/a (1)	(1)	UP (1) - 172.18.0.10:8082
---------------------	---------	-----	---------------------------

- docker-compose

```

appointment-service:
  build: appointment-service
  container_name: appointment-service

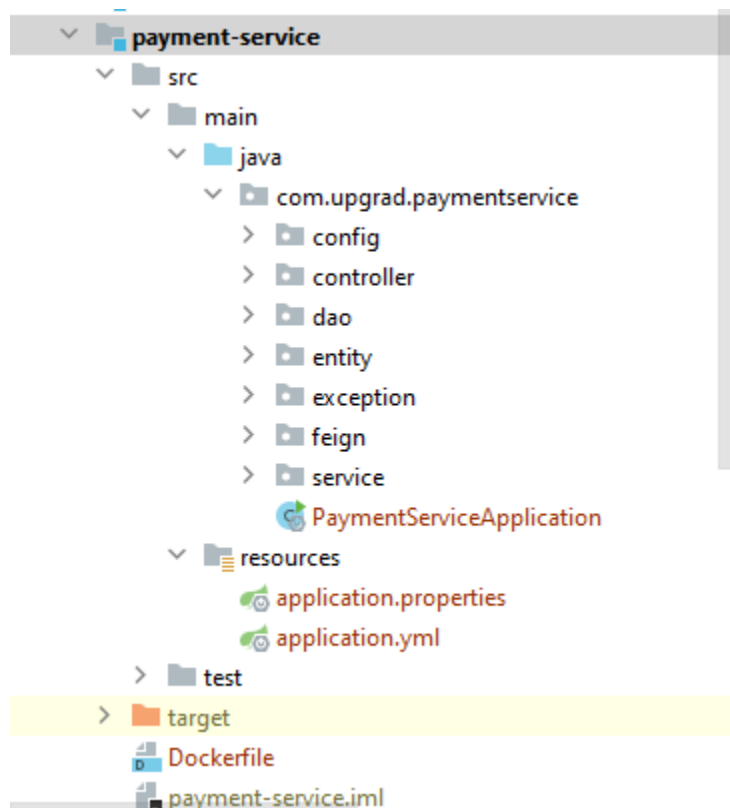
```

```

image: bookmyconsultation/appointment-service:1.0.0
ports:
  - "8082:8082"
networks:
  - bookmyconsultationnet
environment:
  EUREKA_HOST_NAME: eureka-server
  MONGODB_HOST_NAME: ec2-44-201-170-28.compute-1.amazonaws.com
  KAFKA_HOST_NAME: kafka
depends_on:
  - eureka-server
  - user-service
  - doctor-service
  - kafka

```

29. payment-service



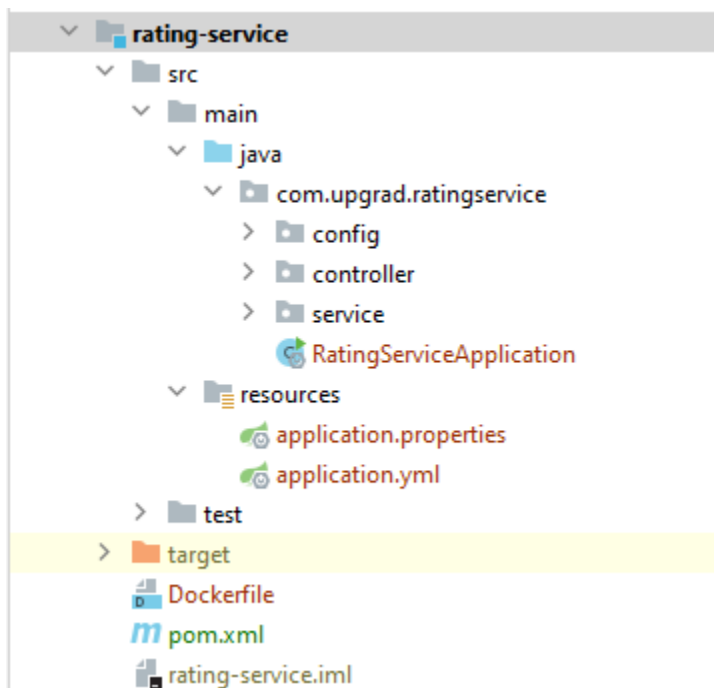
This module is a microservice responsible for appointment related functions. The service is registered as payment-service in eureka and hosted on 8086

PAYMENT-SERVICE	n/a (1)	(1)	UP (1) - 172.18.0.11:8086
-----------------	---------	-----	---------------------------

- docker-compose

```
payment-service:
  build: payment-service
  container_name: payment-service
  image: bookmyconsultation/payment-service:1.0.0
  ports:
    - "8086:8086"
  networks:
    - bookmyconsultationnet
  environment:
    EUREKA_HOST_NAME: eureka-server
    MONGODB_HOST_NAME: ec2-44-201-170-28.compute-1.amazonaws.com
    KAFKA_HOST_NAME: kafka
    APPOINTMENT_SERVICE_HOST_NAME: BMC-GATEWAY
  depends_on:
    - eureka-server
    - appointment-service
    - kafka
```

30. rating-service



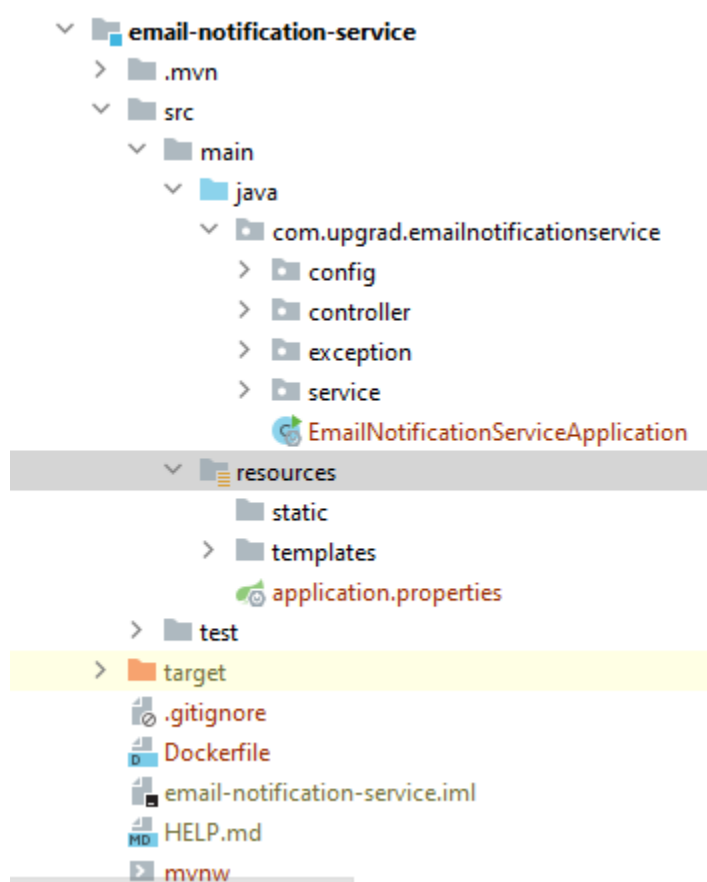
This module is a microservice responsible for rating doctor.

The service is registered as rating-service in eureka and hosted on 8084

- docker-compose

```
rating-service:
build: rating-service
container_name: rating-service
image: bookmyconsultation/rating-service:1.0.0
ports:
  - "8084:8084"
networks:
  - bookmyconsultationnet
environment:
  EUREKA_HOST_NAME: eureka-server
  MONGODB_HOST_NAME: ec2-44-201-170-28.compute-1.amazonaws.com
  KAFKA_HOST_NAME: kafka
depends_on:
  - eureka-server
  - kafka
```

31. Email-notification-service



This module is a microservice responsible for send notification emails using AWS SES

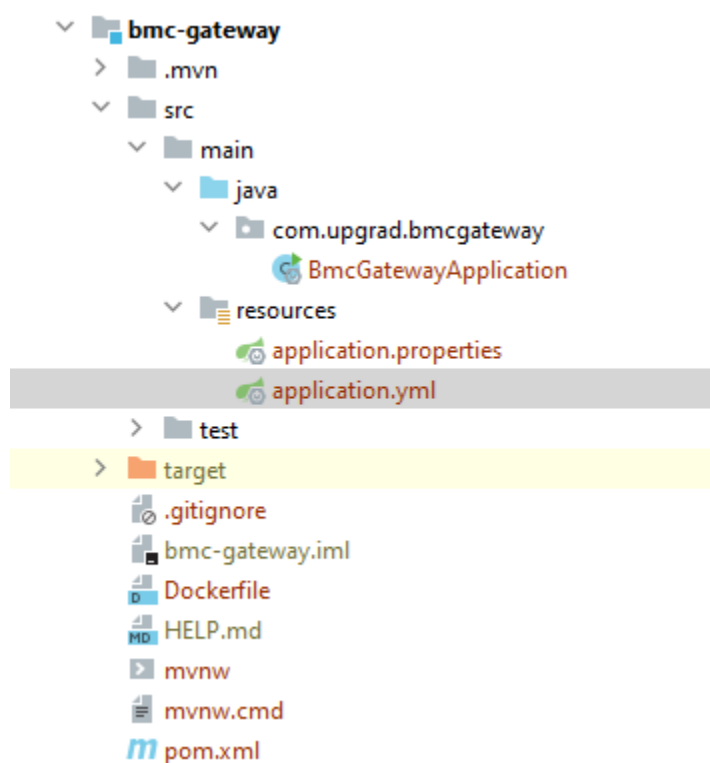
The service is registered as email-notification-service in eureka and hosted on 8087

EMAIL-NOTIFICATION-SERVICE	n/a (1)	(1)	UP (1) - 172.18.0.8:8087
----------------------------	---------	-----	--------------------------

- docker-compose

```
email-notification-service:
  build: email-notification-service
  container_name: email-notification-service
  image: bookmyconsultation/email-notification-service:1.0.0
  ports:
    - "8087:8087"
  networks:
    - bookmyconsultationnet
  environment:
    EUREKA_HOST_NAME: eureka-server
    KAFKA_HOST_NAME: kafka
    AWS_SES_ACCESS_KEY: AKIAUPFTNDWLKA5S34X4
    AWS_SES_SECRET_KEY: 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
    SMTP_AWS_ENDPOINT: email-smtp.us-east-1.amazonaws.com
    SMTP_AWS_USERNAME: AKIAUPFTNDWLKHKTK4NZ
    SMTP_AWS_PASSWORD: BBziyBqXJG/k0pbF1qU543YTihTB2+nWOeb2kn/XIO4b
    SMTP_AWS_FROMEMAIL: sandesh.ayyod89@gmail.com
  depends_on:
    - eureka-server
    - kafka
```

32. bmc-gateway



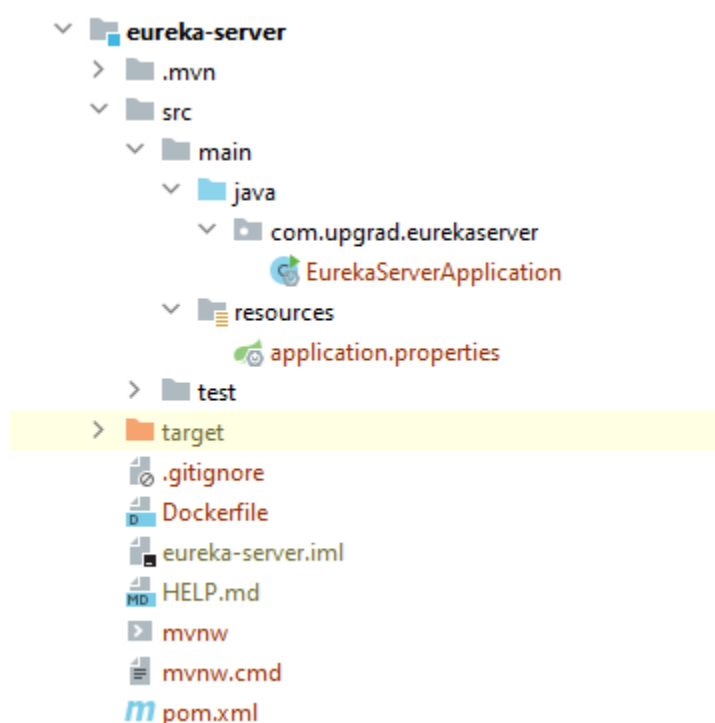
This module is a microservice acts as a gateway for UI to communicate between microservices. The service is registered as bmc-gateway in eureka and hosted on 9191.

BMC-GATEWAY	n/a (1)	(1)	UP (1) - 172.18.0.9:9191
-------------	---------	-----	--------------------------

- docker-compose

```
bmc-gateway:
  build: bmc-gateway
  container_name: bmc-gateway
  image: bookmyconsultation/bmc-gateway:1.0.0
  ports:
    - "9191:9191"
  networks:
    - bookmyconsultationnet
  environment:
    EUREKA_HOST_NAME: eureka-server
  depends_on:
    - eureka-server
    - doctor-service
```

33. eureka-server



This module is the service registry and hosted on port 8761

Application	AMIs	Availability Zones	Status
APPOINTMENT-SERVICE	n/a (1)	(1)	UP (1) - 172.18.0.10:8082
BMC-GATEWAY	n/a (1)	(1)	UP (1) - 172.18.0.9:9191
DOCTOR-SERVICE	n/a (1)	(1)	UP (1) - 172.18.0.6:8080
EMAIL-NOTIFICATION-SERVICE	n/a (1)	(1)	UP (1) - 172.18.0.8:8087
PAYMENT-SERVICE	n/a (1)	(1)	UP (1) - 172.18.0.11:8086
RATING-SERVICE	n/a (1)	(1)	UP (1) - 172.18.0.5:8084
USER-SERVICE	n/a (1)	(1)	UP (1) - 172.18.0.7:8083

<http://localhost:8761/>

- docker-compose

```
eureka-server:
  build: eureka-server
  container_name: eureka-server
  image: bookmyconsultation/eureka-server:1.0.0
  ports:
    - "8761:8761" # Map the exposed port 3000 on container to port 3000
    on the host machine
  networks:
    - bookmyconsultationnet
  environment:
    EUREKA_HOST_NAME: eureka-server
```

34. Database

MongoDB Compass - ec2-44-201-170-28.compute-1.amazonaws.com:27017/bmcDB

Connect View Help

Local

4 DBS COLLECTIONS

☆ FAVORITE

Filter your data

> admin

▼ bmcDB

- appointment
- consultationSummary
- doctor
- doctorAvailability
- doctorRating
- payment
- user

> config

> local

Collections

CREATE COLLECTION

Collection Name	Documents	Avg. Document Size	Total Document Size	Num. Indexes	Total Index Size	Properties
appointment	1	359.0 B	359.0 B	1	36.0 KB	
consultationSummary	1	566.0 B	566.0 B	1	20.0 KB	
doctor	4	403.0 B	1.6 KB	1	36.0 KB	
doctorAvailability	3	242.0 B	726.0 B	1	36.0 KB	
doctorRating	1	142.0 B	142.0 B	1	20.0 KB	
payment	4	160.0 B	640.0 B	1	36.0 KB	
user	2	241.5 B	483.0 B	1	20.0 KB	

MongoDB created on EC2 server.

And the Public IPv4 DNS is updated in the docker-compose file

Instances (1/1) info

Search

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP
-	i-02c2bee691dbdad3f	Running	t2.micro	-	No alarms	us-east-1d	ec2-44-201-170-28.co...	44.201.170.28	-

Instance: i-02c2bee691dbdad3f

Select an instance above

Details Security Networking Storage Status checks Monitoring Tags

Instance summary info

Instance ID i-02c2bee691dbdad3f	Public IPv4 address 44.201.170.28 open address	Private IPv4 addresses 172.31.80.239
IPv6 address -	Instance state Running	Public IPv4 DNS ec2-44-201-170-28.compute-1.amazonaws.com open address
Hostname type IP name: ip-172-31-80-239.ec2.internal	Private IP DNS name (IPv4 only) ip-172-31-80-239.ec2.internal	Answer private resource DNS name IPv4 (A)

`MONGODB_HOST_NAME: ec2-44-201-170-28.compute-1.amazonaws.com`

35. S3 Bucket & SES

Buckets (1) Info					Refresh	Copy ARN	Empty	Delete	Create bucket
Buckets are containers for data stored in S3. Learn more					<input type="text" value="Find buckets by name"/>				
					< 1 > ⚙				
	Name ▲	AWS Region ▼	Access ▼	Creation date ▼					
<input type="radio"/>	bmc-sandesh	US East (N. Virginia) us-east-1	Objects can be public	April 4, 2022, 12:10:32 (UTC+05:30)					

Access Key and Secret key needs to be generated with IAM and they need to be updated in the docker file

A new IAM user created & Permissions given for s3 and ses

Users (2) Info							Refresh	Delete	Add users
An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.							<input type="text" value="Find users by username or access key"/>		
							< 1 > ⚙		
<input type="checkbox"/>	User name ▼	Groups ▼	Last activity ▼	MFA ▼	Password age ▼	Active key age ▼			
<input type="checkbox"/>	ses-smtp-user.20220403-105008	None	13 hours ago	None	None	2 days ago			

Permissions

Groups

Tags (1)

Security credentials

Access Advisor

▼ Permissions policies (2 policies applied)

Add permissions

Add inline policy

Policy name ▼	Policy type ▼	
Attached directly		
▶  AmazonS3FullAccess	AWS managed policy	✕
▶  AmazonSESEFullAccess	AWS managed policy	✕

Permissions	Groups	Tags (1)	Security credentials	Access Advisor
Sign-in credentials				
Summary • User does not have console management access				
Console password Disabled Manage				
Assigned MFA device Not assigned Manage				
Signing certificates None 📄				

Access keys

Use access keys to make programmatic calls to AWS from the AWS CLI, Tools for PowerShell, AWS SDKs, or direct AWS API calls. You can have a maximum of two access keys (active or inactive) at a time.

For your protection, you should never share your secret keys with anyone. As a best practice, we recommend frequent key rotation.

If you lose or forget your secret key, you cannot retrieve it. Instead, create a new access key and make the old key inactive. [Learn more](#)

Create access key				
Access key ID	Created	Last used	Status	
AKIAUPFTNDWLK4HTK4NZ	2022-04-03 10:51 UTC+0530	2022-04-05 08:19 UTC+0530 with ses-smtp in us-east-1	Active Make inactive	✕
AKIAUPFTNDWLKA5S34X4	2022-04-04 11:47 UTC+0530	2022-04-05 20:46 UTC+0530 with s3 in us-east-1	Active Make inactive	✕

SSH keys for AWS CodeCommit

Use SSH public keys to authenticate access to AWS CodeCommit repositories. [Learn more](#)

For doctor-service and user-service in docker-compose update the env variables

S3_ACCESS_KEY: AKIAUPFTNDWLKA5S34X4

S3_SECRET_KEY: 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

For email-notification service

SMTP_AWS_ENDPOINT: email-smtp.us-east-1.amazonaws.com

SMTP_AWS_USERNAME: AKIAUPFTNDWLKHKTK4NZ

SMTP_AWS_PASSWORD: BBziyBqXJG/k0pbF1qU543YTihTB2+nWOeb2kn/XIO4b

SMTP_AWS_FROMEMAIL: sandesh.ayyod89@gmail.com

36. Postman screen shots

- Register doctor

Doctor-service persists doctor into mongoDB

Publishes the dto onto kafka

Email notification listens to kafka and sends verification email to the email id

The screenshot shows a Postman interface for a POST request to `http://localhost:8080/doctors`. The request body is a JSON object representing a doctor's details. The response is a 200 OK status with a JSON object containing the doctor's ID, registration date, and status.

Request:

```
POST http://localhost:8080/doctors
{
  "firstName": "Shanthala",
  "lastName": "K",
  "dob": "1991-04-18",
  "emailId": "sandesh.ayyod89@gmail.com",
  "mobile": "8105648796",
  "pan": "AZIPA2000B",
  "speciality": "GYNECOLOGIST"
}
```

Response:

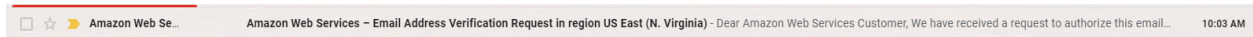
```
200 OK
{
  "id": "4d0b812d-5fc9-4846-86f8-ccc00f94bdeb",
  "firstName": "Shanthala",
  "lastName": "K",
  "mobile": "8105648796",
  "dob": "1991-04-18",
  "emailId": "sandesh.ayyod89@gmail.com",
  "pan": "AZIPA2000B",
  "speciality": "GYNECOLOGIST",
  "registrationDate": "2022-04-06",
  "status": "PENDING",
  "approvedBy": null,
  "approverComments": null,
  "verificationDate": null
}
```

Log:

```
email-notification-service | 2022-04-06 04:33:58.354 INFO 1 --- [container#1-0-C-1] c.u.e.service.DoctorConsumerService : Listening to Doctor@topic-4d0b812d-5fc9-4846-86f8-ccc00f94bdeb, firstName=Shanthala, lastName=K, mobile=8105648796, dob=1991-04-18, emailId=sandesh.ayyod89@gmail.com, pan=AZIPA2000B, speciality=GYNECOLOGIST, registrationDate=2022-04-06, status=PENDING, approvedBy=null, approverComments=null, verificationDate=null
email-notification-service | 2022-04-06 04:33:58.357 INFO 1 --- [container#1-0-C-1] c.u.e.service.DoctorConsumerService : Sending verification email to sandesh.ayyod89@gmail.com
```

```
_id: "4d0b812d-5fc9-4846-86f8-ccc00f94bdeb"  
firstName: "Shanthala"  
lastName: "K"  
mobile: "8105648796"  
dob: "1991-04-18"  
emailId: "sandesh.ayyod89@gmail.com"  
pan: "AZIPA20008"  
speciality: "GYNECOLOGIST"  
registrationDate: 2022-04-06T00:00:00.000+00:00  
status: "PENDING"  
_class: "com.upgrad.doctorservice.entity.Doctor"
```

Email received



- Fetch doctor

GET fetch

BookMyConsultation / DoctorOnboarding / fetch

GET http://localhost:8080/doctors/4d0b812d-5fc9-4846-86f8-ccc00f94bdeb

Params Authorization Headers (6) Body Pre-request Script Tests Settings

☒ none ☐ form-data ☐ x-www-form-urlencoded ☐ raw ☐ binary ☐ GraphQL

This request does not have a body

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

```
1  {
2    "id": "4d0b812d-5fc9-4846-86f8-ccc00f94bdeb",
3    "firstName": "Shanthala",
4    "lastName": "K",
5    "mobile": "8105648796",
6    "dob": "1991-04-18",
7    "emailId": "sandesh.ayyod89@gmail.com",
8    "pan": "AZIPA2000B",
9    "speciality": "GYNECOLOGIST",
10   "registrationDate": "2022-04-06",
11   "status": "PENDING",
12   "approvedBy": null,
13   "approverComments": null,
14   "verificationDate": null
15 }
```

- Upload document

POST upload doctor docum

BookMyConsultation / DoctorOnboarding / upload doctor document

Save

Send

POST

http://localhost:8080/doctors/4d0b812d-5fc9-4846-86f8-ccc00f94bdeb/document

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

Cookies

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

	KEY	VALUE	DESCRIPTION		Bulk Edit
<input checked="" type="checkbox"/>	files	doc_certificate.pdf			
	Key	Value	Description		

Body

Cookies

Headers (5)

Test Results

Status: 200 OK

Time: 5.46 s

Size: 193 B

Save Response

Pretty

Raw

Preview

Visualize

Text

1

File(s) uploaded Successfully

Amazon S3 > Buckets > bmc-sandesh > 4d0b812d-5fc9-4846-86f8-ccc00f94bdeb/

4d0b812d-5fc9-4846-86f8-ccc00f94bdeb/

Copy S3 URI

Objects

Properties

Objects (1)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

Copy S3 URI

Copy URL

Download

Open

Delete

Actions

Create folder

Upload

Find objects by prefix

	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	doc_certificate.pdf	pdf	April 6, 2022, 10:13:48 (UTC+05:30)	80.5 KB	Standard

- Approve doctor

PUT approve doctor

BookMyConsultation / DoctorOnboarding / approve doctor

PUT

http://localhost:8080/doctors/4d0b812d-5fc9-4846-86f8-ccc00f94bdeb/approve

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

```
1 {
2   ... "approvedBy": "sandesh",
3   ... "approverComments": "Approved doctor"
4 }
```

Body

Cookies

Headers (5)

Test Results

Pretty

Raw

Preview

Visualize

JSON

```
1 {
2   "id": "4d0b812d-5fc9-4846-86f8-ccc00f94bdeb",
3   "firstName": "Shanthala",
4   "lastName": "K",
5   "mobile": "8105648796",
6   "dob": "1991-04-18",
7   "emailId": "sandesh.ayyod89@gmail.com",
8   "pan": "AZIPA2000B",
9   "speciality": "GYNECOLOGIST",
10  "registrationDate": "2022-04-06",
11  "status": "ACTIVE",
12  "approvedBy": "sandesh",
13  "approverComments": "Approved doctor",
14  "verificationDate": "2022-04-06"
15 }
```

Email notification received

me

Registration Approved - Hi Dr sandesh.ayyod89@gmail.com Congratulations !!! Your Registration is Approved in BMC app Regards, BMC

- Reject doctor

The screenshot displays a REST client interface with the following details:

- Request Method:** PUT
- URL:** `http://localhost:8080/doctors/d0bc9c03-9d07-4688-b8f0-edb4fe47d9d5/reject`
- Body:** A JSON object with the following structure:

```
{  "approvedBy": "sandesh",  "approverComments": "rejected doctor"}
```
- Response:** A JSON object with the following structure:

```
{  "id": "d0bc9c03-9d07-4688-b8f0-edb4fe47d9d5",  "firstName": "Suman",  "lastName": "K",  "mobile": "8105648796",  "dob": "1991-04-18",  "emailId": "sandesh.ayyod89@gmail.com",  "pan": "AZIPA2000B",  "speciality": "DENTIST",  "registrationDate": "2022-04-06",  "status": "REJECTED",  "approvedBy": "sandesh",  "approverComments": "rejected doctor",  "verificationDate": "2022-04-06"}
```
- Status:** Registration Rejected - Hi Dr sandesh.ayyod89@gmail.com Unfortunately your Registration is Rejected. Please check with BMC customer-support Regards, BMC

- Get document metadata

GET doctor Documents Met

+ ...

BookMyConsultation / DoctorOnboarding / doctor Documents Metadata

GET

http://localhost:8080/doctors/4d0b812d-5fc9-4846-86f8-ccc00f94bdeb/documents/metadata

ParamsAuthorizationHeaders (7)BodyPre-request ScriptTestsSettings

Headers

6 hidden

	KEY	VALUE
<input checked="" type="checkbox"/>	Authorization	{{Authorization}}
	Key	Value

BodyCookiesHeaders (5)Test Results

PrettyRawPreviewVisualizeJSON

1

2

3

"4d0b812d-5fc9-4846-86f8-ccc00f94bdeb/doc_certificate.pdf"

- Download uploaded document

GET get doctor Document

+ ...

No Environment

BookMyConsultation / DoctorOnboarding / get doctor Document

GET

http://localhost:8080/doctors/4d0b812d-5fc9-4846-86f8-ccc00f94bdeb/documents/doc_certificate.pdf

Send

ParamsAuthorizationHeaders (6)BodyPre-request ScriptTestsSettingsCookies

Query Params

	KEY	VALUE	DESCRIPTION		Bulk Edit
	Key	Value	Description		

BodyCookiesHeaders (6)Test Results

Status: 200 OKTime: 1739 msSize: 80.77 KBSave Response

Git Cheat Sheet

01 Git configuration

\$ git config --global user.name "Your Name"

Set the name that will be attached to your commits and tags.

\$ git config --global user.email "you@example.com"

Set the e-mail address that will be attached to your commits and tags.

03 Day-To-Day Work

\$ git status

Displays the status of your working directory. Options include new, staged, and modified files. It will retrieve branch name, current commit identifier, and changes pending commit.

\$ git add [file]

- User Registration

BookMyConsultation / UserOnboarding / enroll user

POST ▼ http://localhost:8083/users

Params Authorization Headers (8) **Body** ● Pre-request Script Tests Settings

● none ● form-data ● x-www-form-urlencoded ● raw ● binary ● GraphQL **JSON** ▼

```
1 {
2   ... "firstName": "Avish",
3   ... "lastName": "K",
4   ... "dob": "1956-07-01",
5   ... "emailId": "sandesh.ayyod89@gmail.com",
6   ... "mobile": "8105232606"
7 }
```

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON ▼

```
1 {
2   "id": "644c84c3-e8a7-4ba2-887b-490b6603c228",
3   "firstName": "Avish",
4   "lastName": "K",
5   "dob": "1956-07-01",
6   "emailId": "sandesh.ayyod89@gmail.com",
7   "mobile": "8105232606",
8   "createdDate": "2022-04-06"
9 }
```

me **Welcome User** - Hi sandesh.ayyod89@gmail.com Congratulations !!! You are registered as a user in BMC app Regards, BMC

- User retrieval

The screenshot shows a REST client interface with a single tab titled "GET fetch user". The breadcrumb path is "BookMyConsultation / UserOnboarding / fetch user". The request method is "GET" and the URL is "http://localhost:8083/users/644c84c3-e8a7-4ba2-887b-490b6603c228".

The "Params" tab is selected, showing "Query Params" with a table:

KEY	VALUE
Key	Value

The "Body" tab is selected, showing the response in JSON format:

```
1 {
2   "id": "644c84c3-e8a7-4ba2-887b-490b6603c228",
3   "firstName": "Avish",
4   "lastName": "K",
5   "dob": "1956-07-01",
6   "emailId": "sandesh.ayyod89@gmail.com",
7   "mobile": "8105232606",
8   "createdDate": "2022-04-06"
9 }
```

- User document upload

POST upload user document

BookMyConsultation / UserOnboarding / upload user document

POST http://localhost:8083/users/644c84c3-e8a7-4ba2-887b-490b6603c228/document

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

none form-data x-www-form-urlencoded raw binary GraphQL

KEY	VALUE	D
<input checked="" type="checkbox"/> files	doc_certificate.pdf ×	
Key	Value	D

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize Text

1 File(s) uploaded Successfully

- User document get names

GET get User Documents n. + ...

BookMyConsultation / UserOnboarding / get User Documents name

GET http://localhost:8083/users/644c84c3-e8a7-4ba2-887b-490b6603c228/documents/metadata

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

```
1 [
2   "644c84c3-e8a7-4ba2-887b-490b6603c228/doc_certificate.pdf"
3 ]
```

- User document download

GET get user document + ... No Env

BookMyConsultation / UserOnboarding / get user document Save

GET http://localhost:8083/users/644c84c3-e8a7-4ba2-887b-490b6603c228/documents/doc_certificate.pdf

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body Cookies Headers (6) Test Results Status: 200 OK Time: 1735 ms Size: 80.7

Git Cheat Sheet

01 Git configuration

```
$ git config --global user.name "Your Name"
```

Set the name that will be attached to your commits and tags.

03 Day-To-Day Work

```
$ git status
```

Displays the status of your working directory. Options include new, staged, and modified files. It will also show branch name, current commit.

- Update doctor availability

POST Post Doctor Availabili

No Environment

BookMyConsultation / Appointments / Post Doctor Availability

POST

http://localhost:8082/doctor/4d0b812d-5fc9-4846-86f8-ccc00f94bdeb/availability

Send

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

1

2

3

4

5

6

7

1

2

3

4

5

6

7

Body

Cookies

Headers (5)

Test Results

Pretty

Raw

Preview

Visualize

Text

1

Status: 200 OK

Time: 9.12 s

Size: 163 B

Save Response

- Get doctor availability

GET Get Doctor Availability

BookMyConsultation / Appointments / Get Doctor Availability

GET

http://localhost:8082/doctor/4d0b812d-5fc9-4846-86f8-ccc00f94bdeb/availability

Params

Authorization

Headers (6)

Body

Pre-request Script

Tests

Settings

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body

Cookies

Headers (5)

Test Results

Pretty

Raw

Preview

Visualize

JSON

1

2

3

4

5

6

7

8

9

10

11

12

13

1

2

3

4

5

6

7

8

9

10

11

12

13

Status: 200 OK

Time: 9.12 s

Size: 163 B

Save Response

- Book appointment

The screenshot displays a REST client interface for a POST request to `http://localhost:8082/appointments`. The request body is a JSON object with the following fields:

```
{  "doctorId": "4d0b812d-5fc9-4846-86f8-ccc00f94bdeb",  "userId": "1d12a4ae-be2a-457a-9fd6-8940833cd639",  "appointmentDate": "2021-03-02",  "timeSlot": "11PM-12PM"}
```

The response status is 200 OK, with a time of 1693 ms and a size of 200 B. The response body is a single UUID: `31ef0cfd-b2da-4017-9003-a4ac7f103cb4`.

Below the REST client, a browser notification is visible with the text: "Appointment Confirmed - Hi sandesh.ayyod89@gmail.com Your appointment is confirmed. Please check your appointments for details Regards, BMC".

- Get appointment detail

GET get appointment

+

...

BookMyConsultation / Appointments / get appointment

GET

▼

http://localhost:8082/appointments/31ef8cfd-b2da-4817-8083-a4ac7f183cb4

ParamsAuthorizationHeaders (6)BodyPre-request ScriptTestsSettings

Query Params

KEY	VALUE
Key	Value

BodyCookiesHeaders (5)Test Results

PrettyRawPreviewVisualizeJSON▼

```
1 {
2   "appointmentId": "31ef8cfd-b2da-4817-8083-a4ac7f183cb4",
3   "doctorId": "4d0b812d-5fc9-4846-86f8-ccc00f94bdeb",
4   "userId": "1d12a4ae-be2a-457a-9fd6-8940833cd639",
5   "userEmailId": "sandesh.ayyod89@gmail.com",
6   "appointmentDate": "2021-03-02",
7   "timeSlot": "11PM-12PM"
8 }
```

- Get all the user appointments

The screenshot shows a REST client interface with a single tab titled "GET get user appointments". The breadcrumb path is "BookMyConsultation / Appointments / get user appointments". The request method is "GET" and the URL is "http://localhost:8082/appointments/1d12a4ae-be2a-457a-9fd6-8940833cd639/appointments".

The "Params" tab is selected, showing "Query Params" with a table:

KEY	VALUE	DE
Key	Value	De

The "Body" tab is selected, showing the response in "Pretty" JSON format:

```
1 {
2   "appointmentId": "31ef8cfd-b2da-4817-8083-a4ac7f183cb4",
3   "doctorId": "4d0b812d-5fc9-4846-86f8-ccc0f94bdeb",
4   "userId": "1d12a4ae-be2a-457a-9fd6-8940833cd639",
5   "userEmailId": "sandesh.ayyod89@gmail.com",
6   "appointmentDate": "2021-03-02",
7   "timeSlot": "11PM-12PM"
8 }
9
10
```

- Save prescriptions

POST save prescriptions + ...

BookMyConsultation / Appointments / save prescriptions

POST http://localhost:8082/prescriptions

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

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   ... "appointmentId": "31ef8cfd-b2da-4817-8083-a4ac7f183cb4",
3   ... "doctorId": "4d0b812d-5fc9-4846-86f8-ccc00f94bdeb",
4   ... "userId": "1d12a4ae-be2a-457a-9fd6-8940833cd639",
5   ... "diagnosis": "head-ache",
6   ... "medicineList": [
7     {
8       ... "name": "Calpol",
9       ... "type": "Tablet",
10      ... "dosage": "1 week",
11      ... "duration": "1 week",
12      ... "frequency": "3 times a day",
13      ... "remarks": "after food"
14    },
15    {
16      ... "name": "PainKill",
17      ... "type": "Syrup",
18      ... "dosage": "1 week",
19      ... "duration": "1 week",
20      ... "frequency": "3 times a day",
```

Body Cookies Headers (4) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "errorCode": "NOT_WORKING",
3   "errorMessage": "Service not working. Please try after sometime",
4   "errorFields": null
5 }
```

Exception because pending payment:

```
2022-04-06 05:06:16.193 INFO [appointment-service,321e40b1a99151cd,321e40b1a99151cd] 1 --- [nio-8082-exec-5] c.u.a.s.impl.ConsultationServiceImpl : Unloading consultation summary for Appointment31ef8cfd-b2da-4817-8083-a4ac7f183cb4
2022-04-06 05:06:16.193 ERROR [appointment-service,321e40b1a99151cd,321e40b1a99151cd] 1 --- [nio-8082-exec-5] c.u.a.s.impl.ConsultationServiceImpl : Payment pending for 31ef8cfd-b2da-4817-8083-a4ac7f183cb4
2022-04-06 05:06:16.197 INFO [appointment-service,321e40b1a99151cd,321e40b1a99151cd] 1 --- [nio-8082-exec-5] c.u.a.aspect.LoggingAspect : In GlobalControllerAdvice, entering handleServiceException at time: Wed Apr 06 05:06:16 GMT 2022
2022-04-06 05:06:16.213 INFO [appointment-service,321e40b1a99151cd,321e40b1a99151cd] 1 --- [nio-8082-exec-5] c.u.a.aspect.LoggingAspect : In GlobalControllerAdvice, exiting handleServiceException at time: Wed Apr 06 05:06:16 GMT 2022, total time taken: 16ms
```

- Payment-service

The screenshot shows a REST client interface with a POST request to `http://localhost:8086/payments?appointmentId=31ef8cfd-b2da-4817-8083-a4ac7f183cb4`. The response status is 200 OK. The response body is empty, indicated by the text "This request does not have a body".

Payment service will take the payment and generate a payment id. And sends the payment dto to kafka.

Appointment service will update the status of appointment to payment confirmed


The screenshot shows a log viewer with logs from various services. The logs include:


- `appointment-service`: `2022-04-06 05:10:49:642 INFO [appointment-service, B1864a60b4e0c5f, 0aa802434f19f8c5] 1 --- [container#0-D-C-1] c.u.a.s.impl.PaymentConsumerService : Listening to PaymentDto(paymentId=d41cf638-38d4-45f5-8fa2-772457e2ad3d, appointmentId=31ef8cfd-b2da-4817-8083-a4ac7f183cb4, createDate=2022-04-06T05:10:45.235777)`
- `appointment-service`: `2022-04-06 05:10:49:872 INFO [appointment-service, B1864a60b4e0c5f, 0aa802434f19f8c5] 1 --- [container#0-D-C-1] c.u.a.s.impl.PaymentConsumerService : Updating the PaymentStatus to CONFIRMED for appointmentId:31ef8cfd-b2da-4817-8083-a4ac7f183cb4`
- `kafka-server`: `2022-04-06 05:10:49:110 INFO 1 --- [a-ExclusionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 0ms`
- `eureka-server`: `2022-04-06 05:10:49:120 INFO 1 --- [a-ExclusionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 0ms`
- `istc-gateway`: `2022-04-06 05:10:03:487 INFO 1 --- [trap-executor-0] c.n.d.s.r.aws.ConfigClusterResolver : Resolving eureka endpoints via configuration`








The screenshot shows a JSON object representing an appointment entity:

```
{  "_id": "624d20629c123e452bebe69e",  "appointmentId": "31ef8cfd-b2da-4817-8083-a4ac7f183cb4",  "doctorId": "4d0b812d-5fc9-4846-86f8-ccc0f94bdeb",  "userId": "1d12a4ae-be2a-457a-9fd6-8940833cd639",  "userEmailId": "sandesh.ayyod89@gmail.com",  "appointmentDate": "2021-03-02T00:00:00.000+00:00",  "timeSlot": "11PM-12PM",  "paymentStatus": "CONFIRMED",  "_class": "com.upgrad.appointmentservice.entity.Appointment"}
```

- Retry consultation summary after payment



POST  http://localhost:8082/prescriptions

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

 none  form-data  x-www-form-urlencoded  **raw**  binary  GraphQL **JSON** 

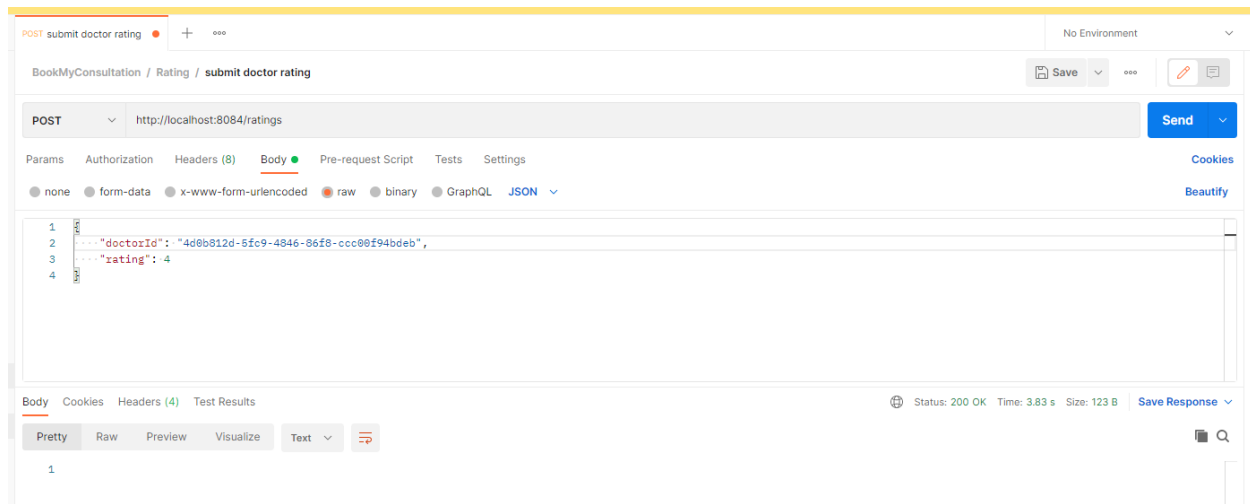
```
1  {
2    .... "appointmentId": "31ef8cfd-b2da-4817-8083-a4ac7f183cb4",
3    .... "doctorId": "4d0b812d-5fc9-4846-86f8-ccc00f94bdeb",
4    .... "userId": "1d12a4ae-be2a-457a-9fd6-8940833cd639",
5    .... "diagnosis": "head-ache",
6    .... "medicineList": [
7      .... {
8        .... "name": "Calpol",
9        .... "type": "Tablet",
10       .... "dosage": "1 week",
11       .... "duration": "1 week",
12       .... "frequency": "3 times a day",
13       .... "remarks": "after food"
14     },
15     {
16       .... "name": "PainKill",
17       .... "type": "Syrup",
18       .... "dosage": "1 week",
19       .... "duration": "1 week",
20       .... "frequency": "3 times a day",
```

Body **Cookies** Headers (5) Test Results

Pretty Raw Preview Visualize Text  

1 Prescriptions Uploaded

- Rating service



Rating service will send the rating dto to kafka, doctor-service will calculate the average and update in db

```
_id: ObjectId("624bd67821616e032bb4886c")
doctorId: "961177f3-7670-4cf1-8e6a-603d23511d5c"
rating: 4
_class: "com.upgrad.doctorservice.entity.DoctorRating"
```

```
_id: ObjectId("624d21540a3a76481d7a713e")
doctorId: "4d0b812d-5fc9-4846-86f8-ccc00f94bdeb"
rating: 4
_class: "com.upgrad.doctorservice.entity.DoctorRating"
```

```
2022-04-06 05:12:59.155 INFO [doctor-service,993b0fdcd1d19051,b6b85e30a5c1884] 1 --- [ntainer#0-0-C-1] c.u.d.service.impl.ConsumerService : Received doctor rating from kafka for DoctorRatingId:4d0b812d-5fc9-4846-86f8-ccc00f94bdeb, rating=4
2022-04-06 05:12:59.155 INFO [doctor-service,993b0fdcd1d19051,b6b85e30a5c1884] 1 --- [ntainer#0-0-C-1] c.u.d.service.impl.ConsumerService : Successfully updated doctor rating for DoctorId: 4d0b812d-5fc9-4846-86f8-ccc00f94bdeb
2022-04-06 05:12:59.121 INFO 1 --- [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 0ms
2022-04-06 05:12:59.121 INFO 1 --- [a-EvictionTimer] c.n.e.registry.AbstractInstanceRegistry : Running the evict task with compensationTime 0ms
```

37. Security

Security has not been implemented. This needs OAuth2 and is not covered as part of the course

38. Things to update to run the project

In docker compose file update the following

`S3_ACCESS_KEY`: <<update iam access key for s3>>

`S3_SECRET_KEY`: <<update iam secret key for s3>>

`MONGODB_HOST_NAME`: <<update ec3 instance public ipv4 >>

`SMTP_AWS_ENDPOINT`: email-smtp.us-east-1.amazonaws.com

SMTP_AWS_USERNAME: <<update smtp aws user name generated >>
SMTP_AWS_PASSWORD: <<update smtp aws password generated >>
SMTP_AWS_FROMEMAIL: <<update verified email id in ses >>