My Movie Kart

an online movie ticket booking platform

Problem statement (Version 1.3.3)

- Online movie ticket booking platform for:
  - B2B (theatre partners)
  - B2C (end customers) clients
- Owner – XYZ

Goals

- Enable theatre partners to onboard their theatres
- Enable end customers to browse the platform to get access to movies across different cities, languages, and genres, as well as book tickets

Tech Stack

- Java 21
- Spring-Boot 3.5.6
- Maven 3.3.4
- Mapstruct 1.6.3
- Lombok 1.18.32
- Springdoc-openapi 2.6.0
- Kafka 3.7.0
- Ollama llama3
- IDE - IntelliJ

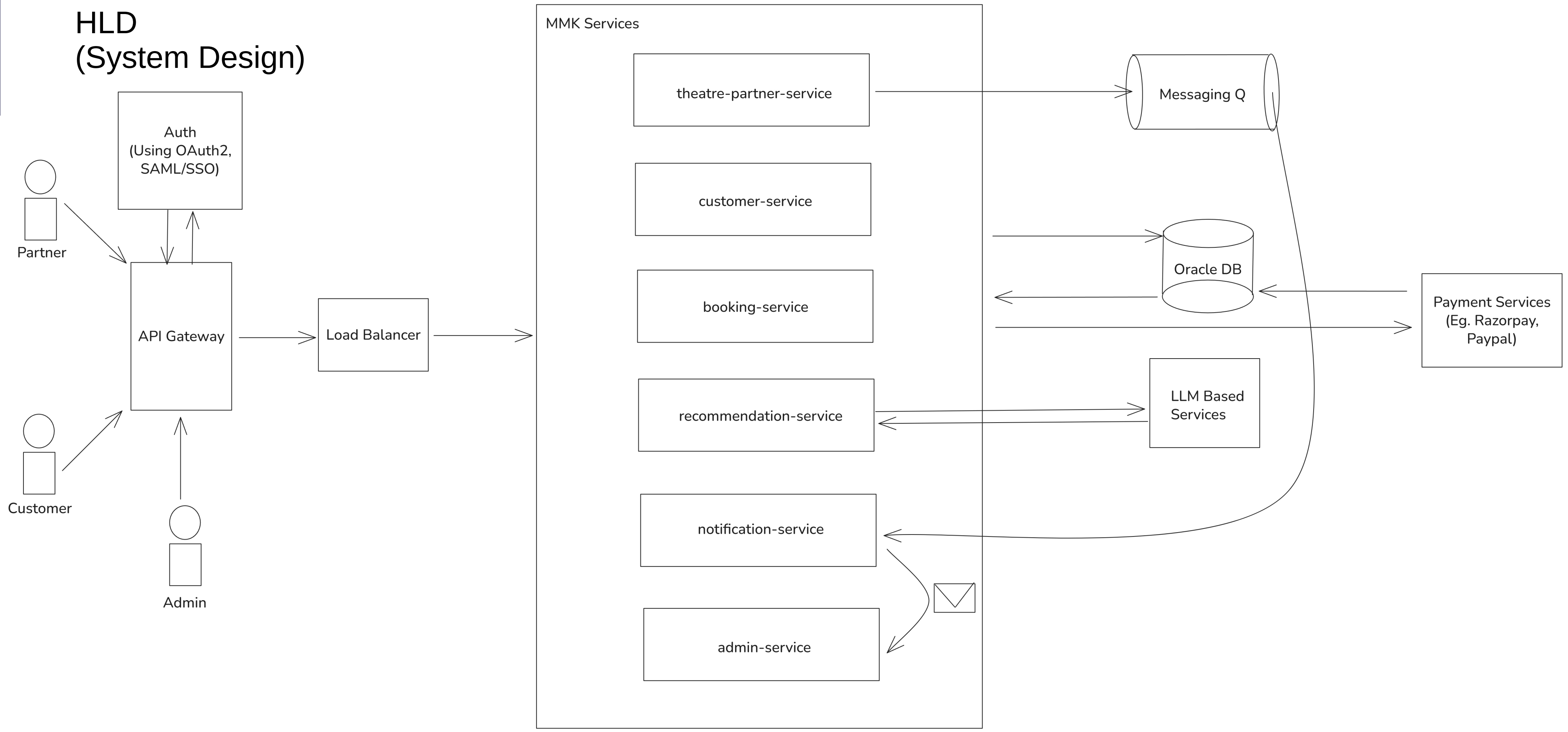
Functional Requirements

- Platform user as a partner, should be able to create and onboard entities like theatres with respective auditoriums, seat layout and playing movies
- Platform user as a customer, should be able to browse through the approved list of theatres based on geo parameters like state and city to find the movies and select seats, as well as vice-versa selecting movies and corresponding hosting theatres
- Platform user as an admin, should be able to approve or reject the theatre onboarding request, raised by the partners
- Application should be multi-tenancy based, capable of serving partners, customers and admins together
- Movie details need to be onboarded by the partners with respective show-timings and seat layouts (which would differ in prices)
- Both partners and customers can access the application with onboarded details like name, location, KYC, etc.

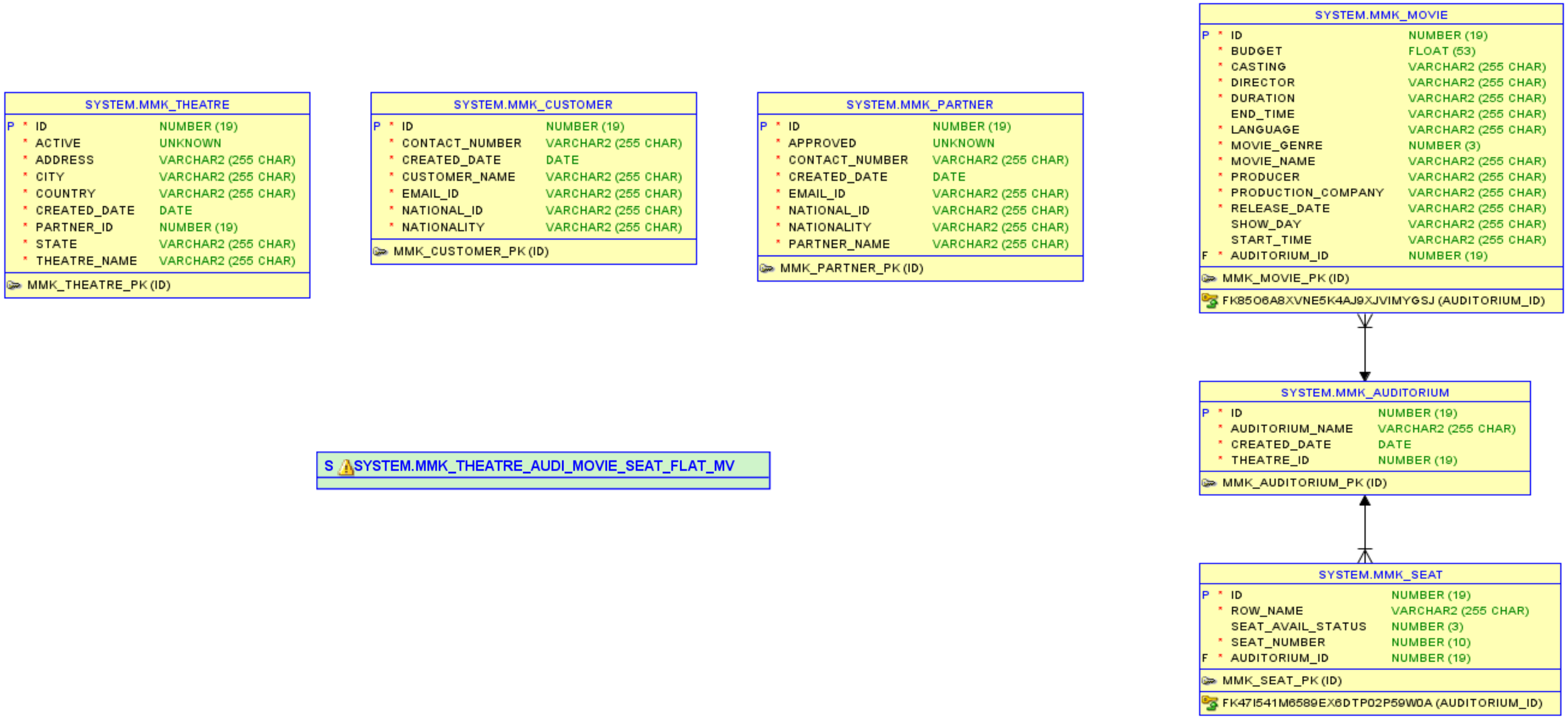
Non Functional Requirements

- Application should have high-availability and quick api response time
- APIs should leverage security approaches like spring-security and Oauth
- Multi-module microservice architecture should help in better code management and easier deployment
- For async. communication across microservices, like sending notifications, should use messaging queues like Kafka, RabbitMQ, etc.
- Project should be open for improvment scopes like scalability, availability, CI/CD, resilience, fault-tolerance, auto-recovery
- Database should support multiple reads and writes without impacting performance or entering into locks
- Database should be open for improvement scopes like partitioning, sharding and replication
- For front-end queries, database should provide periodic and frequent snapshots for faster queries, like materialized views or adding an document database layer on top of the underlying RDBMS tables
- For faster application response, caching mechanism should be used, like Redis

HLD
(System Design)



DB Data Modeling



SYSTEM.MMK_THEATRE

P	*	ID	NUMBER (19)
	*	ACTIVE	UNKNOWN
	*	ADDRESS	VARCHAR2 (255 CHAR)
	*	CITY	VARCHAR2 (255 CHAR)
	*	COUNTRY	VARCHAR2 (255 CHAR)
	*	CREATED_DATE	DATE
	*	PARTNER_ID	NUMBER (19)
	*	STATE	VARCHAR2 (255 CHAR)
	*	THEATRE_NAME	VARCHAR2 (255 CHAR)

MMK_THEATRE_PK (ID)

SYSTEM.MMK_CUSTOMER

P	*	ID	NUMBER (19)
	*	CONTACT_NUMBER	VARCHAR2 (255 CHAR)
	*	CREATED_DATE	DATE
	*	CUSTOMER_NAME	VARCHAR2 (255 CHAR)
	*	EMAIL_ID	VARCHAR2 (255 CHAR)
	*	NATIONAL_ID	VARCHAR2 (255 CHAR)
	*	NATIONALITY	VARCHAR2 (255 CHAR)

MMK_CUSTOMER_PK (ID)

SYSTEM.MMK_PARTNER

P	*	ID	NUMBER (19)
	*	APPROVED	UNKNOWN
	*	CONTACT_NUMBER	VARCHAR2 (255 CHAR)
	*	CREATED_DATE	DATE
	*	EMAIL_ID	VARCHAR2 (255 CHAR)
	*	NATIONAL_ID	VARCHAR2 (255 CHAR)
	*	NATIONALITY	VARCHAR2 (255 CHAR)
	*	PARTNER_NAME	VARCHAR2 (255 CHAR)

MMK_PARTNER_PK (ID)

SYSTEM.MMK_MOVIE

P	*	ID	NUMBER (19)
	*	BUDGET	FLOAT (53)
	*	CASTING	VARCHAR2 (255 CHAR)
	*	DIRECTOR	VARCHAR2 (255 CHAR)
	*	DURATION	VARCHAR2 (255 CHAR)
		END_TIME	VARCHAR2 (255 CHAR)
	*	LANGUAGE	VARCHAR2 (255 CHAR)
	*	MOVIE_GENRE	NUMBER (3)
	*	MOVIE_NAME	VARCHAR2 (255 CHAR)
	*	PRODUCER	VARCHAR2 (255 CHAR)
	*	PRODUCTION_COMPANY	VARCHAR2 (255 CHAR)
	*	RELEASE_DATE	VARCHAR2 (255 CHAR)
		SHOW_DAY	VARCHAR2 (255 CHAR)
		START_TIME	VARCHAR2 (255 CHAR)
F	*	AUDITORIUM_ID	NUMBER (19)

MMK_MOVIE_PK (ID)

FK8506A8XVNE5K4AJ9XJVIMYGSJ (AUDITORIUM_ID)

S SYSTEM.MMK_THEATRE_AUDI_MOVIE_SEAT_FLAT_MV

SYSTEM.MMK_AUDITORIUM

P	*	ID	NUMBER (19)
	*	AUDITORIUM_NAME	VARCHAR2 (255 CHAR)
	*	CREATED_DATE	DATE
	*	THEATRE_ID	NUMBER (19)

MMK_AUDITORIUM_PK (ID)

SYSTEM.MMK_SEAT

P	*	ID	NUMBER (19)
	*	ROW_NAME	VARCHAR2 (255 CHAR)
		SEAT_AVAIL_STATUS	NUMBER (3)
	*	SEAT_NUMBER	NUMBER (10)
F	*	AUDITORIUM_ID	NUMBER (19)

MMK_SEAT_PK (ID)

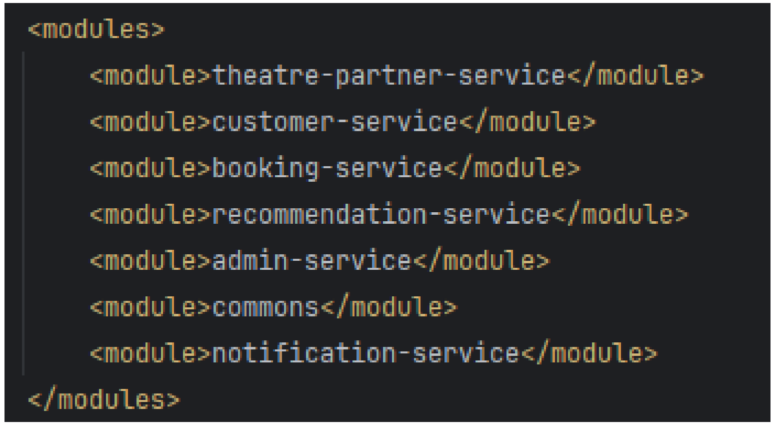
FK47I541M6589EX6DTP02P59W0A (AUDITORIUM_ID)

Working Demo Snapshots

My-Movie-Kart is a multi-module project.

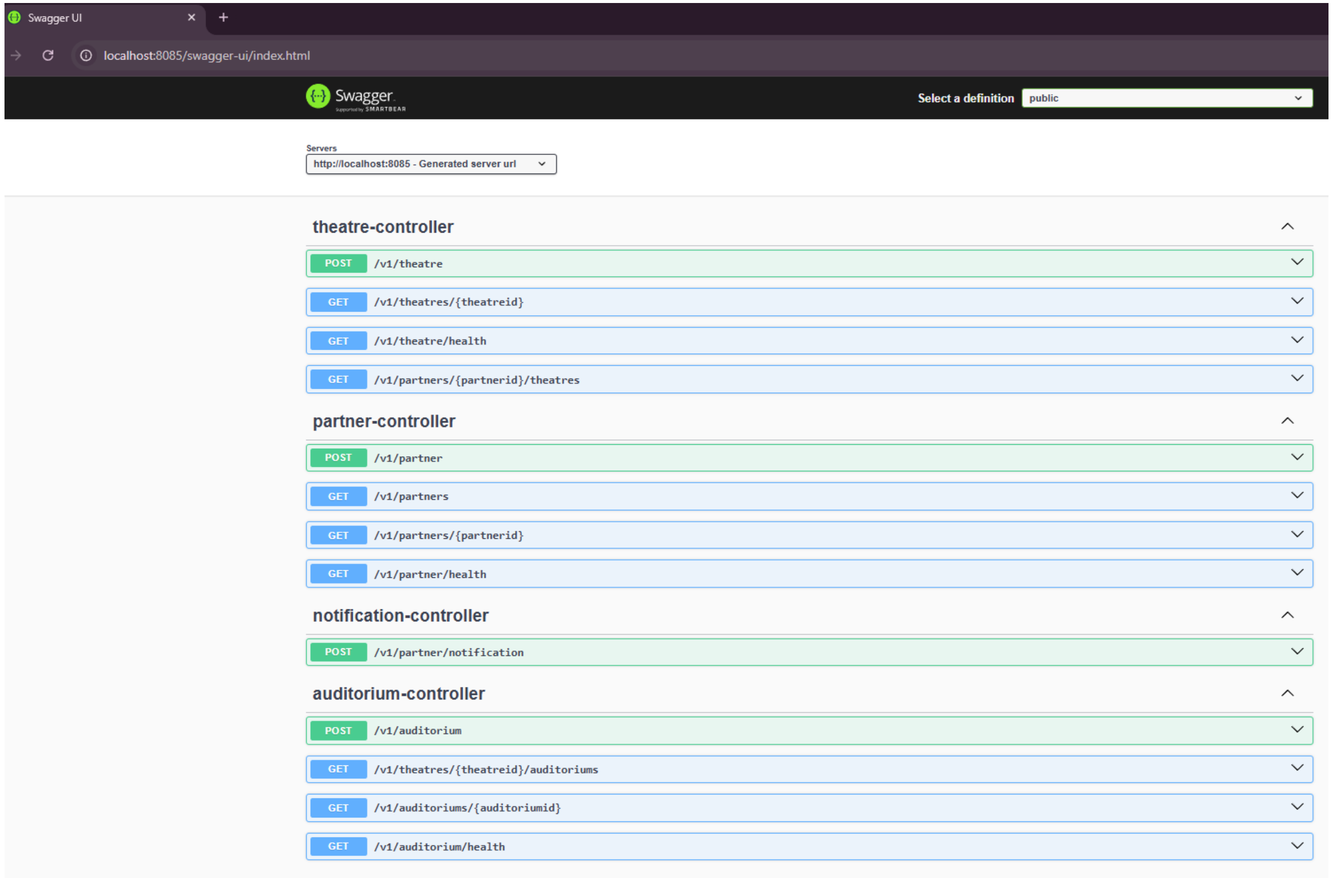
Each module represents a microservice or a repository.

In the following sections, we will see the services implemented as of now, and their respective work flows.



```
<modules>
    <module>theatre-partner-service</module>
    <module>customer-service</module>
    <module>booking-service</module>
    <module>recommendation-service</module>
    <module>admin-service</module>
    <module>commons</module>
    <module>notification-service</module>
</modules>
```

theatre-partner-service


Swagger UI

localhost:8085/swagger-ui/index.html

Select a definition: public

Servers: http://localhost:8085 - Generated server url

theatre-controller

POST /v1/theatre

GET /v1/theatres/{theatreid}

GET /v1/theatre/health

GET /v1/partners/{partnerid}/theatres

partner-controller

POST /v1/partner

GET /v1/partners

GET /v1/partners/{partnerid}

GET /v1/partner/health

notification-controller

POST /v1/partner/notification

auditorium-controller

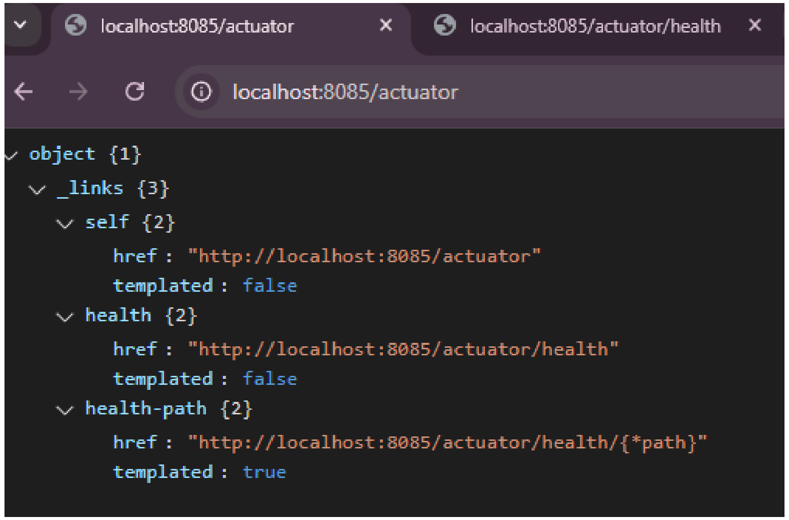
POST /v1/auditorium

GET /v1/theatres/{theatreid}/auditoriums

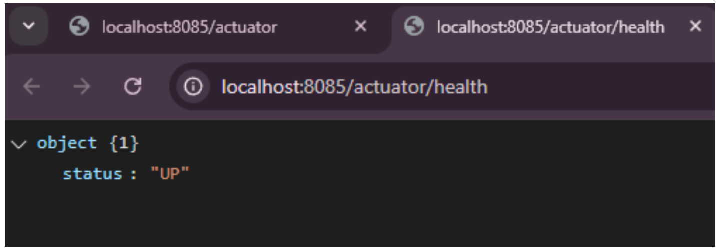
GET /v1/auditoriums/{auditoriumid}

GET /v1/auditorium/health

theatre-partner-service

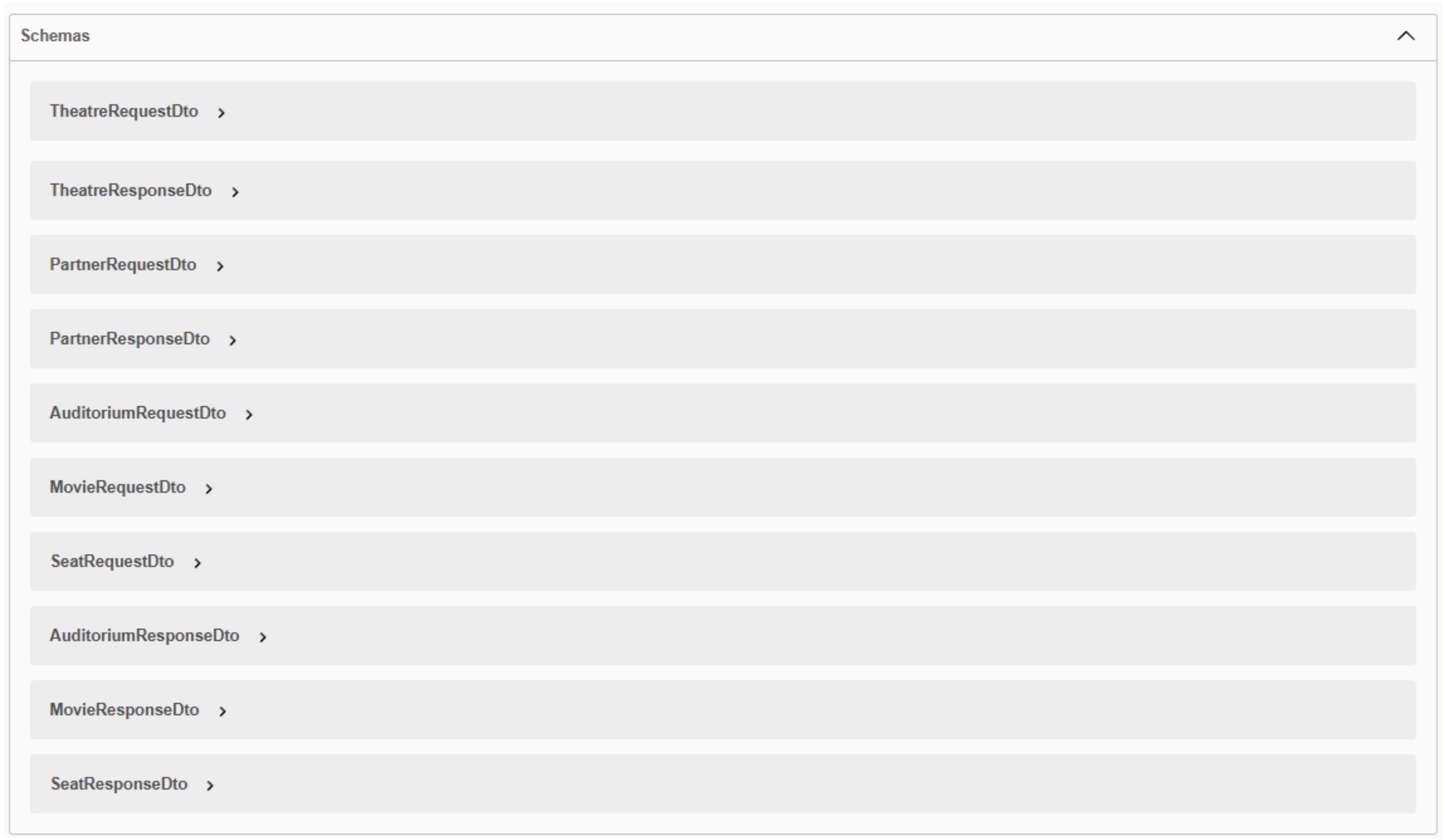


```
object {1}
  _links {3}
    self {2}
      href : "http://localhost:8085/actuator"
      templated : false
    health {2}
      href : "http://localhost:8085/actuator/health"
      templated : false
    health-path {2}
      href : "http://localhost:8085/actuator/health/{*path}"
      templated : true
```


localhost:8085/actuator
localhost:8085/actuator/health
localhost:8085/actuator/health

```
object {1}
  status : "UP"
```

theatre-partner-service



Schemas

TheatreRequestDto

TheatreResponseDto

PartnerRequestDto

PartnerResponseDto

AuditoriumRequestDto

MovieRequestDto

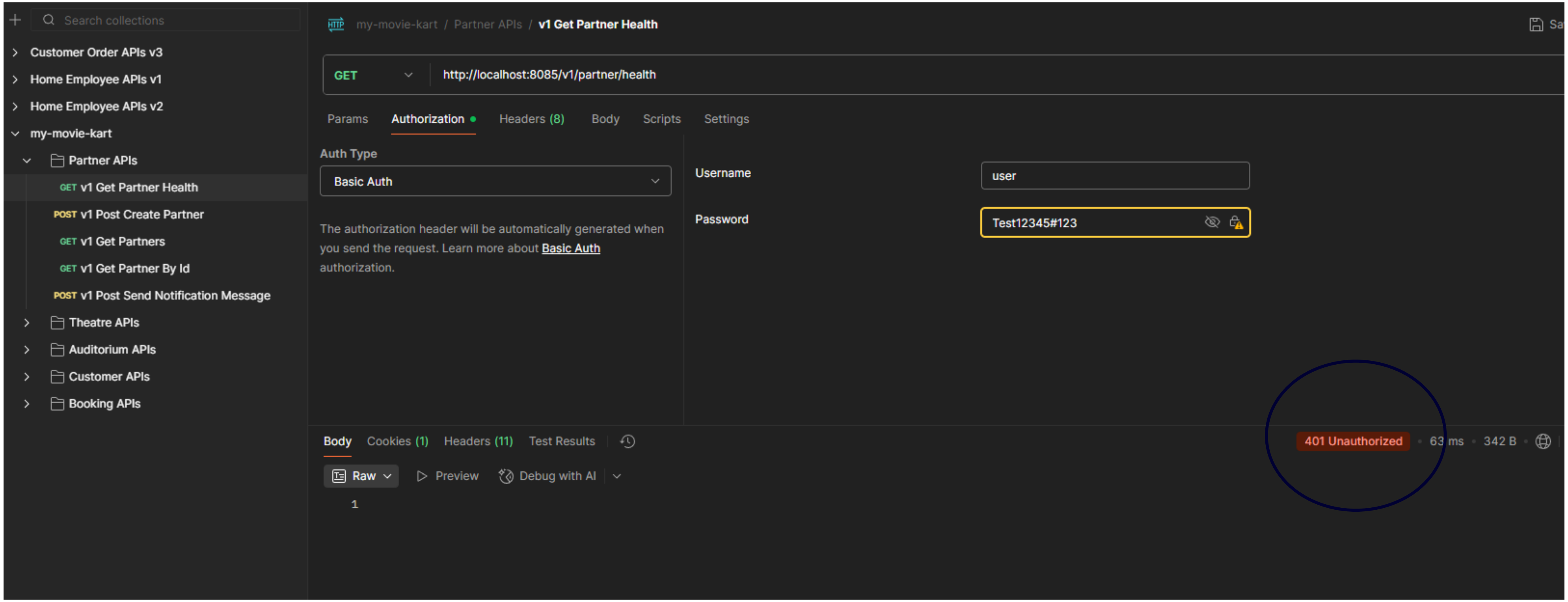
SeatRequestDto

AuditoriumResponseDto

MovieResponseDto

SeatResponseDto

theatre-partner-service



GET http://localhost:8085/v1/partner/health

Auth Type: Basic Auth

The authorization header will be automatically generated when you send the request. Learn more about Basic Auth authorization.

Username: user

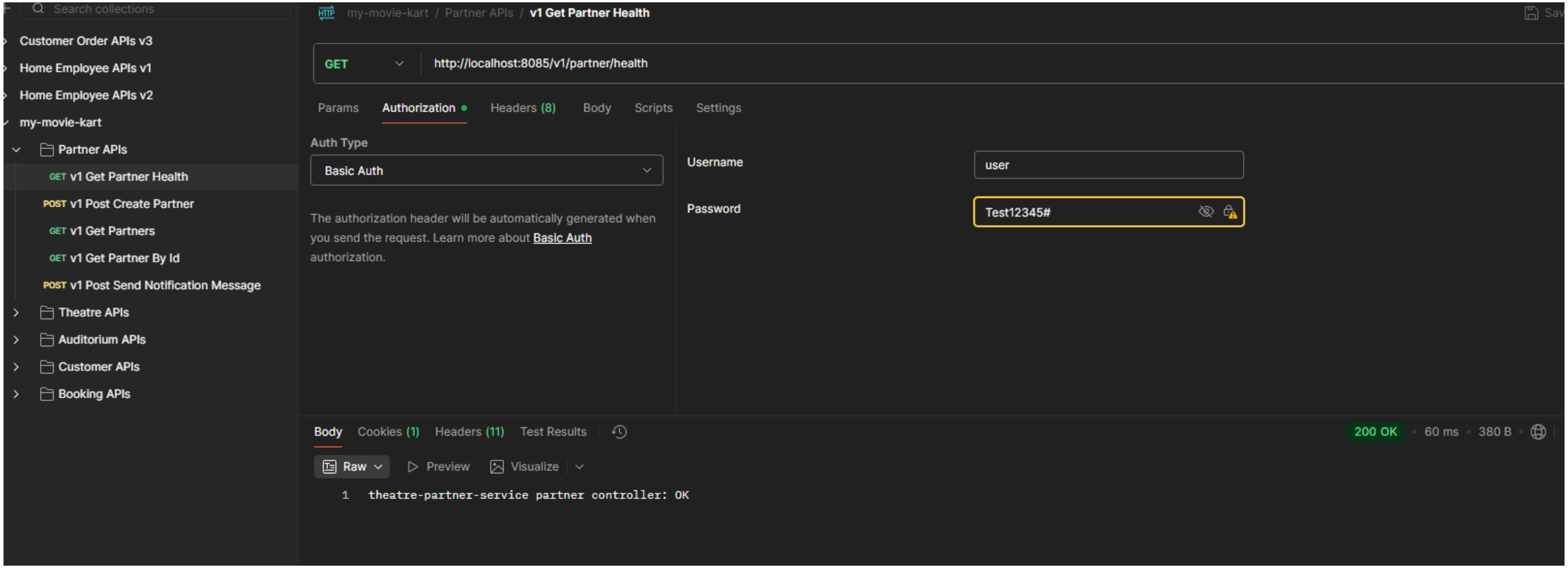
Password: Test12345#123

401 Unauthorized · 63 ms · 342 B

```
1
```

AuthZ failed, due to Spring Security in Action

theatre-partner-service



my-movie-kart / Partner APIs / **v1 Get Partner Health**

GET http://localhost:8085/v1/partner/health

- Customer Order APIs v3
- Home Employee APIs v1
- Home Employee APIs v2
- my-movie-kart
  - Partner APIs
    - GET v1 Get Partner Health
    - POST v1 Post Create Partner
    - GET v1 Get Partners
    - GET v1 Get Partner By Id
    - POST v1 Post Send Notification Message
  - Theatre APIs
  - Auditorium APIs
  - Customer APIs
  - Booking APIs

Params | **Authorization** | Headers (8) | Body | Scripts | Settings

Auth Type: Basic Auth

The authorization header will be automatically generated when you send the request. Learn more about Basic Auth authorization.

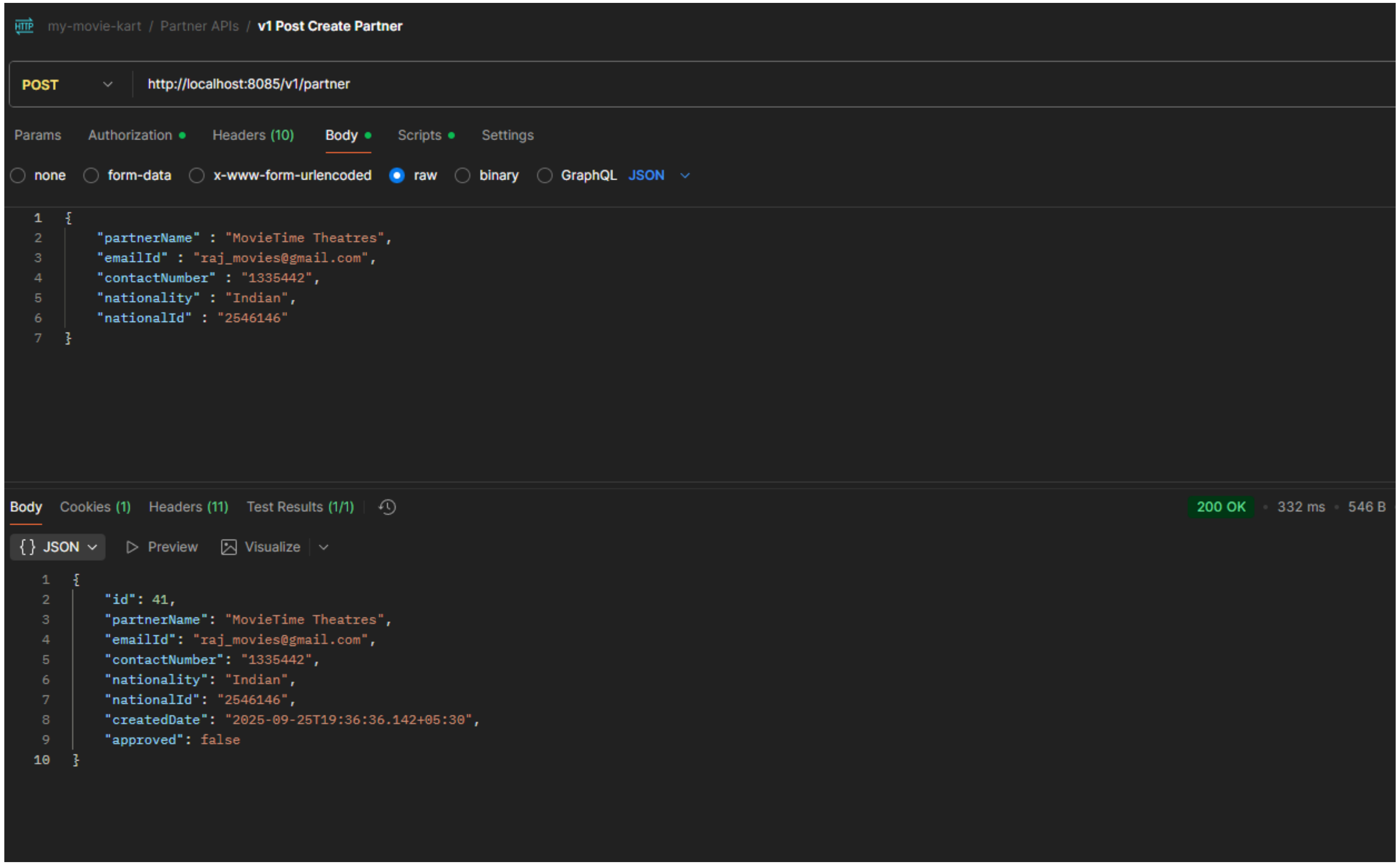
Username	user
Password	Test12345#

Body | Cookies (1) | Headers (11) | Test Results — 200 OK · 60 ms · 380 B

```
theatre-partner-service partner controller: OK
```

AuthZ : success

theatre-partner-service



POST http://localhost:8085/v1/partner

Body: none form-data x-www-form-urlencoded [✓ raw] binary GraphQL JSON

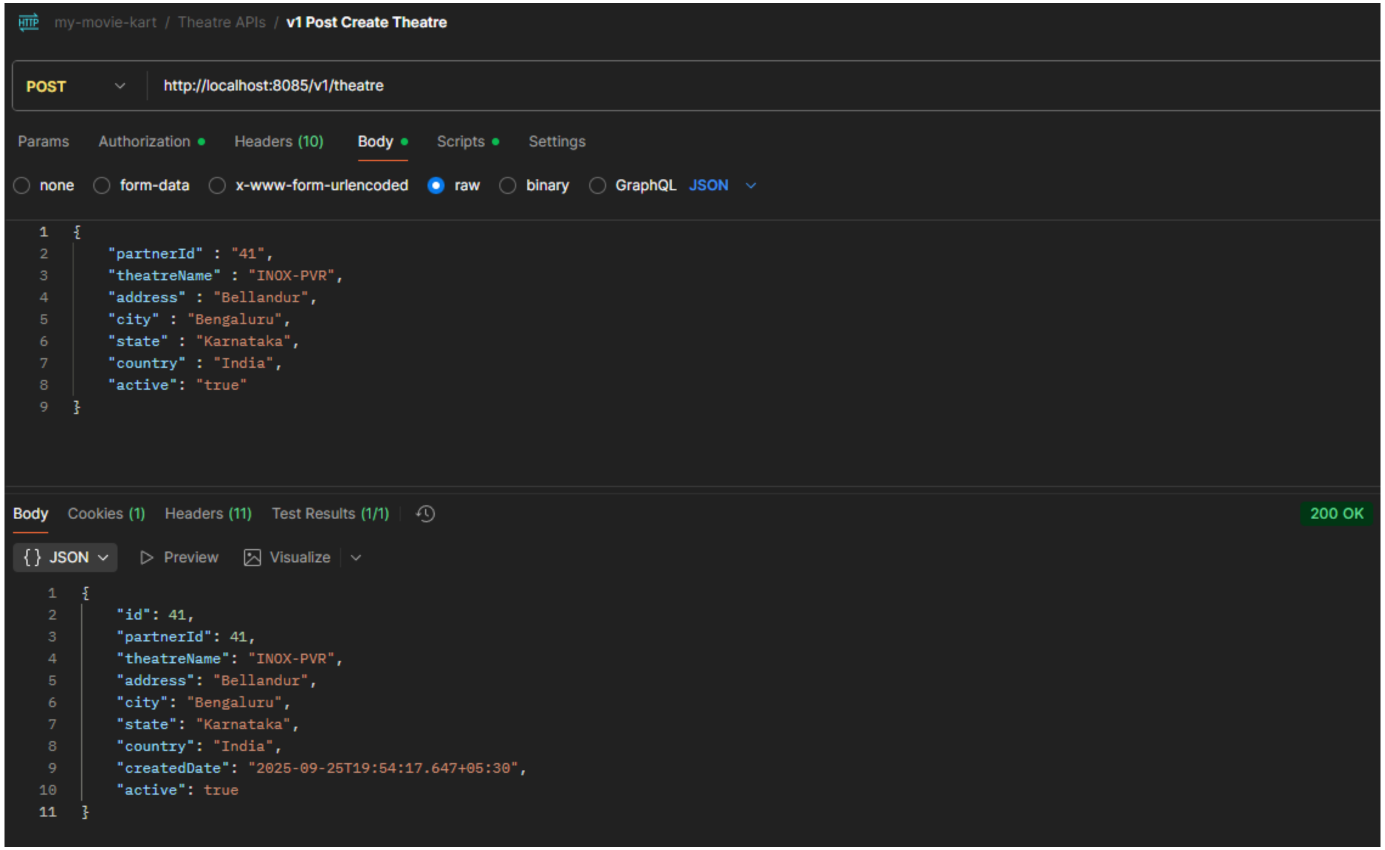
```
{
    "partnerName" : "MovieTime Theatres",
    "emailId" : "raj_movies@gmail.com",
    "contactNumber" : "1335442",
    "nationality" : "Indian",
    "nationalId" : "2546146"
}
```

200 OK • 332 ms • 546 B

{ } JSON

```
{
    "id": 41,
    "partnerName": "MovieTime Theatres",
    "emailId": "raj_movies@gmail.com",
    "contactNumber": "1335442",
    "nationality": "Indian",
    "nationalId": "2546146",
    "createdDate": "2025-09-25T19:36:36.142+05:30",
    "approved": false
}
```

theatre-partner-service



my-movie-kart / Theatre APIs / v1 Post Create Theatre

POST http://localhost:8085/v1/theatre

Params | Authorization ● | Headers (10) | Body ● | Scripts ● | Settings

none | form-data | x-www-form-urlencoded | [✓ raw] | binary | GraphQL | JSON

```
{
    "partnerId" : "41",
    "theatreName" : "INOX-PVR",
    "address" : "Bellandur",
    "city" : "Bengaluru",
    "state" : "Karnataka",
    "country" : "India",
    "active": "true"
}
```

Body | Cookies (1) | Headers (11) | Test Results (1/1) — 200 OK

{} JSON | Preview | Visualize

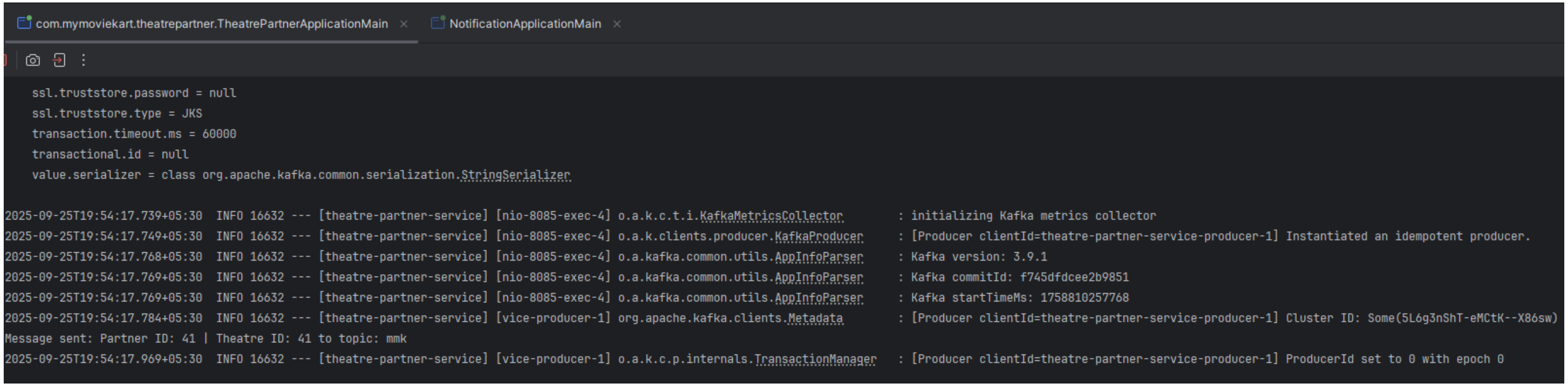
```
{
    "id": 41,
    "partnerId": 41,
    "theatreName": "INOX-PVR",
    "address": "Bellandur",
    "city": "Bengaluru",
    "state": "Karnataka",
    "country": "India",
    "createdDate": "2025-09-25T19:54:17.647+05:30",
    "active": true
}
```

For use-case:

Theatres can create, update, and delete shows for the day.

Theatres can allocate seat inventory and update them for the show

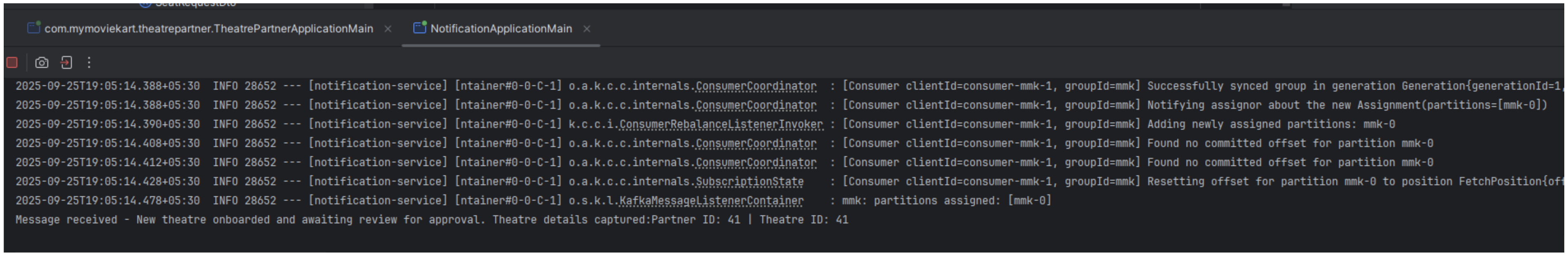
theatre-partner-service



com.mymoviekart.theatrepartner.TheatrePartnerApplicationMain ×   NotificationApplicationMain ×

```
    ssl.truststore.password = null
    ssl.truststore.type = JKS
    transaction.timeout.ms = 60000
    transactional.id = null
    value.serializer = class org.apache.kafka.common.serialization.StringSerializer

2025-09-25T19:54:17.739+05:30  INFO 16632 --- [theatre-partner-service] [nio-8085-exec-4] o.a.k.c.t.i.KafkaMetricsCollector        : initializing Kafka metrics collector
2025-09-25T19:54:17.749+05:30  INFO 16632 --- [theatre-partner-service] [nio-8085-exec-4] o.a.k.clients.producer.KafkaProducer     : [Producer clientId=theatre-partner-service-producer-1] Instantiated an idempotent producer.
2025-09-25T19:54:17.768+05:30  INFO 16632 --- [theatre-partner-service] [nio-8085-exec-4] o.a.kafka.common.utils.AppInfoParser     : Kafka version: 3.9.1
2025-09-25T19:54:17.769+05:30  INFO 16632 --- [theatre-partner-service] [nio-8085-exec-4] o.a.kafka.common.utils.AppInfoParser     : Kafka commitId: f745dfdcee2b9851
2025-09-25T19:54:17.769+05:30  INFO 16632 --- [theatre-partner-service] [nio-8085-exec-4] o.a.kafka.common.utils.AppInfoParser     : Kafka startTimeMs: 1758810257768
2025-09-25T19:54:17.784+05:30  INFO 16632 --- [theatre-partner-service] [vice-producer-1] org.apache.kafka.clients.Metadata        : [Producer clientId=theatre-partner-service-producer-1] Cluster ID: Some(5L6g3nShT-eMCtK--X86sw)
Message sent: Partner ID: 41 | Theatre ID: 41 to topic: mmk
2025-09-25T19:54:17.969+05:30  INFO 16632 --- [theatre-partner-service] [vice-producer-1] o.a.k.c.p.internals.TransactionManager   : [Producer clientId=theatre-partner-service-producer-1] ProducerId set to 0 with epoch 0
```

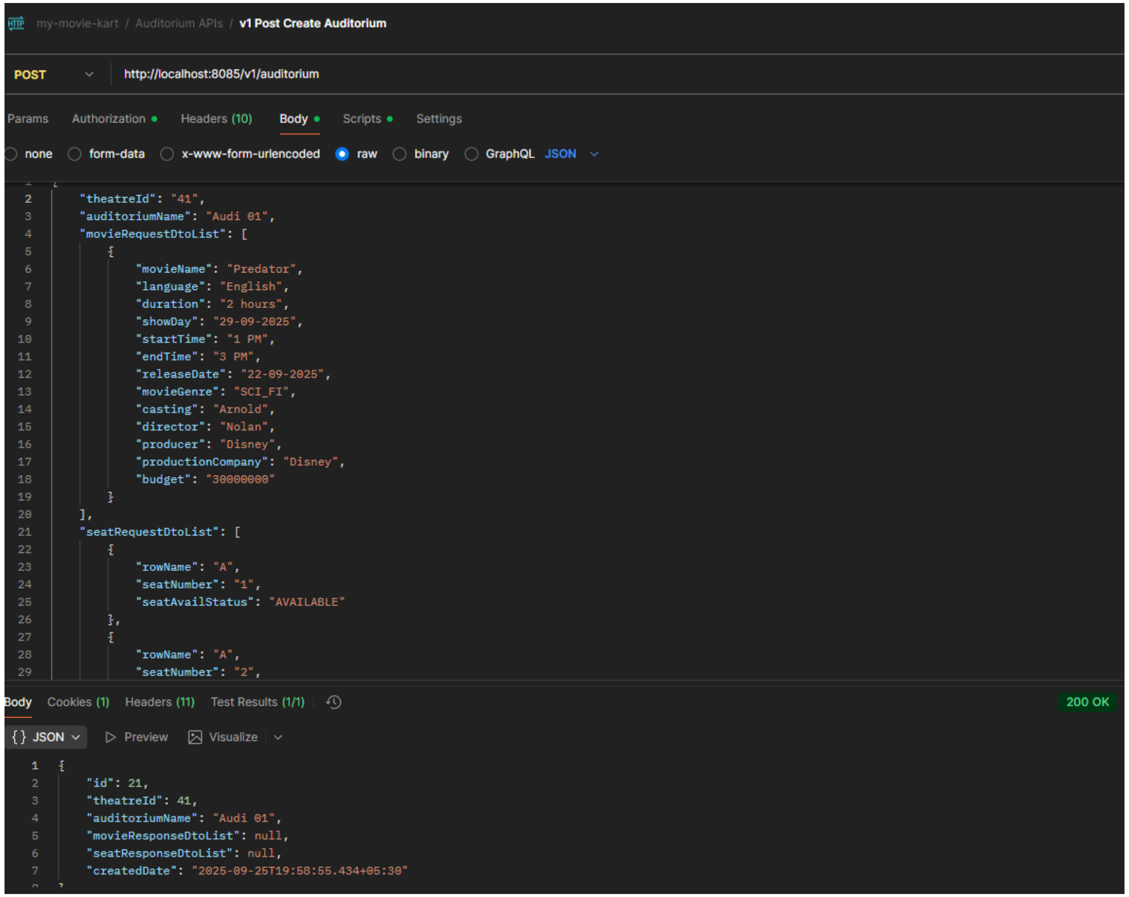


com.mymoviekart.theatrepartner.TheatrePartnerApplicationMain ×   NotificationApplicationMain ×

```
 2025-09-25T19:05:14.388+05:30  INFO 28652 --- [notification-service] [ntainer#0-0-C-1] o.a.k.c.c.internals.ConsumerCoordinator  : [Consumer clientId=consumer-mmk-1, groupId=mmk] Successfully synced group in generation Generation{generationId=1,
 2025-09-25T19:05:14.388+05:30  INFO 28652 --- [notification-service] [ntainer#0-0-C-1] o.a.k.c.c.internals.ConsumerCoordinator  : [Consumer clientId=consumer-mmk-1, groupId=mmk] Notifying assignor about the new Assignment(partitions=[mmk-0])
 2025-09-25T19:05:14.390+05:30  INFO 28652 --- [notification-service] [ntainer#0-0-C-1] k.c.c.i.ConsumerRebalanceListenerInvoker : [Consumer clientId=consumer-mmk-1, groupId=mmk] Adding newly assigned partitions: mmk-0
 2025-09-25T19:05:14.408+05:30  INFO 28652 --- [notification-service] [ntainer#0-0-C-1] o.a.k.c.c.internals.ConsumerCoordinator  : [Consumer clientId=consumer-mmk-1, groupId=mmk] Found no committed offset for partition mmk-0
 2025-09-25T19:05:14.412+05:30  INFO 28652 --- [notification-service] [ntainer#0-0-C-1] o.a.k.c.c.internals.ConsumerCoordinator  : [Consumer clientId=consumer-mmk-1, groupId=mmk] Found no committed offset for partition mmk-0
 2025-09-25T19:05:14.428+05:30  INFO 28652 --- [notification-service] [ntainer#0-0-C-1] o.a.k.c.c.internals.SubscriptionState    : [Consumer clientId=consumer-mmk-1, groupId=mmk] Resetting offset for partition mmk-0 to position FetchPosition{off
 2025-09-25T19:05:14.478+05:30  INFO 28652 --- [notification-service] [ntainer#0-0-C-1] o.s.k.l.KafkaMessageListenerContainer    : mmk: partitions assigned: [mmk-0]
 Message received - New theatre onboarded and awaiting review for approval. Theatre details captured:Partner ID: 41 | Theatre ID: 41
```

Kafka messaging in action

theatre-partner-service



POST http://localhost:8085/v1/auditorium

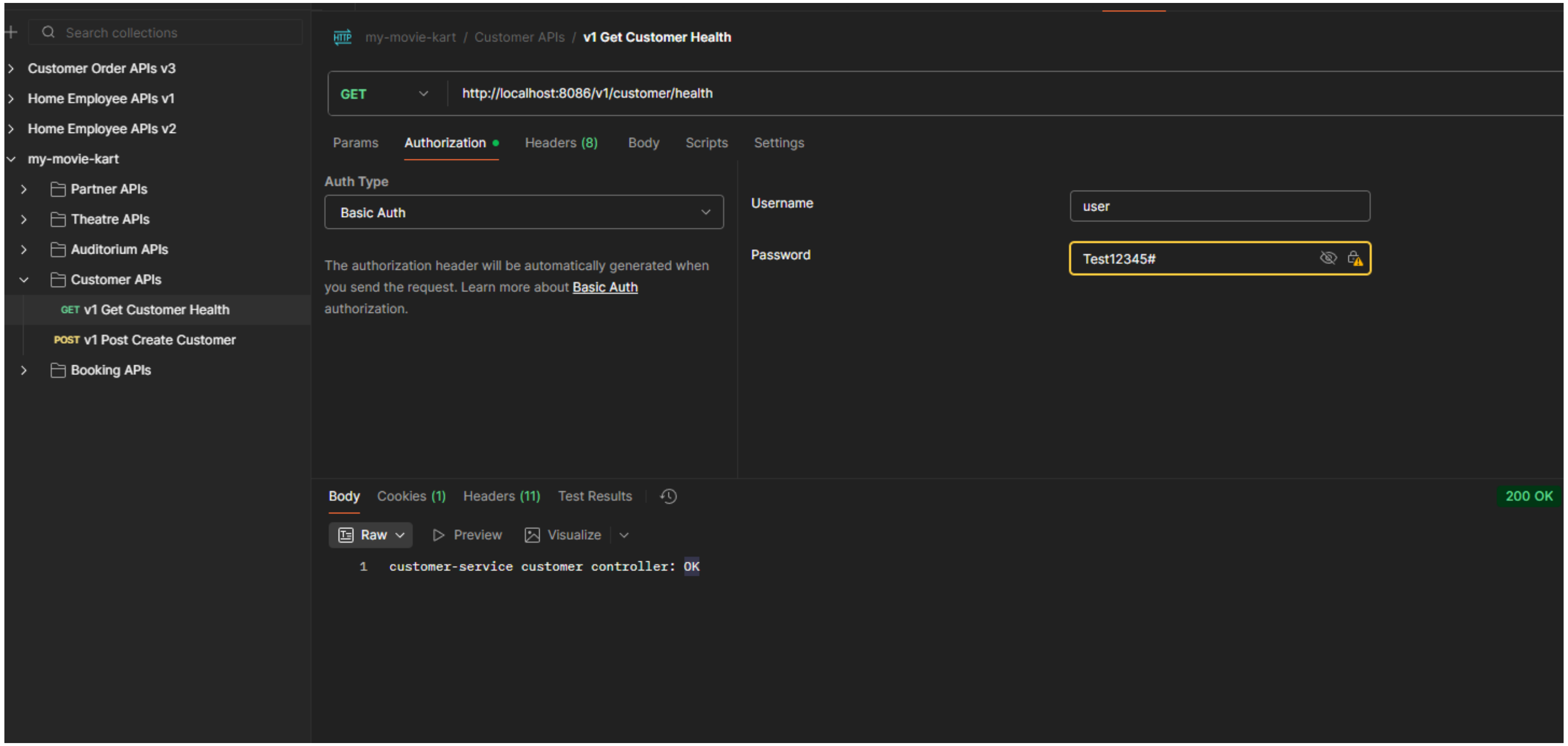
```
    "theatreId": "41",
    "auditoriumName": "Audi 01",
    "movieRequestDtoList": [
        {
            "movieName": "Predator",
            "language": "English",
            "duration": "2 hours",
            "showDay": "29-09-2025",
            "startTime": "1 PM",
            "endTime": "3 PM",
            "releaseDate": "22-09-2025",
            "movieGenre": "SCI_FI",
            "casting": "Arnold",
            "director": "Nolan",
            "producer": "Disney",
            "productionCompany": "Disney",
            "budget": "30000000"
        }
    ],
    "seatRequestDtoList": [
        {
            "rowName": "A",
            "seatNumber": "1",
            "seatAvailStatus": "AVAILABLE"
        },
        {
            "rowName": "A",
            "seatNumber": "2",
```

```
{
    "id": 21,
    "theatreId": 41,
    "auditoriumName": "Audi 01",
    "movieResponseDtoList": null,
    "seatResponseDtoList": null,
    "createdDate": "2025-09-25T19:58:55.434+05:30"
```

Hierarchy followed:

theatre > audi > seats and movies

customer-service



GET http://localhost:8086/v1/customer/health

Auth Type: Basic Auth

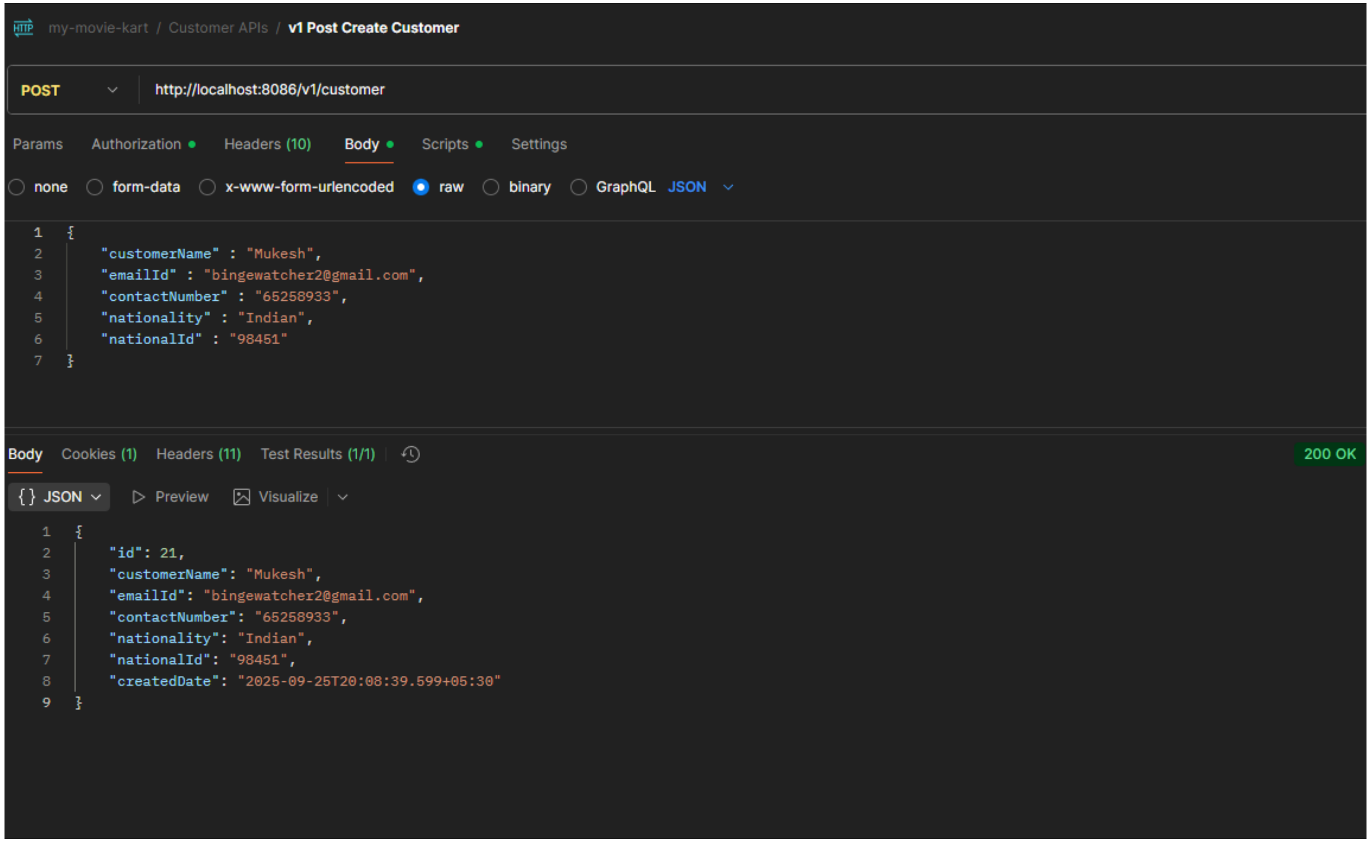
The authorization header will be automatically generated when you send the request. Learn more about Basic Auth authorization.

Username: user

Password: Test12345#

```
customer-service customer controller: OK
```

customer-service



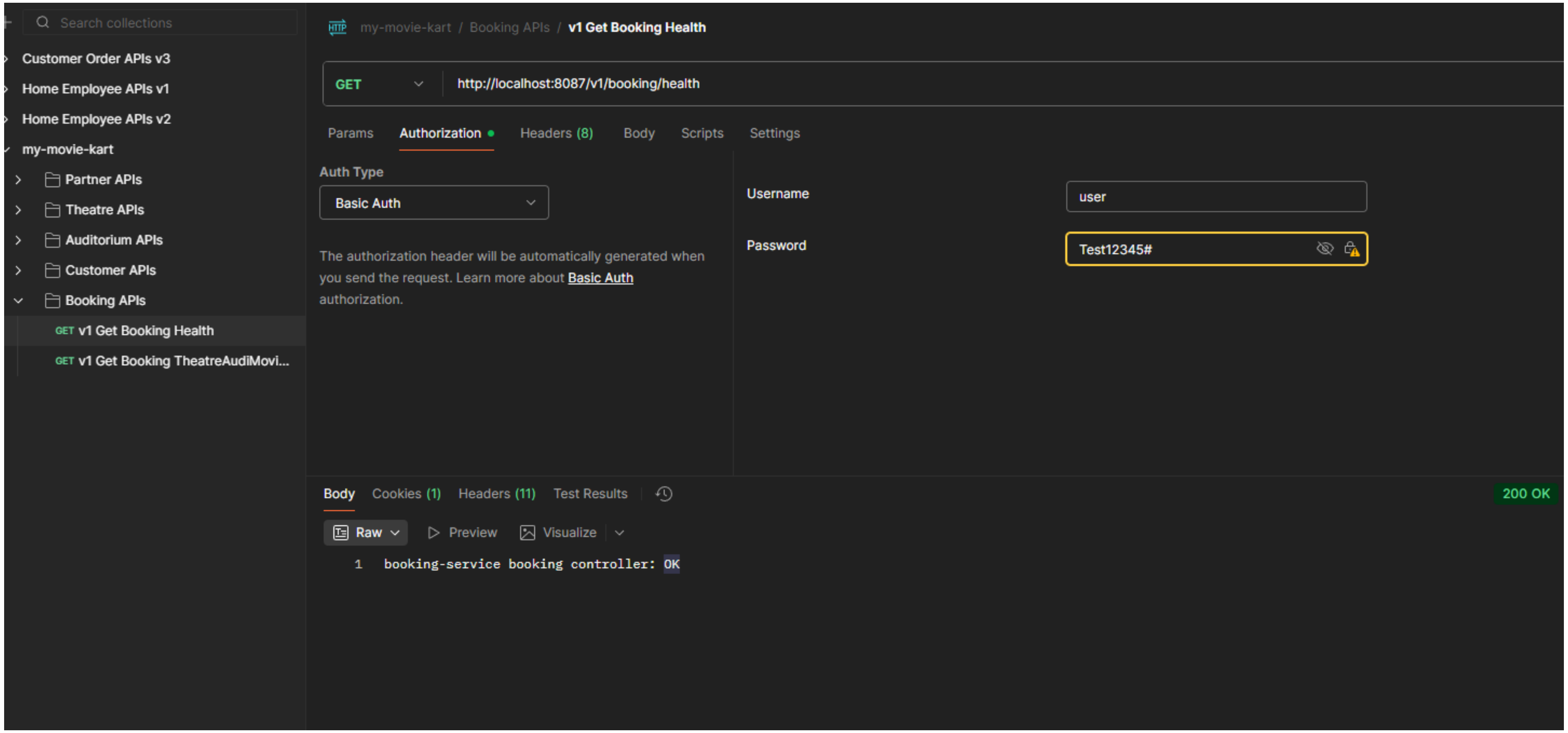
POST http://localhost:8086/v1/customer

```
{
    "customerName" : "Mukesh",
    "emailId" : "bingewatcher2@gmail.com",
    "contactNumber" : "65258933",
    "nationality" : "Indian",
    "nationalId" : "98451"
}
```

200 OK

```
{
    "id": 21,
    "customerName": "Mukesh",
    "emailId": "bingewatcher2@gmail.com",
    "contactNumber": "65258933",
    "nationality": "Indian",
    "nationalId": "98451",
    "createdDate": "2025-09-25T20:08:39.599+05:30"
}
```

booking-service



GET http://localhost:8087/v1/booking/health

Auth Type: Basic Auth

The authorization header will be automatically generated when you send the request. Learn more about Basic Auth authorization.

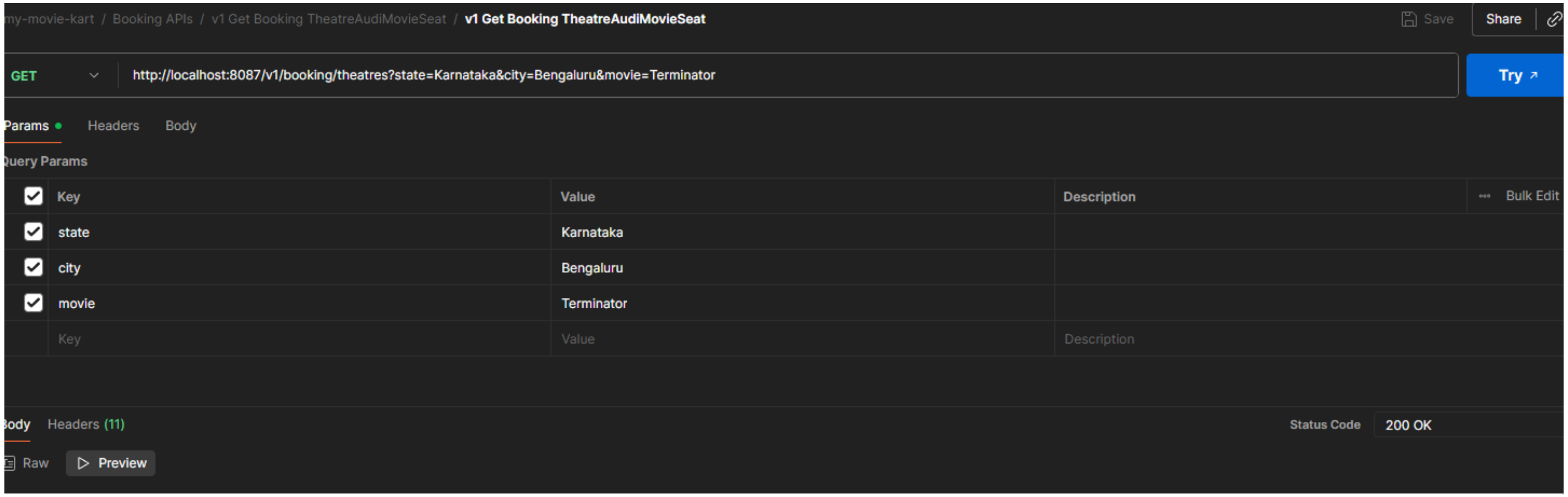
Username: user

Password: Test12345#

200 OK

```
booking-service booking controller: OK
```

booking-service


my-movie-kart / Booking APIs / v1 Get Booking TheatreAudiMovieSeat / **v1 Get Booking TheatreAudiMovieSeat**

GET http://localhost:8087/v1/booking/theatres?state=Karnataka&city=Bengaluru&movie=Terminator

Query Params

Key	Value	Description
✓ state	Karnataka	
✓ city	Bengaluru	
✓ movie	Terminator	

Status Code 200 OK

[{ "theatreId": 1, "theatreName": "PVR", "theatreActive": true, "theatreAddress": "Bellandur", "theatreCity": "Bengaluru", "theatreState": "Karnataka", "theatreXountry": null, "theatrePartnerId": 1, "auditoriumId": 2,
"auditoriumName": "Audi 22", "movieId": 2, "movieName": "Terminator", "movieBudget": 30000000, "movieCasting": "Arnold", "movieDirector": "Nolan", "movieProducer": "Disney", "movieProductionCompany":
"Disney", "movieReleasedDate": "22-09-2025", "movieGenre": "SCI_FI", "movieLanguage": "Hinid", "movieDuration": "2 hours", "movieStartTime": "9 PM", "movieEndTime": "12 PM", "movieShowDate": "22-09-2025",
"seatId": 6, "seatRow": "A", "seatNumber": 1, "seatAvailability": "AVAILABLE" }, { "theatreId": 1, "theatreName": "PVR", "theatreActive": true, "theatreAddress": "Bellandur", "theatreCity": "Bengaluru", "theatreState":
"Karnataka", "theatreXountry": null, "theatrePartnerId": 1, "auditoriumId": 2, "auditoriumName": "Audi 22", "movieId": 2, "movieName": "Terminator", "movieBudget": 30000000, "movieCasting": "Arnold",
"movieDirector": "Nolan", "movieProducer": "Disney", "movieProductionCompany": "Disney", "movieReleasedDate": "22-09-2025", "movieGenre": "SCI_FI", "movieLanguage": "Hinid", "movieDuration": "2 hours",
"movieStartTime": "9 PM", "movieEndTime": "12 PM", "movieShowDate": "22-09-2025", "seatId": 6, "seatRow": "A", "seatNumber": 1, "seatAvailability": "AVAILABLE" }, { "theatreId": 1, "theatreName": "PVR",
"theatreActive": true, "theatreAddress": "Bellandur", "theatreCity": "Bengaluru", "theatreState": "Karnataka", "theatreXountry": null, "theatrePartnerId": 1, "auditoriumId": 2, "auditoriumName": "Audi 22", "movieId": 2,
"movieName": "Terminator", "movieBudget": 30000000, "movieCasting": "Arnold", "movieDirector": "Nolan", "movieProducer": "Disney", "movieProductionCompany": "Disney", "movieReleasedDate": "22-09-2025",
"movieGenre": "SCI_FI", "movieLanguage": "Hinid", "movieDuration": "2 hours", "movieStartTime": "9 PM", "movieEndTime": "12 PM", "movieShowDate": "22-09-2025", "seatId": 6, "seatRow": "A", "seatNumber": 1,
"seatAvailability": "AVAILABLE" }, { "theatreId": 1, "theatreName": "PVR", "theatreActive": true, "theatreAddress": "Bellandur", "theatreCity": "Bengaluru", "theatreState": "Karnataka", "theatreXountry": null,
"theatrePartnerId": 1, "auditoriumId": 2, "auditoriumName": "Audi 22", "movieId": 2, "movieName": "Terminator", "movieBudget": 30000000, "movieCasting": "Arnold", "movieDirector": "Nolan", "movieProducer":
"Disney", "movieProductionCompany": "Disney", "movieReleasedDate": "22-09-2025", "movieGenre": "SCI_FI", "movieLanguage": "Hinid", "movieDuration": "2 hours", "movieStartTime": "9 PM", "movieEndTime": "12
PM", "movieShowDate": "22-09-2025", "seatId": 6, "seatRow": "A", "seatNumber": 1, "seatAvailability": "AVAILABLE" }, { "theatreId": 1, "theatreName": "PVR", "theatreActive": true, "theatreAddress": "Bellandur",
"theatreCity": "Bengaluru", "theatreState": "Karnataka", "theatreXountry": null, "theatrePartnerId": 1, "auditoriumId": 2, "auditoriumName": "Audi 22", "movieId": 2, "movieName": "Terminator", "movieBudget":
30000000, "movieCasting": "Arnold", "movieDirector": "Nolan", "movieProducer": "Disney", "movieProductionCompany": "Disney", "movieReleasedDate": "22-09-2025", "movieGenre": "SCI_FI", "movieLanguage":
"Hinid", "movieDuration": "2 hours", "movieStartTime": "9 PM", "movieEndTime": "12 PM", "movieShowDate": "22-09-2025", "seatId": 6, "seatRow": "A", "seatNumber": 1, "seatAvailability": "AVAILABLE" }]

For use-case:

Browse theatres currently running the show (movie selected) in the town, including show timing by a chosen date.

Additional features to be added (*WIP*)

- Extend the recommendation POC to leverage the movie database for recommending movies
- Add end to end jUnits with Mockito framework for all the service modules
- Add exhaustive error handling using custom exceptions
- Adding seat booking logic
- Adding admin cotrol at platform level and add email notifications
- Adding containerization of the remaining services
- Adding api gateway and service discovery
- Adding SSO and OAuth2
- Active monitoring using dashboards like Grafana

Scope for future enhancements

- Adding Load Balancing to handle increasing load
- Adding HPA using helm charts after adding Docker and K8
- Enhancing model for better recommendations using fine tuning
- Scaling up the database and adding Redis cache at every service level with frequent api calls
- Pro-active vulnerability detection and remediation, for compliance
- Monetization of the platforms showing relevant ads and campaigns

Quick notes on:

- Platform provisioning, sizing & Release requirements
- Product and Stakeholder Management

- Defined goals, sponsors, timelines, budget, and resource allocation
- Regular evaluation of technology investment against estimated ROI
- Regularly tracking progress, measuring key performance indicators (KPIs), assessing risks, and adjusting priorities per business/market needs
- Proactive identification, adherence and mitigation of security vulnerabilities, regulatory compliances
- Optimal allocation of IT resources and reasoning for opting cloud solutions over on-prem and vice-versa
- Aligning development objectives towards COTS (Commercial-Off-The-Shelf) enterprise systems (pre-built and provide advantages like affordability, faster implementation, and scalability), in case business demands so
- Active governance and stakeholder engagement to ensure alignment and accountability at every level of hierarchy
- Seeking continuous improvement to stay aligned with evolving business goals

Thank you