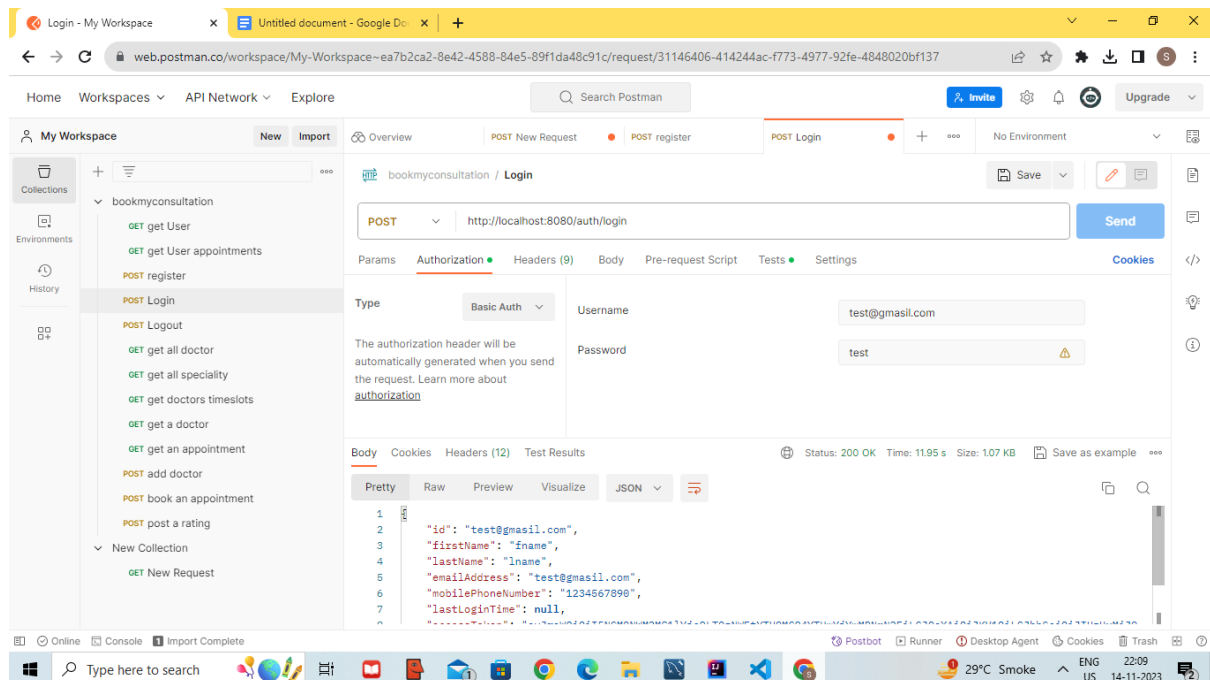


Register:<http://localhost:8080/users/register>

Register:<http://localhost:8080/users/register>



Login : <http://localhost:8080/auth/login>



Login failed

The screenshot shows the Postman interface with a workspace named "My Workspace". The "bookmyconsultation" collection is selected, and the "POST Login" request is highlighted in the history. The request is a POST to `http://localhost:8080/auth/login` with Basic Auth. The username is `test@gmail.com` and the password is `test1`. The response status is `401 Unauthorized` with a time of `533 ms` and a size of `703 B`. The response body is:

```
1 {
2   "code": "USR-003",
3   "message": "Password match failed",
4   "root_cause": null
5 }
```

Logout: `http://localhost:8080/auth/logout`

The screenshot shows the Postman interface with the same workspace. The "POST Logout" request is highlighted in the history. The request is a POST to `http://localhost:8080/auth/logout`. The response status is `401 Unauthorized` with a time of `19 ms` and a size of `736 B`. The response body is:

```
1 {
2   "timestamp": "2023-11-14T16:47:02.323+00:00",
3   "status": 401,
4   "error": "Unauthorized",
5   "path": "/auth/logout"
6 }
```

Get User: <http://localhost:8080/users/test@gmasil.com>

The screenshot shows the Postman interface with a workspace named 'My Workspace'. The 'GET get User' request is selected, showing a successful response with status 200 OK. The response body is a JSON object with the following details:

Key	Value	Description
emailId	"test@gmasil.com"	
firstName	"fname"	
lastName	"lname"	
dob	"1993-08-06"	
mobile	"1234567890"	
password	"742985f9c5e2e78"	
createdDate	"2023-11-14"	

Get user Failed when user not present

The screenshot shows the Postman interface with the same workspace. The 'GET get User' request is selected, but the response is a 401 Unauthorized error. The response body is a JSON object with the following details:

Key	Value	Description
timestamp	"2023-11-14T16:57:07.546+08:00"	
status	401	
error	"Unauthorized"	
path	"/users/test@gmasil.com"	

Get User Appointment when no appointment is present

The screenshot shows the Postman interface with a GET request to `http://localhost:8080/users/test@gmail.com/appointments`. The request is part of a collection named 'bookmyconsultation'. The response status is 200 OK, and the body is empty, indicating no appointments are present for the user.

Key	Value	Description
Key	Value	Description

Get User Appointments: `http://localhost:8080/users/test@gmail.com/appointments`

The screenshot shows the Postman interface with a GET request to `http://localhost:8080/users/test@gmail.com/appointments`. The response status is 200 OK, and the body contains a JSON array with one appointment object.

```
1
2
3  {
4    "appointmentId": "172d958b-a464-47e5-b78d-bc36a2415245",
5    "doctorId": "UUID-34",
6    "doctorName": "Meghan Jordan",
7    "userId": "test@gmail.com",
8    "userName": "fname",
9    "userEmailId": "test@gmail.com",
10   "timeSlot": "06PM-06PM",
11 }
```

Add Doctor: <http://localhost:8080/doctors>

The screenshot shows the Postman interface with a workspace named 'bookmyconsultation'. The 'POST' method is selected for the endpoint 'http://localhost:8080/doctors'. The 'Body' tab is active, displaying a JSON response in 'Pretty' format. The response status is 200 OK, with a response time of 677 ms and a size of 970 B.

Key	Value	Description
id	"8272c56b-92c3-4de3-bd18-4a9f712b4988"	
firstName	"fname"	
lastName	"lname"	
speciality	"GENERAL_PHYSICIAN"	
dob	"1983-08-06"	
address	{ "id": "9cc4a5f4-9f7d-42ea-a0e5-d26a56835a83", "addressLine1": "add1", "addressLine2": "add1", }	

Get a Doctor: <http://localhost:8080/doctors/UUID-2>

The screenshot shows the Postman interface with a workspace named 'bookmyconsultation'. The 'GET' method is selected for the endpoint 'http://localhost:8080/doctors/UUID-2'. The 'Body' tab is active, displaying a JSON response in 'Pretty' format. The response status is 200 OK, with a response time of 2.89 s and a size of 1.05 KB.

Key	Value	Description
id	"UUID-2"	
firstName	"Blossom"	
lastName	"Valentine"	
speciality	"PULMONOLOGIST"	
dob	"2021-09-16"	
address	{ "id": "UUID-2", "addressLine1": "103-4867 Nullam Ave", "addressLine2": "-48.86634, 146.33318", "city": "Patalillo", }	

The screenshot displays the Postman application interface. At the top, the browser address bar shows the URL: `web.postman.co/workspace/My-Workspace~ea7b2ca2-8e42-4588-84e5-89f1da48c91c/request/31146406-e5c9e6d6-fd7d-41b6-ba79-e45e47c4d84e`.

The main interface is divided into three panes:

- Left Pane (Collections):** Shows the 'My Workspace' collection. Under the 'bookmyconsultation' folder, the endpoint 'GET get a doctor' is selected.
- Middle Pane (Request Details):** Displays the details for the 'GET get a doctor' endpoint. The URL is `http://localhost:8080/doctors/UUID-1014`. The 'Params' section is empty. The 'Query Params' table is also empty.
- Right Pane (Response):** Shows the response body in the 'Body' tab. The status is '500 Internal Server Error' with a time of 1783 ms and a size of 732 KB. The response body is a JSON object:


```

{
  "code": "GEN-001",
  "message": "An unexpected error occurred. Please contact System Administrator",
  "root_cause": "com.upgrad.bookmyconsultation.exception.ResourceUnavailableException\r\n\tat com.upgrad.bookmyconsultation.service.DoctorService.getDoctor(DoctorService.java:64)\r\n\tat com.upgrad.bookmyconsultation.controller.DoctorController.getDoctorDetails(DoctorController.java:33)\r\n\tat java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke(Native Method)\r\n\tat java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:77)\r\n\tat java.base/jdk.internal.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)\r\n\tat java.base/java.lang.reflect.Method.invoke(Method.java:568)\r\n\tat org.springframework.web.method.support.InvocableHandlerMethod.doInvoke"
}
```

```
:http://localhost:8080/appointments/172d950b-a464-47e5-b78d-bc36a2415245
```

The screenshot displays the Postman web application interface. On the left sidebar, the 'My Workspace' section is active, showing a collection named 'bookmyconsultation' with various API endpoints. The main workspace area is configured for a 'GET' request to the endpoint 'http://localhost:8080/appointments/172d950b-a464-47e5-b78d-bc36a2415245'. The request body is set to 'JSON' and contains the following data:

```

{
  "firstName": "fname",
  "lastName": "lname",
  "dob": "1903-08-08",
  "mobile": "1234567890",
  "password": "test",
  "emailId": "test@gmail.com",
}

```

The response is displayed in the 'Body' tab, showing a 302 Found status. The response body is in JSON format:

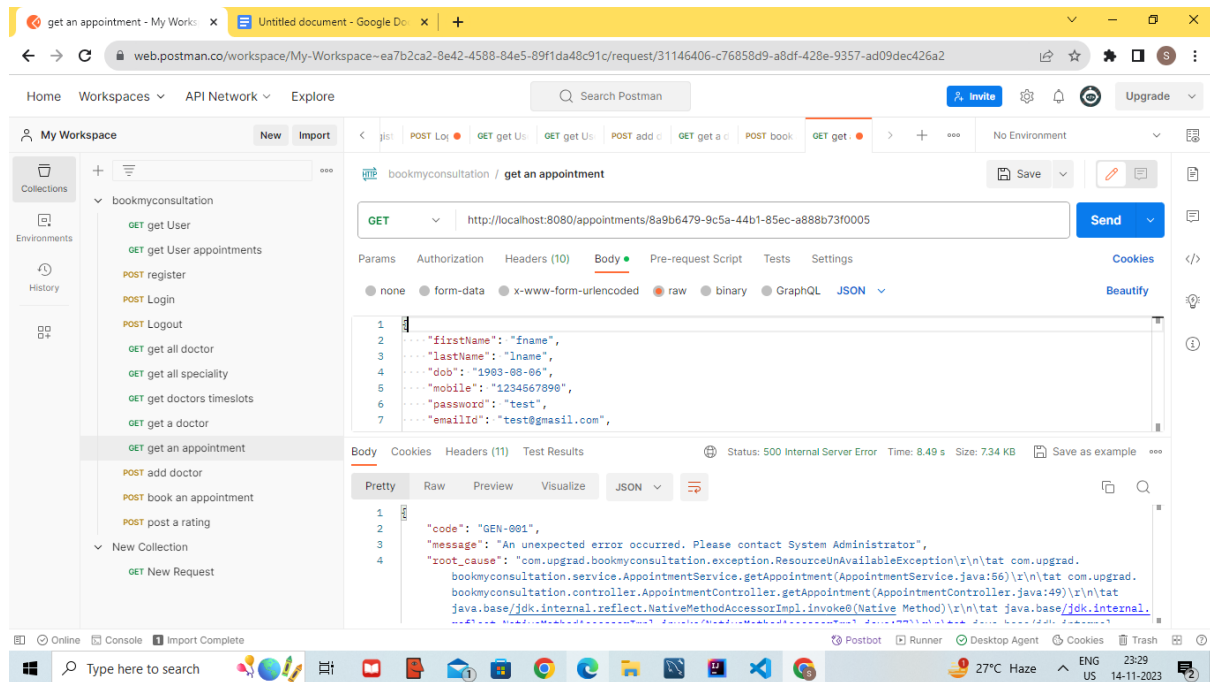
```

{
  "appointmentId": "172d950b-a464-47e5-b78d-bc36a2415245",
  "doctorId": "UUID-34",
  "doctorName": "Meghan Jordan",
  "userId": "test@gmail.com",
  "userName": "fname",
  "userEmailId": "test@gmail.com",
}

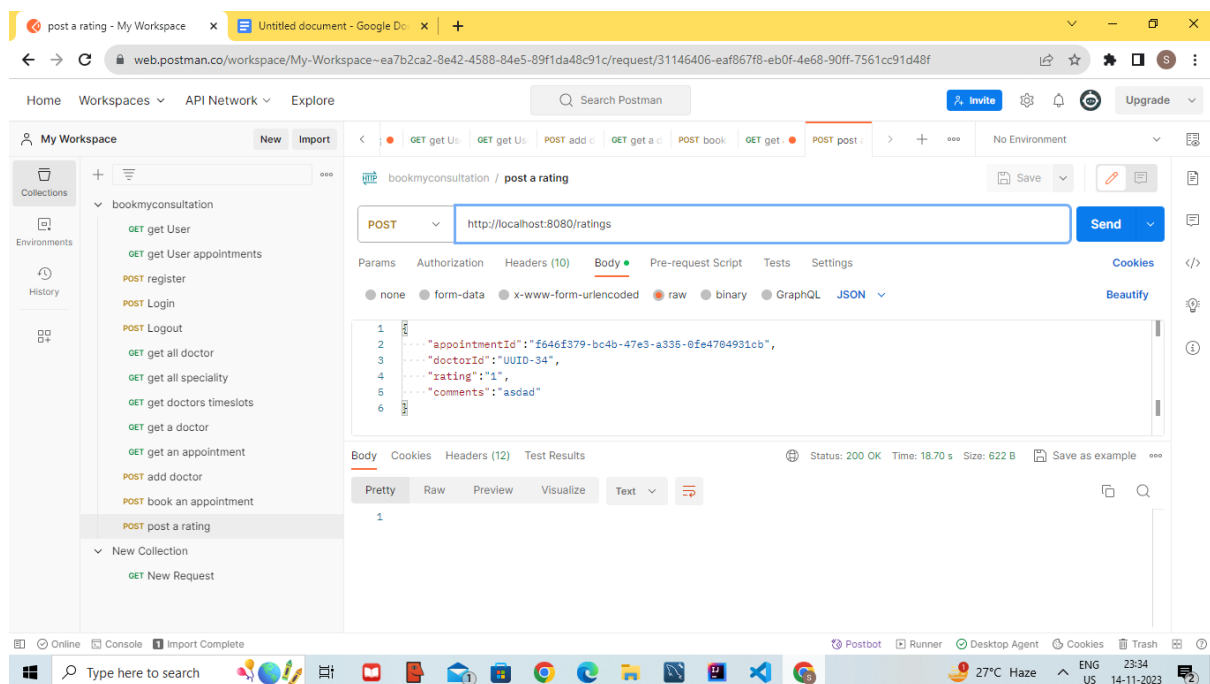
```

The bottom status bar indicates the status is 302 Found, time is 117 ms, and size is 909 B. The bottom of the screen shows the Windows taskbar with various application icons and the system clock showing 23:27 on 14-11-2023.

Get an Appointment when appointment not found



Post a Rating: <http://localhost:8080/ratings>



Get All Doctor: <http://localhost:8080/doctors?speciality=CARDIOLOGIST>

The screenshot shows the Postman interface with a workspace named 'bookmyconsultation'. The 'GET all doctor' collection is selected, and the 'GET all doctor' request is active. The URL is `http://localhost:8080/doctors?speciality=CARDIOLOGIST`. The 'Query Params' table shows a single parameter: `speciality` with the value `CARDIOLOGIST`. The 'Body' tab shows the response in 'Pretty' format:

```
1 {
2   "id": "UUID-42",
3   "firstName": "Alexis",
4   "lastName": "Rodriguez",
5   "speciality": "CARDIOLOGIST",
6   "dob": "2020-11-07",
7   "address": {
8     "id": "UUID-42",
```

The status bar at the bottom indicates a 200 OK response with a time of 1940 ms and a size of 8.48 KB.

Get All Specialty: <http://localhost:8080/doctors/speciality>

The screenshot shows the Postman interface with the same workspace. The 'GET all specialty' collection is selected, and the 'GET all specialty' request is active. The URL is `http://localhost:8080/doctors/speciality`. The 'Body' tab shows the response in 'Pretty' format:

```
1 [
2   "CARDIOLOGIST",
3   "GENERAL_PHYSICIAN",
4   "DENTIST",
5   "PULMONOLOGIST",
6   "ENT",
7   "GASTRO"
8 ]
```

The status bar at the bottom indicates a 200 OK response with a time of 2.94 s and a size of 700 B.

Get Doctor Timeslots: <http://localhost:8080/doctors/UUID-34/timeSlots?date=2021-05-26>

The screenshot shows the Postman interface with a GET request to `http://localhost:8080/doctors/UUID-34/timeSlots?date=2021-05-26`. The request is saved in a collection named 'bookmyconsultation'. The 'Query Params' tab is active, showing a single parameter: 'date' with the value '2021-05-26'. The 'Body' tab is also visible, showing a JSON response in 'Pretty' format:

```
{
  "doctorId": "UUID-34",
  "availableDate": "2021-05-26",
  "timeSlot": [
    "08PM-09PM",
    "10AM-11AM",
    "12AM-01PM",
    "07PM-08PM",
    "11AM-12AM"
  ]
}
```

The status bar at the bottom indicates a 200 OK response with a time of 7.46 s and a size of 747 B.

Database schema and Tables

The screenshot shows the MySQL Workbench interface. The 'SCHEMAS' panel on the left displays the database structure, including tables like 'consultation', 'address', 'appointment', 'doctor', 'rating', 'user', and 'user_auth_token'. The 'SQL File 7*' editor shows a query:

```
1 use consultation;
2
3 select * from consultation.address;
```

The 'Result Grid' displays the query results, showing columns: id, address_line1, address_line2, city, postcode, and state. The results are as follows:

id	address_line1	address_line2	city	postcode	state
fe26ffdc-0944-4e90-846a-04b18367878f	add1	add1	BHN	2000	TN
#f6b07a7-ec24-4b65-91eb-b26cbb5824f	add1	add1	BHN	2000	TN
UUID-1	P.O. Box 49...	-12-24821, 8...	Vienna	80066	Wle...
UUID-10	3578 Neil St...	80-86218, 4...	Cork	67297	Mun...
UUID-100	8068 Pulvin...	4-15097, -72...	Mur...	88570-6...	PU
UUID-11	P.O. Box 77...	-79-27567, 3...	Bello	33457	Ant...
UUID-12	5865 Purus St...	-58-41035, ...	Gre...	817559	Ore...
UUID-13	665-7419 Sit...	-52-90714, 6...	Alhué	RD4 7PG	RM
UUID-14	P.O. Box 86...	16-44007, 2...	Cork	29966	M
UUID-15	512-2479 Sit...	52-92, -1.72...	Car...	45763	Bolivar

The 'Output' panel at the bottom shows the execution of the query, indicating that 254 rows were returned.

