Test Document

for

SafeSpace

Version 1.0

Prepared by

Group #: 10 **Group Name: 404 Team Not Found**

Anirudh Singh	230142	sanirudh23@iitk.ac.in
Archita Goyal	230187	architag23@iitk.ac.in
Brinda Fadadu	230307	brindaf23@iitk.ac.in
Nakul Patel	230676	nakulpatel23@iitk.ac.in
Naman Yadav	230680	namanyadav23@iitk.ac.in
Om Chaudhari	230715	omcc23@iitk.ac.in
Rohit Yadav	230873	rohity23@iitk.ac.in
Sayani Patra	230943	sayanip23@iitk.ac.in
Vivek	231169	vivek23@iitk.ac.in
Yatharth Sharma	231198	yatharths23@iitk.ac.in

Course: CS253

Mentor TA: Souvik Mukherjee

Date: 5 April 2025

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
1.0	Group 10 404 Team Not Found	Completed version of Test Document of the web app SafeSpace built by Group 10.	05/04/2025

1 Introduction

Test Strategy

SafeSpace is a confidential and supportive online platform designed exclusively for students at IITK. The platform aims to address students' mental health challenges and provide a safe environment to connect with peers and counselors from the Institute Counseling Service. Its primary objective is to foster mental wellness, open communication, and mutual understanding within the student community.

Our team's testing strategy for such an app with a plethora of options was manual testing wherein each of the testers would individually test the app for various inputs or cases and match the obtained output with the desired output.

The testers have subjected the app to different possible input combinations and verified the proper response of the app as and when required. The tests not only consisted of proper verification of output on the app (frontend), but also consisted of constant verification for the proper entries in the database (backend). This leads to a precise check of all the functional and non-functional requirements to be fulfilled by the app as mentioned in the SRS document.

Testing Period of the App

The testing was an integral part of the development process. During the implementation, the app was subjected to tests corresponding to basic functions such as logins, bookings, chattings, posting etc.

However, the majority of the testing of the app was done after the implementation of the app. Since the basic functionality of the app was set in stone and tested, any bugs encountered in the app were easier to tackle since they involved changes only in specific features of the app (certain functions of the source code). This made testing a lot more efficient and less time-consuming.

Testers of the App

The developers of the app were the testers themselves. The benefit of having the developers being the testers was that we knew the different possible inputs and hence were able to carry out extensive testing to ensure the robustness of the application. Multiple errors were identified and corrected as and when discovered.

Coverage Criterion for Testing

Functional requirements were the criterion used for the testing.

Tools used in testing

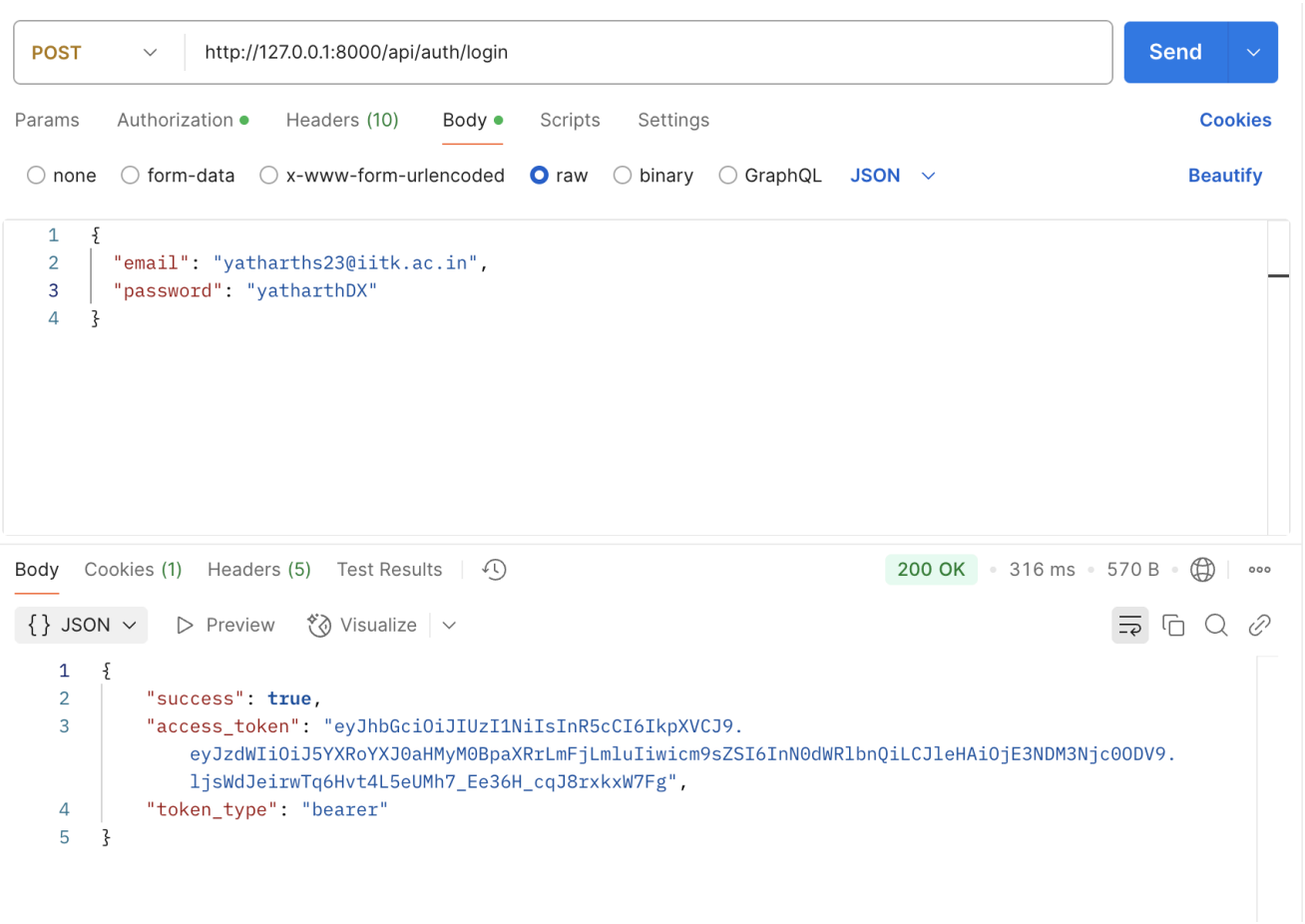
We used Postman for backend testing, a tool that enables testers to send HTTP requests to the server and view responses in a unified interface. It was utilized to test all backend endpoints.. The entire frontend was manually tested.

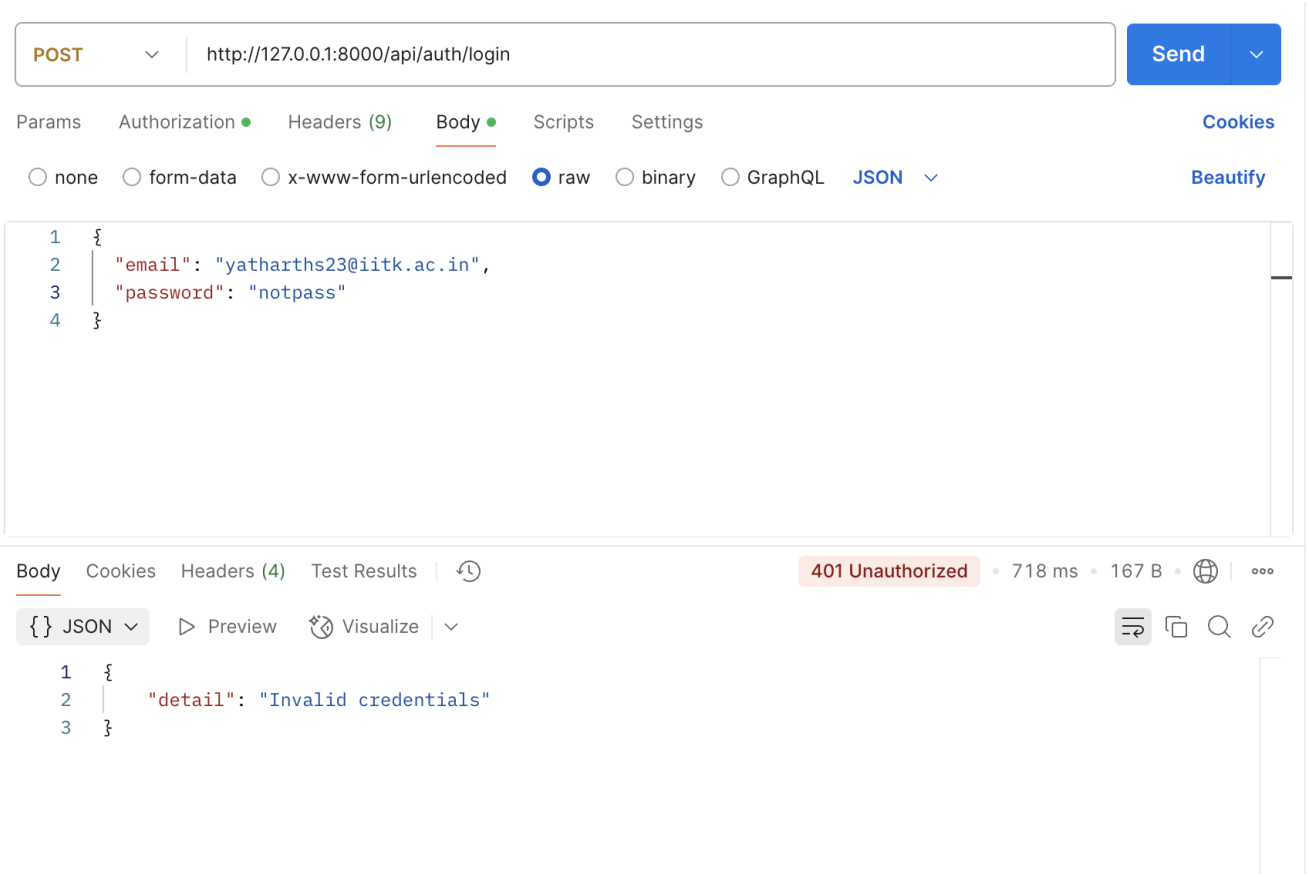
2 Unit Testing

1. POST /login

The backend function for login was tested. All the testable branches functioned as expected.

Unit Details: Backend function to handle Post Request “/login”
Test Owner: Yatharth Sharma
Test Date: [03/26/2025] - [03/26/2025]
Test Results:
Correct Username and Password - “Login Successful”
Incorrect Username or Password - “Invalid credentials”
Structural Coverage: Branch Coverage 100%





2. POST /register

The backend function for registering was tested. All the testable branches functioned as expected.

Unit Details: Backend function to handle Post Request “/register”

Test Owner: Yatharth Sharma

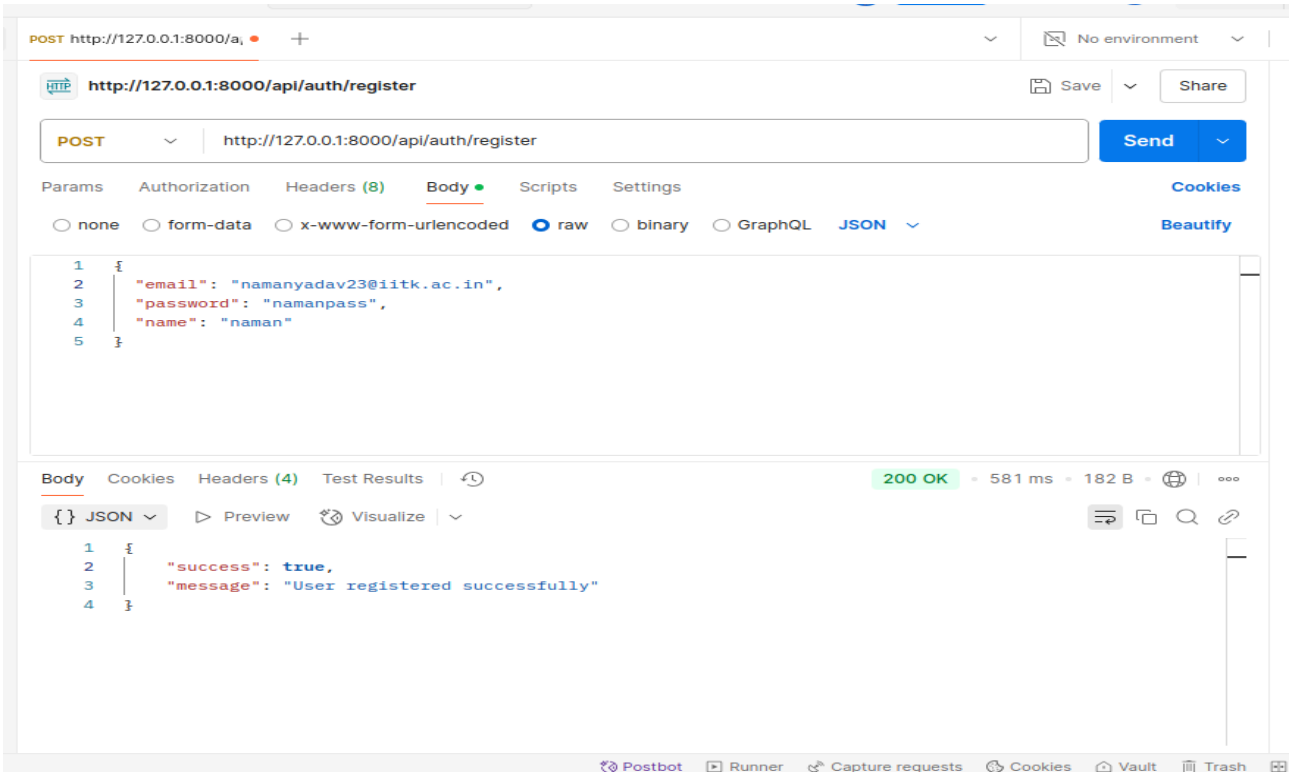
Test Date: [03/26/2025] - [03/26/2025]

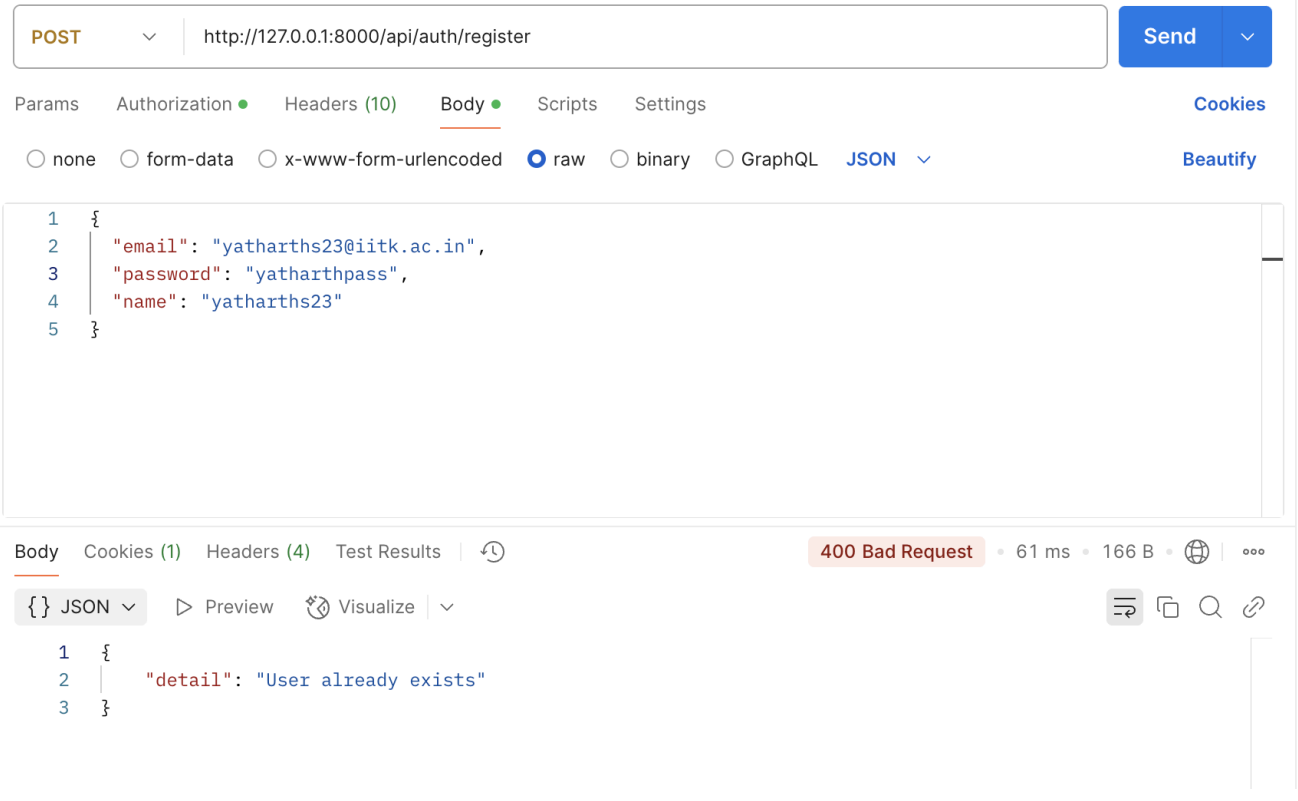
Test Results:

Existing User - “User already exists”

New Username - “User registered successfully”

Structural Coverage: Branch Coverage 100%



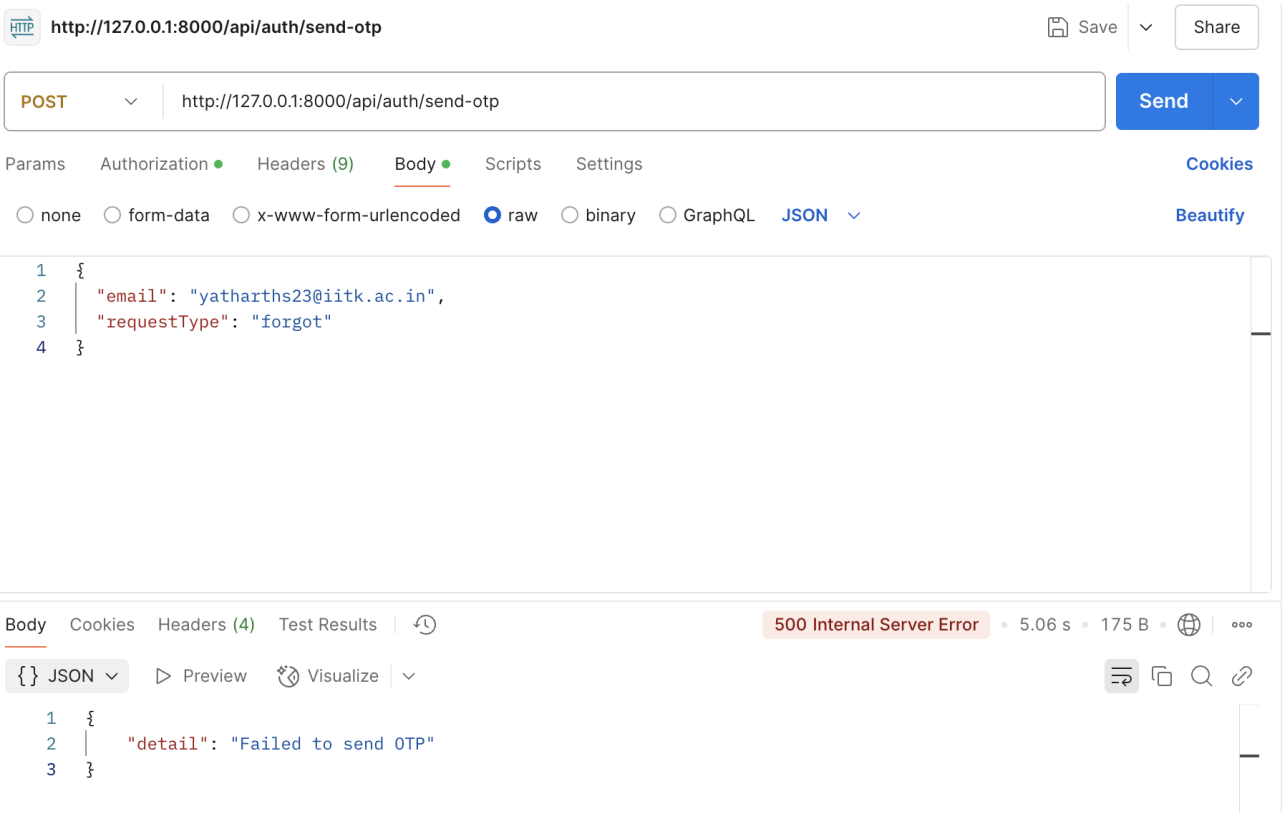


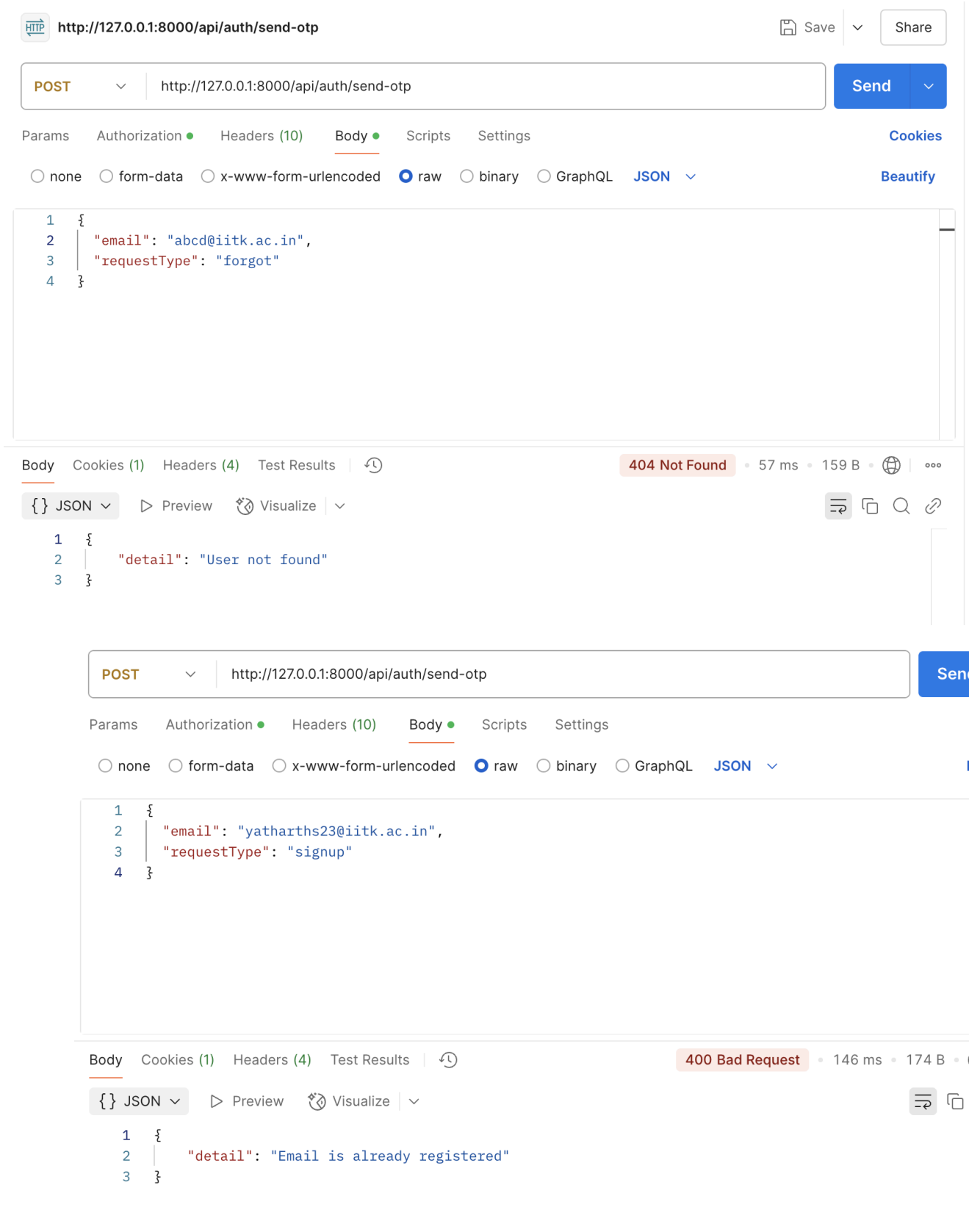
3. POST /send-otp

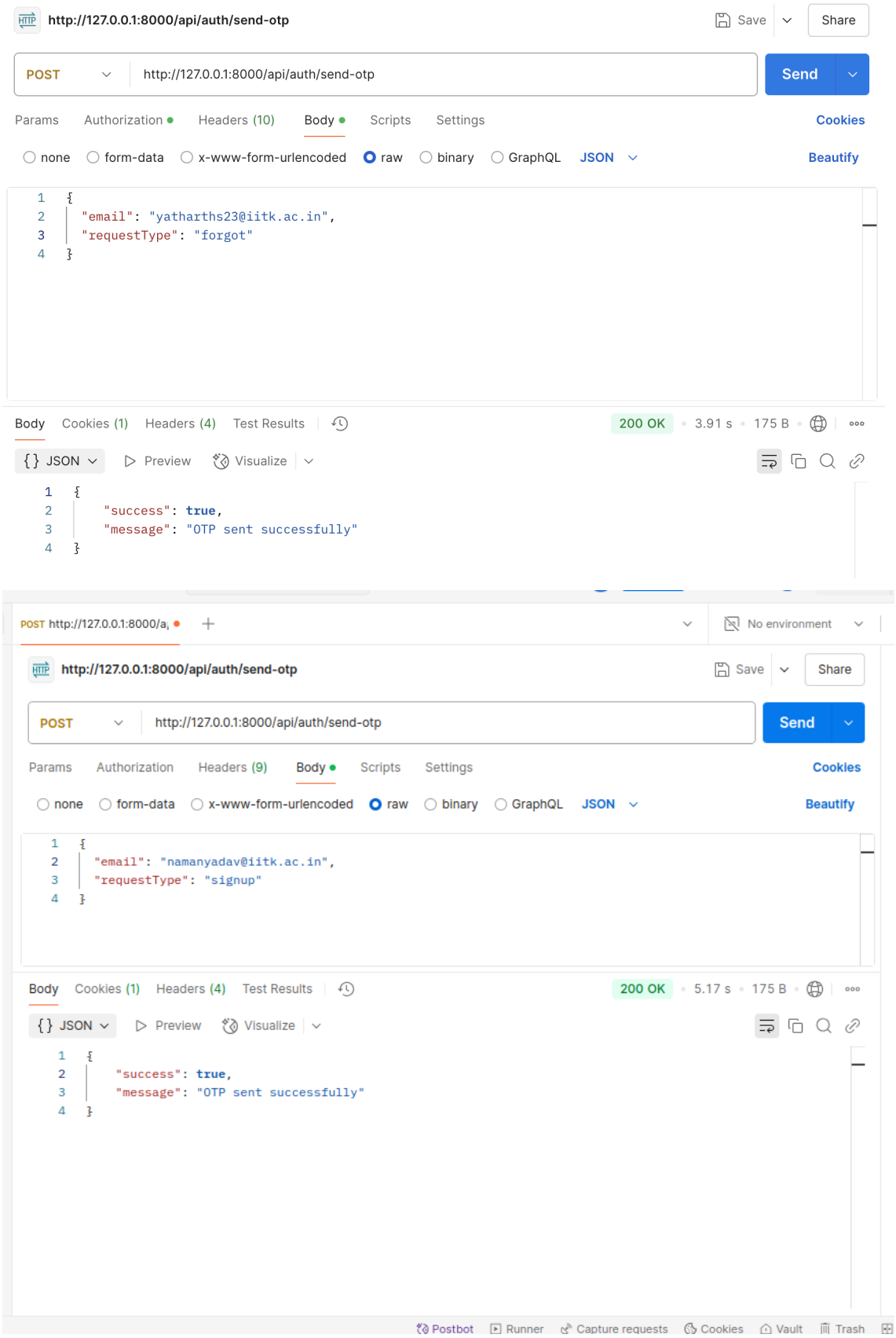
The backend function for send-otp was tested. All the testable branches functioned as

expected.

Unit Details: Backend function to handle Post Request “/send-otp”
Test Owner: Sayani Patra
Test Date: [03/26/2025] - [03/26/2025]
Test Results:
Request for registering, with already existing registered email - “Email is already registered”
Request for forgot password (reset password), with email not already registered email - “User not found”
Email credential of admin is wrong, failed to send opt - “Failed to send OTP”
Sent OTP for signup - “OTP sent successfully”
Sent OTP for forgot - “OTP sent successfully”
Structural Coverage: Branch Coverage 100%
If we try to send otp with incorrect admin privileges it leads to failure



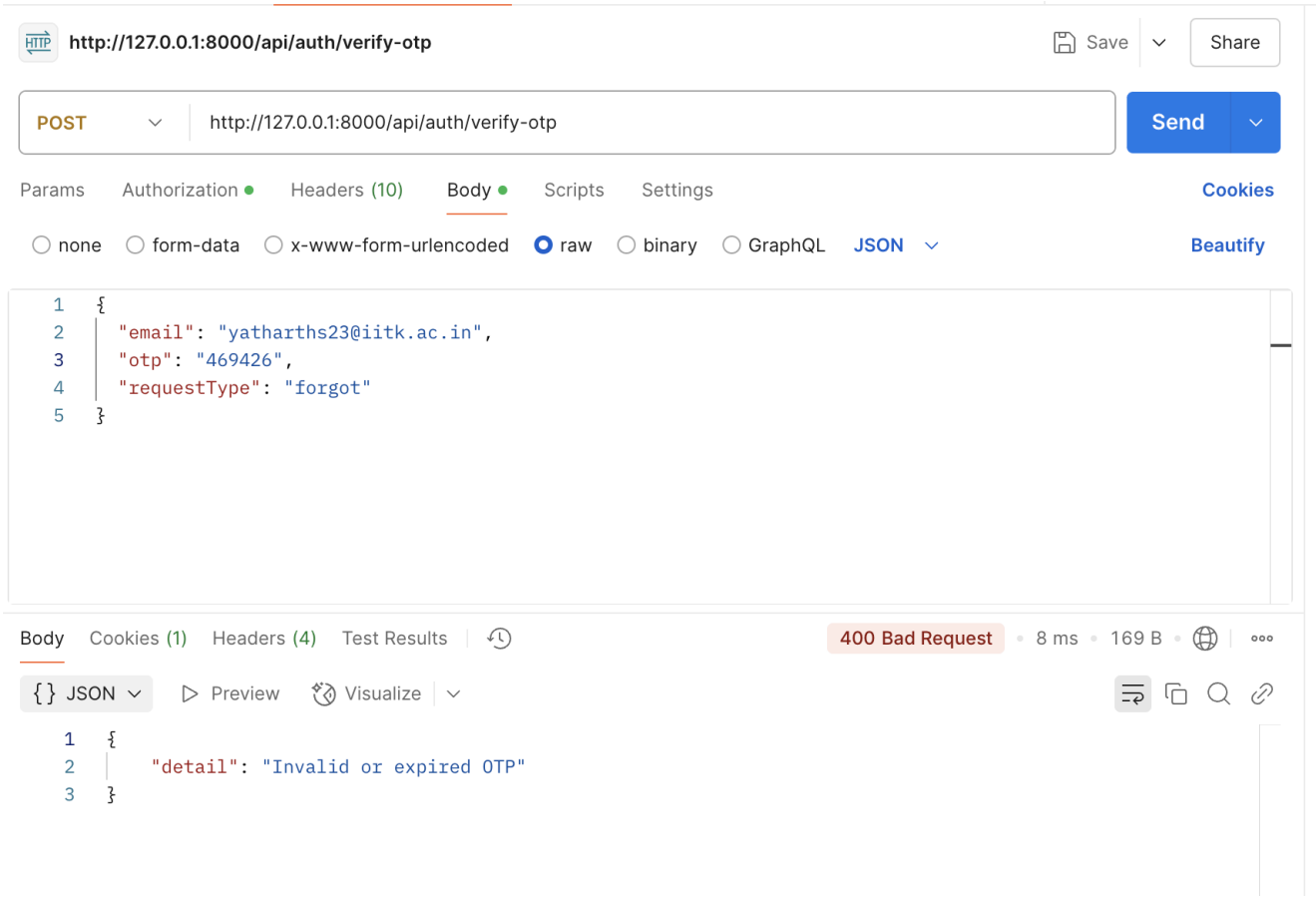

http://127.0.0.1:8000/api/auth/send-otp
Save
Share
POST
http://127.0.0.1:8000/api/auth/send-otp
Send
Params
Authorization
Headers (10)
Body
Scripts
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"email": "abcd@iitk.ac.in",
"requestType": "forgot"
}
Body
Cookies (1)
Headers (4)
Test Results
404 Not Found
57 ms
159 B
JSON
Preview
Visualize
{
"detail": "User not found"
}
POST
http://127.0.0.1:8000/api/auth/send-otp
Send
Params
Authorization
Headers (10)
Body
Scripts
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"email": "yatharths23@iitk.ac.in",
"requestType": "signup"
}
Body
Cookies (1)
Headers (4)
Test Results
400 Bad Request
146 ms
174 B
JSON
Preview
Visualize
{
"detail": "Email is already registered"
}

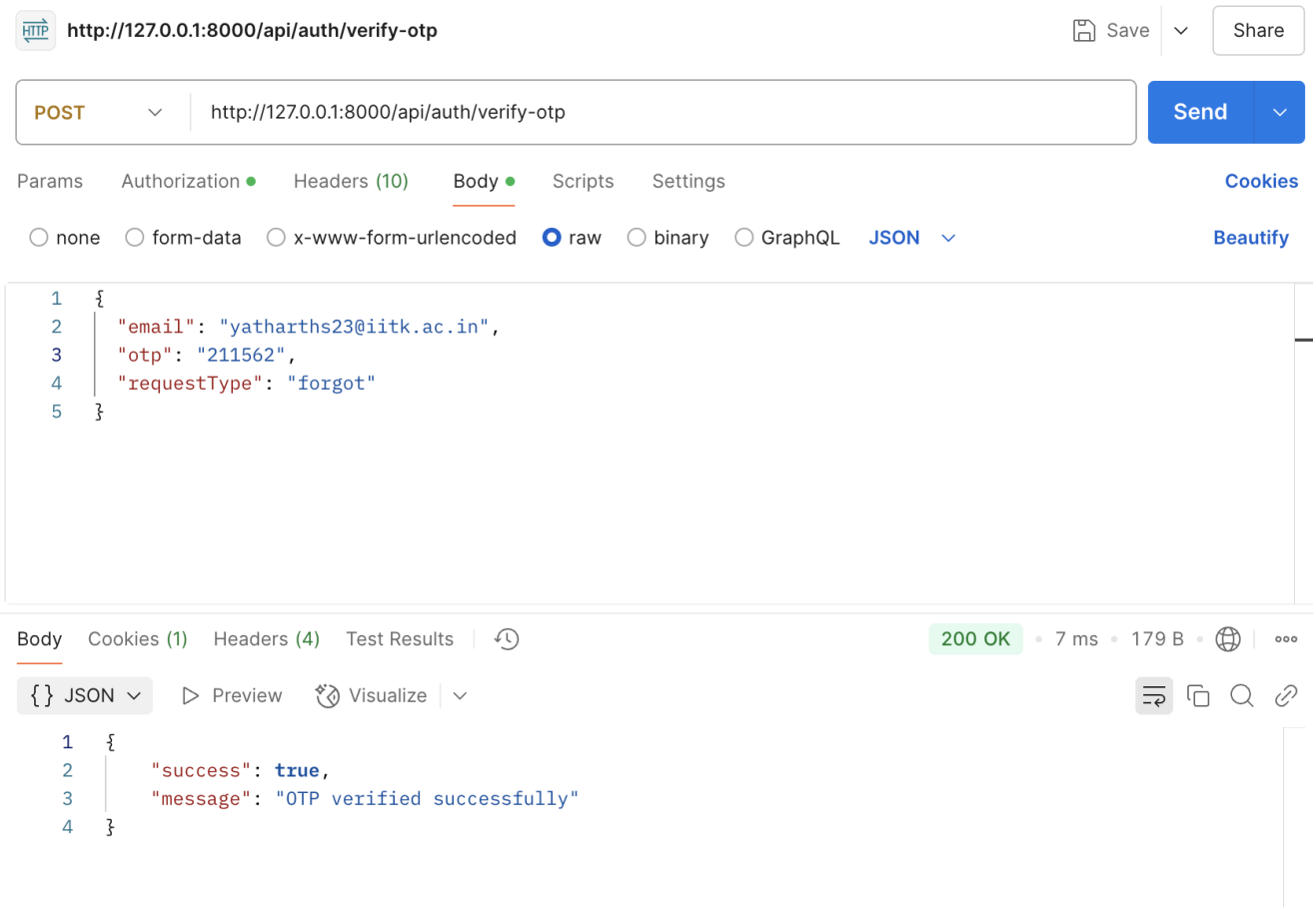

http://127.0.0.1:8000/api/auth/send-otp
Save
Share
POST
http://127.0.0.1:8000/api/auth/send-otp
Send
Params
Authorization
Headers (10)
Body
Scripts
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"email": "yatharths23@iitk.ac.in",
"requestType": "forgot"
}
Body
Cookies (1)
Headers (4)
Test Results
200 OK
3.91 s
175 B
JSON
Preview
Visualize
{
"success": true,
"message": "OTP sent successfully"
}
POST http://127.0.0.1:8000/a
No environment
http://127.0.0.1:8000/api/auth/send-otp
Save
Share
POST
http://127.0.0.1:8000/api/auth/send-otp
Send
Params
Authorization
Headers (9)
Body
Scripts
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"email": "namanyadav@iitk.ac.in",
"requestType": "signup"
}
Body
Cookies (1)
Headers (4)
Test Results
200 OK
5.17 s
175 B
JSON
Preview
Visualize
{
"success": true,
"message": "OTP sent successfully"
}
Postbot
Runner
Capture requests
Cookies
Vault
Trash

4. POST /verify-otp

The backend function for verify-otp was tested. All the testable branches functioned as expected.

Unit Details: Backend function to handle Post Request “/verify-otp”
Test Owner: Anirudh Singh
Test Date: [03/26/2025] - [03/26/2025]
Test Results:
Wrong OTP or submitting OTP after 10 minutes - “Invalid or expired OTP”
OTP verified - “OTP verified successfully”
Structural Coverage: Branch Coverage 100%

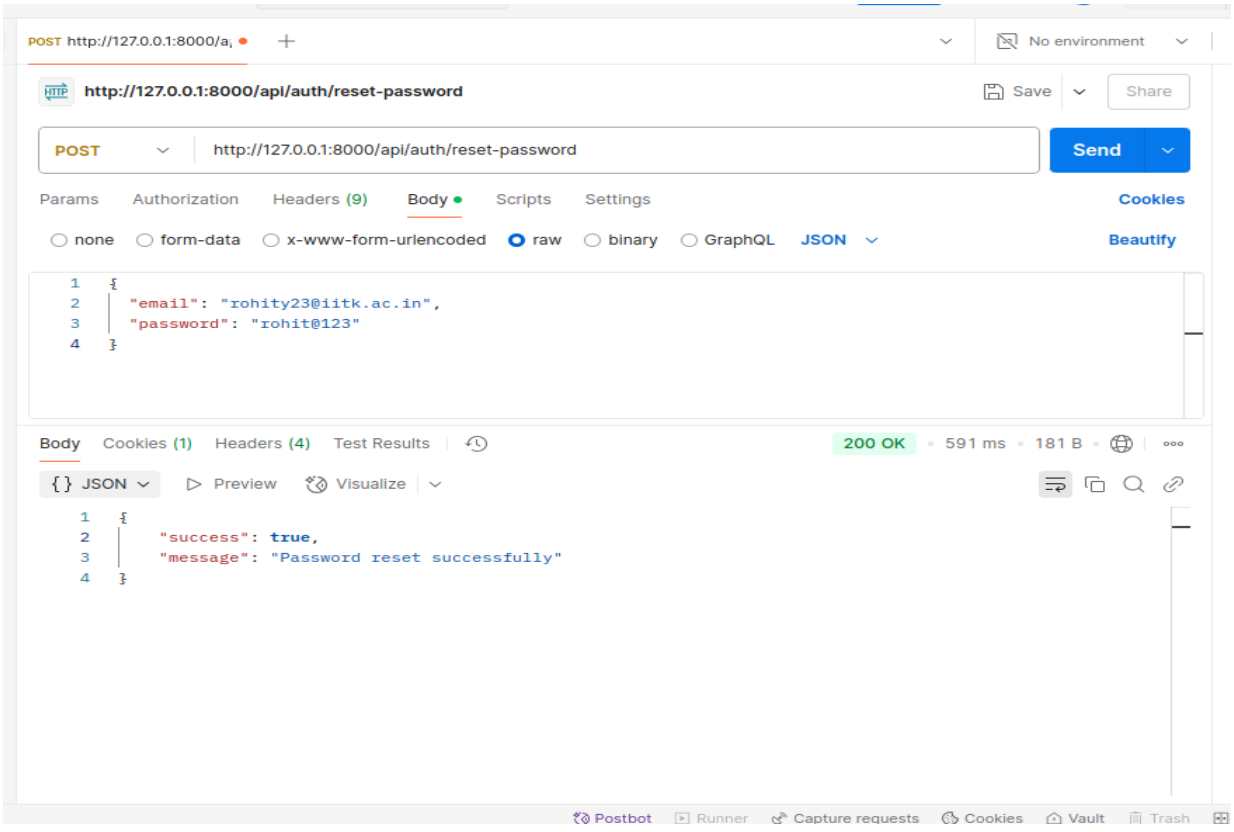


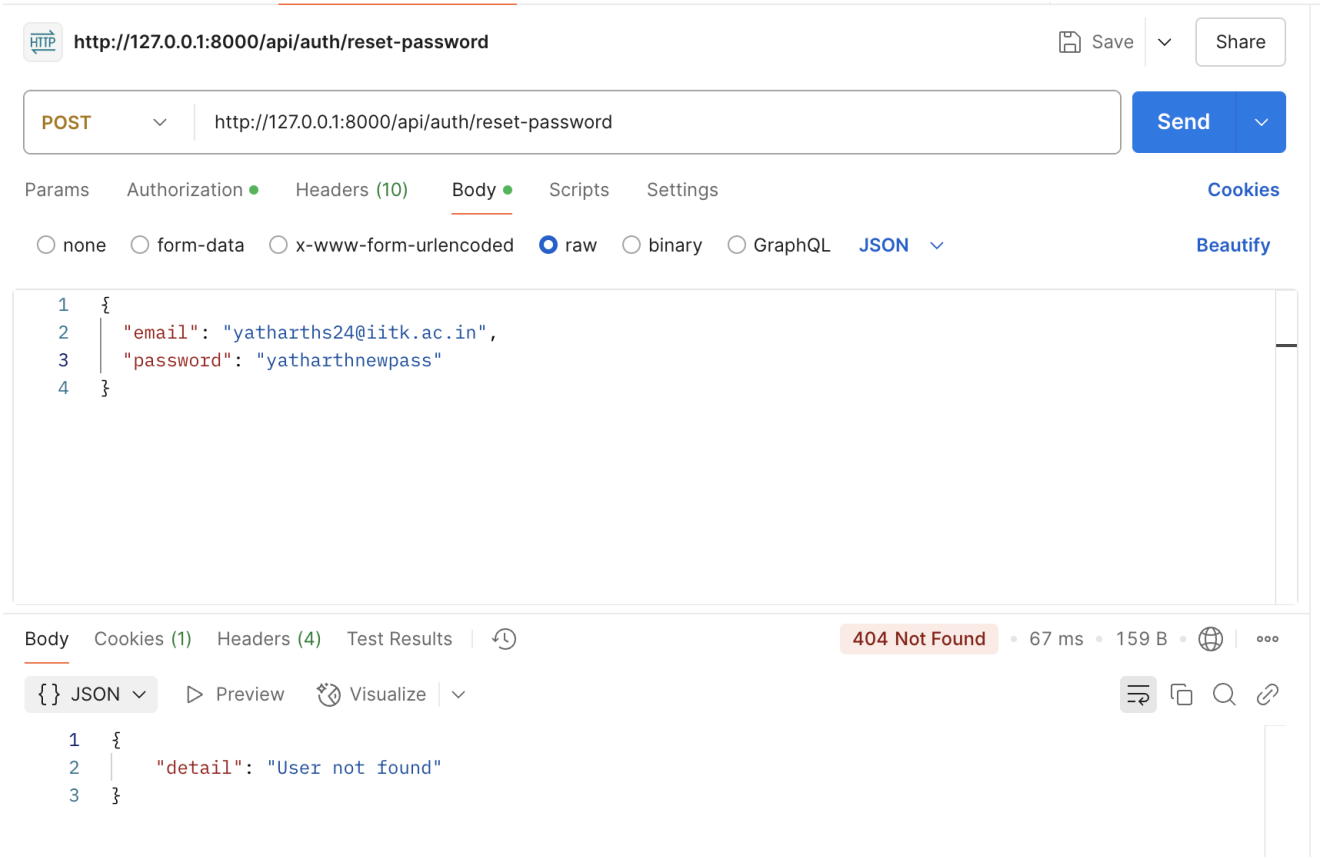


5. POST /reset-password

The backend function for reset-password was tested. All the testable branches functioned as expected.

Unit Details: Backend function to handle Post Request “/reset-password”
Test Owner: Rohit Yadav
Test Date: [03/26/2025] - [03/26/2025]
Test Results:
Invalid User - “User not found”
Reset Password - “Password reset successfully”
Structural Coverage: Branch Coverage 100%

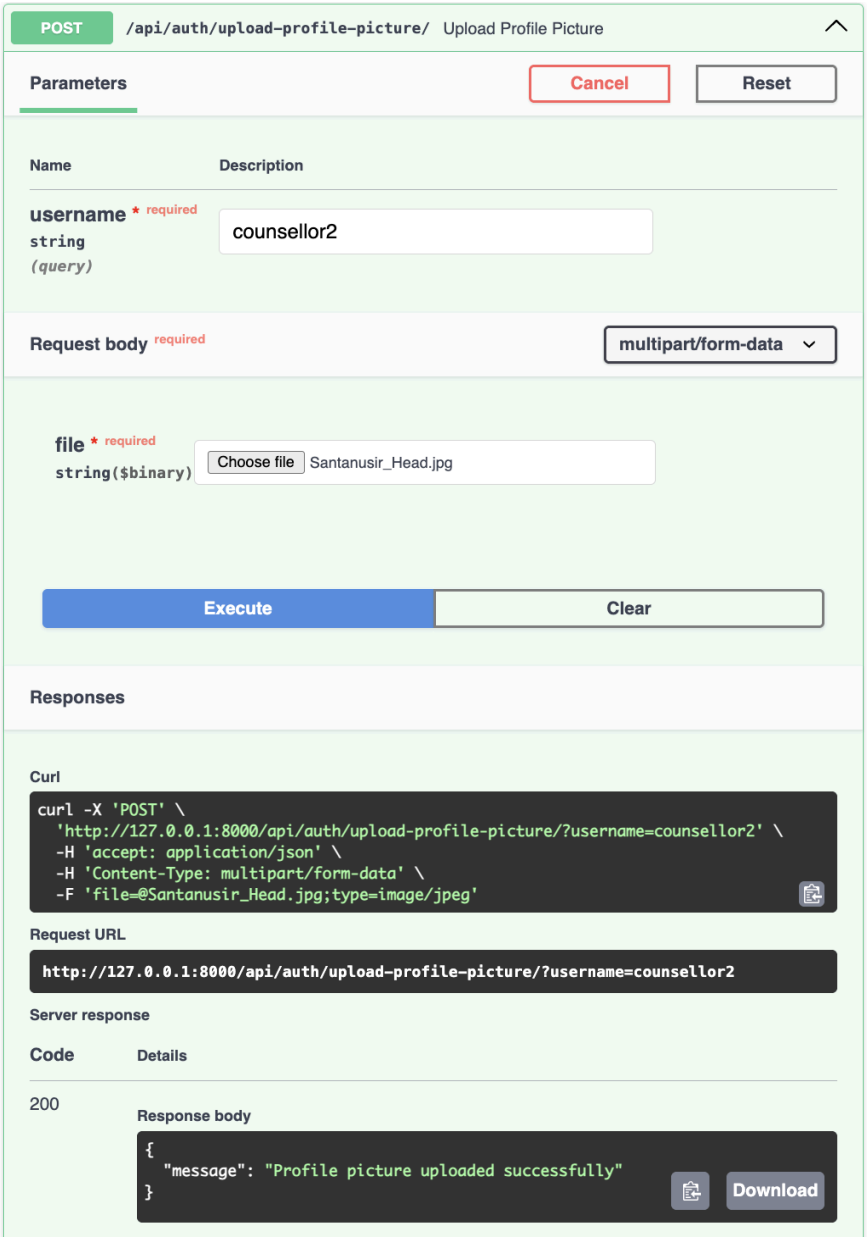

POST http://127.0.0.1:8000/a
No environment
http://127.0.0.1:8000/api/auth/reset-password
Save
Share
POST
http://127.0.0.1:8000/api/auth/reset-password
Send
Params
Authorization
Headers (9)
Body
Scripts
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"email": "rohity23@iitk.ac.in",
"password": "rohit@123"
}
Body
Cookies (1)
Headers (4)
Test Results
200 OK
591 ms
181 B
JSON
Preview
Visualize
{
"success": true,
"message": "Password reset successfully"
}
Postbot
Runner
Capture requests
Cookies
Vault
Trash


http://127.0.0.1:8000/api/auth/reset-password
Save
Share
POST
http://127.0.0.1:8000/api/auth/reset-password
Send
Params
Authorization
Headers (10)
Body
Scripts
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"email": "yatharths24@iitk.ac.in",
"password": "yatharthnewpass"
}
Body
Cookies (1)
Headers (4)
Test Results
404 Not Found
67 ms
159 B
JSON
Preview
Visualize
{
"detail": "User not found"
}

6. POST /upload-profile-picture

The backend function for uploading the profile picture of a user was tested. All the testable branches functioned as expected.

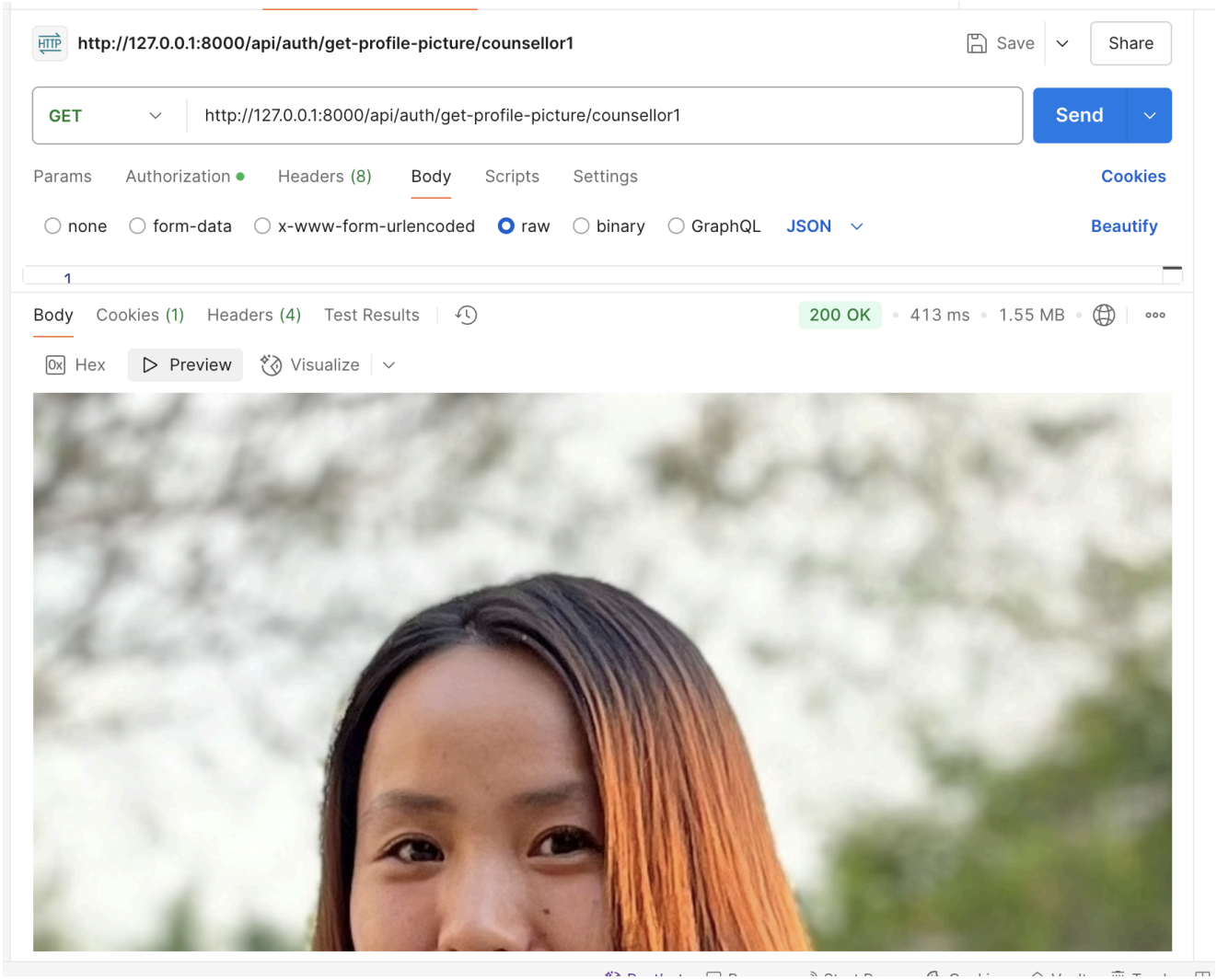
Unit Details: Backend function to handle Post Request “/upload-profile-picture”
Test Owner: Yatharth Sharma
Test Date: [03/26/2025] - [03/26/2025]
Test Results:
Uploaded correct format image - “Profile picture uploaded successfully”
Structural Coverage: Branch Coverage 100%



7. GET /get-profile-picture/{username}

The backend function to get the profile picture of a particular user was tested. All the testable branches functioned as expected.

Unit Details: Backend function to handle Get Request "/get-profile-picture/{username}"
Test Owner: Archita Goyal
Test Date: [03/26/2025] - [03/26/2025]
Test Results:
No profile picture uploaded for the user - "No profile picture found"
Authorised user - "Fetched profile picture successfully"
Structural Coverage: Branch Coverage 100%

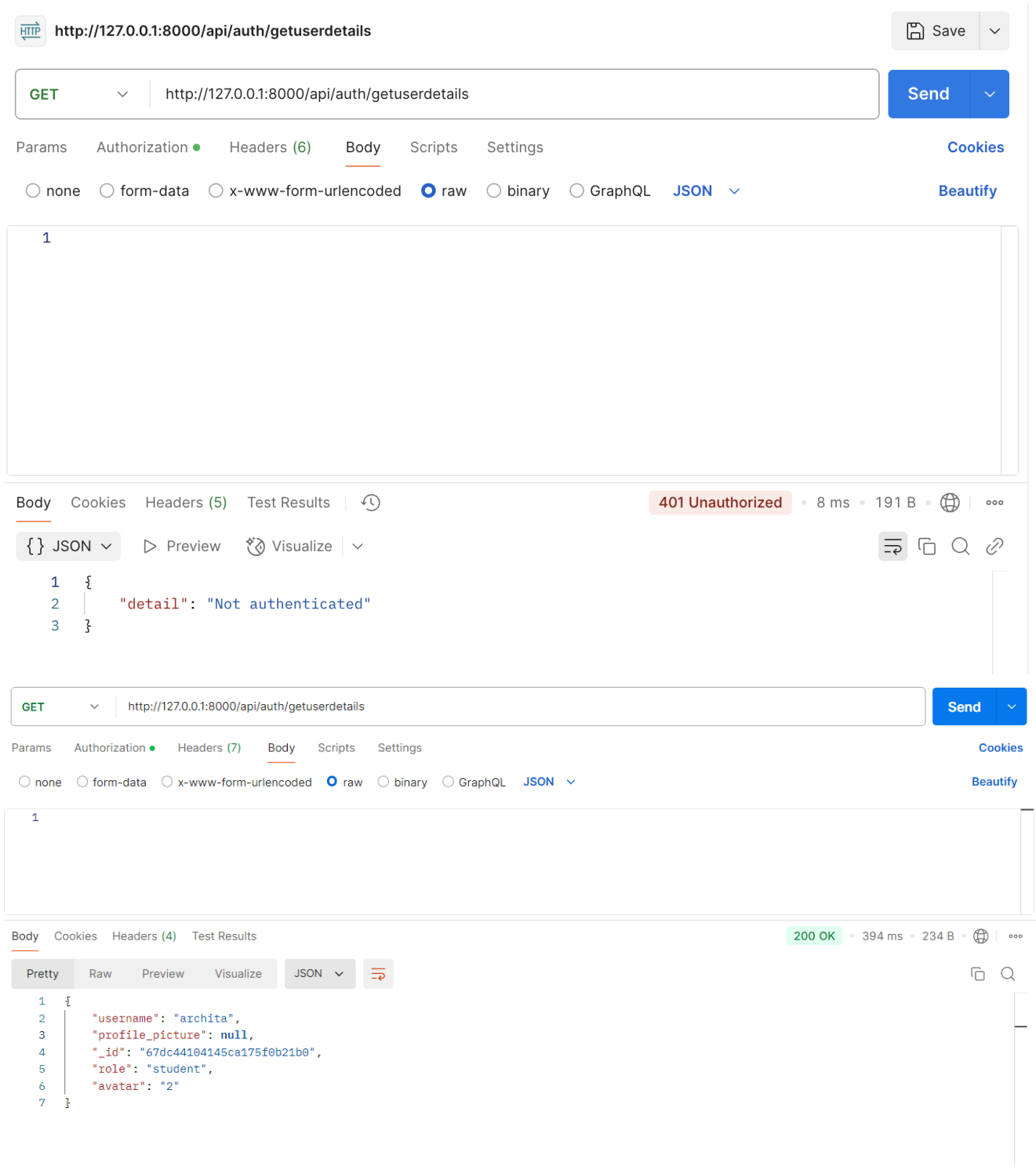


8. GET /getuserdetails

The backend function to get the details of the current user was tested. All the testable branches functioned as expected.

Unit Details: Backend function to handle Get Request "/get-user-details"

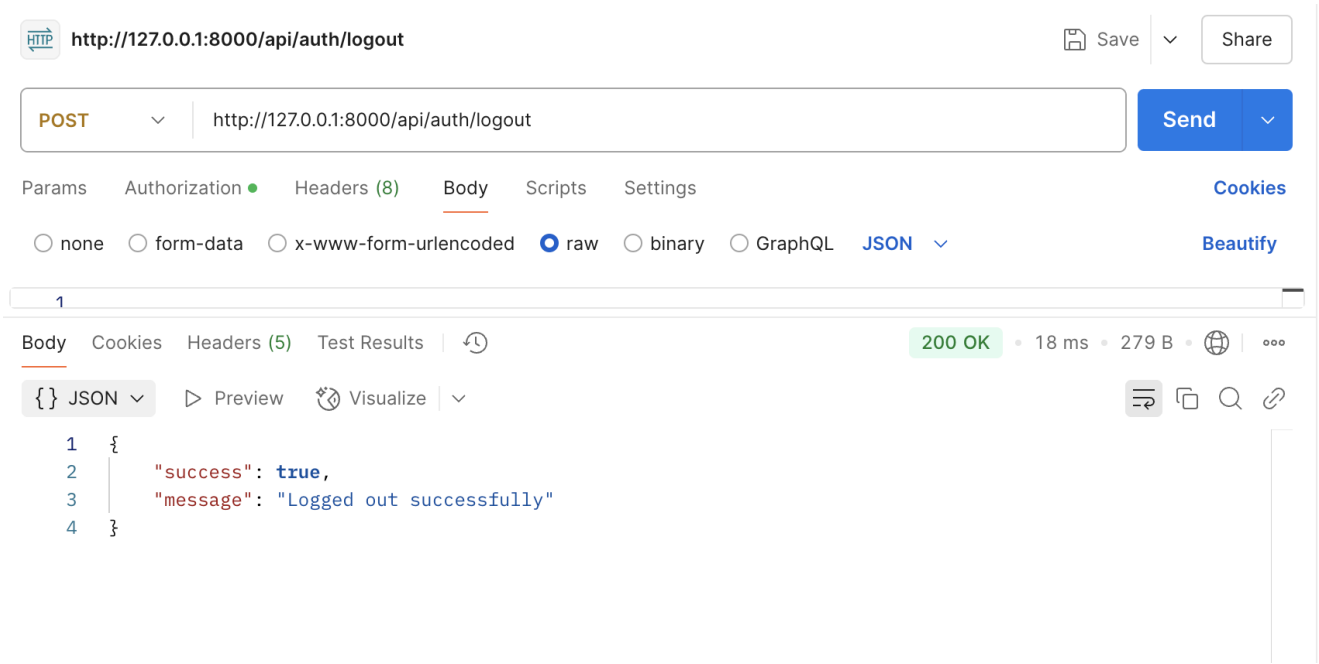
Test Owner: Archita Goyal
Test Date: [03/26/2025] - [03/26/2025]
Test Results:
Unauthorised User - “Unauthorised User”
Authorised User - “Fetched user details successfully”
Structural Coverage: Branch Coverage 100%



9. POST /logout

The backend function for logout was tested. All the testable branches functioned as expected.

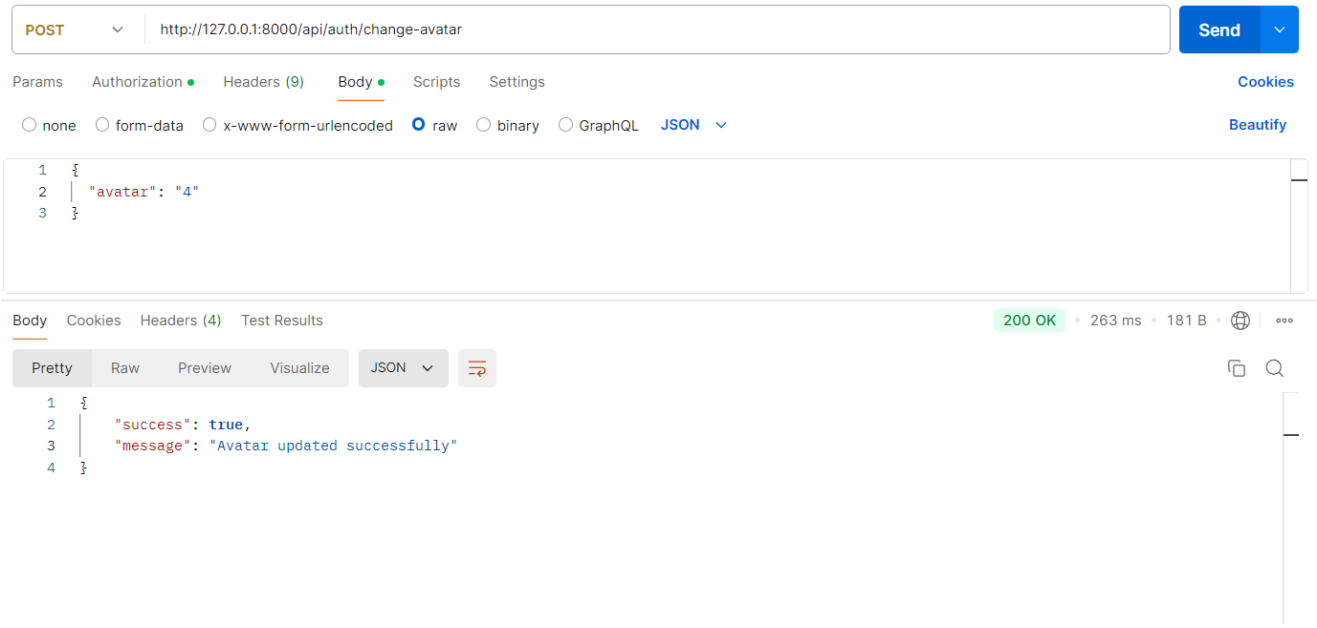
Unit Details: Backend function to handle Post Request “/logout”
Test Owner: Om Chaudhari
Test Date: [03/26/2025] - [03/26/2025]
Test Results:
Cookies updated (JWT token removed) - “Logged out successfully”
Structural Coverage: Branch Coverage 100%



10. POST /change-avatar

The backend function to change the avatar of the current user was tested. All the testable branches functioned as expected.

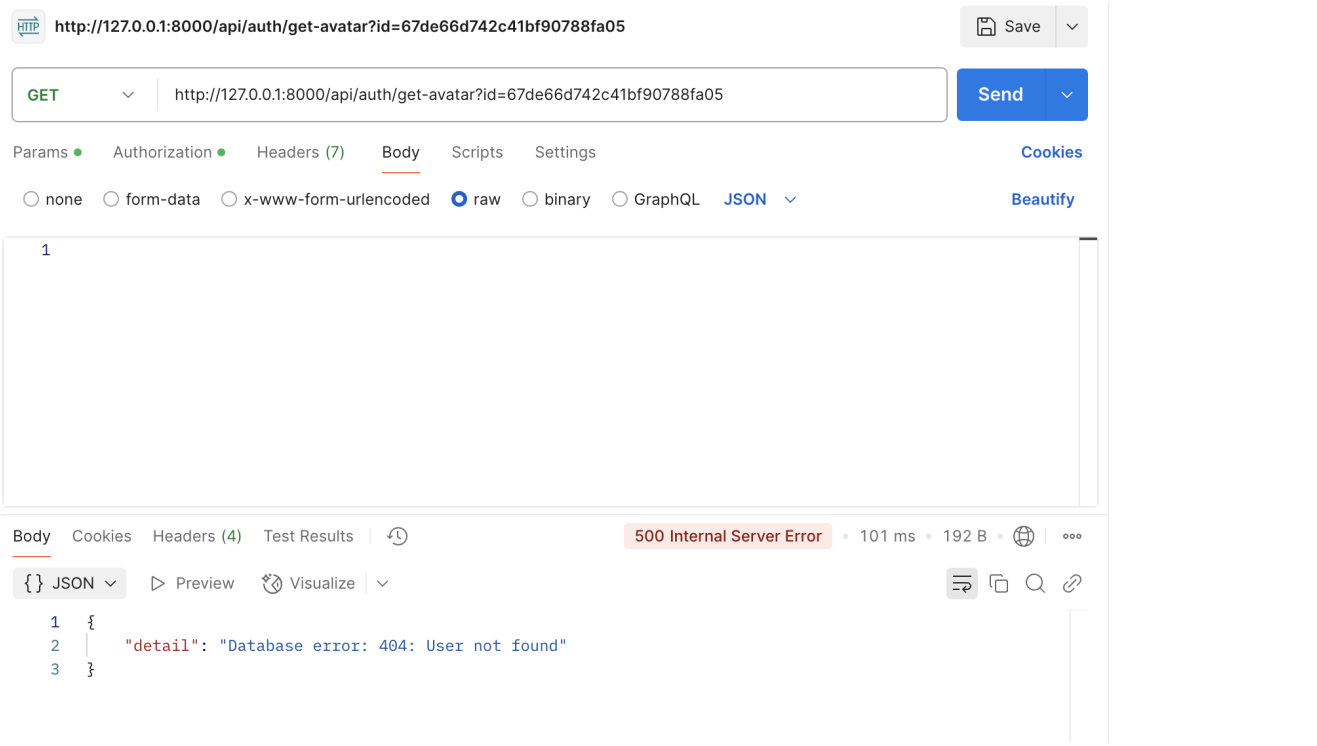
Unit Details: Backend function to handle Post Request “/change-avatar”
Test Owner: Yatharth Sharma
Test Date: [03/26/2025] - [03/26/2025]
Test Results:
No change - “User not found or avatar not changed”
Authorised user - “Avatar changes successfully”
Structural Coverage: Branch Coverage 100%



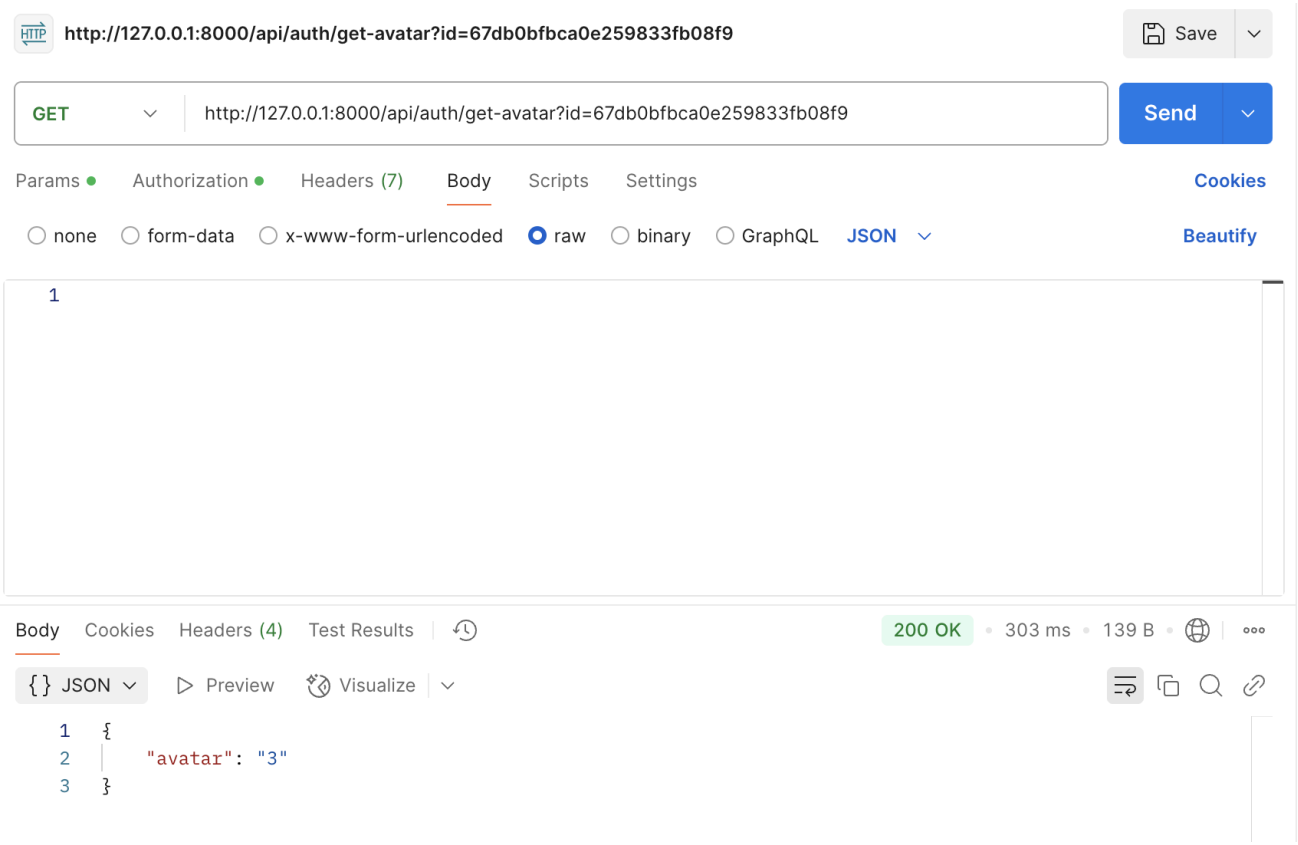
11. GET /get-avatar

The backend function to get the avatar of a particular user was tested. All the testable branches functioned as expected.

Unit Details: Backend function to handle Get Request “/get-avatar”
Test Owner: Archita Goyal
Test Date: [03/26/2025] - [03/26/2025]
Test Results:
User whose avatar is accessed is not found - “User not found”



Authorised user - “Fetched the avatar of the current user”

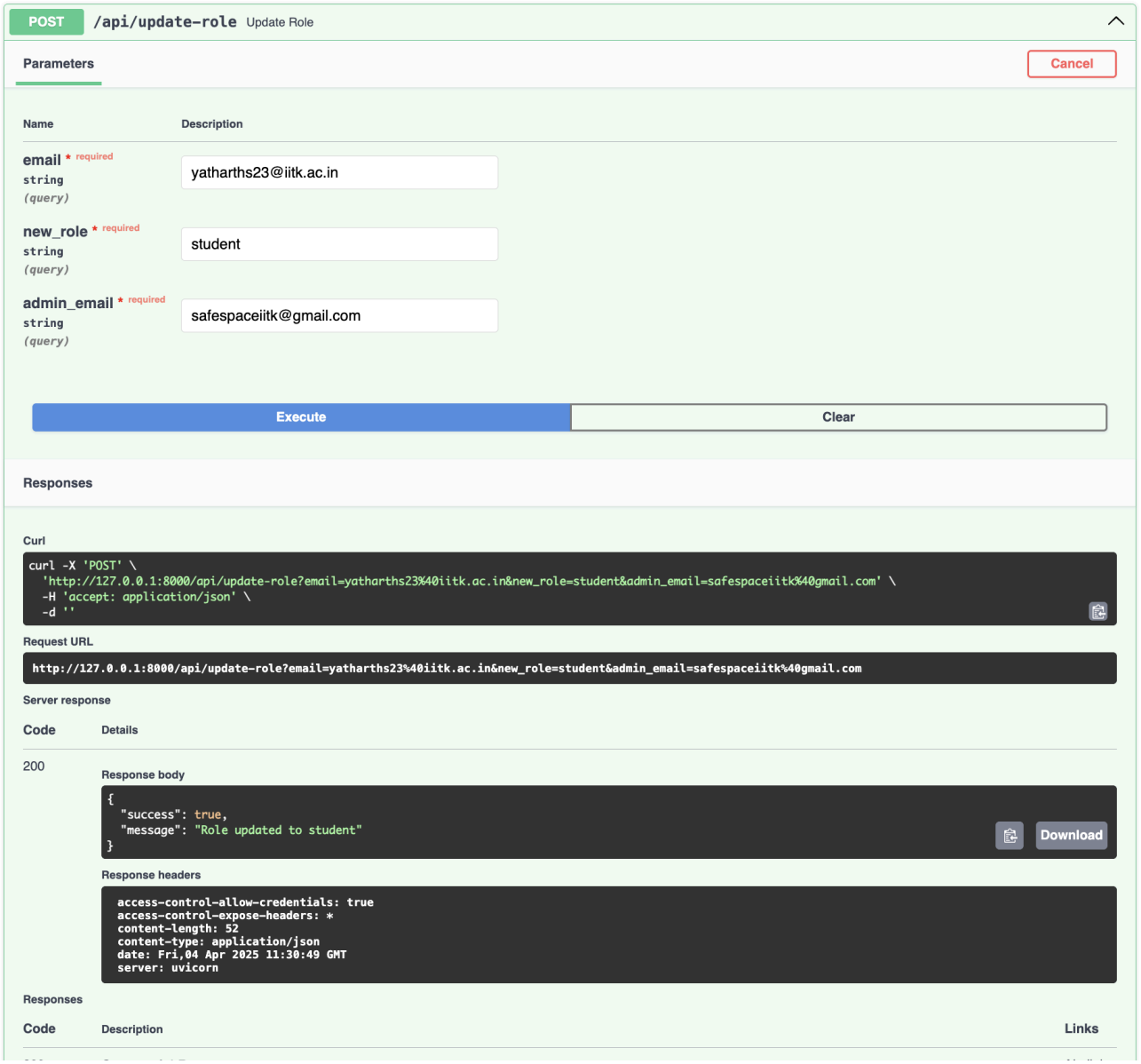


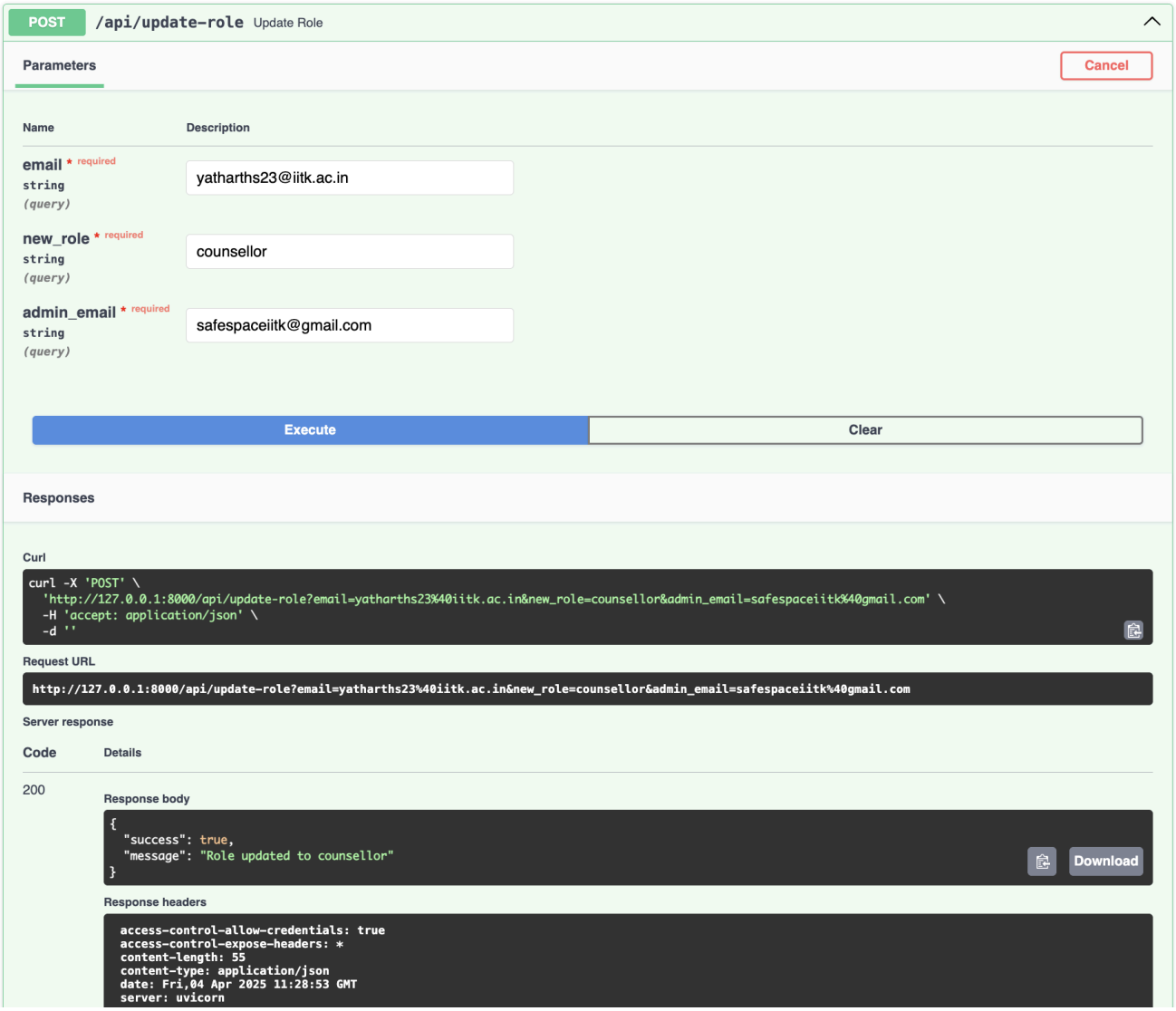
Structural Coverage: Branch Coverage 100%

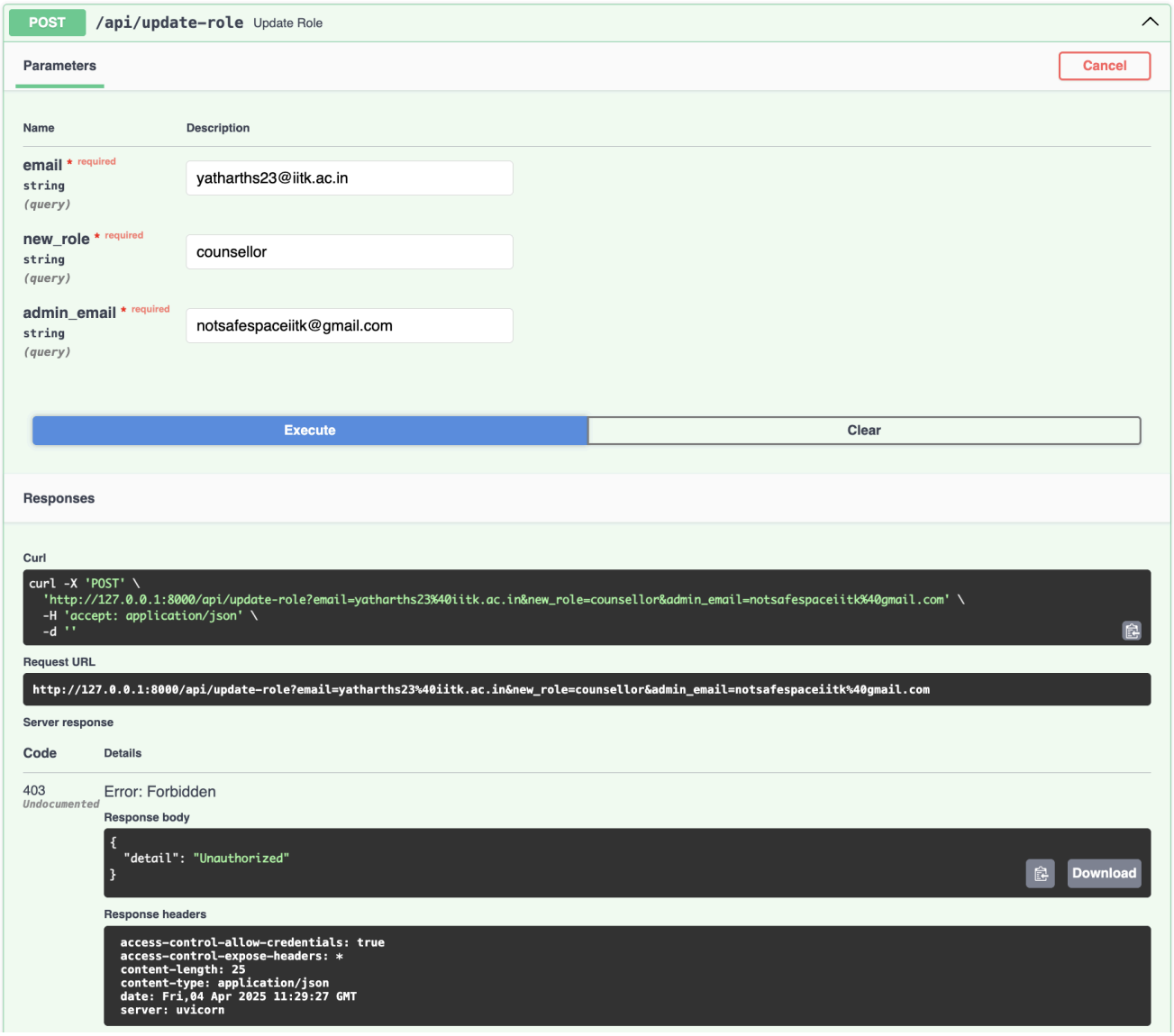
12. POST /update-role

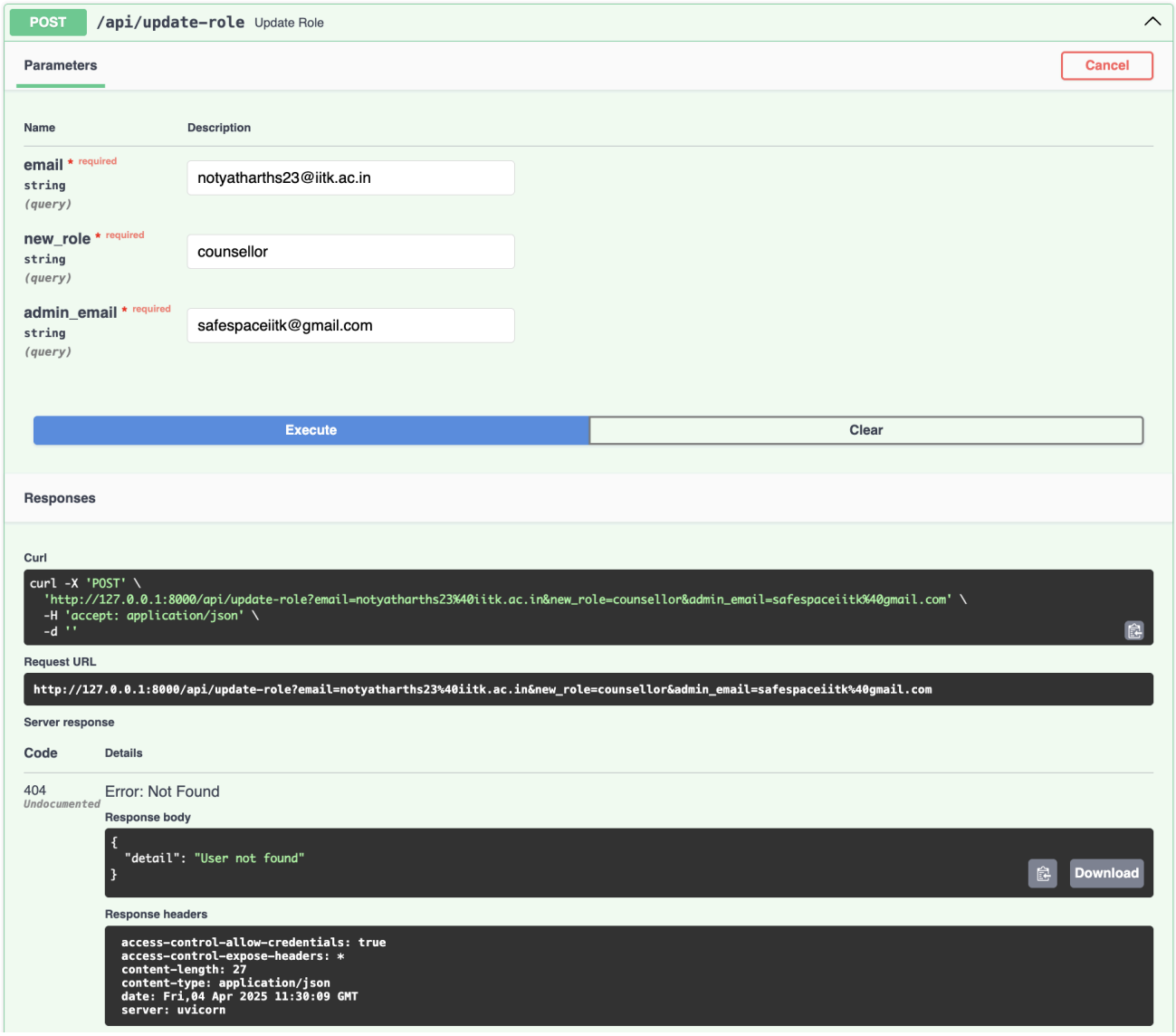
The backend function to change the role of a particular user was tested. All the testable branches functioned as expected.

Unit Details: Backend function to handle Post Request “/update-role”
Test Owner: Yatharth Sharma
Test Date: [03/26/2025] - [03/26/2025]
Test Results:
Unauthorised - “Unauthorised”
User not found - “User not found”
Admin request for change of role of a user to counsellor - “Role updated to counsellor”
Admin request for change of role of a user to student - “Role updated to student”
Structural Coverage: Branch Coverage 100%


POST
/api/update-role
Update Role
Parameters
Cancel
Name
Description
email * required
string
(query)
yatharths23@iitk.ac.in
new_role * required
string
(query)
student
admin_email * required
string
(query)
safespaceiitk@gmail.com
Execute
Clear
Responses
Curl
curl -X 'POST' \
'http://127.0.0.1:8000/api/update-role?email=yatharths23%40iitk.ac.in&new_role=student&admin_email=safespaceiitk%40gmail.com' \
-H 'accept: application/json' \
-d ''
Request URL
http://127.0.0.1:8000/api/update-role?email=yatharths23%40iitk.ac.in&new_role=student&admin_email=safespaceiitk%40gmail.com
Server response
Code
Details
200
Response body
{
"success": true,
"message": "Role updated to student"
}
Download
Response headers
access-control-allow-credentials: true
access-control-expose-headers: *
content-length: 52
content-type: application/json
date: Fri,04 Apr 2025 11:30:49 GMT
server: uvicorn
Responses
Code
Description
Links


POST
/api/update-role
Update Role
Parameters
Cancel
Name
Description
email * required
string
(query)
yatharths23@iitk.ac.in
new_role * required
string
(query)
counsellor
admin_email * required
string
(query)
safespaceiitk@gmail.com
Execute
Clear
Responses
Curl
curl -X 'POST' \
'http://127.0.0.1:8000/api/update-role?email=yatharths23%40iitk.ac.in&new_role=counsellor&admin_email=safespaceiitk%40gmail.com' \
-H 'accept: application/json' \
-d ''
Request URL
http://127.0.0.1:8000/api/update-role?email=yatharths23%40iitk.ac.in&new_role=counsellor&admin_email=safespaceiitk%40gmail.com
Server response
Code
Details
200
Response body
{
"success": true,
"message": "Role updated to counsellor"
}
Download
Response headers
access-control-allow-credentials: true
access-control-expose-headers: *
content-length: 55
content-type: application/json
date: Fri,04 Apr 2025 11:28:53 GMT
server: uvicorn

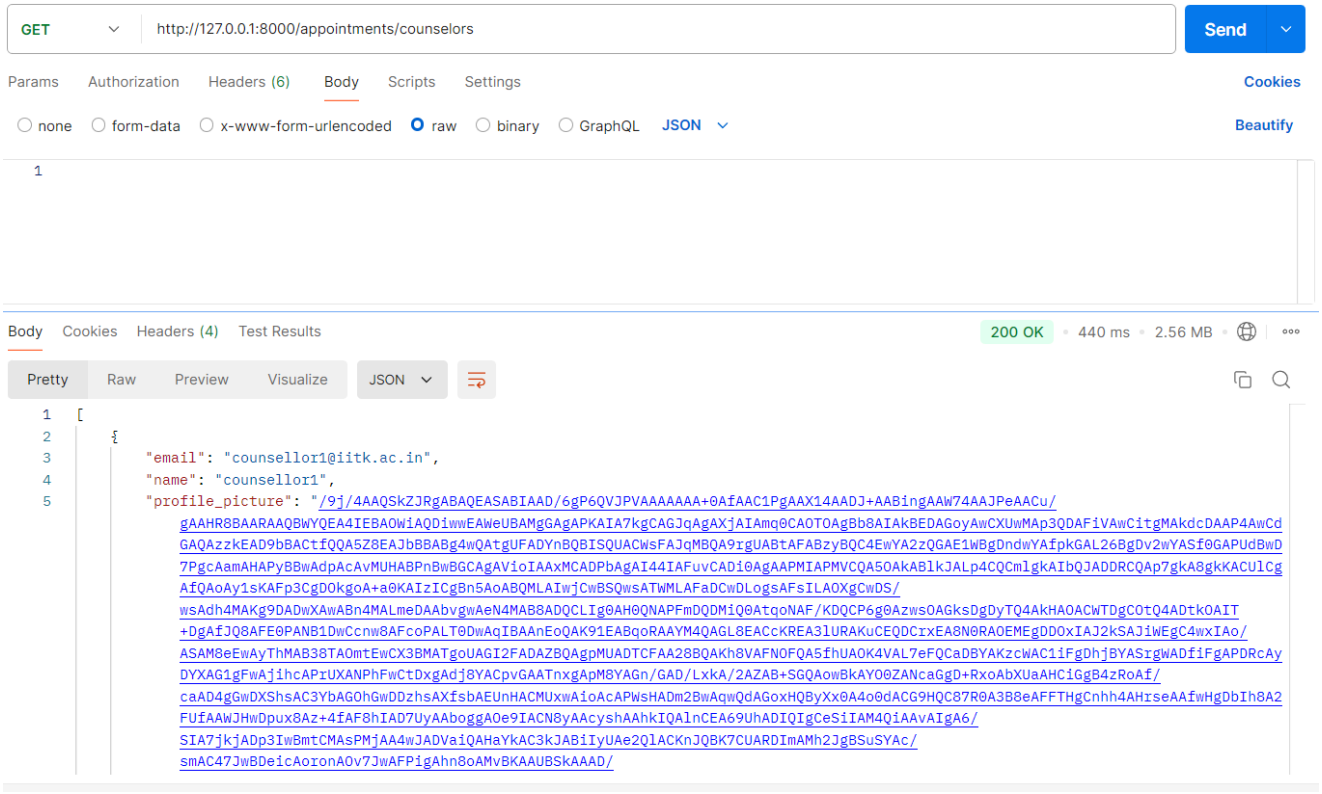

POST
/api/update-role
Update Role
Parameters
Cancel
Name
Description
email * required
string
(query)
yatharths23@iitk.ac.in
new_role * required
string
(query)
counsellor
admin_email * required
string
(query)
notsafespaceiitk@gmail.com
Execute
Clear
Responses
Curl
curl -X 'POST' \
'http://127.0.0.1:8000/api/update-role?email=yatharths23%40iitk.ac.in&new_role=counsellor&admin_email=notsafespaceiitk%40gmail.com' \
-H 'accept: application/json' \
-d ''
Request URL
http://127.0.0.1:8000/api/update-role?email=yatharths23%40iitk.ac.in&new_role=counsellor&admin_email=notsafespaceiitk%40gmail.com
Server response
Code
Details
403
Undocumented
Error: Forbidden
Response body
{
"detail": "Unauthorized"
}
Download
Response headers
access-control-allow-credentials: true
access-control-expose-headers: *
content-length: 25
content-type: application/json
date: Fri,04 Apr 2025 11:29:27 GMT
server: uvicorn



13. GET /counselors

The backend function to get the list of all the counsellors was tested. All the testable branches functioned as expected.

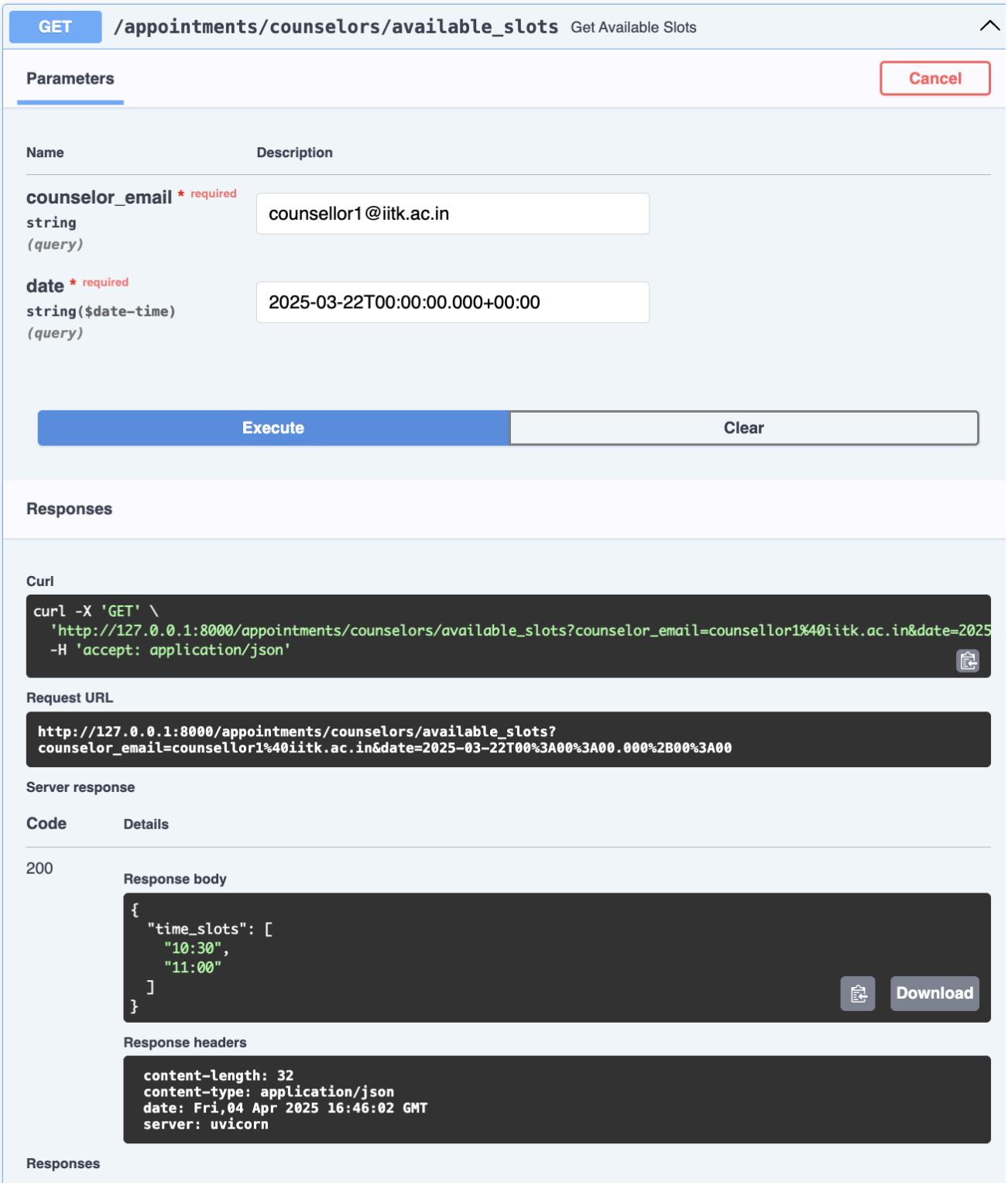
Unit Details: Backend function to handle Get Request “/counselors”
Test Owner: Sayani Patra
Test Date: [03/26/2025] - [03/26/2025]
Test Results:
Requested for the list of current counsellors - “Fetched the list of current counsellors”
Structural Coverage: Branch Coverage 100%



14. GET /counselors/available_slots

The backend function to get available slots for appointment of a particular counsellor for a particular date was tested. All the testable branches functioned as expected.

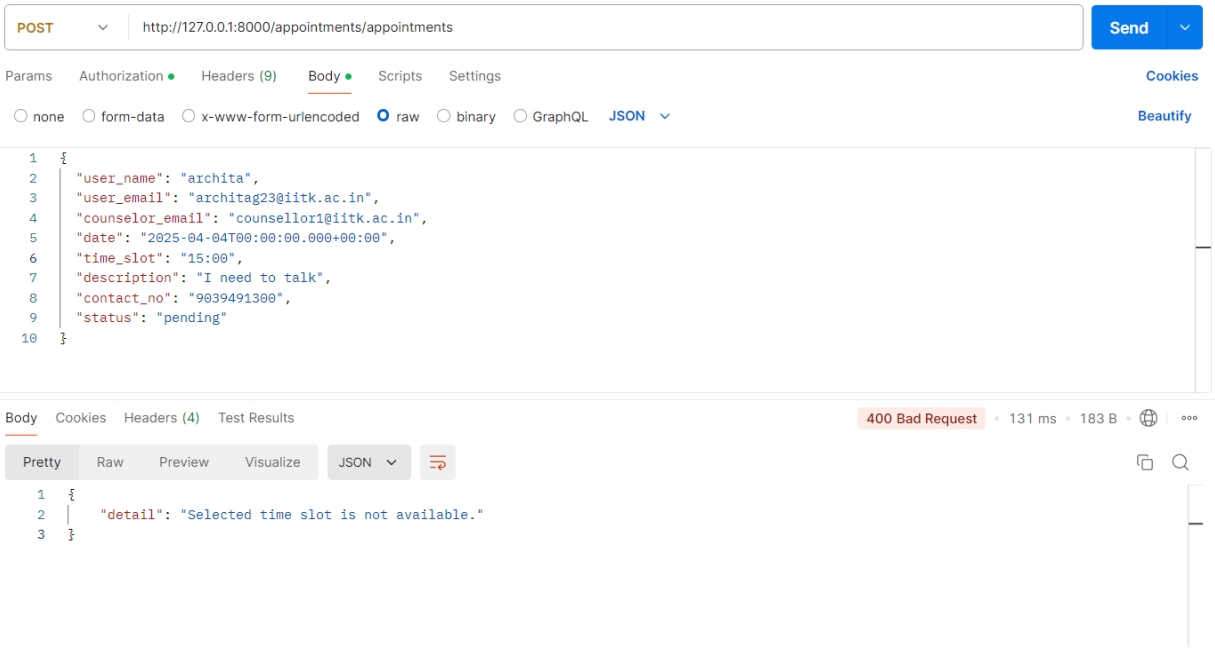
Unit Details: Backend function to handle Get Request “/counselors/available_slots”
Test Owner: Archita Goyal
Test Date: [03/26/2025] - [03/26/2025]
Test Results:
Requested for available slots of appointment of the counselor for a particular date - time_slots (list) for that day
Structural Coverage: Branch Coverage 100%

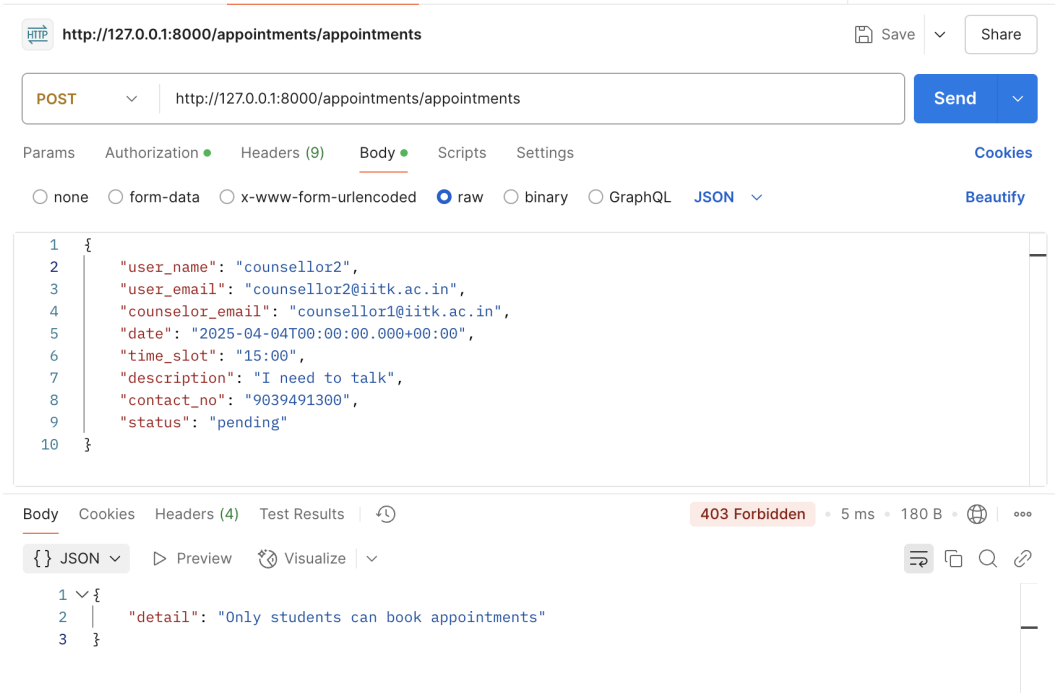


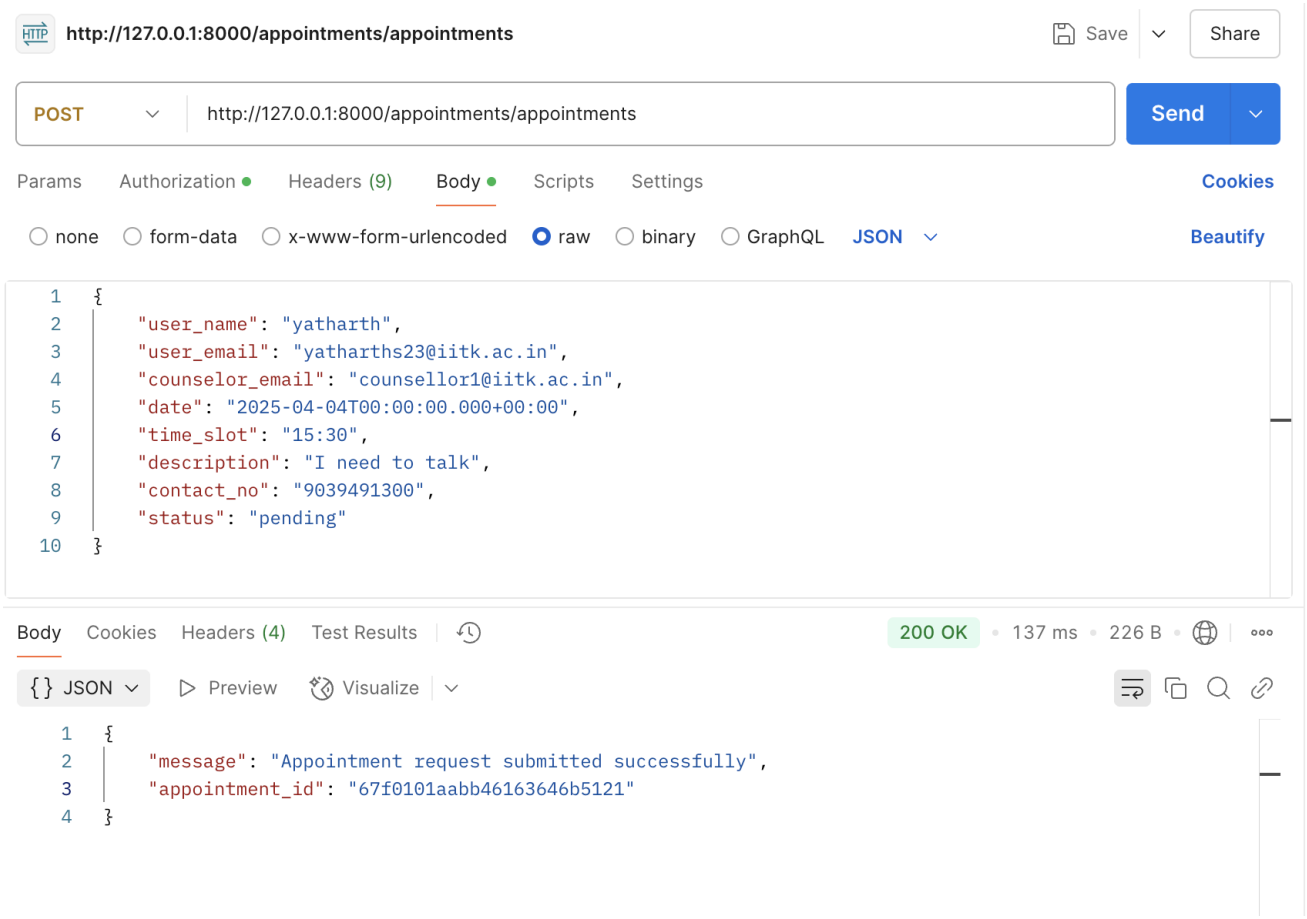
15. POST /appointments

The backend function to create a new appointment was tested. All the testable branches functioned as expected.

Unit Details: Backend function to handle Post Request “/appointments”
Test Owner: Archita Goyal
Test Date: [04/04/2025] - [04/04/2025]
Test Results:
User is not student - “Only Students can book appointments”
User is student and asked for unavailable time slot - “Selected time slot is not available”
User is student and asked for available time slot - “Booked the appointment successfully”
Structural Coverage: Branch Coverage 100%



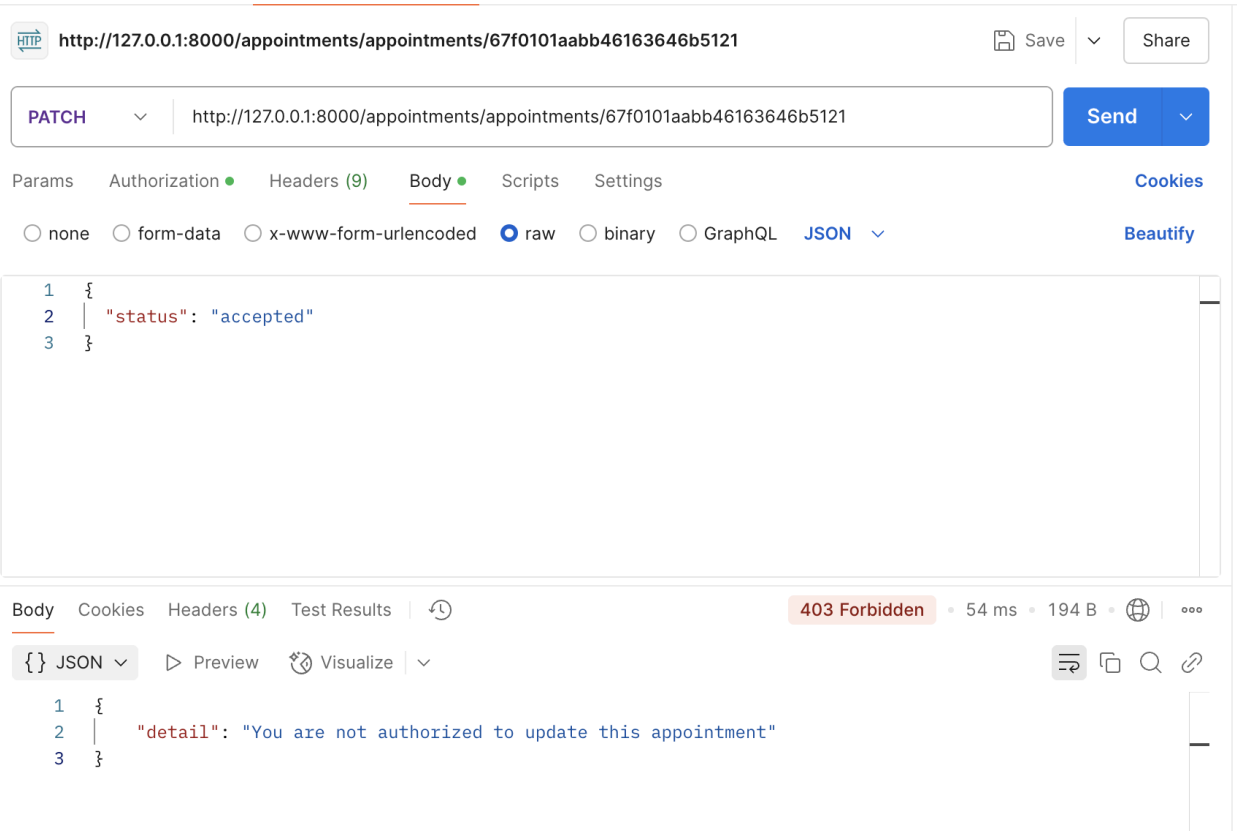


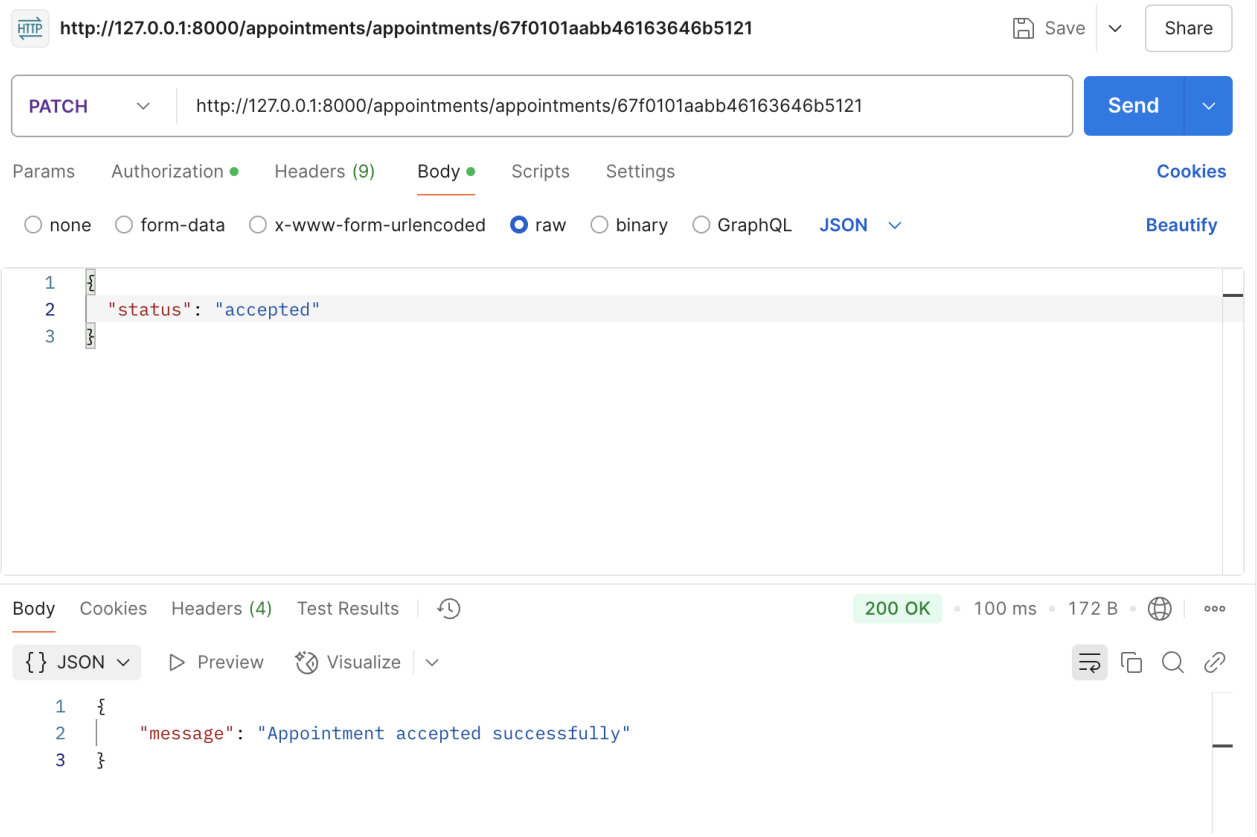


16. PATCH /appointments/{appointment_id}

The backend function to update the status of an appointment was tested. All the testable branches functioned as expected.

Unit Details: Backend function to handle Patch Request “/appointments/{appointment_id}”
Test Owner: Yatharth Sharma
Test Date: [03/26/2025] - [03/26/2025]
Test Results:
Current user is not the counsellor whose appointment is taken - “You are not authorised to update this appointment”
Current user is counsellor and accepting/rejecting his/her appointment request - “Appointment accepted/rejected successfully”
Structural Coverage: Branch Coverage 100%

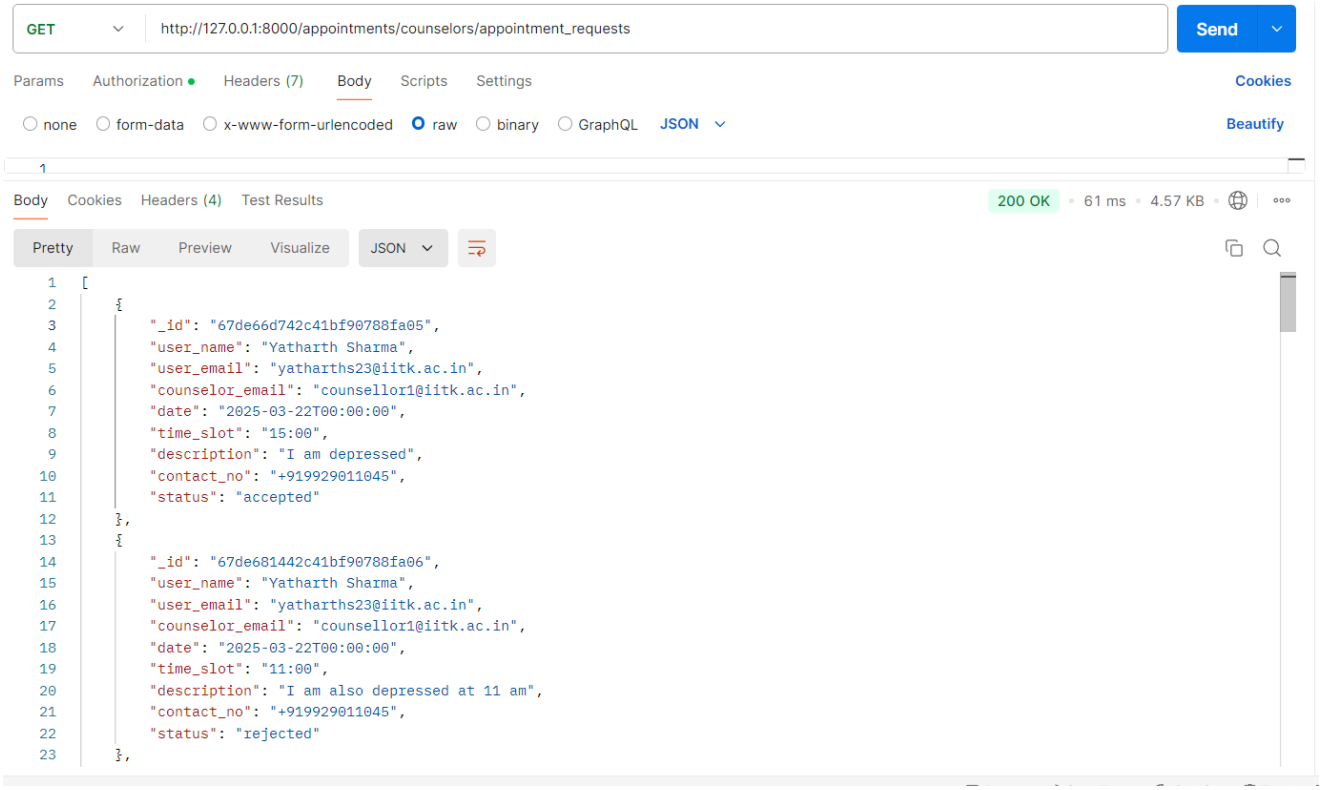

http://127.0.0.1:8000/appointments/appointments/67f0101aabb46163646b5121
Save
Share
PATCH
http://127.0.0.1:8000/appointments/appointments/67f0101aabb46163646b5121
Send
Params
Authorization
Headers (9)
Body
Scripts
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"status": "accepted"
}
Body
Cookies
Headers (4)
Test Results
403 Forbidden
54 ms
194 B
JSON
Preview
Visualize
{
"detail": "You are not authorized to update this appointment"
}

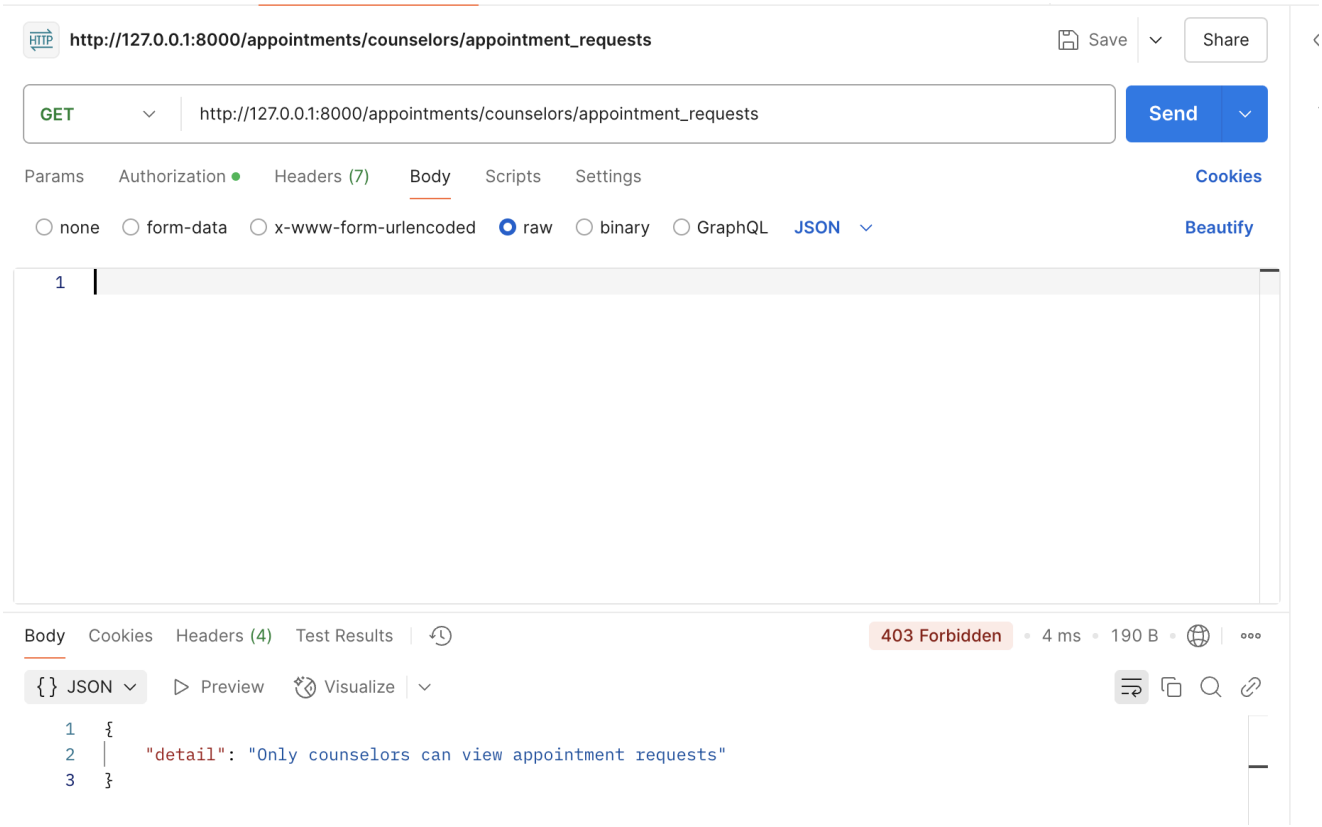

http://127.0.0.1:8000/appointments/appointments/67f0101aabb46163646b5121
Save
Share
PATCH
http://127.0.0.1:8000/appointments/appointments/67f0101aabb46163646b5121
Send
Params
Authorization
Headers (9)
Body
Scripts
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"status": "accepted"
}
Body
Cookies
Headers (4)
Test Results
200 OK
100 ms
172 B
JSON
Preview
Visualize
{
"message": "Appointment accepted successfully"
}

17. GET /counselors/appointment_requests

The backend function to get appointment requests for counsellor was tested. All the testable branches functioned as expected.

Unit Details: Backend function to handle Get Request "/counselors/appointment_requests"
Test Owner: Yatharth Sharma
Test Date: [04/04/2025] - [04/04/2025]
Test Results:
Current user is not a counsellor - "Only counsellor can view appointment requests"
Current user is counsellor and requested for the list of his/her appointment requests - "Fetched the counsellor's appointment requests"
Structural Coverage: Branch Coverage 100%

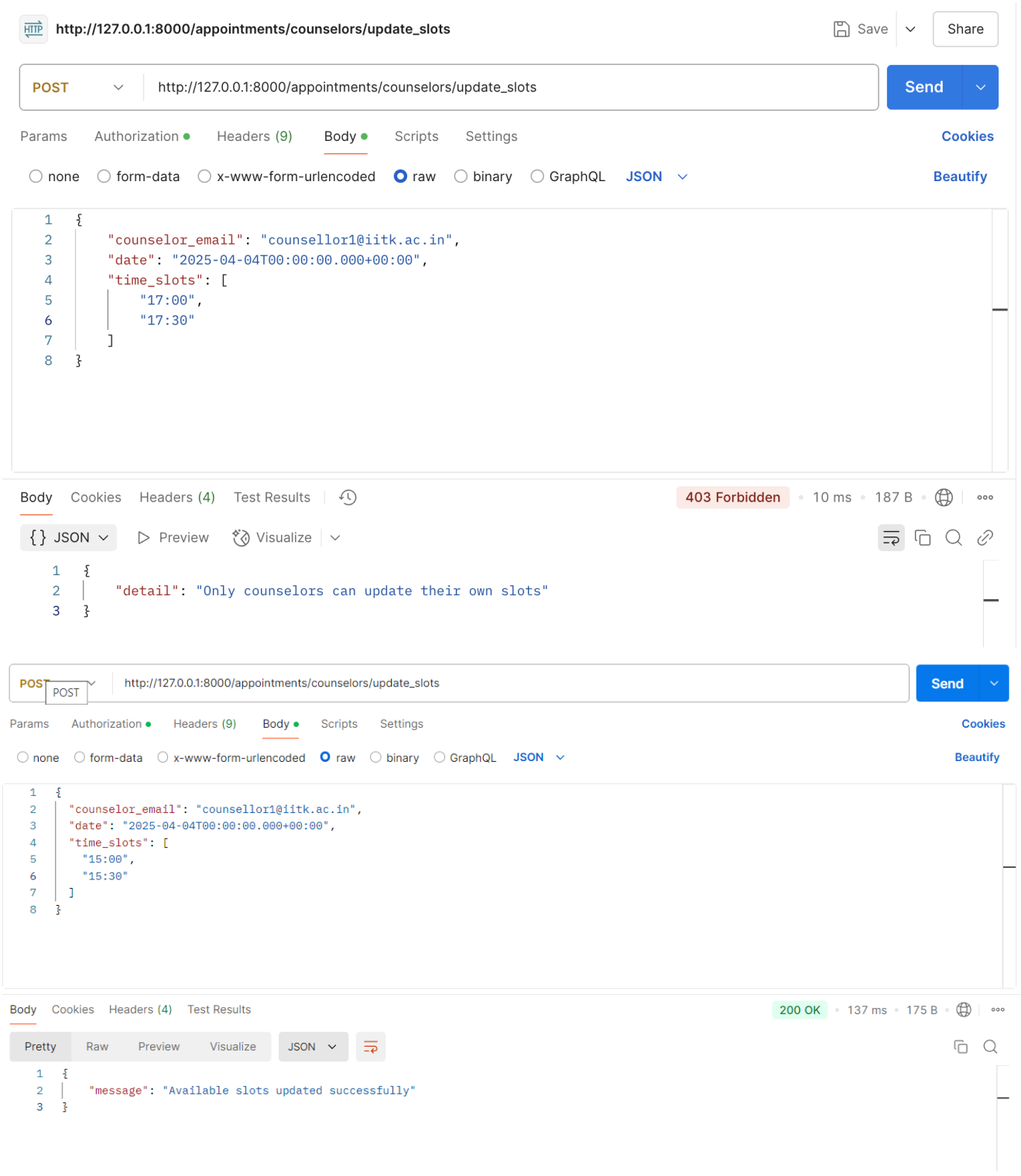




18. POST /counselors/update_slots

The backend function to update the available slots of appointments of counsellor was tested. All the testable branches functioned as expected.

Unit Details: Backend function to handle Post Request “/counselors/update_slots”
Test Owner: Archita Goyal
Test Date: [04/04/2025] - [04/04/2025]
Test Results:
User is not counsellor - “Only counsellor can update their own slots”
Date and list of time slots (of that date) to be made available - “Available slots updated”
Structural Coverage: Branch Coverage 100%

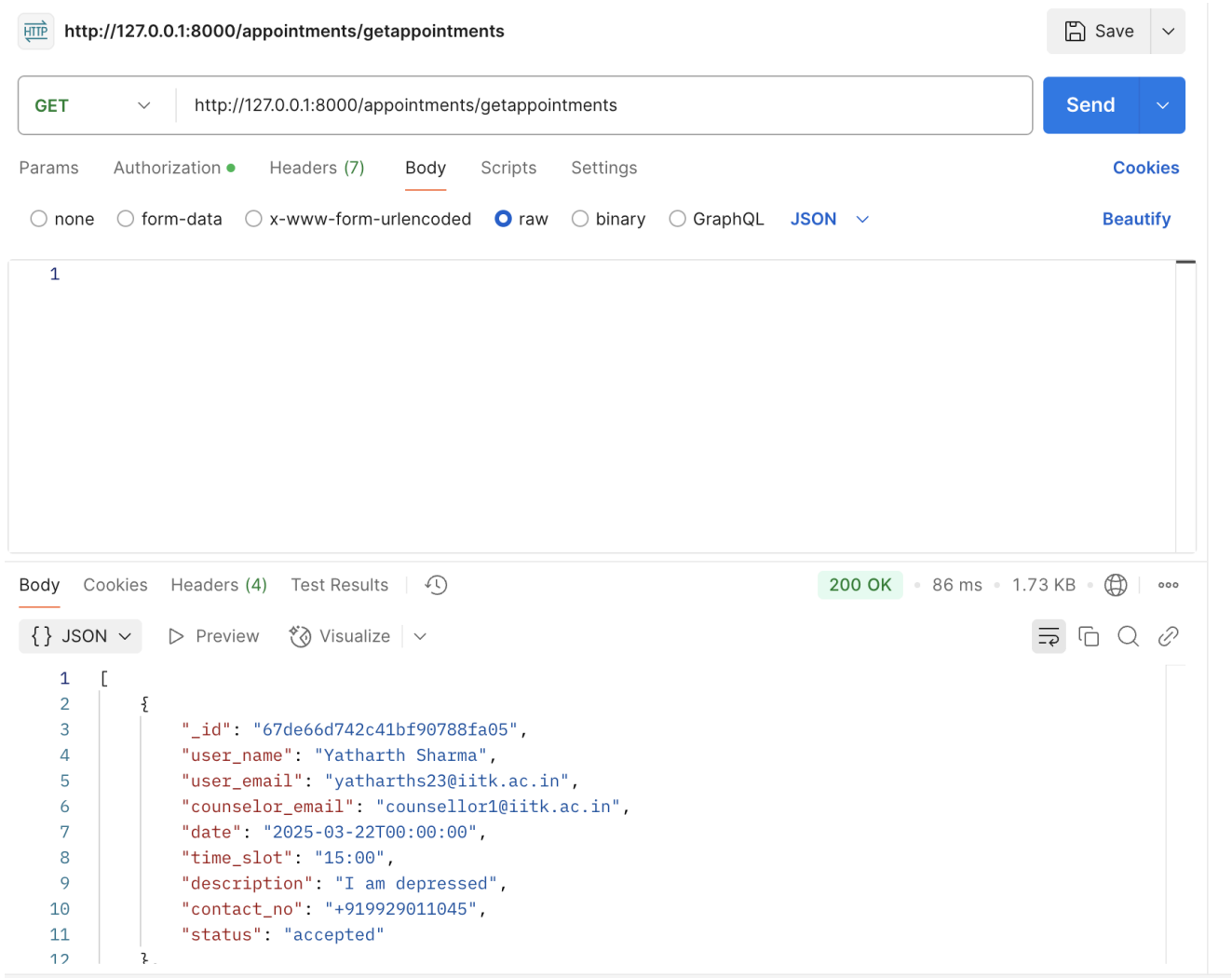


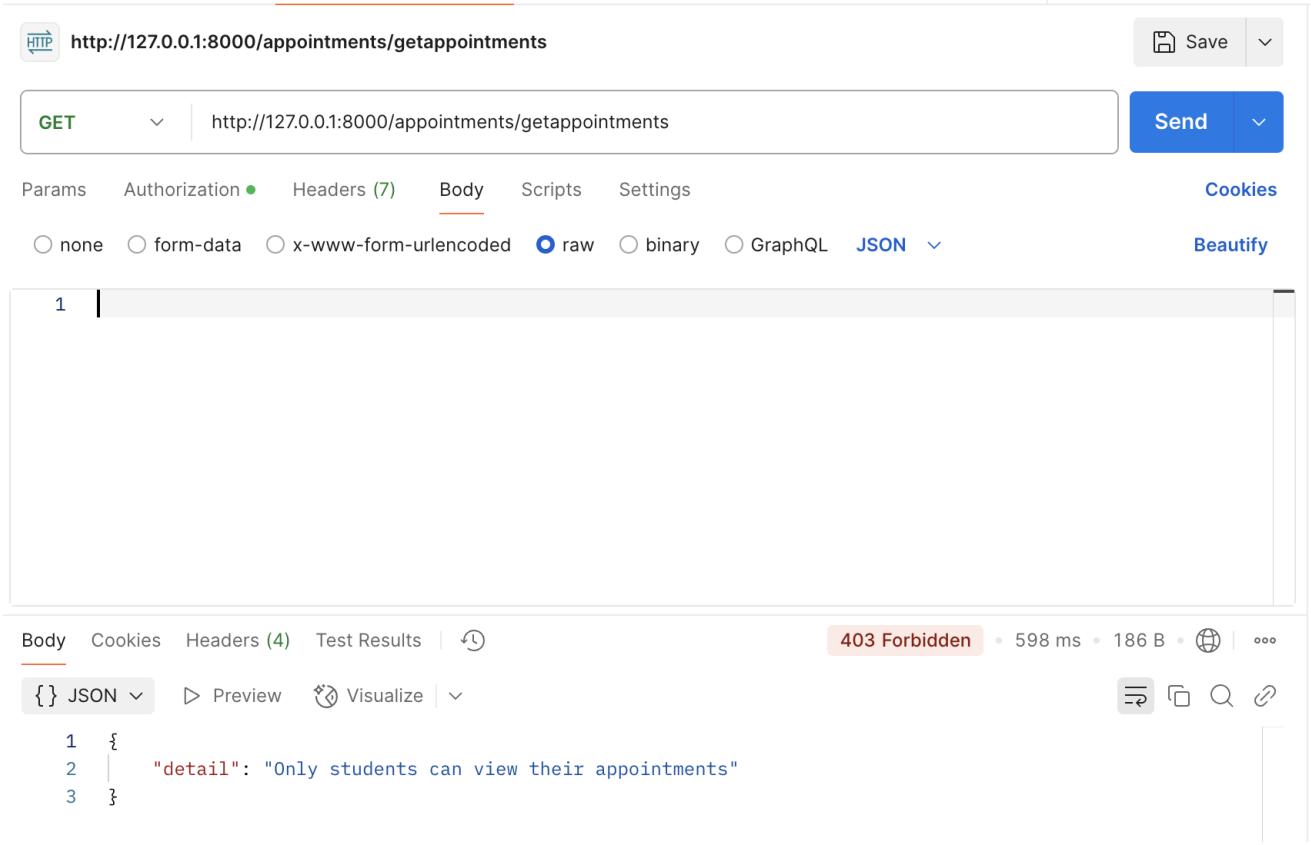
19. GET /getappointments

The backend function to get the appointments of a student was tested. All the testable branches functioned as expected.

Unit Details: Backend function to handle Get Request “/getappointments”

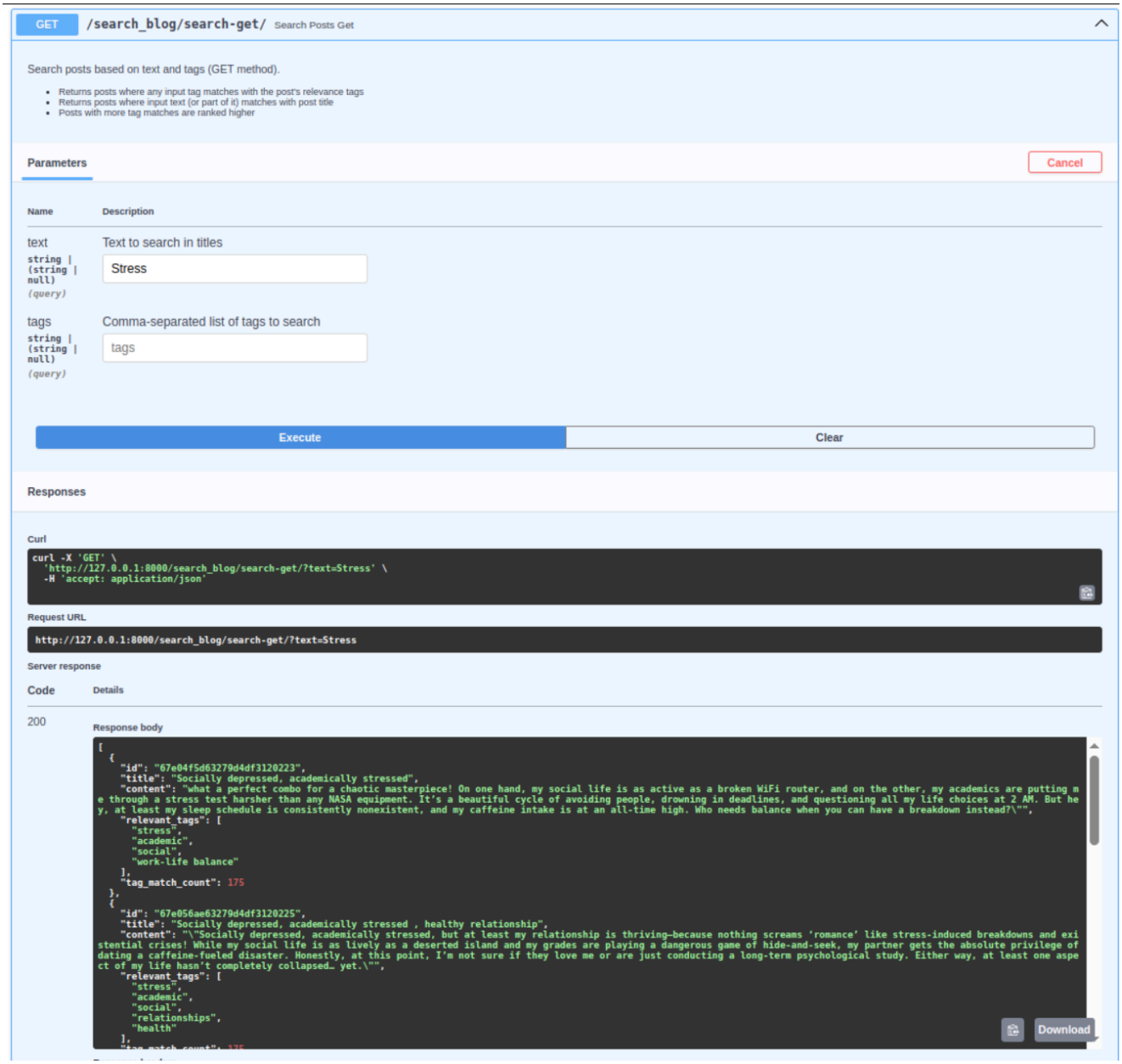
Test Owner: Yatharth Sharma
Test Date: [03/26/2025] - [03/26/2025]
Test Results:
Current user is student and requested for his/her current appointments - Appointment details of that student
Current user is not student - “Only students can view their appointments”
Structural Coverage: Branch Coverage 100%





20. GET /search_blog_search-get

The backend function to search for blogs with specific keywords/tags was tested. The functionality to present the user with the blogs with such keywords/tags was verified and all the testable branches performed as expected.



21. POST /blogs/blogs

The backend function to create a new blog post was tested. The functionality to handle blog creation, including input validation and database storage, was verified, and all testable branches performed as expected.

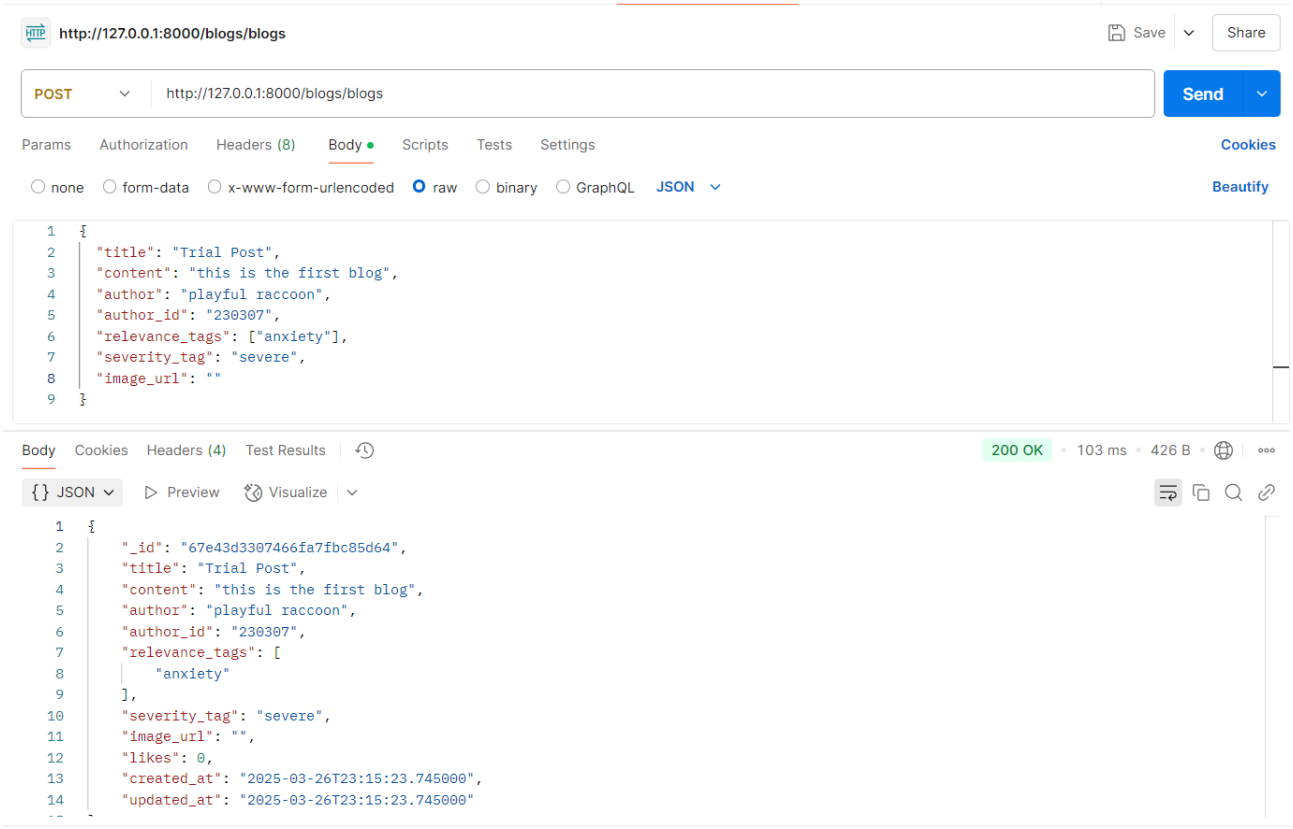
Unit Details: Backend function to handle POST Request "/blogs/blogs"

Test Owner: Vivek

Test Date: [04/04/2025] - [04/04/2025]

Test Results: Blog title, content, and metadata provided - "Blog successfully created and stored in the database"

Structural Coverage: Branch Coverage 100%



22. GET /blogs/blogs

The backend function to retrieve all blogs was tested. The response containing a list of blogs was validated, and all testable branches functioned correctly.

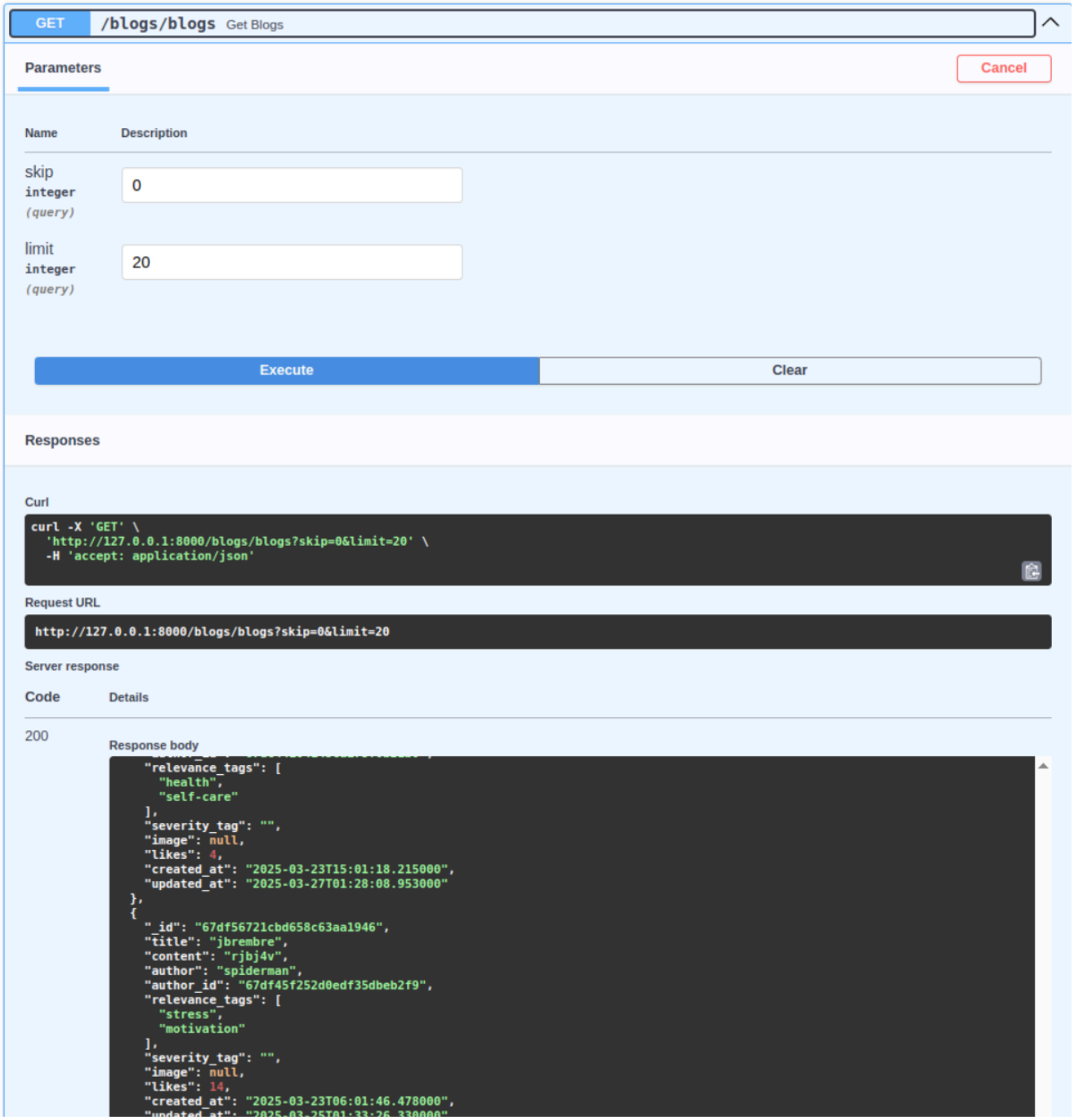
Unit Details: Backend function to handle GET Request “/blogs/blogs”

Test Owner: Vivek

Test Date: [04/04/2025] - [04/04/2025]

Test Results: Request made - “List of all blogs returned successfully”

Structural Coverage: Branch Coverage 100%



23. POST /blogs/classify

The backend function to classify text in a blog was tested. The classification logic and output were verified, and all testable branches worked as expected.

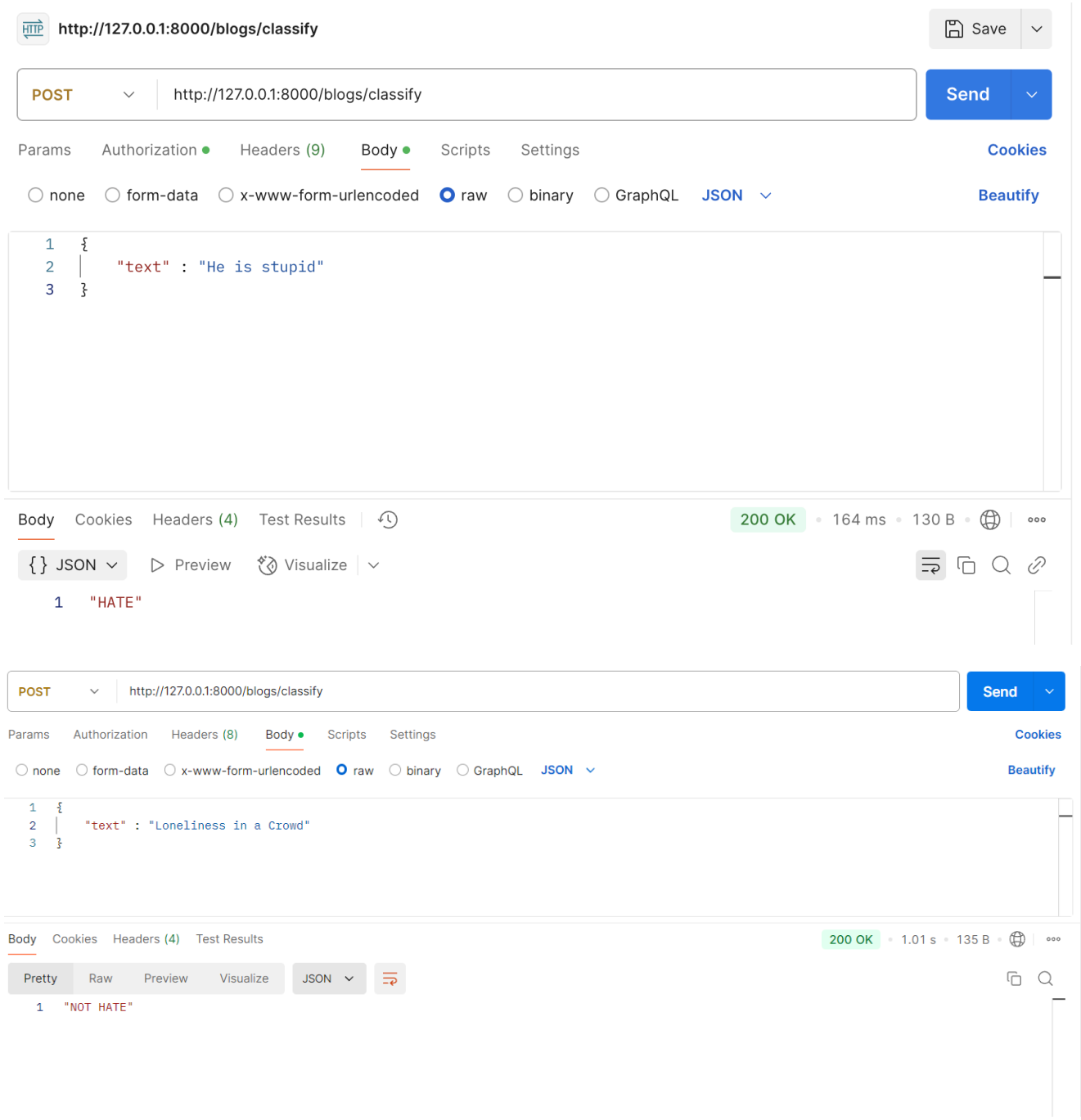
Unit Details: Backend function to handle POST Request “/blogs/classify”

Test Owner: Vivek

Test Date: [04/04/2025] - [04/04/2025]

Test Results:
Text input provided - Text classified with appropriate label

Structural Coverage: Branch Coverage 100%



24. POST /blogs/classify-severity

The backend function to classify the severity of blog content was tested. The severity scoring logic was validated, and all testable branches performed correctly.

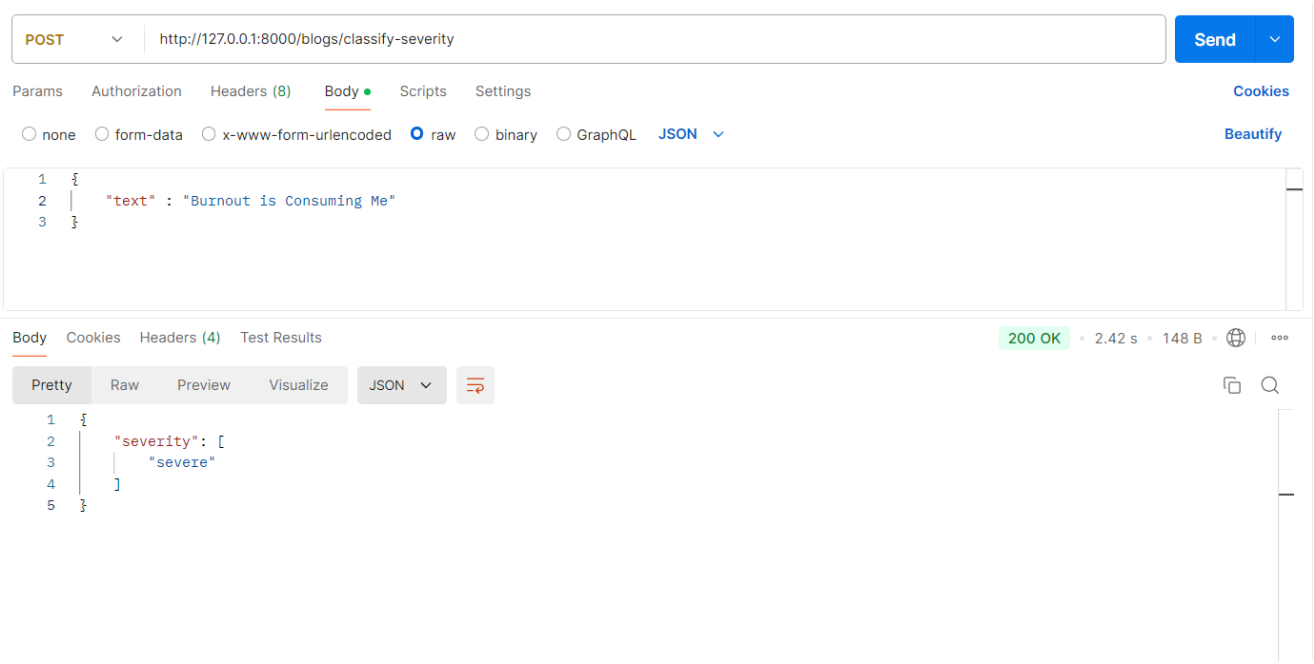
Unit Details: Backend function to handle POST Request “/blogs/classify-severity”

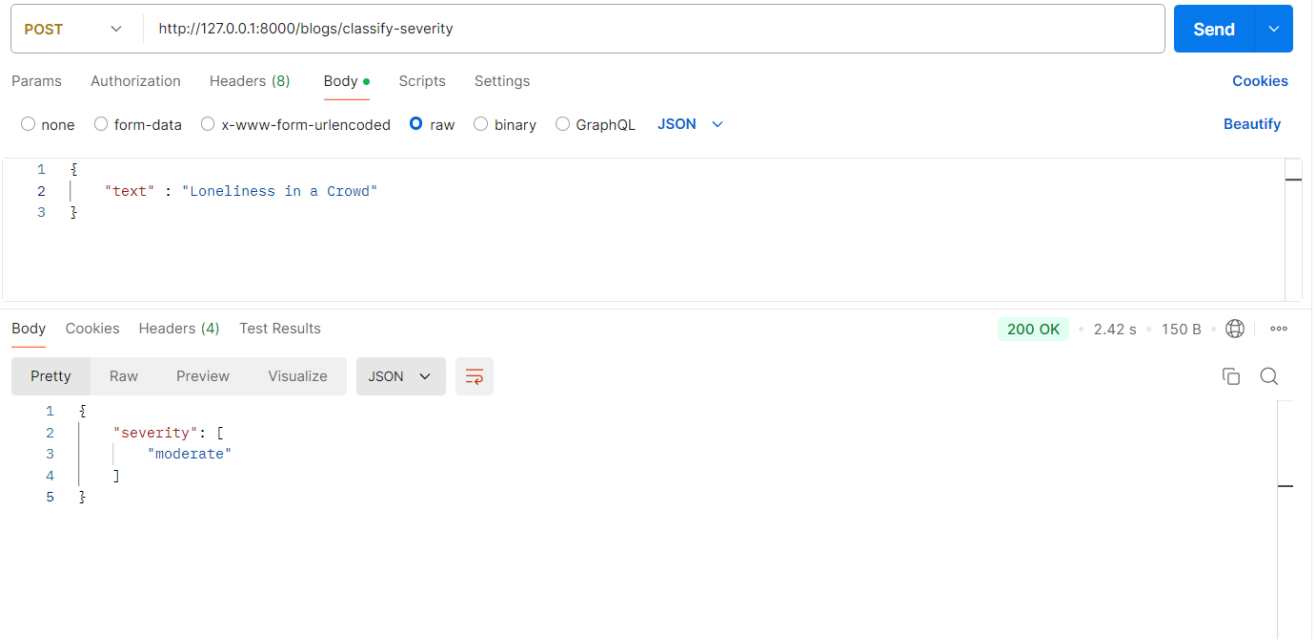
Test Owner: Vivek

Test Date: [04/04/2025] - [04/04/2025]

Test Results: Content provided - “Severity score assigned successfully”

Structural Coverage: Branch Coverage 100%





25. GET /blogs/blogs/{blog_id}

The backend function to retrieve a specific blog by ID was tested. The response with blog details was verified, and all testable branches worked as expected.

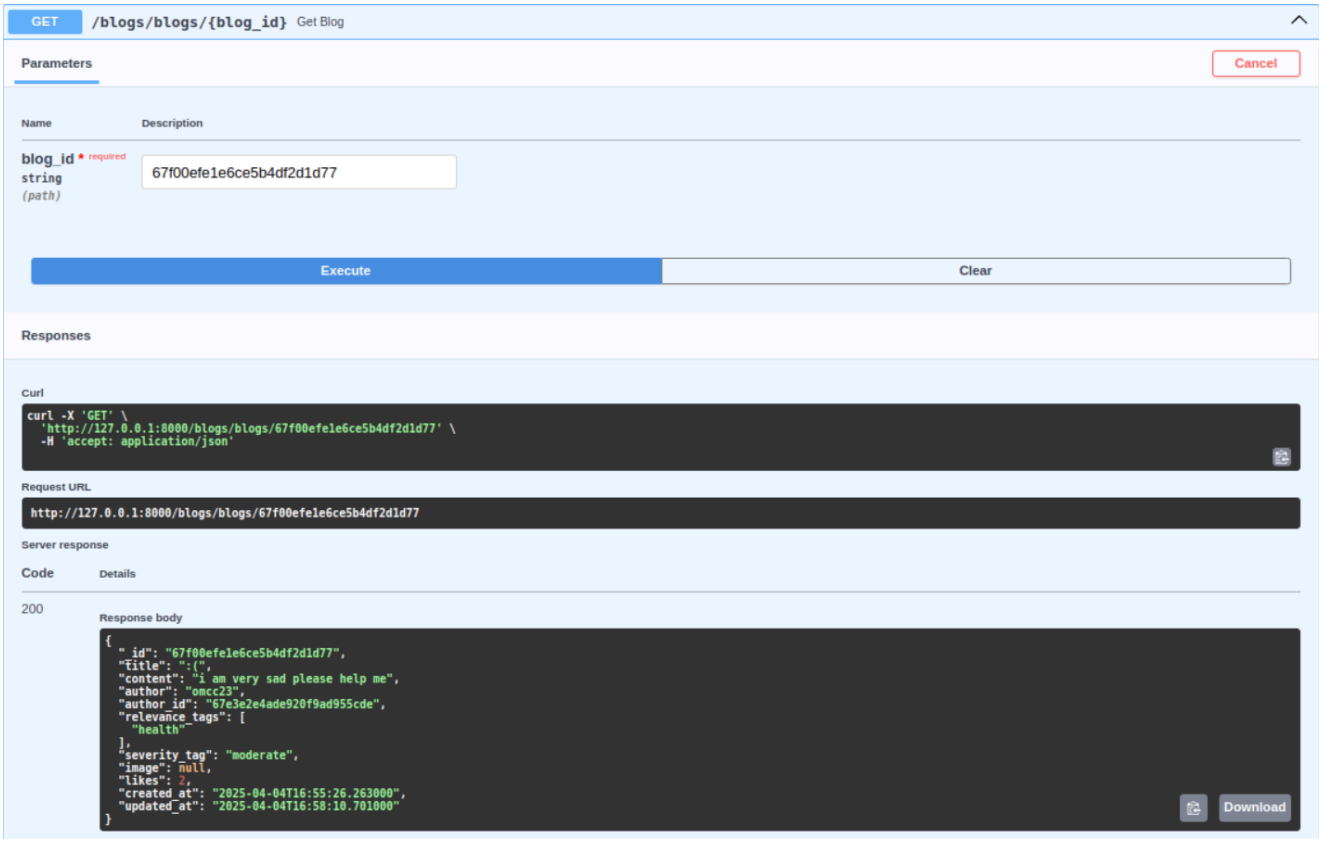
Unit Details: Backend function to handle GET Request "/blogs/blogs/{blog_id}"

Test Owner: Vivek

Test Date: [04/04/2025] - [04/04/2025]

Test Results: Valid blog ID provided - "Blog details returned successfully"

Structural Coverage: Branch Coverage 100%



26. PUT /blogs/blogs/{blog_id}

The backend function to update a specific blog by ID was tested. The update logic and database changes were validated, and all testable branches functioned correctly.

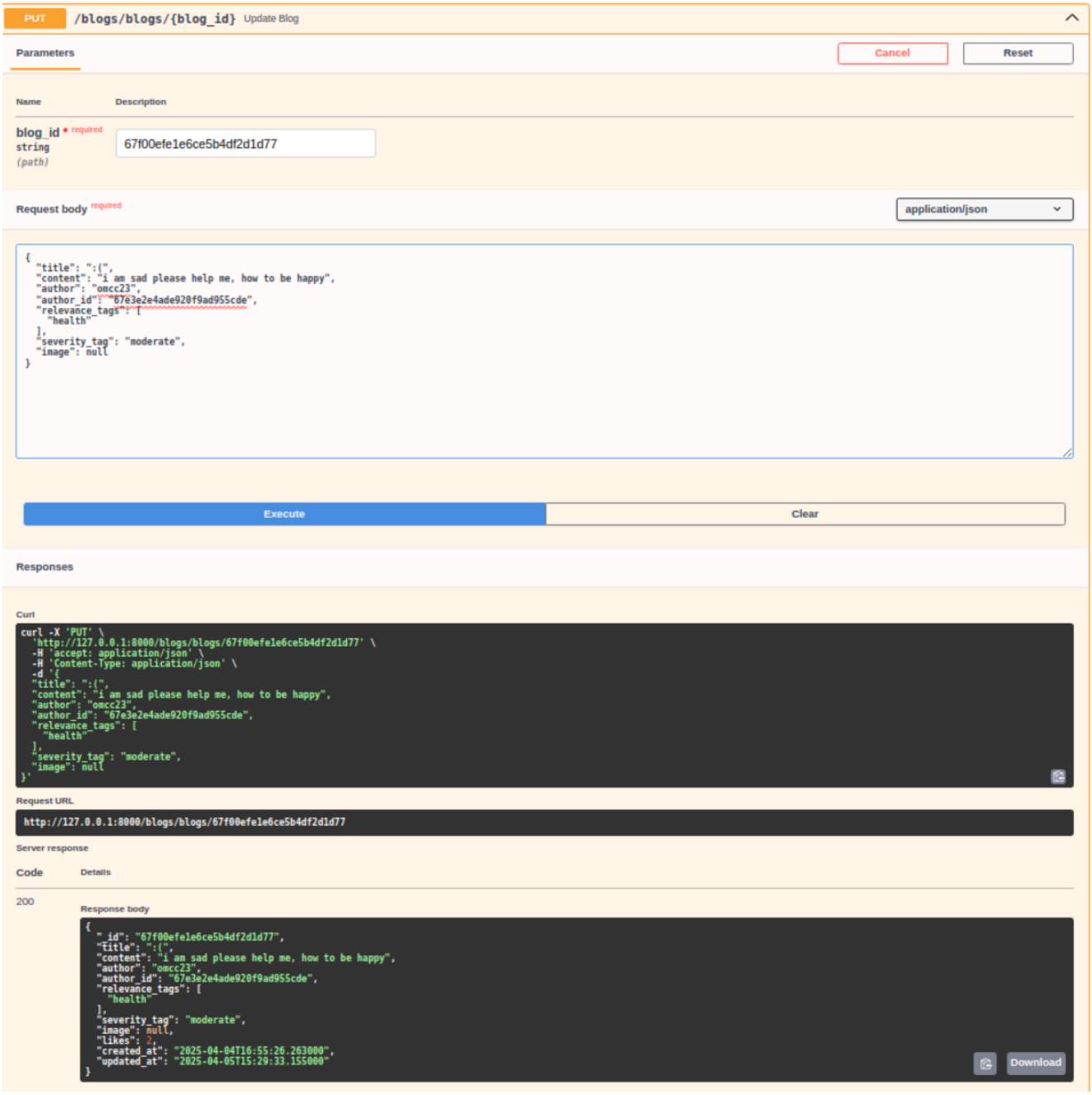
Unit Details: Backend function to handle PUT Request "/blogs/blogs/{blog_id}"

Test Owner: Vivek

Test Date: [04/04/2025] - [04/04/2025]

Test Results: Updated blog data provided - "Blog updated successfully in the database"

Structural Coverage: Branch Coverage 100%



27. DELETE /blogs/blogs/

The backend function to delete a specific blog by ID was tested. The deletion logic and database cleanup were verified, and all testable branches worked as expected.

Unit Details: Backend function to handle DELETE Request “/blogs/blogs/{blog_id}”

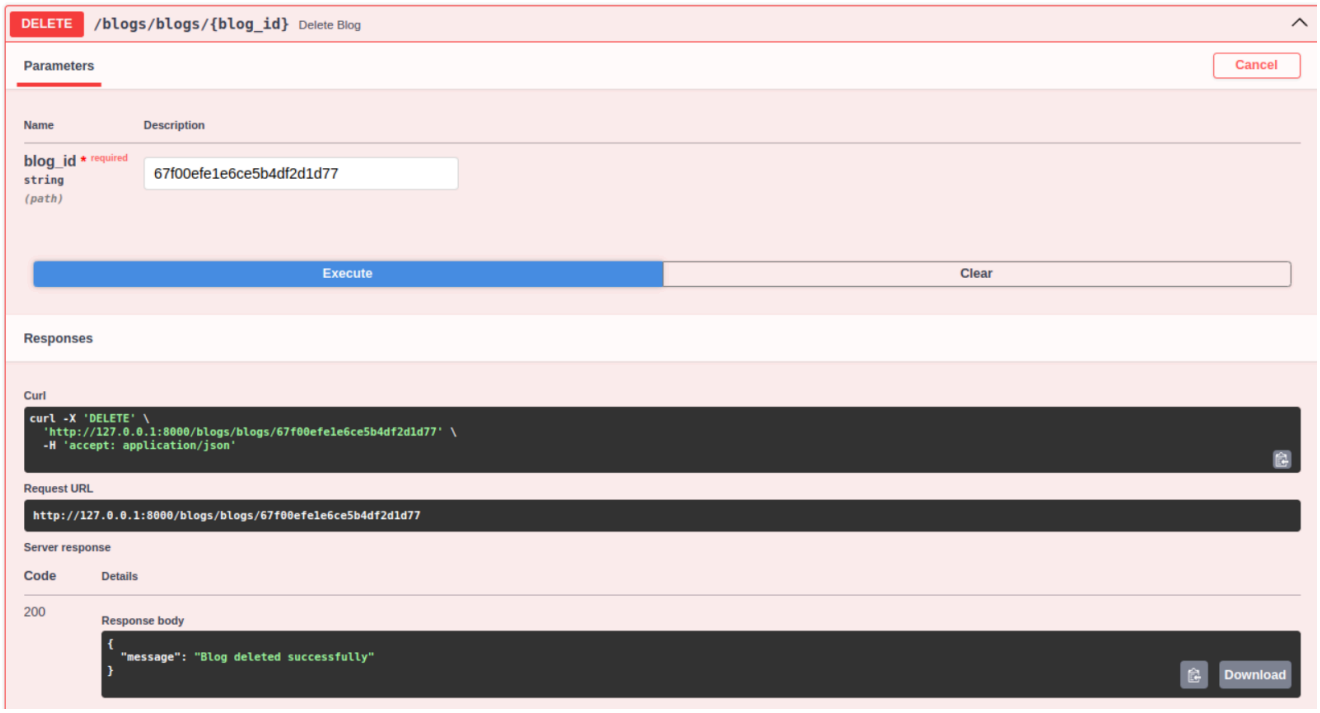
Test Owner: Vivek

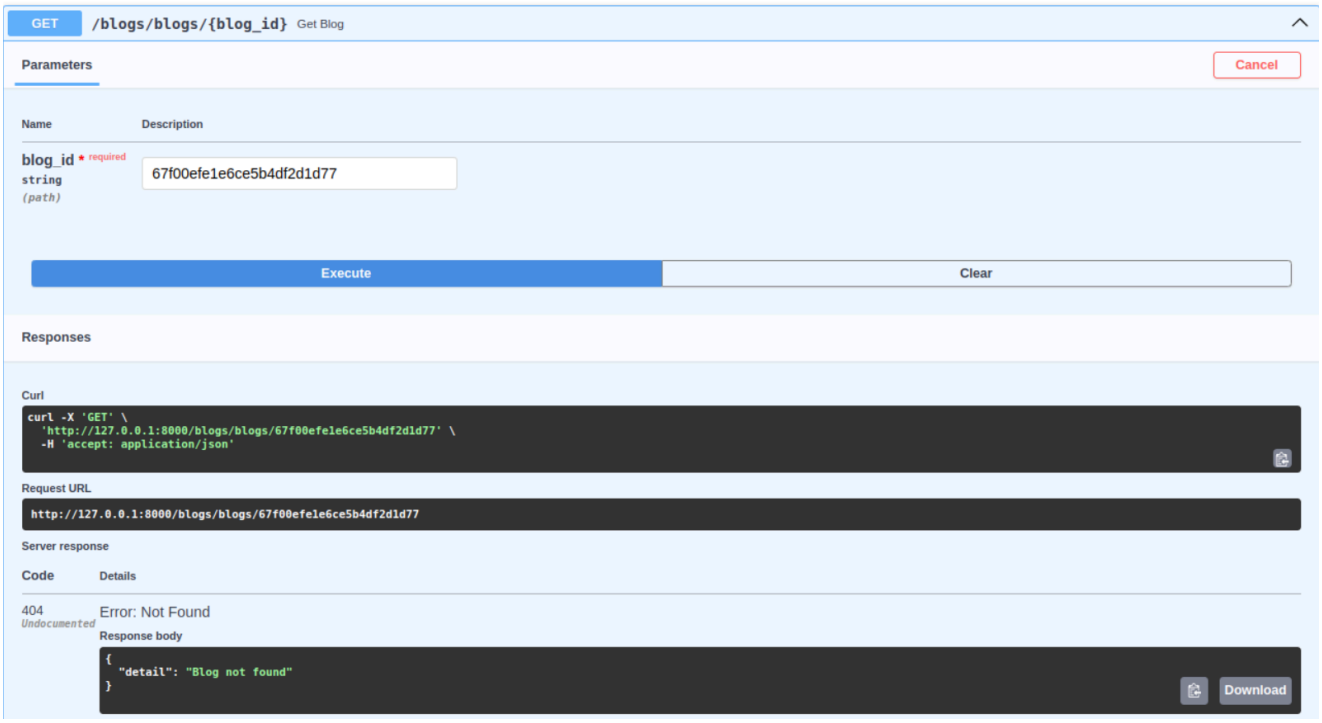
Test Date: [04/04/2025] - [04/04/2025]

Test Results: Valid blog ID provided - “Blog deleted successfully from the database” else

"Blog not found"

Structural Coverage: Branch Coverage 100%





28. POST /blogs/blogs/{blog_id}/upload-image

The backend function to upload an image for a specific blog was tested. Image handling and storage were validated, and all testable branches performed correctly.

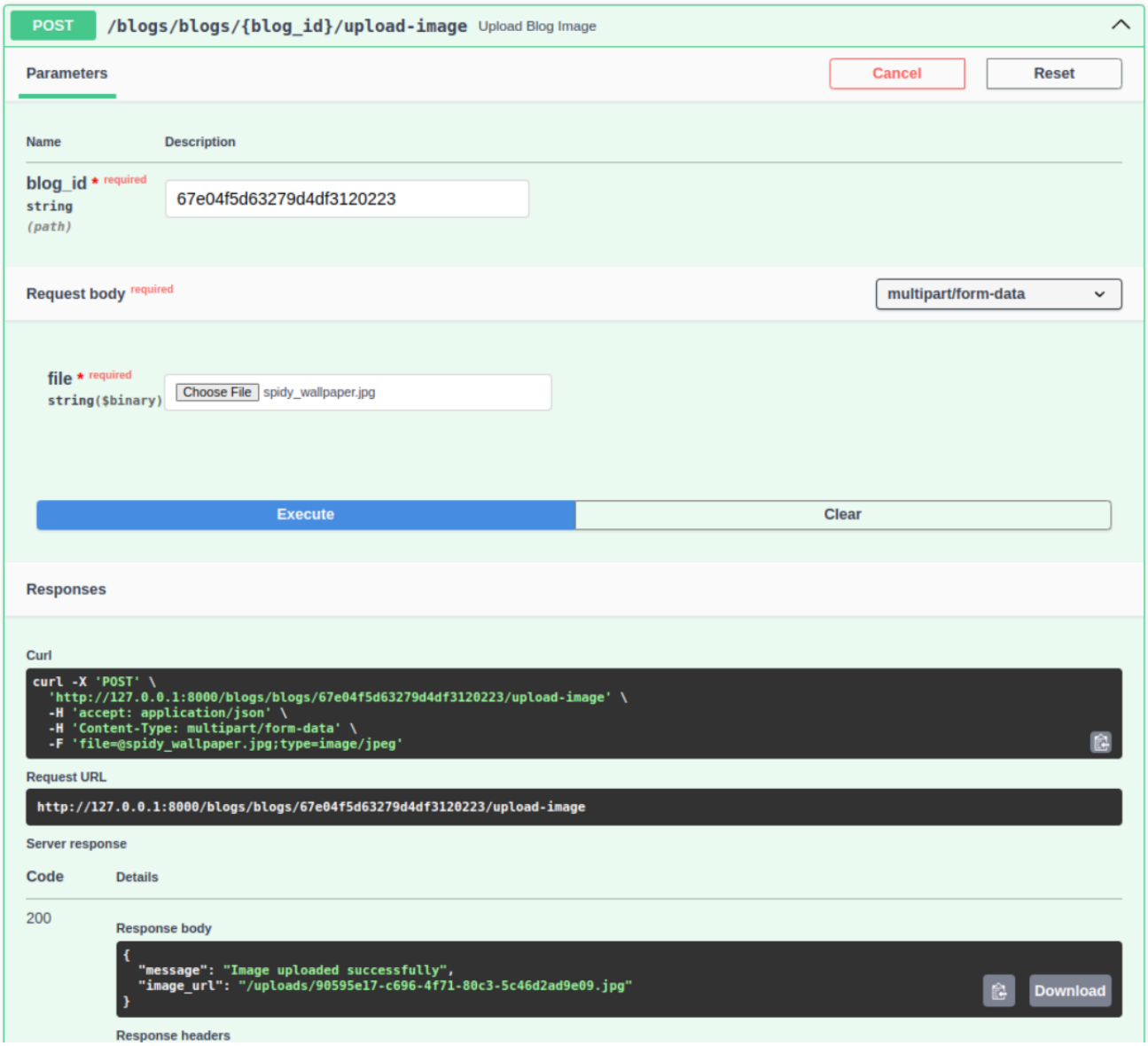
Unit Details: Backend function to handle POST Request "/blogs/blogs/{blog_id}/upload-image"

Test Owner: Vivek

Test Date: [04/04/2025] - [04/04/2025]

Test Results: Image file provided - "Image uploaded and linked to blog successfully"

Structural Coverage: Branch Coverage 100%



29. GET /blogs/blogs/{blog_id}/comments

The backend function to retrieve comments for a specific blog was tested. The response with comment data was verified, and all testable branches worked as expected.

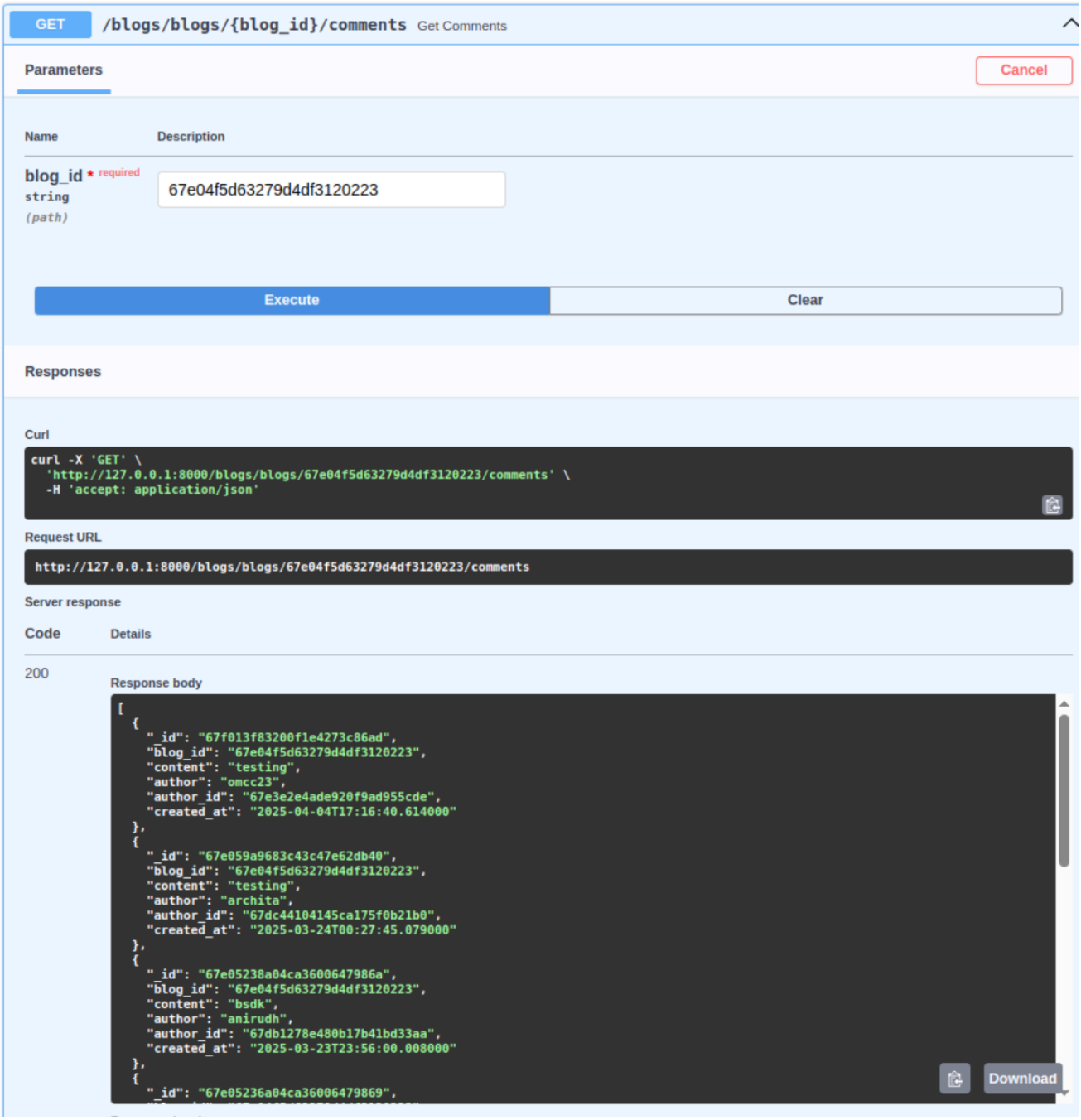
Unit Details: Backend function to handle GET Request "/blogs/blogs/{blog_id}/comments"

Test Owner: Vivek

Test Date: [04/04/2025] - [04/04/2025]

Test Results: Valid blog ID provided - “List of comments returned successfully”

Structural Coverage: Branch Coverage 100%



30. POST /blogs/blogs/{blog_id}/comments

The backend function to create a comment for a specific blog was tested. Comment creation and storage were validated, and all testable branches functioned correctly.

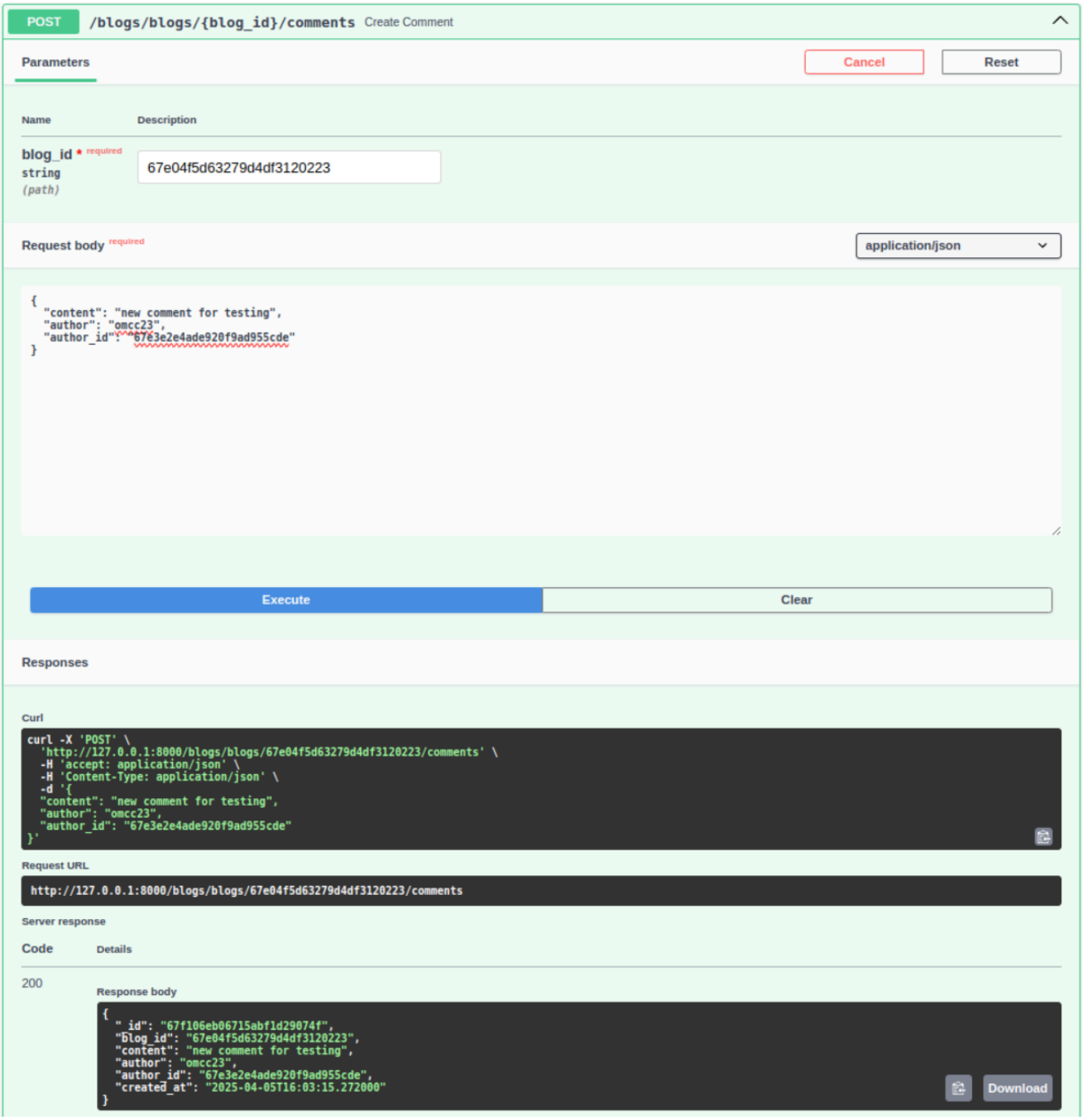
Unit Details: Backend function to handle POST Request “/blogs/blogs/{blog_id}/comments”

Test Owner: Vivek

Test Date: [04/04/2025] - [04/04/2025]

Test Results: Comment text provided - “Comment created and linked to blog successfully”

Structural Coverage: Branch Coverage 100%



31. PUT /blogs/blogs/{blog_id}/likes

The backend function to update likes for a specific blog was tested. The like count adjustment

logic was verified, and all testable branches worked as expected.

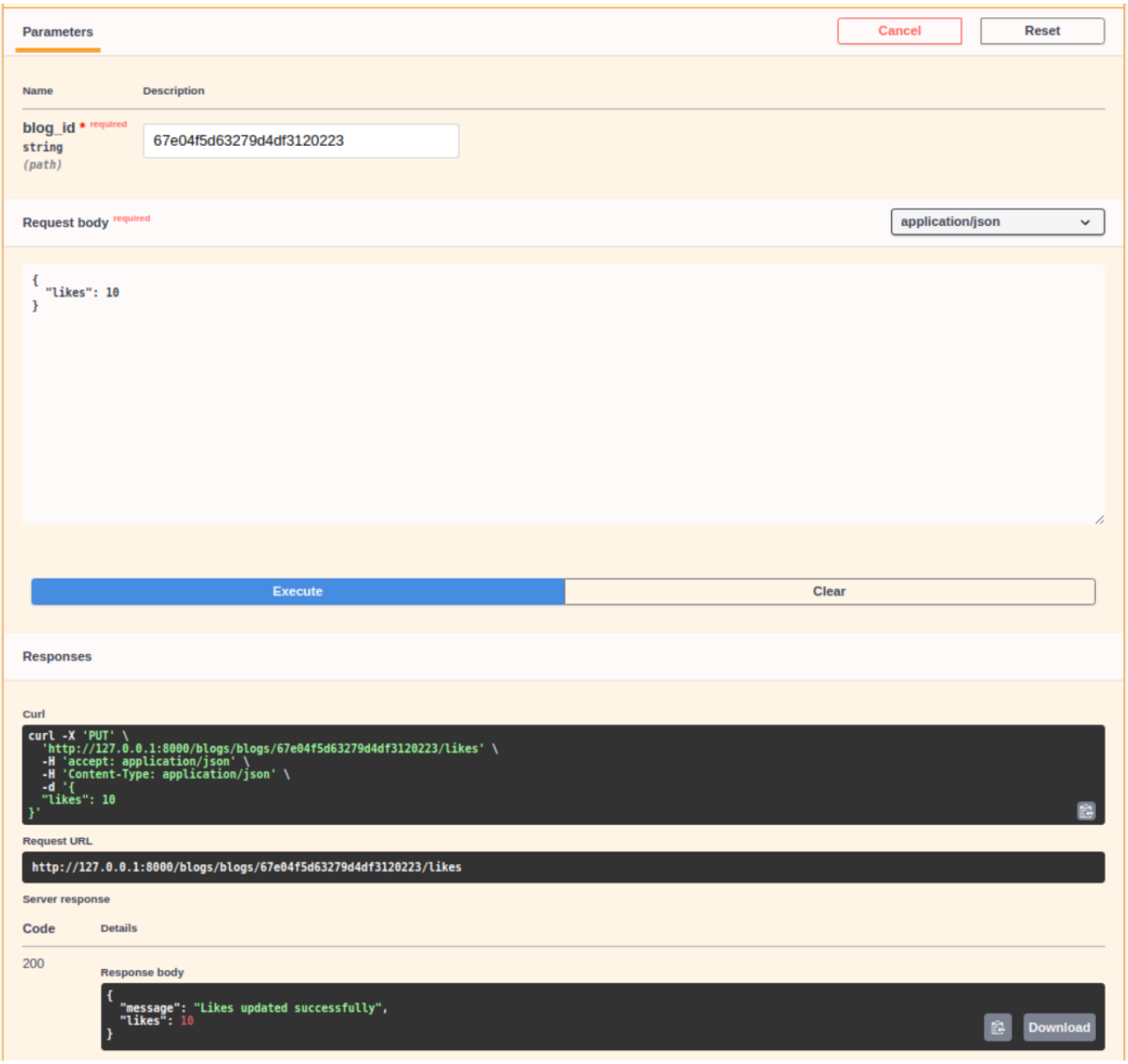
Unit Details: Backend function to handle PUT Request “/blogs/blogs/{blog_id}/likes”

Test Owner: Vivek

Test Date: [04/04/2025] - [04/04/2025]

Test Results: Like data provided - “Likes updated successfully”

Structural Coverage: Branch Coverage 100%



32. POST /blogs/blogs/{blog_id}/like

The backend function to like a specific blog was tested. The like action and database update were validated, and all testable branches performed correctly.

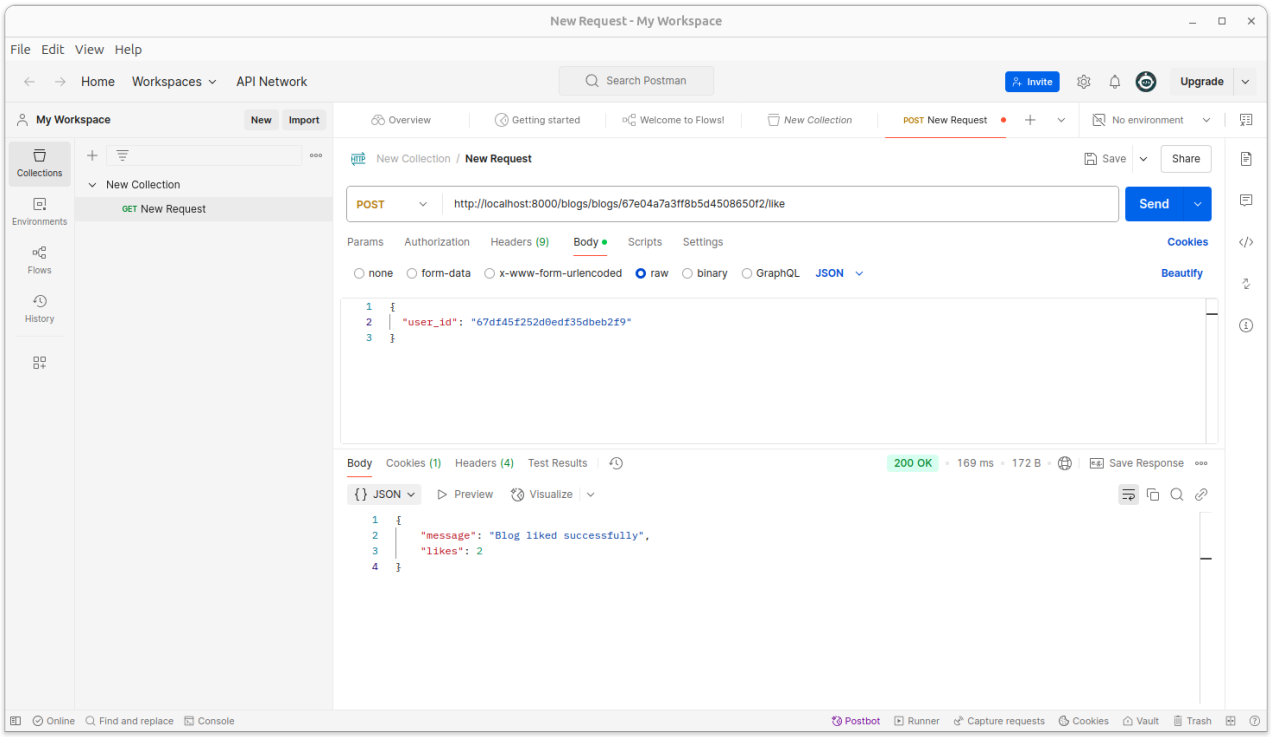
Unit Details: Backend function to handle POST Request “/blogs/blogs/{blog_id}/like”

Test Owner: Vivek

Test Date: [04/04/2025] - [04/04/2025]

Test Results: Like request made - “Blog liked successfully by user”

Structural Coverage: Branch Coverage 100%



33. POST /blogs/blogs/{blog_id}/unlike

The backend function, unlike a specific blog, was tested. The unlike action and database update were verified, and all testable branches worked as expected.

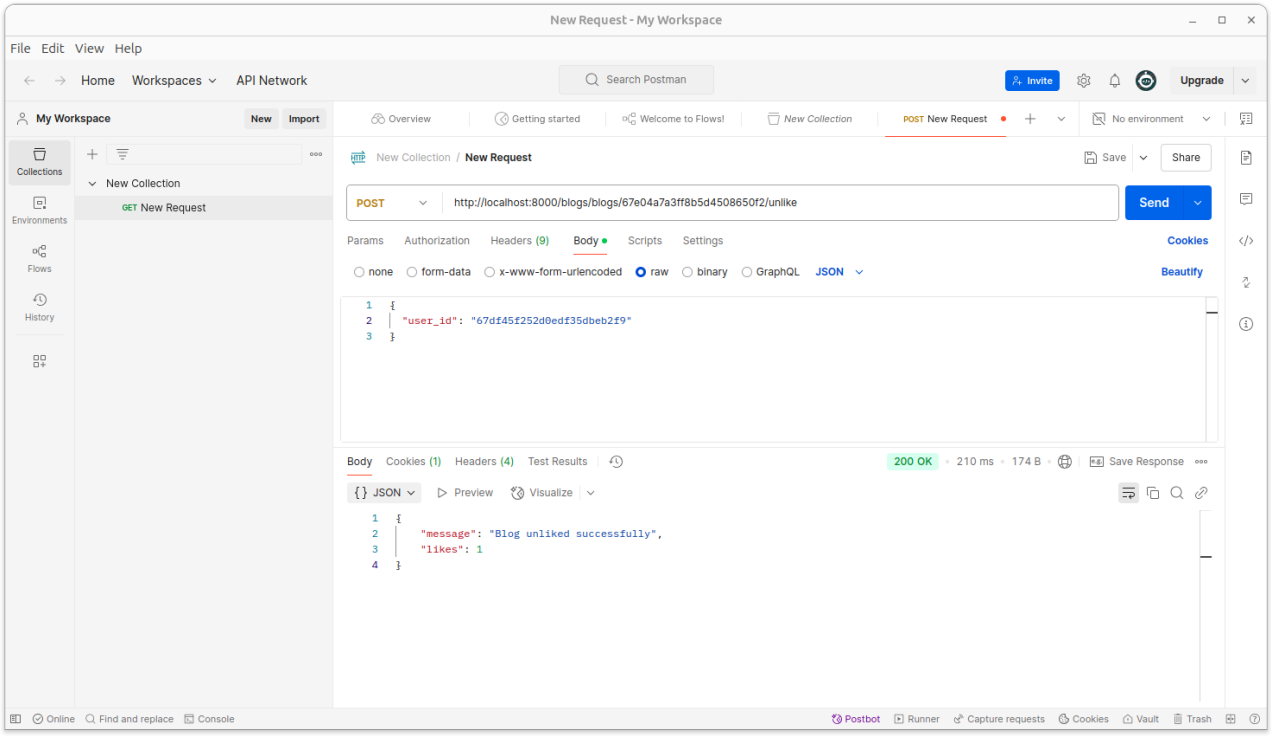
Unit Details: Backend function to handle POST Request “/blogs/blogs/{blog_id}/unlike”

Test Owner: Vivek

Test Date: [04/04/2025] - [04/04/2025]

Test Results: Unlike request made - “Blog unliked successfully by user”

Structural Coverage: Branch Coverage 100%



34. GET /blogs/user/liked-posts

The backend function to retrieve a user's liked posts was tested. The response with liked blog data was validated, and all testable branches functioned correctly.

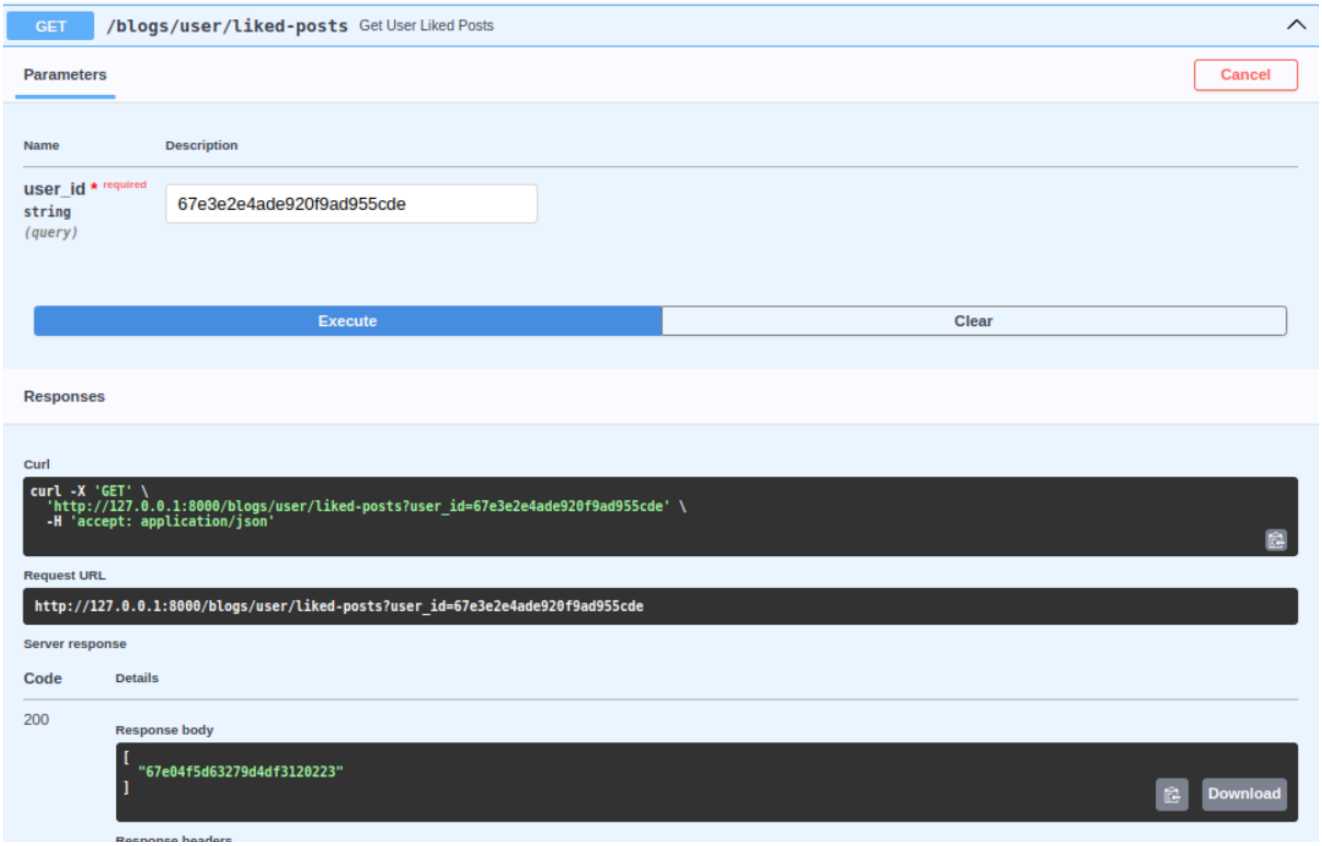
Unit Details: Backend function to handle GET Request "/blogs/user/liked-posts"

Test Owner: Vivek

Test Date: [04/04/2025] - [04/04/2025]

Test Results: User ID provided - "List of liked posts returned successfully"

Structural Coverage: Branch Coverage 100%



35. GET /blogs/user/{user_id}/blogs

The backend function to retrieve blogs by a specific user was tested. The response with user blog data was verified, and all testable branches worked as expected.

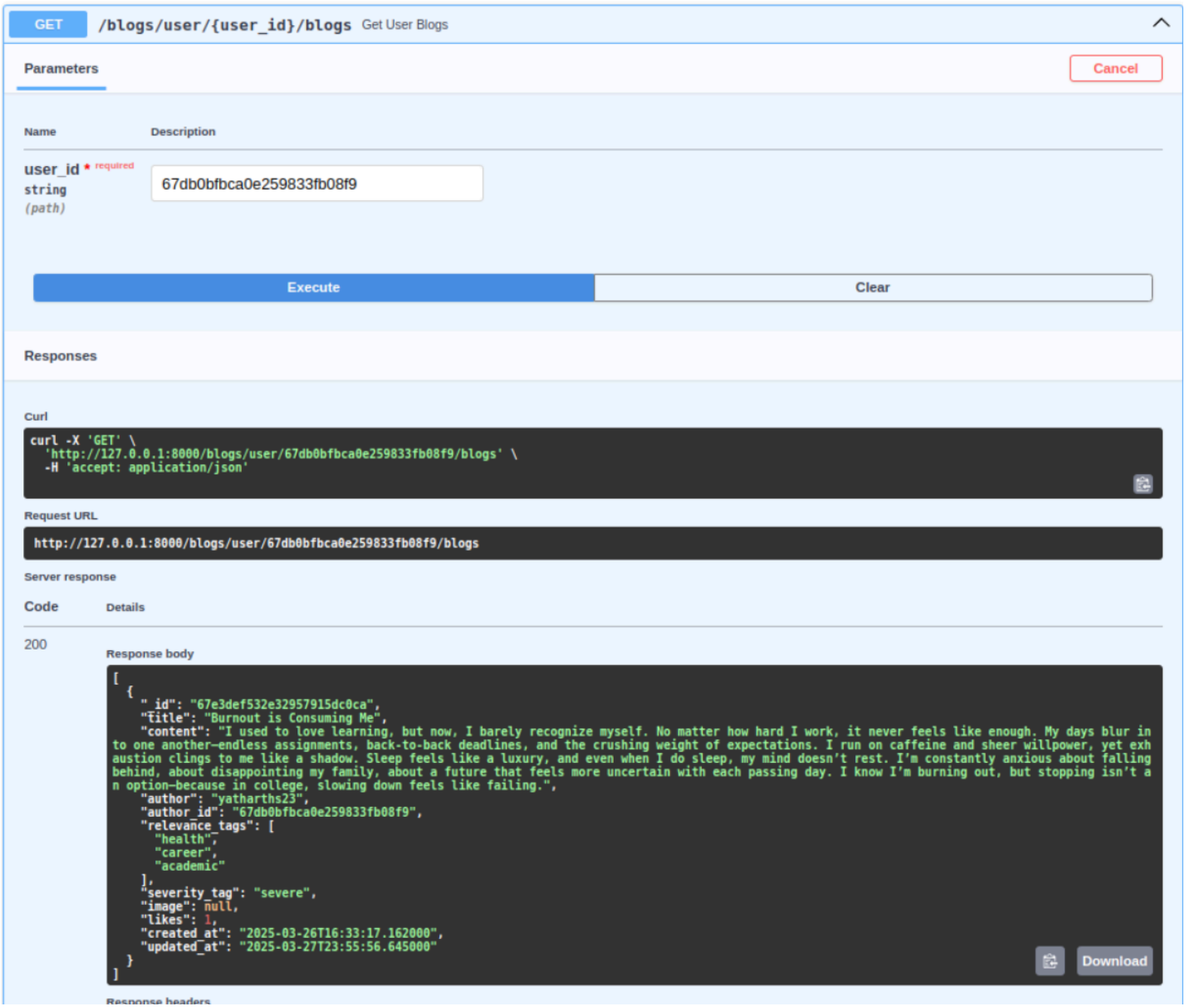
Unit Details: Backend function to handle GET Request “/blogs/user/{user_id}/blogs”

Test Owner: Vivek

Test Date: [04/04/2025] - [04/04/2025]

Test Results: Valid user ID provided - “List of user blogs returned successfully”

Structural Coverage: Branch Coverage 100%



36. POST /blogs/report-blog

The backend function to report a blog was tested. The reporting logic and storage were validated, and all testable branches performed correctly.

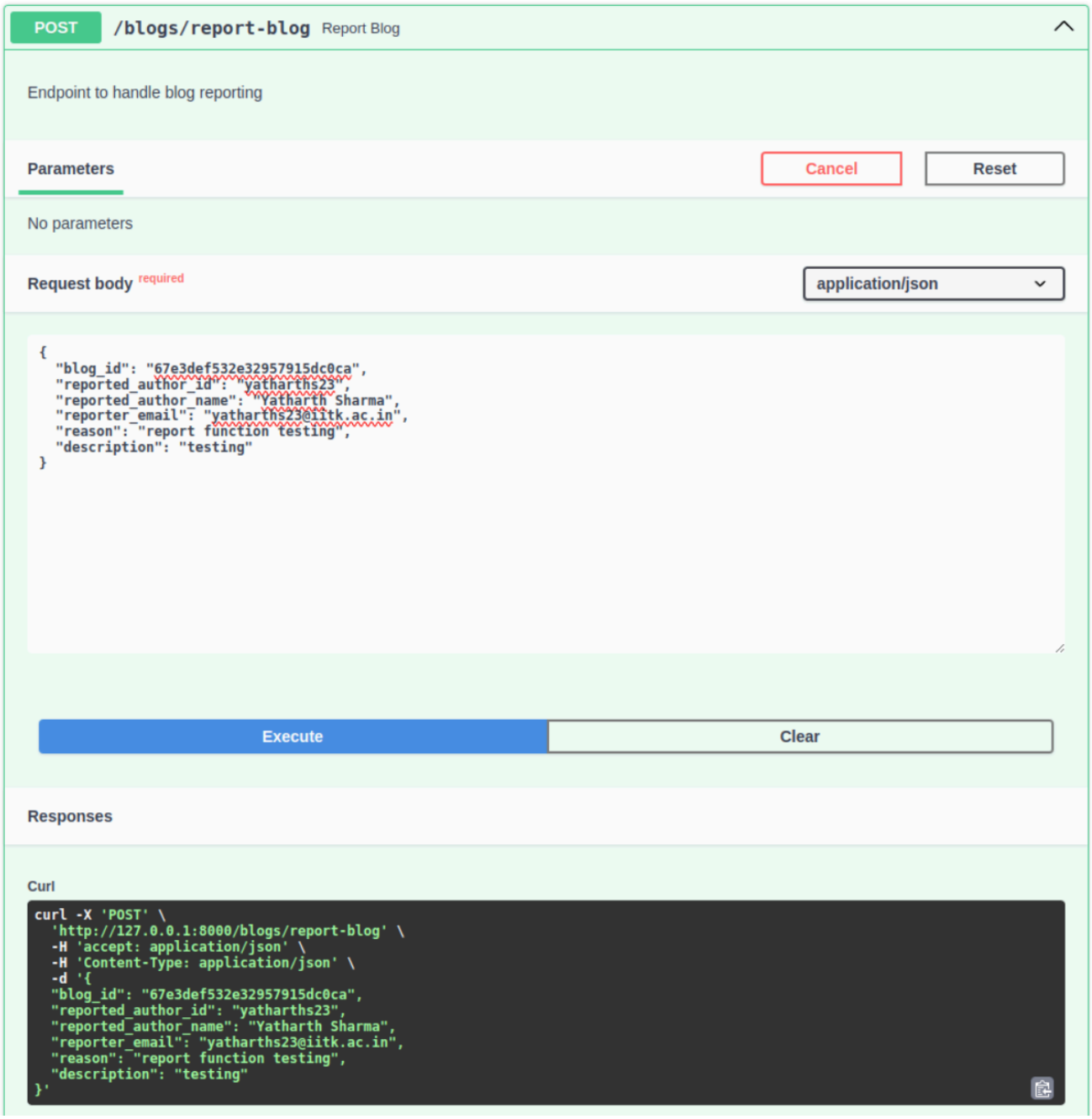
Unit Details: Backend function to handle POST Request “/blogs/report-blog”

Test Owner: Vivek

Test Date: [04/04/2025] - [04/04/2025]

Test Results: Report details provided - “Blog reported successfully”

Structural Coverage: Branch Coverage 100%



37. GET /api/messages/{user_id}

The backend function to fetch all the messages sent and received by a particular user was tested

Unit Details: Backend function to handle GET Request “/api/messages/{user_id}”

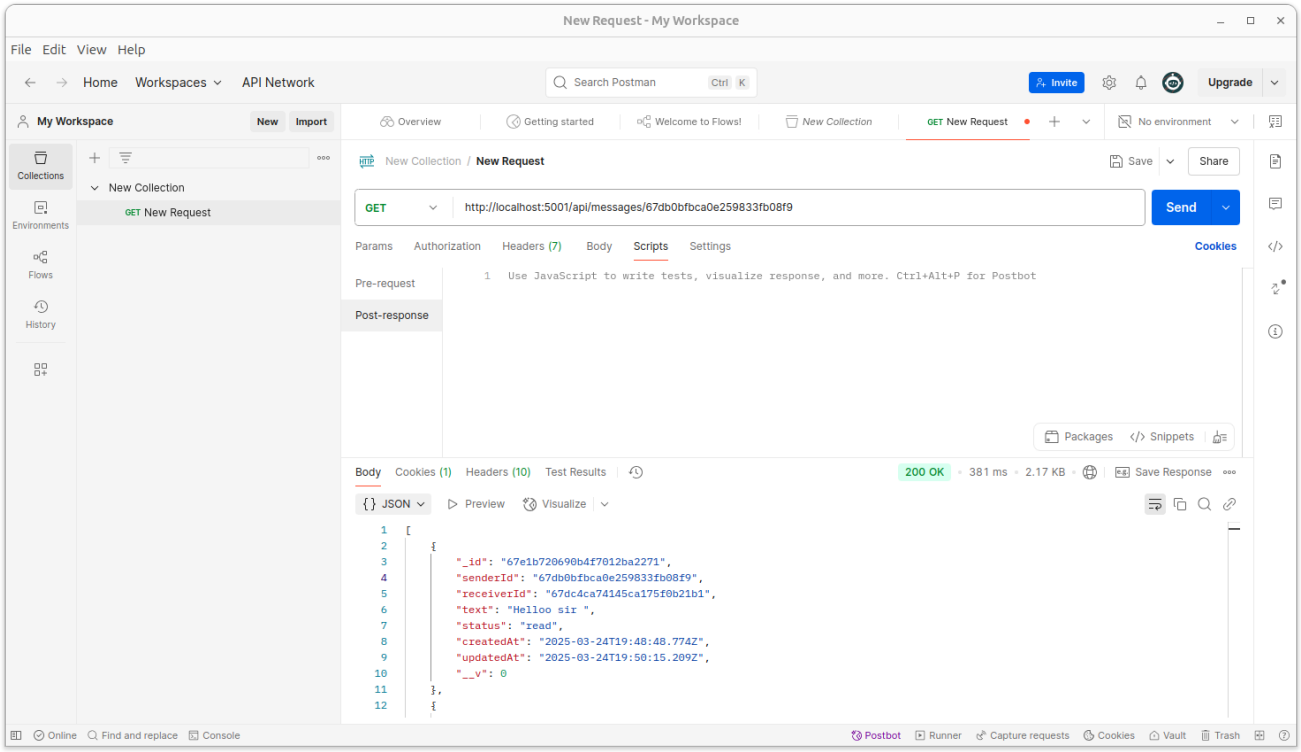
Test Owner: Anirudh Singh

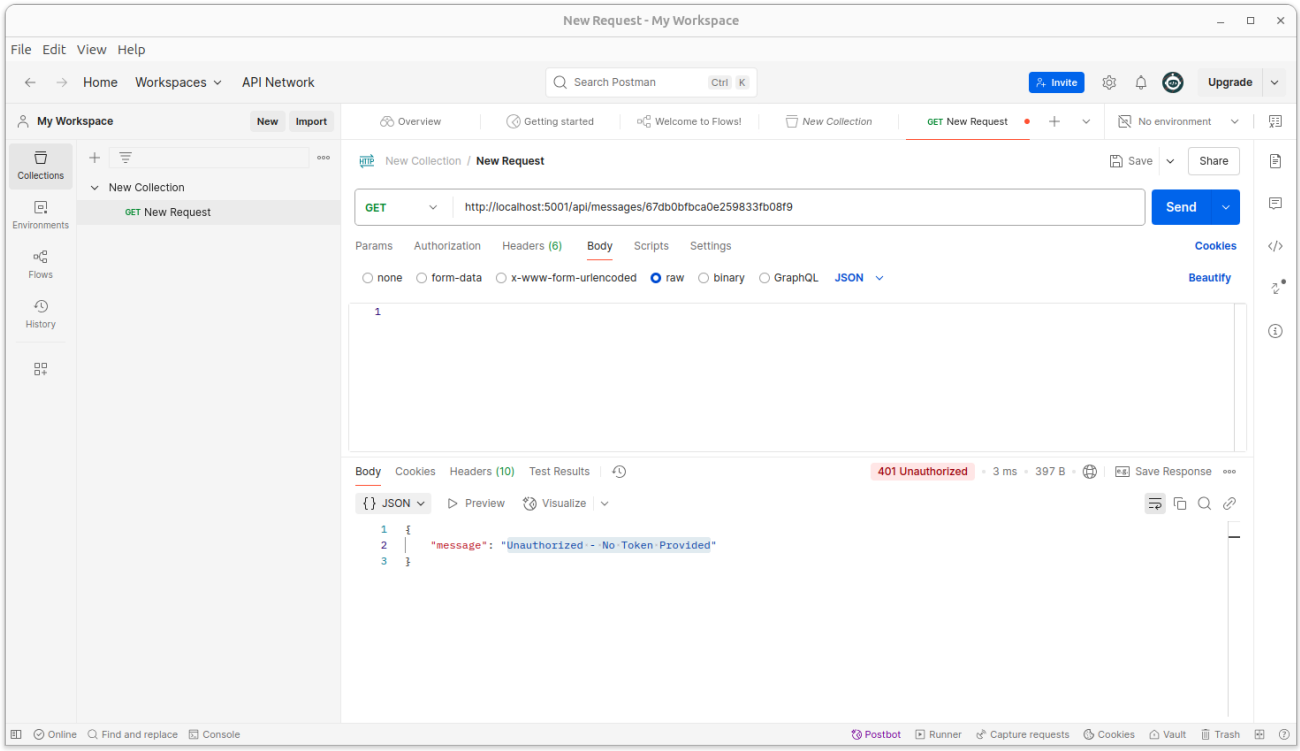
Test Date: [04/04/2025] - [04/04/2025]

Test Results: Report details provided - “” (details are not provided, rather only the messages are returned)

User not authorised - “Unauthorized - No Token Provided”

Structural Coverage: Branch Coverage 100%





38. GET /api/messages/users

The backend function to fetch all the users and their details except their passwords was tested. This function is used to fetch and display all the relevant users in the chat interface.

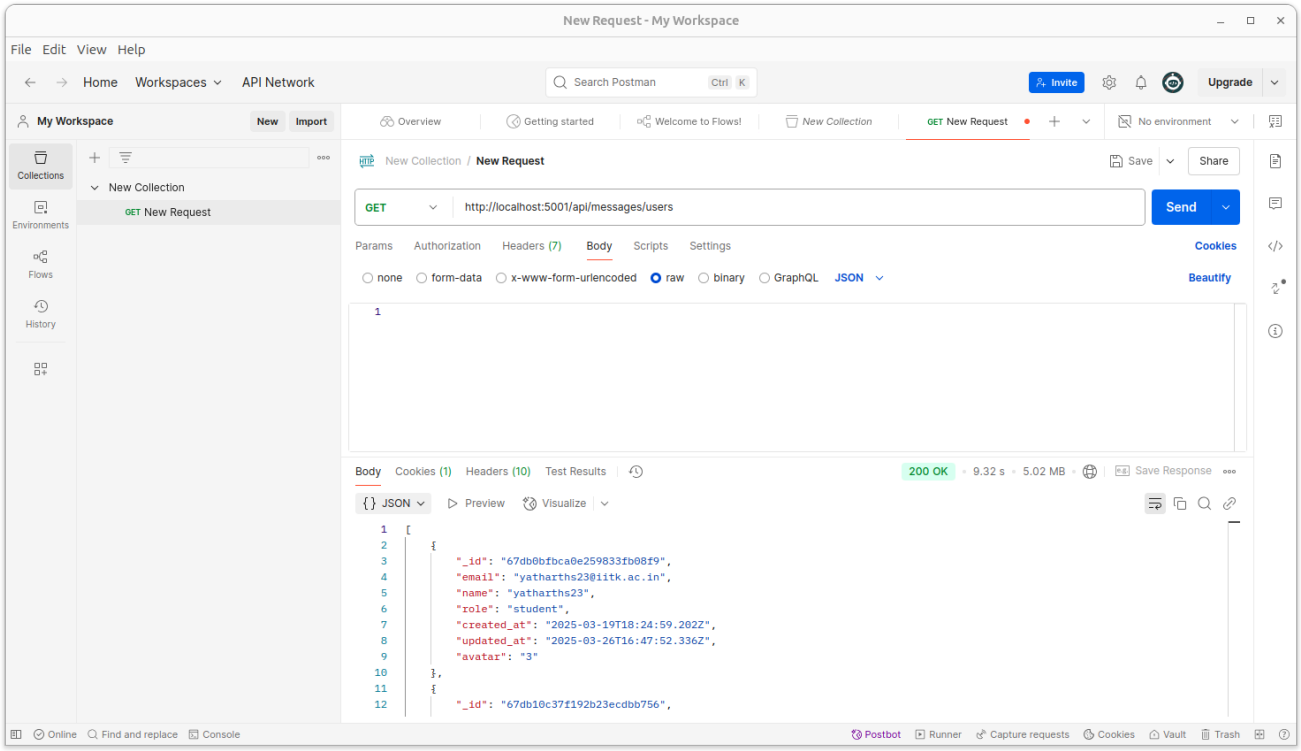
Unit Details: Backend function to handle GET Request "/api/messages/users"

Test Owner: Archita Goyal

Test Date: [04/04/2025] - [04/04/2025]

Test Results: An array of objects of users is returned.

Structural Coverage: Branch Coverage 100%



39. POST /api/messages/send/{receiver_id}

The backend function to send message to a particular user by their id

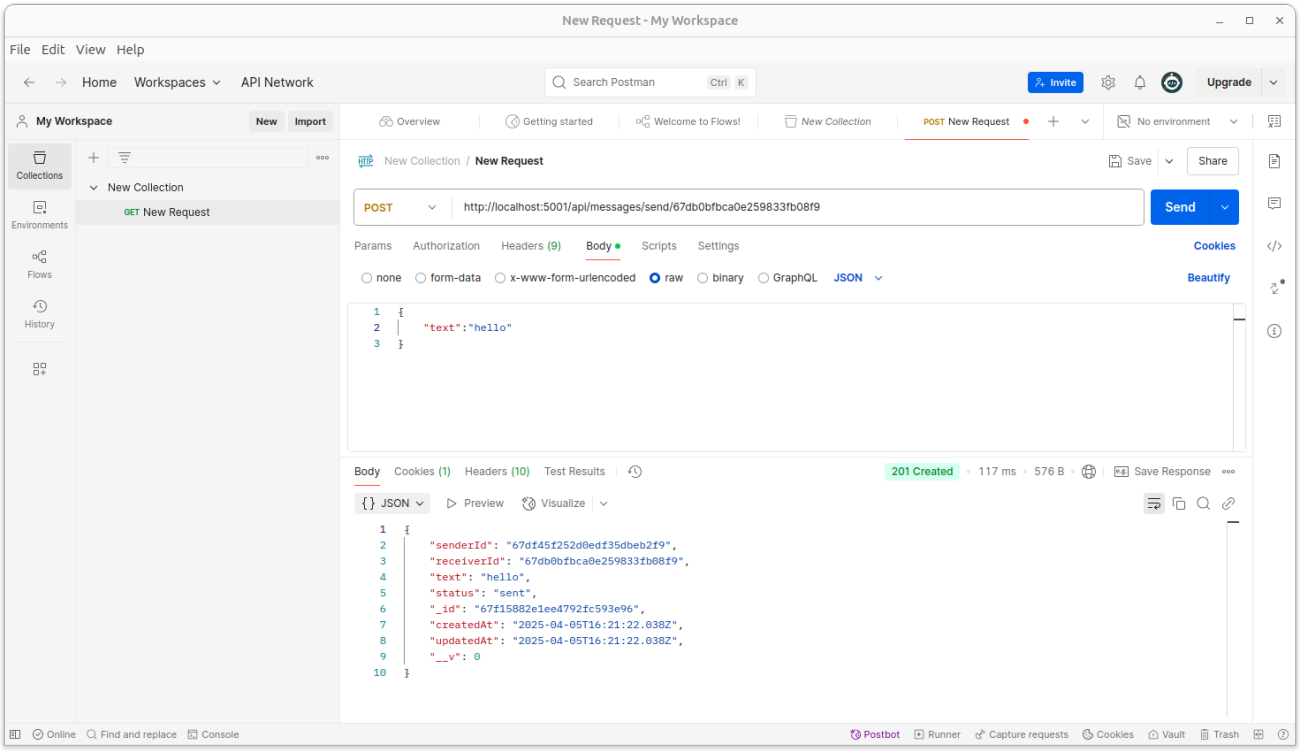
Unit Details: Backend function to handle POST Request “/api/messages/send/{receiver_id}”

Test Owner: Yatharth Sharma

Test Date: [04/04/2025] - [04/04/2025]

Test Results: A copy of the sent message object is returned

Structural Coverage: Branch Coverage 100%



40. PUT /api/messages/read/{senders_id}

The backend function to mark messages sent by a sender as read by the current user.

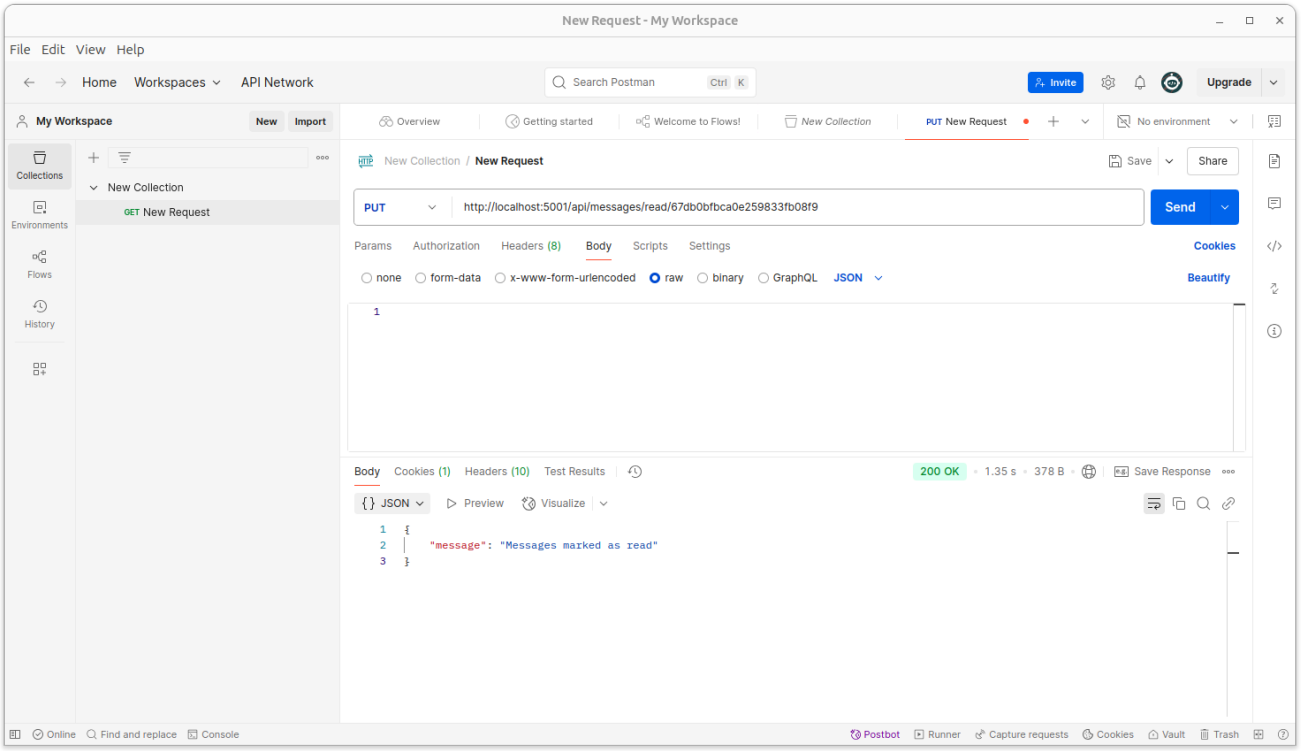
Unit Details: Backend function to handle PUT Request “/api/messages/read/{senders_id}”

Test Owner: Om Chaudhary

Test Date: [04/04/2025] - [04/04/2025]

Test Results: Positive status response- “Message marked as read”

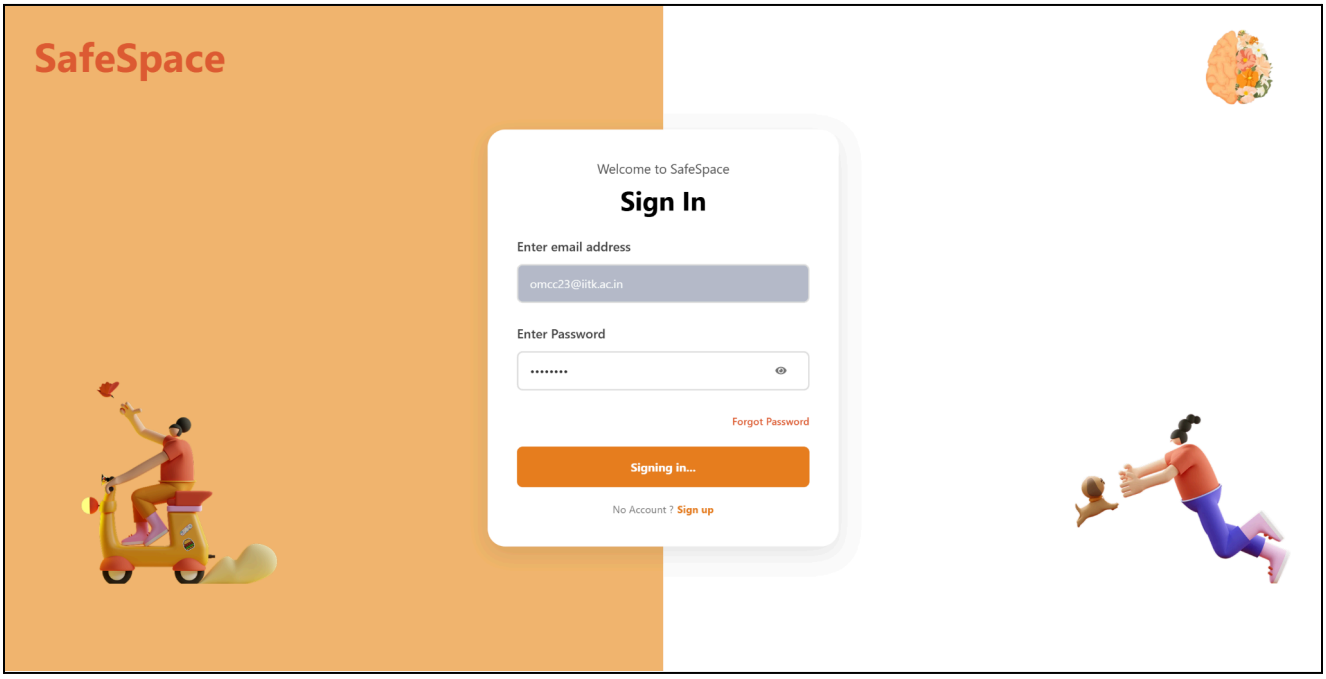
Structural Coverage: Branch Coverage 100%


New Request - My Workspace
File Edit View Help
Home Workspaces API Network
Search Postman Ctrl K
Invite
Upgrade
My Workspace
New Import
Collections
Environments
Flows
History
New Collection
GET New Request
Overview
Getting started
Welcome to Flows!
New Collection
PUT New Request
No environment
New Collection / New Request
Save
Share
PUT
http://localhost:5001/api/messages/read/67db0bfbca0e259833fb08f9
Send
Params Authorization Headers (8) Body Scripts Settings
Cookies
none form-data x-www-form-urlencoded raw binary GraphQL JSON
Beautify
1
Body Cookies (1) Headers (10) Test Results
200 OK 1.35 s 378 B
Save Response
JSON Preview Visualize
1 {
2 "message": "Messages marked as read"
3 }
Online Find and replace Console
Postbot Runner Capture requests Cookies Vault Trash

3 Integration Testing

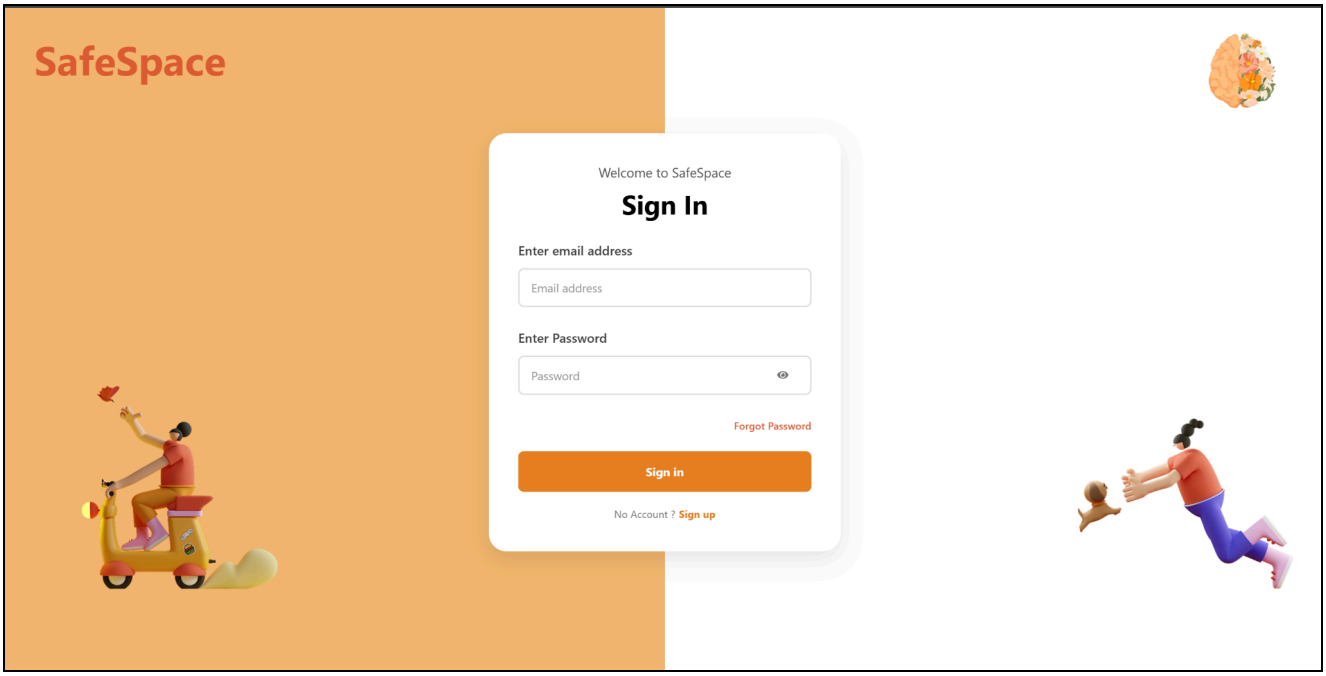
1. Checking the interface between login and Home Page for User

Module Details: login
Test Owner: Sayani Patra
Test Date: [04/02/2025] - [04/02/2025]
Test Results: When entering the correct user login credentials, the software redirects to the corresponding home page.



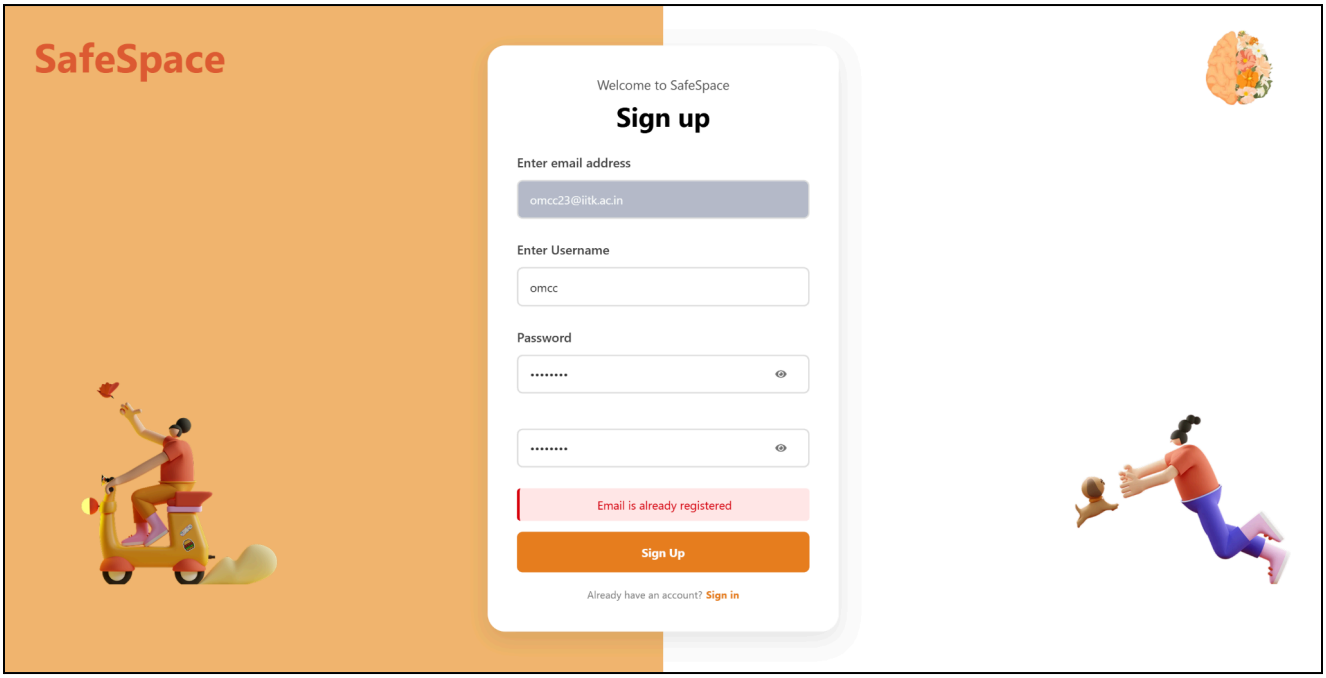
2. Logout redirects any user to the login page

Module Details: logout
Test Owner: Anirudh Singh
Test Date: [04/02/2025] - [04/02/2025]
Test Results: Test is successful for all roles (student and counsellor)



3. User cannot register with an already-used email

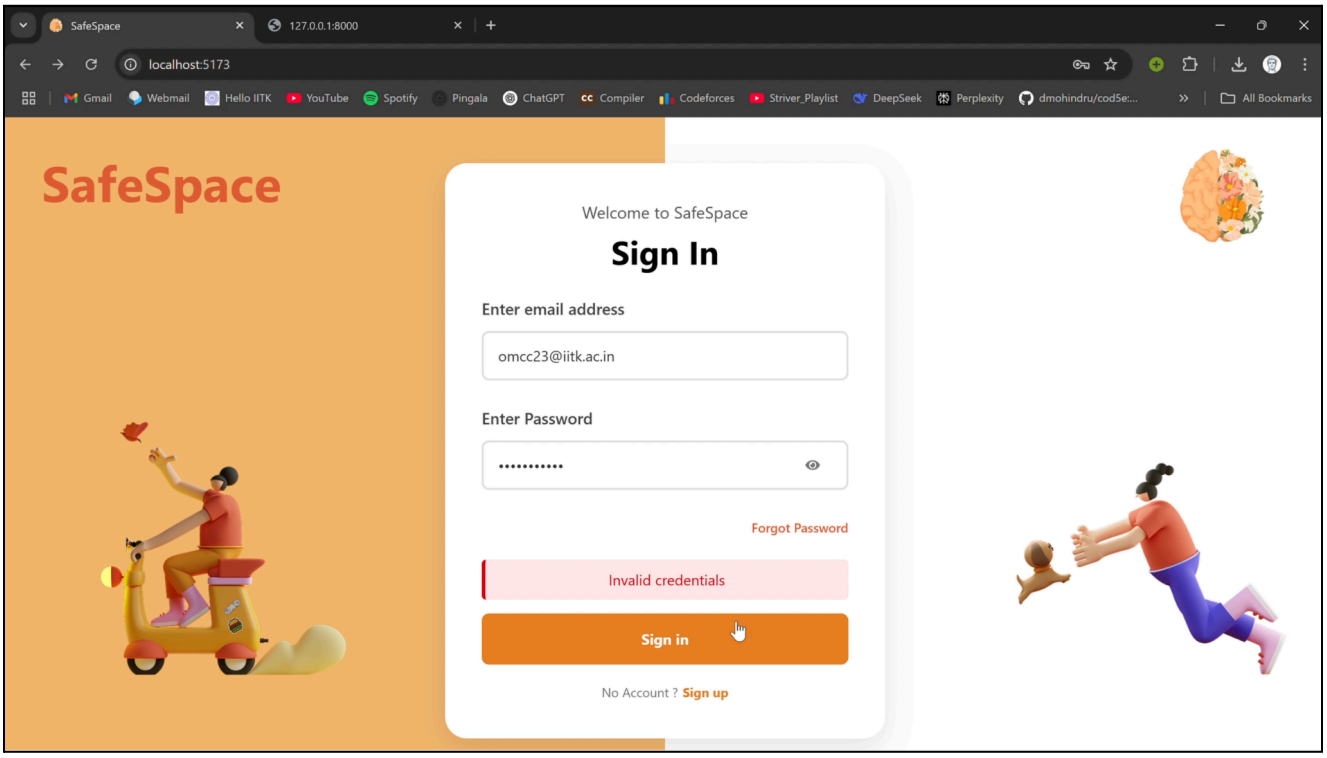
Module Details: sign-up
Test Owner: Yatharth Sharma
Test Date: [04/02/2025] - [04/02/2025]
Test Results: Registration is denied, and a prompt appears saying “Email is already registered”.



```
_id: ObjectId('67e3e2e4ade920f9ad955cde')
email : "omcc23@iitk.ac.in"
password : "$2b$12$H2jWba.vZooTOh21v88LYOMQUJHyuOrlwVu3h55qP.VC4C8yuEPM."
name : "omcc23"
role : "student"
created_at : 2025-03-26T11:20:04.613+00:00
updated_at : 2025-04-04T16:47:40.175+00:00
```

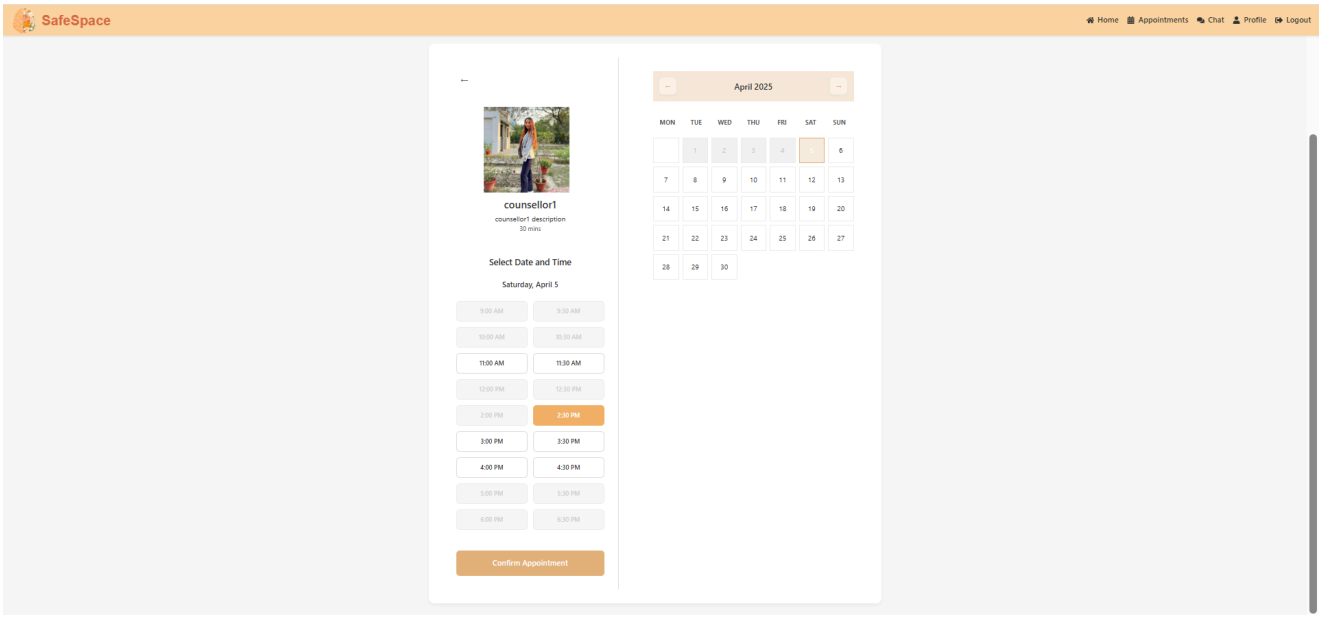

4. Incorrect credentials do not allow login

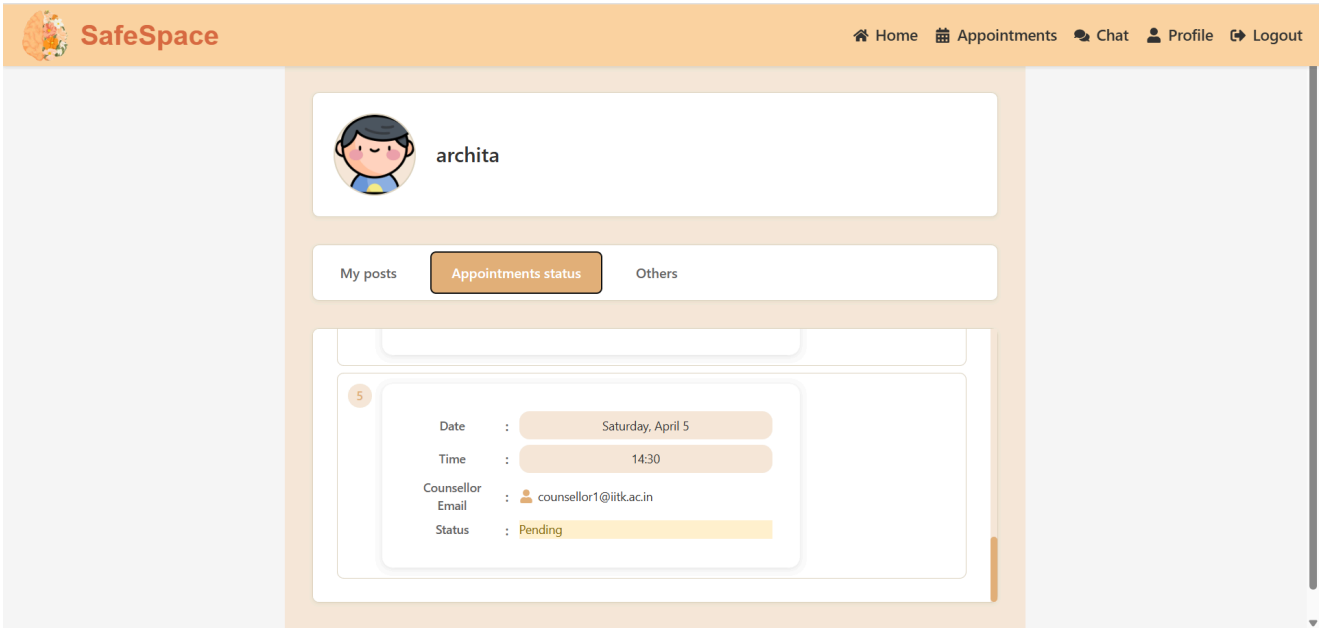
Module Details: login
Test Owner: Archita Goyal
Test Date: [04/02/2025] - [04/02/2025]
Test Results: The user was not not allowed to login with incorrect credentials and was shown a prompt saying “Invalid credentials” when tried to do so.



5. Students can book an appointment and this will be reflected under ‘Appointments Status’ tab in ‘Profile Page’

Module Details: appointments, getappointments
Test Owner: Archita Goyal
Test Date: [04/02/2025] - [04/02/2025]
Test Results: Test passed. When the “Confirm Appointment” button was clicked after selecting a particular slot, the same time slot was reflected in the profile page of the student as well as updated in the database.

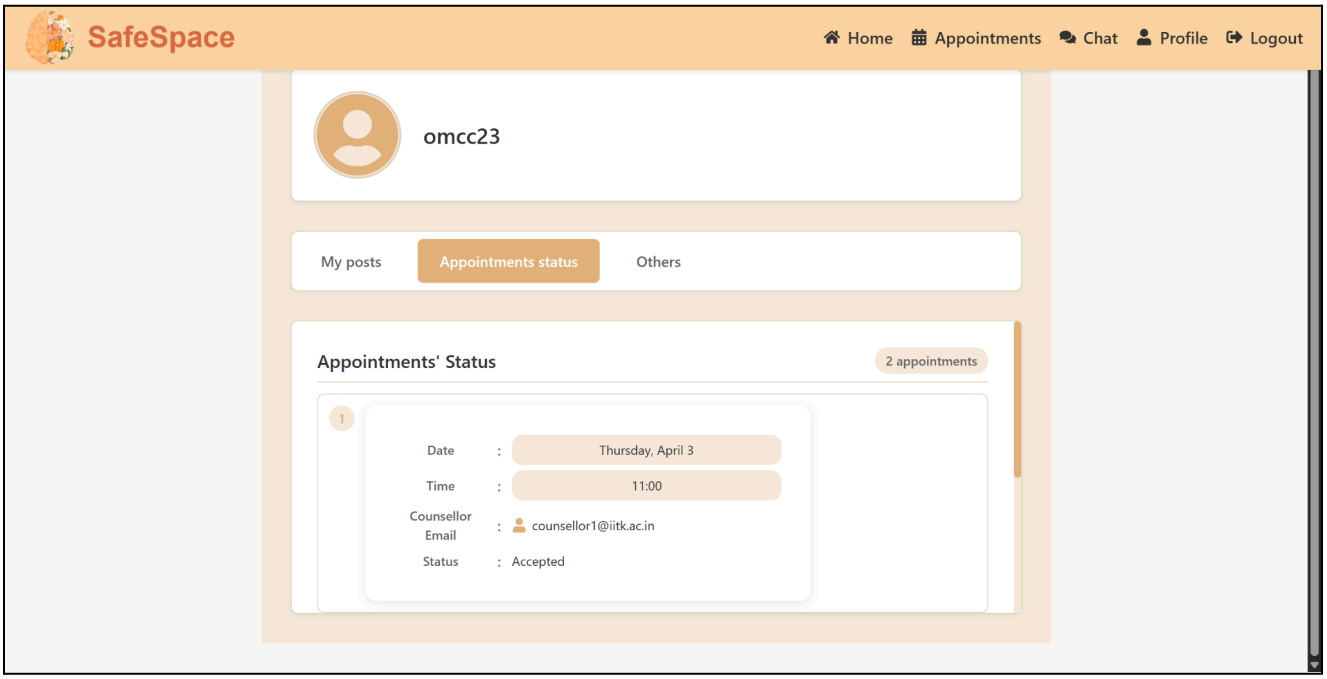




```
_id: ObjectId('67f1593025f69891f5183048')
user_name : "archita goyal"
user_email : "architag23@iitk.ac.in"
counselor_email : "counsellor1@iitk.ac.in"
date : 2025-04-05T00:00:00.000+00:00
time_slot : "14:30"
description : "I need to talk."
contact_no : "982*******"
status : "pending"
```


6. Students can view his/her appointments status

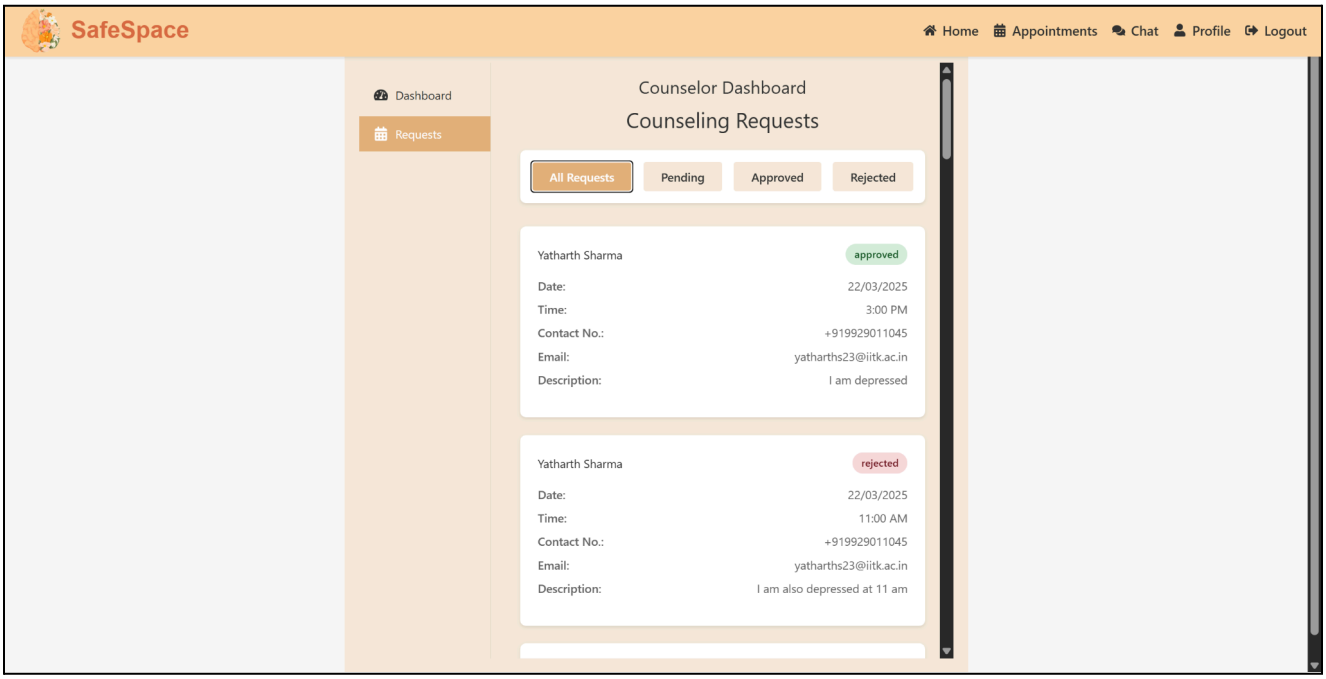
Module Details: getappointments
Test Owner: Om Chaudhari
Test Date: [03/04/2025] - [03/04/2025]
Test Results: Appointment details are shown as “pending”, “accepted” or “rejected” in ‘Appointments Status’ tab in ‘Profile Page’.



```
_id: ObjectId('67ee7893b24598334364d080')
user_name : "Om "
user_email : "omcc23@iitk.ac.in"
counselor_email : "counsellor1@iitk.ac.in"
date : 2025-04-03T00:00:00.000+00:00
time_slot : "11:00"
description : "me sed"
contact_no : "9699776686"
status : "accepted"
```


7. Counsellors can see the all the appointments

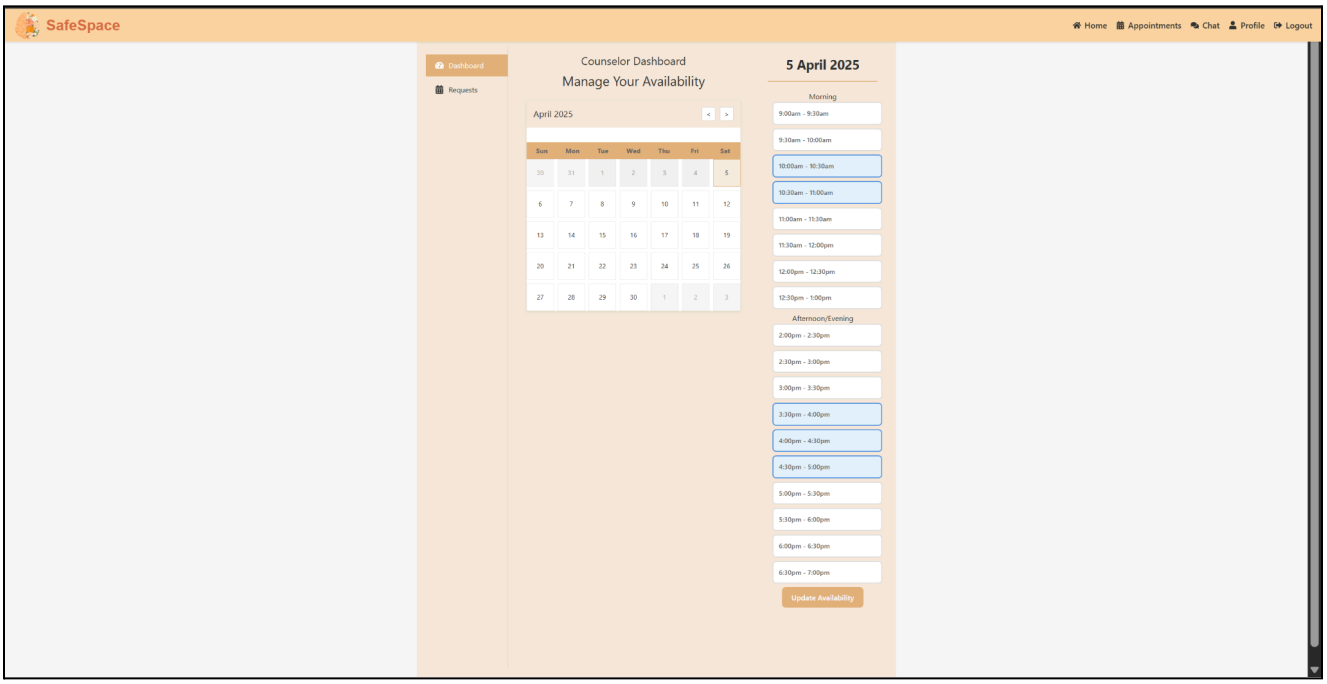
Module Details: appointment_requests
Test Owner: Nakul Patel
Test Date: [04/02/2025] - [04/02/2025]
Test Results: Appointments details are shown as “pending”, “accepted”, “rejected”, “all requests” under the ‘Requests’ tab in ‘counsellor dashboard’.



8. Counsellors can update his/her available time slots

Module Details: available_slots, update_slots
Test Owner: Naman Yadav

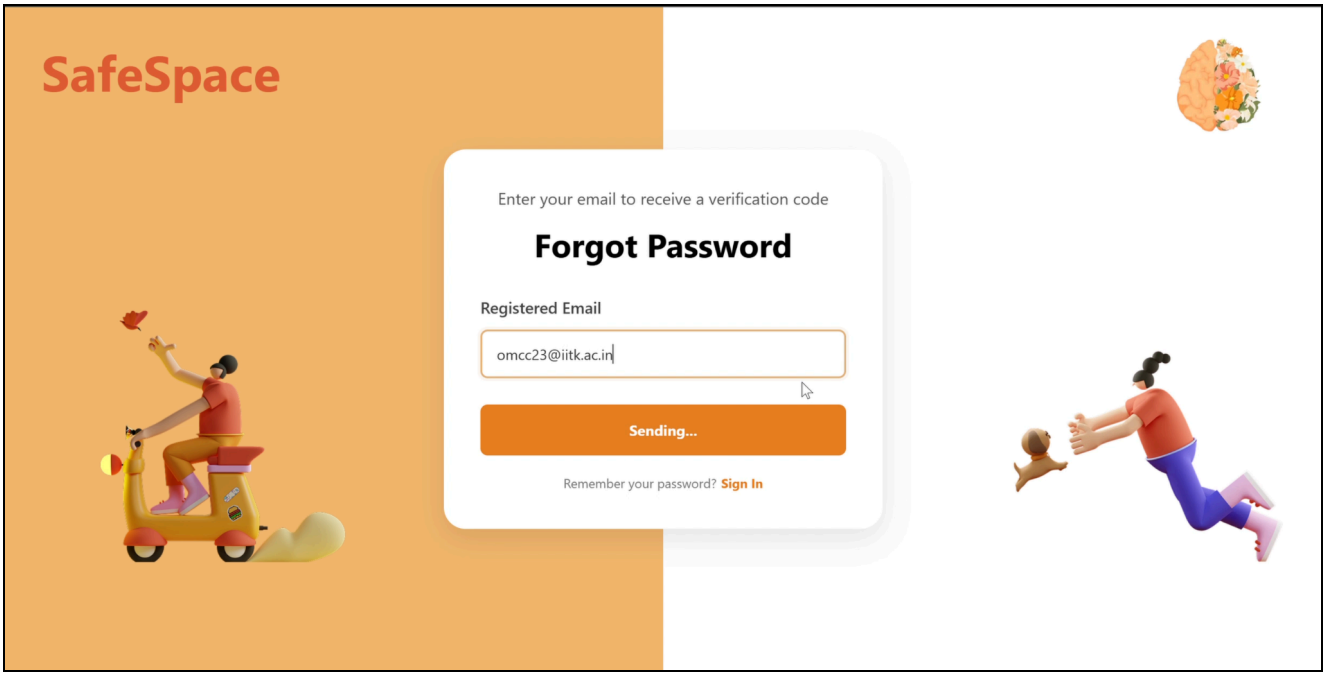
Test Date: [04/02/2025] - [04/02/2025]
Test Results: In the counsellor dashboard, one can update the availability by choosing date and time from the given calendar.

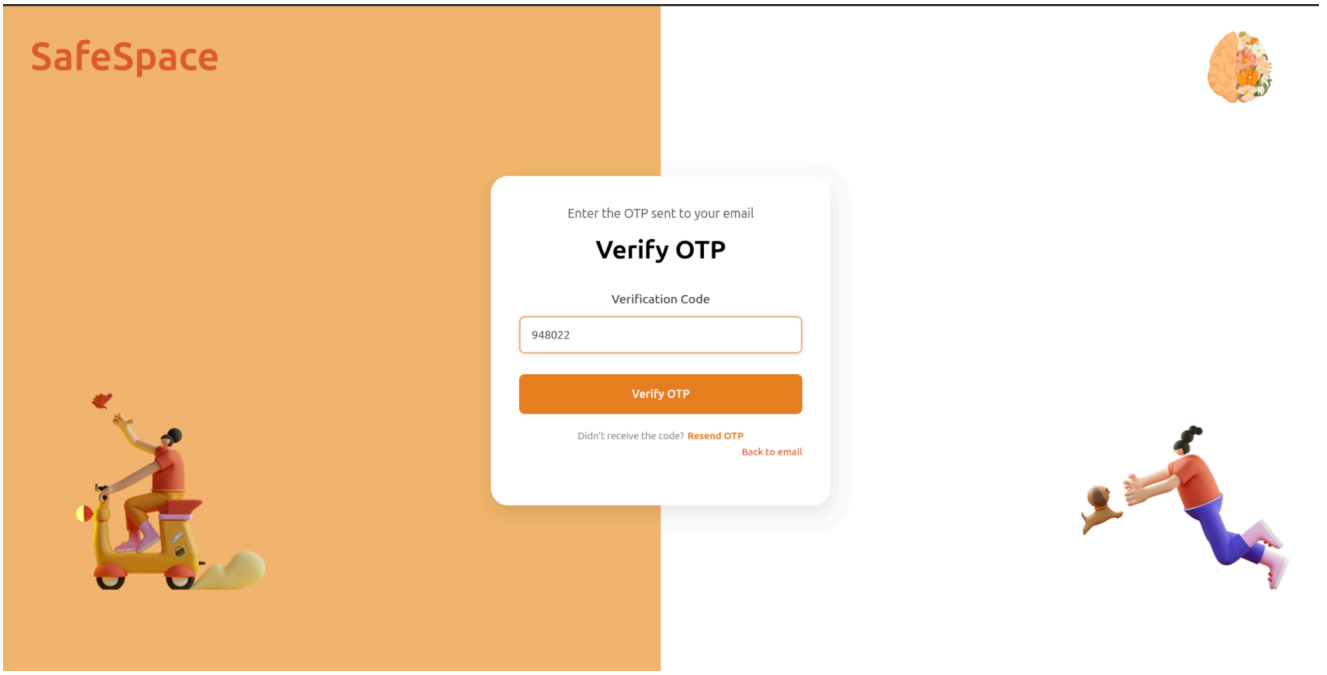


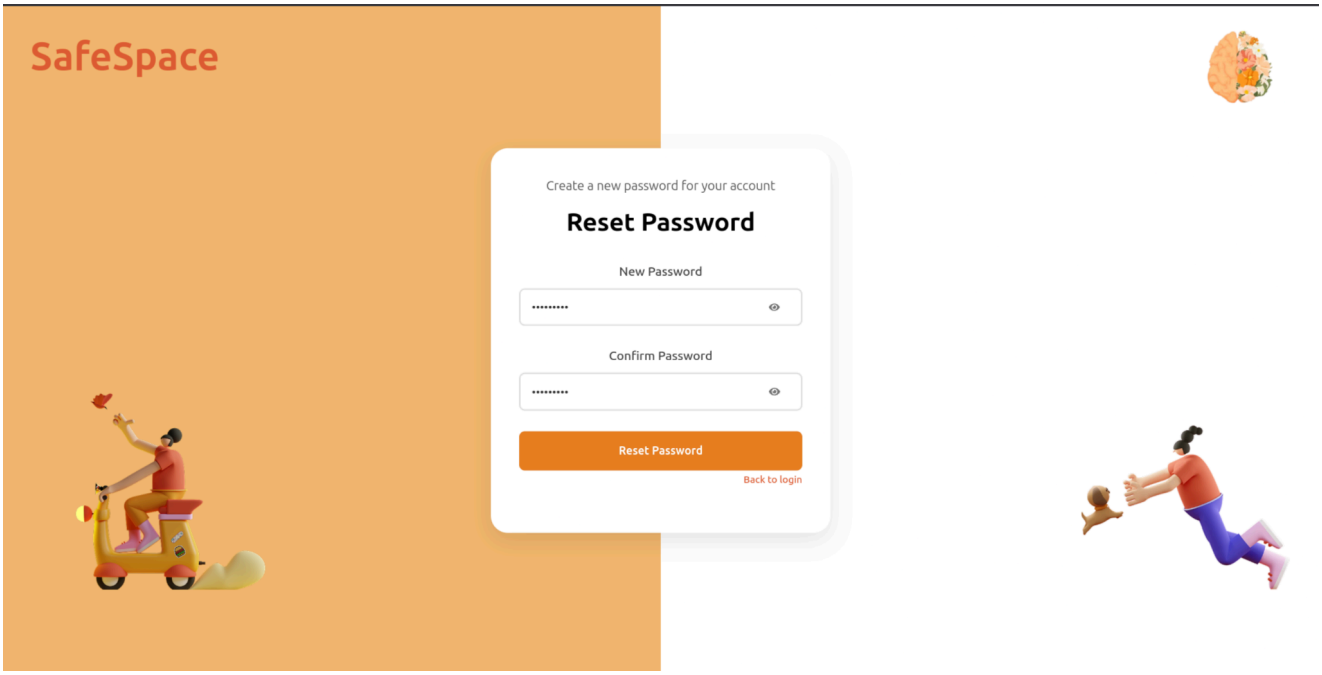

```
_id: ObjectId('67f0e55b09118eade436adb6')
counselor_email : "counsellor2@iitk.ac.in"
date : 2025-04-05T00:00:00.000+00:00
▾ time_slots : Array (5)
    0: "15:30"
    1: "16:00"
    2: "16:30"
    3: "10:00"
    4: "10:30"
```

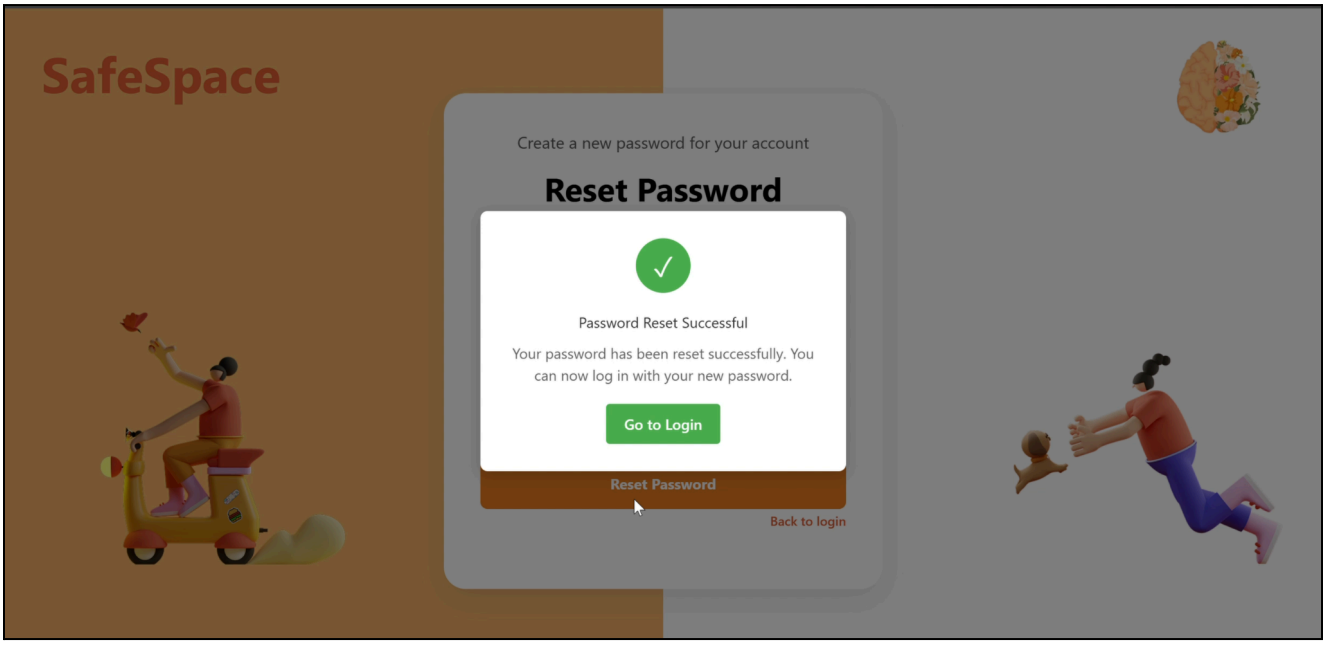

9. Registered users can reset their password

Module Details: reset-password
Test Owner: Om Chaudhari
Test Date: [04/02/2025] - [04/02/2025]
Test Results: On clicking ‘Forgot Password’, an OTP will be sent to the registered email id using which one can change their password.


SafeSpace
Enter your email to receive a verification code
Forgot Password
Registered Email
omcc23@iitk.ac.in
Sending...
Remember your password? Sign In

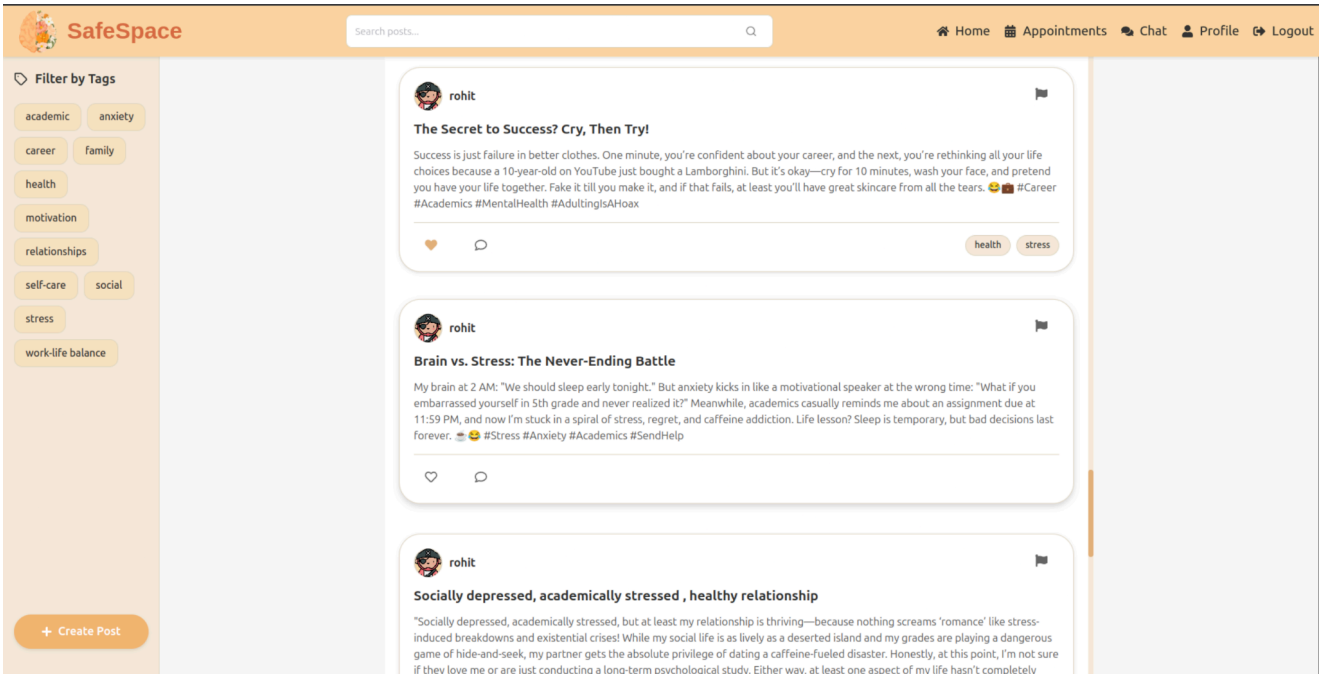

SafeSpace
Enter the OTP sent to your email
Verify OTP
Verification Code
948022
Verify OTP
Didn't receive the code? Resend OTP
Back to email





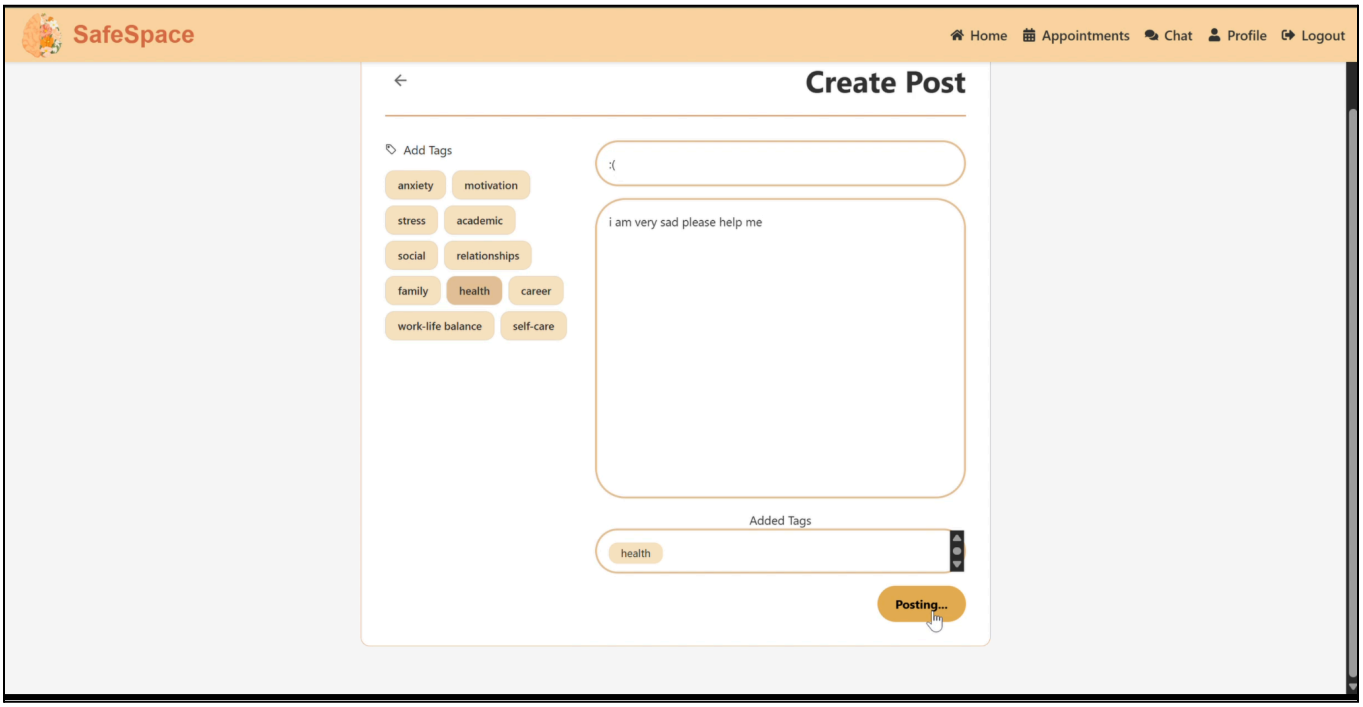
10. Users can see the posts

Module Details: see posts
Test Owner: Sayani Patra
Test Date: [04/02/2025] - [04/02/2025]
Test Results: The user can see the posts of counselors and other students on the home page.



11. Users can upload posts

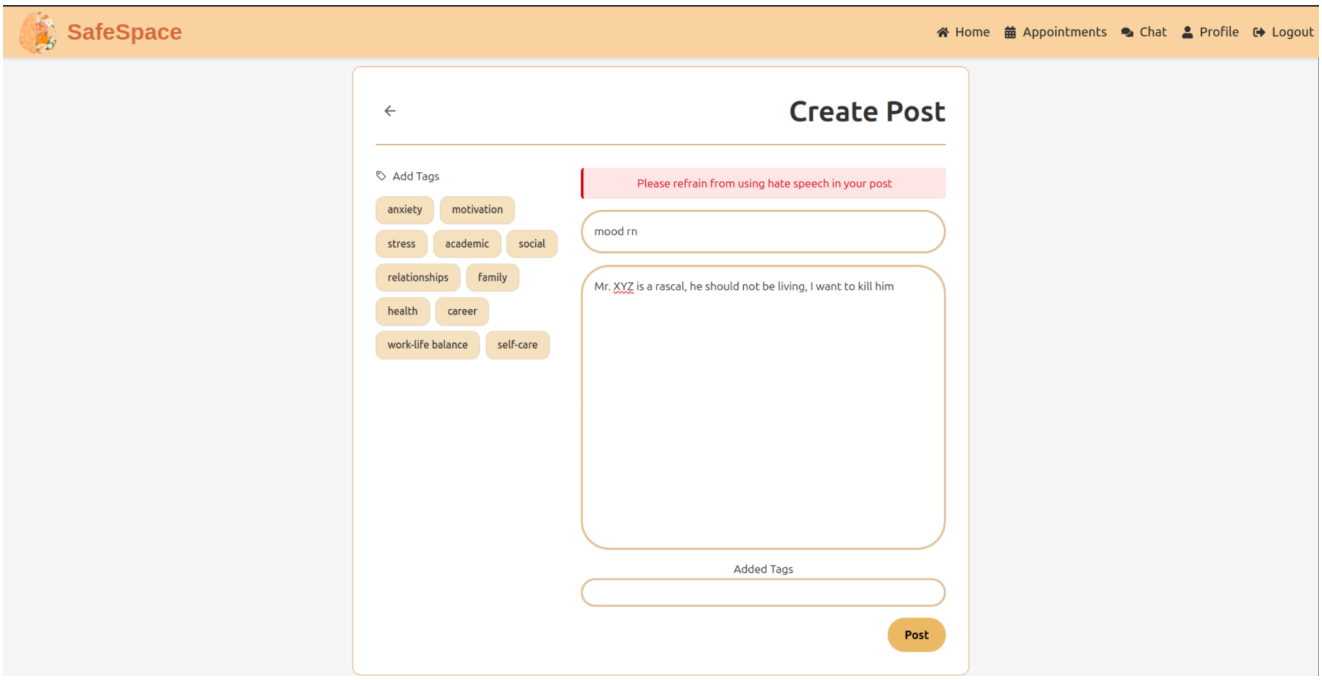
Module Details: login
Test Owner: Om Chaudhari
Test Date: [04/02/2025] - [04/02/2025]
Test Results: The user: 1) student: can upload posts by clicking on create button and then entering text in it. 2) counsellor: can also post images, videos along with the text.



```
_id: ObjectId('67f00efe1e6ce5b4df2d1d77')
title : ":("
content : "i am very sad please help me"
author : "omcc23"
author_id : "67e3e2e4ade920f9ad955cde"
▸ relevance_tags : Array (1)
severity_tag : "moderate"
image : null
created_at : 2025-04-04T16:55:26.263+00:00
updated_at : 2025-04-04T16:58:10.701+00:00
likes : 2
image_url : null
```

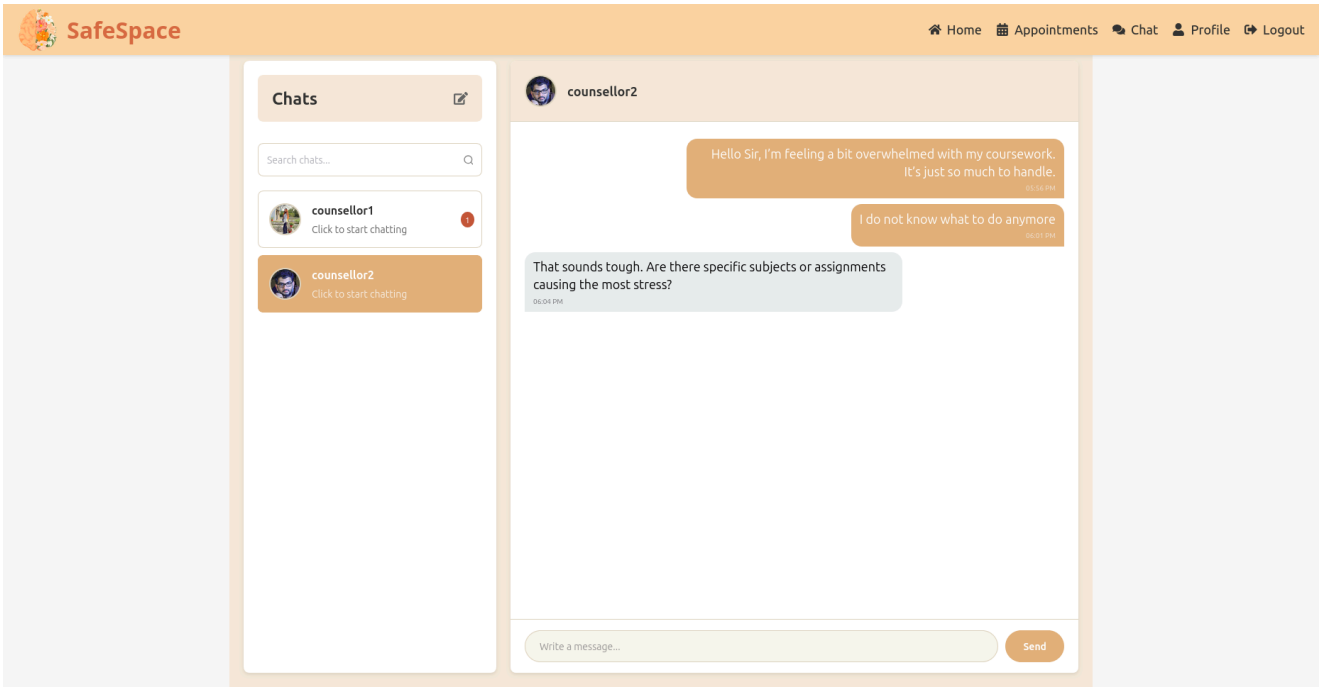

12. Users will be restricted from using foul language while creating posts.

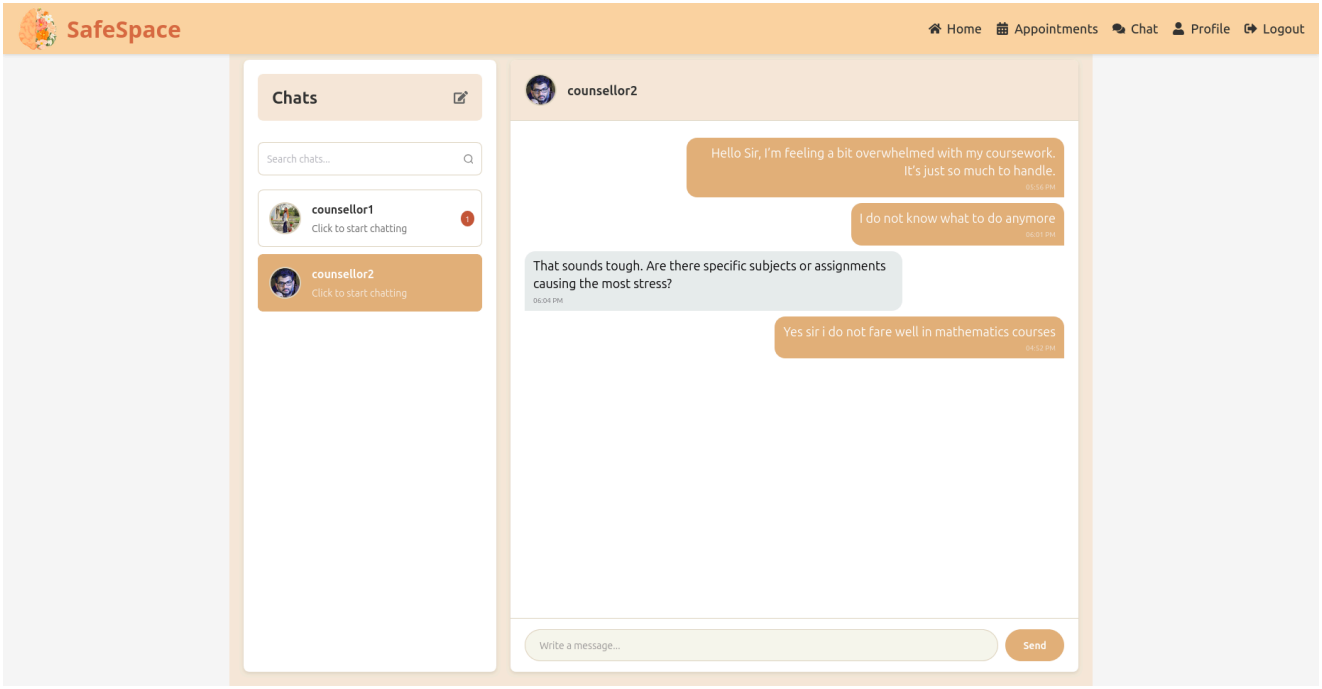
Module Details: posts
Test Owner: Vivek
Test Date: [04/02/2025] - [04/02/2025]
Test Results: If a user or counselor attempts to create a post containing hateful content or offensive language, the system automatically prevents the submission.

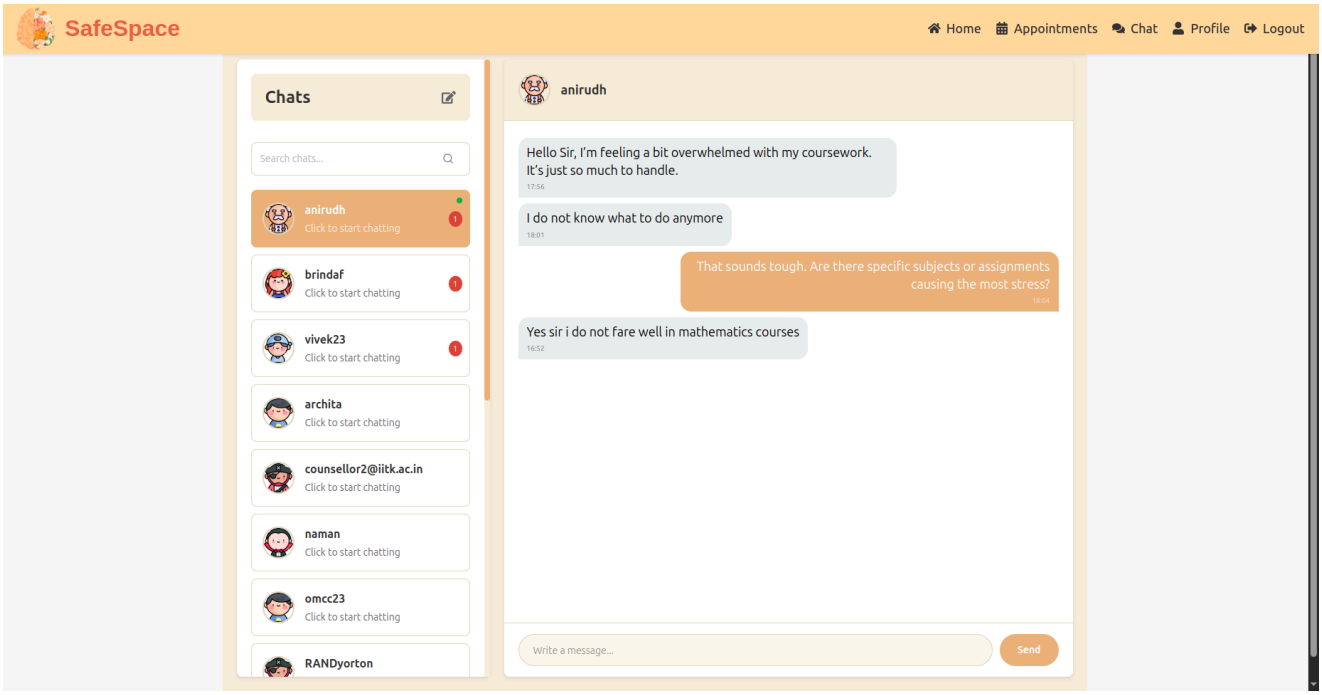


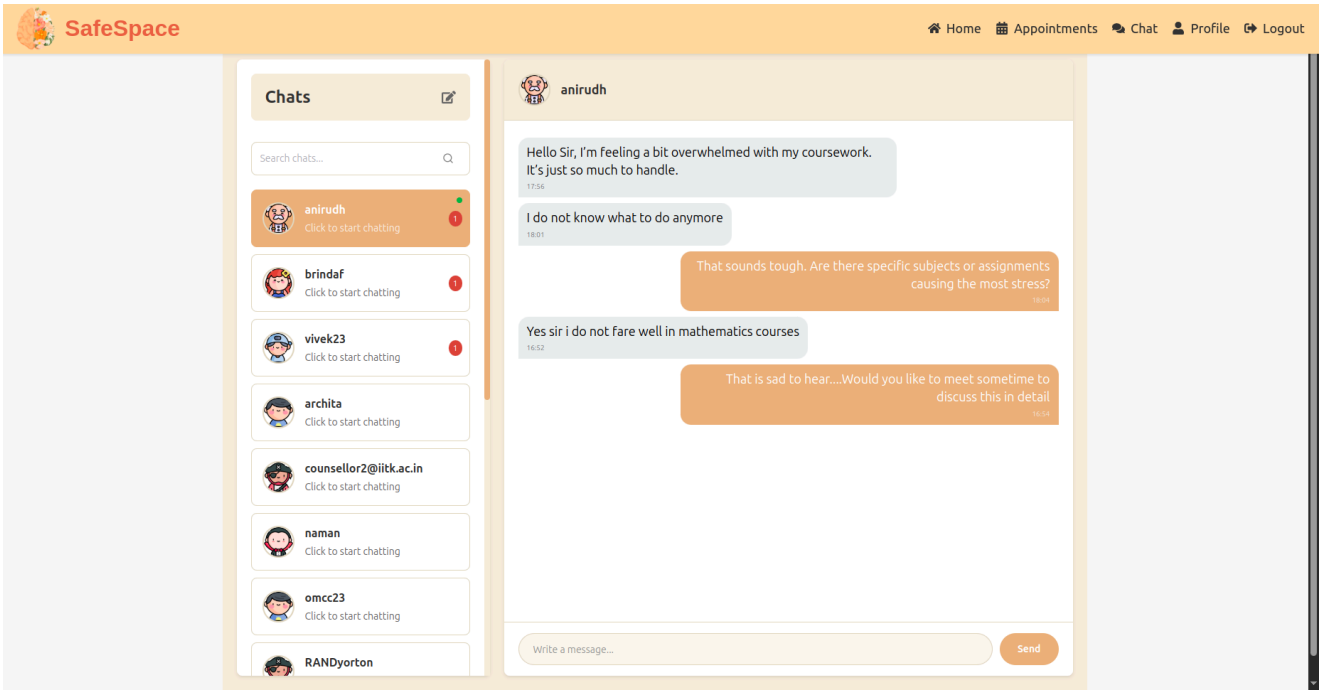
13. Students can Chat with the Counsellors and vice-versa.

Module Details: Chats
Test Owner: Anirudh Sngh
Test Date: [04/02/2025] - [04/02/2025]
Test Results: A student can send messages to a counsellor to seek guidance While a counsellor can receive the said messages and reply to them , thus initiating a chat.




SafeSpace
Home
Appointments
Chat
Profile
Logout
Chats
Search chats...
counsellor1
Click to start chatting
counsellor2
Click to start chatting
counsellor2
Hello Sir, I'm feeling a bit overwhelmed with my coursework. It's just so much to handle.
I do not know what to do anymore
That sounds tough. Are there specific subjects or assignments causing the most stress?
Yes sir i do not fare well in mathematics courses
Write a message...
Send


SafeSpace
Home
Appointments
Chat
Profile
Logout
Chats
Search chats...
anirudh
Click to start chatting
brindaf
Click to start chatting
vivek23
Click to start chatting
archita
Click to start chatting
counsellor2@iitk.ac.in
Click to start chatting
naman
Click to start chatting
omcc23
Click to start chatting
RANDyorton
anirudh
Hello Sir, I'm feeling a bit overwhelmed with my coursework. It's just so much to handle.
I do not know what to do anymore
That sounds tough. Are there specific subjects or assignments causing the most stress?
Yes sir i do not fare well in mathematics courses
Write a message...
Send




```
_id: ObjectId('67f263fdd6677ed9ff230b65')
senderId : ObjectId('67db1278e480b17b41bd33aa')
receiverId : ObjectId('67dc4cb24145ca175f0b21b2')
text : "Yes sir i do not fare well in mathematics courses"
status : "read"
createdAt : 2025-04-06T11:22:37.032+00:00
updatedAt : 2025-04-06T11:23:06.422+00:00
__v : 0
```


QUERY RESULTS: **1-1 OF 1**


```
_id: ObjectId('67f2645ed6677ed9ff230bac')
senderId : ObjectId('67dc4cb24145ca175f0b21b2')
receiverId : ObjectId('67db1278e480b17b41bd33aa')
text : "That is sad to hear....Would you like to meet sometime to discuss this…"
status : "sent"
createdAt : 2025-04-06T11:24:14.182+00:00
updatedAt : 2025-04-06T11:24:14.182+00:00
__v : 0
```

4 System Testing

1. **Requirement:** User can create account using the sign-up page

Test Owner: Archita Goyal
Test Date: [02/03/2025] - [02/03/2025]
Test Results: Test was successfully conducted. For testing, users had to click on the ‘sign up’ button on the login page. When redirected to the registration page, filled up the registration form. Registration was only successful using institute mail as no otp will be received otherwise. After entering otp, registration was completed. The student or counsellor was then able to login to the system using login credentials entered in the registration form.
Additional Comments:We are considering allowing both students and counselors to register in the same way. Counsellors then have to further mail to SafeSpace to change the role.

2. **Requirement:** User can login using the login page

Test Owner: Archita Goyal
Test Date: [02/03/2025] - [02/03/2025]
Test Results: Test was successfully conducted. For testing, the user had to enter login credentials on the login page and click on the ‘login’ button. If the credentials were the same as those entered in the registration form, the user was then able to login and redirected to the home page. An error message was displayed otherwise.

3. **Requirement:** Forgot Password button present on the login page and Reset Password button in the profile page.

Test Owner: Archita Goyal
Test Date: [02/03/2025] - [02/03/2025]
Test Results: Test was successfully conducted. For testing, users had to click on the forgot password button on the login page that had redirected him to the forgot password page, where he had to enter the institute mail ID and was able to change password after verifying the OTP sent on that ID. The user was also able to change the password by clicking on “Reset Password” button in the “Others” section of the “Profile” tab.
Additional Comments: Both students and counselors can reset their passwords.

4. **Requirement:** The homepage will feature a navigation bar containing links to key sections such as Chat, User Profile, Appointments, Homepage and also a search bar.

Test Owner: Archita Goyal
Test Date: [02/03/2025] - [03/03/2025]
Test Results: Test passed. A working navbar was present at the top of the webpage and contained the options to navigate to the Appointments tab, chat section, profile page, homepage and a button to logout of the application.
Additional Comments: All the buttons worked as expected.

5. **Requirement:** The user can edit his/her current avatar through the profile page.

Test Owner: Archita Goyal

Test Date: [01/04/2025] - [01/04/2025]
Test Results: Test was successfully conducted. For testing, users had to navigate to the profile page and click on the avatar and a designated box of available avatars . User was able to click on any of them and the avatar was updated.
Additional Comments: Counsellors have profile pictures additionally. Also, banner images have not been added in this version.

6. **Requirement:** A student can directly chat with any of the counsellors but not with any other student in the chat section.

Test Owner: Rohit Yadav
Test Date: [03/04/2025] - [03/04/2025]
Test Results: Test was successfully conducted. For testing,students can navigate to the chat page from the navigation bar where only counselors are available to chat on that page that ensures that no students can chat with other students maintaining the anonymity.This was verified by login as counsellor and the chat page shows the message sent by the student. Additionally, counselors can chat with any registered students .

7. **Requirement:** Users can create posts containing: Title, Body text, and Relevant tags. Users can publish blogs immediately, save blogs as drafts. Counselors can also include images in their posts.

Test Owner: Archita Goyal
Test Date: [03/04/2025] - [03/04/2025]
Test Results: The test was successfully conducted. For testing, users had to click the “Create Post” button on the bottom left of the home page. There was an interface to write the content of a blog including its heading,content and attach the relevant tags. There is an additional feature of adding images which appears only when a counsellor tries to create a post.
Additional Comments: The user can delete past posts but not edit them or save them as drafts in this version.

8. **Requirement:** Users can book an appointment with available counsellors and choose a slot based on the counsellor’s specified availability.

Test Owner: *Rohit Yadav*
Test Date: [01/04/2025] - [01/04/2025]
Test Results: Test was successfully conducted. For testing, users had to select a counsellor from the available list on the appointments page. Available slots for the selected counsellor appeared on selecting a date. The suitable slot was selected and clicked “Confirm Appointment” redirecting to a form to fill personal details and description of the reason for booking the appointment. Once the details were filled and the “Confirm” button was clicked, the user was able to see the appointment status in their profile page.

9. **Requirement:** A student can see the current status (accepted,rejected,pending) of all the appointments requested by him to any counsellor.

Test Owner: Rohit Yadav
Test Date: [01/04/2025] - [01/04/2025]

Test Results: Test was successfully conducted. For testing, users had to click on the profile page button on navbar on the home page. When redirected to the profile page, click on appointments status button to see a designated box for list of past booked appointments with their status.

10. Requirement: Counsellors can accept/reject an appointment request received from a student.
Test Owner: *Rohit Yadav*
Test Date: [01/04/2025] - [01/04/2025]
Test Results: Test was successfully conducted. For testing, the counsellor had to select “Requests” tab on the appointments page and when redirected to the requests page, selected the “Pending Requests” and clicked on “Accept” or “Reject” for a particular request. The status of that request was accordingly updated in the profile page of that student as well as in the appointment requests of the counsellor.
Additional Comments: The counsellor can initiate chat with the student by searching for name in the chat section after rejecting the request if they wish to.

11. Requirement: Counsellor can choose his/her availability in a particular slot on a particular day.
Test Owner: Rohit Yadav
Test Date: [01/04/2025] - [01/04/2025]
Test Results: Test was successfully conducted. The counsellor navigated to the appointments page, selected a specific day using the calendar in the dashboard and updated available time slots for that day. This update was then verified by logging in as a student and confirming the availability of the selected slots for that counsellor.

12. Requirement: Users can like posts and add their comments to different users’ posts.
Test Owner: Archita Goyal
Test Date: [01/04/2025] - [01/04/2025]
Test Results: Test was successfully conducted. The user navigated to the specific post they liked and clicked the heart-shaped button beneath the post. To comment, the user clicks on the comment icon and inputs a text. This was verified by checking the updated number of likes on the post and checking if the comment was added in the comment section of the post.

13. Requirement: Users can search specific posts.
Test Owner: Archita Goyal
Test Date: [03/04/2025] - [03/04/2025]
Test Results: Test was successfully conducted. The user navigated to the search bar in the top navigation bar. Then, the user inputs the title or tag names of desired posts. All the posts which matched the regular expressions of the title and the user-inputted text and the posts which contain the tags.

14. Requirement: Users can search specific counsellors by their names.
Test Owner: Nakul Patel
Test Date: [03/04/2025] - [03/04/2025]
Test Results: Test was successfully conducted. The user navigated to the search bar in the top in the chats window.

15. Requirement: User cannot sign up with an already registered email.
Test Owner: Archita Goyal
Test Date: [03/04/2025] - [03/04/2025]
Test Results: Test was successfully conducted. When the user tried to sign-up with an already registered email, he was not allowed to do that and an error message was displayed showing “Email is already registered”.

16. Requirement: User cannot log in with incorrect credentials.
Test Owner: Archita Goyal
Test Date: [04/04/2025] - [04/04/2025]
Test Results: Test was successfully conducted. When the users tried to login with incorrect username or password, they were not allowed to do that and an error message was displayed showing “Invalid Credentials”.

17. Requirement: OTP expires after a certain time.
Test Owner: Archita Goyal
Test Date: [04/04/2025] - [04/04/2025]
Test Results: Test was successfully conducted. When the user entered the OTP (generated during signup or changing password) after 10 minutes, it was no longer valid and needed to regenerate a new OTP.

18. Requirement: Message cannot be sent with an empty body.
Test Owner: Archita Goyal
Test Date: [04/04/2025] - [04/04/2025]
Test Results: Test was successfully conducted. The user was not allowed to send a message with only spaces or an empty string.

19. Requirement: Message delivery status updates correctly
Test Owner: Archita Goyal
Test Date: [04/04/2025] - [04/04/2025]
Test Results: Test was successfully conducted. The number of unread messages was correctly displayed along with the name of every user and vanished on opening the chat(i.e, the messages were marked as seen).

5 Conclusion

As far as we can see, the testing was sufficiently exhaustive and we do not expect a significant number of errors to arise if the system were deployed.

The number of bugs we found and corrected was more than ___, indicating that the testing was effective.

The testing process can be improved by having a larger number of students in the team and dividing the team into groups, so as to reduce the amount of communication between the developers and testers of each feature.

There is also a conflict of interest in the testing process if the testers and developers are part of the same group, therefore we tried to keep the two teams separate, and no developer tested a feature that they had developed. Separating the two groups and the assessments of their work would incentivise testers to think as end-users rather than custodians of the software. In fact, this is standard practice in the certification of safety-critical software, i.e. independent regulators certify systems as safe.

Giving more time to the testing process would also have improved the testing process. However, as we had limited time, implementation was given priority, with as exhaustive testing as possible.

Appendix A - Group Log

Sr. No.	Date	Agenda
1	31/03/25	First meet for the discussion for Testing and Manual Document. Work distribution done.
2	01/04/25	Manual System Testing was done and doubts discussed on User Manual
3	02/04/25	Initiated component testing. User Manual reviewed
4	03/04/25	Initiated integration testing. User Manual was finalized
5	04/04/25	User Manual Doc submitted and Testing Doc was Finalized
6	05/04/25	Testing Doc finalized and Submitted