SOFTWARE ENGINEERING

(Course ID: BSCS3001)

Online Ticketing System with Discourse and Gchat Integration

Milestone 5

By

Anupam Kumar Jha (21f1004905)

Kevin Joshua T (22f1001410)

Pranav Wankhedkar (21f1000120)

Sachin Singh (21f1003251)

Ujit Kumar (21f3000786)

Utpal Dutta (21f2000524)

Milestone 5 : API Endpoints, Test Cases and Results

API being tested	API Description	Input	Expected Output	Actual Output	Result
/api/send_gchat_msg	Send message to Gchat if priority is beyond 5	Payload: {ticket_id: 3, priority_measure: 6, message: "...", webhook_url: ""}	Status code: 200, Message: "Message posted successfully."	Status code: 200, Message: "Message posted successfully."	Success
/api/send_gchat_msg	Send message to Gchat if priority is beyond 5	Payload: {priority_measure: 6, message: "...", webhook_url: "https://example.com/gchat/webhook"}	Status code: 400, Message: "Invalid request body..."	Status code: 400, Message: "Invalid request body..."	Success
/api/send_gchat_msg	Send message to Gchat if priority is beyond 5	Payload: {ticket_id: 3, priority_measure: 6, message: "...", webhook_url: "https://invalid-url"}	Status code: 500, Message: "Internal server error..."	Status code: 500, Message: "Internal server error..."	Success
/check_flagged_posts	Check for flagged posts and send message to Support Manager	Payload: {post_id: 123, flag_message: "...", user_id: 456, message: "...", webhook_url: "..."}	Status code: 200, Message: "Message sent to Support Manager..."	Status code: 200, Message: "Message sent to Support Manager..."	Success

/check_flagged_posts	Check for flagged posts and send message to Support Manager	Payload: {flag_message: "...", user_id: 456, message: "...", webhook_url: ""}	Status code: 400, Message: "Invalid request body..."	Status code: 400, Message: "Invalid request body..."	Success
/check_flagged_posts	Check for flagged posts and send message to Support Manager	Payload: {post_id: 123, flag_message: "...", user_id: 456, webhook_url: "https://invalid-url"}	Status code: 500, Message: "Internal server error..."	Status code: 500, Message: "Internal server error..."	Success
/submit_previous_discussion_links	Submit clickable links for a previous discussion.	Payload: {post_id: 123, discussion_links: ["...", "..."]}	Status code: 200, Message: "Previous discussion links submitted successfully."	Status code: 200, Message: "Previous discussion links submitted successfully."	Success
/submit_previous_discussion_links	Submit clickable links for a previous discussion.	Payload: {discussion_links: ["...", "..."]}	Status code: 400, Message: "Invalid request body..."	Status code: 400, Message: "Invalid request body..."	Success
/submit_previous_discussion_links	Submit clickable links for a previous discussion.	Payload: {post_id: 123, discussion_links: ["https://invalid-url"]}	Status code: 500, Message: "Internal server error..."	Status code: 500, Message: "Internal server error..."	Success
/api/search_discussions_by_tag	Search discussions based on a tag.	Tag: "python"	Status code: 200, Error message: None	Status code: 200	Success

/api/search_discussions_by_tag	Search discussions based on a tag.	Tag: "java"	Status code: 404, Error message: "No discussions found for the provided tag."	Status code: 404, Error message: "No discussions found for the provided tag."	Success
/api/search_discussions_by_tag	Search discussions based on a tag.	Tag: ""	Status code: 400, Error message: "Invalid request. Please provide a valid tag."	Status code: 400, Error message: "Invalid request. Please provide a valid tag."	Success
/api/search_discussions_by_tag	Search discussions based on a tag.	Tag: "invalid_tag"	Status code: 400, Error message: "Invalid request. Please provide a valid tag."	Status code: 400, Error message: "Invalid request. Please provide a valid tag."	Success
/profile	Provides details of a specific user by taking user id as parameter.	User ID: 1001	Status code: 200, Message: None	Status code: 200	Success
/profile	Provides details of a specific user by taking user id as parameter.	User ID: None	Status code: 403, Error message: "Please provide a user id."	Status code: 403, Error message: "Please provide a user id."	Success
/profile	Provides details of a specific user by taking user id as parameter.	User ID: 9999	Status code: 404, Error message: "There are no user with user id 9999"	Status code: 404, Error message: "There are no user with user id 9999"	Success
/api/search_discussions_by_category	Filter based on category and tag.	Category: "python", Tag: "quiz-2"	Status code: 200, Message: None	Status code: 200	Success

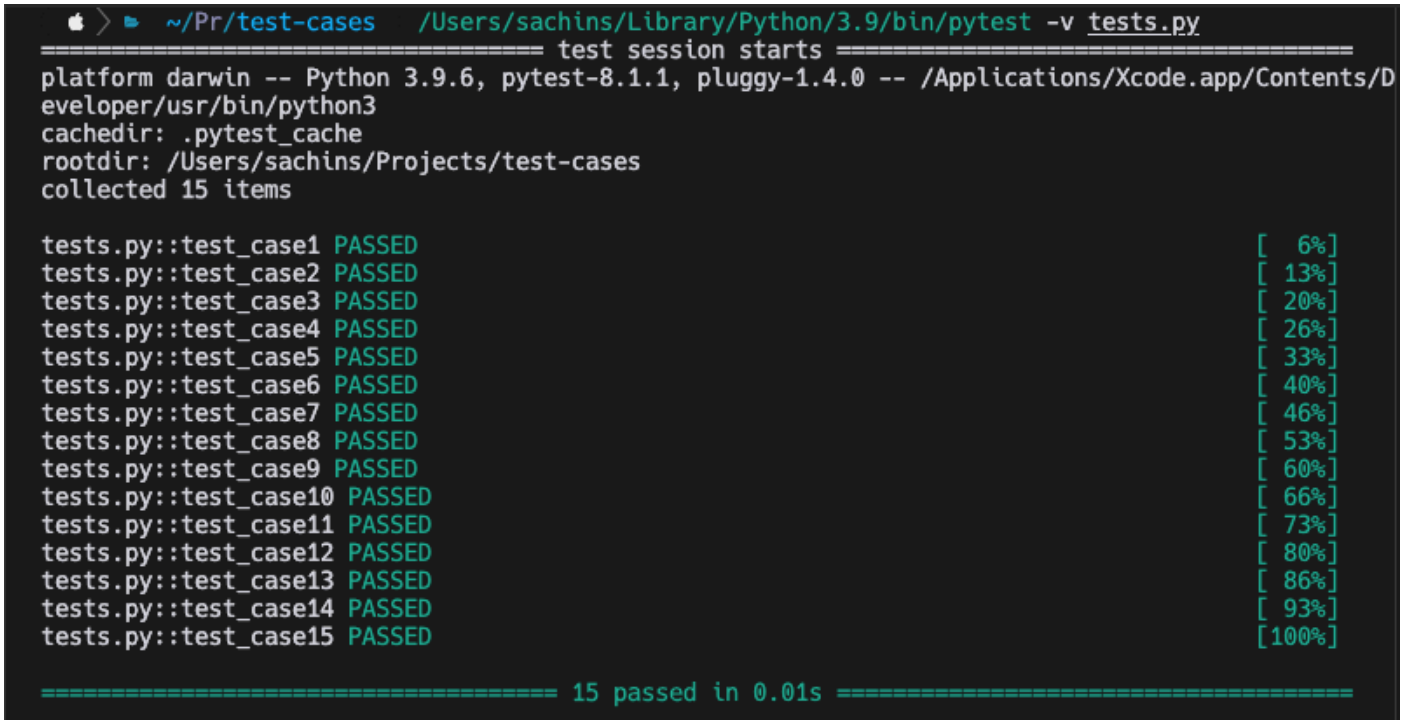
/api/search_discussions_by_category	Filter based on category and tag.	Category: "", Tag: ""	Status code: 400, Error message: "Invalid request. Please provide a valid category."	Status code: 400, Error message: "Invalid request. Please provide a valid category."	Success
/api/search_discussions_by_category	Filter based on category and tag.	Category: "nonexistent_category", Tag: "quiz-2"	Status code: 404, Error message: "No discussions found for the provided category."	Status code: 404, Error message: "No discussions found for the provided category."	Success
/api/search_discussions_by_category	Filter based on category and tag.	Category: "python", Tag: "nonexistent_tag"	Status code: 404, Error message: "No discussions found for the provided tag."	Status code: 404, Error message: "No discussions found for the provided tag."	Success
/api/search_discussions_by_category	Filter based on category and tag.	Category: "", Tag: "nonexistent_tag"	Status code: 400, Error message: "Invalid request. Please provide a valid category."	Status code: 400, Error message: "Invalid request. Please provide a valid category."	Success
/api/search_discussions_by_category	Filter based on category and tag.	Category: "nonexistent_category", Tag: "nonexistent_tag"	Status code: 404, Error message: "No discussions found for the provided category."	Status code: 404, Error message: "No discussions found for the provided category."	Success
/support_staff/activity_status	Get the list of all support staff members which are not responding.	N/A	Status code: 200, Response is a list of staff members with: username, isActive, lastActiveTime	Status code: 200, Response is a list of staff members with: username, isActive, lastActiveTime	Success
/support_staff/activity_status	Get the list of all support staff members which are not	Invalid API key	Status code: 401, Error message: "Unauthorized"	Status code: 401, Error message: "Unauthorized"	Success

	responding.				
/posts	Tag the concerned person.	Title: "Test Post", Description: "This is a test post", Tags: ["user1", "user2"]	Status code: 201	Status code: 201	Success
/posts	Tag the concerned person.	Missing title and description fields	Status code: 400	Status code: 400	Success
/search_users	Search for users to tag.	Search term: "user"	Status code: 200, Response contains data	Status code: 200, Response contains data	Success
/support_staff/activity_status	Get the list of all support staff members which are not responding.	Invalid API key	Status code: 401, Error message: "Unauthorized"	Status code: 401, Error message: "Unauthorized"	Success
/posts	Tag the concerned person.	Title: "Test Post", Description: "This is a test post", Tags: ["user1", "user2"]	Status code: 201	Status code: 201	Success
/posts	Tag the concerned person.	Missing title and description fields	Status code: 400	Status code: 400	Success

/profile	Provides details of a specific user by taking user id as parameter.	User ID: 1004	Output name should not be an integer	Output name is not an integer	Success
/profile	Provides details of a specific user by taking user id as parameter.	User ID: 1004	isspam: True, num_posts: > 14	isspam: True, num_posts: 15	Success
/profile	Provides details of a specific user by taking user id as parameter.	User ID: 1005	isspam: False, num_posts: 10	isspam: False, num_posts: 10	Success
/posts/{id}/locked.json	Lock a post from being edited	API key: your-api-key, API username: your-api-username, Post ID: 123	Status code: 200, Response contains "locked" field set to True	Status code: 200, Response contains "locked" field set to True	Success
/posts.json	Api for creating a query or replying on a query on discourse	Valid payload: Dictionary containing post data	Status code: 200, Response contains various fields related to the created post	Status code: 200, Response contains various fields related to the created post	Success
/search_users	Search for users to tag.	Search term: "user"	Status code: 200, Response contains data	Status code: 200, Response contains data	Success

/profile	Provides details of a specific user by taking user id as parameter.	User ID: 1004	Output name should not be an integer	Output name is not an integer	Success
/profile	Provides details of a specific user by taking user id as parameter.	User ID: 1004	isspam: True, num_posts: > 14	isspam: True, num_posts: 15	Success
/profile	Provides details of a specific user by taking user id as parameter.	User ID: 1005	isspam: False, num_posts: 10	isspam: False, num_posts: 10	Success
/posts/{id}/locked.json	Lock a post from being edited	API key: your-api-key, API username: your-api-username, Post ID: 123	Status code: 200, Response contains "locked" field set to True	Status code: 200, Response contains "locked" field set to True	Success
/posts.json	Api for creating a query or replying on a query on discourse	Valid payload: Dictionary containing post data	Status code: 200, Response contains various fields related to the created post	Status code: 200, Response contains various fields related to the created post	Success

Test Cases Result:


~/Pr/test-cases /Users/sachins/Library/Python/3.9/bin/pytest -v tests.py
test session starts
platform darwin -- Python 3.9.6, pytest-8.1.1, pluggy-1.4.0 -- /Applications/Xcode.app/Contents/Developer/usr/bin/python3
cachedir: .pytest_cache
rootdir: /Users/sachins/Projects/test-cases
collected 15 items
tests.py::test_case1 PASSED [6%]
tests.py::test_case2 PASSED [13%]
tests.py::test_case3 PASSED [20%]
tests.py::test_case4 PASSED [26%]
tests.py::test_case5 PASSED [33%]
tests.py::test_case6 PASSED [40%]
tests.py::test_case7 PASSED [46%]
tests.py::test_case8 PASSED [53%]
tests.py::test_case9 PASSED [60%]
tests.py::test_case10 PASSED [66%]
tests.py::test_case11 PASSED [73%]
tests.py::test_case12 PASSED [80%]
tests.py::test_case13 PASSED [86%]
tests.py::test_case14 PASSED [93%]
tests.py::test_case15 PASSED [100%]
15 passed in 0.01s

Sample Test Cases:

```python
import pytest
import requests

base_url = 'http://127.0.0.1:5000'


@pytest.mark.parametrize("payload, expected_status, expected_message", [
    ({
        'ticket_id': 3,
        'priority_measure': 6,
        'message': 'Priority measure exceeded 5. Immediate action required!',
        'webhook_url': 'https://chat.googleapis.com/v1/spaces/AAAAaBileJo/messages?key=AIzaS
    }, 200, 'Message posted successfully.'),
    ({
        'priority_measure': 6,
        'message': 'Priority measure exceeded 5. Immediate action required!',
        'webhook_url': 'https://example.com/gchat/webhook'
    }, 400, 'Invalid request body. Please provide valid ticket ID and priority measure.'),
    ({
        'ticket_id': 3,
        'priority_measure': 6,
        'message': 'Priority measure exceeded 5. Immediate action required!',
        'webhook_url': 'https://invalid-url'
    }, 500, 'Internal server error. Please try again later.')
])
def test_send_gchat_msg(payload, expected_status, expected_message):
    response = requests.post(f'{base_url}/api/send_gchat_msg', json=payload)
    assert response.status_code == expected_status
    assert response.json().get('message') == expected_message
```

```python
@pytest.fixture
def valid_payload():
    return {
        "title": "Test Post",
        "raw": "This is a test post.",
        "topic_id": 123,
        "category": 456,
        "created_at": "2024-04-04T12:00:00Z",
        "reply_to_post_number": None
    }
```

```python
def test_create_post_with_tags(client):
    # Test creating a new post with tags
    post_data = {
        "title": "Test Post",
        "description": "This is a test post",
        "tags": ["user1", "user2"]
    }
    response = client.post('/posts', json=post_data)
    assert response.status_code == 201

def test_create_post_missing_fields(client):
    # Test creating a post with missing fields
    post_data = {
        "description": "This is a test post",
        "tags": ["user1", "user2"]
    }
    response = client.post('/posts', json=post_data)
    assert response.status_code == 400

def test_search_users(client):
    # Test searching for users
    response = client.get('/search_users?search_term=user')
    assert response.status_code == 200
    data = json.loads(response.data)
    assert len(data) > 0

def test_search_users_no_results(client):
    # Test searching for users with no results
    response = client.get('/search_users?search_term=nonexistentuser')
    assert response.status_code == 404
```