BS in Data Science and Applications

Software Engineering

Milestone 5

(Test Cases)

Topic	**Gen AI Integration with Seek portal.**	
Team	30	
1	Anjali Panchal	21F2000411
2	Sahil Sandhu	21F1002317
3	Abdul Ahad Rauf	21F3002590
4	Sanket Metrewar	21F3000961
5	Aman Verma	21F1006376
6	Neon Thapa	21F1000706
7	Pratham Sharma	21F1006197
8	Leo Tom	21F1005835

Table of Contents

Super Seek API Test Cases 3

1. CONTENT MANAGEMENT 3
2. COURSE MANAGEMENT 9

Super Seek API Test Cases

This Markdown file documents tests for the Super Seek API, which manages educational content and user interactions.

1. CONTENT MANAGEMENT

1.1. /contents

1.1.1 Test: test_get_available_weeks_successful()

- **Description:** Tests successful retrieval of available weeks for content.
- **Example - True Scenario:**

```
response = requests.get(f"{BASE_URL}/contents")
assert response.status_code == 200
assert "weeks" in response.json()
```

- **Example – False Scenario:**

```
response = requests.get(f"{BASE_URL}/contents")
assert response.status_code != 404
```

1.2. /contents/{week}

1.2.1 Test: test_get_content_for_week_successful()

- **Description:** Tests successful retrieval of content titles for a specific week.
- **Example - True Scenario:**

```
week_number = 1
response = requests.get(f"{BASE_URL}/contents/{week_number}")
assert response.status_code == 200
assert response.json()["week"] == week_number
assert "lectures" in response.json()
assert "assignments" in response.json()
```

- **Example – False Scenario:**

```
week_number = 10
response = requests.get(f"{BASE_URL}/contents/{week_number}")
assert response.status_code == 404
```

1.2.2 Test: test_get_content_for_invalid_week()

- **Description:** Tests retrieval of content for a non-existing week.
- **Example - True Scenario:**

```
week_number = -1
response = requests.get(f"{BASE_URL}/contents/{week_number}")
assert response.status_code == 400
```

- **Example – False Scenario:**

```
week_number = 1
response = requests.get(f"{BASE_URL}/contents/{week_number}")
assert response.status_code != 400
```

1.3. /lec/{lec_id}/{week}

1.3.1. Test: test_get_lecture_details_successful()

- **Description:** Tests successful retrieval of lecture details.
- **Example - True Scenario:**

```
lec_id = 101
week_number = 1
response = requests.get(f"{BASE_URL}/lec/{lec_id}/{week_number}")
assert response.status_code == 200
assert response.json()["lec_id"] == lec_id
```

- **Example – False Scenario:**

```
lec_id = 999
week_number = 1
response = requests.get(f"{BASE_URL}/lec/{lec_id}/{week_number}")
assert response.status_code == 404
```

1.3.2. Test: test_get_lecture_details_invalid_id()

- **Description:** Tests retrieval of lecture details with an invalid lecture ID.
- **Example - True Scenario:**

```
lec_id = "invalid"
week_number = 1
response = requests.get(f"{BASE_URL}/lec/{lec_id}/{week_number}")
assert response.status_code == 400
```

- **Example – False Scenario:**

```
lec_id = 101
week_number = 1
response = requests.get(f"{BASE_URL}/lec/{lec_id}/{week_number}")
assert response.status_code != 400
```

1.4. /assignment/{assign_id}/{week}

1.4.1. Test: test_get_assignment_details_successful()

- **Description:** Tests successful retrieval of assignment details.
- **Example - True Scenario:**

```
assign_id = 201
week_number = 1
response = requests.get(f"{BASE_URL}/assignment/{assign_id}/{week_number}
assert response.status_code == 200
assert response.json()["assign_id"] == assign_id
```

- **Example – False Scenario:**

```
assign_id = 888
week_number = 1
response = requests.get(f"{BASE_URL}/assignment/{assign_id}/{week_number}")
assert response.status_code == 404
```

1.4.2. Test: test_get_assignment_details_invalid_id()

- **Description:** Tests retrieval of assignment details with an invalid assignment ID.
- **Example - True Scenario:**

```
assign_id = -1
week_number = 1
response = requests.get(f"{BASE_URL}/assignment/{assign_id}/{week_number}")
assert response.status_code == 400
```

- **Example – False Scenario:**

```
assign_id = 201
week_number = 1
response = requests.get(f"{BASE_URL}/assignment/{assign_id}/{week_number}")
assert response.status_code != 400
```

1.5. /solve_assignment/{assign_id}

1.5.1. Test: test_submit_assignment_successful()

- **Description:** Tests successful submission of an assignment.
- **Example - True Scenario:**

```
assign_id = 201
submission_data = {
    "user_id": 1,
    "question_id": 301,
    "selected_answer": "4",
    "code_submission": "def add(a, b):\n  return a + b"
}

response = requests.post(f"{BASE_URL}/solve_assignment/{assign_id}",
                            json=submission_data)
assert response.status_code == 200
assert "message" in response.json()
assert "is_correct" in response.json()
```

- **Example - False Scenario:**

```
assign_id = 201
submission_data = {
    "user_id": 1
}
response = requests.post(f"{BASE_URL}/solve_assignment/{assign_id}",
                         json=submission_data)
assert response.status_code == 400
```

1.5.2. Test: test_submit_assignment_missing_fields()

- **Description:** Tests submission of an assignment with missing required fields.
- **Example - True Scenario:**

```
assign_id = 201
submission_data = {}
response = requests.post(f"{BASE_URL}/solve_assignment/{assign_id}",
                         json=submission_data)
assert response.status_code == 400
```

- **Example – False Scenario:**

```
assign_id = 201
submission_data = {
    "user_id": 1,
    "question_id": 301,
    "selected_answer": "4",
    "code_submission": "def add(a, b):\n  return a + b"
}
response = requests.post(f"{BASE_URL}/solve_assignment/{assign_id}",
                          json=submission_data)
assert response.status_code != 400
```

1.6. /get_summary/{lec_id}\

1.6.1. Test: test_get_lecture_summary_successful()

- **Description:** Tests successfully getting a summary for a lecture.
- **Example - True Scenario:**

```
lec_id = 101
response = requests.get(f"{BASE_URL}/get_summary/{lec_id}")
assert response.status_code == 200
assert "summary" in response.json()
```

- **Example – False Scenario:**

```
lec_id = 999
response = requests.get(f"{BASE_URL}/get_summary/{lec_id}")
assert response.status_code == 404
```

1.6.2. Test: test_get_lecture_summary_invalid_id()

- **Description:** Tests getting a summary with an invalid lecture ID.
- **Example - True Scenario:**

```
lec_id = "abc"
response = requests.get(f"{BASE_URL}/get_summary/{lec_id}")
assert response.status_code == 400
```

- **Example – False Scenario:**

```
lec_id = 101
response = requests.get(f"{BASE_URL}/get_summary/{lec_id}")
assert response.status_code != 400
```

1.7. /feedback/{assign_id}

1.7.1. Test: test_get_feedback_successful()

- **Description:** Tests successfully getting feedback for an assignment.
- **Example - True Scenario:**

```
assign_id = 201
response = requests.get(f"{BASE_URL}/feedback/{assign_id}")
assert response.status_code == 200
assert "feedback" in response.json()
```

- **Example - False Scenario:**

```
assign_id = 888
response = requests.get(f"{BASE_URL}/feedback/{assign_id}")
assert response.status_code == 404
```

1.7.2. Test: test_get_feedback_invalid_id()

- **Description:** Tests getting feedback with an invalid assignment ID.
- **Example - True Scenario:**

```
assign_id = "wrong_id"
response = requests.get(f"{BASE_URL}/feedback/{assign_id}")
assert response.status_code == 400
```

- **Example – False Scenario:**

```
assign_id = 201
response = requests.get(f"{BASE_URL}/feedback/{assign_id}")
assert response.status_code != 400
```

1.8. /get_links/{lec_id}

1.8.1. Test: test_get_links_successful()

- **Description:** Tests successfully getting links for a lecture.
- **Example - True Scenario:**

```
lec_id = 101
response = requests.get(f"{BASE_URL}/get_links/{lec_id}")
assert response.status_code == 200
assert "links" in response.json()
```

- **Example – False Scenario:**

```
lec_id = 999
response = requests.get(f"{BASE_URL}/get_links/{lec_id}")
assert response.status_code == 404
```

1.8.2. Test: test_get_links_invalid_id()

- **Description:** Tests getting links with an invalid lecture ID.
- **Example - True Scenario:**

```
lec_id = "not_a_number"
response = requests.get(f"{BASE_URL}/get_links/{lec_id}")
assert response.status_code == 400
```

- **Example – False Scenario:**

```
lec_id = 101
response = requests.get(f"{BASE_URL}/get_links/{lec_id}")
assert response.status_code != 400
```

1.9. /make_data/{que_id}

1.9.1. Test: test_generate_data_successful()

- **Description:** Tests successful generation of data for a question.
- **Example - True Scenario:**

```
que_id = 301
response = requests.get(f"{BASE_URL}/make_data/{que_id}")
assert response.status_code == 200
assert "data" in response.json()
```

- **Example – False Scenario:**

```
que_id = 777
response = requests.get(f"{BASE_URL}/make_data/{que_id}")
assert response.status_code == 404
```

1.9.2. Test: test_generate_data_invalid_id()

- **Description:** Tests generating data with an invalid question ID.
- **Example - True Scenario:**

```
que_id = 0
response = requests.get(f"{BASE_URL}/make_data/{que_id}")
assert response.status_code == 400
```

- **Example – False Scenario:**

```
que_id = 301
response = requests.get(f"{BASE_URL}/make_data/{que_id}")
assert response.status_code != 400
```

1.10. /logout

1.10.1. Test: test_logout_successful()

- **Description:** Tests successful user logout.
- **Example - True Scenario:**

```
response = requests.post(f"{BASE_URL}/logout")
assert response.status_code == 200
assert "message" in response.json()
```

2. COURSE MANAGEMENT

2.1. /courses

2.1.1. Test: test_get_available_courses_successful()

- **Description:** Tests successful retrieval of available courses.
- **Example - True Scenario:**

```
response = requests.get(f"{BASE_URL}/courses")
assert response.status_code == 200
assert isinstance(response.json(), list)
```

2.2. /enroll/{course_id}

2.2.1. Test: test_enroll_in_course_successful()

- **Description:** Tests successful enrollment in a course.
- **Example - True Scenario:**

```
course_id = 1
response = requests.post(f"{BASE_URL}/enroll/{course_id}")
assert response.status_code == 200
assert "message" in response.json()
```

2.2.2. Test: test_enroll_in_course_invalid_id()

- **Description:** Tests enrolling in a course with an invalid course ID.
- **Example - True Scenario:**

```
course_id = 999
response = requests.post(f"{BASE_URL}/enroll/{course_id}")
assert response.status_code == 404
```

2.3. /progress/{course_id}

2.3.1. Test: test_get_user_progress_successful()

- **Description:** Tests successful retrieval of user progress for a course.
- **Example - True Scenario:**

```
course_id = 1
response = requests.get(f"{BASE_URL}/progress/{course_id}")
assert response.status_code == 200
assert "progress" in response.json()
```

2.3.2. Test: test_get_user_progress_invalid_id()

- **Description:** Tests retrieving user progress with an invalid course ID.
- **Example - True Scenario:**

```
course_id = -5
response = requests.get(f"{BASE_URL}/progress/{course_id}")
assert response.status_code == 400
```