Software Engineering Project

Milestone 5

Testing Report

Team 7 Details

- Anirudh Murthy - 21f1000931@ds.study.iitm.ac.in
- Ashwin Hebbar - 21f1003155@ds.study.iitm.ac.in
- Prabuddh Mathur - 21f2001016@ds.study.iitm.ac.in
- Rituparna Das - 21f1003954@ds.study.iitm.ac.in
- Sayan Ghosh - 21f1006567@ds.study.iitm.ac.in
- Vaishnavi Dwivedi - 21f1000147@ds.study.iitm.ac.in
- Vignesh Babu - 22f1001225@ds.study.iitm.ac.in

Testing Script

```python
import requests
import pytest

BASE_URL = "http://localhost:5000"
JWT_TOKEN = # redacted

headers = {
    "Authorization": f"Bearer {JWT_TOKEN}",
    "Content-Type": "application/json"
}

REPORT_FILE = "test_report.txt"

def log_result(api, inputs, expected_status_code, response, expected_output=None,
 ↪  method=None):
    actual_output = response.json() if response.content else "No Content"
    success = (response.status_code == expected_status_code) and (actual_output ==
 ↪  expected_output if expected_output is not None else True)
    result = "Success" if success else "Fail"

    report_entry = f"""
    API: [{method}] {api}
    Inputs: {inputs}
    Expected Output: Status Code {expected_status_code}, {expected_output}
    Actual Output: Status Code {response.status_code}, {actual_output}
    Result: {result}
    """

    print(report_entry)  # Print to console (optional)

    with open(REPORT_FILE, "a") as file:
        file.write(report_entry)

def clear_report_file():
    with open(REPORT_FILE, "w") as file:
        file.write("Test Report\n")
        file.write("="*50 + "\n")

def make_request(method, api, payload=None, expected_status_code=200,
 ↪  expected_output=None):
    url = f"{BASE_URL}{api}"
    response = method(url, json=payload, headers=headers)
    log_result(api, payload or {}, expected_status_code, response, expected_output,
 ↪  method.__name__.upper())
    assert response.status_code == expected_status_code
    return response

clear_report_file()

# Test cases
def test_post_debug_db_populate():
    make_request(requests.post, "/debug/db_populate", expected_output={"message":
 ↪  "Database populated"})

def test_get_assignment():
    make_request(requests.get, "/assignment/1", expected_output={'id': 1, 'title': 'GA1',
 ↪  'is_graded': True, 'due_date': '2024-08-01T23:59:59', 'week_id': 1})

def test_put_assignment():
    payload = {'id': 1, 'title': 'GA1', 'is_graded': True, 'due_date': '2024-08-01
 ↪  23:59:59', 'week_id': 1}
    make_request(requests.put, "/assignment/1", payload, expected_output=payload)

def test_get_week_assignments():
    make_request(requests.get, "/week/1/assignments", expected_output=[{'id': 1, 'title':
 ↪  'GA1', 'is_graded': True, 'due_date': '2024-08-01T23:59:59', 'week_id': 1}])

def test_get_assignment_questions():
    expected_output = [
        {'id': 1, 'text': 'What is the data type of the following variable in
 ↪  Python?\nmy_var = "Hello, World!"', 'is_msq': False, 'assignment_id': 1},
        {'id': 2, 'text': 'Which of the following is used to define a multi-line comment
 ↪  in Python?', 'is_msq': False, 'assignment_id': 1}
    ]
    make_request(requests.get, "/assignment/1/questions",
 ↪  expected_output=expected_output)

def test_post_assignment_questions():
    payload = {'errors': {'text': 'Text of the question Missing required parameter in the
 ↪  JSON body or the post body or the query string'}, 'message': 'Input payload
 ↪  validation failed'}
    make_request(requests.post, "/assignment/1/questions",
 ↪  payload,expected_status_code=400, expected_output={"id": 1, **payload})

def test_get_chats():
    make_request(requests.get, "/chats", expected_output=[])


def test_delete_chats():
    make_request(requests.delete, "/chats", expected_output={"message": "Chats deleted"})

def test_get_course():
    make_request(requests.get, "/course/1", expected_output={'name': 'Programming in
 ↪  Python', 'level': 1, 'summary': None})


def test_put_course():
    payload = {
        "name": "Programming in Python",
        "level": 1,
        "summary": "This course is about programming in Python"
    }
    make_request(requests.put, "/course/1", payload, expected_output=payload)

def test_get_course_chats():
    expected_output = [
        {'id': 1, 'prompt': 'How do you swap the values of two variables in Python
    ↪   without using a temporary variable?', 'response': 'Use tuple unpacking: a, b = b, a',
    ↪   'datetime': '2024-07-28T20:50:59', 'user_id': 2, 'course_id': 1},
        {'id': 2, 'prompt': 'What is the difference between a list and a tuple in
    ↪   Python?', 'response': 'Lists are mutable (can be changed), while tuples are immutable
    ↪   (cannot be changed).', 'datetime': '2024-07-28T20:52:59', 'user_id': 2, 'course_id':
    ↪   1},
        {'id': 3, 'prompt': 'How do you check if a string is a palindrome in Python?',
    ↪   'response': "Reverse the string and compare it to the original string. If they are
    ↪   equal, it's a palindrome.", 'datetime': '2024-07-28T20:51:59', 'user_id': 3,
    ↪   'course_id': 1},
        {'id': 4, 'prompt': 'Explain the use of the enumerate() function in Python.',
    ↪   'response': 'enumerate() adds a counter to an iterable and returns it as an enumerate
    ↪   object.', 'datetime': '2024-07-28T20:53:59', 'user_id': 3, 'course_id': 1}
    ]
    make_request(requests.get, "/course/1/chats", expected_output=expected_output)


def test_delete_course_chats():
    make_request(requests.delete, "/course/1/chats", expected_output={"message": "Chats
 ↪   deleted"})

def test_get_course_events():
    make_request(requests.get, "/course/1/events", expected_output=[])

def test_post_course_events():
    payload = {
        "title": "OPPE due",
        "start": "2024-08-01 23:59:59",
        "end": "2024-08-04 23:59:59"
    }
    make_request(requests.post, "/course/1/events", payload, expected_output={'id': 5,
    ↪   'title': 'OPPE due', 'start': '2024-08-01T23:59:59', 'end': '2024-08-04T23:59:59',
    ↪   'course_id': 1, 'user_id': 1})

def test_get_course_students():
    make_request(requests.get, "/course/1/students", expected_output=[{'id': 2, 'cgpa':
 ↪   8.5}, {'id': 3, 'cgpa': 8.5}, {'id': 4, 'cgpa': 8.5}, {'id': 5, 'cgpa': 8.5}])

def test_get_course_weeks():
    make_request(requests.get, "/course/1/weeks", expected_output=[{'id': 1, 'number': 1,
    ↪   'course_id': 1, 'summary': 'Basics of python'}, {'id': 2, 'number': 2, 'course_id':
    ↪   1, 'summary': 'Using Replit'}, {'id': 3, 'number': 3, 'course_id': 1, 'summary':
    ↪   'Datatypes in python'}])

def test_post_course_weeks():
    payload = {'errors': {'course_id': 'Course ID Missing required parameter in the JSON
    ↪   body or the post body or the query string'}, 'message': 'Input payload validation
    ↪   failed'}
    make_request(requests.post, "/course/1/weeks", payload,expected_status_code=400,
 ↪   expected_output={"id": 1, **payload})

def test_get_all_courses():
    make_request(requests.get, "/courses", expected_output=[{'name': 'Programming in
↪   Python', 'level': 1, 'summary': 'This course is about programming in Python'},
↪   {'name': 'System Commands', 'level': 2, 'summary': None}, {'name': 'Programming, Data
↪   Structures and Algorithms', 'level': 2, 'summary': None}, {'name': 'Modern
↪   Application Development I', 'level': 2, 'summary': None}])

def test_post_course():
    payload = {'name': 'Programming in Python', 'level': 1, 'summary': 'This course is
↪   about programming in Python'}
    make_request(requests.post, "/courses", payload, expected_output={"id": 1,
↪   **payload})

def test_get_event():
    make_request(requests.get, "/event/1",expected_status_code=401,
↪   expected_output={'message': 'Unauthorized'})


def test_put_event():
    payload = {
        "title": "OPPE due",
        "start": "2024-08-01 23:59:59",
        "end": "2024-08-04 23:59:59"
    }
    make_request(requests.put, "/event/1", payload,expected_status_code=401,
↪   expected_output={'message': 'Unauthorized'})

def test_get_instructor():
    make_request(requests.get, "/instructor/10", expected_output={"id": 10})

def test_get_instructor_teach():
    make_request(requests.get, "/instructor/10/teach/1", expected_output={'course_id': 1,
↪   'user_id': 10})

def test_post_instructor_teach():
    payload = {
        "course_id": 1,
        "instructor_id": 1
    }
    make_request(requests.post, "/instructor/10/teach/1", payload,
↪   expected_status_code=400, expected_output={'message': 'Instructor already teaching
↪   course'})

def test_delete_instructor_teach():
    make_request(requests.delete, "/instructor/10/teach/1", expected_output={'message':
↪   'Teaching removed'})

def test_get_instructors():
    make_request(requests.get, "/instructors", expected_output=[{'id': 9}, {'id': 10},
↪   {'id': 11}, {'id': 12}])

def test_post_instructors():
    payload = {
        "name": "Instructor Name"
    }
    make_request(requests.post, "/instructors", payload,expected_status_code=401,
 ↪  expected_output={'message': 'User is an admin'})

def test_get_lecture():
    make_request(requests.get, "/lecture/1", expected_output={'id': 1, 'week_id': 1,
 ↪  'title': 'Introduction to python', 'url':
 ↪  'https://www.youtube.com/watch?v=8ndsDXohLMQ&list=PLDsnL5pk7-N_9oy2RN4A65Z-PEnvtc7rf',
 ↪  'summary': "Summary of the lecture 'Intro to python'"})

def test_delete_event():
    make_request(requests.delete, "/event/1", expected_status_code=401,
 ↪  expected_output={"message": "Unauthorized"})

def test_delete_instructor():
    make_request(requests.delete, "/instructor/1", expected_status_code=401,
 ↪  expected_output={'message': 'Instructor not found'})


def test_delete_lecture():
    make_request(requests.delete, "/lecture/1", expected_output={"message": "Lecture
 ↪  deleted"})
```

Testing Report

Endpoint /debug/db_populate - POST Method:

```
Inputs:: {}
Expected Output: Status Code 200, {'message': 'Database populated'}
Actual Output: Status Code 200, {'message': 'Database populated'}
Result: Success
```

Endpoint /assignment/1 - GET Method:

```
  Inputs:: {}
  Expected Output: Status Code 200, {'id': 1, 'title': 'GA1', 'is_graded': True,
↪ 'due_date': '2024-08-01T23:59:59', 'week_id': 1}
  Actual Output: Status Code 200, {'id': 1, 'title': 'GA1', 'is_graded': True,
↪ 'due_date': '2024-08-01T23:59:59', 'week_id': 1}
  Result: Success
```

Endpoint /assignment/1 - PUT Method:

```
  Inputs:: {'id': 1, 'title': 'GA1', 'is_graded': True, 'due_date': '2024-08-01
↪ 23:59:59', 'week_id': 1}
  Expected Output: Status Code 200, {'id': 1, 'title': 'GA1', 'is_graded': True,
↪ 'due_date': '2024-08-01 23:59:59', 'week_id': 1}
  Actual Output: Status Code 200, {'id': 1, 'title': 'GA1', 'is_graded': True,
↪ 'due_date': '2024-08-01T23:59:59', 'week_id': 1}
  Result: Fail
```

Endpoint /week/1/assignments - GET Method:

```
  Inputs:: {}
  Expected Output: Status Code 200, [{'id': 1, 'title': 'GA1', 'is_graded': True,
↪ 'due_date': '2024-08-01T23:59:59', 'week_id': 1}]
  Actual Output: Status Code 200, [{'id': 1, 'title': 'GA1', 'is_graded': True,
↪ 'due_date': '2024-08-01T23:59:59', 'week_id': 1}]
  Result: Success
```

Endpoint /assignment/1/questions - GET Method:

```
  Inputs:: {}
  Expected Output: Status Code 200, [{'id': 1, 'text': 'What is the data type of the
↪ following variable in Python?\nmy_var = "Hello, World!"', 'is_msq': False,
↪ 'assignment_id': 1}, {'id': 2, 'text': 'Which of the following is used to define a
↪ multi-line comment in Python?', 'is_msq': False, 'assignment_id': 1}]
  Actual Output: Status Code 200, [{'id': 1, 'text': 'What is the data type of the
↪ following variable in Python?\nmy_var = "Hello, World!"', 'is_msq': False,
↪ 'assignment_id': 1}, {'id': 2, 'text': 'Which of the following is used to define a
↪ multi-line comment in Python?', 'is_msq': False, 'assignment_id': 1}]
  Result: Success
```

Endpoint /assignment/1/questions - POST Method:

```
   Inputs:: {'errors': {'text': 'Text of the question Missing required parameter in the
↪  JSON body or the post body or the query string'}, 'message': 'Input payload
↪  validation failed'}
   Expected Output: Status Code 400, {'id': 1, 'errors': {'text': 'Text of the question
↪  Missing required parameter in the JSON body or the post body or the query string'},
↪  'message': 'Input payload validation failed'}
   Actual Output: Status Code 400, {'errors': {'text': 'Text of the question Missing
↪  required parameter in the JSON body or the post body or the query string'},
↪  'message': 'Input payload validation failed'}
   Result: Fail
```

Endpoint /chats - GET Method:

```
   Inputs:: {}
   Expected Output: Status Code 200, []
   Actual Output: Status Code 200, []
   Result: Success
```

Endpoint /chats - DELETE Method:

```
   Inputs:: {}
   Expected Output: Status Code 200, {'message': 'Chats deleted'}
   Actual Output: Status Code 200, {'message': 'Chats deleted'}
   Result: Success
```

Endpoint /course/1 - GET Method:

```
   Inputs:: {}
   Expected Output: Status Code 200, {'name': 'Programming in Python', 'level': 1,
↪  'summary': None}
   Actual Output: Status Code 200, {'name': 'Programming in Python', 'level': 1,
↪  'summary': None}
   Result: Success
```

Endpoint /course/1 - PUT Method:

```
   Inputs:: {'name': 'Programming in Python', 'level': 1, 'summary': 'This course is
↪  about programming in Python'}
   Expected Output: Status Code 200, {'name': 'Programming in Python', 'level': 1,
↪  'summary': 'This course is about programming in Python'}
   Actual Output: Status Code 200, {'name': 'Programming in Python', 'level': 1,
↪  'summary': 'This course is about programming in Python'}
   Result: Success
```

Endpoint /course/1/chats - GET Method:

```
Inputs:: {}
Expected Output: Status Code 200, [{'id': 1, 'prompt': 'How do you swap the values of
↪ two variables in Python without using a temporary variable?', 'response': 'Use tuple
↪ unpacking: a, b = b, a', 'datetime': '2024-07-28T20:50:59', 'user_id': 2,
↪ 'course_id': 1}, {'id': 2, 'prompt': 'What is the difference between a list and a
↪ tuple in Python?', 'response': 'Lists are mutable (can be changed), while tuples are
↪ immutable (cannot be changed).', 'datetime': '2024-07-28T20:52:59', 'user_id': 2,
↪ 'course_id': 1}, {'id': 3, 'prompt': 'How do you check if a string is a palindrome in
↪ Python?', 'response': "Reverse the string and compare it to the original string. If
↪ they are equal, it's a palindrome.", 'datetime': '2024-07-28T20:51:59', 'user_id': 3,
↪ 'course_id': 1}, {'id': 4, 'prompt': 'Explain the use of the enumerate() function in
↪ Python.', 'response': 'enumerate() adds a counter to an iterable and returns it as an
↪ enumerate object.', 'datetime': '2024-07-28T20:53:59', 'user_id': 3, 'course_id': 1}]
Actual Output: Status Code 200, [{'id': 1, 'prompt': 'How do you swap the values of
↪ two variables in Python without using a temporary variable?', 'response': 'Use tuple
↪ unpacking: a, b = b, a', 'datetime': '2024-07-28T20:50:59', 'user_id': 2,
↪ 'course_id': 1}, {'id': 2, 'prompt': 'What is the difference between a list and a
↪ tuple in Python?', 'response': 'Lists are mutable (can be changed), while tuples are
↪ immutable (cannot be changed).', 'datetime': '2024-07-28T20:52:59', 'user_id': 2,
↪ 'course_id': 1}, {'id': 3, 'prompt': 'How do you check if a string is a palindrome in
↪ Python?', 'response': "Reverse the string and compare it to the original string. If
↪ they are equal, it's a palindrome.", 'datetime': '2024-07-28T20:51:59', 'user_id': 3,
↪ 'course_id': 1}, {'id': 4, 'prompt': 'Explain the use of the enumerate() function in
↪ Python.', 'response': 'enumerate() adds a counter to an iterable and returns it as an
↪ enumerate object.', 'datetime': '2024-07-28T20:53:59', 'user_id': 3, 'course_id': 1}]
Result: Success
```

Endpoint /course/1/chats - DELETE Method:

```
Inputs:: {}
Expected Output: Status Code 200, {'message': 'Chats deleted'}
Actual Output: Status Code 200, {'message': 'Chats deleted'}
Result: Success
```

Endpoint /course/1/events - GET Method:

```
Inputs:: {}
Expected Output: Status Code 200, []
Actual Output: Status Code 200, []
Result: Success
```

Endpoint /course/1/events - POST Method:

```
Inputs:: {'title': 'OPPE due', 'start': '2024-08-01 23:59:59', 'end': '2024-08-04
↪ 23:59:59'}
Expected Output: Status Code 200, {'id': 5, 'title': 'OPPE due', 'start':
↪ '2024-08-01T23:59:59', 'end': '2024-08-04T23:59:59', 'course_id': 1, 'user_id': 1}
Actual Output: Status Code 200, {'id': 5, 'title': 'OPPE due', 'start':
↪ '2024-08-01T23:59:59', 'end': '2024-08-04T23:59:59', 'course_id': 1, 'user_id': 1}
Result: Success
```

Endpoint /course/1/students - GET Method:

```
Inputs:: {}
Expected Output: Status Code 200, [{'id': 2, 'cgpa': 8.5}, {'id': 3, 'cgpa': 8.5},
↪ {'id': 4, 'cgpa': 8.5}, {'id': 5, 'cgpa': 8.5}]
Actual Output: Status Code 200, [{'id': 2, 'cgpa': 8.5}, {'id': 3, 'cgpa': 8.5},
↪ {'id': 4, 'cgpa': 8.5}, {'id': 5, 'cgpa': 8.5}]
Result: Success
```

Endpoint /course/1/weeks - GET Method:

```
Inputs:: {}
Expected Output: Status Code 200, [{'id': 1, 'number': 1, 'course_id': 1, 'summary':
↪ 'Basics of python'}, {'id': 2, 'number': 2, 'course_id': 1, 'summary': 'Using
↪ Replit'}, {'id': 3, 'number': 3, 'course_id': 1, 'summary': 'Datatypes in python'}]
Actual Output: Status Code 200, [{'id': 1, 'number': 1, 'course_id': 1, 'summary':
↪ 'Basics of python'}, {'id': 2, 'number': 2, 'course_id': 1, 'summary': 'Using
↪ Replit'}, {'id': 3, 'number': 3, 'course_id': 1, 'summary': 'Datatypes in python'}]
Result: Success
```

Endpoint /course/1/weeks - POST Method:

```
Inputs:: {'errors': {'course_id': 'Course ID Missing required parameter in the JSON
↪ body or the post body or the query string'}, 'message': 'Input payload validation
↪ failed'}
Expected Output: Status Code 400, {'id': 1, 'errors': {'course_id': 'Course ID
↪ Missing required parameter in the JSON body or the post body or the query string'},
↪ 'message': 'Input payload validation failed'}
Actual Output: Status Code 400, {'errors': {'number': 'Number of the week Missing
↪ required parameter in the JSON body or the post body or the query string'},
↪ 'message': 'Input payload validation failed'}
Result: Fail
```

Endpoint /courses - GET Method:

```
Inputs:: {}
Expected Output: Status Code 200, [{'name': 'Programming in Python', 'level': 1,
↪ 'summary': 'This course is about programming in Python'}, {'name': 'System Commands',
↪ 'level': 2, 'summary': None}, {'name': 'Programming, Data Structures and Algorithms',
↪ 'level': 2, 'summary': None}, {'name': 'Modern Application Development I', 'level':
↪ 2, 'summary': None}]
Actual Output: Status Code 200, [{'name': 'Programming in Python', 'level': 1,
↪ 'summary': 'This course is about programming in Python'}, {'name': 'System Commands',
↪ 'level': 2, 'summary': None}, {'name': 'Programming, Data Structures and Algorithms',
↪ 'level': 2, 'summary': None}, {'name': 'Modern Application Development I', 'level':
↪ 2, 'summary': None}]
Result: Success
```

Endpoint /courses - POST Method:

```
Inputs:: {'name': 'Programming in Python', 'level': 1, 'summary': 'This course is
↪ about programming in Python'}
Expected Output: Status Code 200, {'id': 1, 'name': 'Programming in Python', 'level':
↪ 1, 'summary': 'This course is about programming in Python'}
```

```
Actual Output: Status Code 200, {'name': 'Programming in Python', 'level': 1,
↪ 'summary': 'This course is about programming in Python'}
Result: Fail
```

Endpoint /event/1 - GET Method:

```
Inputs:: {}
Expected Output: Status Code 401, {'message': 'Unauthorized'}
Actual Output: Status Code 401, {'message': 'Unauthorized'}
Result: Success
```

Endpoint /event/1 - PUT Method:

```
Inputs:: {'title': 'OPPE due', 'start': '2024-08-01 23:59:59', 'end': '2024-08-04
↪ 23:59:59'}
Expected Output: Status Code 401, {'message': 'Unauthorized'}
Actual Output: Status Code 401, {'message': 'Unauthorized'}
Result: Success
```

Endpoint /instructor/10 - GET Method:

```
Inputs:: {}
Expected Output: Status Code 200, {'id': 10}
Actual Output: Status Code 200, {'id': 10}
Result: Success
```

Endpoint /instructor/10/teach/1 - GET Method:

```
Inputs:: {}
Expected Output: Status Code 200, {'course_id': 1, 'user_id': 10}
Actual Output: Status Code 200, {'course_id': 1, 'user_id': 10}
Result: Success
```

Endpoint /instructor/10/teach/1 - POST Method:

```
Inputs:: {'course_id': 1, 'instructor_id': 1}
Expected Output: Status Code 400, {'message': 'Instructor already teaching course'}
Actual Output: Status Code 400, {'message': 'Instructor already teaching course'}
Result: Success
```

Endpoint /instructor/10/teach/1 - DELETE Method:

```
Inputs:: {}
Expected Output: Status Code 200, {'message': 'Teaching removed'}
Actual Output: Status Code 200, {'message': 'Teaching removed'}
Result: Success
```

Endpoint /instructors - GET Method:

```
Inputs:: {}
Expected Output: Status Code 200, [{'id': 9}, {'id': 10}, {'id': 11}, {'id': 12}]
Actual Output: Status Code 200, [{'id': 9}, {'id': 10}, {'id': 11}, {'id': 12}]
Result: Success
```

Endpoint /instructors - POST Method:

```
Inputs:: {'name': 'Instructor Name'}
Expected Output: Status Code 401, {'message': 'User is an admin'}
Actual Output: Status Code 401, {'message': 'User is an admin'}
Result: Success
```

Endpoint /lecture/1 - GET Method:

```
Inputs:: {}
Expected Output: Status Code 200, {'id': 1, 'week_id': 1, 'title': 'Introduction to
↪ python', 'url': 'https://www.youtube.com/watch?v=8ndsDXohLMQ', 'summary': "Summary of
↪ the lecture 'Intro to python'"}
Actual Output: Status Code 200, {'id': 1, 'week_id': 1, 'title': 'Introduction to
↪ python', 'url': 'https://www.youtube.com/watch?v=8ndsDXohLMQ', 'summary': "Summary of
↪ the lecture 'Intro to python'"}
Result: Success
```

Endpoint /event/1 - DELETE Method:

```
Inputs:: {}
Expected Output: Status Code 401, {'message': 'Unauthorized'}
Actual Output: Status Code 401, {'message': 'Unauthorized'}
Result: Success
```

Endpoint /instructor/1 - DELETE Method:

```
Inputs:: {}
Expected Output: Status Code 401, {'message': 'Instructor not found'}
Actual Output: Status Code 401, {'message': 'Instructor not found'}
Result: Success
```

Endpoint /lecture/1 - DELETE Method:

```
Inputs:: {}
Expected Output: Status Code 200, {'message': 'Lecture deleted'}
Actual Output: Status Code 200, {'message': 'Lecture deleted'}
Result: Success
```