Endpoint Being Tested: http://127.0.0.1:5000/api/feedback-recommendations

---

Case: *Successful feedback generation*

Request Method: POST

Inputs:


```
{
  "submitted_answers": [
    {
      "qid": 1,
      "answer": [
        "D"
      ]
    },
    {
      "qid": 2,
      "answer": [
        "B"
      ]
    },
    {
      "qid": 3,
      "answer": [
        "B",
        "D"
      ]
    },
    {
      "qid": 4,
      "answer": [
        "B",
        "C"
      ]
    },
    {
      "qid": 5,
      "answer": [
        "B"
      ]
    },
    {
      "qid": 6,
      "answer": [
        "1020"
      ]
    }
```

```
  ]
}
```

Expected Output:

```
HTTP Status Code: 200 and JSON with 'comprehensive_feedback'
```

Actual Output:

```
HTTP Status Code: 500
JSON: {
    "error": "Internal server error: 403 Request had insufficient authentication
scopes. [reason: \"ACCESS_TOKEN_SCOPE_INSUFFICIENT\"\ndomain:
\"googleapis.com\"\nmetadata {\n  key: \"service\"\n  value:
\"generativelanguage.googleapis.com\"\n}\nmetadata {\n  key: \"method\"\n  value:
\"google.ai.generativelanguage.v1beta.GenerativeService.GenerateContent\"\n}\n]"
}
```

Result: Failed

Pytest Code:

```
def test_feedback_success(client):
    payload = {
        "submitted_answers": [
            {"qid": 1, "answer": ["D"]},
            {"qid": 2, "answer": ["B"]},
            {"qid": 3, "answer": ["B", "D"]},
            {"qid": 4, "answer": ["B", "C"]},
            {"qid": 5, "answer": ["B"]},
            {"qid": 6, "answer": ["1020"]}
        ]
    }

    response = client.post("/api/feedback-recommendations", json=payload)
    try:
        data = response.get_json(force=True, silent=True)
    except Exception:
        data = None
    json_output = response.data.decode('utf-8')

    expected_status = 200
    result = "Success" if response.status_code == expected_status and data and
"comprehensive_feedback" in data else "Failed"

    write_test_doc(
```

```
        title="***Case:*** *Successful feedback generation*",
        endpoint="http://127.0.0.1:5000/api/feedback-recommendations",
        method="POST",
        inputs=json.dumps(payload, indent=2),
        expected="HTTP Status Code: 200 and JSON with 'comprehensive_feedback'",
        actual=f"HTTP Status Code: {response.status_code}\nJSON: {json_output}",
        result=result
    )

    assert response.status_code == 200
    assert "comprehensive_feedback" in data
```

Case: *Missing submitted_answers field*

Request Method: POST

Inputs:

```
{}
```

Expected Output:

```
HTTP Status Code: 500 and error message
```

Actual Output:

```
HTTP Status Code: 500
JSON: {
    "error": "Internal server error: division by zero"
}
```

Result: Success

Pytest Code:

```
def test_feedback_missing_field(client):
    payload = {
        # submitted_answers key is intentionally missing
    }

    response = client.post("/api/feedback-recommendations", json=payload)
    try:
        data = response.get_json(force=True, silent=True)
```

```
    except Exception:
        data = None
    json_output = response.data.decode('utf-8')

    expected_status = 500  # was 400 before
    result = "Success" if response.status_code == expected_status else "Failed"

    write_test_doc(
        title="***Case:*** *Missing submitted_answers field*",
        endpoint="http://127.0.0.1:5000/api/feedback-recommendations",
        method="POST",
        inputs=json.dumps(payload, indent=2),
        expected="HTTP Status Code: 500 and error message",
        actual=f"HTTP Status Code: {response.status_code}\nJSON: {json_output}",
        result=result
    )

    assert response.status_code == 500
    assert data and "error" in data
```

Case: *Internal server error due to invalid answer format*

Request Method: POST

Inputs:

```
{
  "submitted_answers": [
    {
      "qid": 1,
      "answer": "INVALID FORMAT"
    }
  ]
}
```

Expected Output:

```
HTTP Status Code: 400 and error message
```

Actual Output:

```
HTTP Status Code: 400
JSON: {
    "error": "answer must be a list of strings"
}
```

Result: Success

Pytest Code:

```
def test_feedback_internal_server_error(client):
    payload = {
        "submitted_answers": [
            {"qid": 1, "answer": "INVALID FORMAT"}  # 'answer' should be a list
        ]
    }

    response = client.post("/api/feedback-recommendations", json=payload)
    try:
        data = response.get_json(force=True, silent=True)
    except Exception:
        data = None
    json_output = response.data.decode('utf-8')

    expected_status = 400
    result = "Success" if response.status_code == expected_status else "Failed"

    write_test_doc(
        title="***Case:*** *Internal server error due to invalid answer format*",
        endpoint="http://127.0.0.1:5000/api/feedback-recommendations",
        method="POST",
        inputs=json.dumps(payload, indent=2),
        expected="HTTP Status Code: 400 and error message",
        actual=f"HTTP Status Code: {response.status_code}\nJSON: {json_output}",
        result=result
    )

    assert response.status_code == 400
    assert data and "error" in data if isinstance(data, dict) else True
```
