

Software Engineering Project

Sep 25 - Team 38

Milestone 5



Team Members

Name	Roll Number
Omkar Shankar Pawar	23F1001898
Palash Johri	21F1000512
Prabhu	21F2001015
Raghavendra Narayan Jha	21F1003534
Rishav Kumar	22F3001352
Treasa Janet	21F1002081
Sohini Sarkar	21F2001369

EduAssist Backend – Comprehensive Test Documentation

1. Introduction

This document provides a comprehensive overview of the backend test cases implemented for the EduAssist platform, covering all major modules such as assessment generation, knowledge assistant flows, study guide generation, slide deck creation, workflow automation, feedback system, and intelligent chat processing.

Each section includes:

- API tested
- Preconditions
- Input payload
- Expected output
- Behavior / purpose of the test
- Failure scenarios

The structure follows the same format used in the uploaded reference document.

2. Assessment Generator

Test File: test_assessment.py

APIs Tested

- GET /assessments/{assessment_id}
-

2.1 Test Case: Get Assessment – Success

API:

GET /assessments/1

Preconditions

- Assessment id = 1 exists
- created_by = 10 (current instructor user)

Input:

No body parameters (path only)

Expected Output

Status: 200 OK

```
{  
  "id": 1,  
  "title": "Unit Test Assessment",  
  "created_by": 10  
}
```

Behavior Validated

- Returns full assessment details

- Ensures authenticated instructor can access their own assessment
-

2.2 Test Case: Get Assessment – Not Found (FAILURE)

API:

GET /assessments/9999

Preconditions

- Assessment with ID 9999 does not exist

Expected Output

Status: 404

```
{  
  "detail": "Assessment not found"  
}
```

Behavior Validated

- Proper 404 handling
 - Validates missing resource error flow
-

3. Knowledge Assistant – Conversation System

Test File: test_knowledge.py

APIs Tested

- **POST /knowledge/conversations**
 - **GET /knowledge/conversations/{conversation_id}**
-

3.1 Create Conversation – Success

API:

POST /knowledge/conversations

Input

```
{  
  "course_id": 1  
}
```

Preconditions

- **Logged-in student: id = 101**

Expected Output

Status: 201

```
{  
  "id": 1,  
  "course_id": 1,  
  "student_id": 101,
```

```
"messages": []  
}
```

Behavior Validated

- Conversation gets created for the student
 - Ensures correct ownership
-

3.2 Get Conversation – Forbidden (FAILURE)

API:

GET /knowledge/conversations/1

Preconditions

- Conversation belongs to student_id = 999
- Current student = 101

Expected Output

Status: 403

```
{  
  "detail": "Not authorized"  
}
```

Behavior Validated

- Access is blocked when user does not own the conversation

4. Study Guide Generator

Test File: test_study_guide.py

APIs Tested

- POST /study-guides/
- GET /study-guides/{guide_id}

4.1 Create Study Guide – Success

Input

```
{  
  "youtube_url": "https://youtube.com/watch?v=abc123",  
  "title": "Test Guide",  
  "course_id": 1,  
  "priority_segments": []  
}
```

Mocked Preconditions

- Extracted video ID: "abc123"
- Metadata returned: title, duration, thumbnail
- Captions available

- CRUD returns guide id = 1

Expected Output

Status: 201

```
{  
  "id": 1,  
  "title": "Test Guide"  
}
```

Behavior Validated

- Successfully creates a study guide when required metadata exists

4.2 Create Study Guide – No Captions (FAILURE)

Mock

- check_captions_available → False

Expected Output

Status: 400

```
{  
  "detail": "Video does not have captions available"  
}
```


Behavior Validated

- System rejects videos without captions
 - Prevents unusable study guides
-

5. Slide Deck Generator

Test File: test_slide_deck.py

APIs Tested

- POST /slide-decks/
 - GET /slide-decks/{deck_id}
-

5.1 Create Slide Deck – Success

Input

```
{  
  "course_id": 1,  
  "title": "Intro Slides",  
  "source_type": "DOCUMENT",  
  "source_id": 5  
}
```

Expected Output

Status: 201

```
{  
  "id": 1,  
  "title": "Intro Slides"  
}
```

Behavior Validated

- Slide deck gets created from a document source

5.2 Slide Deck – Not Found (FAILURE)

API:

GET /slide-decks/9999

Expected Output

Status: 404

```
{  
  "detail": "Slide deck not found"  
}
```

Behavior Validated

- Ensures proper missing-deck handling

6. Workflow Automation

Test File: test_workflow_agent.py

APIs Tested

- POST /workflow/workflow-requests/
 - GET /workflow/workflow-requests/{id}
-

6.1 Create Workflow Request – Success

Input

```
{  
  "course_id": 1,  
  "description": "Need help setting grading rules",  
  "priority": "MEDIUM"  
}
```

Expected Output

Status: 201

```
{  
  "id": 1,  
  "course_id": 1,  
  "status": "PENDING"  
}
```

Behavior Validated

- New workflow request is created and stored

6.2 Workflow Request Not Found – FAILURE

API:

GET /workflow/workflow-requests/9999

Expected Output

Status: 404

```
{  
  "detail": "Workflow request not found"  
}
```

7. Feedback System

Test File: test_feedback.py

APIs Tested

- POST /feedback/
- GET /feedback/

7.1 Create Feedback – Success

Input

```
{  
  "rating": 4,  
  "comment": "Helpful answer",  
  "context": {  
    "type": "chat",  
    "id": 10  
  }  
}
```

Expected Output

Status: 201

```
{  
  "id": 1,  
  "rating": 4,  
  "comment": "Helpful answer"  
}
```

7.2 Create Feedback – Invalid Rating (FAILURE)

Input

```
{
```

```
"rating": 11,  
"comment": "Too high",  
"context": { "type": "chat", "id": 10 }  
}
```

Expected Output

- Status: 400 or 422 (Pydantic validation error)

Behavior Validated

- Ensures rating boundaries

8. Knowledge Assistant – Chat Flow

Test File: test_knowledge_chat.py

API Tested

- POST /knowledge/chat

8.1 Chat – New Conversation Success

Input

```
{  
  "course_id": 1,
```

```
"query": "What is RAG?",  
"conversation_id": null  
}
```

Expected Response

Status: 200

```
{  
  "message": { "content": "AI answer" },  
  "confidence_score": <0-1>,  
  "should_escalate": true/false  
}
```

8.2 Conversation Not Found – FAILURE

Input

```
{  
  "course_id": 1,  
  "query": "Hello",  
  "conversation_id": 9999  
}
```

Expected Output

Status: 404

```
{  
  "detail": "Conversation not found"  
}
```

9. Conclusion

This document consolidates all backend test scenarios for EduAssist. It provides clear documentation for:

- Positive and negative test paths
- Validation of business rules
- Authorization flows
- Handling of missing resources
- Integration with external services (RAG engine, etc.)