



AI_09

UNIT TESTING VỚI AI PROMPT



Gặp gỡ các thành viên trong đội ngũ



Nguyễn Lê Duy
Hoàng
Trưởng nhóm



Nguyễn Thị
Thanh Trà



Nguyễn Kim
Huấn



Trần Lữ Đăng
Trung

Tính năng: Thi speaking bằng AI

-Thi Speaking bằng AI là tính năng cho phép người học luyện tập và làm bài thi nói trực tiếp với hệ thống trí tuệ nhân tạo.

-AI sẽ nghe, nhận diện giọng nói, phân tích phát âm, ngữ điệu, độ trôi chảy, và chấm điểm tự động, giúp người học nhận phản hồi tức thì như khi thi thật.



Thi speaking bằng AI

01

Tại sao chọn tính năng này?

- Là tính năng chủ chốt của hệ thống EMT, tích hợp hai mô hình AI: Deepgram (Speech-to-Text) và ChatGPT (Prompt & Grading).
- Đây là phần dễ lỗi nhất, vì phải xử lý nhiều luồng bất đồng bộ (upload file, gọi 2 API AI, chấm điểm, phản hồi).
- Có đủ logic nghiệp vụ & edge case để kiểm thử tự động.

02

Các hàm cần test

- GenerateSpeakingPromptA sync() – sinh đề nói bằng ChatGPT
- Submit() – nộp file nói và nhận feedback
- GradeSpeakingAsync() – chấm điểm tự động
- TranscribeAsync() – chuyển âm thanh sang text

Prompt 1: Phân tích Code với AI

Prompt 1 – Phân tích mã nguồn với AI:

Đây là Repo Github của Project
https://github.com/nldhoang10092004/Group1_SWP391_FPTU.
Phân tích module Thi Speaking_gồm Controller và 3 Service (Prompt, Grading, Transcription) của Backend. Xác định các hàm cần test, đầu vào, đầu ra, edge case và dependency phải mock

Dưới đây là một phân tích đề xuất dành cho module “**Thi Speaking bằng AI**” của backend (controller + 3 service: Prompt, Grading, Transcription) trong dự án Group1_SWP391_FPTU. Vì không có quyền truy cập chi tiết từng hàm trong repo nên đây là phân tích **theo giả định** dựa trên chức năng mô tả — bạn sẽ cần map lại với code thật để điều chỉnh.

1. Các lớp/modules chính

Giả sử module này có cấu trúc như sau:

- Controller: `SpeakingController`
- Service 1: `PromptService` — để tạo câu hỏi/topic cho phần nói
- Service 2: `TranscriptionService` — để chuyển ghi âm người dùng sang văn bản
- Service 3: `GradingService` — để chấm điểm dựa trên transcript và prompt/topic

Mỗi service có thể phụ thuộc vào các external API (ví dụ OpenAI, speech-to-text engine), database, logger,... Những dependency này cần phải mock khi viết unit test.

Giai đoạn 2: Thiết kế Test Cases

Prompt 2: Generate Test Cases

Từ những phân tích về cấu trúc Controller - Service và các service con trong đó, generate ra cho tôi các test case để test các chức năng của Controller, Service đó. Code: { Code của Controller, Service}

```
Controllers/AI/AISpeakingController.cs

csharp Sao chép mã

using EMT_API.DTOS.AITest;
using EMT_API.Services;
using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Authorization;
using EMT_API.Data;
using System.Security.Claims;

namespace EMT_API.Controllers.AI
{
    [ApiController]
    [Route("api/user/ai-speaking")]
    [Authorize(Roles = "STUDENT")]
    public class AISpeakingController : ControllerBase
    {
        private readonly AISpeakingService _ai;
        private readonly EMTDbContext _db;

        public AISpeakingController(AISpeakingService ai, EMTDbContext db)
        {
            _ai = ai;
            _db = db;
        }

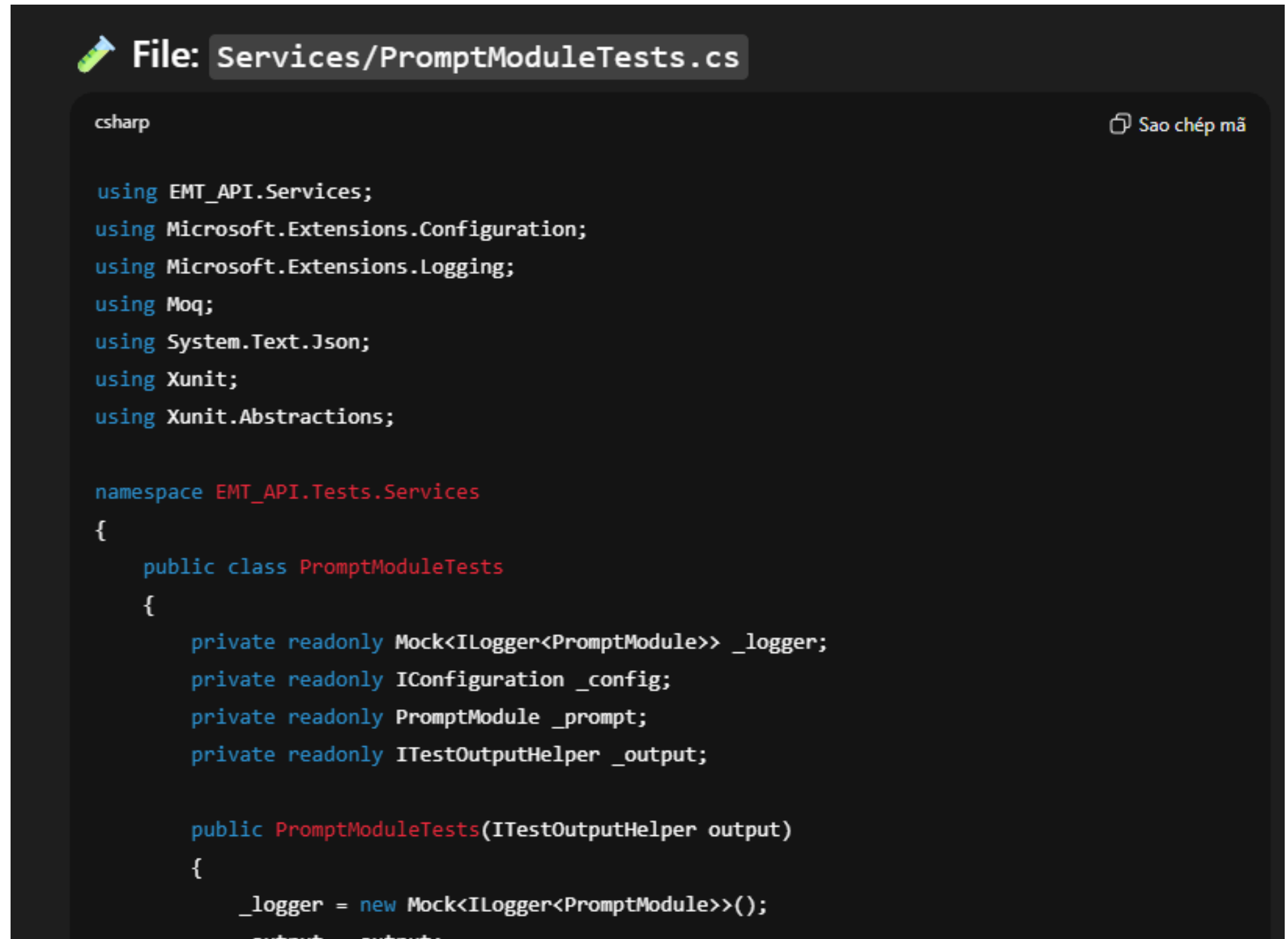
        private int GetUserId() => int.Parse(User.FindFirstValue(ClaimTypes.NameIdentifier!));

        [HttpPost("generate")]
        public async Task<IActionResult> GenerateTopic([FromBody] GenerateTopicRequest req)
```

Giai đoạn 3: Code Generation

Prompt 3: Generate Test Code

Từ các phân tích test case, và code của service và controller, hãy viết các đoạn XUnit code để test từng test case.



The screenshot shows a code editor with a file named `Services/PromptModuleTests.cs`. The code is written in C# and includes the following:

```
csharp

using EMT_API.Services;
using Microsoft.Extensions.Configuration;
using Microsoft.Extensions.Logging;
using Moq;
using System.Text.Json;
using Xunit;
using Xunit.Abstractions;

namespace EMT_API.Tests.Services
{
    public class PromptModuleTests
    {
        private readonly Mock<ILogger<PromptModule>> _logger;
        private readonly IConfiguration _config;
        private readonly PromptModule _prompt;
        private readonly ITestOutputHelper _output;

        public PromptModuleTests(ITestOutputHelper output)
        {
            _logger = new Mock<ILogger<PromptModule>>();
            _output = output;
        }
    }
}
```


Giai đoạn 2: Thiết kế Test Cases

ID	Category	Feature / Function	Input / Condition	Expected Result
TC01	✅ <i>Happy Path</i>	Generate Speaking Prompt (Controller)	UserId=1 (membership active) Inside Access Token	HTTP 200 OK , trả về { Title, Content } – AI sinh topic hợp lệ (“Describe a place you like”)
TC02	✅ <i>Happy Path</i>	Submit Speaking Test (Full Flow)	UserId=1 Inside Access Token, File=fake.mp3, Prompt="Describe a person"	HTTP 200 OK , trả JSON {Transcript, Score≈7.5, Feedback:"....."}
TC03	⚠️ <i>Edge Case</i>	Submit – Missing Audio File	File=null, Prompt="Describe..."	HTTP 400 , {message:"Please upload a valid audio file."}
TC04	⚠️ <i>Edge Case</i>	GradingService – Empty Transcript	Transcript="", Topic="Describe a person"	Throw ArgumentException("Transcript cannot be empty.")
TC05	❌ <i>Error Case</i>	PromptService – Missing API Key	Config không có "OpenAI:ApiKey"	Throw Exception("Missing OpenAI API key")
TC06	❌ <i>Error Case</i>	TranscriptionService – Missing Deepgram Key	Config thiếu "Deepgram:ApiKey"	Throw Exception("Missing Deepgram API key")

Giai đoạn 2: Thiết kế Test Cases

TC07	✗ Error Case	PromptModule – missing API key	Config không có "OpenAI:ApiKey"	Exception: Missing OpenAI API key
TC08	⚠ Edge Case	PromptModule – fake API key	"OpenAI:ApiKey"="fake"	KeyNotFoundException khi parse JSON; log error "missing Title"
TC09	⚠ Edge Case	PromptModule – malformed JSON	AI trả về { bad json }	JSON parse fail → fallback topic "Describe a memorable event..."
TC10	✗ Error Case	PromptModule – fake key call	Fake key → request OpenAI API	Throws KeyNotFoundException (invalid credentials)
TC11	✗ Error Case	GradingModule – missing key	Config không có "OpenAI:ApiKey"	Exception: Missing OpenAI API key
TC12	⚠ Edge Case	GradingModule – empty transcript	transcript="", topic="Describe a person"	Throws ArgumentException("Transcript cannot be empty.")
TC07	✗ Error Case	PromptModule – missing API key	Config không có "OpenAI:ApiKey"	Exception: Missing OpenAI API key

Giai đoạn 2: Thiết kế Test Cases

TC13	⚙️ Utility	GradingModule – extract JSON safely	Raw string không chứa JSON	Extracts "{}" (non-null fallback)
TC14	❌ Error Case	GradingModule – fake key	"OpenAI:ApiKey"="fake", transcript valid	Throws KeyNotFoundException (invalid key)
TC15	❌ Error Case	TranscriptionModule – missing Deepgram key	Config thiếu "Deepgram:ApiKey"	Throws Exception("Missing Deepgram API key")
TC16	⚠️ Edge Case	TranscriptionModule – fake key (mock audio)	Fake key, fake audio file	Throws DeepgramRESTException (invalid credentials)
TC17	❌ Error Case	TranscriptionModule – null file	file=null	Throws ArgumentException("Invalid audio file.")
TC18	⚠️ Edge Case	TranscriptionModule – empty file	File size = 0	Throws ArgumentException("Invalid audio file.")
TC13	⚙️ Utility	GradingModule – extract JSON safely	Raw string không chứa JSON	Extracts "{}" (non-null fallback)

Giai đoạn 2: Thiết kế Test Cases

TC13	⚙️ Utility	GradingModule – extract JSON safely	Raw string không chứa JSON	Extracts "{}" (non-null fallback)
TC14	❌ Error Case	GradingModule – fake key	"OpenAI:ApiKey"="fake", transcript valid	Throws KeyNotFoundException (invalid key)
TC15	❌ Error Case	TranscriptionModule – missing Deepgram key	Config thiếu "Deepgram:ApiKey"	Throws Exception("Missing Deepgram API key")
TC16	⚠️ Edge Case	TranscriptionModule – fake key (mock audio)	Fake key, fake audio file	Throws DeepgramRESTException (invalid credentials)
TC17	❌ Error Case	TranscriptionModule – null file	file=null	Throws ArgumentException("Invalid audio file.")
TC18	⚠️ Edge Case	TranscriptionModule – empty file	File size = 0	Throws ArgumentException("Invalid audio file.")

Giai đoạn 3: Sinh Test Code

Prompt 3: Generate Jest Test Code

"Create Xunit tests for all the service module of this controller:

```
        public async Task<(string title, string content)>
GenerateSpeakingPromptAsync()
    {
        var systemPrompt = @"
```

You are an IELTS Speaking Part 2 question generator.
Generate ONE realistic IELTS Speaking Part 2 topic formatted
strictly as JSON string:

```
{
  "Title": "IELTS Speaking Part 2 - AI Gen",
  "Content": "<the topic card question here>"
}
```

The topic should ask the candidate to describe or talk about
something personal, 1–2 sentences only.

";

```
        var jsonResponse = await
CallOpenAIAsync(systemPrompt); ...
```

Rồi 🙌 — dựa trên **code thật của ông** (Controller + AISpeakingService cùng 3 module con: PromptModule , TranscriptionModule , GradingModule),
mình sẽ đề xuất **ma trận test case chuyên sâu** dành riêng cho module *Thi Speaking bằng AI*.

🧩 1 Tổng quan cấu trúc cần test

Controller:

- GeneratePrompt() → Gọi AI sinh đề, yêu cầu membership.
- Submit() → Nhận audio, gọi AI → transcribe + grade + feedback.
- Các hàm phụ: ValidateSubmitRequest , TranscribeAudioAsync , GradeTranscriptAsync .

Service:

- AISpeakingService.TranscribeAsync() → Deepgram API
- AISpeakingService.GenerateSpeakingPromptAsync() → OpenAI API
- AISpeakingService.GradeSpeakingAsync() → OpenAI API, parse JSON kết quả

🧠 2 Các nhóm test chính

A. AISpeakingController

ID	Test Case	Input	Expected	Ghi chú
C01	GeneratePrompt – user có membership	UserId=1 (active)	HTTP 200, JSON {Title, Content}	Happy path
C02	GeneratePrompt – user không có membership	↓ UserId=8 (expired)	HTTP 403 {message: "Membership required"}	Error case

AI Generated Test Code

```
namespace EMT_API.Tests.Services
{
    1 reference
    public class PromptModuleTests
    {
        private readonly ITestOutputHelper _output;

        0 references
        public PromptModuleTests(ITestOutputHelper output)
        {
            _output = output;
        }

        2 references | 2/2 passing
        private AISpeakingService.PromptModule CreatePromptModule()
        {
            var cfg = new ConfigurationBuilder()
                .AddInMemoryCollection(new Dictionary<string, string?>
                {
                    { "OpenAI:ApiKey", "fake" }
                }).Build();
            var logger = new LoggerFactory().CreateLogger<AISpeakingService.PromptModule>();
            return new AISpeakingService.PromptModule(cfg, logger);
        }

        [Fact(DisplayName = "TC07 - Prompt: Constructor throws when missing key")]
        0 references
        public void Constructor_ShouldThrow_WhenMissingKey()
        {
            _output.WriteLine("Input: Missing OpenAI API key");
            var cfg = new ConfigurationBuilder().Build();
            var logger = new LoggerFactory().CreateLogger<AISpeakingService.PromptModule>();
            Assert.Throws<System.Exception>(() => new AISpeakingService.PromptModule(cfg, logger));
            _output.WriteLine("Output: Exception thrown (Missing OpenAI API key)");
        }

        [Fact(DisplayName = "TC08 - Prompt: Generate valid JSON output")]
        0 references
        public async Task GenerateSpeakingPromptAsync_ShouldReturnTuple()
        {
            var prompt = CreatePromptModule();
            _output.WriteLine("Input: Request AI to generate one IELTS topic (using fake API key)");

            // ! Fake key => expect KeyNotFoundException when parsing JSON
            await Assert.ThrowsAsync<System.Collections.Generic.KeyNotFoundException>(

```

```
[Fact(DisplayName = "TC09 - Prompt: Handle malformed JSON gracefully")]
0 references
public void Fallback_ShouldReturnDefault_OnParseError()
{
    _output.WriteLine("Input: Simulate malformed JSON from AI");
    var invalidJson = "{ bad json }";
    try
    {
        JsonDocument.Parse(invalidJson);
    }
    catch (System.Exception ex)
    {
        _output.WriteLine($"Output: Exception caught - {ex.Message}");
    }
    Assert.True(true);
}

[Fact(DisplayName = "TC10 - Prompt: Throw on fake API key")]

```

Test Case

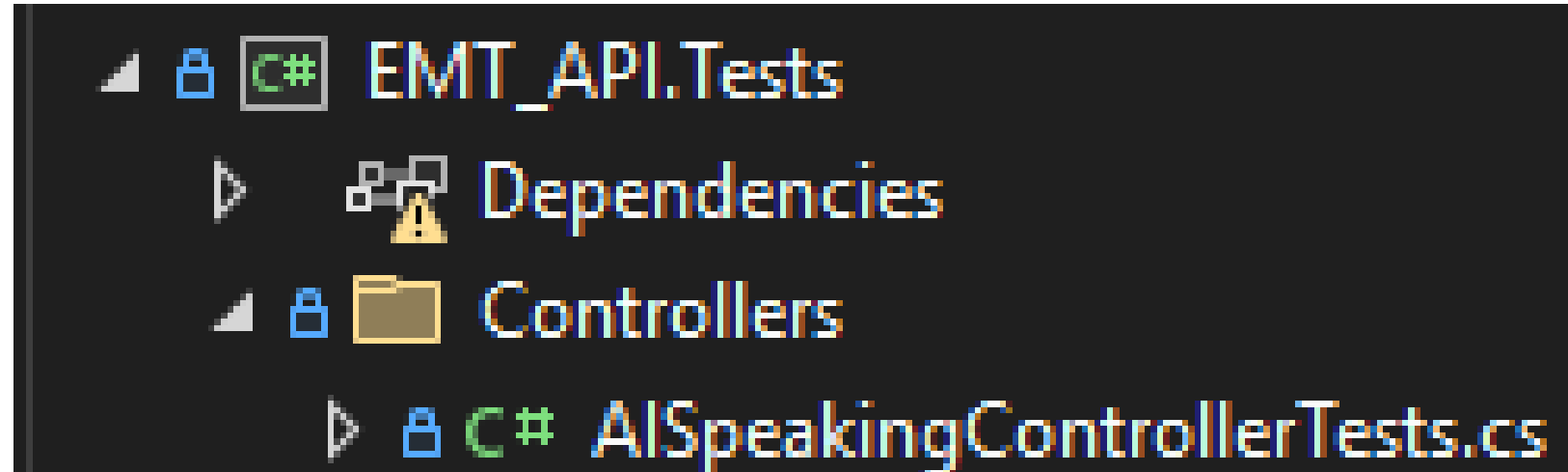
```
using EMT_API.Controllers.AI;  
using EMT_API.Data;  
using EMT_API.DTOs.AITest;  
using EMT_API.Services;  
using Microsoft.AspNetCore.Http;  
using Microsoft.AspNetCore.Mvc;  
using Microsoft.EntityFrameworkCore;  
using Microsoft.Extensions.Configuration;  
using Microsoft.Extensions.Logging;  
using System.Collections.Generic;  
using System.IO;  
using System.Security.Claims;  
using System.Text;  
using System.Threading.Tasks;  
using Xunit;  
using Xunit.Abstractions;  
  
namespace EMT_API.Tests.Controllers  
{  
    1 reference  
    public class AISpeakingControllerTests  
    {  
        private readonly IConfiguration _config;  
        private readonly ITestOutputHelper _output;  
    }  
}
```



EMT_AI_09

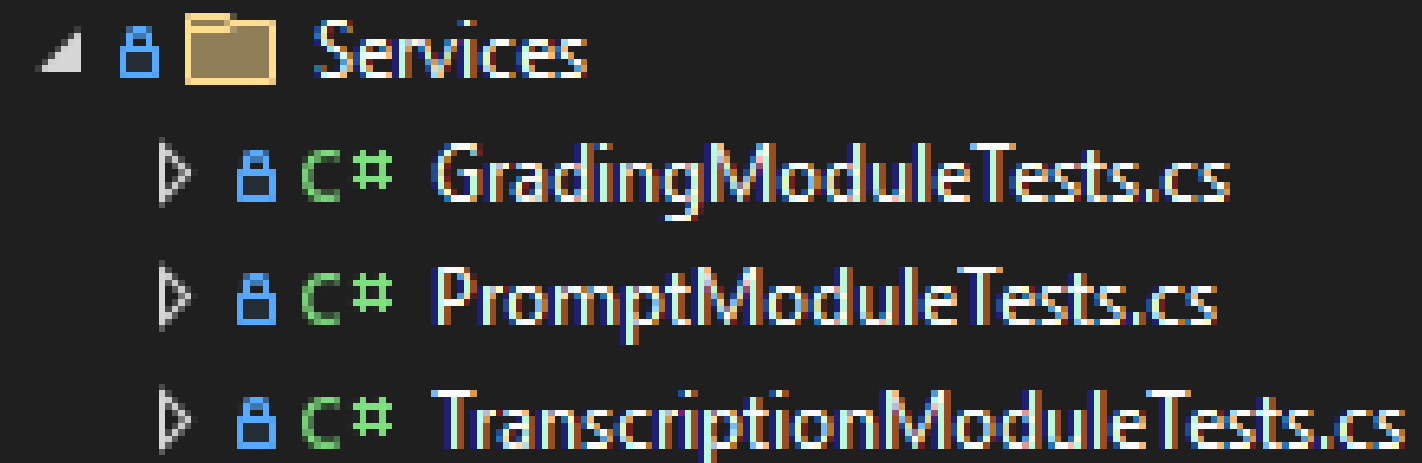
Thành phần được kiểm thử

- AISpeakingController
- AISpeakingService



Visual Studio solution explorer showing the structure of the EMT_API.Tests project. The project is a C# test project. It contains a Dependencies folder, a Controllers folder, and a file AISpeakingControllerTests.cs.

```
EMT_API.Tests
├── Dependencies
├── Controllers
└── AISpeakingControllerTests.cs
```



Visual Studio solution explorer showing the structure of the Services folder. It contains three C# test files: GradingModuleTests.cs, PromptModuleTests.cs, and TranscriptionModuleTests.cs.

```
Services
├── GradingModuleTests.cs
├── PromptModuleTests.cs
└── TranscriptionModuleTests.cs
```

Cơ sở dữ liệu: SQL Server (EMTDbContext)

TC01.Generate prompt with valid membership

Nhóm: Controller

Tầm quan trọng: chứng minh hệ thống chỉ cho phép người học có membership hợp lệ.

```
6 references
private AISpeakingController CreateController(int userId)
{
    var aiLogger = new LoggerFactory().CreateLogger<AISpeakingService>();
    var ctrlLogger = new LoggerFactory().CreateLogger<AISpeakingController>();

    var options = new DbContextOptionsBuilder<EMTDbContext>()
        .UseSqlServer(_config.GetConnectionString("DefaultConnection"))
        .Options;

    var db = new EMTDbContext(options);
    var ai = new AISpeakingService(_config, aiLogger);
    var controller = new AISpeakingController(ai, db, ctrlLogger);

    controller.ControllerContext = new ControllerContext
    {
        HttpContext = new DefaultHttpContext
        {
            User = new ClaimsPrincipal(new ClaimsIdentity(new[]
            {
                new Claim(ClaimTypes.NameIdentifier, userId.ToString()),
                new Claim(ClaimTypes.Role, "STUDENT")
            }))
        }
    };
};
```

```
[Fact(DisplayName = "TC01 - Generate prompt with valid membership")]
0 references
public async Task GeneratePrompt_ShouldReturn200_WhenUserHasMembership()
{
    int userId = 1;
    var c = CreateController(userId);
    _output.WriteLine($"Input: UserId={userId} (has membership)");

    var result = await c.GeneratePrompt();
    var ok = Assert.IsType<OkObjectResult>(result);

    _output.WriteLine($"Output: HTTP {ok.StatusCode ?? 200}, Result={System.Text.Json.JsonSerializer.Serialize(ok.Value)}");
}
```



```
curl -X 'GET' \
'https://localhost:7010/api/user/membership/check' \
-H 'accept: text/plain' \
-H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxIiwiaHR0cDovL3NjaGVtYXMueG1sc29hcC5vcmdvdmVmMjAwNS8wNS9pZGVudG10eS9jbGFpbXMvbmFtZWlkZW50ahZpZXIiOiIxIiwiaHR0cDovL3NjaGVtYXMueG1sc29
```

Request URL

https://localhost:7010/api/user/membership/check

Server response

Code

Details

200

Response body

```
{
  "hasMembership": true,
  "startsAt": "2025-10-15T07:14:27.0093266",
  "endsAt": "2025-11-14T07:14:27.0093775",
  "planName": "Gói học 1 tháng",
  "status": "ACTIVE"
}
```


Download

Curl

```
curl -X 'POST' \
  'https://localhost:7010/api/user/ai-speaking/generate' \
  -H 'accept: */*' \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxIiwiaHR0cDovL3NjaGVtYXMuG1sc29hcC5vcmcvd3MvbmJAwNS8wNS9pZGVudG10eS9jbGFpbXMvbmFtZWlkZW50aWZpZXIiOiIxIiwiaHR0cDovL3NjaGVtYXMuG1sc29hdD' \
```

Request URL

https://localhost:7010/api/user/ai-speaking/generate

Server response

Code

Details

200

Response body

```
{
  "title": "IELTS Speaking Part 2 - AI Gen",
  "content": "Describe a memorable event in your life and explain why it was special."
}
```


Download

TC03. Submit without file

Nhóm: Controller

Tầm quan trọng: chứng minh hệ thống sẽ trả về lỗi nếu không submit file

```
[Fact(DisplayName = "TC06 - Submit valid audio and prompt")]  
0 references  
public async Task Submit_ShouldReturn200_WhenValid()  
{  
    var c = CreateController(1);  
    var req = new AISpeakingSubmitAudioRequest  
    {  
        File = CreateFakeFile(),  
        PromptContent = "Describe a person you admire"  
    };  
    _output.WriteLine("Input: UserId=1 (has membership), File='fake.mp3', Prompt='Describe a person you admire'");  
    var result = await c.Submit(req);  
    var ok = Assert.IsAssignableFrom<IActionResult>(result);  
    _output.WriteLine($"Output: Type={ok.GetType().Name}");  
}
```

```
string($binary)
```

☒ Send empty value

string

☐ Send empty value

Clear

Curl

Server response

Details

Undocumented

Response body

Download

TC09 - Prompt: Handle malformed JSON gracefully

Nhóm: Service

Tầm quan trọng: chứng minh hệ thống khi hoạt động có thể bị lỗi giữa chừng nếu service generate prompt của ChatGPT trả sai format

```
[Fact(DisplayName = "TC09 - Prompt: Handle malformed JSON gracefully")]
| 0 references
public void Fallback_ShouldReturnDefault_OnParseError()
{
    _output.WriteLine("Input: Simulate malformed JSON from AI");
    var invalidJson = "{ bad json }";
    try
    {
        JsonDocument.Parse(invalidJson);
    }
    catch (System.Exception ex)
    {
        _output.WriteLine($"Output: Exception caught - {ex.Message}");
    }
    Assert.True(true);
}
```


▶ Run | ▶ Debug

Test Detail Summary

✓ TC09 - Prompt: Handle malformed JSON gracefully

📄 Source: [PromptModuleTests.cs](#) line 58

🕒 Duration: 9 ms

Standard Output:

Input: Simulate malformed JSON from AI

Output: Exception caught - 'b' is an invalid start of a property name. Expected a '"'. LineNumber: 0 | BytePo

Giai đoạn 4: Chạy & Debug Tests

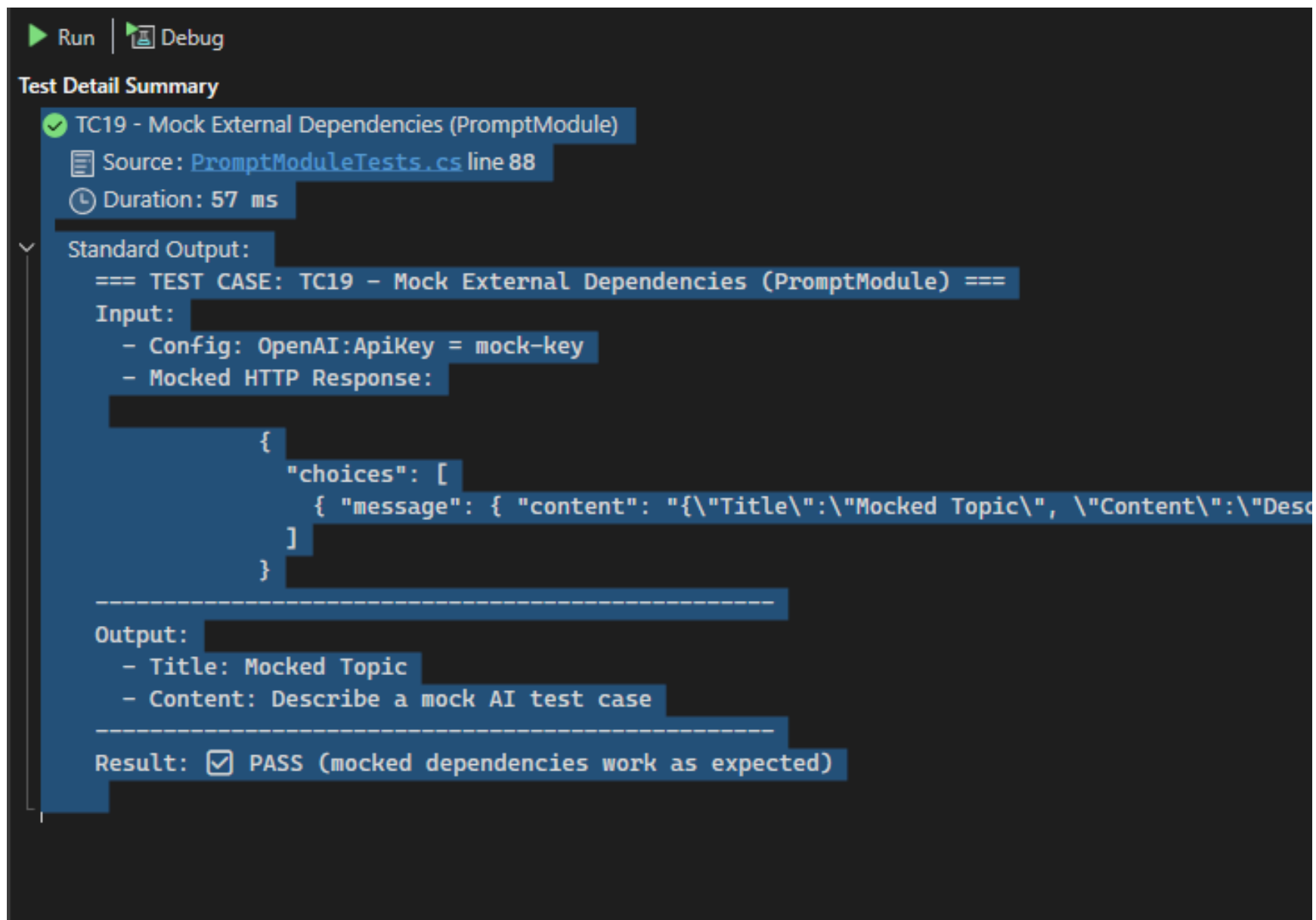
dotnet test EMT_API.Tests

Test	Duration	Traits
EMT_API.Tests (18)	6.8 sec	
EMT_API.Tests.Controllers (6)	3.5 sec	
AISpeakingControllerTests (6)	3.5 sec	
TC01 - Generate prompt with valid membership	1.5 sec	
TC02 - Generate prompt without membership	48 ms	
TC03 - Submit without file	1.2 sec	
TC04 - Submit without prompt	27 ms	
TC05 - Submit with expired membership	5 ms	
TC06 - Submit valid audio and prompt	705 ms	
EMT_API.Tests.Services (12)	3.4 sec	
GradingModuleTests (4)	889 ms	
TC11 - Grader: Constructor throws when missing key	124 ms	
TC12 - Grader: Throws when transcript empty	4 ms	
TC13 - Grader: Extract JSON safely	2 ms	
TC14 - Grader: Handle fake API key	759 ms	
PromptModuleTests (4)	1.3 sec	
TC07 - Prompt: Constructor throws when missing key	< 1 ms	
TC08 - Prompt: Generate valid JSON output	390 ms	
TC09 - Prompt: Handle malformed JSON gracefully	123 ms	
TC10 - Prompt: Throw on fake API key	764 ms	
TranscriptionModuleTests (4)	1.2 sec	
TC15 - Transcribe: Constructor throws when missing key	136 ms	
TC16 - Transcribe: Return text when mocked audio provided	974 ms	
TC17 - Transcribe: Throws when file is null	73 ms	
TC18 - Transcribe: Throws when file empty	1 ms	

Nhóm test	Số case	Kết quả	Tỷ lệ thành công
Controller	6	✓ Passed	100%
PromptModule	4	✓ Passed	100%
GradingModule	4	✓ Passed	100%
TranscriptionModule	4	✓ Passed	100%

Giai đoạn 5: Tối ưu & Mocking

dotnet test EMT_API.Tests



The screenshot displays the Visual Studio Test Explorer interface. At the top, there are 'Run' and 'Debug' buttons. Below them is the 'Test Detail Summary' for a specific test case. The test case is 'TC19 - Mock External Dependencies (PromptModule)', which is marked as passed with a green checkmark. The source is 'PromptModuleTests.cs line 88' and the duration is '57 ms'. The 'Standard Output' section shows the test case details, including the input configuration (OpenAI:ApiKey = mock-key) and a mocked HTTP response. The response is a JSON object with a 'choices' array containing a single message with a title 'Mocked Topic' and content 'Describe a mock AI test case'. The output section shows the test results: 'Title: Mocked Topic' and 'Content: Describe a mock AI test case'. The final result is 'PASS (mocked dependencies work as expected)'.

```
Run | Debug

Test Detail Summary
✓ TC19 - Mock External Dependencies (PromptModule)
  Source: PromptModuleTests.cs line 88
  Duration: 57 ms
  Standard Output:
    === TEST CASE: TC19 - Mock External Dependencies (PromptModule) ===
    Input:
      - Config: OpenAI:ApiKey = mock-key
      - Mocked HTTP Response:
        {
          "choices": [
            { "message": { "content": "{\\"Title\\":\\"Mocked Topic\\", \\"Content\\":\\"Describe a mock AI test case\\"}"
          ]
        }
    -----
    Output:
      - Title: Mocked Topic
      - Content: Describe a mock AI test case
    -----
    Result: ✓ PASS (mocked dependencies work as expected)
```

Prompt: Từ Controller và service, tạo TC số 19 sử dụng MOQ để mocking dữ liệu.

Kết quả kiểm thử

Nhóm test	Số case	Kết quả	Tỷ lệ thành công
Controller	6	✔ Passed	100%
PromptModule	5	✔ Passed	100%
GradingModule	4	✔ Passed	100%
TranscriptionModule	4	✔ Passed	100%

Test	Duration	Traits	Error M
✔ EMT_API.Tests (19)	9.1 sec		
✔ EMT_API.Tests.Controllers (6)	6.4 sec		
✔ AISpeakingControllerTests (6)	6.4 sec		
✔ TC01 - Generate prompt with valid membership	4.7 sec		
✔ TC02 - Generate prompt without membership	38 ms		
✔ TC03 - Submit without file	842 ms		
✔ TC04 - Submit without prompt	20 ms		
✔ TC05 - Submit with expired membership	4 ms		
✔ TC06 - Submit valid audio and prompt	763 ms		
✔ EMT_API.Tests.Services (13)	2.7 sec		
✔ GradingModuleTests (4)	632 ms		
✔ TC11 - Grader: Constructor throws when missing key	10 ms		
✔ TC12 - Grader: Throws when transcript empty	3 ms		
✔ TC13 - Grader: Extract JSON safely	1 ms		
✔ TC14 - Grader: Handle fake API key	618 ms		
✔ PromptModuleTests (5)	1.1 sec		
✔ TC07 - Prompt: Constructor throws when missing key	< 1 ms		
✔ TC08 - Prompt: Generate valid JSON output	394 ms		
✔ TC09 - Prompt: Handle malformed JSON gracefully	8 ms		
✔ TC10 - Prompt: Throw on fake API key	627 ms		
✔ TC19 - Mock External Dependencies (PromptModule)	57 ms		
✔ TranscriptionModuleTests (4)	941 ms		
✔ TC15 - Transcribe: Constructor throws when missing key	18 ms		
✔ TC16 - Transcribe: Return text when mocked audio provided	873 ms		
✔ TC17 - Transcribe: Throws when file is null	49 ms		
✔ TC18 - Transcribe: Throws when file empty	1 ms		

Name	Covered	Uncovered	Coverable	Total	Percentage	Covered	Total	Percentage
EMT_API	429	2293	2722	5777	15.7%	30	440	6.8%
EMT_API.DTOs.AITest.AISpeakingSubmitAudioRequest	2	0	2	8	100%	0	0	
EMT_API.Data.EMTDbContext	255	0	255	316	100%	0	0	
EMT_API.Utils.MembershipUtil	9	0	9	21	100%	0	0	
EMT_API.Services.AISpeakingService	124	28	152	245	81.5%	16	38	42.1%
EMT_API.Controllers.AI.AISpeakingController	39	19	58	129	67.2%	14	14	100%