using Microsoft.VisualStudio.TestTools.UnitTesting;

using System.Threading.Tasks;

[TestClass]

public class SemanticSimilarityTests

{

private SemanticSimilarity \_similarityChecker;

[TestInitialize]

public void Setup()

{

\_similarityChecker = new SemanticSimilarity();

}

[TestMethod]

public async Task TestHighSimilarity()

{

string text1 = "The cat is on the mat.";

string text2 = "A cat is sitting on the mat.";

float similarity = await \_similarityChecker.CalculateSimilarityAsync(text1, text2);

Assert.IsTrue(similarity > 0.8, $"Expected high similarity, but got {similarity}");

}

[TestMethod]

public async Task TestLowSimilarity()

{

string text1 = "The cat is on the mat.";

string text2 = "It is raining heavily today.";

float similarity = await \_similarityChecker.CalculateSimilarityAsync(text1, text2);

Assert.IsTrue(similarity < 0.4, $"Expected low similarity, but got {similarity}");

}

[TestMethod]

public async Task TestExactMatch()

{

string text1 = "The quick brown fox jumps over the lazy dog.";

string text2 = "The quick brown fox jumps over the lazy dog.";

float similarity = await \_similarityChecker.CalculateSimilarityAsync(text1, text2);

Assert.IsTrue(similarity >= 0.99, $"Expected similarity close to 1.0, but got {similarity}");

}

[TestMethod]

[ExpectedException(typeof(Exception))]

public async Task TestEmptyText()

{

string text1 = "";

string text2 = "The cat is on the mat.";

\_ = await \_similarityChecker.CalculateSimilarityAsync(text1, text2);

}

}