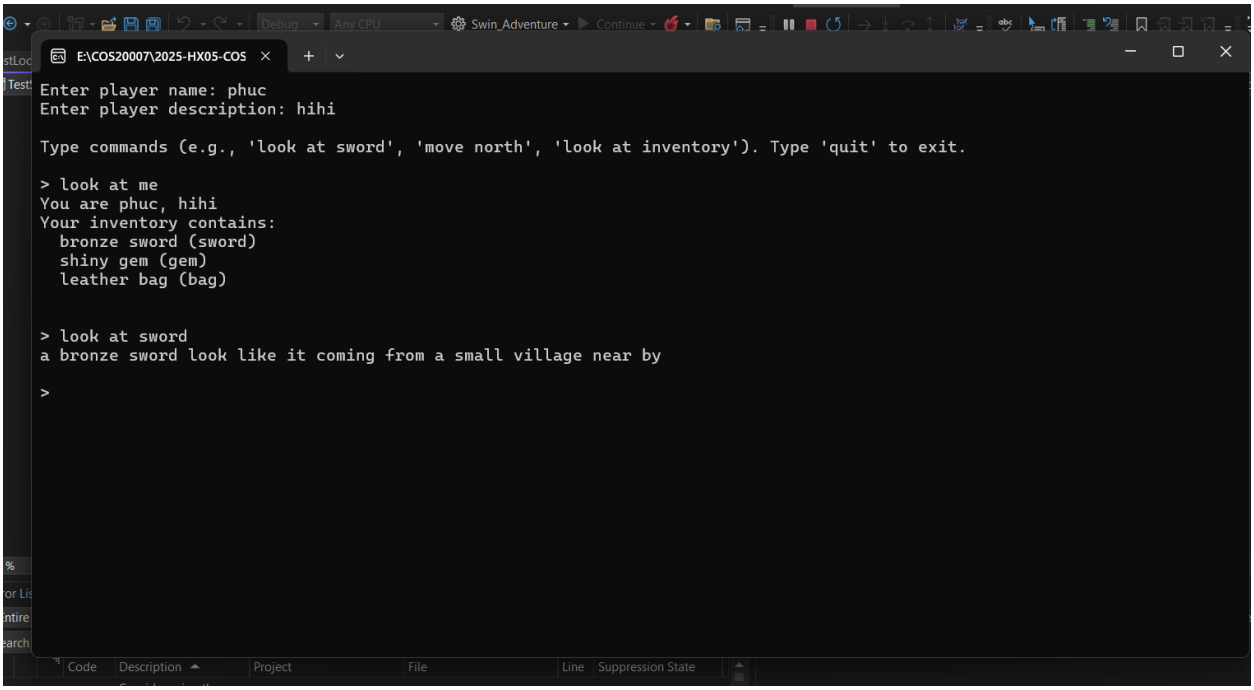
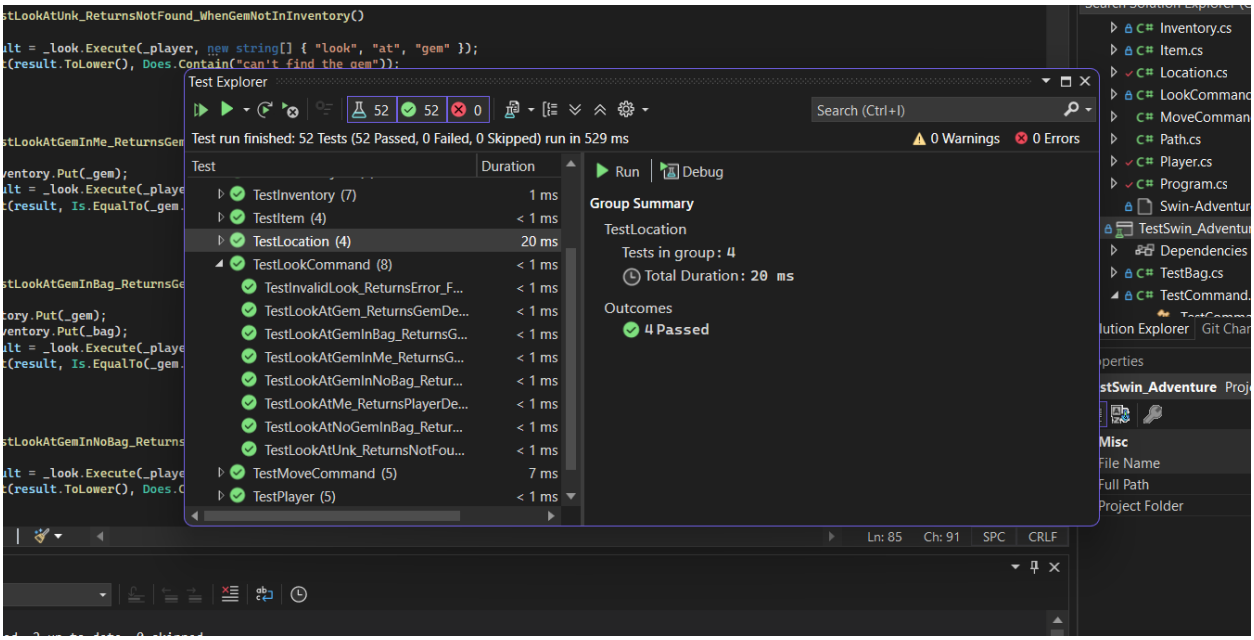
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Swin_Adventure
{
    public class LookCommand : Command
    {
        public LookCommand() : base(new string[] { "look" }) { }


        public override string Execute(Player p, string[] text)
        {
            // Support 'look around'
            if (text.Length == 2 && text[0] == "look" && text[1] == "around")
            {
                if (p.Location != null)
                    return p.Location.FullDescription;
                else
                    return "You are nowhere.";
            }

            // Support 'look at location'
            if (text.Length == 3 && text[0] == "look" && text[1] == "at" && text[2] == "location")
            {
                if (p.Location != null)
                    return p.Location.FullDescription;
                else
                    return "You are nowhere.";
            }

            if (text.Length != 3 && text.Length != 5)
                return "I don't know how to look like that";

            if (text[0] != "look")
                return "Error in look input";

            if (text[1] != "at")
                return "What do you want to look at";

            string itemId = text[2];
            IHaveInventory container;

            if (text.Length == 3)
            {
                container = p;
            }
            else
            {
                if (text[3] != "in")
                    return "What do you want to look in";

                string containerId = text[4];
                container = FetchContainer(p, containerId);
                if (container == null)
                    return $"can't find the {containerId}";
            }

            return LookAtIn(itemId, container);
        }


        private IHaveInventory FetchContainer(Player p, string containerId)
        {
            GameObject obj = p.Locate(containerId);
            return obj as IHaveInventory;
        }

        private string LookAtIn(string thingId, IHaveInventory container)
        {
            GameObject item = container.Locate(thingId);
            if (item == null)
                return $"can't find the {thingId}";
            return item.FullDescription;
        }
    }
}
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using NUnit.Framework;
using Swin_Adventure;

namespace TestSwin_Adventure
{
    [TestFixture]
    public class TestLookCommand
    {
        private Player _player;
        private Item _gem;
        private Bag _bag;
        private LookCommand _look;

        [SetUp]
        public void Setup()
        {
            _player = new Player("player", "the player");
            _gem = new Item(new string[] { "gem" }, "a gem", "a shiny gem");
            _bag = new Bag(new string[] { "bag" }, "a bag", "a small bag");
            _look = new LookCommand();
        }

        [Test]
        public void TestLookAtMe_ReturnsPlayerDescription_WhenLookingAtInventory()
        {
            string result = _look.Execute(_player, new string[] { "look", "at", "inventory" });
            Assert.That(result, Is.EqualTo(_player.FullDescription));
        }

        [Test]
        public void TestLookAtGem_ReturnsGemDescription_WhenGemInPlayerInventory()
        {
            _player.Inventory.Put(_gem);
            string result = _look.Execute(_player, new string[] { "look", "at", "gem" });
            Assert.That(result, Is.EqualTo(_gem.FullDescription));
        }

        [Test]
        public void TestLookAtUnk_ReturnsNotFound_WhenGemNotInInventory()
        {
            string result = _look.Execute(_player, new string[] { "look", "at", "gem" });
            Assert.That(result.ToLower(), Does.Contain("can't find the gem"));
        }

        [Test]
        public void TestLookAtGemInMe_ReturnsGemDescription_WhenLookingAtGemInInventory()
        {
            _player.Inventory.Put(_gem);
            string result = _look.Execute(_player, new string[] { "look", "at", "gem", "in", "inventory" });
            Assert.That(result, Is.EqualTo(_gem.FullDescription));
        }

        [Test]
        public void TestLookAtGemInBag_ReturnsGemDescription_WhenGemInBagInInventory()
        {
            _bag.Inventory.Put(_gem);
            _player.Inventory.Put(_bag);
            string result = _look.Execute(_player, new string[] { "look", "at", "gem", "in", "bag" });
            Assert.That(result, Is.EqualTo(_gem.FullDescription));
        }

        [Test]
        public void TestLookAtGemInNoBag_ReturnsNotFound_WhenBagNotInInventory()
        {
            string result = _look.Execute(_player, new string[] { "look", "at", "gem", "in", "bag" });
            Assert.That(result.ToLower(), Does.Contain("can't find the bag"));
        }

        [Test]
        public void TestLookAtNoGemInBag_ReturnsNotFound_WhenGemNotInBag()
        {
            _player.Inventory.Put(_bag);
            string result = _look.Execute(_player, new string[] { "look", "at", "gem", "in", "bag" });
            Assert.That(result.ToLower(), Does.Contain("can't find the gem"));
        }

        [Test]
        public void TestInvalidLook_ReturnsError_ForInvalidInputs()
        {
            // When player has no location, 'look around' should return 'You are nowhere.'
            Assert.That(_look.Execute(_player, new string[] { "look", "around" }), Is.EqualTo("You are nowhere."));
            Assert.That(_look.Execute(_player, new string[] { "hello", "105505856" }), Does.Contain("i don't know how to look like that"));
            Assert.That(_look.Execute(_player, new string[] { "look", "at", "phuc" }), Does.Contain("can't find the phuc"));
        }
    }
}
```


E:\COS20007\2025-HX05-COS
Enter player name: phuc
Enter player description: hihi
Type commands (e.g., 'look at sword', 'move north', 'look at inventory'). Type 'quit' to exit.
> look at me
You are phuc, hihi
Your inventory contains:
bronze sword (sword)
shiny gem (gem)
leather bag (bag)
> look at sword
a bronze sword look like it coming from a small village near by
>


Test Explorer
Test run finished: 52 Tests (52 Passed, 0 Failed, 0 Skipped) run in 529 ms
0 Warnings
0 Errors
Test
Duration
TestInventory (7) 1 ms
TestItem (4) < 1 ms
TestLocation (4) 20 ms
TestLookCommand (8) < 1 ms
TestMoveCommand (5) 7 ms
TestPlayer (5) < 1 ms
Run
Debug
Group Summary
TestLocation
Tests in group: 4
Total Duration: 20 ms
Outcomes
4 Passed