```
using System.Collections.Generic;
using NST.Enums;
using NST.Models.Nst;
using NST.ViewModels;
using NSTTest.IntegrationTesting.Shared;
using Xunit;

namespace NSTTest.UnitTesting
{
    public class SkillUnitTest
    {
        private readonly Skill _skill = new Skill();

        [Fact]
        public void FilterByCompletion_SkillsReturned()
        {
            var inputEvals = new List<SkillEvaluation>
            {
                new SkillEvaluation
                {
                    Skill = new Skill {SkillId = 1},
                    StudentEvaluation = new StudentEvaluation {SkillRatingId =
Rating.Satisfactory}
                },
                new SkillEvaluation
                {
                    Skill = new Skill {SkillId = 2},
                    StudentEvaluation = new StudentEvaluation {SkillRatingId =
Rating.Satisfactory}
                },
                new SkillEvaluation
                {
                    Skill = new Skill {SkillId = 3}
                },
                new SkillEvaluation
                {
                    Skill = new Skill {SkillId = 4},
                    StudentEvaluation = new StudentEvaluation {SkillRatingId =
Rating.Unsatisfactory}
                },
                new SkillEvaluation
                {
                    Skill = new Skill {SkillId = 5},
                    StudentEvaluation = new StudentEvaluation {SkillRatingId =
Rating.NeedsImprovement}
                }
            };

            var expectedSkills = new List<SkillEvaluation>
            {
                new SkillEvaluation
                {
                    Skill = new Skill {SkillId = 3}
                },
                new SkillEvaluation
                {
                    Skill = new Skill {SkillId = 4},
```

```
                    StudentEvaluation = new StudentEvaluation {SkillRatingId =
Rating.Unsatisfactory}
                }
            };

            var skills = _skill.FilterByCompletion(new[] {0, 1}, inputEvals);

            Assert.Equal(expectedSkills.Count, skills.Count);
            foreach (var skill in skills)
            {
                Assert.Contains(skill, expectedSkills, new
SkillSkillEvaluationComparer());
            }
        }

        [Fact]
        public void FilterByCompletion_NoSkillsReturned()
        {
            var inputEvals = new List<SkillEvaluation>
            {
                new SkillEvaluation
                {
                    Skill = new Skill {SkillId = 1},
                    StudentEvaluation = new StudentEvaluation {SkillRatingId =
Rating.Satisfactory}
                },
                new SkillEvaluation
                {
                    Skill = new Skill {SkillId = 2},
                    StudentEvaluation = new StudentEvaluation {SkillRatingId =
Rating.Satisfactory}
                },
                new SkillEvaluation
                {
                    Skill = new Skill {SkillId = 3},
                    StudentEvaluation = new StudentEvaluation {SkillRatingId =
Rating.NeedsImprovement}
                },
                new SkillEvaluation
                {
                    Skill = new Skill {SkillId = 4},
                    StudentEvaluation = new StudentEvaluation {SkillRatingId =
Rating.Satisfactory}
                },
                new SkillEvaluation
                {
                    Skill = new Skill {SkillId = 5},
                    StudentEvaluation = new StudentEvaluation {SkillRatingId =
Rating.NeedsImprovement}
                }
            };

            var skills = _skill.FilterByCompletion(new[] {0, 1}, inputEvals);

            Assert.Empty(skills);
        }

    }
```

```
}
```