```
using Microsoft.VisualStudio.TestTools.UnitTesting;
using CupPlaner.Controllers;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Web.Mvc;

namespace CupPlaner.Controllers.Tests
{
    [TestClass()]
    public class FieldControllerTests
    {
        FieldController controller = new FieldController();

        [TestMethod()]
        public void CreateTest()
        {
            //Create a new field
            dynamic jsonResult = ((JsonResult)controller.Create("Test Field", 8,
ID.TournamentId)).Data;
            ID.FieldId = jsonResult.id;
            Assert.AreEqual("success", jsonResult.status);

            //Create a new field using null values
            jsonResult = ((JsonResult)controller.Create(null, 8, ID.TournamentId)).Data;
            Assert.AreEqual("error", jsonResult.status);

            //Create a new field, but to a non-existing tournament
            jsonResult = ((JsonResult)controller.Create("Test Field", 8, 999999)).Data;
            Assert.AreEqual("error", jsonResult.status);
        }

        [TestMethod()]
        public void GetAllTournamentFieldsTest()
        {
            //Get all the tournament fields from the test tournament
            dynamic jsonResult =
((JsonResult)controller.GetAllTournamentFields(ID.TournamentId)).Data;
            Assert.AreEqual("success", jsonResult.status);
            Assert.AreEqual(1, jsonResult.Fields.Count);
            Assert.AreEqual("Test Field", jsonResult.Fields[0].Name);

            //Get the tournament fields from a non-existing tournament
            jsonResult = ((JsonResult)controller.GetAllTournamentFields(999999)).Data;
            Assert.AreEqual("error", jsonResult.status);
        }

        [TestMethod()]
        public void DetailsTest()
        {
            //Find the created field
            dynamic jsonResult = ((JsonResult)controller.Details(ID.FieldId)).Data;
            Assert.AreEqual("success", jsonResult.status);
            Assert.AreEqual(ID.FieldId, jsonResult.Id);
            Assert.AreEqual("Test Field", jsonResult.name);
            Assert.AreEqual(FieldSize.EightMan, jsonResult.size);
```

```
            //Find a field that does not exist
            jsonResult = ((JsonResult)controller.Details(999999)).Data;
            Assert.AreEqual("error", jsonResult.status);
        }

        [TestMethod()]
        public void DeleteTest()
        {
            //Delete the created field
            dynamic jsonResult = ((JsonResult)controller.Delete(ID.FieldId)).Data;
            Assert.AreEqual("success", jsonResult.status);

            //Delete a field that does not exist
            jsonResult = ((JsonResult)controller.Delete(999999)).Data;
            Assert.AreEqual("error", jsonResult.status);
        }
    }
}
```