```
using LuqTwoList_ToDoListApp.Controllers;
using LuqTwoList_ToDoListApp.Models;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Xunit;

namespace LuqTwoList_ToDoListApp.Unit_Tests
{
    public class TodosControllerTests
    {
        private static ToDoListContext Stworz_CreateDbContext(string
DbTesting_DbTestowanie)
        {
            var options_opcje = new DbContextOptionsBuilder<ToDoListContext>()
                .UseInMemoryDatabase(databaseName: DbTesting_DbTestowanie)
                .Options;
            return new ToDoListContext(options_opcje);
        }
        [Fact]
        public async Task GetTodos_ZwrocWszystkieZadania()
        {
            using var context_zawartosc = Stworz_CreateDbContext("GetTodosDb");
            var controller = new TodosController(context_zawartosc);

            context_zawartosc.TodosZadania.AddRange(
                new ToDoList
            {
                TitleTytul = "TestNoA_TestNrA",
                CompletedUkonczone = false
            },
                new ToDoList
            {
                TitleTytul = "TestNoB_TestNrB",
                CompletedUkonczone = true
            }
                );
            await context_zawartosc.SaveChangesAsync();

            var result_wynik = await controller.GetTodos();

            var actionResult_wynikDzialania =
Assert.IsType<ActionResult<IEnumerable<ToDoList>>>(result_wynik);
            var todos_zadania =
Assert.IsType<List<ToDoList>>(actionResult_wynikDzialania.Value);
            Assert.Equal(2, todos_zadania.Count);
        }
        [Fact]
        public async Task PostTodos_DodajKilkaZadan()
        {
            using var context_zawartosc = Stworz_CreateDbContext("PostTodoDb");
            var controller = new TodosController(context_zawartosc);

            // dodanie pierwszego zadania (todo)
            var newTodo1_noweZadanie1 = new ToDoList
            {
                Id = 7,
                TitleTytul = "TestNo1_TestNr1",
                CompletedUkonczone = false
            };

            var result1 = await controller.PostTodo(newTodo1_noweZadanie1);
```

```
            var createdAtActionResult1 =
Assert.IsType<CreatedAtActionResult>(result1.Result);
            var createdTodo1_stworzoneZadanie1 =
Assert.IsType<ToDoList>(createdAtActionResult1.Value);

            Assert.NotNull(createdTodo1_stworzoneZadanie1);
            Assert.Equal(newTodo1_noweZadanie1.TitleTytul,
createdTodo1_stworzoneZadanie1.TitleTytul);

            // dodanie drugiego zadania (todo)
            var newTodo2_noweZadanie2 = new ToDoList
            {
                Id = 12,
                TitleTytul = "TestNo2_TestNr2",
                CompletedUkonczone = false
            };

            var result2 = await controller.PostTodo(newTodo2_noweZadanie2);
            var createdAtActionResult2 =
Assert.IsType<CreatedAtActionResult>(result2.Result);
            var createdTodo2_stworzoneZadanie2 =
Assert.IsType<ToDoList>(createdAtActionResult2.Value);

            Assert.NotNull(createdTodo2_stworzoneZadanie2);
            Assert.Equal(newTodo2_noweZadanie2.TitleTytul,
createdTodo2_stworzoneZadanie2.TitleTytul);

            // sprawdzenie, czy oba zadania zostały dodane
            var allTodos_wszystkieZadania = await
context_zawartosc.TodosZadania.ToListAsync();
            Assert.Equal(2, allTodos_wszystkieZadania.Count);
        }
        [Fact]
        public async Task PutTodo_UpdatesTodo()
        {
            using var context_zawartosc = Stworz_CreateDbContext("PutTodoDb");
            var controller = new TodosController(context_zawartosc);

            var existingTodo_istniejaceZadanie = new ToDoList
            {
                Id = 7,
                TitleTytul = "TestChanged_TestZmieniony",
                CompletedUkonczone = false
            };
            context_zawartosc.TodosZadania.Add(existingTodo_istniejaceZadanie);
            await context_zawartosc.SaveChangesAsync();

            existingTodo_istniejaceZadanie.TitleTytul = "Updated Title";
            await controller.PutTodo(existingTodo_istniejaceZadanie.Id,
existingTodo_istniejaceZadanie);

            var updatedTodo_zaktualizowaneZadanie = await
context_zawartosc.TodosZadania.FindAsync(existingTodo_istniejaceZadanie.Id);
            Assert.Equal("Updated Title",
updatedTodo_zaktualizowaneZadanie.TitleTytul);
        }
        [Fact]
        public async Task DeleteTodo_RemovesTodo()
        {
            using var context_zawartosc = Stworz_CreateDbContext("DeleteTodoDb");
            var controller = new TodosController(context_zawartosc);

            var todoToDelete_ZadanieDoUsuniecia = new ToDoList
            {
                Id = 12,
```

```
                TitleTytul = "TestNo2_TestNr2",
                CompletedUkonczone = false
            };
            context_zawartosc.TodosZadania.Add(todoToDelete_ZadanieDoUsuniecia);
            await context_zawartosc.SaveChangesAsync();

            var result_wynik = await
controller.DeleteTodo(todoToDelete_ZadanieDoUsuniecia.Id);

            Assert.IsType<NoContentResult>(result_wynik);
            var todoExists_zadanieIstnieje = context_zawartosc.TodosZadania.Any(t =>
t.Id == todoToDelete_ZadanieDoUsuniecia.Id);
            Assert.False(todoExists_zadanieIstnieje);
        }
    }
}
```