```
using DatabaseConnect;
using DatabaseConnect.Entities;
using Microsoft.AspNetCore.Authorization;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using Microsoft.Extensions.Configuration;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;

namespace LibraryAppMVC.Controllers
{
    [Route("/simple/")] // All endpoints checked 2/25/18
    public class SimpleController : Controller
    {
        private Context _ctx;
        private IConfiguration _cfg;
        public SimpleController(Context context, IConfiguration config)
        {
            _ctx = context;
            _cfg = config;
        }

        [Route("books")]
        [AllowAnonymous]
        [HttpGet]
        public IActionResult GetABook(string title, int page = 1) // Checked 2/25/18 working
        {
            int PerPageCount = Int32.Parse(_cfg["PerPageCount"]);
            List<Book> a;
            if (title != null)  // Title specified
            {
                a = _ctx.Books
                    .Where(b => b.Title.Contains(title))
                    .Include(book => book.AuthorBooks)
                        .ThenInclude(ab => ab.Author)
                    .ToList();
            }
            else  // Title not specified
            {
                a = _ctx.Books
                            .Include(book => book.AuthorBooks)
                                .ThenInclude(ab => ab.Author)
                            .ToList();
                if (Boolean.Parse(_cfg["DoPages"]))
                {
                    if (page < 1) { page = 1; }
                    int pos_i = (page - 1) * 10;
                    int pos_f = page * 10;
                    int count = _ctx.Books.Count();
                    if (pos_f > count) { pos_f = count; }
                    if (pos_i > count)
                    {
                        a = new List<Book>();
                    }
                    else
                    {
                        a = a.GetRange(pos_i, (pos_f - pos_i));
                    }
                }
            }

            foreach (Book b in a)
            {
                List<String> AuthorList = new List<String>();
                foreach (AuthorBook ab in b.AuthorBooks)
                {
                    AuthorList.Add(ab.Author.Name);
                }
                b.Authors = AuthorList;
                b.AuthorBooks = null;
            }
            return Json(a);
        }

        [Route("checkouts")]
        [AllowAnonymous]
        [HttpGet]
        public IActionResult GetCheckouts() // Checked 2/25/18 working
        {
            var CheckoutList = _ctx.Checkouts
                .Include(c => c.Book)
                .Where(c => c.Active)
                .ToList();
            return Json(CheckoutList);
        }

        [Route("reservations")]
        [AllowAnonymous]
        [HttpGet]
        public IActionResult GetReservations() // Checked 2/25/18 working
        {
            var CheckoutList = _ctx.Reservations
                .Include(r => r.Book)
                .Where(r => r.Active)
                .ToList();
            return Json(CheckoutList);
        }
    }
}
```