```
using Meditech.Core.Auth.Authorization;
using Meditech.Core.Auth.Entities;
using Meditech.Core.Common.Services;
using Meditech.Core.Common.ViewModels;
using Microsoft.AspNetCore.Mvc;

namespace Meditech.App.CommonService.Controllers
{
    [MediAuthorize]
    [Route("api/[controller]")]
    [ApiController]
    public class ErpNotificationController : ControllerBase
    {
        private readonly ILogger<AttachmentController> _logger;
        private readonly IErpNotificationService _erpNotificationService;
        public ErpNotificationController(ILogger<AttachmentController> logger,
                               IErpNotificationService erpNotificationService)
        {
            _logger = logger;
            _erpNotificationService = erpNotificationService;
        }
        [HttpPost]
        [Route("search")]
        public IActionResult SearchMobile([FromBody] ErpNotificationQuerySearch model)
        {
            try
            {
                var requestUser = (BaseUser)Request.HttpContext.Items["User"];
                if (requestUser == null) return Forbid();
                var data = _erpNotificationService.SearchNotification(model, requestUser.Id);
                return Ok(new { Data = data });
            }
            catch (Exception ex)
            {
                _logger.LogError(ex, "SearchMobile exception");
                return StatusCode(StatusCodes.Status500InternalServerError);
            }
        }
        [HttpPost]
        [Route("get-all-noti")]
        public async Task<IActionResult> SearchNotiNoPaging([FromBody] ErpNotificationQuerySearchNoPaging model)
        {
            try
            {
                var requestUser = (BaseUser)Request.HttpContext.Items["User"];
                if (requestUser == null) return Forbid();
                var data = await _erpNotificationService.SearchNotificationNoPaging(model, requestUser.Id);
                return Ok(new { Data = data });
            }
            catch (Exception ex)
            {
                _logger.LogError(ex, "SearchNotiNoPaging exception");
                return StatusCode(StatusCodes.Status500InternalServerError);
            }
        }


        [HttpGet]
        [Route("read-all")]
        public IActionResult ReadAll()
        {
            try
            {
                var requestUser = (BaseUser)Request.HttpContext.Items["User"];
                if (requestUser == null) return Forbid();
                _erpNotificationService.ReadAll(requestUser.Id);
                return Ok();
            }
            catch (Exception ex)
            {
                _logger.LogError(ex, "ReadAll exception");
                return StatusCode(StatusCodes.Status500InternalServerError);
            }
        }

        [HttpGet]
        [Route("delete-all")]
        public IActionResult DeleteAll()
        {
            try
            {
                var requestUser = (BaseUser)Request.HttpContext.Items["User"];
                if (requestUser == null) return Forbid();
                _erpNotificationService.DeleteAllNo(requestUser.Id);
                return Ok();
            }
            catch (Exception ex)
            {
                _logger.LogError(ex, "ReadAll exception");
                return StatusCode(StatusCodes.Status500InternalServerError);
            }
        }

        [HttpGet]
        [Route("mark-read/{notiId}")]
        public IActionResult MarkRead(Guid notiId)
        {
            try
            {
                var requestUser = (BaseUser)Request.HttpContext.Items["User"];
                if (requestUser == null) return Forbid();
                _erpNotificationService.ReadNotification(notiId, requestUser.Id);
                return Ok();
            }
            catch (Exception ex)
            {
                _logger.LogError(ex, "MarkRead exception");
                return StatusCode(StatusCodes.Status500InternalServerError);
            }
        }

        [HttpDelete]
        [Route("delete-by-id")]
        public IActionResult DeleteNotiById(Guid id)
        {
            try
            {
                var requestUser = (BaseUser)Request.HttpContext.Items["User"];
                if (requestUser == null) return Forbid();
                var data = _erpNotificationService.DeleteNotiById(id);
                if (data.IsOk) return Ok();
                return StatusCode(600, data);
            }
            catch (Exception ex)
            {
                _logger.LogError(ex, "DeleteNotiById exception");
                return StatusCode(StatusCodes.Status500InternalServerError);
            }
        }
    }
}
```