# ДОДАТОК А

## Код програми

Котролер “EmailsController”

```
using EmailReceiverAPI.Services.Interfaces;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using System.Collections;
using System.Collections.Generic;
using System.Threading.Tasks;
using TemperatureReceiverAPI.Services.Response;
using TemperatureReceiverAPI.Shared.Models;

namespace TemperatureReceiverAPI.Controllers
{
   [Route("api/[controller]")]
   [ApiController]
   public class EmailsController : ControllerBase
   {
      private readonly IEmailService service;

      public EmailsController(IEmailService service)
      {
         this.service = service;
      }

      [HttpGet]
      Public async Task<ActionResult<APIResponse<IEnumerable<EmailModel>>>>
GetEmails([FromQuery] int count = 5, [FromQuery] int page = 1)
      {
         return Ok(await service.GetEmails(count, page));
```

```
        }

        [HttpPost]
        public async Task<ActionResult<APIResponse<EmailModel>>>
CreateEmail([FromBody] EmailModel model)
        {
            return Ok(await service.AddEmail(model));
        }

        [HttpGet("{id}")]
        public async Task<ActionResult<APIResponse<EmailModel>>>
GetEmail(string id)
        {
            return Ok(await service.GetEmail(id));
        }

        [HttpDelete("{id}")]
        public async Task<ActionResult<APIResponse<EmailModel>>>
DeleteEmail(string id)
        {
            return Ok(await service.DeleteEmail(id));
        }
    }
}
```

Контролер “MessageSenderController”

```
using EmailReceiverAPI.Services.Interfaces;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
```

```
using Microsoft.Extensions.Configuration;
using System.Linq;
using System.Threading.Tasks;
using TemperatureReceiverAPI.MessageSenderAPI.Implementations;
using TemperatureReceiverAPI.Senders;
using TemperatureReceiverAPI.Services.Implementations;
using TemperatureReceiverAPI.Services.Response;
using TemperatureReceiverAPI.Shared.Models;

namespace TemperatureReceiverAPI.Controllers
{
   [Route("api/[controller]")]
   [ApiController]
   public class MessageSenderController : ControllerBase
   {
      private readonly IEmailService emailService;
      private readonly IConfiguration config;
      private readonly MessageSender messageSender;

      public MessageSenderController(IEmailService emailService,
IConfiguration config, MessageSender messageSender)
      {
         this.emailService = emailService;
         this.config = config;
         this.messageSender = messageSender;
      }

      [HttpPost]
      public async Task<ActionResult<APIResponse<TemperatureModel>>>
SendAlertMessage([FromBody] TemperatureModel model)
```

```
        {
            if (!ModelState.IsValid)
            {
                return BadRequest(new APIResponse<TemperatureModel>(ModelState.Values.SelectMany(x => x.Errors).Select(x => x.ErrorMessage)));
            }

            if (model.TempValue >= model.CriticalTempValue)
            {
                await messageSender.SendMessageViaTelegramAsync("Temperature alert!", $"Current temperature '{model.TempValue} {model.TempType}' exceeds the critical value '{model.CriticalTempValue} {model.TempType}'\n\nDetected at: {model.DetectedAt.ToString("yyyy-MM-dd hh:mm:ss")}", config["SenderTelegram:ChannelName"]);

                var emailsResult = await emailService.GetEmails(0, 0);

                if (emailsResult.IsSuccess)
                {
                    foreach (var email in emailsResult.Data)
                    {
                        await messageSender.SendMessageViaEmailAsync("Temperature alert!", $"Current temperature '{model.TempValue} {model.TempType}' exceeds the critical value '{model.CriticalTempValue} {model.TempType}'\n\nDetected at: {model.DetectedAt.ToString("yyyy-MM-dd hh:mm:ss")}", email.Address);
                    }
                }

                return Ok(new APIResponse<TemperatureModel>(model));
```

```
            }

            return Ok();
        }
    }
}
```

Контролер “TemperaturesController”

```
using TemperatureReceiverAPI.MessageSenderAPI.Implementations;
using TemperatureReceiverAPI.MessageSenderAPI.Interfaces;
using TemperatureReceiverAPI.Services.Interfaces;
using TemperatureReceiverAPI.Services.Response;
using TemperatureReceiverAPI.Shared.Models;

namespace TemperatureReceiverAPI.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class TemperaturesController : ControllerBase
    {
        private readonly ITemperatureService tempService;

        public TemperaturesController(ITemperatureService tempService)
        {
            this.tempService = tempService;
        }

        [HttpPost]
```

```
        public async Task<ActionResult<APIResponse<TemperatureModel>>>
AddTemperature([FromBody]TemperatureModel model)
        {
            if(!ModelState.IsValid)
            {
                return Ok(new
APIResponse<TemperatureModel>(ModelState.Values.SelectMany(x =>
x.Errors).Select(x => x.ErrorMessage)));
            }

            return Ok(await tempService.AddTemperature(model));
        }

        [HttpDelete("{id}")]
        public async Task<ActionResult<APIResponse<TemperatureModel>>>
DeleteTemperature(string id)
        {
            return Ok(await tempService.DeleteTemperature(id));
        }

        [HttpGet]
        public async
Task<ActionResult<APIResponse<IEnumerable<TemperatureModel>>>>
GetTemperatures([FromQuery] int count = 5, [FromQuery] int page = 1)
        {
            return Ok(await tempService.GetTemperatures(count, page));
        }

        [HttpGet("{id}")]
```

```
        public async Task<ActionResult<APIResponse<TemperatureModel>>>
GetTemperature(string id)
        {
            return Ok(await tempService.GetTemperature(id));
        }
    }
}
```

## Додаток Б. Демонстраційні графічні матеріали

| Формат | Позиція. | ПОЗНАЧЕННЯ | НАЙМЕНУВАННЯ | Кільк. | Примітки |
|---|---|---|---|---|---|
| | | | Текстові документи | | |
| А4 | 1 | ГЮІК 010115.013 ПЗ | Пояснювальна записка | 51 | с |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | Графічні документи | | |
| | 2 | | Демонстраційний матеріал у | 8 | формат |
| | | | вигляді презентації | | *.ppt |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | | | | | ГЮІК 010115.013 ВД |
| Змін. | Арк. | Номер докум. | Підпис | Дата | |
| Розроб. | | Мороз | | | Розробка системи автоматизації для розсилки email повідомлень по результатам аналізу подій з автоматизованою розсилкою в месенджери |
| Перевір. | | Іванов | | | |
| | | | | | |
| Н.контр. | | Бронніков | | | |
| Затв. | | Невлюдов | | | |

| Літера | Арку | Аркуш |
|---|---|---|
| Н | 1 | 1 |

Кафедра КІТАР

ХНУРЕ