**МIНIСТЕРСТВО ОСВIТИ I НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ “ОСТРОЗЬКА АКАДЕМІЯ”**

**Кафедра економіко-математичного моделювання та інформаційних технологій**

**ЗВІТ**

**Лабораторна робота №2**

**Основи роботи з ASP.NET Core MVC**

здобувачів вищої освіти

першого (бакалаврського) рівня

третього року навчання групи КН-21

спеціальності 122 Комп’ютерні науки

ОПП «Комп’ютерні науки»

Лайтер Ярини Семенівни

Колоди Станіслава Геннадійовича

(прізвище, ім’я, по батькові)

База лабораторної - кафедра економіко-математичного моделювання  
 та інформаційних технологій

Керівник лабораторної від університету

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(науковий ступінь, учене звання керівника)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Власне ім’я та ПРІЗВИЩЕ)

Члени комісії:

\_\_\_\_\_ Юрій КЛЕБАН

Острог - 2023

[**1. Опис завдання 3**](#_i48fsqfd073o)

[**3. Опис класів у вигляді коду, з коментарями 4**](#_vcjtkjfrr57x)

[Код інтерфейсу IEmailSender: 4](#_kw6bw3w34ok9)

[Код класу IEmailSender: 4](#_4xmajjhbm2nm)

[Частина коду класу ResponceController: 5](#_bojc90zbellz)

[Частина коду класу AssignmentController: 5](#_8yn8fmmb4kuv)

[Частина коду класу ComplaintController(customer): 6](#_rxgbicv4o1uc)

[**4. Опис графічного інтерфейсу програми: 8**](#_2bsvyd5t71go)

[**5. Висновок 9**](#_s7w9sp56kjo9)

[**6. Список використаних джерел: 10**](#_1rdve09bb2z5)

**Inspector**

**Мета:** Метою лабораторної роботи є набуття вмінь з інтеграція дизайну, побудови архітектури рішення на [ASP.NET](http://asp.net/) Core, а також закріплення навичок колективної роботи.

# 1. **Опис завдання**

*Основне завдання лабораторної роботи №1: реалізувати CRUD для усіх сутностей, додати управління користувачами, інтегрувати дизайн, реалізувати трирівневу архітектуру (або складнішу).*

* Продовжуємо працювати у тих же репозиторіях.
* Кожну задачу виконує визначений учасник у окремій гілці, Team Lead робить merge у main.
* Обрати шаблон для інтеграції у проєкт.
* Інтегрувати дизайну проєкт.
* Реалізувати CRUD-операції для усіх сутностей.
* Додати можливість управління користувачами, розділи функції за ролями + вхід/реєстрація.
* Реалізувати як мінімум трирівневу архітектуру з організацією коду у різних збірках.
* Виконати додаткове завдання до Лаб 2. Щоб знайти додаткове завдання потрібно натиснути OPEN на назву проєкту. (Додати сповіщення на елетронну пошту про відповідь на скаргу або про перенаправлення скарги іншому виконавцю.)
* Підготовка звіту.
* Публічний захист.

# 

# 

# 

# 

# 

# 

# 3**. Опис класів у вигляді коду, з коментарями**

## **Код інтерфейсу IEmailSender:**

**public** **interface** IEmailSender

{

Task SendEmailAsync(string email, string? subject, string? htmlMessage, int? orgID);

}

Лістинг 1.1. Код програми

**Короткий опис коду:**

Цей код визначає інтерфейс IEmailSender, який містить один метод SendEmailAsync. Метод призначений для асинхронної відправки електронних листів

## **Код класу IEmailSender:**

**public** **class** **EmailSender** : IEmailSender

{

**public** Task SendEmailAsync(string email, string? subject, string? htmlMessage, int? orgID)

{

var mail = "inspectorapp2023@gmail.com";

subject = "Response from inspector.web";

**if** (orgID != **null**)

{

htmlMessage = "You have received a new task from your company!" +

"\r\nGo check the web site to see more." +

"\r\n\r\nRegards," +

"\r\nThe Inspector team";

}

**else**

{

htmlMessage = "You have received a response for your complaint!" +

"\r\nGo check the web site to see more." +

"\r\n\r\nRegards," +

"\r\nThe Inspector team";

}

var client = **new** SmtpClient("smtp.outlook.com", 587)

{

EnableSsl = **true**,

UseDefaultCredentials = **false**,

Credentials = **new** NetworkCredential(mail, "Inspector123!")

};

**return** client.SendMailAsync(

**new** MailMessage(**from**: mail,

to: email,

subject,

htmlMessage));

}

}

Лістинг 1.2. Код програми

**Короткий опис коду:**

Цей код описує клас, який відповідає за відправку листів користувачам. Листи надходять від корпоративної пошти нашої програми.

## **Частина коду класу ResponceController:**

….

[HttpPost]

**public** IActionResult Create(Responce responce, IFormFile? file)

{

**if** (ModelState.IsValid)

{

\_responceRepo.Add(responce);

\_responceRepo.Save();

*//sending email for the user when the response is given*

var user = \_userRepo.Get(u => u.Id == responce.UserTakeId);

SendResponseEmail(user.Email, user.OrganizationId);

TempData["success"] = "Responce and email sent successfully";

Complaint obj = \_complaintRepo.Get(u => u.Id == responce.ComplaintId);

obj.ResponceId = responce.Id;

obj.Status = "done";

\_complaintRepo.Save();

**return** RedirectToAction("Index", "Complaint");

}

**return** View(responce);

}

**public** **async** Task SendResponseEmail(string email, int? orgID)

{

**await** \_emailSender.SendEmailAsync(email, **null**, **null**, orgID);

}

….

Лістинг 1.3. Код програми

**Короткий опис коду:**

Цей код надає можливість відправляти листи на пошту та сповіщати користувача, що його скарга була розглянута та виконана (за допомогою асинхронної функції).

## **Частина коду класу AssignmentController:**

[HttpPost]

**public** IActionResult Create(AssignmentVM assignmentVM)

{

**if** (ModelState.IsValid)

{

\_assignmentRepo.Add(assignmentVM.Assignment);

var user = \_userRepo.Get(u => u.Id == assignmentVM.Assignment.UserTakeId);

SendResponseEmail(user.Email, user.OrganizationId);

TempData["success"] = "Assignment created successfully";

\_assignmentRepo.Save();

Complaint complaint = \_complaintRepo.Get(u => u.Id == assignmentVM.Assignment.ComplaintId);

\_complaintRepo.Get(u => u.Id == assignmentVM.Assignment.ComplaintId).Status = "in process";

\_complaintRepo.Save();

**return** RedirectToAction("Index", "Complaint");

}

**return** View();

}

**public** **async** Task SendResponseEmail(string email, int? orgID)

{

**await** \_emailSender.SendEmailAsync(email, **null**, **null**, orgID);

}

Лістинг 1.4. Код програми

**Короткий опис коду:**

Цей код надає можливість відправляти листи на пошту робітникам, яким доручили виконувати роботу\скаргу (за допомогою асинхронної функції).

## **Частина коду класу ComplaintController(customer):**

[HttpPost]

**public** IActionResult Upsert(ComplaintVM complaintVM, List<IFormFile>? files)

{

**if** (ModelState.IsValid)

{

string wwwRootPath = \_webHostEnvironment.WebRootPath;

**if** (complaintVM.Complaint.Id == 0)

{

complaintVM.Complaint.UserId = User.FindFirstValue(ClaimTypes.NameIdentifier);

\_complaintRepo.Add(complaintVM.Complaint);

TempData["success"] = "Complaint created successfully";

}

**else**

{

complaintVM.Complaint.UserId = User.FindFirstValue(ClaimTypes.NameIdentifier);

\_complaintRepo.Update(complaintVM.Complaint);

*//видалення файлів які були збережені на цю скаргу раніше*

var complaintFiles = \_complaintFileRepo.GetAll().Where(cf => cf.ComplaintId == complaintVM.Complaint.Id).ToList();

**foreach** (var compFile **in** complaintFiles)

{

**if** (!string.IsNullOrEmpty(compFile.FilePath))

{

var oldImagePath = Path.Combine(wwwRootPath, compFile.FilePath.TrimStart('/'));

**if** (System.IO.File.Exists(oldImagePath))

{

System.IO.File.Delete(oldImagePath);

\_complaintFileRepo.Remove(compFile);

\_complaintFileRepo.Save();

}

}

}

TempData["success"] = "Complaint update successfully";

}

\_complaintRepo.Save();

**if** (files != **null** && files.Count > 0)

{

**foreach** (var file **in** files)

{

**if** (file != **null** && file.Length > 0)

{

string fileName = Guid.NewGuid().ToString() + Path.GetExtension(file.FileName);

string filePath = Path.Combine(wwwRootPath, @"files");

**using** (var fileStream = **new** FileStream(Path.Combine(filePath, fileName), FileMode.Create))

{

file.CopyTo(fileStream);

}

var complaintFile = **new** ComplaintFile

{

FileName = fileName,

FilePath = "/files/" + fileName,

ComplaintId = complaintVM.Complaint.Id

};

\_complaintFileRepo.Add(complaintFile);

\_complaintFileRepo.Save();

}

}

}

**return** RedirectToAction("Index");

}

**else**

{

complaintVM.OrganizationList = \_organizationRepo.GetAll().Select(u => **new** SelectListItem

{

Text = u.Name,

Value = u.Id.ToString()

});

**return** View(complaintVM);

}

}

Лістинг 1.5. Код програми

**Короткий опис коду:**

Цей код надає можливість користувачу завантажувати декілька файлів у свою заявку. Якщо користувач вирішить редагувати заявку, всі попередні файли будуть видалені, а підкріплені до цієї заявки будуть тільки нові файли.

# **4. Опис графічного інтерфейсу програми:**

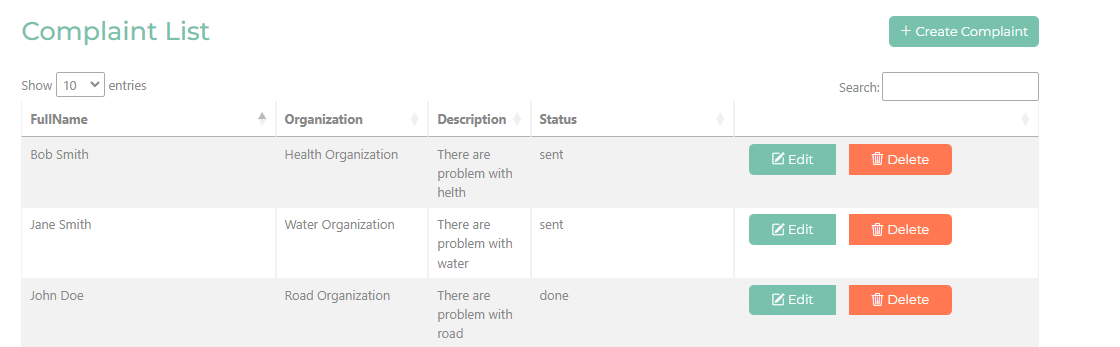


Рис 1.1. Інтерфейс програми

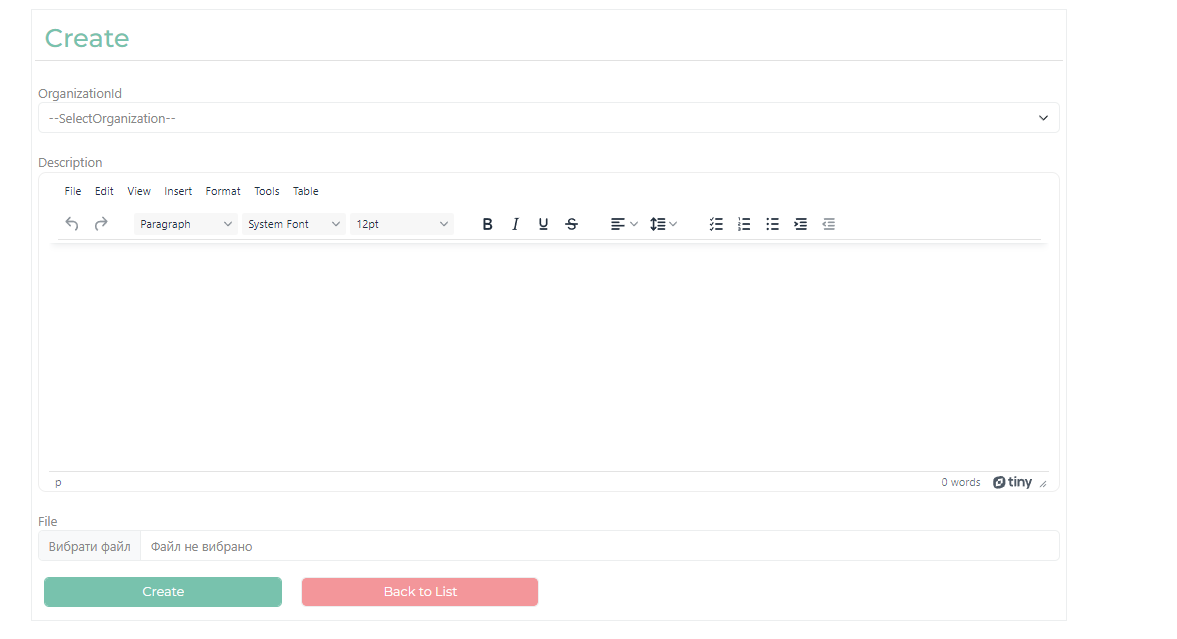


Рис 1.2. Інтерфейс програми

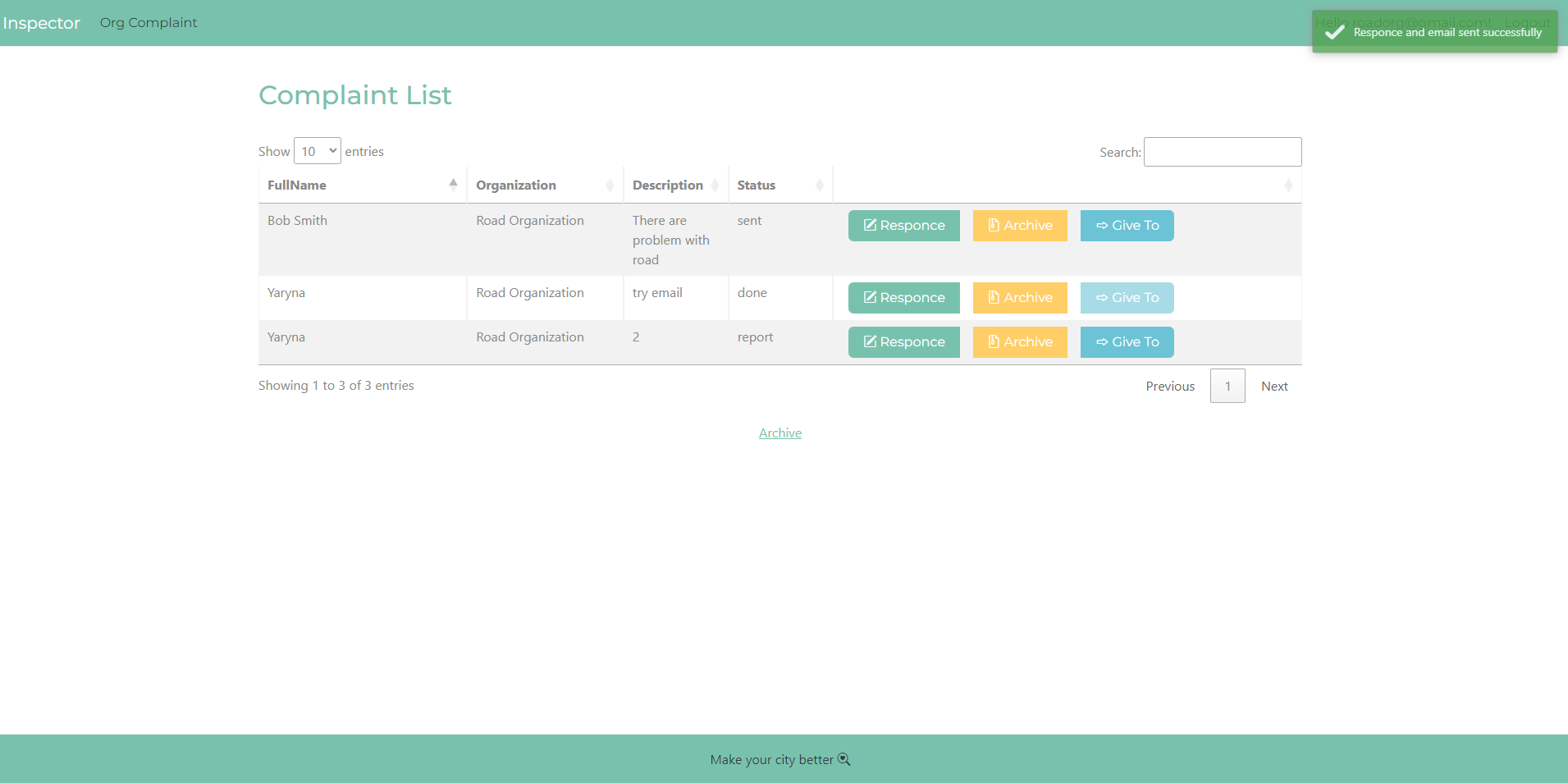


Рис 1.3. Інтерфейс програми



Рис 1.4. Інтерфейс програми

# 

# **5. Висновок**

На цій лабораторній роботі ми успішно інтегрувати дизайн, побудували багаторівневу архітектуру на платформі ASP.NET Core та реалізувати CRUD-операції для сутностей.

Ми також додали управління користувачами з ролями, покращили зовнішній вигляд і вирішили додаткове завдання з електронною поштою.

Все це допомогло нам розвинути навички розробки веб-додатків, вивчити ASP.NET Core та поліпшити колективну роботу.

# **6. Список використаних джерел:**

1. Звіт №1 - [Звіт\_Лаб1\_Лайтер\_Колода](https://docs.google.com/document/d/1_DGs-YYdBlzHYTO4_DUqzgC_ACzAjRJO_jWIruxo-mE/edit)
2. Репозиторій на github - <https://github.com/oa-ek/lab-1-inspector>
3. C# Guide - <https://docs.microsoft.com/en-us/dotnet/csharp/>.
4. C# Station - <https://csharp-station.com/>.
5. Working with C# - <https://code.visualstudio.com/docs/languages/csharp>.
6. Основи програмування на C# (Prometheus) , <https://edx.prometheus.org.ua/courses/Microsoft/CS201/2016_T1/info>
7. Introduction to C# (Edx), <https://www.edx.org/course/introduction-to-c>
8. DotNetMastery - <https://dotnetmastery.com/>