НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

**«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»**

Кафедра автоматизованих систем обробки інформації і управління

**КУРСОВА РОБОТА**

**“Системи баз даних”**

Виконала: *Стрільцова А.О.*

Група: *ІС-62*

Варіант 19

Керівник: *доцент Попенко В.Д.*

Київ 2018

Завдання

на курсову роботу по дисципліні

**«Системи баз даних»**

фахове спрямування – "Інформаційні управляючі системи та технології" напрям 6.050101 "Комп'ютерні науки"

Група ІС-62

(П.І.Б.) Стрільцова Аліна Олегівна

Тема Курсової роботи:

*Тестування програмного продукту.*



Дата представлення роботи до захисту до « » 20 р.

|  |  |  |
| --- | --- | --- |
|  | Завдання видано: |  |
|  | Дата: | « » 20 р. |
|  | Керівник: | (В. Д. Попенко) |

Студент: ( )

# Зміст

[1. Зміст 3](file:///C:\Users\Alina\Downloads\Telegram%20Desktop\Курсова_БД.docx#_Toc533597801)

[2. Вступ 4](file:///C:\Users\Alina\Downloads\Telegram%20Desktop\Курсова_БД.docx#_Toc533597802)

[3. Розроблення ER-діаграми для предметної області 5](file:///C:\Users\Alina\Downloads\Telegram%20Desktop\Курсова_БД.docx#_Toc533597803)

[· Предметна область 5](file:///C:\Users\Alina\Downloads\Telegram%20Desktop\Курсова_БД.docx#_Toc533597804)

[· ER-діаграма 6](file:///C:\Users\Alina\Downloads\Telegram%20Desktop\Курсова_БД.docx#_Toc533597805)

[· Модель бази у 3НФ 7](file:///C:\Users\Alina\Downloads\Telegram%20Desktop\Курсова_БД.docx#_Toc533597806)

[4. Вибір та обґрунтування вибору бази даних для задачі 7](file:///C:\Users\Alina\Downloads\Telegram%20Desktop\Курсова_БД.docx#_Toc533597807)

[5. Створення бази даних і наповнення її тестовими даними 8](file:///C:\Users\Alina\Downloads\Telegram%20Desktop\Курсова_БД.docx#_Toc533597808)

[· Запити для створення бази даних: 8](file:///C:\Users\Alina\Downloads\Telegram%20Desktop\Курсова_БД.docx#_Toc533597809)

[· Заповнення таблиці даними: 10](file:///C:\Users\Alina\Downloads\Telegram%20Desktop\Курсова_БД.docx#_Toc533597810)

[6. Розроблення запитів до бази даних 11](file:///C:\Users\Alina\Downloads\Telegram%20Desktop\Курсова_БД.docx#_Toc533597811)

[· Запити від двох або більше таблиць 11](file:///C:\Users\Alina\Downloads\Telegram%20Desktop\Курсова_БД.docx#_Toc533597812)

[· Групуючі запити та аналітичні запити 12](file:///C:\Users\Alina\Downloads\Telegram%20Desktop\Курсова_БД.docx#_Toc533597813)

[· Запити на вставку, оновлення, видалення значень 13](file:///C:\Users\Alina\Downloads\Telegram%20Desktop\Курсова_БД.docx#_Toc533597814)

[7. Розроблення необхідних форм введення інформації 14](file:///C:\Users\Alina\Downloads\Telegram%20Desktop\Курсова_БД.docx#_Toc533597815)

[8. Розроблення форм необхідних звітів для виводу даних 16](file:///C:\Users\Alina\Downloads\Telegram%20Desktop\Курсова_БД.docx#_Toc533597816)

[9. Висновки 19](file:///C:\Users\Alina\Downloads\Telegram%20Desktop\Курсова_БД.docx#_Toc533597817)

[10. Список використаної літератури 20](file:///C:\Users\Alina\Downloads\Telegram%20Desktop\Курсова_БД.docx#_Toc533597818)

# Вступ

**Актуальність теми** полягає в тому, що сучасне суспільство неможливо уявити без використання інформаційних технологій і тому людина намагається як-найбільше використовувати їх в кожній сфері суспільства. Тестування програмного продукту не є виключенням, так як автоматизація процесу, а самое баг-трекінгові системи дають змогу як слідкувати у більш зручному вигляді за дефектами і вимогами, так і аналізувати які проблеми потрібно вирішувати якнайшвидше.

**Мета роботи**: поглиблення і закріплення теоретичних знань з дисципліни «Системи баз даних», набуття вмінь аналізувати опрацьований матеріал, робити відповідні узагальнення та висновки.

**Завдання роботи:**

* Розроблення ER-діаграми для своєї предметної області.
* Створення бази даних на основі ER-діаграми.
* Написання запитів на мові SQL.
* Розроблення додатку для роботи з базою даних.
* Розроблення форм необхідних звітів для виводу даних.

# Розроблення ER-діаграми для предметної області

## Предметна область

У процесі супроводу системи класу ERP у користувачів різних клієнтів виникають зауваження до системи, які можуть мати характер повідомлення про помилку (зауваження) або побажання (вимоги) додаткової функціональності (доробка).

Технічний директор фірми-розробника розглядає кожне зауваження і направляє його одному з тестувальників. Тестувальник перевіряє наявність проблеми, викладеної в зауваженні, і направляє зауваження одному з програмістів. Програміст виправляє зауваження і повідомляє про це тестувальника. Тестувальник перевіряє усунення проблеми і, якщо це так, повідомляє всіх про це і включає виправлений код у поточну версію; в іншому разі надсилає нове зауваження програмісту.

В разі необхідності доробки технічний директор формулює постановку задачі і після обговорення її з одним з програмістів направляє її програмісту, а також одному з тестувальників. Далі все відбувається як у попередньому процесі: по завершенню розробки тестувальник перевіряє нову функціональність і повідомляє про неї.

Раз на місяць технічний директор створює наступну версію продукту і повідомляє клієнтів про зміни в ньому у формі звіту по виправлених зауваженнях і виконаних доробках.

## D:\KPI\БД\КУРСАЧ\ER_kursach.pngER-діаграма

Рисунок 1 - ER-діаграма

## Модель бази у 3НФ

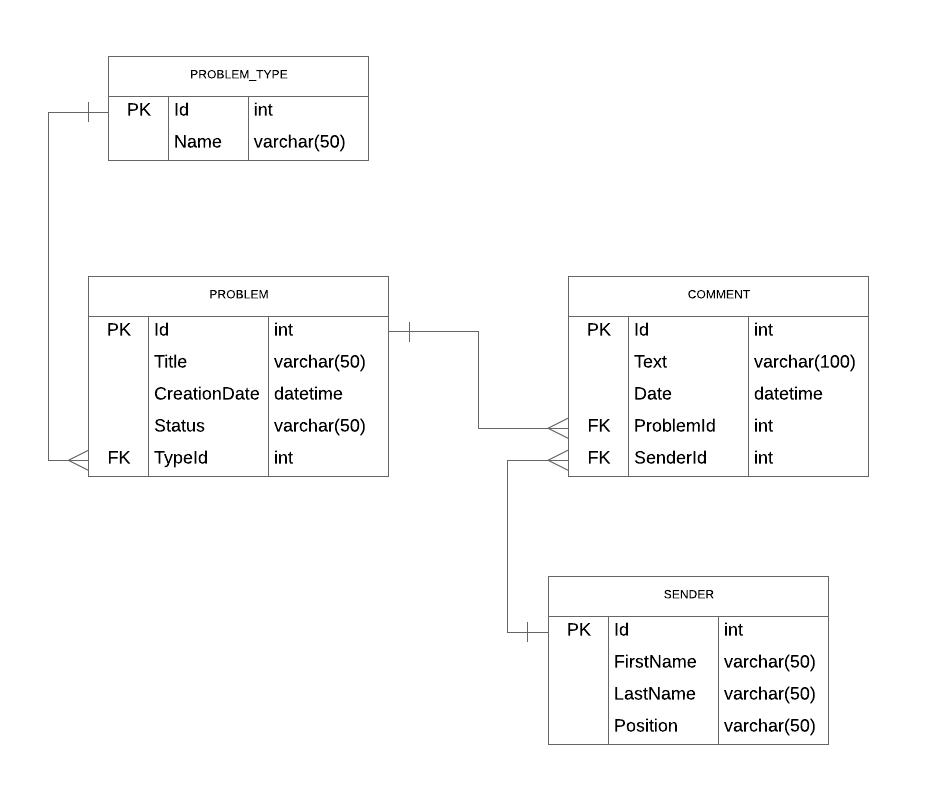


Рисунок 2 – Модель бази у 3НФ

# Вибір та обґрунтування вибору бази даних для задачі

Для даної роботи мною була вибрана система керування реляційними базами даних MS SQL. Особливість цієї СУБД є використання мови для запитів Transact-SQL. Оскільки цей стандарт мови SQL мені знайомий, це і послугувало для того, щоб обрати MSSQL. Також було обрано MSSQL базу даних для того, щоб можна було зручно зберігати дані у веб-застосунку за обраної теми.

# Створення бази даних і наповнення її тестовими даними

## Запити для створення бази даних:

CREATE TABLE [COMMENT] (

[Id] int NOT NULL IDENTITY(1,1),

[Text] varchar(100),

[Date] datetime,

[ProblemId] int,

[SenderId] int,

PRIMARY KEY ([Id]),

FOREIGN KEY(ProblemId) REFERENCES PROBLEM(Id),

FOREIGN KEY(SenderId) REFERENCES SENDER(Id)

);

CREATE TABLE [PROBLEM] (

[Id] int NOT NULL IDENTITY(1,1),

[Title] varchar(50),

[CreationDate] datetime,

[Status] varchar(50),

[TypeId] int,

PRIMARY KEY ([Id]),

FOREIGN KEY(TypeId) REFERENCES PROBLEM\_TYPE(Id)

);

CREATE TABLE [SENDER] (

[Id] int NOT NULL IDENTITY(1,1),

[FirstName] varchar(50),

[LastName] varchar(50),

[Position] varchar(50),

PRIMARY KEY ([Id])

);

CREATE TABLE [PROBLEM\_TYPE] (

[Id] int NOT NULL IDENTITY(1,1),

[Name] varchar(50),

PRIMARY KEY ([Id])

);

ALTER TABLE [dbo].[COMMENT] ADD CONSTRAINT [df\_Date] DEFAULT (getdate()) FOR [Date]

ALTER TABLE [dbo].[PROBLEM] ADD CONSTRAINT [df\_CreationDate] DEFAULT (getdate()) FOR [CreationDate]

## Заповнення таблиці даними:

INSERT INTO Car(Car\_ID,Employee\_ID,Type\_ID) VALUES (1,2,1);

INSERT INTO Car(Car\_ID,Employee\_ID,Type\_ID) VALUES (2,1,1);

INSERT INTO Command(Command\_ID,Crop\_ID,Date,Sum) VALUES (1,1,'2018-10-04',100);

INSERT INTO Command(Command\_ID,Crop\_ID,Date,Sum) VALUES (2,3,'2018-10-04',345);

INSERT INTO Command(Command\_ID,Crop\_ID,Date,Sum) VALUES (3,2,'2018-10-04',1000);

INSERT INTO Command(Command\_ID,Crop\_ID,Date,Sum) VALUES (4,5,'2018-10-10',2345);

INSERT INTO Command(Command\_ID,Crop\_ID,Date,Sum) VALUES (5,5,'2018-10-05',1500);

INSERT INTO Command(Command\_ID,Crop\_ID,Date,Sum) VALUES (6,7,'2018-10-10',1700);

INSERT INTO Crop(Crop\_ID,Name) VALUES (1,'Кукурудза');

INSERT INTO Crop(Crop\_ID,Name) VALUES (2,'Пшениця');

INSERT INTO Crop(Crop\_ID,Name) VALUES (3,'Овес');

INSERT INTO Crop(Crop\_ID,Name) VALUES (4,'Рапс');

INSERT INTO Crop(Crop\_ID,Name) VALUES (5,'Соняшник');

INSERT INTO Crop(Crop\_ID,Name) VALUES (6,'Просо');

INSERT INTO Crop(Crop\_ID,Name) VALUES (7,'Гречка');

INSERT INTO Employee(Employee\_ID,FullName,Passport,BirthDate,Specialization,Salary,Command\_ID) VALUES (1,'Джон Сібірський','AF1234','1978-10-04 00:00:00.000','Адміністратор',11119,1);

INSERT INTO Employee(Employee\_ID,FullName,Passport,BirthDate,Specialization,Salary,Command\_ID) VALUES (2,'Антон Кових','PA3456','1976-06-04 00:00:00.000','Водій трактору',1999,2);

INSERT INTO Employee(Employee\_ID,FullName,Passport,BirthDate,Specialization,Salary,Command\_ID) VALUES (3,'Петро Іванов','KU1234','1973-06-04 00:00:00.000','Сіяч',3000,2);

INSERT INTO Employee(Employee\_ID,FullName,Passport,BirthDate,Specialization,Salary,Command\_ID) VALUES (4,'Олександр Петров','AS1234','1975-06-04 00:00:00.000','Адміністратор',11200,1);

INSERT INTO Invoice(Invoice\_Id,Crop\_ID,Date,Sum) VALUES (1,1,'2018-10-04',10000);

INSERT INTO Invoice(Invoice\_Id,Crop\_ID,Date,Sum) VALUES (2,1,'2018-10-04',10300);

INSERT INTO Invoice(Invoice\_Id,Crop\_ID,Date,Sum) VALUES (3,2,'2018-10-04',15000);

INSERT INTO Invoice(Invoice\_Id,Crop\_ID,Date,Sum) VALUES (4,3,'2018-10-10',10000);

INSERT INTO Invoice(Invoice\_Id,Crop\_ID,Date,Sum) VALUES (5,4,'2018-10-10',13000);

INSERT INTO Invoice(Invoice\_Id,Crop\_ID,Date,Sum) VALUES (6,5,'2018-10-10',13560);

INSERT INTO Invoice(Invoice\_Id,Crop\_ID,Date,Sum) VALUES (7,6,'2018-10-10',13500);

INSERT INTO Requirement(Requirement\_ID,Date,Sum,Car\_ID) VALUES (1,'2018-10-04',120,1);

INSERT INTO Requirement(Requirement\_ID,Date,Sum,Car\_ID) VALUES (2,'2018-10-04',123,1);

INSERT INTO Requirement(Requirement\_ID,Date,Sum,Car\_ID) VALUES (3,'2018-10-04',145,2);

INSERT INTO Requirement(Requirement\_ID,Date,Sum,Car\_ID) VALUES (4,'2018-10-04',150,2);

INSERT INTO CarType(Type\_ID,Type) VALUES (1,’Трактор’);

INSERT INTO CarType(Type\_ID,Type) VALUES (2,’Комбайн’);

INSERT INTO Roadmap(Roadmap\_ID,Scope,Date,Sum,Crop\_ID,Car\_ID) VALUES (1,50,'2018-10-04',123,1,1);

INSERT INTO Roadmap(Roadmap\_ID,Scope,Date,Sum,Crop\_ID,Car\_ID) VALUES (2,53,'2018-10-04',140,1,2);

INSERT INTO Roadmap(Roadmap\_ID,Scope,Date,Sum,Crop\_ID,Car\_ID) VALUES (3,54,'2018-10-04',155,2,2);

INSERT INTO Roadmap(Roadmap\_ID,Scope,Date,Sum,Crop\_ID,Car\_ID) VALUES (4,50,'2018-10-04',100,2,2);

INSERT INTO Roadmap(Roadmap\_ID,Scope,Date,Sum,Crop\_ID,Car\_ID) VALUES (5,70,'2018-10-10',200,4,2);

INSERT INTO Roadmap(Roadmap\_ID,Scope,Date,Sum,Crop\_ID,Car\_ID) VALUES (6,75,'2018-10-10',300,5,1);

# Розроблення запитів до бази даних

## Запити від двох або більше таблиць

**Запит 1:**

запит на отримання інформації про зауваження по типам зауважень:

SELECT PROBLEM.Title,

CreationDate,

[Status],

PROBLEM\_TYPE.Name

FROM

PROBLEM\_TYPE

JOIN PROBLEM on PROBLEM\_TYPE.Id = PROBLEM.TypeId

where PROBLEM\_TYPE.Id = 1

**Результат**:

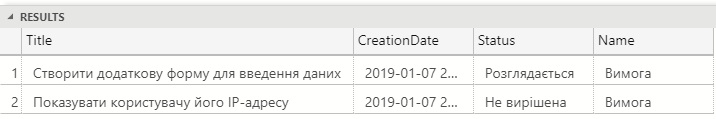


Рисунок 3 – Результат запиту 1

**Запит 2:**

запит на отримання імені відправника по номеру коментаря.

SELECT SENDER.\*

FROM COMMENT

RIGHT JOIN SENDER on SENDER.Id = COMMENT.SenderId

where COMMENT.Id=4

**Результат**:

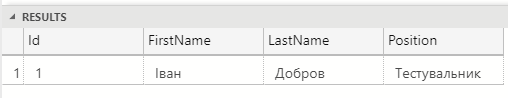


Рисунок 4 – Результат запиту 2

## Групуючі запити та аналітичні запити

**Запит 3:**

запит на отримання кількості повідомлень(коментарів) залишених відправником.

SELECT (FirstName +' '+ LastName) as FullName,

Position,

Count(COMMENT.Id) as CommentCount

FROM

SENDER

LEFT JOIN COMMENT on COMMENT.SenderId = SENDER.Id

GROUP BY (FirstName +' '+ LastName),

Position

ORDER BY Count DESC

**Результат**:

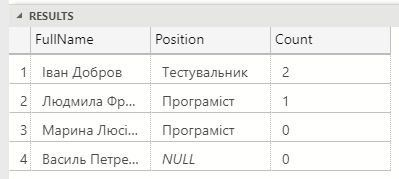


Рисунок 5 – Результат запиту 3

**Запит 4:**

запит на отримання кількості зауважень(проблем) по типам проблем.

SELECT PROBLEM\_TYPE.Name,

COUNT(PROBLEM.Id) as Count

FROM PROBLEM

RIGHT JOIN PROBLEM\_TYPE on PROBLEM\_TYPE.Id = PROBLEM.TypeId

GROUP BY PROBLEM\_TYPE.Name

ORDER BY Count DESC

**Результат**:

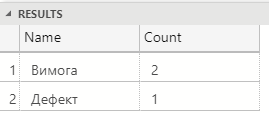
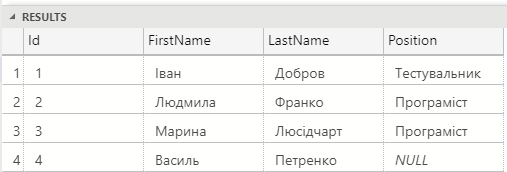


Рисунок 6 – Результат запиту 4

## Запити на вставку, оновлення, видалення значень

**Запит 5 (на вставку):**

INSERT INTo SENDER (FirstName,LastName,Position)

 VALUES('Олена','Петрівна','Тестувальник')

**Результат**:

Рисунок 7 – Таблиця SENDER до запиту 5

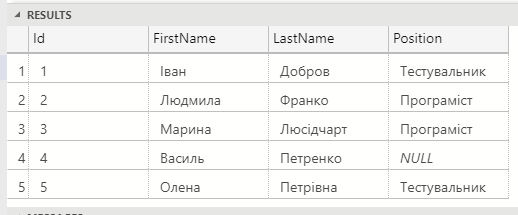


Рисунок 8 – Таблиця SENDER після запиту 5

**Запит 6 (на видалення):**

Delete FROM SENDER

where Position is NULL

**Результат**:

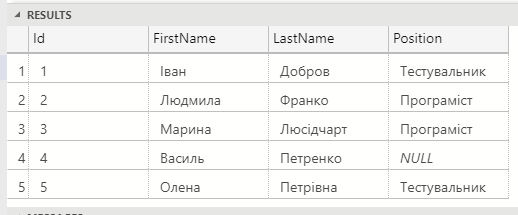


Рисунок 9 – Таблиця SENDER до запиту 6



Рисунок 10 – Таблиця SENDER після запиту 6

**Запит 7 (на оновлення):**

UPDATE COMMENT

SET SenderId = 2

where Id = 5

**Результат**:

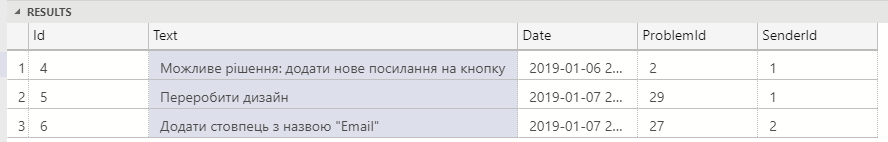


Рисунок 11 – Таблиця Employee до запиту 7

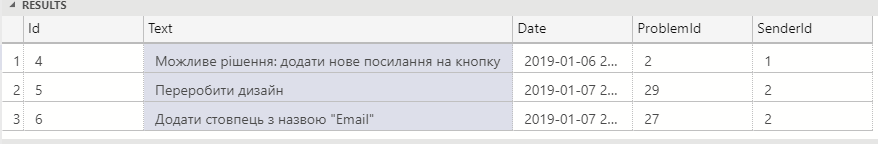


Рисунок 12 – Таблиця Employee після запиту 7

# Розроблення необхідних форм введення інформації

Для розробки веб-застосування, за допомогою якого можна було б зручно оброблювати дані, мною було обрано мову програмування C# та фреймворк ASP.NET Core (Model-View-Controller).

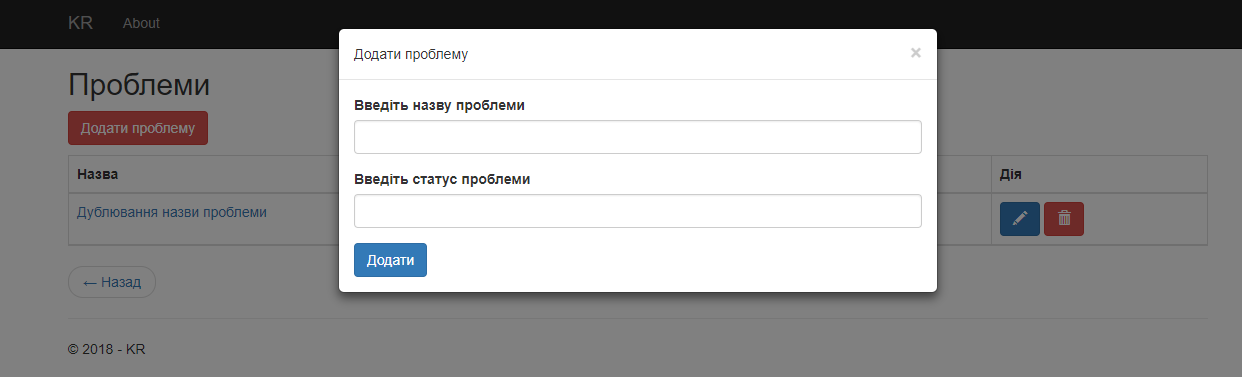


Рисунок 13 – Форма для додавання нової проблеми у БД

Для додавання нової проблеми я використала технологію Entity Framework Core :

[HttpPost]

[Route("problem")]

public async Task<IActionResult> AddProblem(Problem problem)

{

problem.CreationDate = DateTime.Now;

\_dbContext.Set<Problem>().Add(problem);

await \_dbContext.SaveChangesAsync();

var problems = await \_dbContext.Set<Problem>()

.Where(x => x.TypeId == problem.TypeId)

.AsNoTracking()

.ToArrayAsync();

return View("Problem", problems);

}

Форма коректно оброблює неправильно введені значення.

Для видалення даних з бази даних знову використовуюється технологія Entity Framework Core:

[HttpPost]

[Route("deleteproblem")]

public async Task<IActionResult> DeleteProblem(Problem problem)

{

\_dbContext.Set<Problem>().Remove(problem);

await \_dbContext.SaveChangesAsync();

var problems = await \_dbContext.Set<Problem>()

.AsNoTracking()

.ToArrayAsync();

return View("Problem", problems);

}

# Розроблення форм необхідних звітів для виводу даних

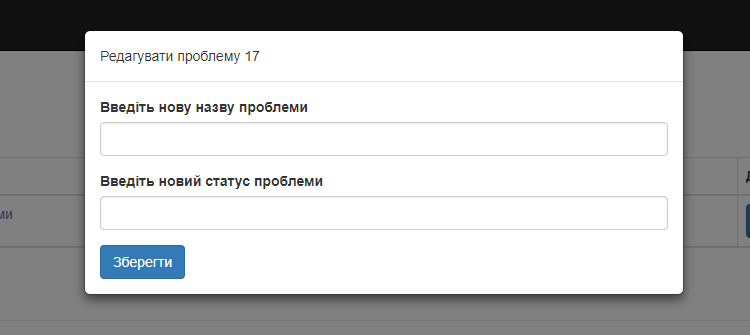


Рисунок14.Форма редагування проблеми.

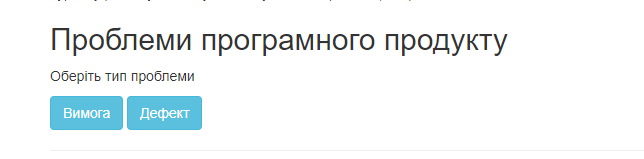


Рисунок 15. Форма вибору типу проблем.

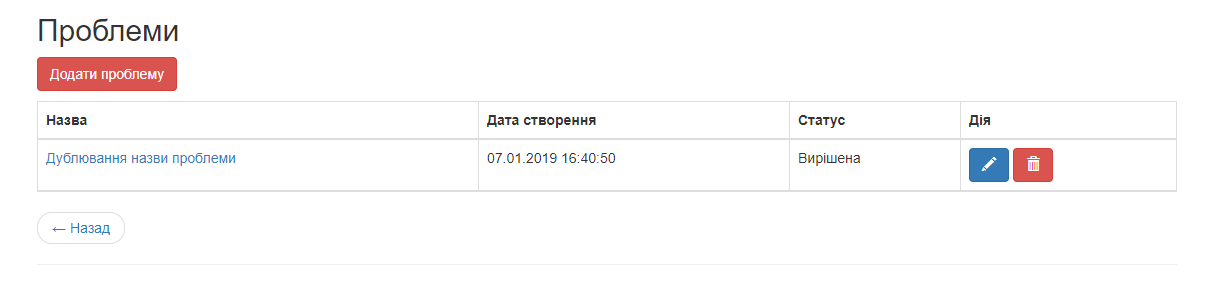


Рисунок16. Представлення проблем у вигляді таблиці.

[HttpGet]

[Route("{typeId:int}")]

public async Task<IActionResult> Problem(int typeId)

{

try

{

ViewBag.TypeId = typeId;

var typeIdParam = new SqlParameter("@typeIdParam", typeId);

var query = @"

SELECT PROBLEM.\*

FROM

PROBLEM\_TYPE

JOIN PROBLEM on PROBLEM\_TYPE.Id = PROBLEM.TypeId

where PROBLEM\_TYPE.Id = @typeIdParam";

var problems = await \_dbContext.Set<Problem>()

.FromSql(query, typeIdParam)

.AsNoTracking()

.ToArrayAsync();

return View(problems);

}

catch

{

return View(new ErrorViewModel { RequestId = Activity.Current?.Id ?? HttpContext.TraceIdentifier });

}

}

При обранні типу проблеми повертаєься таблиця з відповідними проблемами та діями, які доступні для них.

Запит SQL:

SELECT PROBLEM.\*

FROM

PROBLEM\_TYPE

JOIN PROBLEM on PROBLEM\_TYPE.Id = PROBLEM.TypeId

where PROBLEM\_TYPE.Id = @typeIdParam";

При натисненні на посилання для кожної проблеми з’явиться інша сторінка з даними про коментарі до обраної проблеми.

SELECT Employee.fullName, Employee.birthDate, Employee.passport, Employee.specialization, Employee.salary

FROM Employee

WHERE Employee.employeeId = 1

static func selectEmployee(id: Int64) -> Employee? {

do {

if let row = try db?.pluck(employee.filter(self.employee[Employee.expressions["id"] as! Expression<Int64>] == id)) {

return Employee(id: row[Employee.expressions["id"] as! Expression<Int64>],

fullName: row[Employee.expressions["fullName"] as! Expression<String>],

passport: row[Employee.expressions["passport"] as! Expression<String>],

birthDate: row[Employee.expressions["birthDate"] as! Expression<String>],

specialization: row[Employee.expressions["specialization"] as! Expression<String>],

salary: row[Employee.expressions["salary"] as! Expression<Int64>],

commandId: row[Employee.expressions["commandId"] as! Expression<Int64>])

}

} catch {

print(error)

}

return nil

}

# Висновки

База даних та застосунок, які були створені мною можуть допомогти на підприємстві. У зручному вигляді надається інформація про доходи від реалізації кожної культури. Можна вести облік техніки та робітників, зберігати інформацію за датою. Реляційна база даних є оптимальною для цієї області через те, що схема стає більш нормалізованою.

Дану систему можна сприймати лише як прототип, адже в ній передбачений лише базовий функціонал. Втім, система спроектована таким чином, що легко підтримує масштабування и відкрита для додавання нового функціоналу, тому вона легко може бути використана як основа майбутнього проекту.

# Список використаної літератури

1. <https://stackoverflow.com/>
2. <https://docs.microsoft.com/ru-ru/aspnet/core/?view=aspnetcore-2.2>
3. <https://www.appcoda.com/sqlite-database-ios-app-tutorial/>
4. https://github.com/stephencelis/sqlite.swift