Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Кафедра Програмної інженерії

КУРСОВА РОБОТА

ПОЯСНЮВАЛЬНА ЗАПИСКА

з дисципліни «Об’єктно-орієнтоване програмування»

ДОВІДНИК АБІТУРІЄНТА

Керівник , професор Володимир Михайлович Бондарєв

Студент гр. ПЗПІ-19-4 Гриненко Анастасія Олексіївна

Комісія:

старший викладач \_\_\_\_\_\_\_\_\_\_\_\_ Черепанова Ю.Ю.,

професор \_\_\_\_\_\_\_\_\_\_\_\_ Бондарєв В. М.,

доцент \_\_\_\_\_\_\_\_\_\_\_\_\_ Побіженко І. О.

Харків 2019

**ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ**

Кафедра: ***Програмної інженерії***

Дисципліна: ***Об’єктно-орієнтоване програмування***

Спеціальність: ***121***  ***Інженерія програмного забезпечення***

Освітня програма: ***Програмна інженерія***

Курс 1 .Група *ПЗПІ-19-4* . Семестр *2* .

***ЗАВДАННЯ***

***на курсовий проект студента***

***Гриненко Анастасії Олексіївни***

(Прізвище, Ім'я, По батькові)

1 Тема проекту: *Довідник абітурієнта*

2 Термін здачі студентом закінченого проекту: ***“31” - травня - 2019 р***.

3 Вихідні дані до проекту:

*Специфікація програми, методичні вказівки до виконання курсової роботи*

4 Зміст розрахунково-пояснювальної записки:

*Вступ,* *специфікація програми, проектна специфікація, інструкція користувача, висновки\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

5 Перелік графічного матеріалу:

*Діаграми класів,* *схема об’єктної моделі, алгоритми, приклади екранних форм\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**КАЛЕНДАРНИЙ ПЛАН**

|  |  |  |
| --- | --- | --- |
| *№* | *Назва етапу* | *Термін виконання* |
| 1 | Видача теми, узгодження і затвердження теми. | 30-03-2019 р. |
| 2 | Формулювання вимог до програми, оформлення специфікації. | 30-03-2019 – 05-04-2019 р. |
| 3 | Проектування моделі. | 06-04-2019 – 12-04-2019 р. |
| 4 | Розробка моделі, тобто системи основних класів та функцій зберігання та завантаження даних. | 13-04-2019 – 26-04-2019 р. |
| 5 | Встановлення різних рівнів доступу до функціоналу шляхом створення двох типів користувачів – диспетчера та клієнта. | 27-04-2019 – 03-05-2019 р. |
| 6 | Розробка інтерфейсу користувача. | 04-05-2019 – 17-05-2019 р. |
| 7 | Тестування і доопрацювання розробленої програмної системи. | 18-05-2019 – 20-05-2019 р. |
| 8 | Оформлення пояснювальної записки, додатків, графічного матеріалу | 21-05-2019– 31-05-2019 р. |
| 9 | Захист | 01-06-2019 – 05-06-2019 р. |

Студент Гриненко Анастасія Олексіївна

Керівник Бондарєв Володимир Михайлович

«30 » березня 2019 р.

**РЕФЕРАТ**

Пояснювальна записка до курсової роботи: 49 с., 15 рис., 5 додатків, 5 джерел.

ДОВІДНИК, УНІВЕРСИТЕТ, АБІТУРІЄНТ, КЛАС, МОВА ПРОГРАМУВАННЯ C#, ОБ’ЕКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ, ПЛАТФОРМА .NET FRAMEWORK, ПРОГРАМА.

Метою роботи є розробка програми «Довідник абітурієнта» на засадах об'єктно-орієнтованого програмування.

Методи розробки базуються на використанні середи розробки Microsoft Visual Studio 2017, Windows Forms, платформи .NET Framework 4.7, мови програмування C#.

У результаті отримана програма під назвою «Довідник абітурієнта», яка дозволяє користувачу переглядати заклади вищої освіти та інформацію про них, додавати та видаляти університети зі Списку обраних. Адміністратор може видаляти вже існуючі університети та додавати нові.

**ЗМІСТ**

[ВСТУП 6](#_Toc41761216)

[1. СПЕЦИФІКАЦІЯ ПРОГРАМИ 7](#_Toc41761217)

[1.1 Функції абітурієнта 7](#_Toc41761218)

[1.2 Функції адміністратора 7](#_Toc41761219)

[1.3 Функціональні вимоги 7](#_Toc41761220)

[1.4 Нефункціональні вимоги 8](#_Toc41761221)

[1.5 Опис інтерфейсу користувача 8](#_Toc41761222)

[2. ПРОЕКТНА СПЕЦИФІКАЦІЯ 16](#_Toc41761223)

[2.1 Архітектура 16](#_Toc41761224)

[2.2 Діаграма класів 16](#_Toc41761225)

[2.3 Структура програми 17](#_Toc41761226)

[2.4 Об’єктна структура програми 18](#_Toc41761227)

[2.4.1 Клас Directory 19](#_Toc41761228)

[2.4.2 Клас University 20](#_Toc41761229)

[2.4.3 Клас Spetiality 21](#_Toc41761230)

[2.4.4 Абстрактний клас User 21](#_Toc41761231)

[2.4.5 Клас Applicant 22](#_Toc41761232)

[2.4.6 Клас Admin 22](#_Toc41761233)

[2.4.7 Клас Dao 22](#_Toc41761234)

[2.5 Доступ до даних 23](#_Toc41761235)

[3. ІНСТРУКЦІЯ КОРИСТУВАЧА 24](#_Toc41761236)

[3.1 Встановлення, запуск та головне вікно 24](#_Toc41761237)

[ВИСНОВКИ 26](#_Toc41761238)

[ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ 27](#_Toc41761239)

[ДОДАТОК А Методи класу Directory 28](#_Toc41761240)

[ДОДАТОК Б Методи класу Dao 30](#_Toc41761241)

ВСТУП

Основна функція програми – допомога абітурієнтам при вступі до вищого навчального закладу. Для цього важливо забезпечити легкість абітурієнтів у перегляді списку навчальних закладів, додавання їх до обраних, щоб порівнювати цікаві для користувача заклади вищої освіти.

Розроблена програма дозволить користувачу переглядати заклади вищої освіти та інформацію про них, додавати та видаляти університети зі Списку обраних. Адміністратор може видаляти вже існуючі університети та додавати нові. Доступ до облікового запису адміністратора здійснюється через пароль, що зберігається в програмі.

Окрім розробки вище описано ї програмної системи метою даної курсової роботи є освоїти мову програмування С#, навчитися об’єктно-орієнтованому програмуванню, розвити навички роботи в Windows Forms.

Задачі роботи:

* Розробка системи зчитування та запису даних;
* Створення логіки роботи системи;
* Робота з користувачем;
* Розробка логіки додавання, пошуку та видалення інформації;
* Тестування готового продукту;
* Створення стійкості програми.

1. СПЕЦИФІКАЦІЯ ПРОГРАМИ
   1. Функції абітурієнта

Серед функцій абітурієнта слід зазначити наступні:

* Перегляд таблиці з університетами;
* Реєстрація абітурієнта;
* Авторизація абітурієнта.
  1. Функції адміністратора
* Авторизація адміністратора;
* Додававння університетів (адміністратор);
* Видалення університетів (адміністратор).
  1. Функціональні вимоги

Серед функціональних вимог слід зазначити наступні:

* Система має зчитувати з файла дані про університети та записувати туди зміни у ньому за необхідності;
* Система має зчитувати з файла Список обраних абітурієнта та записувати туди зміни у ньому за необхідності;
* Система має звіряти введений пароль з паролем, зчитаним з файлу.
  1. Нефункціональні вимоги

Серед нефункціональних вимог слід зазначити наступні:

* Легкість та простота використання;
* Цілісність;
* Ефективність та стійкість до збоїв;
* Взаємодія системи і зовнішнього світу;
* Обмеження дизайну і реалізації.

Серед обмежень дизайну і реалізації слід зазначити наступні:

* Використання платформи .NET Framework 4.5;
* Використання мови програмування C#;
* Використання Windows Forms.
  1. Опис інтерфейсу користувача

Після запуску програми відкривається її форма входу (рис. 1.2), на якій відображається поля для введення імені та пароля, та яка дає можливість зареєструватися в системі.

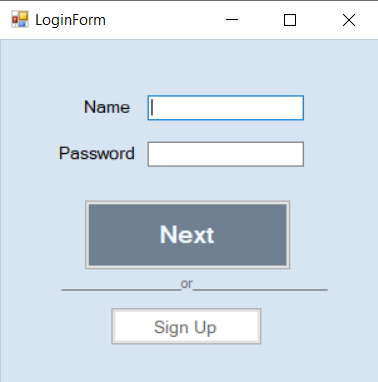


Рисунок 1.2 – Форма входу

При натисканні кнопки «Sign Up» відкривається форма реєстрації абітурієнта з полями для імені, пароля та перевірки пароля (рис. 1.3). Дані зберігаються в програмі після натискання кнопки «ОК», а програма перенаправляє користувача на вікно входу.

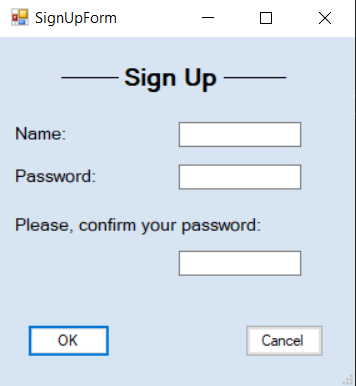


Рисунок 1.3 – Форма реєстрації

Користувач вводить логін на пароль, натискає кнопку «Next». Введені дані перевіряються, та відкривається головне вікно адміністратора (рис.1.4) або головне вікно абітурієнта (рис. 1.5).

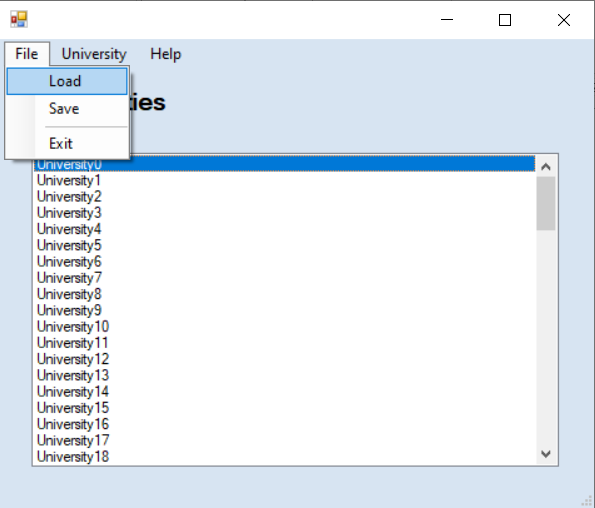


Рисунок 1.4 – Головне вікно адміністратора

Якщо дані не пройшли перевірку, на формі вводу з’являється повідомлення про це, та можна спробувати увійти знову. Кількість спроб не обмежена.

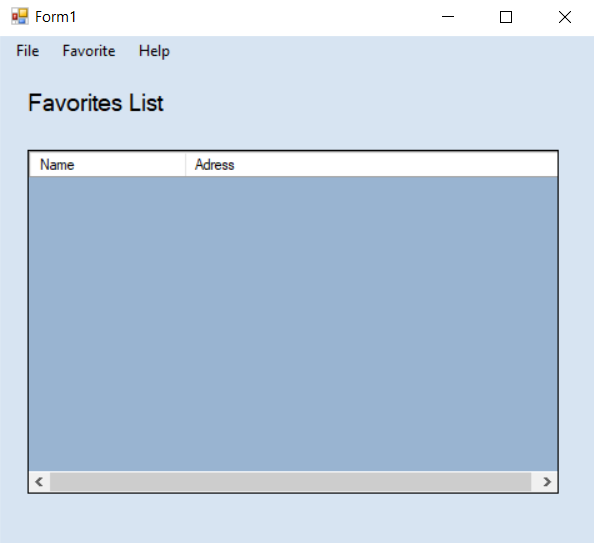


Рисунок 1.5 – Головне вікно абітурієнта

На головному адміністратора вікні є меню з трьох частин. В пункті «File» можна брати «Load», «Save» або «Exit». Обравши «Load», адміністратор завантажує дані з файлу. «Save» зберігає внесені адміністратором зміни, а за допомогою «Exit» можна вийти з програми. У разі виходу з’явиться стандартне діалогове вікно (рис. 1.6).

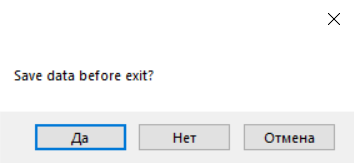


Рисунок 1.6 – Діалогове вікно при виході з програми

Другий пункт меню має підпункти «New» та «Delete». Натиснувши «New», адміністратор переходить до форми університету (рис. 1.7), де може ввести назву та адресу нового університету.

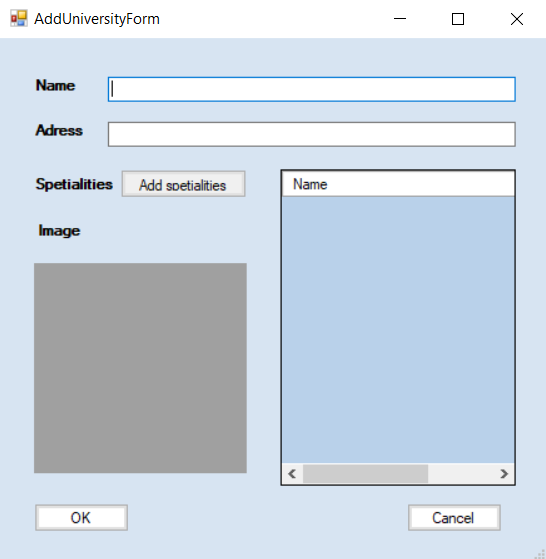


Рисунок 1.7 – Форма університету

Кнопка «Add spetialities» дає можливість додавати спеціальності обраному університету, та відкриває форму спеціальності (рис. 1.8).

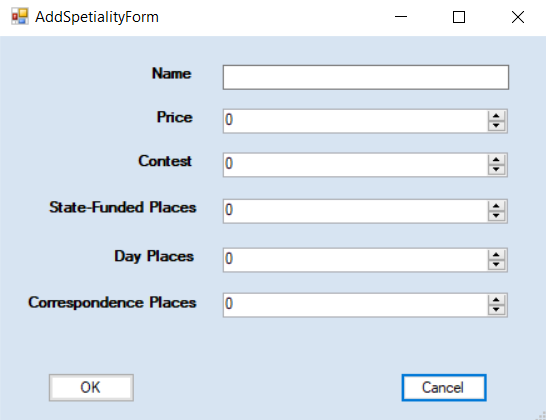


Рисунок 1.8 – Форма спеціальності

Для видалення університету з головної панелі адміністратор має обрати університет і вибрати в меню пункт «Delete». Вибраний університет не відображатиметься в списку.

Третій пункт меню під назвою «Help» має в собі посилання на репозиторій на GitHub цього проекту.

Меню на головній панелі абітурієнта має «File» за такими ж підменю, як і в адміністратора, «Favorite» «Help». Обравши «Favorite», відкривається підменю з позицій «Add» і «Delete». Головна панель – це Список обраних(рис. 1.9).

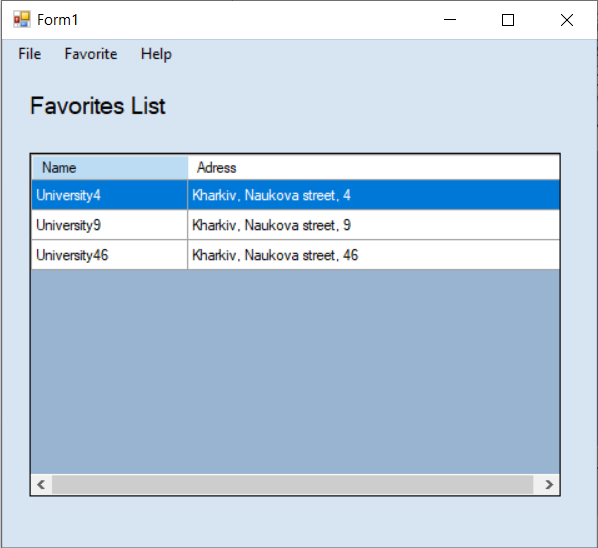


Рисунок 1.9 – Список обраних

Обравши «Add» в меню, відкривається форма додавання університету, де відображається таблиця всіх університетів у базі абітурієнта. При виборі «OK» вибраний університет додається в Список обраних.

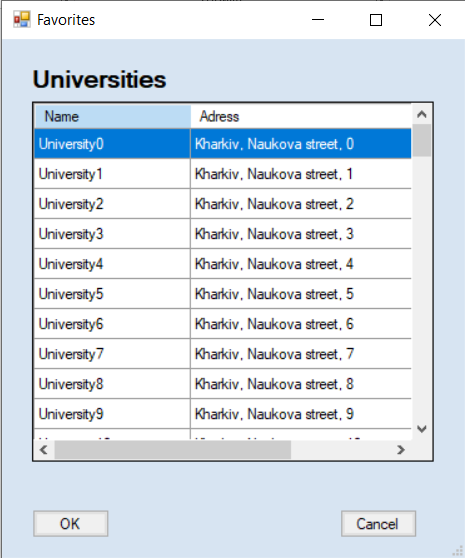


Рисунок 1.9 – Таблиця університетів

Абітурієнт може видаляти університети зі Списку обраних. Для цього потрібно вибрати університет на головній панелі, який треба видалити, а в меню обрати «Delete». Вибраний університет не відображатиметься в Списку.

1. ПРОЕКТНА СПЕЦИФІКАЦІЯ
   1. Архітектура

Програма створена на мові С # з використанням середовища розробки Visual Studio 2019 на платформі .NЕТ Framework 4.7.2 з використанням інтерфейсу програмування додатків Windows Forms.

Вибір мови обумовлений об'єктно-орієнтованим підходом до написання програми. На даний момент мова С # є одним з найпопулярніших, зручних і повноцінних серед об'єктно-орієнтованих мов програмування.

В процесі проектування розглядалися 2 платформи, з якими мова С # має найкраще взаємодію, а саме .NЕТ Core і .NЕТ Framework. Вибір припав на останню, через велику кількість керуючих елементів і більш високий рівень розвитку платформи. На жаль, це означає що додаток буде працювати виключно на машинах з операційною системою Windows.

Для реалізації призначеного для користувача інтерфейсу було вирішено використовувати графічний інтерфейс так як він простий для сприйняття і інтуїтивно зрозумілий. Для реалізації інтерфейсу ми будемо використовувати інтерфейс Windows Forms який є частиною платформи .NЕТ Framework. З його допомогою у нас буде можливість створити повноцінний інтерфейс для користувачів програми.

* 1. Діаграма класів

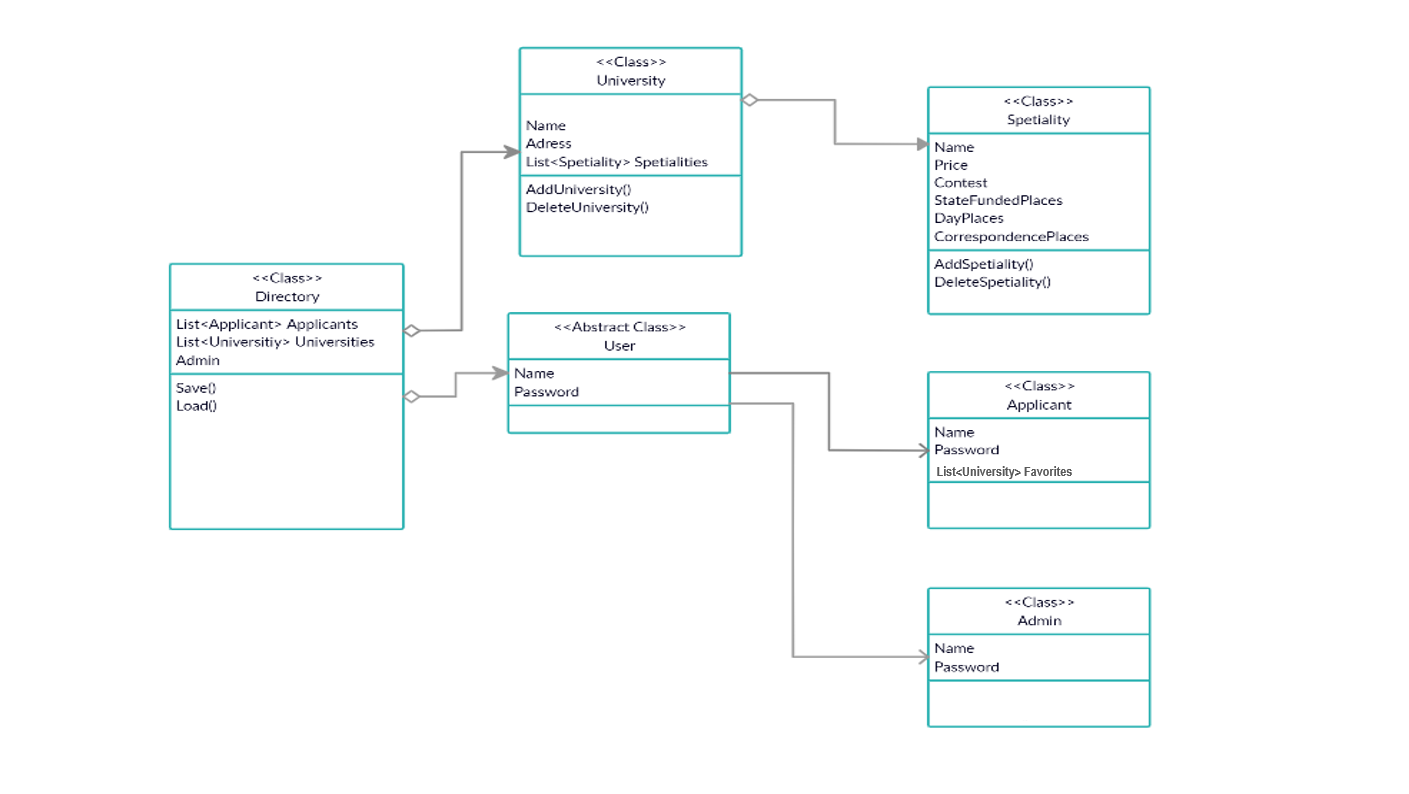


Рисунок 2.1 – Діаграма класів

* 1. Структура програми

Програма складається з 7 класів і 7 форм. Усі форми були розглянуті у попередньому розділі. Програма, описана в специфікації, поділена на три окремих додатки: вхід, прогрма для адміністратора та для абітурієнта, оскільки функції цих типів користувачів між собою не перетинається.

На рисунку 2.2 наведено структуру проекту.

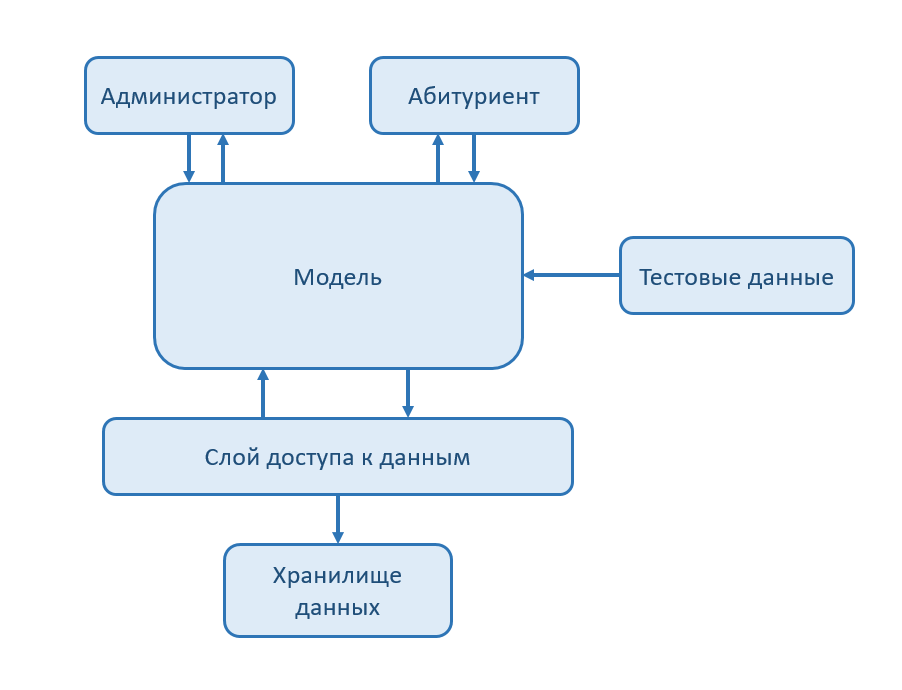


Рисунок 2.2 – Структура проекту

* 1. Об’єктна структура програми

У базі абітурієнта є університети.

Абітурієнт шукає університет, переглядає інформацію про них, додає і видаляє університети зі Списку обраних.

Адміністратор додає та видаляє університети.

Про університет відомо: назва, зображення, адреса, спеціальності.

Про абітурієнта відомо: пароль, ім'я, список обраних.

Про адміністратора відомо: пароль, ім'я.

Про спеціальність відомо: назва, ціна навчання, прохідний бал на бюджет минулого року, кількість місць на навчання на бюджетній основі, кількість всіх місць, кількість місць на денну форму, кількість місць на заочну форму навчання.

Абітурієнт - це пароль + ім'я + список обраних.

База абітурієнтів - це колекція всіх абітурієнтів.

Адміністратор - це пароль + ім'я.

Спеціальність - це назва + ціна навчання + прохідний бал на бюджет минулого року + кількість місць на навчання на бюджетній основі + кількість всіх місць + кількість місць на денну форму + кількість місць на заочну форму навчання.

Університет - це колекція спеціальностей + назва + зображення + адреса.

База університетів - це колекція всіх університетів.

Список обраних - це колекція університетів.

Довідник абітурієнта - це колекція абітурієнтів + колекція університетів + адміністратор.

2.4.1 Клас Directory

Клас Directory описує довідник абітурієнта. Складається з трьох відкритих властивостей:

//Адміністратор.

public Admin Admin{ private set; get; }

//Список університетів.

public List<University> Universities { private set; get; }

//Список абітурієнтів.

public List<Applicant> Applicants { private set; get; }

Клас Directory має три методи (див. додаток А):

public void FillTestData(int n) – заповнює довідник абітурієнта тестовими даними.

public void Save() – звертається до методу, який зберігає дані в файлі .bin.

public void Load() – звертається до методу, який загружає дані з файлу.

2.4.2 Клас University

Клас описує університет з наступними відкритими властивостями:

//Адреса.

public string Adress { set; get; }

//Зображення.

public Image Image { set; get; }

//Назва.

public string Name { set; get; }

//Список спеціальностей.

public List<Spetiality> Spetialities { get; set; }

2.4.3 Клас Spetiality

Клас Spetiality описує спеціальність. Він має наступні властивості:

//Назва.

public string Name { set; get; }

//Ціна.

public decimal Price { set; get; }

//Прохідний бал минулого року.

public decimal Contest { set; get; }

//Бюджетні місця.

public decimal StateFundedPlaces { set; get; }

//Місця на денну форму.

public decimal DayPlaces { set; get; }

//Місця на заочну форму.

public decimal СorrespondencePlaces { set; get; }

//Всього місць.

public decimal Places { set; get; }

2.4.4 Абстрактний клас User

Клас User має відкриті властивості

//Ім’я.

public string Name { set; get; }

//Пароль.

public string Password { set; get; }

2.4.5 Клас Applicant

Клас Applicant описує абітурієнта та є спадкоємцем абстрактного класу User, у нього з’являються такі властивості:

//Список обраних.

public List<University> Favorites = new List<University>();

2.4.6 Клас Admin

Клас Admin описує адміністратра та наслідує клас User.

2.4.7 Клас Dao

Клас Dao потрібен для збереження та загрузки даних. Методи класу виконують саме такі функції (див. додаток Б):

public void Save() – зберігає дані в файлі .bin.

public void Load() – загружає дані з файлу.

* 1. Доступ до даних

Всі дані, зазначені в роботі словом «магазин», а також «список користувачів» будуть представлені у вигляді бінарного файлу в каталозі проекту. Дані будуть переводитися з текстових в потік байтів за допомогою серіалізації з використанням вбудованої бібліотеки ВinaryFormatter. Файл під назвою «directory.bin» розташований за адресою - «\bin\ Debug». Захист даних в рамках роботи не розглядається.

1. ІНСТРУКЦІЯ КОРИСТУВАЧА
   1. Встановлення, запуск та головне вікно

Для користування програмою її необхідно завантажити на ваш комп’ютер. Після цього необхідного відкрити файл за шляхом \ApplicantsGuide\LoginForm\bin\Debug\LoginForm.exe.

* 1. Можливості адміністратора

Після запуску програми відкривається її форма входу (рис. 3.1), на якій відображається поля для введення імені та пароля, яке вже занесено в систему. Щоб зайти у якості адміністратора потрібно ввести ім’я «admin» та пароль «admin».

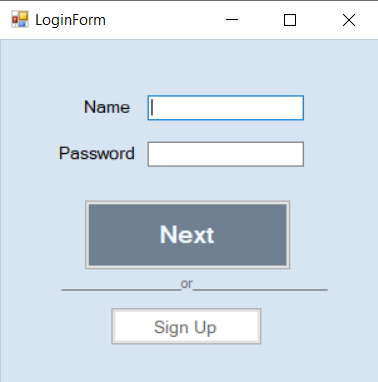


Рисунок 3.1 – Форма входу

Адміністратор вводить логін на пароль, натискає кнопку «Next». Введені дані перевіряються, та відкривається головне вікно абітурієнта (рис.3.3). Якщо дані не пройшли перевірку, то можна спробувати увійти знову. Кількість спроб не обмежена.

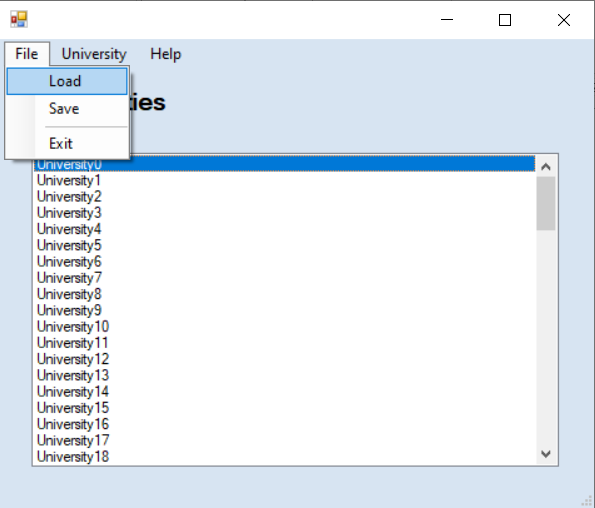


Рисунок 3.2 – Головне вікно адміністратора

На головному адміністратора вікні є меню з трьох частин. В пункті «File» можна брати «Load», «Save» або «Exit». Обравши «Load», адміністратор завантажує дані з файлу. «Save» зберігає внесені адміністратором зміни, а за допомогою «Exit» можна вийти з програми. У разі виходу з’явиться стандартне діалогове вікно.

Другий пункт меню має підпункти «New» та «Delete». Натиснувши «New», адміністратор переходить до форми університету (рис. 3.3), де може ввести назву та адресу нового університету.

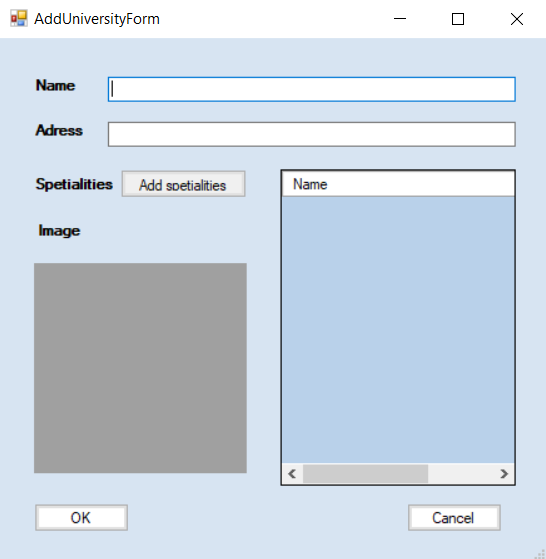


Рисунок 3.3 – Форма університету

Кнопка «Add spetialities» дає можливість додавати спеціальності обраному університету, та відкриває форму спеціальності (рис. 3.4).

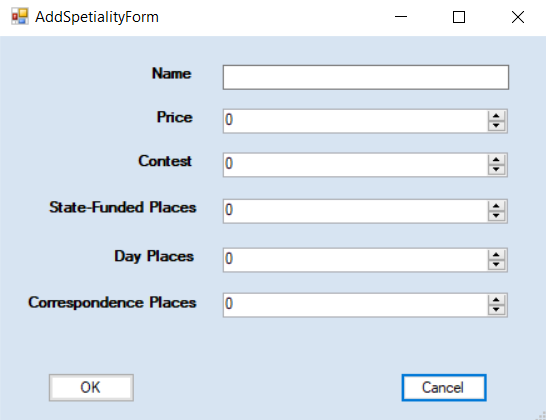


Рисунок 3.4 – Форма спеціальності

Для видалення університету з головної панелі адміністратор має обрати університет і вибрати в меню пункт «Delete». Вибраний університет не відображатиметься в списку.

Третій пункт меню під назвою «Help» має в собі посилання на репозиторій на GitHub цього проекту.

* 1. Можливості абітурієнта

Після запуску програми відкривається її форма входу (рис. 3.1), на якій відображається поля для введення імені та пароля, та яка дає можливість зареєструватися в системі.

При натисканні кнопки «Sign Up» відкривається форма реєстрації абітурієнта з полями для імені, пароля та перевірки пароля (рис. 3.5). Дані зберігаються в програмі після натискання кнопки «ОК», а програма перенаправляє користувача на вікно входу.

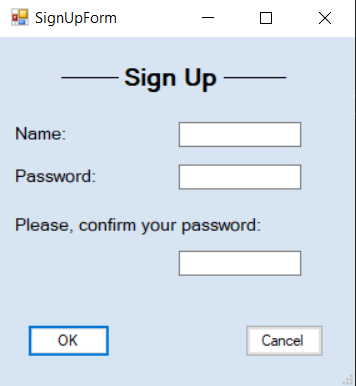


Рисунок 3.5 – Форма реєстрації

Користувач вводить логін на пароль, натискає кнопку «Next». Введені дані перевіряються, та відкривається головне вікно абітурієнта (рис.3.6).

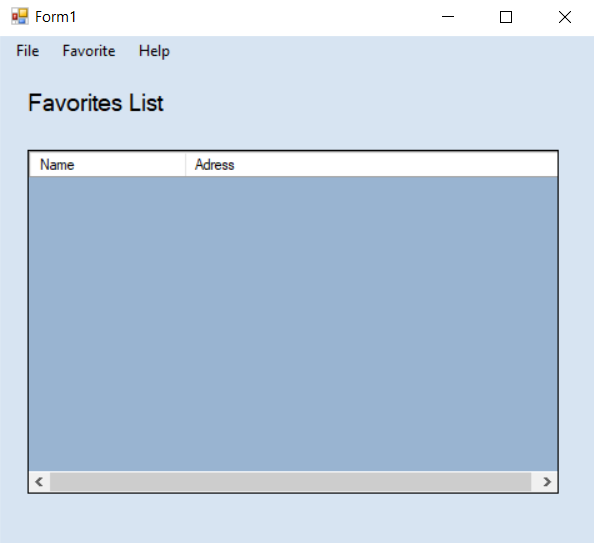


Рисунок 3.6 – Головне вікно абітурієнта

Меню на головній панелі абітурієнта має «File» з такими ж підменю, як і в адміністратора, «Favorite» «Help». Обравши «Favorite», відкривається підменю з позицій «Add» і «Delete». Головна панель – це Список обраних(рис. 3.7).

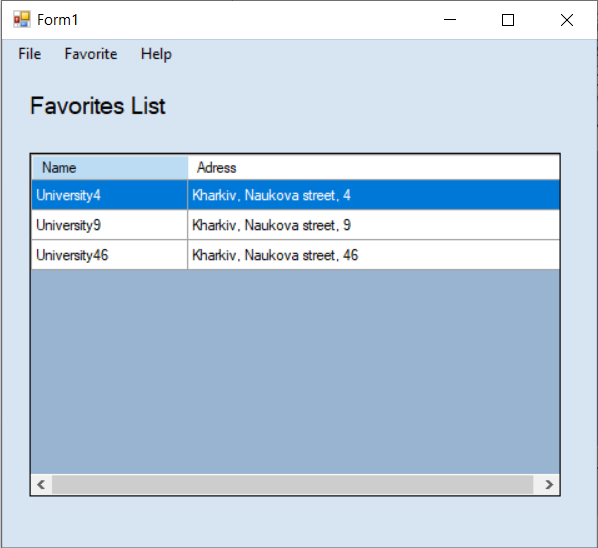


Рисунок 3.7 – Список обраних

Обравши «Add» в меню, відкривається форма додавання університету (рис. 3.8), де відображається таблиця всіх університетів у базі абітурієнта. При виборі «OK» вибраний університет додається в Список обраних.

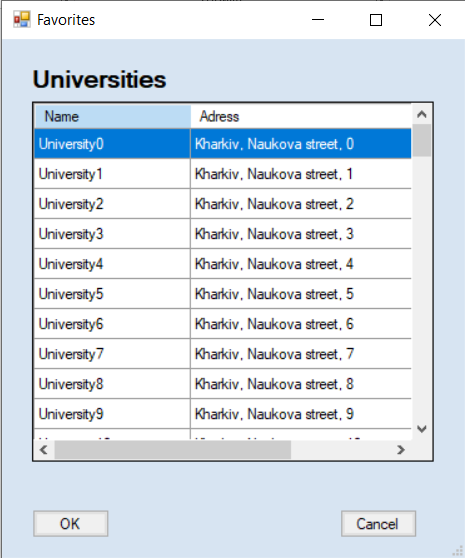


Рисунок 3.8 – Форма додавання університетів

Абітурієнт може видаляти університети зі Списку обраних. Для цього потрібно вибрати університет на головній панелі, який треба видалити, а в меню обрати «Delete». Вибраний університет не відображатиметься в Списку.

ВИСНОВКИ

У ході виконання курсової роботи були освоєні методи створення програмних систем на засадах об’єктно-орієнтованого програмування, набулися навички роботи з програмною системою, створення зв’язків між об’єктами.

У результаті отримана програма під назвою «Довідник абітурієнта», яка дозволяє звичайному користувачу переглядати заклади вищої освіти, продивлятися інформацію про університети, додавати та видаляти університети зі Списку обраних. Авторизований користувач (адміністратор) може змінювати інформацію щодо ЗВО, видаляти вже існуючі університети та додавати нові.

Програма написана на мові програмування C# на платформі .NET Framework 4.5 з використанням технологій Windows Forms у середі розробки Microsoft Visual Studio 2017.

Усі дані програми зберігаються локально у файла .bin. Програма може бути поліпшена застосуванням справжньої бази даних, яка синхронізується з базою даних університетів по мережі.

Галузь застосування – вища освіта.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Полное руководство по языку программирования С# 7.0 и платформе .NET 4.7 [Електронний ресурс] – Режим доступу до ресурсу: https://metanit.com/sharp/tutorial/.
2. Бондарев В. М. Объектно-ориентированное программирование на С# [Текст] : учеб. Пособие / В. М. Бондарев. – Харьков : СМИТ, 2009. –224 с. – Библиогр. : ISBN 978-966-2028-34-8.
3. Microsoft Developer Network. Библиотека MSDN / MSDN – сеть разработчиков Microsoft [Електронний ресурс] – Режим доступу до ресурсу: https://msdn.microsoft.com/ru-ru/.
4. Троелсен, Э. Язык программирования C# 5.0 и платформа .NET 4.5, 6-е изд. [Текст]: пер. с англ./ Э. Троелсен.- М.: Вильямс, 2013 - 1312 с. – ISBN 978-5-8459-1814-7, 978-1-43-024233-8.
5. Либерти Дж. Программирование на С# [Текст] : пер. с англ. / Дж. Либерти. – 2-е изд. – СПб. ; М. : Символ-Плюс, 2003. – 688 с.

ДОДАТОК А  
Методи класу Directory

//Заповнення тестових даних.

public void FillTestData(int n)

{

Universities.Clear();

//var noImage = new Bitmap(Path.GetFullPath("empty.png"));

// Университеты.

Universities = new List<University>();

for (int i = 0; i < n - 5; i++)

{

//Специальности.

var spetialities = new List<Spetiality>();

for (int j = 0; j < 5; j++)

{

spetialities.Add(new Spetiality

(

$"Spetiality{j}",

j \* 1000 + 10000,

170 + j,

30 + j,

80 + j,

j

));

}

Universities.Add(new University

(

//Image = noImage,

$"Kharkiv, Naukova street, {i}",

$"University{i}",

spetialities

));

}

Applicants.Clear();

// Абитуриенты.

Applicants = new List<Applicant>();

for (int i = 0; i < n; i++)

{

Applicants.Add(new Applicant ( $"Applicant{i}", "123" ));

}

////Админ.

}

//Сохранение и загрузка данных.

public void Save()

{

new Dao(this).Save();

}

public void Load()

{

new Dao(this).Load();

}

ДОДАТОК Б  
Методи класу Dao

//Збереження.

public void Save()

{

using (Stream stream = File.Create(filePath))

{

var serializer = new BinaryFormatter();

serializer.Serialize(stream, directory);

}

}

//Загрузка.

public void Load()

{

using (Stream stream = File.OpenRead(filePath))

{

var serializer = new BinaryFormatter();

Directory dr = (Directory)serializer.Deserialize(stream);

Copy(dr.Applicants, directory.Applicants);

Copy(dr.Universities, directory.Universities);

}

void Copy<T>(List<T> from, List<T> to)

{

to.Clear();

to.AddRange(from);

}

}