МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний аерокосмічний університет ім. М. Є. Жуковського

«Харківський авіаційний інститут»

Факультет програмної інженерії та бізнесу

Кафедра інженерії програмного забезпечення

**КУРСОВА РОБОТА**

з курсу: «Об’єктно-орієнтоване програмування»

на тему: «Розроблення програмного забезпечення з використанням

об’єктно-орієнтованої парадигми»

Виконав: студент (ка) 2 курсу групи 622п

Спеціальність 121 – Інженерія програмного забезпечення

(код та найменування)

Зайченка Я. І.

(прізвище й ініціали студента(ки))

Керівник: Лучшев П. О.

(прізвище й ініціали)

Національна шкала \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кількість балів: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Оцінка: ECTS \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Члени комісії \_\_\_\_\_\_\_\_ Лучшев П.О.

(підпис) (прізвище та ініціали)

\_\_\_\_\_\_\_\_ Дем’яненко В.А.

(підпис) (прізвище та ініціали)

Харків – 2024

**ТИПОВЕ ЗАВДАННЯ НА КУРСОВУ РОБОТУ**

Варіант 10. Бронювання житла (Booking)

|  |  |  |
| --- | --- | --- |
| **Критерій оцінювання** | **Бали** | **Штрафи** |
| **Покрокове виконання курсової роботи:** | | |
| Розділ 1(крайній термін – **3 тиждень**) | 0..15 | -3 |
| Розділ 2(крайній термін – **6 тиждень**) | 0..15 | -3 |
| Розділ 3(крайній термін – **9 тиждень**) | 0..15 | -3 |
| Розділ 4(крайній термін – **12 тиждень**) | 0..15 | -3 |
| Розділ 5(крайній термін – **13 тиждень**) | 0..5 | -1 |
| Оформлення пояснювальної записки (термін – **14 тиждень**) | 0..3 | -1 |
| **Підготовка до захисту курсової роботи:** | | |
| доповідь з презентацією (термін – **15 тиждень**) | 0..1 |  |
| відеоролик з демонстрацією роботи програми  (термін – **15 тиждень**) | 0..1 |  |
| **Захист курсової роботи:** | | |
| доповідь з презентацією | 0..5 |  |
| демонстрація роботи програми і github-репозиторію | 0..5 |  |
| відповіді на питання | 0..5 |  |
| використання\*:   1. інкапсуляції 2. статичних членів класів 3. інтерфейсів 4. абстрактних класів 5. спадкування 6. поліморфізму 7. .NET-delegates і events 8. сереалізації/десереалізації об’єктів 9. Collections.Generic 10. LINQ   \* мають бути представленні на захисті курсової роботи у презентації | 0..15 |  |
| **Всього за курсову роботу:** | **0..100** | **0..14** |

**Зміст**

[СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАЧЕННЯ 4](#_Toc157882173)

[ВСТУП 5](#_Toc157882174)

[1 ФУНКЦІОНАЛЬНІ ВИМОГИ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 6](#_Toc157882175)

[1.1 Аналіз функціональності програмних аналогів 6](#_Toc157882176)

[1.2 Побудова Mind Map для заданої предметної області 6](#_Toc157882177)

[1.3 Виділення ролей користувачів і формулювання функціональних вимог 7](#_Toc157882178)

[1.4 Проєктування інтерфейсу користувача 8](#_Toc157882179)

[1.5 Розроблення функціональних тестів 9](#_Toc157882180)

[1.5.1 Функціональні тести для ролі користувача «Власник картки з мешканням» 9](#_Toc157882181)

[1.5.2 Функціональні тести для ролі користувача «Зареєстрований користувач» 10](#_Toc157882182)

[1.5.3 Функціональні тести для ролі користувача «Гість» 10](#_Toc157882183)

[2 МОДЕЛЮВАННЯ ПРЕДМЕТНОЇ ОБЛАСТІ 12](#_Toc157882184)

[2.1 Виділення і опис класів предметної області 12](#_Toc157882185)

[2.2 Встановлення зв’язків між класами 13](#_Toc157882186)

[3 ПРОГРАМНА РЕАЛІЗАЦІЯ КЛАСІВ ПРЕДМЕТНОЇ ОБЛАСТІ І ЇХ ТЕСТУВАННЯ 14](#_Toc157882187)

[3.1 Структура проєкту з реалізацією класів предметної області 14](#_Toc157882188)

[3.2 Реалізація інтерфейсів і каркасів класів предметної області 14](#_Toc157882189)

[3.3 Розроблення unit-тестів для класів предметної області 14](#_Toc157882190)

[3.4 Повна реалізація класів предметної області 15](#_Toc157882191)

[3.5 Результати unit-тестування класів предметної області 15](#_Toc157882192)

[4 ПРОЄКТУВАННЯ І ПРОГРАМНА РЕАЛІЗАЦІЯ КЛАСІВ ІНТЕРФЕЙСУ КОРИСТУВАЧА 16](#_Toc157882193)

[4.1 Структура проєкту з реалізацією класів інтерфейсу користувача 16](#_Toc157882194)

[4.2 Виділення класів для реалізації інтерфейсу користувача 16](#_Toc157882195)

[4.3 Програмна реалізація класів інтерфейсу користувача 16](#_Toc157882196)

[5 ФУНКЦІОНАЛЬНЕ ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 17](#_Toc157882197)

[5.1 Функціональне тестування для ролі користувача «Власник картки з мешканням» 17](#_Toc157882198)

[5.2 Функціональне тестування для ролі користувача «Зареєстрований користувач» 18](#_Toc157882199)

[5.3 Функціональне тестування для ролі користувача «Гість» 18](#_Toc157882200)

[ВИСНОВКИ 19](#_Toc157882201)

[ПЕРЕЛІК ДЖЕРЕЛ ТА ПОСИЛАНЬ 20](#_Toc157882202)

[ДОДАТОК А. Лістинг класів предметної області 21](#_Toc157882203)

[ДОДАТОК Б. Лістинг класів інтерфейсу користувача 22](#_Toc157882204)

[ДОДАТОК В. Лістинг класів unit-тестів 23](#_Toc157882205)

[ДОДАТОК Г. Назва додатку 24](#_Toc157882206)

# СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАЧЕННЯ

|  |  |  |
| --- | --- | --- |
|  | – |  |
| ООП | – | об’єктно-орієнтоване програмування |
| … |  | … |

# ВСТУП

Сучасний світ все більше віддає перевагу онлайн-сервісам, які надають можливість зручно та ефективно здійснювати різноманітні операції. Однією з найпоширеніших потреб у таких сервісах є бронювання житла для подорожей. З урахуванням активного розвитку туристичної галузі та постійного зростання популярності подорожей як для відпочинку, так і для бізнесу, актуальність створення програмного забезпечення для бронювання житла, аналогічного популярним сервісам, таким як Booking.com, стає очевидною.

Потенційними користувачами програми можуть бути:

* Туристи: люди, які шукають тимчасове проживання під час подорожі або відпочинку. Це можуть бути як індивідуальні мандрівники, так і групи туристів.
* Відрядження: представники компаній, які подорожують у бізнес цілях, тимчасово перебуваючи в інших містах або країнах.
* Власники готелів, гостьових будинків, апартаментів та інших об'єктів надання житла: особи, які надають свої об'єкти для здачі в оренду тим, хто шукає житло.
* Менеджери подорожей (турагенти): фахівці, які організовують подорожі для інших людей або груп і шукають оптимальні пропозиції щодо проживання.
* Інші: включає в себе будь-яких осіб, які мають потребу в тимчасовому проживанні та використовують подібні сервіси для цього.

Серед лідерів для бронювання житла можна виділити такі платформи:

* Booking.com: один з найвідоміших та найпопулярніших сервісів для бронювання готелів, апартаментів, гостьових будинків та інших видів житла по всьому світу.
* Airbnb: платформа, яка дозволяє власникам нерухомості здавати її в оренду туристам, шукаючим тимчасове житло, та дозволяє користувачам знаходити унікальні місця для перебування.
* Expedia: сервіс, який пропонує можливість бронювати готелі, літаки, автомобілі для оренди та інші послуги у рамках одного сайту.
* Hotels.com: ще одна популярна платформа для бронювання готелів та інших видів житла, яка пропонує широкий вибір місць по всьому світу.
* Agoda: спеціалізована платформа для бронювання готелів та інших видів житла, особливо популярна в регіонах Азії та Тихоокеанського регіону.

Ці сервіси здебільшого забезпечують користувачам зручність, широкий вибір об'єктів проживання, а також можливість перегляду відгуків та оцінок інших користувачів.

Для розробки програми буде використовуватись такий стек технологій

**Frontend (Користувацький інтерфейс):**

* WinForms: фреймворк для створення Windows-додатків з графічним інтерфейсом користувача.
* .NET 8.0: платформа для розробки програмного забезпечення для Windows.

**Мова програмування:**

C#: основна мова програмування для розробки програмних додатків на платформі .NET.

**Інструменти для розробки:**

Visual Studio: інтегроване середовище розробки (IDE) для програмування на мові C# та роботи з .NET.

**Бібліотеки та інші інструменти:**

* .NET Class Library: Використовується для створення та використання класів та компонентів у програмі.
* Інші бібліотеки .NET: Залежно від потреб проекту.

# ФУНКЦІОНАЛЬНІ ВИМОГИ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## Аналіз функціональності програмних аналогів

Проаналізуємо функціонал 2 аналогічних програмних систем:

Адреса: www.airbnb.com.ua

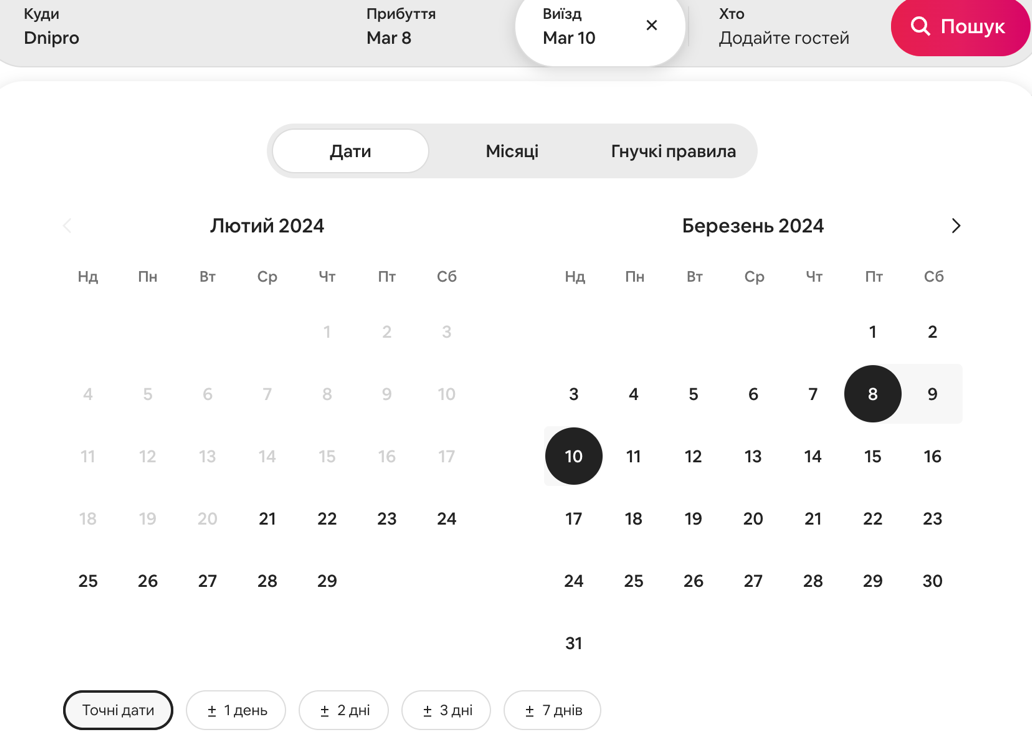


Рисунок 1 – вибір дати бронювання

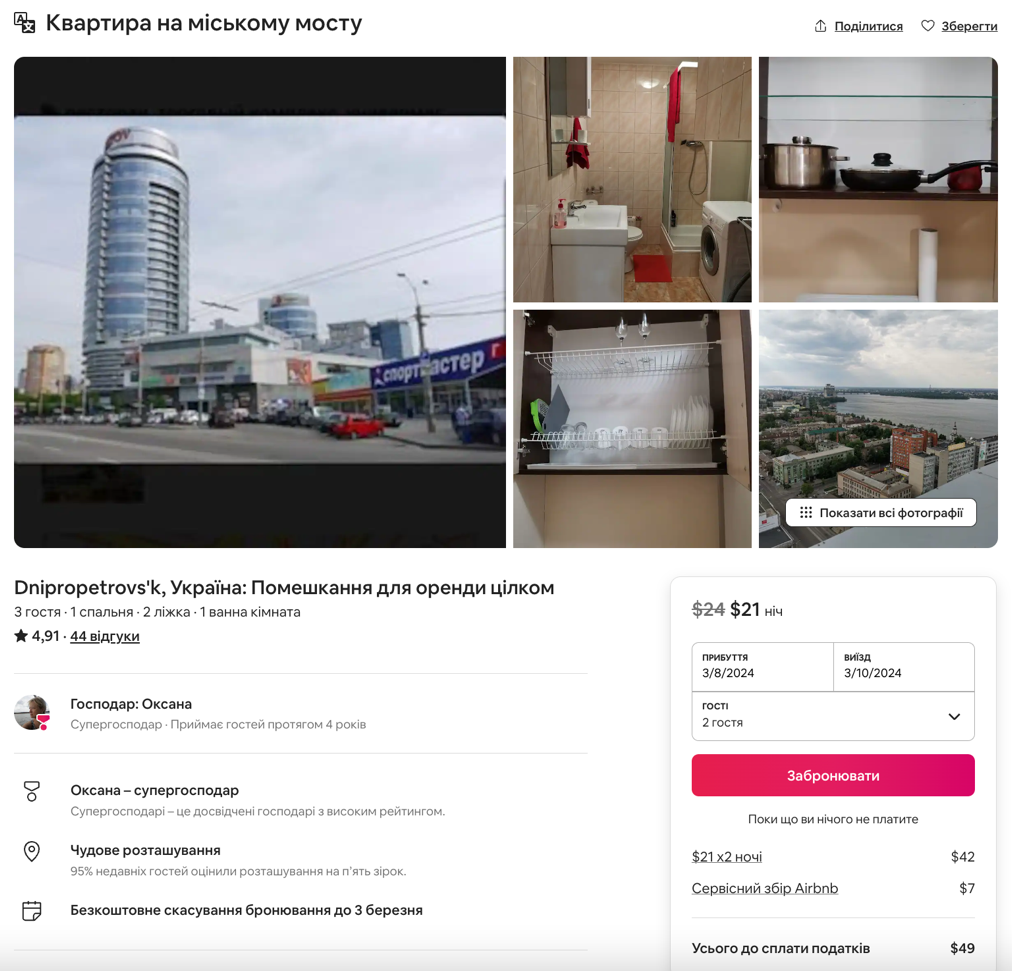


Рисунок 2 – перегляд картки житла та деталі оренди

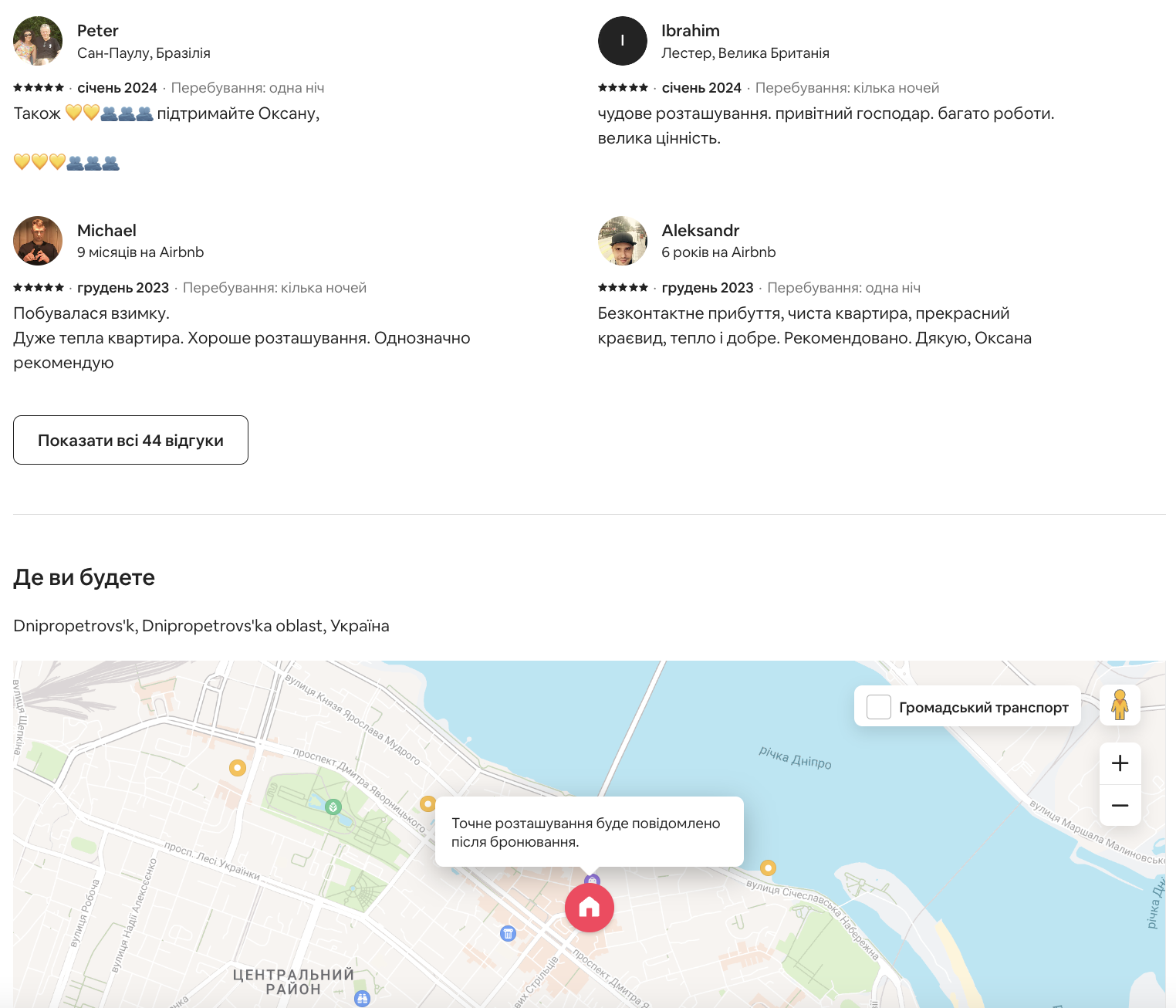


Рисунок 3 – перегляд відгуків про житло та точне місце розташування

Сервіс/система дозволяє: шукати тимчасове помешкання у різних куточках світу з гнучкими фільтрами пошуку, такими як: дата заселення та виселення, ціни, клас (готелю). Також можна легко дізнатись про деталі помешкання, що входить у вартість оренди, безпечне бронювання, чат з власником мешкання чи адміністратором готелю, точне місцерозташування на карті, відгуки інших гостей.

## Побудова Mind Map для заданої предметної області

Mind Map (або мапа думок/асоціативна мапа) дозволить систематизувати ту інформацію, яку Ви отримали на попередньому етапі.

Про те, що таке «Mind Map» можна більш детально прочитати за посиланням <https://www.mindmapping.com/mind-map>

Для побудови Mind Map є купа онлайн-інструментів, наприклад, відомий вам <https://www.lucidchart.com> або <https://www.mindmeister.com> та ін.

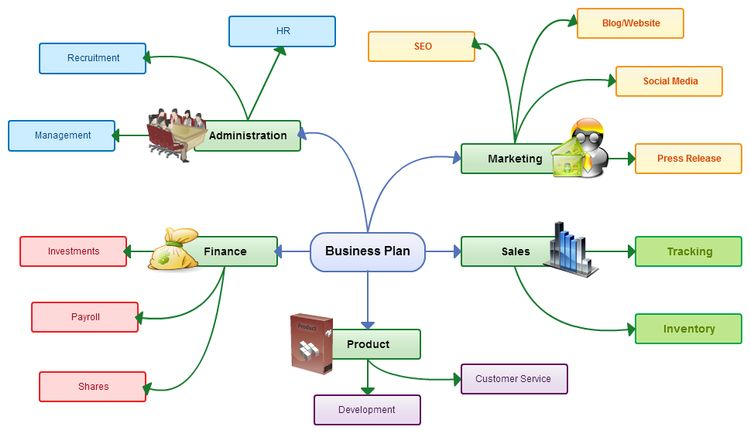


Рисунок 1.2 – Mind map ….

## Виділення ролей користувачів і формулювання функціональних вимог

Були виділені три ролі користувачів: «Власник картки з мешканням», «Зареєстрований користувач», «Гість».

Опишемо функціональні вимоги кожного типу користувачів.

Таблиця 1.1 – Функціональні вимоги користувача з роллю «Власник картки з мешканням»

|  |  |
| --- | --- |
| **Іденти-фікатор** | **Функціональні вимоги** |
| А.1. | Власник картки з мешканням може додати/редагувати/видалити свою картку для оренди житла.  ***Примітка:***   1. не можна редагувати картку, яку вже забронювали; 2. не можна видалити коментарі; 3. не можна підтвердити дату заселення на минулу дату; 4. … |
| А.2. | … |
| А.3. | … |
| … | … |

Таблиця 1.2 – Функціональні вимоги користувача з роллю «Зареєстрований користувач»

|  |  |
| --- | --- |
| **Іденти-фікатор** | **Функціональні вимоги** |
| ЗК.1. | Зареєстрований користувач може авторизуватися.  ***Примітка:***  не можна авторизуватися з невірним логіном і паролем. |
| ЗК.2. | Зареєстрований користувач може забронювати обране місце житла.  ***Примітка:***   1. не можна забронювати мешкань більше ніж є в наявності вільних; 2. не можна забронювати на минулу дату |
| ЗК.3. |  |
| … | … |

Таблиця 1.3 – Функціональні вимоги користувача з роллю «Гість»

|  |  |
| --- | --- |
| **Іденти-фікатор** | **Функціональні вимоги** |
| Г.1. | Гість може подивитися інформацію щодо наявних вільних місць на обрану дату.  ***Примітка:***  Не можна подивитися інформацію щодо наявних номерів на минулу дату.  Не може забронювати місце без наявного акаунту |
| Г.2. | Гість може зареєструватися, вказавши логін і пароль.  ***Примітка:***   1. логін не має збігатися з існуючими; 2. довжина логіну не менше 8 символів; 3. довжина паролю не менше 8 символів; 4. пароль має обов’язково містити будь які символи, без конкретного набору ; |
| … | … |

## Проєктування інтерфейсу користувача

Для реалізації замовником було обрано desktop-застосунок.

На основі сформульованих функціональних вимог були розроблені екранні форми, які наведені на рис. 0.3 – 0.5).

Навести розроблені макети екранних форм (треба буде пригадати такі курси, як «Проєктування інтерфейсу користувача», «Weв дизайн та HTML» ).

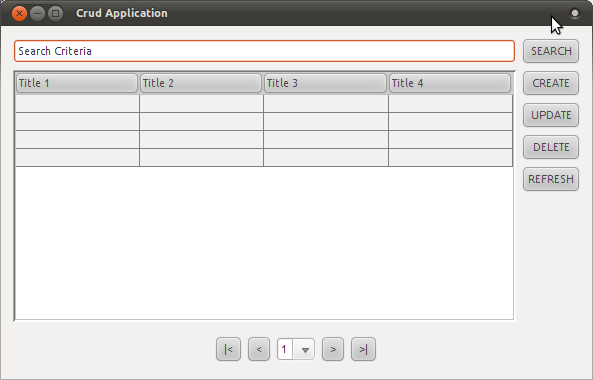


Рисунок 1.3 – Макет екранної форми ….

…

Рисунок 1.4 – Макет екранної форми ….

…

Рисунок 1.5 – Макет екранної форми ….

## Розроблення функціональних тестів

### Функціональні тести для ролі користувача «Власник картки з мешканням»

На основі опису предметної області, а також функціональних вимог і екранних форм були розроблені функціональні тести для ролі користувача «Власник картки з мешканням» (див. табл. 1.4).

Таблиця 1.4 – Функціональні тести для ролі користувача «Власник картки з мешканням»

|  |  |  |
| --- | --- | --- |
| **Іденти-фікатор тесту** | **Послідовність дій користувача** | **Очікуваний результат** |
| *А.1. Власник картки з мешканням може додати/редагувати/видалити сеанси в кінотеатрі.* | | |
| А.1.1 | 1. Ввести дату сеансу: «20.01.2023».  2. Ввести час сеансу: «10:00».  3. Обрати зал у кінотеатрі: «2».  4. Обрати фільм: «Аватар: шлях води».  5. Натиснути кнопку «Додати». | Успішне додавання сеансу |
| А.1.2 | 1.  2.  3.  … | Успішне редагування сеансу |
| А.1.3 | 1.  2.  3.  … | Успішне видалення сеансу |
| А.1.4 | 1.  2.  3.  … | Сеанс не був редагований. Користувач отримав повідомлення «Сеанс не може бути редагований! Сеанс відбувся!» |
| А.1.5 | 1.  2.  3.  … | Сеанс не був видалений. Користувач отримав повідомлення «Сеанс не може бути видалений! На сеанс куплені квитки!» |
| А.1.6 | 1.  2.  3.  … | Сеанс не був доданий. Користувач отримав повідомлення «Сеанс не може бути доданий! Вкажіть коректну дату!» |
| А.2 … | | |
| А.2.1 | … | … |
| А.2.2 | … | … |
| … | … | … |

### Функціональні тести для ролі користувача «Зареєстрований користувач»

На основі опису предметної області, а також функціональних вимог і екранних форм були розроблені функціональні тести для ролі користувача «Зареєстрований користувач» (див. табл. 1.5).

Таблиця 1.5 – Функціональні тести для ролі користувача «Зареєстрований користувач»

|  |  |  |
| --- | --- | --- |
| **Іденти-фікатор**  **тесту** | **Послідовність дій користувача** | **Очікуваний результат** |
| *ЗК.1. Зареєстрований користувач може авторизуватися.* | | |
| ЗК.1.1 | … | … |
| ЗК.1.2 | … | … |
| ЗК.1.3 | … | … |
| … | … | … |
| … | … | … |
| … | … | … |
| *ЗК.2 Зареєстрований користувач може купити квитки на обраний сеанс.* | | |
| ЗК.2.1 | … | … |
| ЗК.2.2 | … | … |
| ЗК.2.3 | … | … |

### Функціональні тести для ролі користувача «Гість»

На основі опису предметної області, а також функціональних вимог і екранних форм були розроблені функціональні тести для ролі користувача «Гість» (див. табл. 1.6).

Таблиця 1.6 – Функціональні тести для ролі користувача «Гість»

|  |  |  |
| --- | --- | --- |
| **Іденти-фікатор**  **тесту** | **Послідовність дій користувача** | **Очікуваний результат** |
| *Г.1. Гість може подивитися інформацію щодо наявних сеансів на обрану дату.* | | |
| Г.1.1 | … | … |
| Г.1.2 | … | … |
| Г.1.3 | … | … |
| … | … | … |
| … | … | … |
| … | … | … |
| *Г.2. Гість може зареєструватися, вказавши логін і пароль.* | | |
| Г.2.1 | … | … |
| Г.2.2 | … | … |
| Г.2.3 | … | … |
| … | … | … |

# МОДЕЛЮВАННЯ ПРЕДМЕТНОЇ ОБЛАСТІ

## Виділення і опис класів предметної області

**Клас «User» – користувач**

Таблиця 2.1 – Виділення характеристик класу «User»

|  |  |  |
| --- | --- | --- |
| **Назва характеристики** | **Тип значення** | **Обмеження** |
| Електронна пошта | Рядок | Не пустий рядок, наявність символу ‘@’ |
| Ім’я | Рядок | Не пустий рядок, довжина: хоча б 1 символ |
| Прізвище | Рядок | Не пустий рядок, довжина: хоча б 1 символ |
| Дата народження | Дата | > 01.01.1900 |
| Телефон | Рядок | Не пустий рядок, відповідність формату:  +38(0XX)-XXXXXXX |

Таблиця 2.2 – Виділення поведінки класу «User»

|  |  |  |  |
| --- | --- | --- | --- |
| **Назва поведінки** | **Опис вхідних параметрів** | **Тип значення, що повертається** | **Бізнес-правила** |
| Оформити купівлю білетів | Фільм,  дата,  час,  кількість білетів | true/false – замовлення оформлено / помилка при оформленні замовлення;  номер замовлення  (у випадку true) | 1. Перевірити чи дійсні вхідні дані (фільм, дата, час). 2. Перевірити наявність необхідної кількості білетів. 3. Якщо пройдені перевірки, то оформити замовлення. |
| Повернути куплені білети | Номер замовлення | true/false – повернення оформлено / помилка при оформленні повернення. | 1. Перевірити чи дійсний номер замовлення. 2. Перевірити чи не відбувся вже сеанс. 3. Якщо до сеансу залишилося більше 24 годин, то повернути повну суму грошей, інакше – 50%. |
| … | … | … | … |

**Клас «Movie» – фільм**

Таблиця 2.3 – Виділення характеристик класу «Movie»

|  |  |  |
| --- | --- | --- |
| **Назва характеристики** | **Тип значення** | **Обмеження** |
| Назва фільма | Рядок | Не пустий рядок, довжина: хоча б 1 символ |
| Тривалість | Ціле число | >0 |
| Режисер | Рядок | Не пустий рядок, довжина: хоча б 1 символ |
| Актори | Рядок | Не пустий рядок, довжина: хоча б 1 символ |
| Мова | Рядок | Не пустий рядок, довжина: хоча б 1 символ |
| Субтітри | Булевське значення |  |
| Дата початку прокату | Дата | Не може бути меншою ніж поточна дата |
| Дата початку прокату | Дата | Має бути більшою за дату початку прокату |

Таблиця 2.4 – Виділення поведінки класу «Movie»

|  |  |  |  |
| --- | --- | --- | --- |
| **Назва поведінки** | **Опис вхідних параметрів** | **Тип значення, що повертається** | **Примітка** |
| … | … | … | … |
| … | … | … | … |
| … | … | … | … |

Описати всі виділені класи предметної області.

## Встановлення зв’язків між класами

На рисунку 0.3 наведена діаграма класів, яка була побудова з використанням сервісу <https://www.lucidchart.com>.

Рисунок 2.1 – Діаграма класів …

Описати всі зв’язки між класами.

Приділити увагу абстрактним класам.

Зробити акцент на власних і .NET-інтерфейсах.

# ПРОГРАМНА РЕАЛІЗАЦІЯ КЛАСІВ ПРЕДМЕТНОЇ ОБЛАСТІ І ЇХ ТЕСТУВАННЯ

## Структура проєкту з реалізацією класів предметної області

Навести файлову/модульну структуру проєкту, яка містить реалізацію класів предметної області.

Рисунок 3.1 – Структура проєкту з реалізацією предметної області

## Реалізація інтерфейсів і каркасів класів предметної області

**Інтерфейс «….» – користувач**

Навести програмний код інтерфейсу.

**Клас «…» – зареєстрований користувач**

Навести програмний код каркасу класу.

**Клас «…» – Власник картки з мешканням**

Навести програмний код каркасу класу.

## Розроблення unit-тестів для класів предметної області

Розроблені unit-тести наведені у Додатку \_\_\_.

Таблиця 3.1 – Покриття unit-тестами класу «User»

|  |  |
| --- | --- |
| **Назва методу** | **Кількість розроблених unit-тестів** |
|  |  |
|  |  |
|  |  |
|  |  |

Таблиця 3.2 – Покриття unit-тестами класу «Movie»

|  |  |
| --- | --- |
| **Назва методу** | **Кількість розроблених unit-тестів** |
|  |  |
|  |  |
|  |  |
|  |  |

…

## Повна реалізація класів предметної області

У Додатку \_\_\_ наведено повну реалізацію програмного коду розроблених класів предметної області.

## Результати unit-тестування класів предметної області

На рисунках 0.5 – 0.7 наведено скриншоти проходження розроблених unit-тестів для класів предметної області.

…

Рисунок 3.2 – Скриншот проходження unit-тестів для класу \_\_\_\_\_\_\_

…

Рисунок 3.3 – Скриншот проходження unit-тестів для класу \_\_\_\_\_\_\_

…

Рисунок 3.4 – Скриншот проходження unit-тестів для класу \_\_\_\_\_\_\_

# ПРОЄКТУВАННЯ І ПРОГРАМНА РЕАЛІЗАЦІЯ КЛАСІВ ІНТЕРФЕЙСУ КОРИСТУВАЧА

## Структура проєкту з реалізацією класів інтерфейсу користувача

Навести файлову/модульну структуру проєкту, яка містить реалізацію класів інтерфейсу користувача.

Рисунок 4.1 – Структура проєкту з інтерфейсом користувача

## Виділення класів для реалізації інтерфейсу користувача

Описати виділені характеристики і методи для класів інтерфейсу користувача.

…

## Програмна реалізація класів інтерфейсу користувача

У Додатку \_\_\_ наведено повну реалізацію програмного коду розроблених класів інтерфейсу користувача.

# ФУНКЦІОНАЛЬНЕ ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## Функціональне тестування для ролі користувача «Власник картки з мешканням»

Для проведення функціонального тестування для ролі «Власник картки з мешканням» були використані розроблені у п. 1.4.1. функціональні тести. Результати проведення функціонального тестування наведені нижче.

**Ідентифікатор тесту:** А.1.1.

**Вхідні дані:**

1. Ввести дату сеансу: «20.01.2023».

2. Ввести час сеансу: «10:00».

3. Обрати зал у кінотеатрі: «2».

4. Обрати фільм: «Аватар: шлях води».

5. Натиснути кнопку «Додати».

**Очікуваний результат:** Успішне додавання сеансу

**Отриманий результат:**

Рисунок 5.1 – Скриншот виконання тесту А.1.1

**Ідентифікатор тесту:**

**Вхідні дані:**

**Очікуваний результат:**

**Отриманий результат:**

Рисунок 5.2 – Скриншот виконання тесту \_\_\_\_\_

## Функціональне тестування для ролі користувача «Зареєстрований користувач»

Для проведення функціонального тестування для ролі «Зареєстрований користувач» були використані розроблені у п. 1.4.2. функціональні тести. Результати проведення функціонального тестування наведені нижче.

**Ідентифікатор тесту:**

**Вхідні дані:**

**Очікуваний результат:**

**Отриманий результат:**

Рисунок 5.3 – Скриншот виконання тесту \_\_\_\_\_

## Функціональне тестування для ролі користувача «Гість»

Для проведення функціонального тестування для ролі «Гість» були використані розроблені у п. 1.4.3. функціональні тести. Результати проведення функціонального тестування наведені нижче.

**Ідентифікатор тесту:**

**Вхідні дані:**

**Очікуваний результат:**

**Отриманий результат:**

Рисунок 5.4 – Скриншот виконання тесту \_\_\_\_\_

# ВИСНОВКИ

У результаті виконання курсової роботи було розроблено програмне забезпечення з використанням об’єктно-орієнтованої парадигми для вказати предметну область.

В ході роботи було…. (коротко описати що саме було зроблено під час виконання курсової роботи).

…

….

**Примітка:**

Посилання на github-репозиторій: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Посилання на відеоролик: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# ПЕРЕЛІК ДЖЕРЕЛ ТА ПОСИЛАНЬ

1. Воробйов, Ю. А. Правила оформлення навчальних і науково-дослідних документів [Текст] : навч. посіб. / Ю. А. Воробйов, Ю. О. Сисоєв. – 4-те вид., випр. і доп. – Харків : Нац. аерокосм. ун-т ім. М. Є. Жуковського «Харків. авіац. ін-т», 2019. – 88 с.
2. Автоматичне оформлення джерел по ВАК України [Електронний ресурс] – Режим доступу до ресурсу: <http://vak.in.ua/do.php>.
3. Назва [Електронний ресурс] – Режим доступу до ресурсу: http://посилання.
4. …

На всі наведені у цьому переліку публікації у тексті пояснювальної записці мають бути посилання.

# ДОДАТОК А. Лістинг класів предметної області

Власноруч створений код програми.

# ДОДАТОК Б. Лістинг класів інтерфейсу користувача

Власноруч створений код програми. Код, який було автоматично створено середовищем розробки, додавати не потрібно.

# ДОДАТОК В. Лістинг класів unit-тестів

Власноруч створений код програми.

# ДОДАТОК Г. Назва додатку

…