Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Кафедра Програмної інженерії

КУРСОВА РОБОТА

ПОЯСНЮВАЛЬНА ЗАПИСКА

з дисципліни «Об’єктно-орієнтоване програмування»

АДМІНІСТРАТОР ГОТЕЛЮ

Керівник , професор Бондарєв В.М.

Студент гр. ПЗПІ-19-4 Боголідова А.А.

Комісія:

ст. викл. \_\_\_\_\_\_\_\_\_\_\_\_\_ Черепанова Ю.Ю.

проф. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Бондарєв В.М.

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

Харків 2020**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***

***Боголідової Аліни Андріївни***

(Прізвище, Ім'я, По батькові)

1 Тема проекту: *Адміністратор готелю*

2 Термін здачі студентом закінченого проекту: ***“19” - червня - 2020 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:

*Діаграма класів,* *схема об’єктної моделі, алгоритми, приклади екранних форм

**КАЛЕНДАРНИЙ ПЛАН**

|  |  |  |
| --- | --- | --- |
| *№* | *Назва етапу* | *Термін виконання* |
| 1 | Видача теми, узгодження і затвердження теми | 21-02-2020 р. |
| 2 | Формулювання вимог до програми | 21-02-2020 – 31-03-2020 р. |
| 3 | Розробка підсистеми роботи з номерами готелю. | 01-04-2020 – 08-04-2020 р. |
| 4 | Розробка підсистеми взаємодії із записами реєстрацій (поселення/виселення постояльців). | 08-04-2020 – 23-04-2020 р. |
| 5 | Розробка підсистеми пошуку постояльців. | 23-04-2020 – 27-04-2020 р. |
| 6 | Встановлення різних рівнів доступу до функціоналу шляхом створення двох типів користувачів – адміністратора та гостя. | 27-04-2020 – 31-04-2020 р. |
| 7 | Розробка функцій зберігання та завантаження даних | 31-04-2020 – 04-05-2020 р. |
| 8 | Тестування і доопрацювання розробленої програмної системи. | 04-05-2020 – 20-05-2020 р. |
| 9 | Оформлення пояснювальної записки, додатків, графічного матеріалу | 20-05-2020 – 05-06-2020 р. |
| 10 | Захист | 01-06-2020 – 19-06-2020 р. |

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Боголідова Аліна Андріївна

Керівник \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Бондарєв Володимир Михайлович

(Прізвище, Ім'я, По батькові)

« 21 »\_лютого\_\_\_\_\_\_\_\_\_\_ 2020 р.

**РЕФЕРАТ**

Пояснювальна записка до курсової роботи: 61 с., 28 рис., 1 табл., 5 додатків, 5 джерел.

АДМІНІСТРАТОР, ВИСЕЛЕННЯ, ВІДГУК, ВІДОМІСТЬ, ГОТЕЛЬ, КВИТАНЦІЯ, КЛАС, МОВА ПРОГРАМУВАННЯ C#, НОМЕР, ОБ’ЕКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ, ПЛАТФОРМА .NET, ПОСЕЛЕННЯ, ПРОГРАМА, РЕЄСТРАЦІЯ.

Метою роботи є розробка програми «Адміністратор готелю» на засадах об’єктно-орієнтованого програмування для обслуговування готелю адміністратором.

Методи розробки базуються на використанні середи розробки Microsoft Visual Studio 2019, Windows Forms, платформи .NET Framework 4.7.2, мови програмування C#.

У результаті отримана програма під назвою «Адміністратор готелю», яка дозволяє звичайному користувачу переглядати відгуки про готель, додавати власний відгук. Авторизований користувач (адміністратор) має доступ перегляду відгуків, а також може заселяти та виселяти постояльців за декількома сценаріями, видаючи квитанції, редагувати дані постояльців, здійснювати роботу з номерами: додавати нові номери, редагувати та видаляти їх. Адміністратору також доступний пошук серед постояльців за критеріями прізвища, імені та дати народження.

**ЗМІСТ**

|  |  |
| --- | --- |
| Вступ……………………………………………………………..……………. | 7 |
| 1 Специфікація програми…………….…………………………..…………... | 8 |
| 1.1. Користувачі програми……………..…………………..…………….. | 8 |
| 1.2. Функції програми................................................................................. | 8 |
| 1.3. Функції адміністратора……..………………………..……………… | 9 |
| 1.4. Функції гостя…………………………………………..……………... | 19 |
| 2 Проектна специфікація………………..……………………..……………... | 22 |
| 2.1. Архітектура проекту……..…………………………..………………. | 22 |
| 2.2. Структура проекту…………………………………..……………….. | 22 |
| 2.3. Об’єктна модель………………..……………………………………... | 24 |
| 2.3.1 Клас Admin………………….…………………………………... | 24 |
| 2.3.2 Клас Guest………………………………...................................... | 25 |
| 2.3.3 Клас Hotel………………………………….................................. | 25 |
| 2.3.4 Клас RegRecord…………………………………..……………... | 27 |
| 2.3.5 Клас Resident……………………………………......................... | 28 |
| 2.3.6 Клас Review………………………………………..………......... | 29 |
| 2.3.7 Клас Room…………………………………........................……. | 29 |
| 2.3.8 Клас Dao……………………………………..………………...... | 29 |
| 3 Інструкція користувача……………………………….……………………. | 30 |
| 3.1 Встановлення, запуск і вікно входу…………………………………. | 30 |
| 3.2 Функціонал адміністратора…………………………………………... | 32 |
| 3.2.1 Поселення постояльців…………………………………………. | 33 |
| 3.2.2 Редагування даних постояльця……………………………..….. | 35 |
| 3.2.3 Виселення постояльця………………………………….……….  3.2.4 Додавання номеру………………………………………………. | 36 |
| 3.2.5 Редагування номеру…………………………………………….. | 39 |
| 3.2.6 Видалення номеру………………………………………………. | 39 |
| 3.2.7 Пошук серед постояльців………………………………………. | 40 |
| 3.2.8 Перегляд відгуків……………………………………………….. | 40 |
| 3.3 Функціонал гостя……………………………………………………… | 41 |
| Висновки………………………………………………………..……………... | 44 |
| Перелік джерел посилання……………………………………………………. | 45 |
| Додаток А Методи класу Hotel………………………………………………. | 46 |
| Додаток Б Конструктор і методи класу RegRecord…………………………. | 53 |
| Додаток В Клас Room………………………………………………………… | 55 |
| Додаток Г Клас Dao…………………………………………………………… | 58 |
| Додаток Д Таблиця доступних для додавання номерів…………………….. | 61 |

37

**ВСТУП**

Функціонування готелю неможливе без чіткого керування адміністратором, який забезпечує взаємодію між постояльцями та готелем як підприємством. Його задача – організувати процеси поселення та виселення, нагляду за номерами так, щоб гість залишився задоволеним послугами готелю.

Повний доступ до функцій програми отримує авторизований користувач – адміністратор, який може здійснювати поселення та виселення постояльців, оформлювати квитанції, додавати, редагувати та видаляти номери, здійснювати пошук серед постояльців, а також переглядати відгуки про готель, аби впевнитися у якості своєї роботи.

Користувач типу гість, який персоніфікується своїми ім’ям, датами приїзду та від’їзду, може писати відгуки про готель і переглядати відгуки попередніх постояльців.

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

Задачі роботи:

* створення логіки роботи системи;
* проектування графічного інтерфейсу користувача;
* розробка логіки додавання, редагування, пошуку та видалення даних;
* розробка системи завантаження та зберігання даних;
* тестування готового продукту;
* створення стійкості програми.

**1 СПЕЦИФІКАЦІЯ ПРОГРАМИ**

**1.1 Користувачі програми**

У проекті присутні дві категорії користувачів, які мають різні функції, – це адміністратор і гість, при цьому проект зорієнтований більшою мірою на адміністратора.

Належність користувача до гостя або адміністратора визначається вибором відповідної програми.

Гостем у проекті вважається людина, яка вже скористалася послугами готелю, і хоче шляхом написання відгуку оцінити його роботу. Гість персоніфікується при вході у програму, зазначаючи свої прізвище-ім’я, дати приїзду та від’їзду, залишає відгук про свій період перебування в готелі, а також переглядає відгуки інших гостей.

Адміністратор авторизується при вході у програму, вводячи набір ім’я-пароль з вже існуючих даних. Адміністратор підтримує список номерів: додає нові, редагує та видаляє існуючі, здійснює поселення та виселення (у поточний день, раніше або пізніше) постояльців, створює квитанції. Сортує постояльців за тим, чи виселяються вони сьогодні, здійснює пошук серед постояльців за критеріями імені, прізвища та дати народження. Також адміністратор переглядає відгуки гостей.

Питання такі, як захист паролів та безпека даних, у даній роботі не розглядаються.

**1.2 Функції програми**

Програма надає гостю наступні можливості:

1. Вхід у програму;
2. Написання відгуку;
3. Перегляд відгуків;
4. Вихід із програми.

Програма надає адміністратору наступні можливості:

1. Вхід у програму;
2. Поселення постояльців;
3. Організація від’їзду постояльців;
4. Робота із записами реєстрацій;
5. Робота зі списком номерів;
6. Пошук постояльців;
7. Перегляд відгуків;
8. Вихід із програми.

Загальні функції:

1. Завантаження даних з постійної пам’яті в оперативну;
2. Збереження поточного стану даних у постійній пам’яті;
3. Генерація тестових даних.

**1.3 Функції адміністратора**

*Функція «Вхід у програму»*

Адміністратор входить у програму за допомогою одного з наборів ім’я – пароль. Імена та паролі адміністраторів зберігається в даних програми. Усі інші функції адміністратора стають доступними одразу після правильного введення імені та пароля.

Основний сценарій

1. Після запуску програми відкривається вікно входу адміністратора (рис. 1.1).
2. Адміністратор вводить ім’я і пароль і натискає на кнопку «Увійти».
3. Введені дані перевіряються, і відкривається головне вікно адміністратора (рис. 1.2).
4. Якщо дані були введені некоректно, на формі вводу з’являється повідомлення про це, і користувач намагається увійти знову. Кількість спроб необмежена.

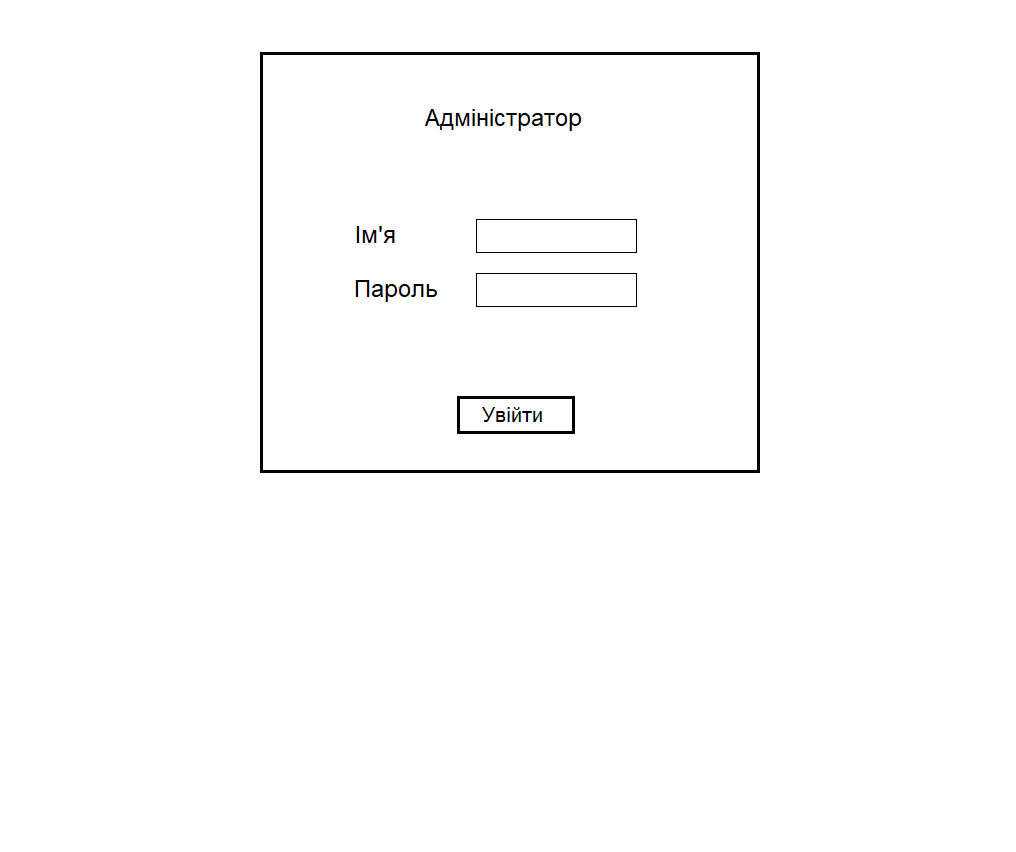


Рисунок 1.1 – Вікно входу адміністратора

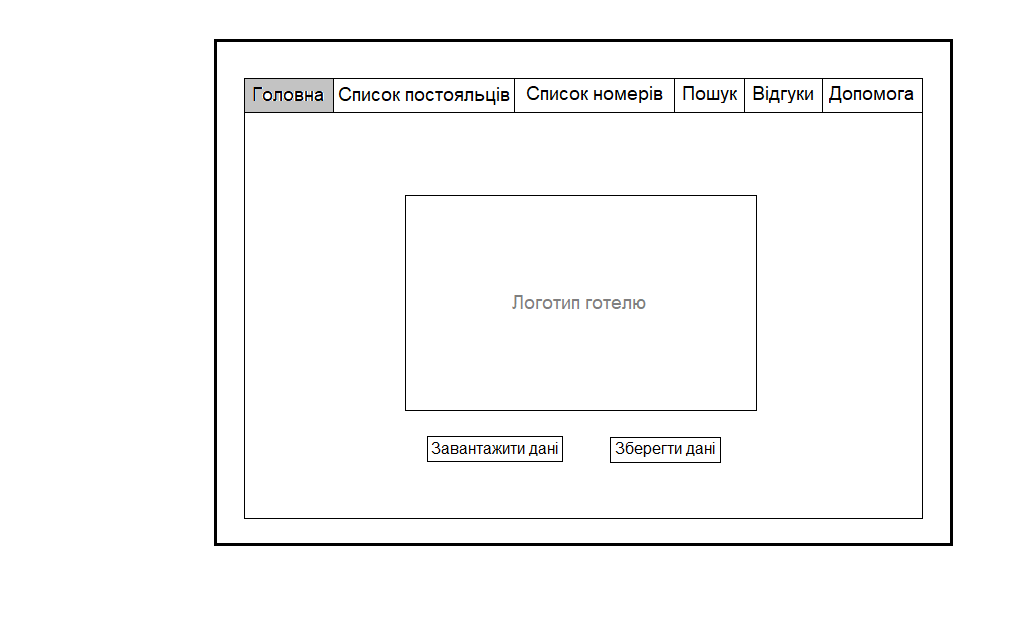


Рисунок 1.2 – Панель адміністратора

*Функція «Поселення постояльців»*

Щоб готель працював, необхідно мати постояльців. Тому в першу чергу адміністратор здійснює поселення нових постояльців, яке можливе за наявності вільних номерів у готелі. Воно складається з обрання адміністратором відповідного номера та реєстрування даних, після заповнення створюється новий запис реєстрації, а також друкується квитанція. У запис реєстрації входить: прізвище, ім’я, дата народження, стать, номер телефону, електронна пошта, дані про номер, дати приїзду та від’їзду, сума до сплати.

Сценарій «Новий постоялець»

1. На панелі адміністратора користувач переходить на вкладку «Список постояльців» (рис. 1.3).
2. Натискає на кнопку «Заселити постояльця». Якщо у готелі є «незайняті» номери, то відкривається форма реєстрації нового постояльця (рис. 1.4). Поля форми порожні.
3. Заповнює поля, обирає відповідні номер і дати перебування.
4. Натискає на кнопку «Квитанція». Створюється новий запис реєстрації, стан зайнятості номера змінюється на «зайнятий». Виконання сценарію «Оформлення квитанції».
5. Натискає на кнопку «Зберегти», після чого користувач повертається на вкладку «Список постояльців».

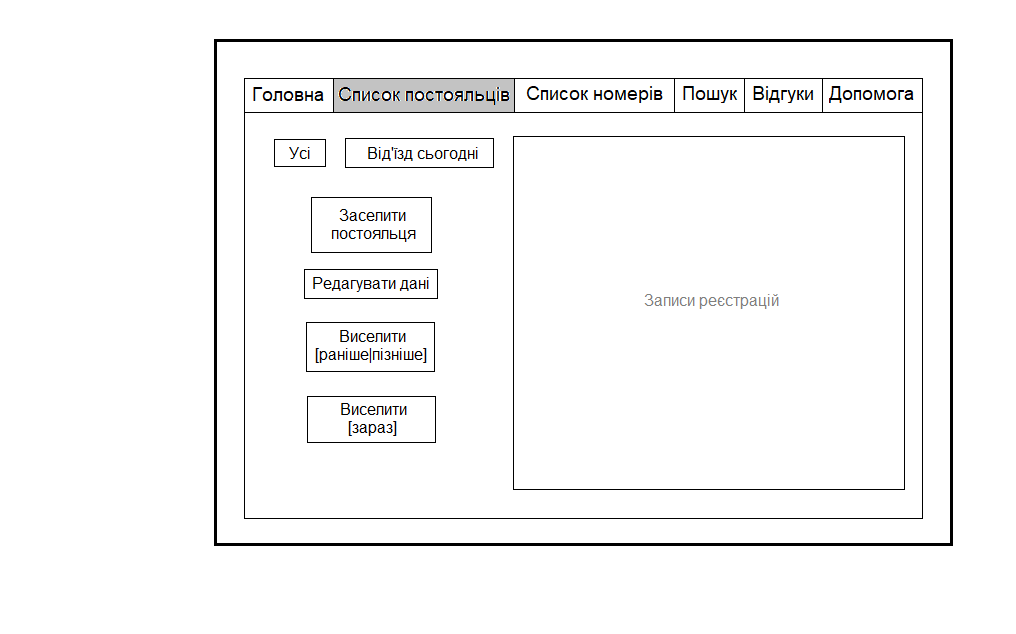


Рисунок 1.3 – Вкладка списку постояльців

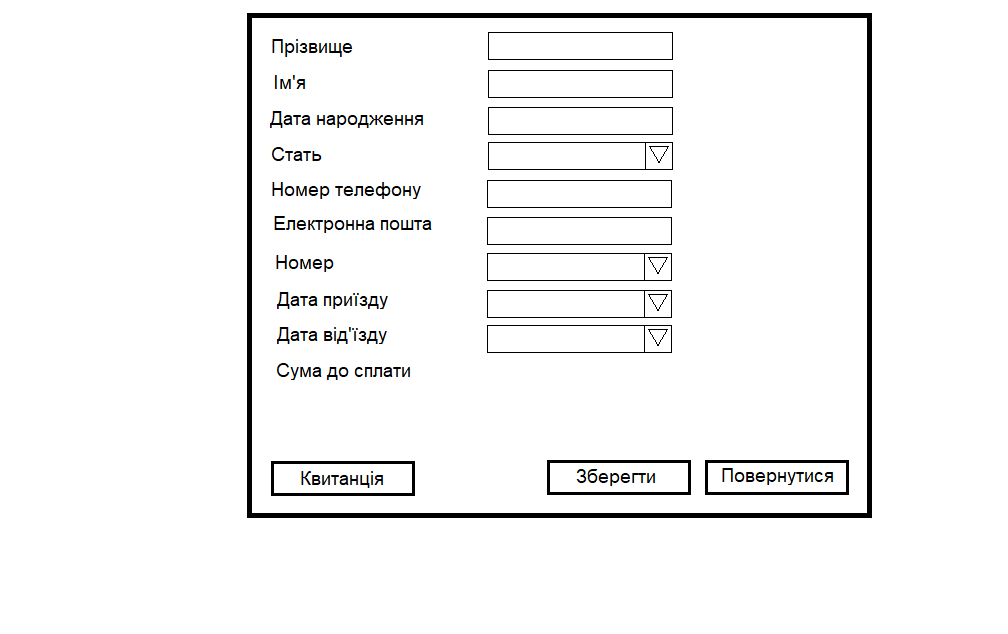


Рисунок 1.4 – Форма запису реєстрації

Сценарій «Оформлення квитанції»

1. У формі запису реєстрації користувач натискає на кнопку «Квитанція».
2. Відкривається нова форма квитанції (рис. 1.5), в якій виводяться дані з запису реєстрації нового постояльця та підрахована сума за номер. Сума рахується: різниця у днях між датами приїзду та від’їзду помножити на ціну номера за ніч.
3. Натискає на кнопку «Роздрукувати». Квитанція зберігається у виді текстового файлу receipt\_(прізвище постояльця).txt у каталозі проекту.

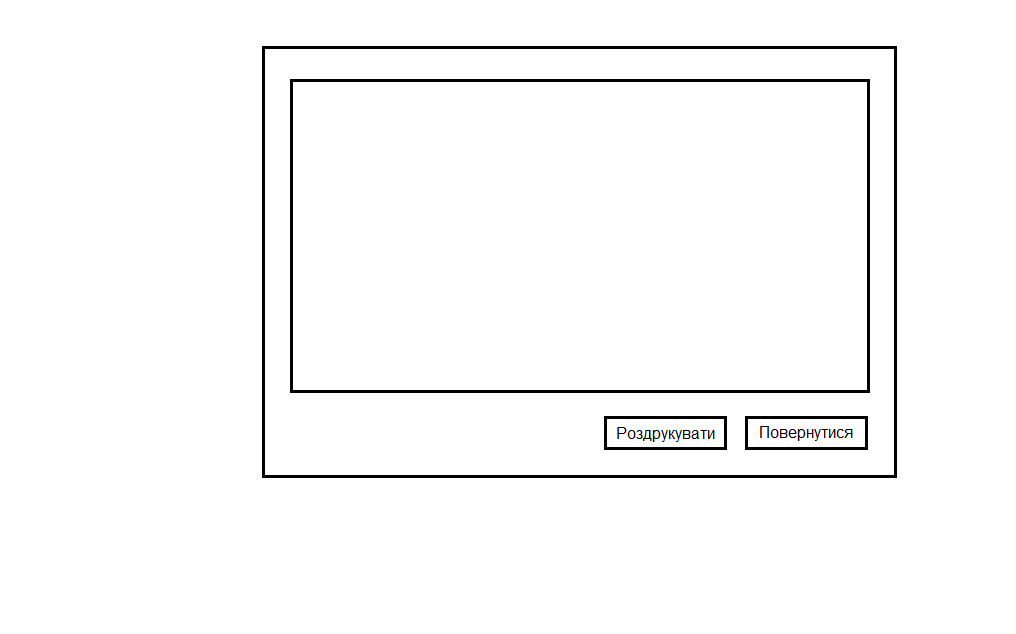


Рисунок 1.5 – Форма квитанції

*Функція «Організація від’їзду постояльців»*

Окрім заселення постояльців, адміністратор може виселяти їх за декількома сценаріями від’їзду. Від’їзд: вибір всіх постояльців, від’їжджаючих сьогодні, звільнення місця; достроковий від’їзд або оформлення затримки з оформленням квитанції-перерахунку.

Сценарій «Звільнення місця»

1. На вкладці «Список постояльців» користувач натискає на кнопку «Від’їзд сьогодні».
2. Записи реєстрацій оновлюються: у списку залишаються дані лише про тих постояльців, які від’їжджають сьогодні. Обирає постояльця, натискає на кнопку «Виселити[зараз]».
3. Запитується підтвердження на видалення запису реєстрації.
4. Після підтвердження видалення, список оновлюється. Стан зайнятості номера змінюється на «вільний».
5. Обраним стає попередній запис реєстрації.

Сценарій «Зміна від’їзду з випискою квитанції-перерахунку»

1. На вкладці «Список постояльців» користувач обирає запис реєстрації постояльця. Натискає на кнопку «Виселити[раніше|пізніше]».
2. Відкривається форма виселення постояльця (рис. 1.6). Поля форми заповнені.
3. Змінює поле дати від’їзду.
4. Натискає на кнопку «Квитанція». У відповідний запис реєстрації вносяться зміни. Виконання сценарію «Оформлення квитанції-перерахунку».
5. Натискає на кнопку «Зберегти». Повертається на вкладку «Список постояльців».

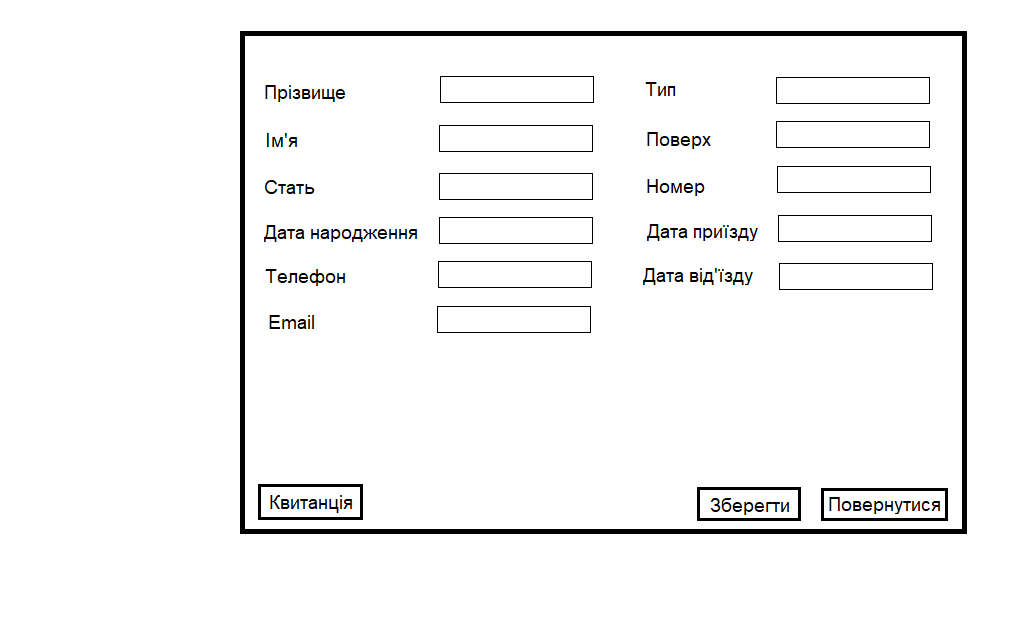


Рисунок 1.6 – Форма виселення постояльця

Сценарій «Оформлення квитанції-перерахунку»

1. У формі виселення постояльця користувач натискає на кнопку «Квитанція».
2. Відкривається нова форма квитанції (рис. 1.5).
3. Натискає на кнопку «Роздрукувати». Квитанція зберігається у виді текстового файлу recalculation\_(прізвище постояльця).txt у каталозі проекту.

*Функція «Робота із записами реєстрацій»*

Дана функція реалізує той випадок, коли під час заповнення даних про постояльця сталася помилка або необхідно переселити постояльця у інший номер.

Сценарій «Зміна даних постояльця»

1. На вкладці «Список постояльців» користувач обирає запис реєстрації постояльця.
2. Натискає на кнопку «Редагувати дані».
3. Відкривається форма запису реєстрації (рис. 1.3) обраного постояльця. Поля форми заповнені.
4. Змінює поля форми.
5. Натискає на кнопку «Квитанція». Виконання сценарію «Оформлення квитанції».
6. Натискає на кнопку «Зберегти». Повертається на вкладку «Список постояльців».

*Функція «Робота зі списком номерів»*

Окрім даних про постояльців готелю, адміністратор керує даними про номери, при цьому розміри готелю фіксовані: 6 поверхів по 20 номерів на кожному. Користувач може здійснювати перегляд списку номерів, додавання, редагування та видалення номерів за умови їх «незайнятості».

Сценарій «Додавання номеру»

1. Користувач переходить на вкладку «Список номерів» (рис. 1.7). Натискає на кнопку «Додати номер».
2. Відкривається форма номеру (рис. 1.8). Поля форми порожні.
3. Заповнює дані номеру, завантажує файл зображення.
4. Натискає кнопку «Зберегти». Список оновлюється.

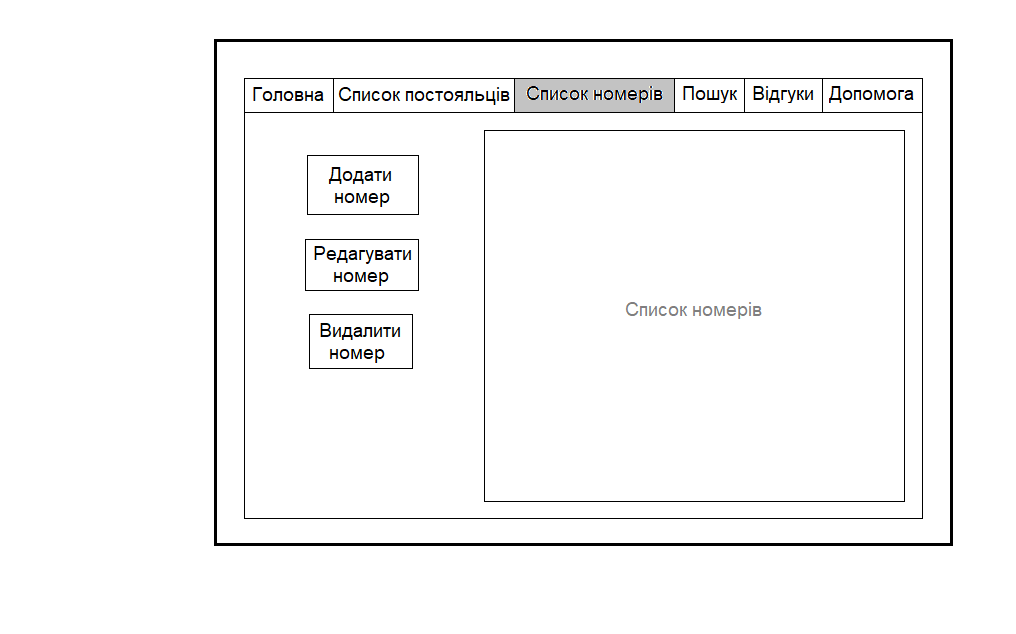


Рисунок 1.7 – Вкладка списку номерів

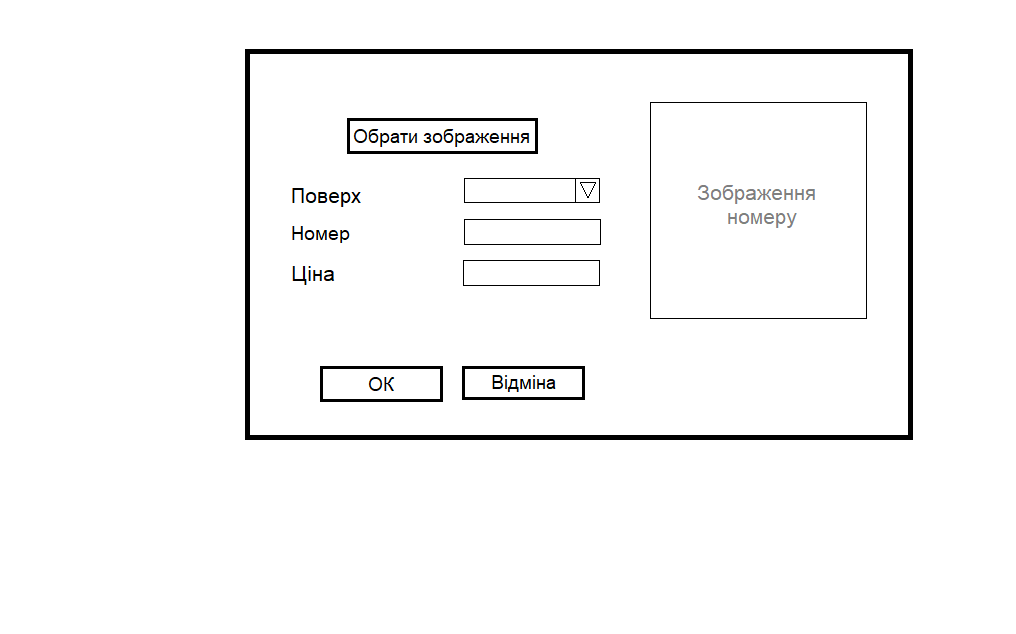


Рисунок 1.8 – Форма номеру

Сценарій «Редагування номеру»

1. На вкладці «Список номерів» користувач обирає номер.
2. Натискає на кнопку «Редагувати номер».
3. Якщо статус номеру «незайнятий», то відкривається форма номеру. Поля форми заповнені.
4. Змінює дані номеру, змінює файл зображення.
5. Натискає кнопку «Зберегти».

Сценарій «Видалення номеру»

1. На вкладці «Список номерів» користувач обирає номер.
2. Натискає на кнопку «Видалити номер». Якщо стан зайнятості номера «вільний», то запитується підтвердження на видалення номера. Якщо стан зайнятості номера «зайнятий», то з’являється вікно з повідомлення про неможливість видалення.
3. Після підтвердження видалення, список оновлюється.
4. Обраним стає наступний номер.

*Функція «Пошук постояльців»*

Дана функція реалізує пошук постояльців за одним з критеріїв: прізвище, ім’я або дата народження, для швидкого доступу до даних необхідного постояльця.

Основний сценарій

1. На панелі адміністратора користувач натискає на кнопку «Пошук постояльців».
2. Відкривається форма пошуку (рис. 1.8). Поля форми незаповнені.
3. Обирає критерій для пошуку. Вводить дані для пошуку. Натискає на кнопку із зображенням лупи.
4. У окремому полі з’являються дані про гостей, з якими співпали введені адміністратором дані для пошуку.

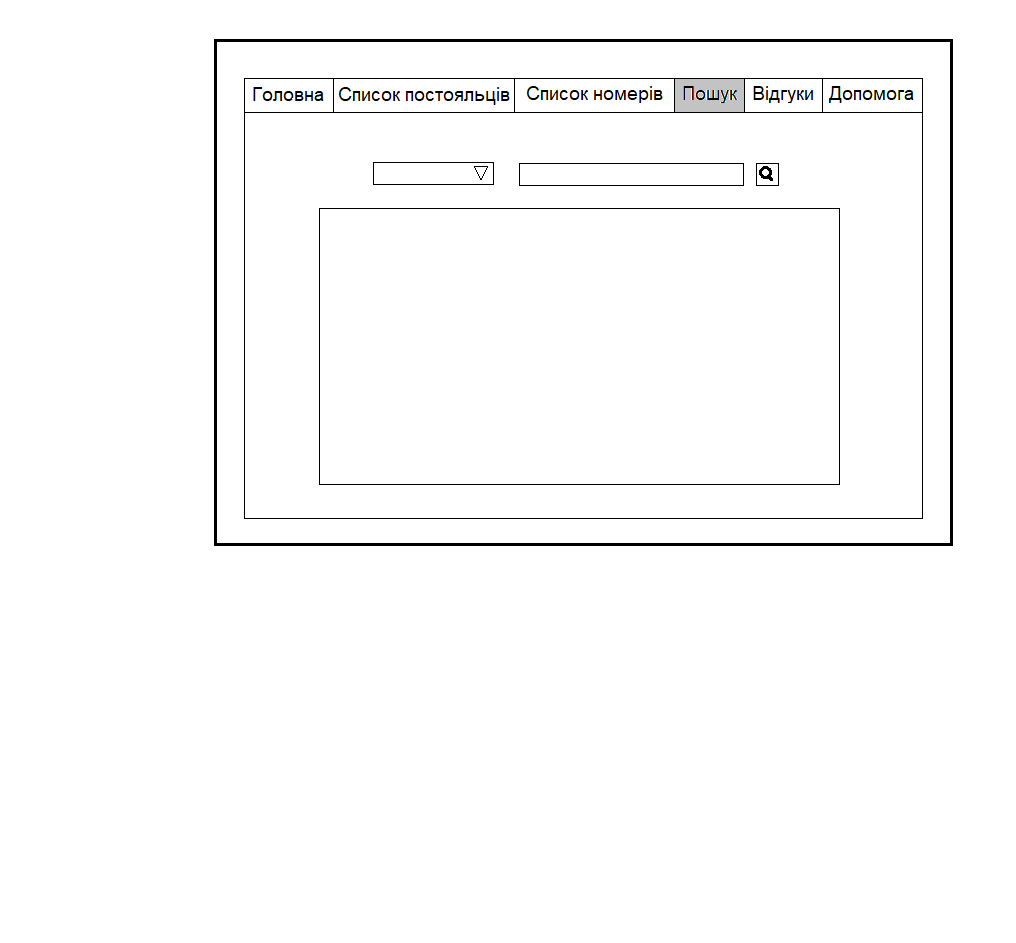


Рисунок 1.8 – Вкладка пошуку

*Функція «Перегляд відгуків»*

Щоб об’єктивно оцінювати роботу готелю, потрібно знати відгуки гостей, тому адміністратор має змогу переглядати відгуки, проте не редагувати.

Основний сценарій

1. На панелі адміністратора користувач переходить на вкладку «Відгуки» (рис. 1.9).
2. Ознайомлюється із записами.

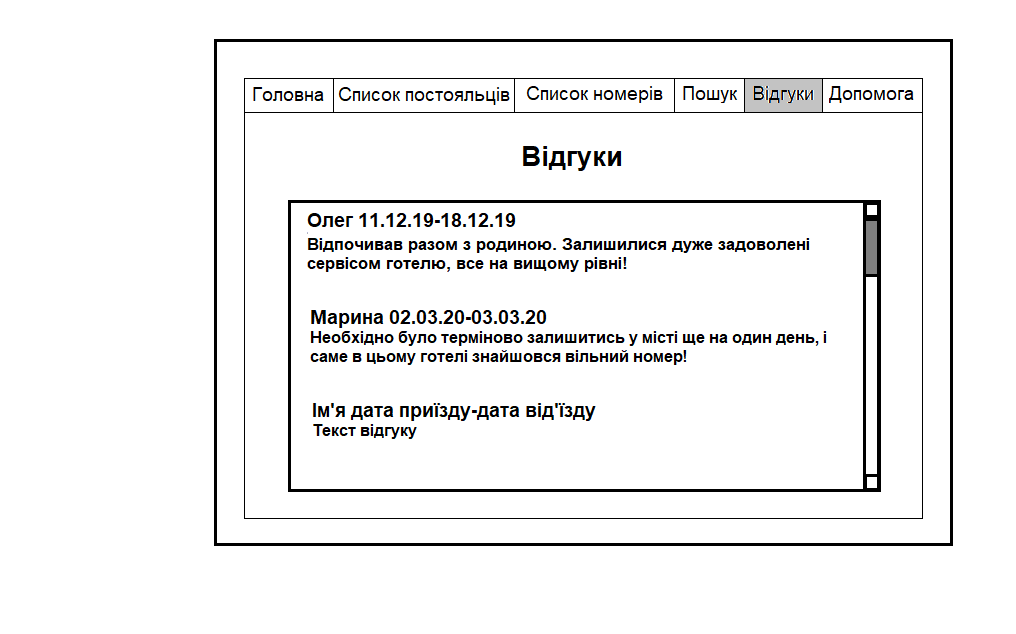


Рисунок 1.9 – Вкладка відгуків

*Функція «Вихід із програми»*

У процесі роботи користувач вносить зміни до даних, які знаходяться в оперативній пам’яті. Якщо дані були змінені, але не збережені у постійній пам’яті, то перед закриттям програми користувач отримує повідомлення з пропозицією зберегти або не зберігати дані.

**1.4 Функції гостя**

*Функція «Вхід у програму»*

Вважається, що гість – людина, яка вже користувалася послугами готелю та бажає залишити відгук або переглянути відгуки інших гостей. Система відгуків функціонує як книга відгуків. Гість запускає відповідну програму і входить за допомогою імені, дат приїзду та від’їзду.

Основний сценарій

1. Гість запускає відповідну програму і відкривається вікно входу гостя (рис. 1.10).
2. Гість вводить ім’я, дати приїзду та від’їзду і натискає кнопку «Увійти».
3. Введені дані зберігаються у програмі, і відкривається гостьова панель – головне вікно гостя (рис. 1.11).

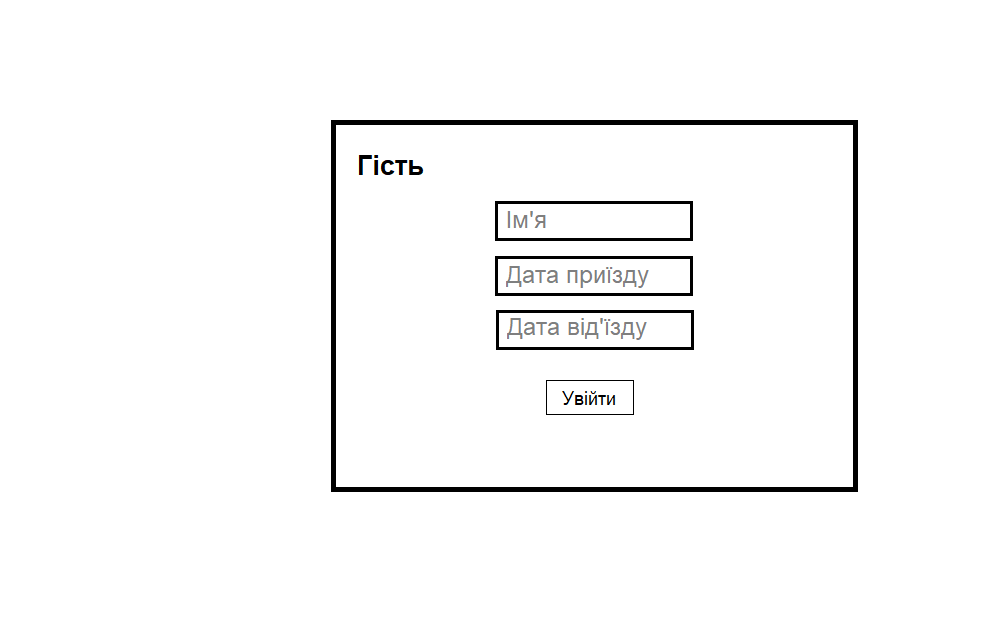


Рисунок 1.10 – Вікно входу гостя

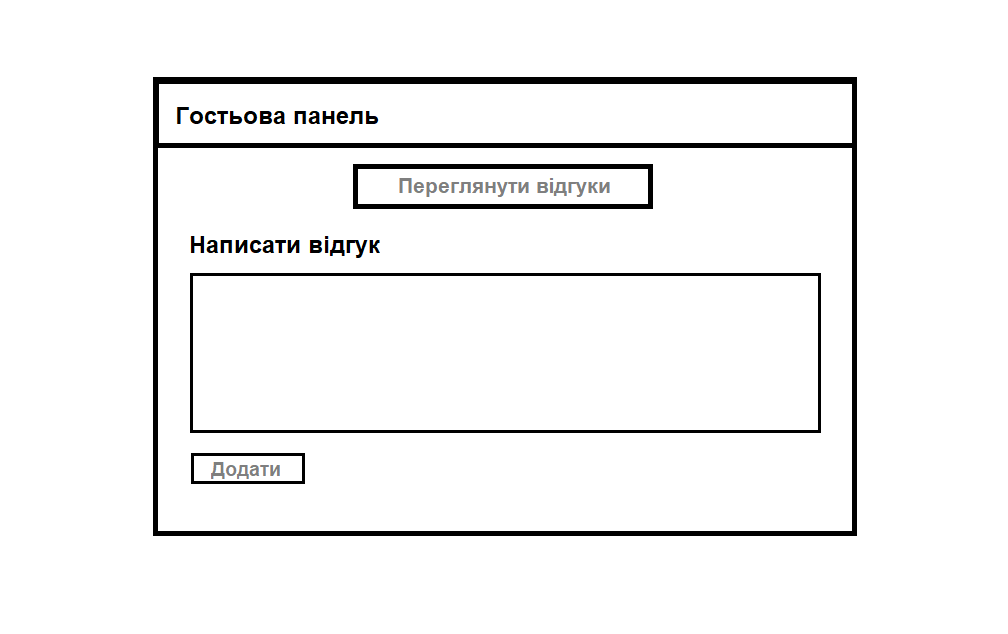


Рисунок 1.11 – Гостьова панель

*Функція «Написання відгуку»*

Основний сценарій

1. На гостьовій панелі у відповідному полі гість пише свій відгук.
2. Натискає на кнопку «Додати». Дані вносяться в систему.
3. Поле для написання відгуку очищується.

*Функція «Перегляд відгуків»*

Основний сценарій

1. На гостьовій панелі гість натискає на кнопку «Переглянути відгуки».
2. Відкривається форма з відгуками (рис. 1.12).
3. Гість переглядає відгуки.

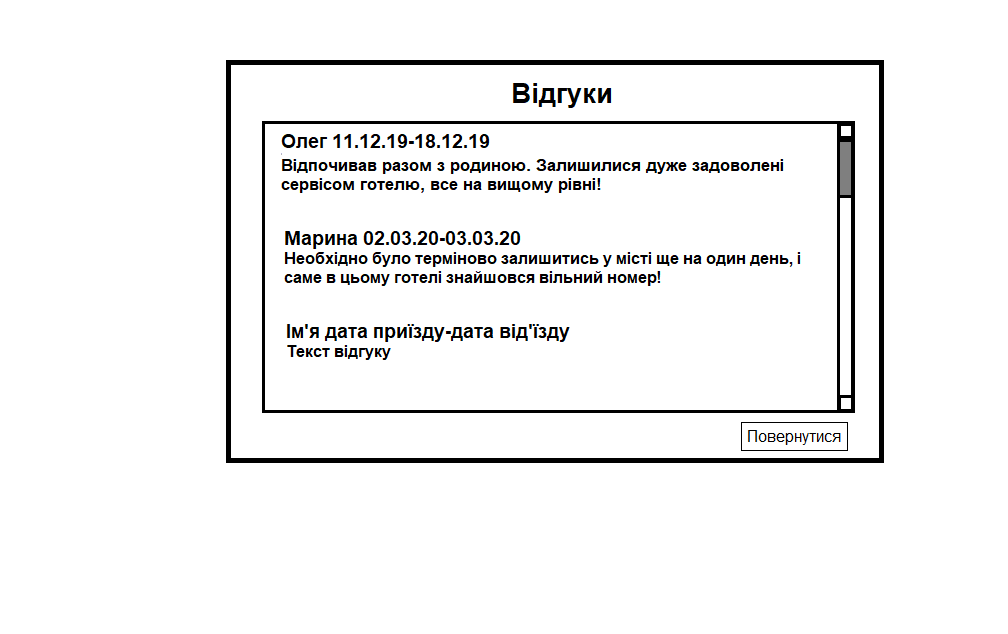


Рисунок 1.12 – Форма відгуків

*Функція «Вихід із програми»*

У процесі роботи користувач вносить зміни до даних, які зберігаються у спільному для адміністратора та гостя файлі даних готелю. Відповідно при авторизації гостя та додаванні ним нового відгуку відбувається автоматичне зберігання змін у цьому файлі. При виході з програми повідомлень про додаткове збереження немає, вона завершує свою роботу.

**2 ПРОЕКТНА СПЕЦИФІКАЦІЯ**

**2.1 Архітектура проекту**

Проект створений на мові програмування C# за допомогою середи розробки Microsoft Visual Studio 2019 на платформі .NET Framework 4.7.2 із застосуванням інтерфейсу програмування додатків Windows Forms.

Вибір мови програмування C# обумовлений об’єктно-орієнтованою парадигмою написання програми. На сьогоднішній день мова C# є однією з найбільш популярних, простих і в той же час потужною серед об’єктно-орієнтованих мов програмування, яка дозволяє створити багатофункціональний додаток.

Вибір платформи .NET Framework пояснюється тим, що вона є більш зрілою і ефективною, має високу сумісність з існуючими додатками та бібліотеками. Таким чином, робота програми підтримується лише комп’ютерами з операційною системою Windows.

Для реалізації інтерфейсу користувача був обраний графічний інтерфейс, оскільки він простий для сприйняття і інтуїтивно зрозумілий. Повноцінний інтерфейс для користувача програми створено за допомогою інтерфейсу Windows Forms, який є частиною платформи .NET Framework.

**2.2 Структура проекту**

Проект розділений на дві окремі програми: для адміністратора (AdminApp) та для гостя готелю (GuestApp), оскільки функції цих типів користувачів відрізняються за рівнем доступу до системи керування готелем. Крім того, для тестування програми створено набори тестових даних для перевірки правильності роботи програми.

Проект складається з 8 класів і 9 форм, при цьому класи зберігаються в бібліотеці проекту HotelManagerLibrary, на яку обидві програми мають посилання. Усі форми були розглянуті у попередньому розділі.

Робота з даними готелю здійснюється за допомогою класу бібліотечного серіалізатора BinaryFormatter. Вони зберігаються у вигляді бінарного файлу “hotel.bin” за шляхом \HotelManager\AdminApp\bin\Debug\. Файл є єдиним для всього проекту.

Квитанції представляються у вигляді текстових файлів формату .txt з відповідними до типу назвами. Зберігання реалізується за допомогою класу StreamWriter; шлях - \HotelManager\AdminApp\bin\Debug\.

Номери готелю мають зображення, представлене файлом зображення .jpg або .png. Усі можливі для вибору зображення зберігаються у каталозі бібліотеки проекту HotelManagerLibrary та копіюються до папок програм при збірці проекту.

Проблема захисту даних у даній роботі не розглядається.

На рис. 2.1 зображено структуру проекту.

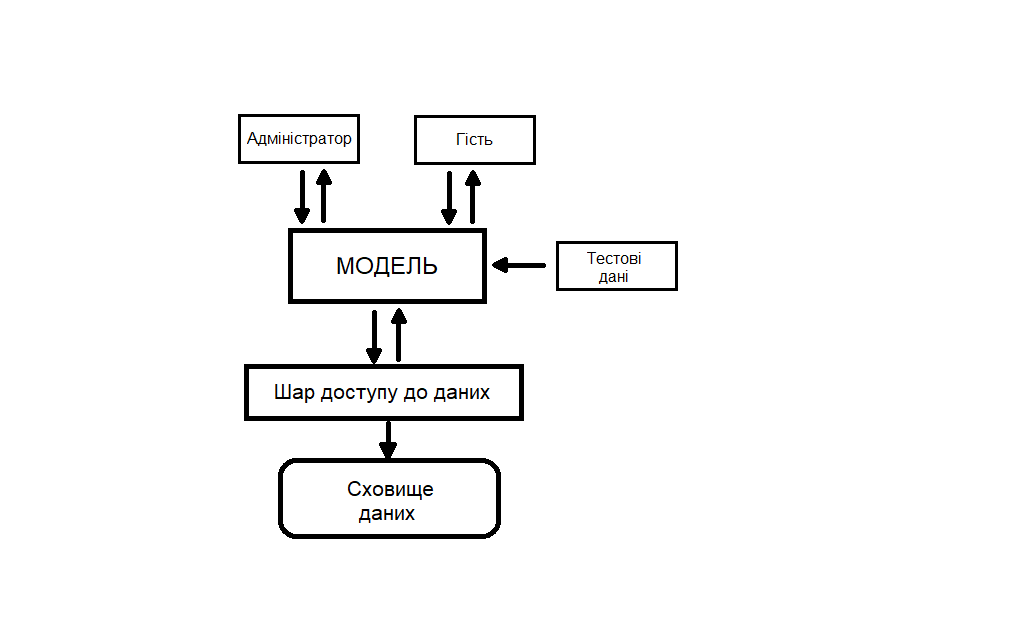


Рисунок 2.1 – Структура проекту

**2.3 Об’єктна модель**

На рис. 2.2 зображена діаграма класів проекту.

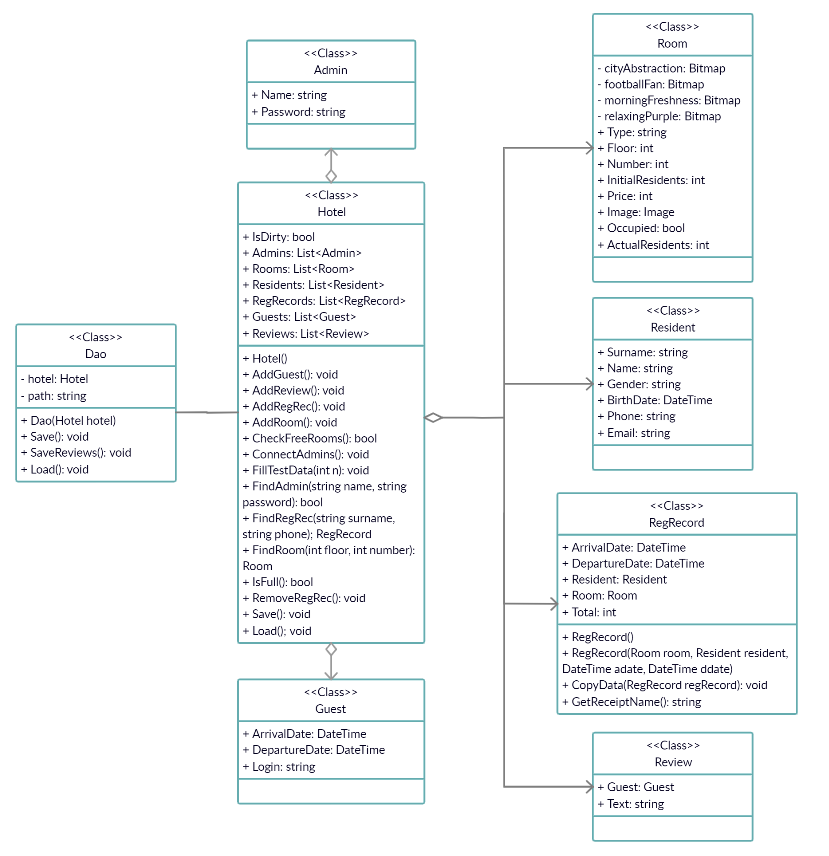


Рисунок 2.2 – Діаграма класів проекту

**2.3.1 Клас Admin**

Клас Admin описує авторизованого користувача – адміністратора, і має дві відкриті властивості:

// Ім’я.

public string Name { get; set; }

// Пароль.

public string Password { get; set; }

**2.3.2 Клас Guest**

Клас Guest описує звичайного користувача – гостя, і має такі властивості:

// Дата приїзду.

public DateTime ArrivalDate { get; set; }

// Дата від’їзду.

public DateTime DepartureDate { get; set; }

// Прізвище-ім’я гостя.

public string Login { get; set; }

**2.3.3 Клас Hotel**

Клас Hotel описує дані готелю – дані адміністраторів, гостей, номерів, записів реєстрацій, постояльців і відгуків. Містить:

Поле, яке позначає, чи були внесені зміни у дані готелю:

public bool IsDirty;

Конструктор:

public Hotel()

{

Admins = new List<Admin>();

Guests = new List<Guest>();

RegRecords = new List<RegRecord>();

Residents = new List<Resident>();

Reviews = new List<Review>();

Rooms = new List<Room>();

ConnectAdmins();

}

Властивості:

// Список адміністраторів.

public List<Admin> Admins { get; private set; }

// Список номерів.

public List<Room> Rooms { get; private set; }

// Список постояльців.

public List<Resident> Residents { get; private set; }

// Список записів реєстрацій.

public List<RegRecord> RegRecords { get; private set; }

// Список гостей.

public List<Guest> Guests { get; private set; }

// Список відгуків.

public List<Review> Reviews { get; private set; }

Набір методів (див. додаток А):

* public void AddGuest() – додає нового гостя;
* public void AddReview() – додає нового відгук;
* public void AddRegRec() – додає новий запис реєстрації;
* public void AddRoom() – додає новий номер;
* public void ConnectAdmins() – під’єднує даних адміністраторів;
* public bool CheckFreeRooms() – перевіряє наявність вільних для додавання номерів;
* public void FillTestData() – заповнює дані готелю тестовими даними;
* public bool FindAdmin() – перевіряє, чи існує адміністратор з такими даними;
* public RegRecord FindRegRec() – здійснює пошук запису реєстрації;
* public Room FindRoom() – здійснює пошук номера;
* public bool IsFull() – перевіряє, чи є в готелі вільні номери;
* public void RemoveRegRec() – видаляє запис реєстрації;
* public void Save() – зберігає дані готелю;
* public void Load() – завантажує дані готелю.

**2.3.4 Клас RegRecord**

Клас RegRecord описує запис реєстрації постояльця та має наступні властивості:

// Дата приїзду постояльця.

public DateTime ArrivalDate { get; set; }

// Дата від’їзду постояльця.

public DateTime DepartureDate { get; set; }

// Дані постояльця.

public Resident Resident { get; set; }

// Дані номеру, у який заселений постоялець.

public Room Room { get; set; }

// Підрахунок суми до сплати.

public int Total

{

get

{

return (DepartureDate.Date - ArrivalDate.Date).Days \* Room.Price;

}

}

Також цей клас має конструктор і два методи (див. додаток Б):

* public RegRecord(Room room, Resident resident, DateTime adate, DateTime ddate) – створює запис реєстрації з переданими даними номера, постояльця, датами приїзду та від’їзду;
* public void CopyData() – копіює дані з одного запису реєстрації в інший;
* public string GetReceiptName() – створює і повертає назву квитанції у вигляді рядка.

**2.3.5 Клас Resident**

Клас Resident описує постояльця готелю і має такі властивості:

// Прізвище.

public string Surname { get; set; }

// Ім’я.

public string Name { get; set; }

// Стать.

public string Gender { get; set; }

// Дата народження.

public DateTime BirthDate { get; set; }

// Номер телефону.

public string Phone { get; set; }

// Електронна пошта.

public string Email { get; set; }

**2.3.6 Клас Review**

Клас Review описує відгук, залишений гостем і має дві властивості:

// Гість, якому належить відгук.

public Guest Guest { get; set; }

// Текст відгука.

public string Text { get; set; }

**2.3.7 Клас Room**

Клас Room описує дані номера готелю: тип, поверх, номер, кількість місць, ціна, зображення, стан зайнятості, кількість постояльців, що проживають. Клас також має поля для зберігання можливих зображень номерів і набір відкритих властивостей (див. додаток В).

**2.3.8 Клас Dao**

Клас Dao містить методи зберігання та завантаження даних готелю (див. додаток Г):

* public void Save() – зберігає дані готелю у файл “hotel.bin”;
* public void SaveReviews() – зберігає відгуки у файл “reviews.txt”;
* public void Load() – завантажує дані готелю;

**3 ІНСТРУКЦІЯ КОРИСТУВАЧА**

**3.1 Встановлення, запуск і вікно входу**

У проекті наявні два типи користувачів: адміністратор і гість, які суттєво відрізняються своїми функціями. З цієї причини проект складається з двох окремих програм, призначених відповідно до адміністратора та гостя.

Після завантаження проекту на комп’ютер необхідно вирішити, можливостями якого типу користувача Ви бажаєте скористатися, після цього відкрити файл за шляхом: 1) для адміністратора - \HotelManager\AdminApp\bin\Release\AdminApp.exe; 2) для гостя - \HotelManager\GuestApp\bin\Release\GuestApp.exe. Усі інші файли проекту залишити без змін.

Після відкриття файлу з’являється вікно входу адміністратора (рис. 3.1) або гостя (рис. 3.2) відповідно до обраної програми типу користувача. Щоб отримати доступ до функцій, необхідно правильно ввести дані. Для адміністратора: один з варіантів імені-паролю серед існуючих (див. табл. 3.1); для гостя: 1) прізвище-ім’я вводяться через пробіл і складаються лише з літер, всього мінімум 5 символів, враховуючи пробіл; 2) між датами приїзду та від’їзду різниця хоча б один день.

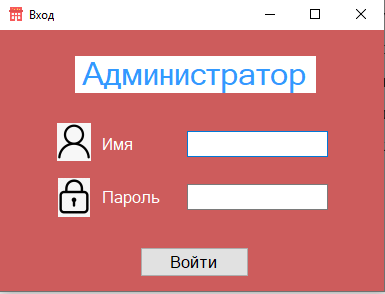


Рисунок 3.1 – Вікно входу адміністратора

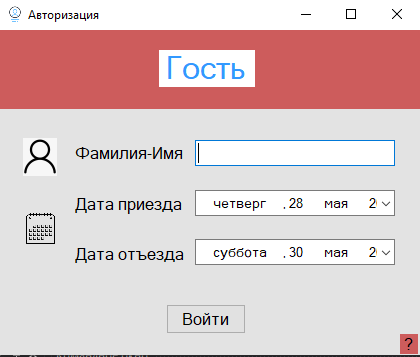


Рисунок 3.2 – Вікно входу гостя

Таблиця 3.1 – Імена та паролі адміністраторів проекту

|  |  |  |
| --- | --- | --- |
| № | Ім’я | Пароль |
| 1 | Дмитрий | 197909 |
| 2 | Елена | 010569 |
| 3 | Сергей | 031963 |

**3.2 Функціонал адміністратора**

Після успішного входу у програму, відкривається головне вікно адміністратора – панель адміністратора (рис. 3.3).

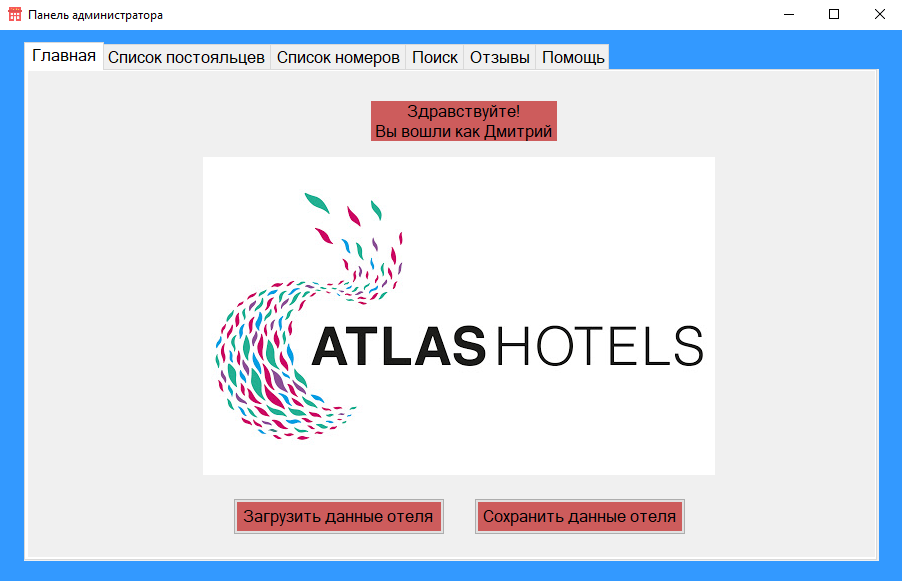


Рисунок 3.3 – Панель адміністратора

На панелі адміністратора програма вітає адміністратора, який зайшов у систему. Також доступне збереження та завантаження даних готелю шляхом натискання кнопок «Сохранить данные отеля» і «Загрузить данные отеля» відповідно.

При відкритті програми готель заповнений тестовими даними. За допомогою них можна швидко перевірити готель на наповненість: не можна поселити постояльців та додати нові номери. Проте увага! Для роботи адміністратора користувач має одразу після відкриття програми натиснути кнопку «Загрузить данные отеля», яка завантажить справні демонстраційні дані готелю. Надалі інструкції користувача надані із врахуванням цього зауваження як виконаного.

Якщо у процесі роботи з даними щось відбулося не так (наприклад, залишаються тестові дані), то для поновлення демонстраційних даних готелю (завантажуються на початку роботи) необхідно скопіювати файли за шляхом \HotelManager\HotelData та вставити у папку \HotelManager\AdminApp\bin\Debug із заміною файлів.

**3.2.1 Поселення постояльців**

Для поселення постояльців необхідно перейти на вкладку «Список постояльцев» (рис. 3.4). На ній розташовані список постояльців готелю та кнопки для роботи з постояльцями.

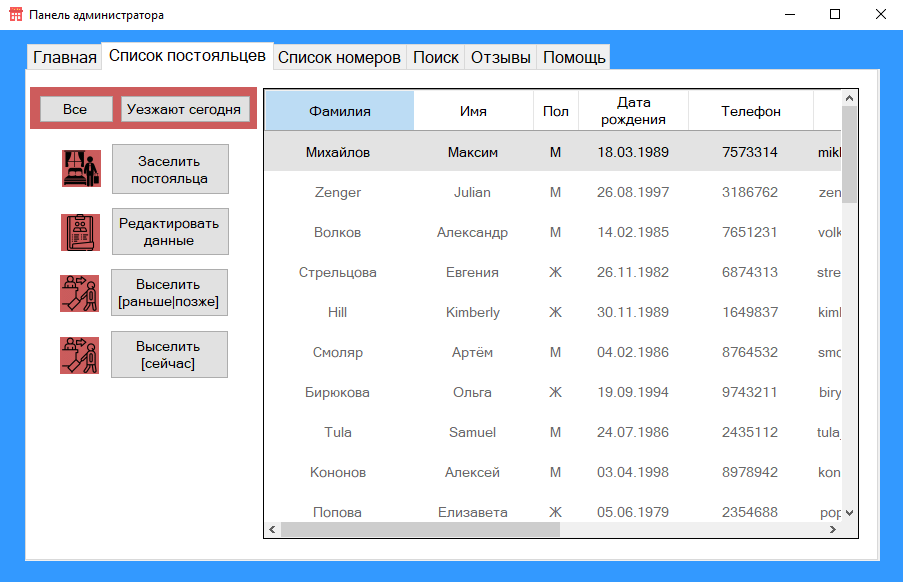


Рисунок 3.4 – Вікно вкладки «Список постояльцев»

Для поселення нового постояльця необхідно натиснути на кнопку «Заселить постояльца». Якщо у готелі є вільні номери, то відкриється вікно для заповнення даних запису реєстрації (рис. 3.5), у противному випадку на екрані з’явиться повідомлення про неможливість поселення.

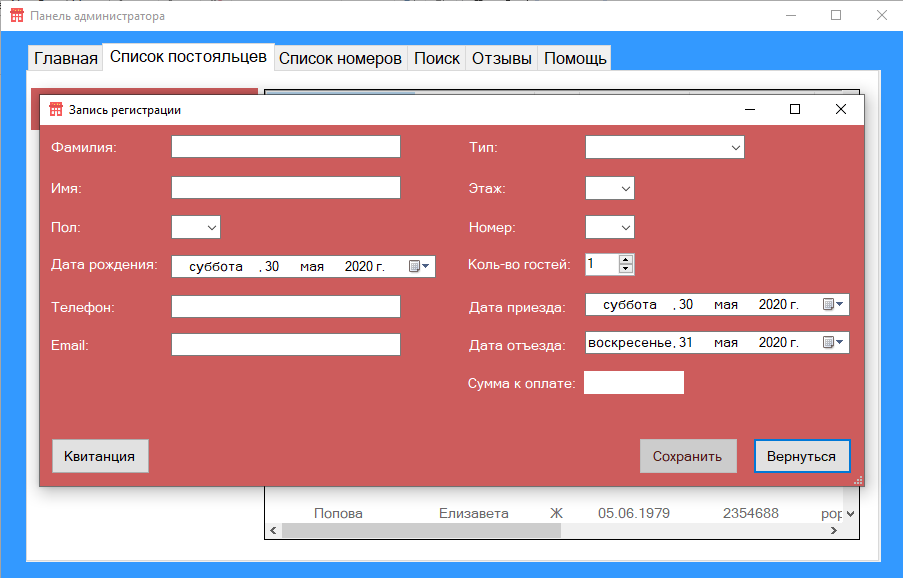


Рисунок 3.5 – Вікно запису реєстрації

Користувач може або одразу повернутися, натиснувши «Вернуться», або правильно заповнити поля для вводу даних і оформити квитанцію шляхом натискання на кнопку «Квитанция». Правильним введенням вважається:

* усі поля заповнені (надруковані або обрані з відповідних елементів вікна);
* прізвище та ім’я заповнені літерами (не менше 2 символів);
* якщо стать не була обрана, то вона буде відмічена як «-»;
* номер не оформлюється на неповнолітнього;
* телефон складається з цифр (не менше 7 символів);
* електронна пошта за форматом латинських букв нижнього реєстру і наявності знаку «равлика» @, допускаються крапки та нижнє підкреслювання (не менше 2 символів до та після знаку «равлика»).

При правильному заповненні за натисканням кнопки «Квитанция» відкривається вікно квитанції (рис. 3.6), де можна ознайомитися з нею та роздрукувати. Натискання на кнопку «Вернуться» скасовує поселення нового постояльця та повертає на вікно запису реєстрації з тільки-но заповненими полями. Натискання на кнопку «Распечатать» підтверджує поселення постояльця. Після цього кнопка «Сохранить» на вікні запису реєстрації стає доступною й можемо повернутися на вкладку списку постояльців, а «Вернуться» вже недоступна – створення нового запису реєстрації більше відмінити не можна. Квитанція зберігається за шляхом \HotelManager\AdminApp\bin\Debug у файлі «receipt\_(прізвище постояльця).txt» .

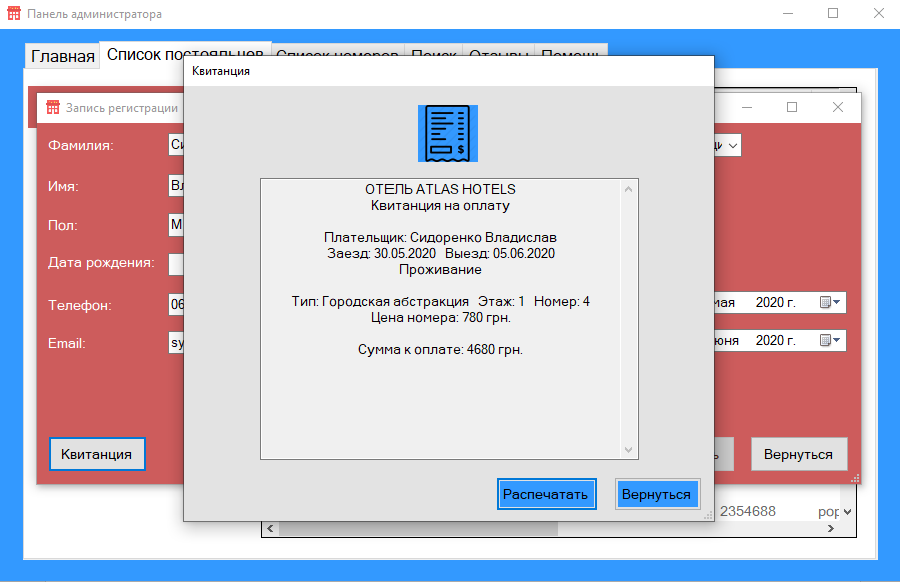


Рисунок 3.6 – Вікно квитанції

**3.2.2 Редагування даних постояльця**

Для редагування даних постояльців необхідно перейти на вкладку «Список постояльцев», обрати постояльця у списку та натиснути на кнопку «Редактировать данные», після чого відкриється вікно запису реєстрації із заповненими полями. До редагування доступні усі поля, окрім кількості гостей, що проживають у номері, і дат приїзду та від’їзду, оскільки вони стосуються процесу виселення постояльця.

Для збереження змін необхідно процес оформлення квитанції за тим же принципом, що й при поселенні постояльця. Відміна змін відбувається шляхом натискання кнопок «Вернуться».

**3.2.3 Виселення постояльця**

Для виселення постояльця необхідно перейти на вкладку «Список постояльцев». Виселяти доступне двома сценаріями:

1. Виселення зараз. Користувач обирає постояльця серед списку, натискає на кнопку «Выселить[сейчас]». На екрані з’являється підтвердження виселення даного постояльця. При підтвердженні постоялець виселяється з готелю, запис реєстрації видаляється, номер знову стає вільним.
2. Виселення зі зміною дати. Користувач обирає постояльця серед списку, натискає на кнопку «Выселить[раньше|позже]». Відкривається вікно виселення постояльця (рис. 3.7), де до редагування доступне лише поле дати від’їзду. Підтвердження виселення відбувається шляхом оформлення квитанції та збереження, як і під час поселення постояльця. Квитанція зберігається за шляхом \HotelManager\AdminApp\bin\Debug у файлі «recalculation\_(прізвище постояльця).txt». У записі реєстрації постояльця змінюється дата від’їзду.

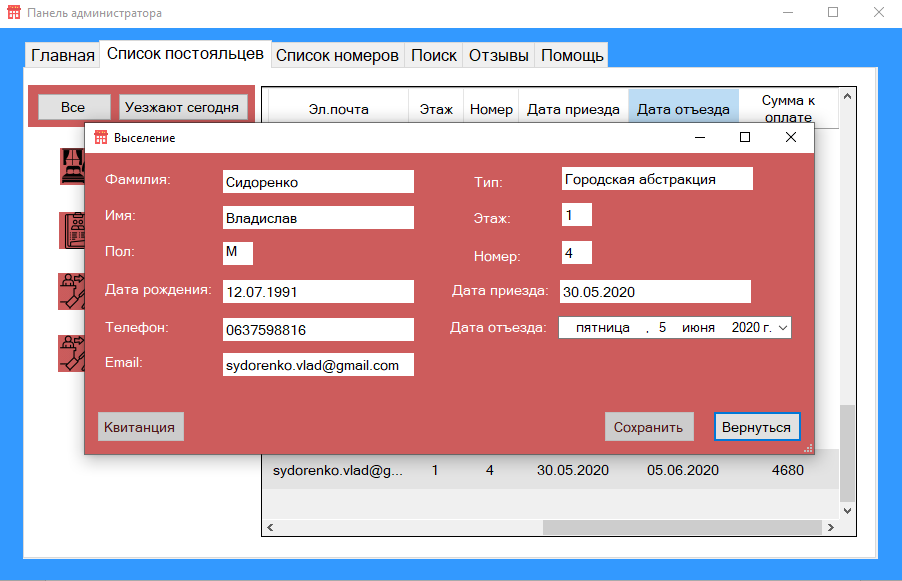


Рисунок 3.7 – Вікно виселення постояльця

За бажанням користувач перед виселенням може залишити у списку постояльців для перегляду лише тих, хто від’їжджає сьогодні, натиснувши на кнопку «Уезжают сегодня». Для відображення усіх постояльців необхідно натиснути на кнопку «Все».

Відміна виселення постояльця відбувається так само натисканням кнопок «Вернуться», як і при поселенні постояльця.

**3.2.4 Додавання номеру**

Для додавання номеру необхідно перейти на вкладку «Список номеров» (рис. 3.8).

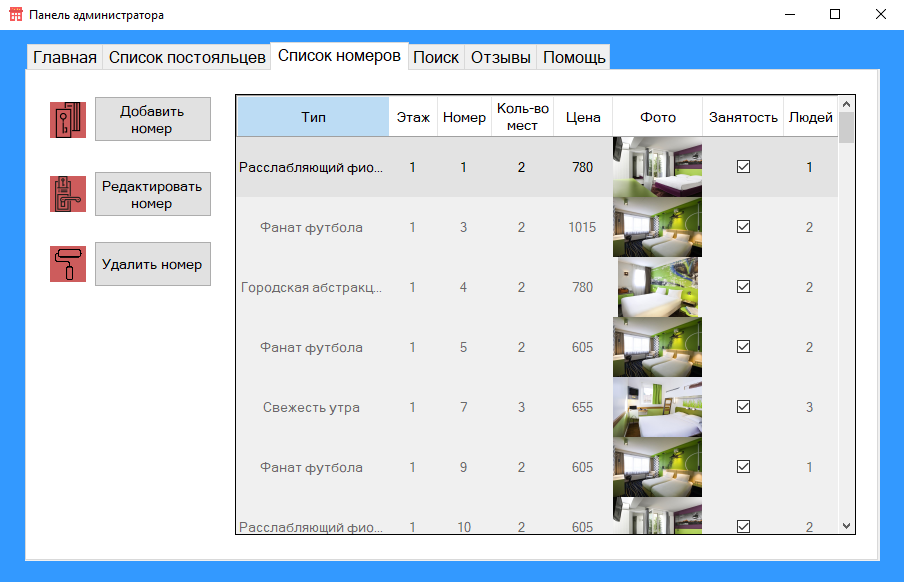


Рисунок 3.8 – Вікно вкладки «Список номеров»

Готель проекту не є безрозмірним і складається з шести поверхів по двадцять номерів на кожному, тобто усього міститься 120 номерів. У списку номерів представлені ті номери, які безпосередньо доступні для поселення постояльців. Так, частина номерів відсутня й доступна для додавання користувачу. Таблиця номерів, які можна додати, знаходиться у додатку Д.

Користувач натискає на кнопку «Добавить номер», і відкривається вікно номеру (рис. 3.9). На ньому обирається зображення номеру за натисканням кнопки «Выбрать изображение» і появою діалогу вибору файлу (обирати лише з відкритої директорії). Встановлюються поверх, номер і ціна (цінова політика готелю знаходиться між 600 і 1200 грн. за ніч). Підтвердження додавання – «ОК», відміна – «Отмена».

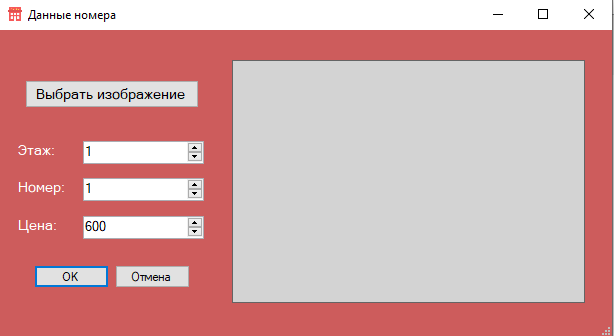


Рисунок 3.9 – Вікно номеру

**3.2.5 Редагування номеру**

Для редагування номеру необхідно перейти на вкладку «Список номеров». Користувач обирає номер у списку номерів і натискає на кнопку «Редактировать номер». Якщо номер незаселений, то відкривається вікно номеру, де до редагування доступні лише зображення та ціна. Підтвердження редагування – «ОК», відміна – «Отмена».

Неможливість зміни поверху та номеру пояснюється тим, що у готелі існують всі 120 номерів відповідно до його розмірів, проте не всі з них можуть в певний проміжок часу бути доступними для поселення.

**3.2.6 Видалення номеру**

Під видаленням номеру мається на увазі видалення номера зі списку як тимчасово недоступного для поселення. При цьому номер з відповідними йому поверхом і номером можна знову додати.

Для видалення номеру необхідно перейти на вкладку «Список номеров». Користувач обирає номер у списку і натискає на кнопку «Удалить номер». Якщо номер незаселений, то з’являється попередження про видалення даного номеру. При підтвердженні номер видаляється.

**3.2.7 Пошук серед постояльців**

Для пошуку серед постояльців необхідно перейти на вкладку «Поиск» (рис. 3.10).

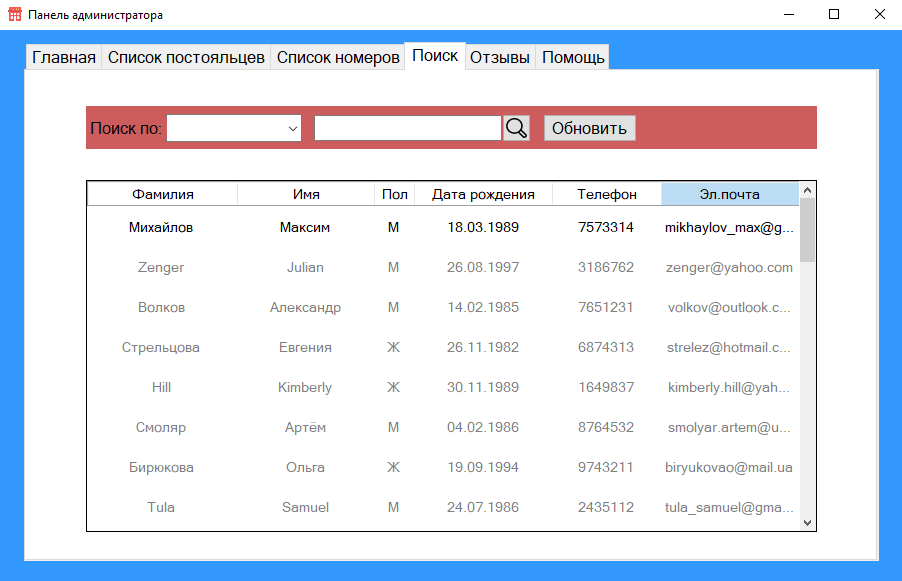


Рисунок 3.10 – Вікно вкладки «Поиск»

Користувач обирає з випадаючого списку справа від надпису «Поиск по:» один з критеріїв пошуку: ім’я, прізвище або дата народження. Вводить дані для знаходження збігів серед даних постояльців у наступне поле, після чого натискає на кнопку з позначкою лупи. Список постояльців оновлюється, в ньому відображаються дані тих постояльців, які співпали із введеними. Для поновлення відображення усіх постояльців необхідно натиснути на кнопку «Обновить».

**3.2.8 Перегляд відгуків**

Для перегляду відгуків необхідно перейти на вкладку «Отзывы» (рис. 3.11). Тут знаходиться поле з текстами відгуків і даними гостей, які їх залишили. Редагування недоступне, оскільки мета відгуків – отримати можливість об’єктивно оцінити роботу адміністратора та готелю. Додатково переглянути усі відгуки можна за шляхом \HotelManager\AdminApp\bin\Debug у файлі «reviews.txt».

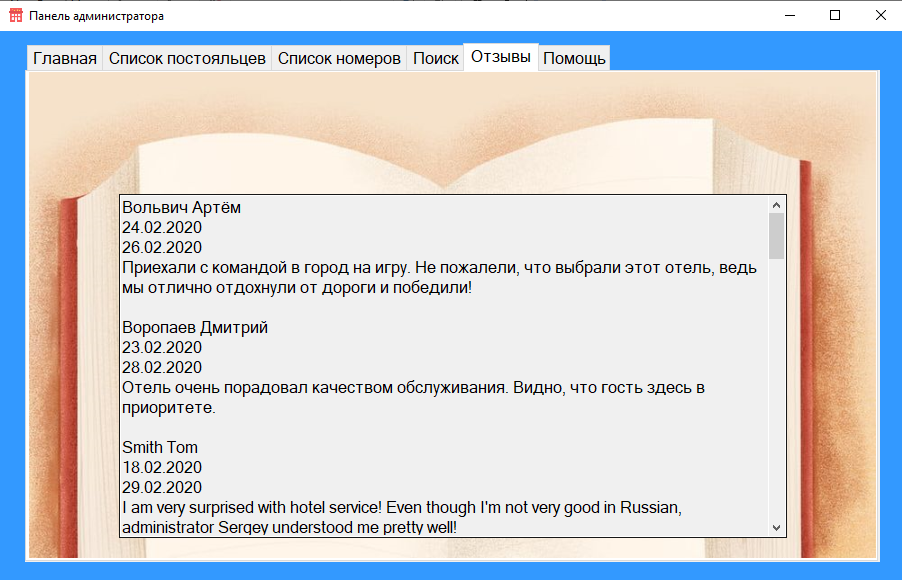


Рисунок 3.11 – Вікно вкладки «Отзывы»

**3.3 Функціонал гостя**

Після успішного входу у програму, відкривається головне вікно гостя (рис. 3.12).

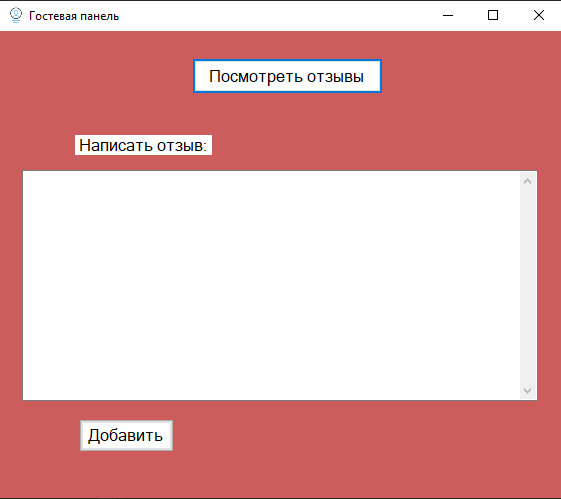


Рисунок 3.12 – Головне вікно гостя

З нього можна перейти до перегляду відгуків гостей готелю, натиснувши на кнопку «Посмотреть отзывы». Відкривається вікно з відгуками (рис. 3.13), з якими можна ознайомитися, проте не редагувати.

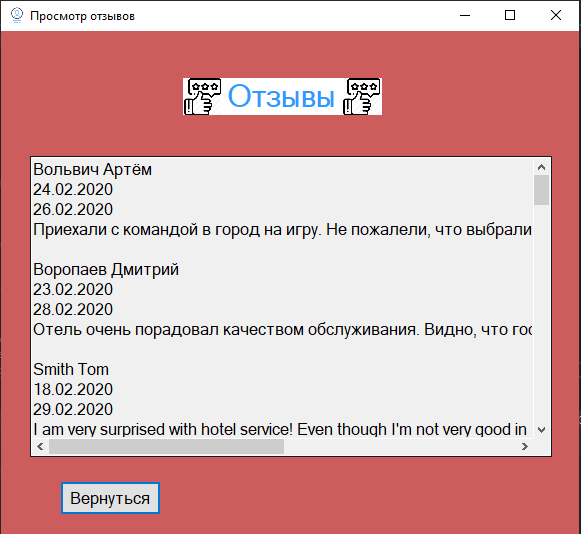


Рисунок 3.13 – Вікно відгуків

Також на головному вікні можна додати власний відгук, який треба надрукувати у вільному полі, після чого натиснути на кнопку «Добавить». При додаванні поле відгуку не може бути порожнім або складатися лише з пробілів, у противному випадку користувач побачить повідомлення про некоректність вводу.

У разі вірно написаного відгуку, відкриється вікно с усіма відгуками гостей готелю. За бажанням користувач може повернутися до головного вікна, натиснувши кнопку «Вернуться», і залишити ще один відгук.

При спробі закрити програму, з’являється вікно з попередженням (рис. 3.14). Натискання «Да» означає, що Ви написали всі відгуки, які хотіли, та бажаєте вийти з програми. Натискання «Нет» означає, що Ви ще не готові вийти з програми.

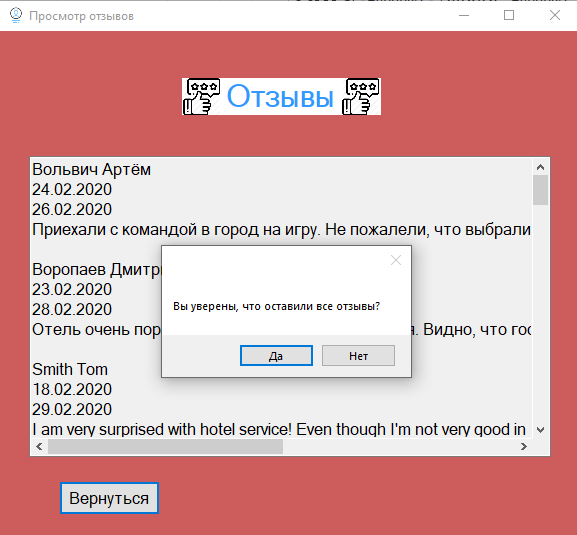


Рисунок 3.14 – Повідомлення про вихід з програми гостя

**ВИСНОВКИ**

У ході виконання курсової роботи були освоєні методи створення програмного забезпечення на засадах об’єктно-орієнтованого програмування, набулися навички роботи з об’єктною програмою, застосування бібліотечних класів для розробки графічного застосування.

У результаті отримана програма під назвою «Адміністратор готелю», яка дозволяє авторизованому користувачу (адміністратор) здійснювати поселення та виселення постояльців, оформлення квитанцій, роботу з номерами, пошук серед постояльців, і дає змогу переглядати відгуки про готель. Звичайний користувач (гість) також може переглядати відгуки інших гостей і додавати власний відгук.

Програма написана на мові програмування C# на платформі .NET Framework 4.7.2 з використанням технологій Windows Forms у середі розробки Microsoft Visual Studio 2019.

Усі дані зберігаються локально: дані готелю – у бінарному файлі, квитанції, відгуки – у текстових файлах формату .txt. Програма може бути поліпшена застосуванням бази даних, яка синхронізується з базами даних інших готелей даної мережі готелей.

Галузь застосування – адміністрування готелю.

**ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ**

1. 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.
2. Либерти Дж. Программирование на С# [Текст] : пер. с англ. / Дж. Либерти. – 2-е изд. – СПб. ; М. : Символ-Плюс, 2003. – 688 с.
3. Документация по .NET Framework. URL: <https://docs.microsoft.com/ru-ru/dotnet/framework/> (дата звернення: 21.02.2020).
4. Полное руководство по языку программирования С# 7.0 и платформе .NET 4.7 [Електронний ресурс] – Режим доступу до ресурсу: https://metanit.com/sharp/tutorial/.
5. Microsoft Developer Network. Библиотека MSDN / MSDN – сеть разработчиков Microsoft. URL: https://msdn.microsoft.com/ru-ru/ (дата звернення: 21.02.2020).

**ДОДАТОК** **А**

**Методи класу Hotel**

// Метод для додавання нового гостя.

public void AddGuest(Guest guest)

{

Guests.Add(guest);

}

// Метод для додавання нового відгуку.

public void AddReview(Review review)

{

Reviews.Add(review);

}

// Метод для додавання нового запису реєстрації.

public void AddRegRec(RegRecord regRecord)

{

Residents.Add(regRecord.Resident);

regRecord.Room.Occupied = true;

RegRecords.Add(regRecord);

}

// Метод для додавання нового номеру.

public void AddRoom(Room room)

{

Rooms.Add(room);

}

// Метод для підключення існуючих пар ім'я (адміністратора) + пароль до системи готелю.

public void ConnectAdmins()

{

Admins.Clear();

Admins.Add(new Admin { Name = "Дмитрий", Password = "197909" });

Admins.Add(new Admin { Name = "Елена", Password = "010569" });

Admins.Add(new Admin { Name = "Сергей", Password = "031963" });

}

// Метод для перевірки наявності номерів для додавання.

public bool CheckFreeRooms()

{

// У готелі 6 поверхів з 20 номерами на кожному.

if (Rooms.Count != 120)

{

return true;

}

return false;

}

// Метод для завантаження тестових даних.

public void FillTestData(int n)

{

// Номери

Rooms.Clear();

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

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

{

Rooms.Add(new Room()

{

ActualResidents = 1,

Occupied = true,

Floor = (i % 6) + 1,

Number = (i % 20) + 1,

Price = i + 600,

Image = noImage

});

}

// Постояльці

Residents.Clear();

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

{

Residents.Add(new Resident()

{

Surname = $"ResidentSurname{i}",

Name = $"ResidentName{i}",

BirthDate = DateTime.Now - TimeSpan.FromDays(i),

Gender = "-",

Phone = "123456789",

Email = "example@gmail.com"

});

}

// Записи реєстрацій

RegRecords.Clear();

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

{

RegRecords.Add(

new RegRecord(Rooms[i],

Residents[i],

DateTime.Today,

DateTime.Today + TimeSpan.FromDays(i + 1)));

}

// Гості

Guests.Clear();

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

{

Guests.Add(new Guest()

{

Login = $"Name{i}",

ArrivalDate = DateTime.Today - TimeSpan.FromDays(i + 1),

DepartureDate = DateTime.Today

});

}

// Відгуки

Reviews.Clear();

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

{

Reviews.Add(new Review

{

Guest = new Guest()

{

Login = $"Name{i}",

ArrivalDate = DateTime.Today - TimeSpan.FromDays(i + 1),

DepartureDate = DateTime.Today

},

Text = $"Text{i}"

});

}

}

// Метод для пошуку адміністратора з переданими даними.

public bool FindAdmin(string name, string password)

{

foreach (var a in Admins)

{

if (a.Name == name && a.Password == password)

return true;

}

return false;

}

// Метод для пошуку запису реєстрації.

public RegRecord FindRegRec(string surname, string phone)

{

foreach (var r in RegRecords)

{

if (r.Resident.Surname == surname && r.Resident.Phone == phone)

return r;

}

return null;

}

// Метод для пошуку номера.

public Room FindRoom(int floor, int number)

{

foreach (var r in Rooms)

{

if (r.Floor == floor && r.Number == number)

return r;

}

return null;

}

// Метод для перевірки наявності вільних номерів у готелі.

public bool IsFull()

{

return Rooms.All(x => x.Occupied == true);

}

// Метод для видалення запису реєстрації.

public void RemoveRegRec(RegRecord regRecord)

{

Residents.Remove(regRecord.Resident);

regRecord.Room.Occupied = false;

regRecord.Room.ActualResidents = 0;

RegRecords.Remove(regRecord);

}

// Метод для збереження даних готелю.

public void Save()

{

new Dao(this).Save();

}

// Метод для завантаження даних готелю.

public void Load()

{

new Dao(this).Load();

}

**ДОДАТОК Б**

**Конструктор і методи класу RegRecord**

// Конструктор, який приймає номер, постояльця, дату приїзду, дату від'їзду.

public RegRecord(Room room, Resident resident, DateTime adate, DateTime ddate)

{

Room = room;

Resident = resident;

ArrivalDate = adate;

DepartureDate = ddate;

}

// Метод для копіювання даних з одного запису реєстрації в інший.

public void CopyData(RegRecord regRecord)

{

this.ArrivalDate = regRecord.ArrivalDate;

this.DepartureDate = regRecord.DepartureDate;

this.Resident.BirthDate = regRecord.Resident.BirthDate;

this.Resident.Email = regRecord.Resident.Email;

this.Resident.Gender = regRecord.Resident.Gender;

this.Resident.Name = regRecord.Resident.Name;

this.Resident.Phone = regRecord.Resident.Phone;

this.Resident.Surname = regRecord.Resident.Surname;

this.Room = regRecord.Room;

}

// Метод для створення назви квитанції.

public string GetReceiptName()

{

foreach (Form form in Application.OpenForms)

{

if (form.GetType().ToString() == "AdminApp.MoveOutForm")

{

return $"recalculation\_{this.Resident.Surname}.txt";

}

}

return $"receipt\_{this.Resident.Surname}.txt";

}

**ДОДАТОК В**

**Клас Room**

public class Room()

{

// Зображення номерів

Bitmap cityAbstraction = new Bitmap(Path.GetFullPath("CityAbstraction.jpg"));

Bitmap footballFan = new Bitmap(Path.GetFullPath("FootballFan.jpg"));

Bitmap morningFreshness = new Bitmap(Path.GetFullPath("MorningFreshness.jpg"));

Bitmap relaxingPurple = new Bitmap(Path.GetFullPath("RelaxingPurple.jpg"));

// Тип.

public string Type

{

get

{

if (Image.Size == CityAbstraction.Size)

return "Городская абстракция";

else if (Image.Size == FootballFan.Size)

return "Фанат футбола";

else if (Image.Size == MorningFreshness.Size)

return "Свежесть утра";

else if (Image.Size == RelaxingPurple.Size)

return "Расслабляющий фиолетовый";

else

return "Тип номера";

}

}

// Поверх.

public int Floor { get; set; }

// Номер.

public int Number { get; set; }

// Кількість місць.

public int InitialResidents

{

get

{

if (Image.Size == CityAbstraction.Size)

return 2;

else if (Image.Size == FootballFan.Size)

return 2;

else if (Image.Size == MorningFreshness.Size)

return 3;

else if (Image.Size == RelaxingPurple.Size)

return 2;

else

return 2;

}

}

// Ціна.

public int Price { get; set; }

// Зображення.

public Image Image { get; set; }

// Стан зайнятості. Початково вільний.

public bool Occupied { get; set; } = false;

// Кількість постояльців, що проживають. Початково 0.

public int ActualResidents { get; set; } = 0;

}

**ДОДАТОК Г**

**Клас Dao**

public class Dao()

{

Hotel hotel;

string path = Path.Combine(Directory.GetParent(Directory.GetCurrentDirectory()).Parent.Parent.FullName,

@"AdminApp\bin\Debug\");

public Dao(Hotel hotel)

{

this.hotel = hotel;

}

// Метод для збереження даних готелю.

public void Save()

{

using (Stream stream = File.Create(path + "hotel.bin"))

{

var serializer = new BinaryFormatter();

serializer.Serialize(stream, hotel);

}

SaveReviews();

}

// Метод для збереження відгуків у окремий файл reviews.txt.

public void SaveReviews()

{

using (var wr = new StreamWriter(path + "reviews.txt"))

{

wr.WriteLine(hotel.Reviews.Count);

foreach (var r in hotel.Reviews)

{

wr.WriteLine();

wr.WriteLine(r.Guest.Login);

wr.WriteLine(r.Guest.ArrivalDate);

wr.WriteLine(r.Guest.DepartureDate);

wr.WriteLine(r.Text);

}

}

}

// Метод для завантаження даних готелю.

public void Load()

{

using (Stream stream = File.OpenRead(path + "hotel.bin"))

{

var serializer = new BinaryFormatter();

Hotel ht = (Hotel)serializer.Deserialize(stream);

Copy(ht.Rooms, hotel.Rooms);

Copy(ht.Residents, hotel.Residents);

Copy(ht.RegRecords, hotel.RegRecords);

Copy(ht.Guests, hotel.Guests);

Copy(ht.Reviews, hotel.Reviews);

}

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

{

to.Clear();

to.AddRange(from);

}

}

**ДОДАТОК Д**

**Таблиця доступних для додавання номерів**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Номер** | **Поверх** | **Номер** | **Номер** | **Поверх** | **Номер** |
| Перший поверх | | Четвертий поверх | |
| 1 | 2 | 4 | 5 |
| 1 | 6 | 4 | 8 |
| 1 | 8 | 4 | 12 |
| 1 | 12 | 4 | 17 |
| 1 | 15 | 4 | 19 |
| 1 | 20 |  |  |
| Другий поверх | | П’ятий поверх | |
| 2 | 8 | 5 | 3 |
| 2 | 11 | 5 | 4 |
| 2 | 16 | 5 | 6 |
| 2 | 19 | 5 | 11 |
|  |  | 5 | 15 |
|  |  | 5 | 20 |
| Третій поверх | | Шостий поверх | |
| 3 | 2 | 6 | 3 |
| 3 | 6 | 6 | 8 |
| 3 | 11 | 6 | 10 |
| 3 | 15 | 6 | 13 |
| 3 | 18 | 6 | 15 |
| 3 | 20 | 6 | 18 |

**Ссылка на гитхаб:** [**https://github.com/AlinaBoholidova/HotelManager**](https://github.com/AlinaBoholidova/HotelManager)