ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра інженерії програмного забезпечення

Пояснювальна записка

до бакалаврської роботи
на ступінь вищої освіти бакалавр

на тему: «**РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ АВТОМАТИЗАЦІЇ РОБОТИ ФІТНЕС-КЛУБУ МОВОЮ ПРОГРАМУВАННЯ C#**»

Виконав: студент 5 курсу, групи ППЗ-52
спеціальності:
121 Інженерія програмного забезпечення
(шифр і назва спеціальності)
Тимко О. Г.
(прізвище та ініціали)
Керівник Залива В. В.
(прізвище та ініціали)
Рецензент
(прізвище та ініціали)

Київ – 2021

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

Навчально-науковий інститут інформаційних технологій

Кафедра Інженерії програмного забезпечення
Ступінь вищої освіти - «Бакалавр»
Напрям підготовки -121 – Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії програмного забезпечення
О.В. Негоденко
“ ” 2021 року

ЗАВДАННЯ
НА БАКАЛАВРСЬКУ РОБОТУ СТУДЕНТУ

Тимко Олег Григорович
(прізвище, ім‘я, по батькові)

1. Тема роботи: «Розробка програмного забезпечення для автоматизації роботи фітнес-клубу мовою програмування C#».
Керівник роботи асистент кафедри ІПЗ, аспірант Залива В.В.
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від — «12» березня 2021 року, №65.
2. Строк подання студентом роботи «01» червня 2021 року.
3. Вхідні дані до роботи:
Операційна система – Windows10
Мова програмування – C#;
Система керування БД – MySQL;
Середовище для розробки Microsoft Visual Studio;
Використана СКБД WorkBench;
4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити).
4.1 Аналіз завдання і вибір методів вирішення.
4.2 Розробка моделі баз даних та структури програмного продукту.
4.3 Організаційно – економічна частина.
4.4 Розробка та тестування програмного продукту.

5. Перелік графічного матеріалу
1 Титульний слайд.
2. Об’єкт, предмет та мета дослідження.
3. Актуальність вибраної теми.
4. Автоматизація роботи фітнес-клубу.
5. Системи аналоги.
6. Постановка задачі.

7. Програмні засоби для вирішення поставленої задачі.
8. UML діаграма прецедентів.
9. Алгоритм роботи програми .
10. UML діаграма компонентів.
11. Схема бази даних.
12. Результати роботи ПЗ.
13. Висновки.

6. Дата видачі завдання 19 квітня 2021 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврської роботи	Строк виконання етапів роботи	Примітка
1	Вибір теми бакалаврської роботи	до 28.03.2020	
2	Розробка завдання до бакалаврської роботи	до 31.03.2020	
3	Вибір оптимальних програмних рішень для написання коду	до 05.04.2020	
4	Складання календарного плану та узгодження з науковим керівником	до 08.04.2020	
5	Збирання теоретичного матеріалу	до 11.04.2020	
6	Участь у наукових конференціях	до 15.04.2020	
7	Написання першого розділу та відправка на перевірку науковому керівнику	до 22.04.2020	
8	Написання другого розділу та відправка на перевірку науковому керівнику	до 01.05.2020	
9	Написання третього розділу, висновку, презентації диплому і відправка на перевірку науковому керівнику	до 12.05.2020	

Студент_____ Тимко О.Г.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Залива В.В.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної роботи: 62 с., 22 рис., 18 табл., 2 додатки, 13 джерел.

Об'єкт дослідження. Робота фітнес-клубу, як підприємства та робота працівників фітнес-клубу.

Предмет дослідження. Можливість автоматизації роботи фітнес-клубу та функції автоматизованих систем, які задовольняють вимоги до автоматизації роботи малого фітнес-клубу.

Методи дослідження. Опитування працівників фітнес-клубу та аналіз аналогів програмного забезпечення для автоматизації роботи фітнес клубу.

Наукова новизна полягає в удосконаленні існуючих методів моделювання та розробки програмного забезпечення, що дає змогу зручно вести клієнтську базу та взаємодіяти з клієнтами фітнес-клубу.

Галузі застосування – дана програма буде застосовуватись в корпоративних цілях.

У результаті роботи здійснена програмна реалізація роботи фітнес-клубу для ведення обліку працівників, відвідувачів, устаткування, документообіг, оплати праці працівників в корпоративній мережі фітнес-клубів.

Головним завданням даної програми є допомога в розвитку малого бізнесу, а саме поодиноких фітнес-клубів, яким не потрібно багато функціоналу, а лише автоматизація роботи та здобуття бази нових та постійних клієнтів .

Програмне забезпечення детально протестоване за допомогою реальних людей, а саме працівниками фітнес-клубу.

В якості середовища для розробки було обрано Visual Studio 2019 від компанії Microsoft.

КОРПОРАТИВНА ПРОГРАМА, ОБЛІК, ДОКУМЕНТООБІГ, C#, MYSQL.

ЗМІСТ

Стор.

Перелік умовних позначень

БД – база даних.
ДСанПіН - Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин.

SQL – Structured query language, декларативна мова програмування для взаємодії користувача з базами даних.

MSIL – Microsoft Intermediate Language, проміжна мова Microsoft.

VS – Visual Studio, включає в себе інтегроване середовище розробки програмного забезпечення.

JMX – Java Management Extensions, керуючі розширення.
JDBC – Java DataBase Connectivity, з'єднання з базами даних на Java

СУБД – Система управління базами даних, набір взаємопов'язаних даних (база даних) і програм для доступу до цих даних.

Вступ

Обґрунтування вибору теми та її актуальність. Оскільки фізичні заняття допомагають турбуватись про своє здоров'я, то вони стають усе більше популярними. У медицині, навіть, існує ряд методик для поліпшення стану хворого за допомогою тренування тіла. Саме тому люди, які бажають виглядати у гарній формі та підтримувати здоров'я стали цікавитись спортом та спортивними тренуваннями. Оскільки далеко не кожна людина може собі дозволити поставити тренажери вдома, з'явились фітнес клуби, де можна підтримувати своє тіло у гарній фізичній формі у зручний час та під наглядом тренерів.

Для повноцінного функціонування фітнес-клубу як підприємства, потрібно вести облік працівників, відвідувачів, устаткування, документообіг, оплати праці працівників. Для комфортного ведення вищезазначених обліків, підприємству необхідна електронна система для автоматизації праці адміністраторів фітнес клубу. Звичайно, є низка комерційних програм, які задовольняють потреби фітнес клубів, які належать до великого бізнесу. Тому фітнес-клуби, які належать до малого та середнього бізнесу використовують не весь функціонал комерційних систем.

Мета роботи. Розробка системи, яка задовольняє потреби фітнес-клубу.

Об'єкт дослідження. Робота фітнес-клубу, як підприємства та робота працівників фітнес клубу.

Предмет дослідження. Можливість автоматизації роботи фітнес-клубу та функції автоматизованих систем, які задовольняють вимоги до автоматизації роботи малого фітнес-клубу.

Завдання дослідження. Виявлення необхідних для автоматизації функцій та створення програмного забезпечення для автоматизації роботи фітнес-клубу.

Методи дослідження. Опитування працівників фітнес-клубу та аналіз аналогів програмного забезпечення для автоматизації роботи фітнес-клубу.

Наукова новизна полягає в удосконаленні існуючих методів моделювання та розробки програмного забезпечення, що дає змогу зручно вести клієнтську базу

та взаємодіяти з клієнтами фітнес-клубу.

Перш за все, потрібно було визначити що потрібно працівникам для зручної роботи з клієнтами в теперішньому положенні в світі, вибрати адекватну, просту архітектуру та стектехнологій із забезпечення ефективного використання програмного продукту. За можливості підтримки його розробником з метою ефективного додавання нового функціоналу або виправлення виявлених недоліків.

Практична значущість результатів: В майбутньому можна буде розширити цей проект в особистих і комерційних цілях. Архітектура добре спланована, тому її розширення не займе багато часу. Подальші розробки дозволять покращити інтерфейс користувача, розширити базу даних та додати багато корисних можливостей та функцій для зручності користувача.

1 ЗАГАЛЬНА ЧАСТИНА

1.1 Огляд систем аналогів

На сучасному ринку програмних продуктів є певний набір автоматизованих систем, які мають ряд корисних та цікавих функцій, що стали прототипами реалізованих функцій у даній розробленій системі. Розглянемо докладніше лише декілька цих проектів:

- 1С: Фітнес-клуб;
- Mobifitness;
- FitBase.

1С: Фітнес-клуб. Програмне рішення для автоматизації масштабних фітнес-клубів. Розроблено мовою програмування 1С у 2004 році компанією ООО «Лабораторія програмного забезпечення». Вже не один рік успішно займає передове місце на ринку забезпечення автоматизованим функціоналом фітнес-клубів. Програмне забезпечення реалізовано за допомогою хмарних сховищ та зі зручним і доволі легко зрозумілим інтерфейсом, який полегшує щоденну роботу працівників фітнес-центру. При автоматизації роботи фітнес-клубу, експерти по встановленню програмного забезпечення аналізують потреби та вимоги фітнес-центру і на основі отриманих даних адаптують програмне забезпечення для замовника. Основний функціонал оригінального програмного забезпечення:

1. Робота з клієнтами, до якої належить:
   - Ведення бази клієнтів;
   - Управління абонементами клієнтів;
   - Облік відвідувань клієнтом фітнес-центру;
   - Попередній запис на заняття;
   - Використання та видача пластикових карт;
   - Система контролю виконання поставлених задач;

2. Управління персоналом, до якого належить:
   - Планування графіка роботи працівників;
   - Облік фактично відпрацьованого часу працівників;
   - Розрахунок заробітної плати;
   - Ведення взаєморозрахунків з працівниками;
3. Ведення аналітики про роботу фітнес-клубу, до якої належить:
   - Звіти по клієнтам;
   - Аналітика фінансових результатів;
   - Звіти по складу;
   - Аналіз роботи працівників;
4. Ведення обліку фінансів, до якого належить:
   - Каса (прибуткові та видаткові ордери з різними видами операцій);
   - Банк (надходження на рахунок і списання з рахунку);
   - Особові рахунки клієнтів (депозит).

Mobifitness. Облікова система для фітнес-клубів та спортивних студій, яка оптимізує робочий процес робітників і підвищує лояльність та довіру клієнтів. Розроблена вона мовами веб-програмування у 2014 році компанією «Академія Mobifitness». Розробники даного програмного продукту забезпечують підтримкою та адаптацією на етапі автоматизації фітнес-клубу, також надають доступ до облікової системи для працівників закладу-замовника та мобільного додатку для клієнтів фітнес-клубу. Всі продукти будуть брендуватись під фірмовий стиль замовника. Основними функціями онлайн-системи для працівників фітнес клубу є:

- Ведення контенту додатку для клієнтів (акції, нові послуги, бонуси);
- Ведення розкладу індивідуальних та групових занять для клієнтів фітнес-клубу;
- Розсилання повідомлень та нагадувань клієнтам фітнес-клубу;
- Ведення клієнтської бази фітнес-клубу;
- Налаштування віджетів для клієнтського додатку;

- Перегляд заявок на отримання клієнтської карти фітнес-клубу;
- Налаштування інтерфейсу та додаткового функціоналу для онлайн-системи.

Для клієнтів передбачено мобільний додаток, який забезпечує потреби саме клієнта:

- Заморозка та подовження клієнтської карти;
- Оплата абонементів, тренувань та додаткових послуг;
- Відображення в особистому кабінеті клієнта повної інформації з приводу його абонементу, історії відвідувань та транзакцій;
- Перегляд та формування особистого розкладу занять згідно з власним графіком життя;
- Перегляд інформації про тренерів фітнес-клубу;
- Оцінювання результатів після певного періоду пройдених тренувань;
- Віджет для відображення нагадувань про заняття у фітнес-клубі.

FitBase. Веб-система для автоматизації роботи однієї корпоративної мережі фітнес-клубів. Система розроблена для того, щоб підвищити лояльність клієнтів та перетворити потенційних клієнтів в дійсних, автоматизувати роботу менеджерів та тренерів. Контролювати фінансову діяльність та зменшити друкований документообіг. Вище зазначені можливості реалізовано за допомогою наступного функціоналу:

- Ведення та синхронізація онлайн-розкладу роботи фітнес-клубу;
- Облік клієнтів, ведення бази клієнтів з історією відвідувань, покупок та користування послугами фітнес-клубу;
- Фільтрація бази клієнтів за цільовими аудиторіями та виконання массових дій (розсилка повідомлень, подовження послуг);
- Автоматизація роботи менеджера (розсилка повідомлень та нагадувань клієнтам про заняття, акції та цікаві пропозиції);
- Фінансовий облік (фіксування доходів та розрахунків фітнес клубу);

– Контроль всієї мережі фітнес-клубу (розподіл на юридичні особи, порівняльні звіти, годинникові пояси, мультиклубні абонементи).

Всі вищезазначені програмні системи забезпечують автоматизацію масштабних фітнес-клубів, шкіл йоги, танцю, пілатесу та басейнів, але для невеликого фітнес клубу ці програмні засоби занадто масштабні. Тому було вирішено, що необхідно виокремити загальний функціонал та розробити програмне забезпечення для автоматизації фітнес-клубу, який не є масштабним фітнес центром і не є частиною мережі фітнес центрів.

1.2 Функціональні вимоги до розроблюваного програмного продукту

Робота фітнес-клубу полягає в обслуговуванні відвідувачів та наданні їм послуг для занять спортом. До цих послуг належать заняття в залі з тренажерами та без тренажерів індивідуально, з тренером або в групі. Також адміністраторам необхідно фіксувати час входу та виходу клієнта з залу, продавати попутні товари (вода, кава, протеїнові солодощі і т.д), продавати абонементи, подовжувати строк дії абонементів. Директорові необхідно аналізувати роботу працівників, розраховувати заробітну плату кожному працівникові окремо за певний період та переглядати звіти покупок та відвідувань фітнес-клубу клієнтами. Ці можливості необхідно реалізувати за допомогою наступного функціоналу:

- Авторизація та реєстрація користувачів;
- Фіксування входу та виходу клієнта з фітнес клубу;
- Перегляд клієнтів, які на даний момент знаходяться в фітнес клубі;
- Продаж користувачеві абонементів, послуг тренера, товарів;
- Подовження строку дії абонементів та послуг тренера;
- Ведення бази клієнтів, адміністраторів та тренерів;
- Пошук клієнта за прізвищем, номером телефону або штрих-кодом карти;
- Генерація звітної інформації про клієнта;
- Генерація звітної інформації про роботу адміністратора або тренера;

- Розрахунок заробітної плати для тренера або адміністратора за певний робочий період;
- Ведення переліків продуктів, абонементів та послуг тренера, які надає фітнес клуб;
- Видалення неіснуючих клієнтів, працівників, що звільнились та прострочених абонементів.

Цільовою аудиторією є власники фітнес-клубів, які є представниками малого бізнесу.

1.3 Вимоги до апаратного та програмного забезпечення

Мінімальні системні вимоги. В ході розробки і тестування було визначено, що для доступу та роботи з застосуванням користувачеві потрібні мінімум наступні характеристики.

Мінімальні вимоги до клієнтського ПК:

Операційна система: Windows XP.

Програмне забезпечення: MySQL версія 5.5

Процесор 32-розрядний (x86) із тактовою частотою 1,0 ГГц.

Оперативна пам’ять: 64 мегабайти (МБ).

Рекомендовані системні вимоги. Для стабільної та безперебійної роботи з застосуванням користувачам рекомендується мати пристрої з наступними параметрами.

Рекомендовані вимоги до клієнтського ПК:

Операційна система: Windows 7 / Windows 10.

Програмне забезпечення: MySQL версія 8.0.20.0

Процесор 64-розрядний (x64) із тактовою частотою 2,16 ГГц.

Оперативна пам’ять: 128 мегабайт (МБ).

1.4 Програмні засоби для вирішення поставленої задачі

Microsoft Visual Studio — це продукт фірми Майкрософт, який включає в себе інтегроване середовище розробки програмного забезпечення та ряд інших інструментальних засобів. Даний продукт дозволяє розробляти як консольні програми, так і програми з графічним інтерфейсом, в тому числі з підтримкою технології Windows Forms. А також веб-сайти, веб-застосунки, веб-служби як в рідному, так і в керованому кодах для всіх платформ, що підтримуються Microsoft Windows, Windows Mobile, Windows Phone, Windows CE, .NET Framework, .NET Compact Framework та Microsoft Silverlight, яке працює під Microsoft Windows і підтримує розробку застосунків для операційних систем Microsoft Windows x86 та x64, Mac OS x86, Android та Apple iOS.

Середовище Visual Studio дозволяє розробляти додатки, використовуючи різні мови програмування, а саме : Visual C#, Visual Basic, Visual F#, Visual C++, Python і т.д.

Версія Visual Studio Community є абсолютно безкоштовною для учнів, студентів та розробників програм з відкритим програмним кодом.

У MS Visual Studio кожне окреме застосування є рішенням (solution), що складається з одного чи декількох проектів (project). Одночасно можливо відкрити тільки одне рішення (з розширенням .sln), при роботі над кількома рішеннями одночасно потрібно запускати декілька вікон Visual Studio.

Visual Studio надає шаблони для проектів найбільш розповсюджених типів. Використання проектів і їх шаблонів дозволяє користувачеві зосередитися на реалізації окремої функції, в той час як проект буде виконувати загальне управління та завдання побудови.

Для створення нового проекту використовується майстер застосувань. Майстер створення програм надає користувальний інтерфейс для створення проекту за шаблоном та створення шаблонів для файлів вихідних текстів. Майстер налаштовує структуру програми, основне меню і панелі інструментів, забезпечує включення деяких заголовних файлів.

Мова програмування C# – це проста, потужна, статично типізована, об’єктно-орієнтована мова програмування від компанії Microsoft. C# входить до сімейства мов програмування C, синтаксис мови буде дуже знайомим програмістам, що працювали з C, C++, Java та JavaScript.

Перша версія мови C# була створена в 1998-2001 роках, групою інженерів Microsoft під керівництвом Андреса Гейлсберга та Скотта Вільтаумота, як основна мова програмування платформи Microsoft .Net.

C# увібрав в себе найкращі властивості попередників – мов C, C++, Modula, Object Pascal, спираючись на практичний досвід їх використання. Деякі проблематичні моделі, що до цього використовувались у мовах програмування, зокрема множинне спадкування класів (яке використовується у мові C++), були свідомо виключені.

В багатьох відношеннях мова C# дуже схожа на Java, це відображено в синтаксисах та основних поняттях цих мов програмування.

Назва мови C# трактується як наступне покоління розвитку C++, а символ # – символізує “++++”. Спочатку в назві фігурував дієз – “♯”(англійською sharp), однак через відсутність цього символу на клавіатурі, використовується знак для позначення номеру “#”. Загалом назву мови можна позначати обома символами.

Розробка мови C# розпочалася в грудні 1998 року, та готувався до випуску разом з продуктами групи Millenium. Проект мав назву COOL (C-style Object Oriented Language), та розроблявся як аналог Java від компанії Oracle. C# був анонсований, широкому загалу, в 2000 році, як основна мова платформи Microsoft .Net Framework. В цьому ж році з’явилася перша загальнодоступна бета-версія.

Перша фінальна версія мови програмування C# була випущена в 2002 році разом з середовищем інтегрованої розробки програмного забезпечення Visual Studio .Net.

Подібно до Java, C# отримав наступні концепції:

– віртуальна машина – платформа .Net виконує програму подібно до віртуальної машини від Java;

– байт-код – програмний код компілюється в проміжкову мову MSIL(Microsoft Intermediate Language), а вже потім перетворюється на машинну мову, в залежності від платформи на якій запускається програма;

– керований код – оскільки програми, написані на C#, виконуються виключно у віртуальному середовищі CLR(Common Language Runtime), це дає змогу контролювати виконання програми та у будь який момент зупинита її, а також контролювати використання пам’яті програмою, за необхідності – збільшувати, чи видаляти частини пам’яті, які використовує програма.

Мова запитів SQL (Structured Query Language) — це декларативна мова програмування для взаємодії користувача з базами даних, що застосовується для формування запитів, оновлення і керування реляційними базами даних, створення схеми бази даних та її модифікації, системи контролю за доступом до бази даних. Сама по собі SQL не є ані системою керування базами даних, ані окремим програмним продуктом. На відміну від дійсних мов програмування, SQL може формувати інтерактивні запити, або бувши вбудованою в прикладні програми, виступати, як інструкції для керування даними. Окрім цього, стандарт SQL містить функції для визначення зміни, перевірки та захисту даних.

Мова SQL включає три підмови:

– Data Definition Language (DDL) – мова визначення даних, що включає такі оператори, як CREATE, ALTER, DROP;

– Data Manipulation Language (DML) – мова обробки даних, який дозволяє запитувати і змінювати дані і включає в себе оператори SELECT, INSERT, UPDATE, DELETE;

– Data Control Language (DCL) – мова керування даними, дозволяє управляти дозволами на доступ до даних і включає оператори GRANT і REVOKE.

Слід враховувати, що стандарт SQL може не повною мірою підтримуватися конкретної СУБД. І навпаки, конкретна СУБД може давати можливість використання в SQL-виразах додаткових функцій, не передбачених стандартом.

Таким чином, більшість компаній-розробників підтримує власний діалект мови SQL, який може бути сумісний з однією з версій стандарту SQL. Нижче наведено перелік деяких популярних СУБД і найменування діалектів SQL:

- Microsoft SQL Server – Transact-SQL або T-SQL;
- Microsoft Access – Jet SQL;
- Oracle Database – PL / SQL;
- IBM DB2 – SQL PL.

Бібліотека для виконання запитів до бази даних MySql.Data. Для того, щоб виконувати різні операції з базою даних та таблицями, які вона містить, потрібно програмно підключитись до серверу та в самій програмі підключити посилання MySql.Data, яке містить загальні функції для підключення, виконання команд та розривання зв’язку з базою даних.

Основними класами для роботи з базою даних є:

1) MySqlConnection – це клас, який використовується, щоб відкрити та керувати з’єднанням з сервером MySQL. Також надає можливість відправляти команди, SQL-запити та читати результати. MySQLConnection конструктор ініціалізує атрибути та виконує з’єднання з сервером MySQL, навіть якщо переданий лише один параметр.

У властивості connectionSt ring потрібно написати дані авторизації для доступу до бази даних:

- Server – назва серврі, на якому знаходиться певна база даних;
- Database – назва бази даних, яка знаходиться на певному сервері;
- Uid – логін, для доступу до даних бази даних;
- Pwd – пароль для доступу до бази даних;
- charset – кодування серверу, на якому знаходиться база даних.

2) MySqlCommand – клас, який представляє оператор SQL для запуску з базою даних MySQL. Конструктор MySqlCommand (String, MySqlConnection) ініціалізує новий екземпляр класу MySqlCommand з текстом запиту та MySqlConnection.

2 ПРАКТИЧНА ЧАСТИНА

2.1. Архітектура програмного забезпечення

2.1.1 UML діаграма прецедентів

Програма розрахована на декілька видів користувачів: директор та адміністратор. Функції системи зображені у вигляді UML діаграми прецедентів з двома акторами: Директор та Адміністратор, яка зображена на рисунку 2.1.1.

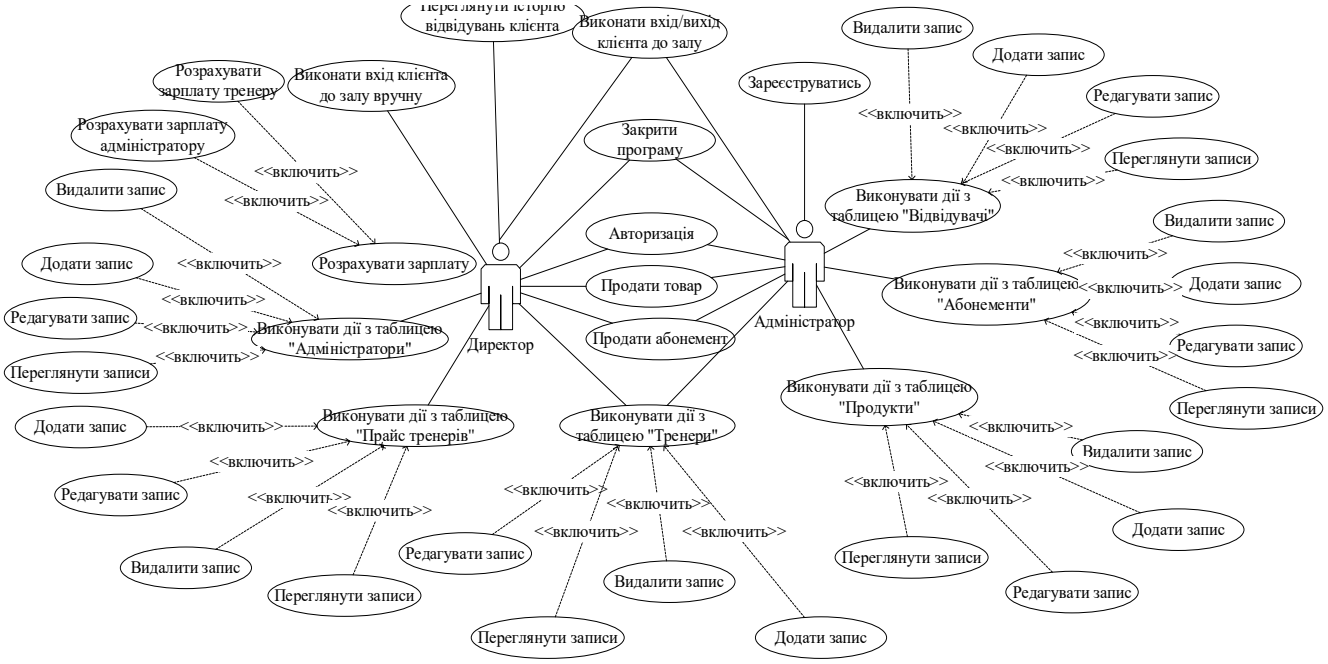


Рисунок 2.1.1 – UML діаграма прецедентів

Діаграма зображує, що користувачі після запуску програми можуть:

- Виконувати дії з таблицею «Адміністратори», тобто переглядати записи у таблиці, редагувати записи у таблиці, додавати записи до таблиці, видаляти записи з таблиці;

- Виконувати дії з таблицею «Адміністратори», тобто переглядати записи у таблиці, редагувати записи у таблиці, додавати записи до таблиці, видаляти записи з таблиці;

- Виконувати дії з таблицею «Адміністратори», тобто переглядати записи у таблиці, редагувати записи у таблиці, додавати записи до таблиці, видаляти записи з таблиці;
- Виконувати дії з таблицею «Адміністратори», тобто переглядати записи у таблиці, редагувати записи у таблиці, додавати записи до таблиці, видаляти записи з таблиці;
- Виконувати дії з таблицею «Адміністратори», тобто переглядати записи у таблиці, редагувати записи у таблиці, додавати записи до таблиці, видаляти записи з таблиці;
- Виконувати дії з таблицею «Адміністратори», тобто переглядати записи у таблиці, редагувати записи у таблиці, додавати записи до таблиці, видаляти записи з таблиці;
- Авторизуватись;
- Зареєструватись;
- Продати товар у вікні виконання входу/виходу клієнта;
- Продати абонемент та/або послуги тренера у вікні виконання входу/виходу клієнта;
- Закрити програму.

Користувач Директор окрім вищеперерахованих можливостей також може:

- Розрахувати заробітну плату, тобто за формулою та вказаним періодом роботи тренера або адміністратора може розрахувати заробітну плату за вказаний період;
- Виконати вхід/вихід клієнта до залу вручну;
- Переглянути історію відвідувань клієнта залу.

2.1.2 Алгоритм роботи програми

Розглянемо докладніше загальний алгоритм роботи програми, блок – схема якого представлена на рисунку 2.1.2. Він має розгалужений вигляд та керує основними діями користувача.

Згідно алгоритму, який зображено на рисунку 3.1.2, програма починає роботу з завантаження головної форми з однією кнопкою у головному меню «Вхід», яка призначена для авторизації користувача. Після авторизації вікно головної форми перезапуститься з трохи зміненим інтерфейсом, тобто у головному меню зникне кнопка «Вхід» з’являться наступні кнопки:

- «Відвідування» - відкриває форму «Відвідування», на якій користувач може зафіксувати факт входу або виходу клієнта до залу, продати абонемент клієнтові, продати послуги тренера клієнтові, заморозити вже проданий абонемент або заняття з тренером на певний період;
- «Довідники» - відкриває наступні підпункти:
- «Продукти» - відкриває форму «Продукти», для перегляду або редагування таблиці «Продукти»;
- «Тренери» - відкриває форму «Продукти», для перегляду або редагування таблиці «Тренери»;
- «Абонементи» - відкриває форму «Продукти», для перегляду або редагування таблиці «Абонементи»;
- «Працівники» - відкриває форму «Продукти», для перегляду або редагування таблиці «Працівники»;
- «Вартість послуг тренера» - відкриває форму «Продукти», для перегляду або редагування таблиці «Вартість послуг тренера»;
- «Відвідувачі» - відкриває форму «Продукти», для перегляду або редагування таблиці «Відвідувачі»;
- «Звіт» - відкриває наступні підпункти:
- «Зарплата» - відкриває форму для розрахунку заробітної плати адміністратора або тренера за певний період роботи;
- «Звіт відвідувань» - відкриває форму для перегляду історії відвідувань залу клієнтом та історію заморозки клієнтом абонементу;
- «Звіт по клієнту» - відкриває форму для перегляду історії покупок клієнта;

- «Звіт зміни» - відкриває форму для перегляду звіту за зміну, тобто суми, яка надійшла до каси фітнес клубу за певний день;
- «Вручну» - відкриває форму для входу або виходу клієнта до залу, а також продовжити період дії абонемента або послуг тренера;
- «Вийти» - виконує вихід користувача з системи та відбувається зміна головної форми до початкового вигляду.

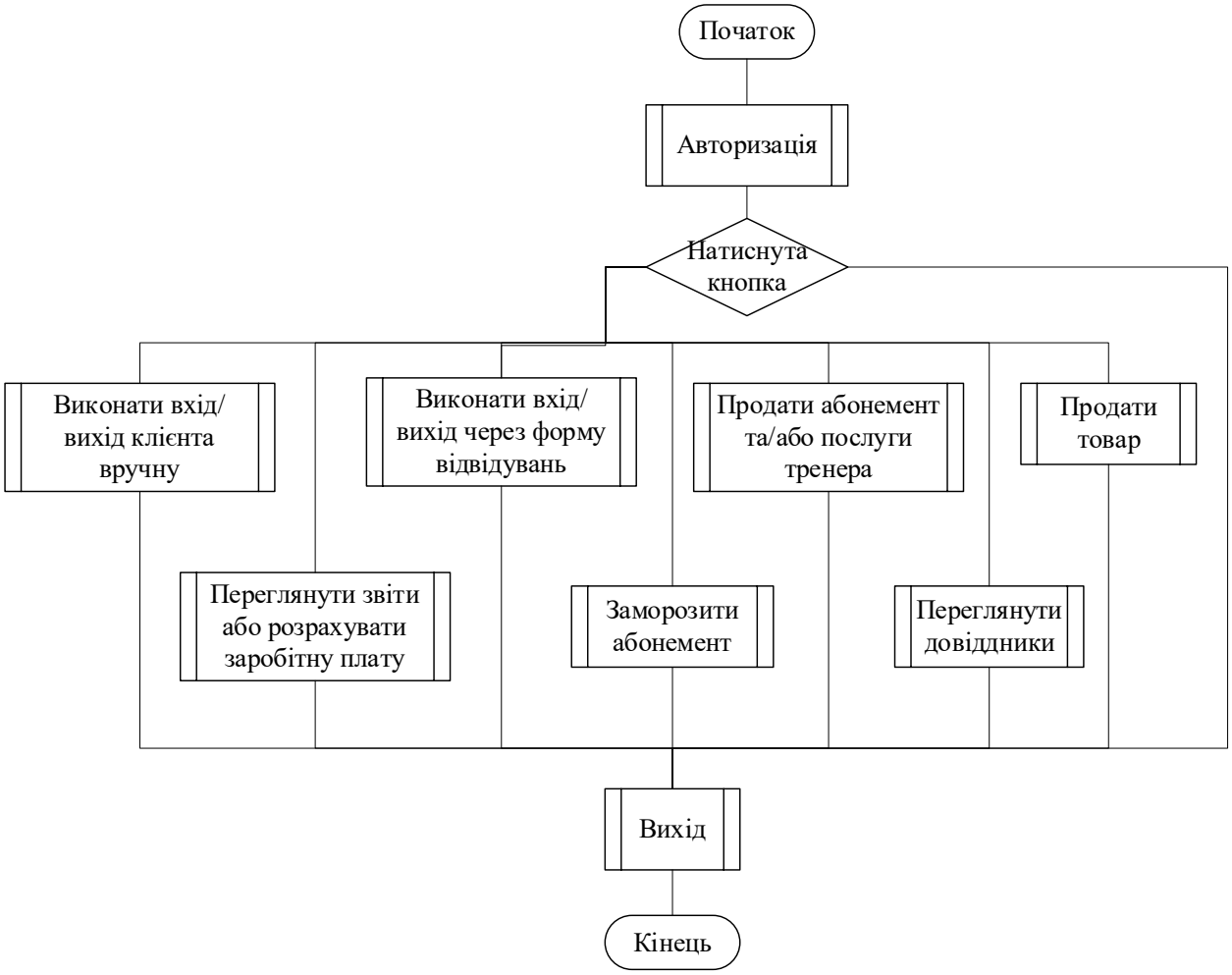


Рисунок 2.1.2 – Алгоритм роботи програми

2.1.3 UML діаграма компонентів

В процесі написання програмного забезпечення для автоматизації роботи фітнес клубу було створено 19 програмних компонентів, які являють собою модулі візуальних форм, що організують інтерфейс користувача та забезпечують взаємодію з користувачем за допомогою візуальних компонентів. Діаграму

компонентів програмної системи представлено на рисунку 2.1.3.

Діаграма зображує лише основні компоненти, не включаючи:

- керуючий модуль «Program.cs», який відповідає за запуск проекту на виконання;
- локальну бібліотеку «DBManager.cs», яка використовується модулями візуальних форм, та містить шаблони запитів обробки даних, які містяться в таблицях бази даних;
- Зовнішню бібліотеку MySql.Data, яка використовується для виконання SQL-запитів та доступу до бази даних.

«fMain» – головна форма проекту. Вона відкривається одразу після запуску програми. На формі розташовано головне меню з пунктом «Вхід», яке після авторизації користувача змінюється на головне меню з пунктами «Відвідування», «Довідники», «Звіт», «Вручну», «Вийти».

«fAutorization» - форма проекту, на якій розташовані поля для введення логіну та паролю користувача, а також кнопки для виконання входу користувача до системи та реєстрації користувача до системи. З’являється після натискання на пункт меню «Вхід» на головній формі проекту «fMain».

«fReestr» - форма проекту, на якій розташовані поля для введення даних користувача та кнопку для реєстрації. З’являється після натискання на кнопку «Зареєструватись» на формі авторизації «fAutorization».

«fUsers» - форма проекту, на якій розташовані поля, таблиця та кнопки, які дозволяють виконувати операції над даними, які містяться в таблиці. З’являється після натискання на пункт меню «Довідники»-«Вхід» на головній формі проекту «fMain».

«fTrener» - форма проекту, на якій розташовані поля, таблиця та кнопки, які дозволяють виконувати операції над даними, які містяться в таблиці. З’являється після натискання на пункт меню «Довідники»-«Тренери» на головній формі проекту «fMain».

«fProducts» - форма проекту, на якій розташовані поля, таблиця та кнопки, які дозволяють виконувати операції над даними, які містяться в таблиці. З’являється після натискання на пункт меню «Довідники»-«Продукти» на головній формі проекту «fMain».

«fAbon» - форма проекту, на якій розташовані поля, таблиця та кнопки, які дозволяють виконувати операції над даними, які містяться в таблиці. З’являється після натискання на пункт меню «Довідники»-«Абонементи» на головній формі проекту «fMain».

«fTrenerPrice» - форма проекту, на якій розташовані поля, таблиця та кнопки, які дозволяють виконувати операції над даними, які містяться в таблиці. З’являється після натискання на пункт меню «Довідники»-«Вартість послуг тренера» на головній формі проекту «fMain».

«fHand» - форма проекту, на якій розташовані поля, таблиця та кнопки, які дозволяють виконувати операції над даними, які містяться в таблиці, та кнопку для продовження періоду дії абонемента. З’являється після натискання на пункт меню «Вручну» на головній формі проекту «fMain».

«fMeetings» - форма проекту, на якій розташовані поля для перегляду інформації певного клієнта та кнопки для входу клієнта до залу, виходу клієнта із залу, продажу абонемента та/або послуг тренера, продажу товарів з бару фітнес клубу, заморозки або розморозки абонемента на певний період. З’являється після натискання на пункт меню «Відвідування» на головній формі проекту «fMain».

«fZarplata» - форма проекту, на якій містяться поля для вибору працівника, тобто адміністратора або тренера, та періоду, за який необхідно порахувати заробітну плату, також кнопку для виконання розрахунку та таблицю, в якій міститься звіт роботи працівника. З’являється після натискання на пункт меню «Звіт»-«Зарплата» на головній формі проекту «fMain».

«fZvitVidv» - форма проекту, на якій розташовані поля для вибору відвідувача, таблиця для перегляду відвідувань обраного клієнта та кнопки для пошуку клієнта зі списку відвідувачів фітнес клубу. З’являється після натискання на пункт меню «Звіт»-«Звіт відвідувань» на головній формі проекту «fMain».

«fZvitKlient» - форма проекту, на якій розташовані поля для вибору відвідувача, таблиця для перегляду покупок обраного клієнта та кнопки для пошуку клієнта зі списку відвідувачів фітнес клубу. З’являється після натискання на пункт меню «Звіт»-«Звіт по клієнту» на головній формі проекту «fMain» або після натискання кнопки «Звіт по клієнту» на формі «fMeetings».

«fZvitZmina» - форма проекту, на якій розташовано поле для вибору дати та поля для перегляду суми каси фітнес клубу за зміну, таблиця для перегляду виконаних покупок за вказаний день. З’являється після натискання на пункт меню «Звіт»-«Звіт зміни» на головній формі проекту «fMain».

«fMoroz» – форма проекту, на якій розташовані поля, кнопки та таблиця, які необхідні для заморозки абонемента клієнта. З’являється після натискання кнопки «Заморозити/Розморозити» на формі «fMeetings».

«fPosetit» - форма проекту, на якій розташовані поля, таблиця та кнопки, які дозволяють виконувати операції над даними, які містяться в таблиці «Відвідувачі». З’являється після натискання на пункт меню «Довідники»-«Відвідувачі» на головній формі проекту «fMain» або після натискання на кнопку «…» на формі «fMeetings».

«fProdAbon» - форма проекту, на якій розташовані поля, кнопки та таблиця, які необхідні для фіксації продажу абонемента та/або послуг тренера клієнтові. З’являється після натискання кнопки «Продати абонемент» на формі «fMeetings».

«fProdTovar» - форма проекту, на якій розташовані поля, кнопки та таблиця, які необхідні для фіксації продажу продукту клієнтові. З’являється після натискання кнопки «Продати товар» на формі «fMeetings».

fSelect – форма проекту, на якій розташовано таблицю та кнопки «Прийняти» та «Відмінити». З’являється після натискання кнопки «…» на формі «fMoroz», «fProdAbon», «fProdTovar», «fPosetit».

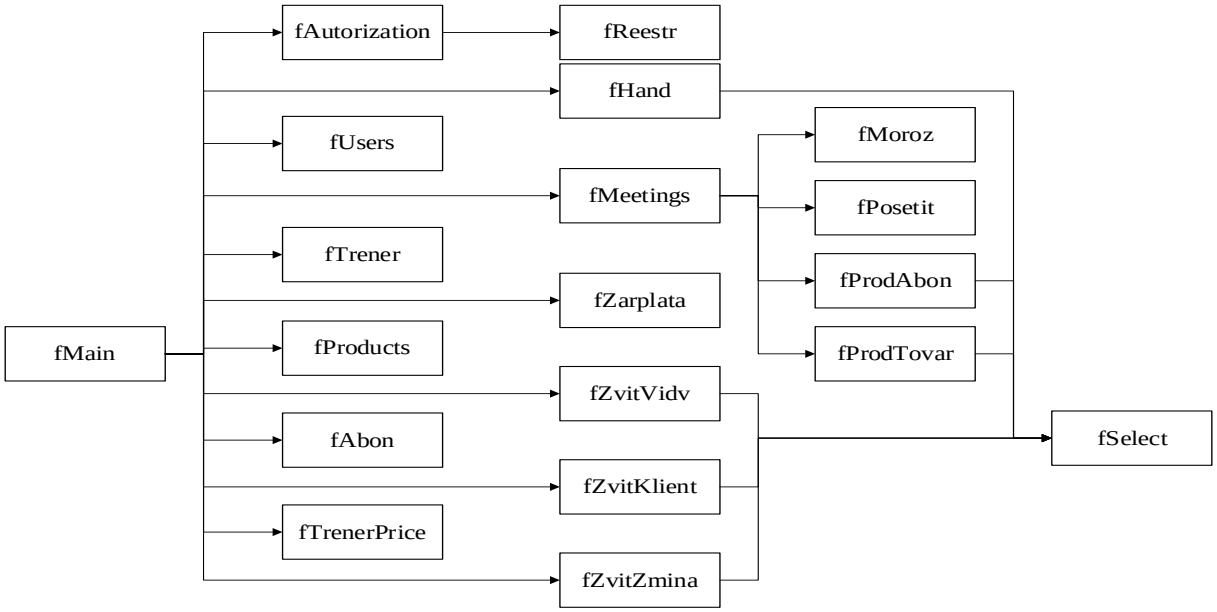


Рисунок 2.1.3 - UML діаграма компонентів

2.1.4 Схема бази даних

Програма використовує реляційну базу даних, розроблену спеціально для дипломного проекту, яка створена за допомогою утиліти «Workbench». База даних складається з десяти таблиць, пов’язаних між собою. Структура бази даних представлена на рисунку 2.1.4.

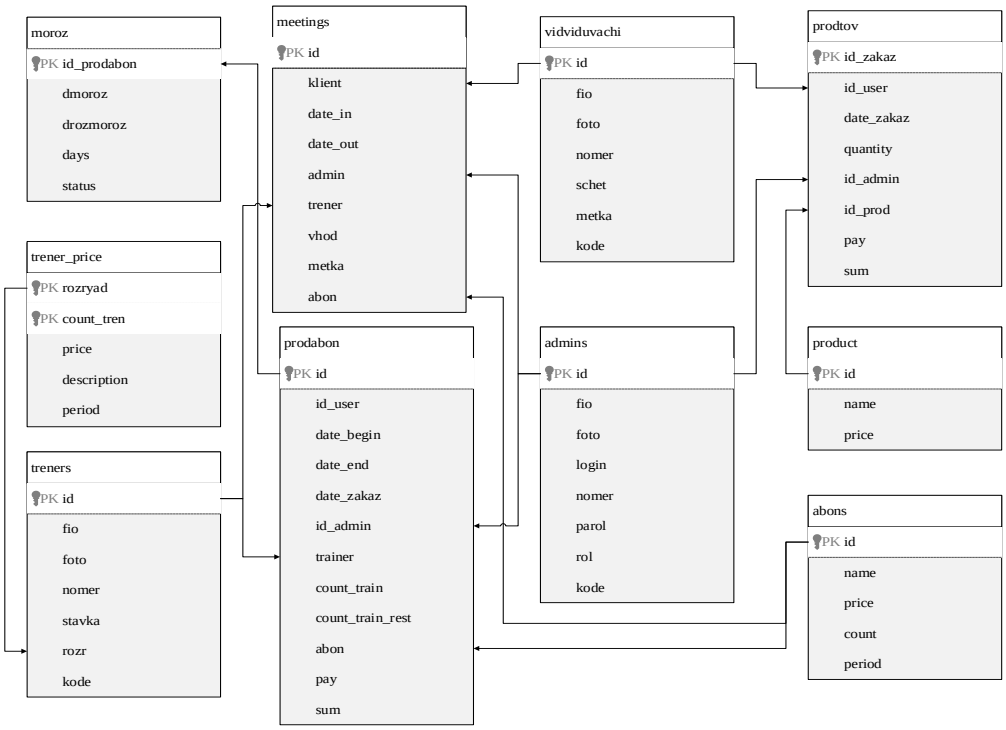


Рисунок 2.1.4 – Схема бази даних

2.1.5 Структура таблиць бази даних представлена нижче та містить опис призначення таблиць та опис властивостей полів.

У таблиці 2.1.1 подано список полів таблиці «moroz», в якій зберігається історія «заморозки/розморозки» абонементів.

Таблиця 2.1.1 – Список полів таблиці «moroz»

Назва	Тип	Розмір	Обмеження
id_prodabon	int	11	primary key
dmoroz	date	17	
drozmoroz	date	17	
days	int	11	
status	bool	1	

У таблиці 2.1.2 подано список полів таблиці «trener_price», яка зберігає вартість послуг тренера.

Таблиця 2.1.2 - Список полів таблиці «trener_price»

Назва	Тип	Розмір	Обмеження
rozryad	int	11	primary key
count_tren	int	11	primary key
price	float		
description	varchar	50	
period	int	11	

У таблиці 2.1.3 подано список полів таблиці «treners», яка зберігає вартість список тренерів.

Таблиця 2.1.3 - Список полів таблиці «treners»

Назва	Тип	Розмір	Обмеження
id	int	11	primary key
fio	varchar	150	
foto	varchar	50	
nomer	varchar	15	
stavka	int	11	
rozr	int	11	
kode	varchar	13	

У таблиці 2.1.4 подано список полів таблиці «meetings», яка зберігає історію відвідувань залу фітнес клубу клієнтами.

Таблиця 2.1.4 - Список полів таблиці «meetings»

Назва	Тип	Розмір	Обмеження
id	int	11	primary key
admin	int	11	foreign key
trener	int	11	foreign key
klient	int	11	foreign key
date_in	varchar	19	
date_out	varchar	19	
vhod	bool	1	
metka	varchar	200	
abon	int	11	foreign key

У таблиці 2.1.5 подано список полів таблиці «prodabon», яка зберігає історію покупок абонементів та послуг тренерів клієнтами.

Таблиця 2.1.5 – Список полів таблиці «prodabon»

Назва	Тип	Розмір	Обмеження
id	int	11	primary key foreign key
id_user	int	11	foreign key
trainer	int	11	foreign key
abon	int	11	foreign key
date_begin	varchar	19	
date_end	varchar	19	
date_zakaz	varchar	19	
id_admin	int	11	foreign key
count_train	int	11	
count_train_rest	int	11	
pay	float		
sum	float		

У таблиці 2.1.6 подано список полів таблиці «vidviduvachi», яка зберігає список відвідувачів.

Таблиця 2.1.6 – Список полів таблиці «vidviduvachi»

Назва	Тип	Розмір	Обмеження
id	int	11	primary key
fio	varchar	150	
foto	varchar	50	
nomer	varchar	15	
schet	float		
metka	varchar	50	
kode	varchar	13	

У таблиці 2.1.7 подано список полів таблиці «admins», яка зберігає список адміністраторів та їхні дані авторизації.

Таблиця 2.1.7 – Список полів таблиці «admins»

Назва	Тип	Розмір	Обмеження
id	int	11	primary key
fio	varchar	150	
foto	varchar	50	
login	varchar	30	
parol	varchar	15	
nomer	varchar	15	
rol	int	11	
kode	varchar	13	

У таблиці 2.1.8 подано список полів таблиці «prodtov», яка зберігає історію покупок продуктів.

Таблиця 2.1.8 – Список полів таблиці «prodtov»

Назва	Тип	Розмір	Обмеження
id_zakaz	int	11	primary key
id_user	int	11	foreign key
date_zakaz	varchar	19	
quantity	int	11	
sum	float		
pay	float		
id_admin	int	11	foreign key
id_prod	int	11	foreign key

У таблиці 2.1.9 подано список полів таблиці «product», яка зберігає список продуктів.

Таблиця 2.1.9 – Список полів таблиці «product»

Назва	Тип	Розмір	Обмеження
id	int	11	primary key
name	varchar	100	
price	float		

У таблиці 2.1.10 подано список полів таблиці «abons», яка зберігає список абонементів.

Таблиця 2.1.10 – Список полів таблиці «abons»

Назва	Тип	Розмір	Обмеження
id	int	11	primary key
name	varchar	100	
period	varchar	50	
price	float		
count	int	11	

2.2 Тестування та демонстрація роботи програмного забезпечення

В ході розробки програмного забезпечення для автоматизації роботи фітнес клубу було проведено тестування можливостей та коректності роботи системи.

Тестування – це процес проведення керованих експериментів з програмним продуктом за допомогою виконання тестів з метою виявлення в ньому невідповідностей, згідно з технічним завданням.

Тест – це контрольне завдання для перевірки коректності роботи функцій системи, або модулів цієї системи.

«Вдалим» тестом вважається такий, при котрому виконання програми закінчилось з помилкою і навпаки. Тестування виконує дві основні задачі:

1. демонстрація якості функціонування програмного забезпечення;
2. знаходження і усунення помилок в програмному забезпеченні.

Головною метою тестування є збільшення ймовірності того, що ПЗ, яке проходить тестування, буде відповідати вимогам.

Тестування – це ітераційний процес. Після того, як виявлено та виправлено помилку обов'язково слідує повторення тестів, що має на меті перевірити працездатність програми. Більш того, для ідентифікації причини виявленої проблеми може бути потрібно проведення спеціального додаткового тестування. При цьому завжди необхідно пам'ятати фундаментальний висновок, що зробив професор Едсгером Дейкстрою в 1972 р.: «Тестування програм може служити доказом наявності помилок, але ніколи не доведе їх відсутність!».

Структурне тестування, також називають тестуванням за принципом «білої скриньки» або «скляної скриньки», полягає у перевірці внутрішньої структури елементів системи.

Основним видом тестування є функціональне. Функціональне тестування застосовують для програмного забезпечення у цілому, а також для програмних об'єктів будь-якого рівня (процедури, модулі, підсистеми, системи). Структурне тестування доповнює функціональне. Цей вид тестування можливий для рівня не вище рівня програмного модуля. Виконання функціонального і структурного тестування системи може бути здійснене незалежно одне від одного.

Структурне тестування програмного забезпечення може бути реалізоване такими методами:

- тестуванням маршрутів;
- тестуванням циклів;
- тестуванням обробки даних.

Перш за все протестовано можливість авторизації та реєстрації користувачів до системи. Для того, щоб авторизуватись, потрібно на головній формі системи, яка зображена на рисунку 2.2.1, натиснути пункт головного меню «Вхід».

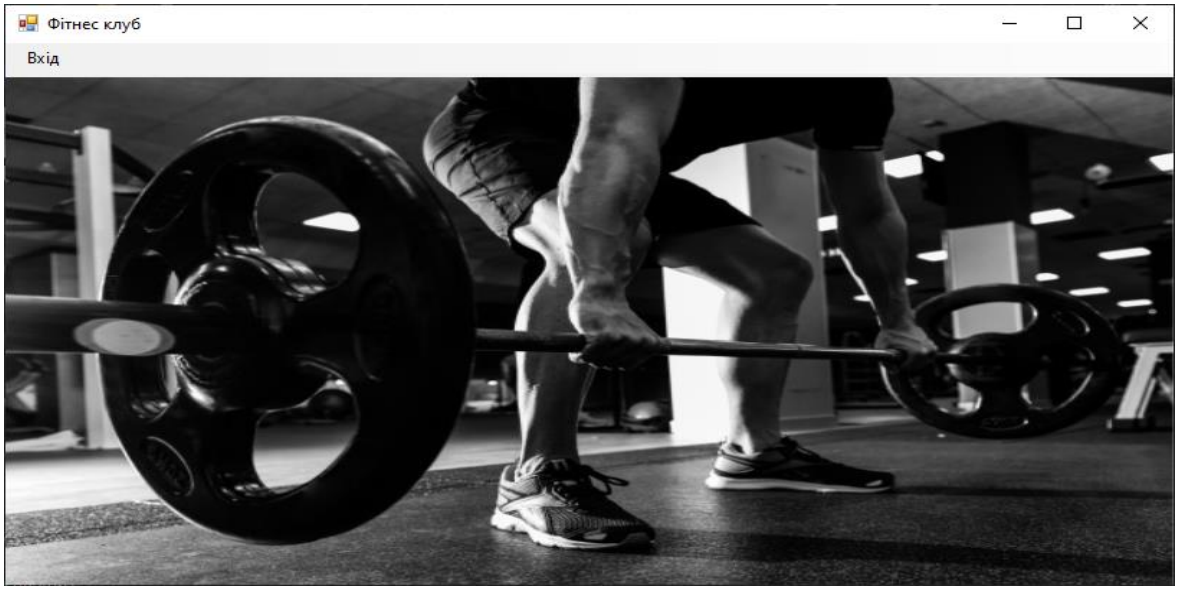


Рисунок 2.2.1 – Головна форма системи

У вікні «Авторизація», яке зображено на рисунку 2.2.2, необхідно заповнити поля «Логін» та «Пароль» значеннями, які попередньо адміністратор ввів при реєстрації в системі.

Якщо адміністратор потрапляє до системи вперше, то він повинен зареєструватись в системі у вікні «Реєстрація», яке можна відкрити натиснувши кнопку «Реєстрація» у вікні «Авторизація».

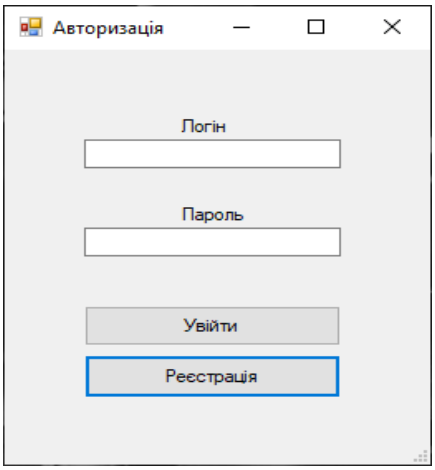


Рисунок 2.2.2 – Вікно «Авторизація»

На робочій області вікна «Реєстрація» розташовано поля, зображено на рисунку 2.2.2, які необхідно заповнити:

– «Прізвище Ім’я По-батькові» з обмеженням 150 символів;
– «Логін» з обмеженням 30 символів;
– «Пароль» з обмеженням 15 символів;
– «Номер телефону» з обмеженням 9 символів.

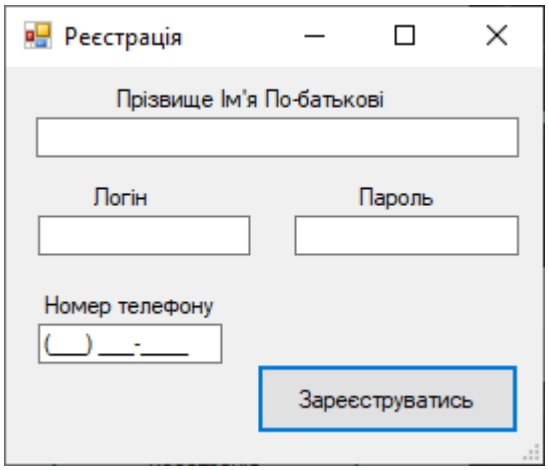


Рисунок 2.2.3 – Вікно «Реєстрація»

Після заповнення всіх полів на вікні «Реєстрація», користувач (адміністратор фітнес клубу) натискає кнопку «Зареєструватись» та переходить на вікно «Авторизація», де заповнює поля «Логін» та «Пароль». І потім натискає кнопку «Увійти».

При правильно введених даних програма виконає вхід та змінить пункти головного меню, а при неправильно введених даних програма відкриє повідомлення про помилку, яке зображено на рисунку 2.2.4.

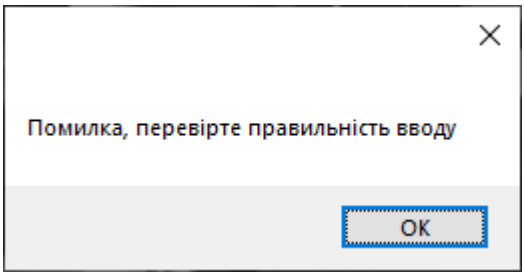


Рисунок 2.2.4 – Вікно повідомлення про помилку4

Після авторизації до системи адміністратора, дані якого зберігаються в базі

даних, пункт головного меню «Вхід» зникає та з’являються наступні пункти (Рисунок 2.2.5):

- «Відвідування» - вікно для виконання послуг, які надає фітнес клуб клієнтові;
- «Довідники» - пункт, що містить наступні підпункти:
- «Відвідувачі» - вікно для ведення переліку відвідувачів фітнес клубу;
- «Працівники» - вікно для ведення переліку адміністраторів фітнес клубу;
- «Тренери» - вікно для ведення переліку тренерів фітнес клубу;
- «Абонементи» - вікно для ведення переліку абонементів, які надає фітнес клуб для своїх клієнтів;
- «Продукти» - вікно для ведення переліку продуктів, які продає фітнес клуб своїм клієнтам;
- «Звіт» - вікно для отримання звітної інформації про надані послуги клієнтові фітнес клубом;
- «Вручну» - виконання входу та виходу клієнтів до фітнес клубу з можливістю видалення записів та редагування дати входу та виходу;
- «Вийти» - вікно для виконання виходу адміністратора з програми.

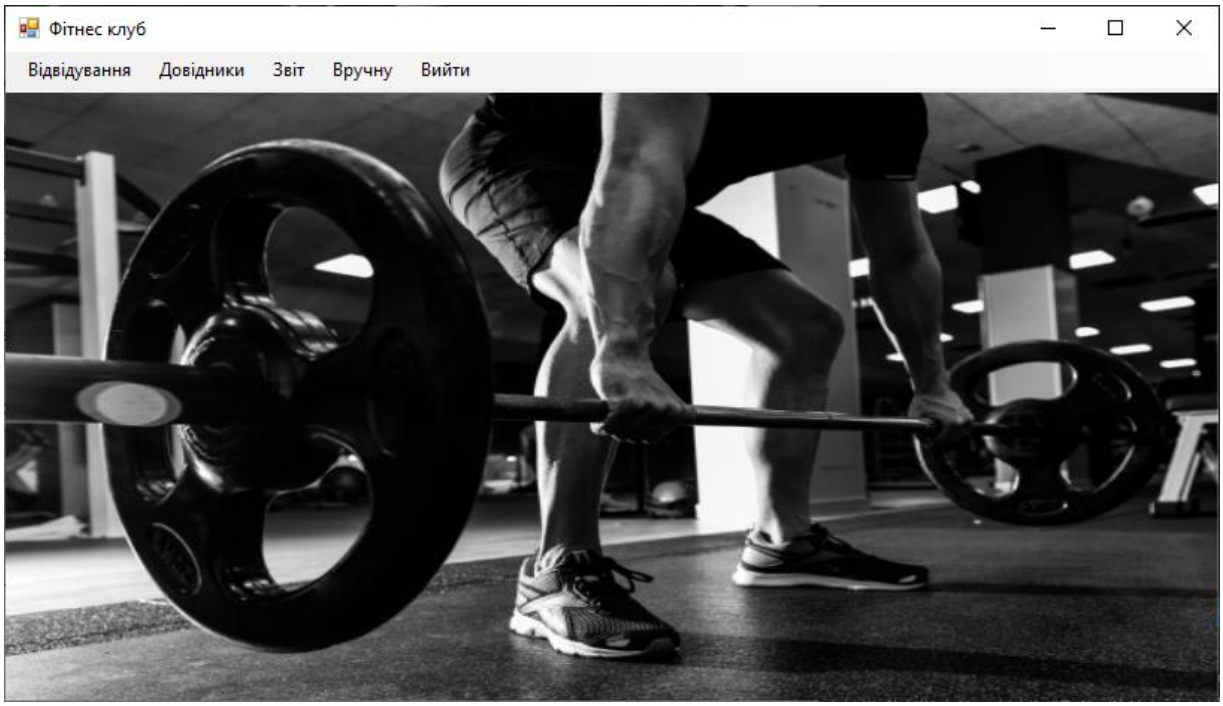


Рисунок 2.2.5 – Головне вікно програми після авторизації користувача

Наступним кроком було протестовано можливість обслуговування клієнтів. Основним вікном для обслуговування клієнтів фітнес клубу є вікно «Відвідування», яке зображено на рисунку 2.2.6, . На нього можна потрапити натиснувши кнопку «Відвідування» у меню на головному вікні програми.

На робочій області вікна «Відвідування» розташовано поля для перегляду інформації про клієнта, таблиця для перегляду дійсних послуг, які йому попередньо продав адміністратор, та кнопки для здійснення обслуговування клієнта та надання йому послуг.

До обслуговування клієнтів належать можливості, які виконують наступні кнопки:

- «Вхід» фіксує вхід клієнта до фітнес клубу;
- «Вихід» фіксує вихід клієнта з фітнес клубу;
- «Клієнти в залі» відкриває вікно з переліком тренерів та клієнтів, які в даний час фітнес клубі;
- «Звіт по клієнту» відкриває вікно для перегляду історії покупок клієнта та відвідувань фітнес клубу;
- «Продати абонемент» відкриває вікно для продажу абонементу та/або послуг тренера;
- «Продати товар» відкриває вікно для продажу товару;
- «Заморозити/ Розморозити» відкриває вікно для встановлення статусу недійсності для куплених клієнтом дійсних послуг.

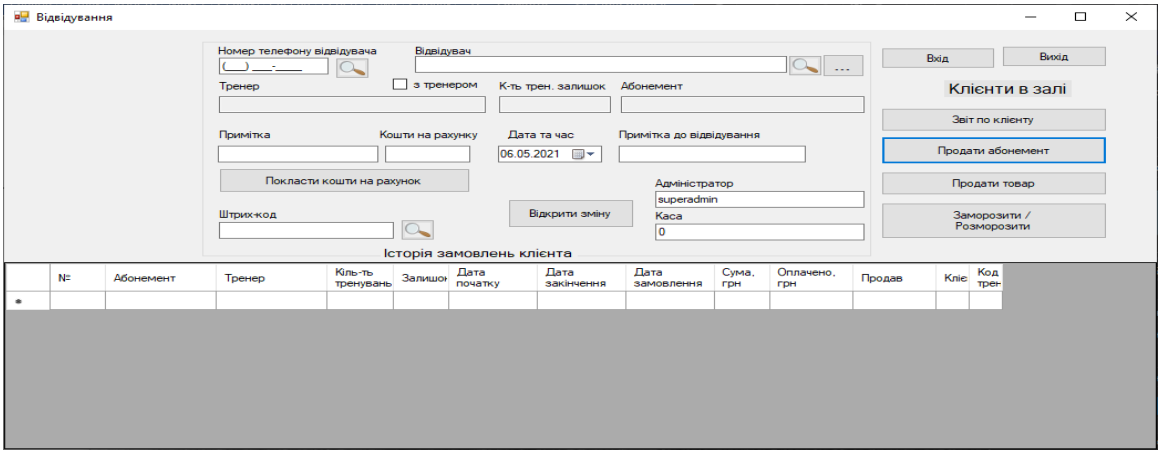


Рисунок 2.2.6 – Вікно «Відвідування»

Для обслуговування клієнта необхідно обрати його з переліку відвідувачів або ввести його прізвище у поле «Відвідувач», або ввести його номер телефону у поле «Номер телефону відвідувача», або ввести штрих-код карти клубу у поле «Штрих-код» на формі «Відвідування» та натиснути кнопку «Пошук». І потім натискати кнопки для обслуговування клієнта.

Якщо поля для пошуку клієнта будуть пустими, то при натисненні на кнопки обслуговування клієнта програма відкриє вікно з повідомленням про помилку, яке зображено на рисунку 2.2.7.

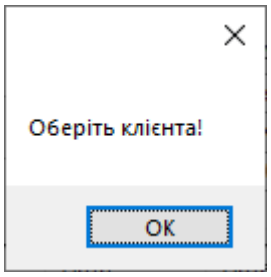


Рисунок 2.2.7 – Вікно з повідомленням про помилку

Наступним кроком було протестовано можливість продажу клієнтові абонементу та/або послуг тренера. Цю можливість реалізовано у вікні «Продаж абонементу», яке зображено на рисунку 2.2.8. На вікні «Продаж абонементу» розташовано наступні поля:

- «№ Замовлення» та «Відвідувач», які заповнюються автоматично під час запуску вікна;
- «Абонемент» та «Тренер», які заповнюються адміністратором;
- «Дата активації», «Дата закінчення» та «Вартість», які заповнюються автоматично після дій адміністратора.

Також на цьому вікні знаходяться кнопки для виконання оплати обраних послуг та редагування вже проданих послуг клієнтові.

Також на цьому вікні знаходиться таблиця для відображення вже куплених послуг клієнтом.

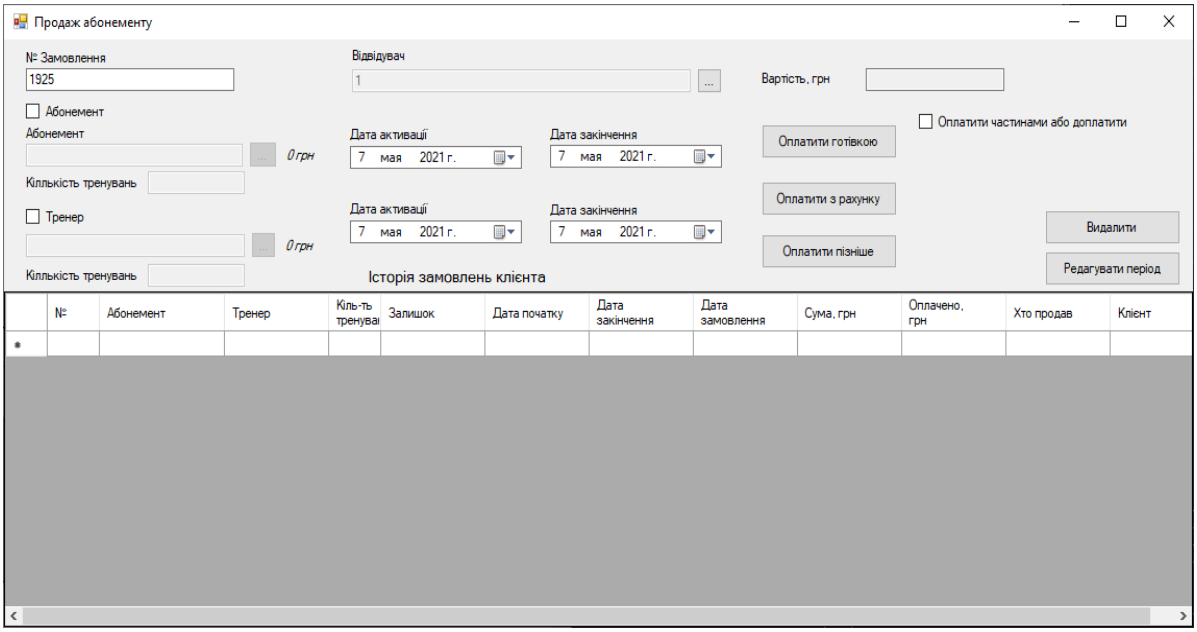


Рисунок 2.2.8 – Вікно «Продаж абонементу»

Після вибору послуг необхідно натиснути кнопку для оплати послуг, тоді у таблиці з’явиться новий запис. Якщо поля «Абонемент», «Тренер» або «Вартість» будуть пустими, тоді програма відкриє вікно з повідомленням про помилку, яке зображено на рисунку 2.2.9.

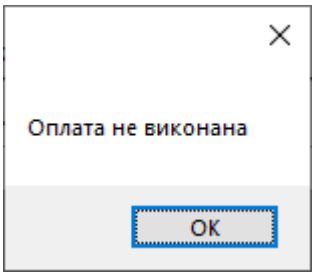


Рисунок 2.2.9 – Вікно з повідомленням про помилку

Наступним кроком було протестовано можливість продажу товару клієнтові. Цю можливість реалізовано у вікні «Продаж товару», яке зображено на рисунку 2.2.10. На вікні «Продаж абонементу» розташовано наступні поля:

- «№» та «Відвідувач», які заповнюються автоматично під час запуску вікна;
- «Продукт» та «Кількість», які заповнюються адміністратором;

- «Сумма» та «Вартість», які заповнюються автоматично після дій адміністратора.

Також на цьому вікні знаходяться кнопки для виконання оплати обраних товарів або видалення вже проданих товарів клієнтові та таблиця для відображення вже куплених товарів клієнтом.

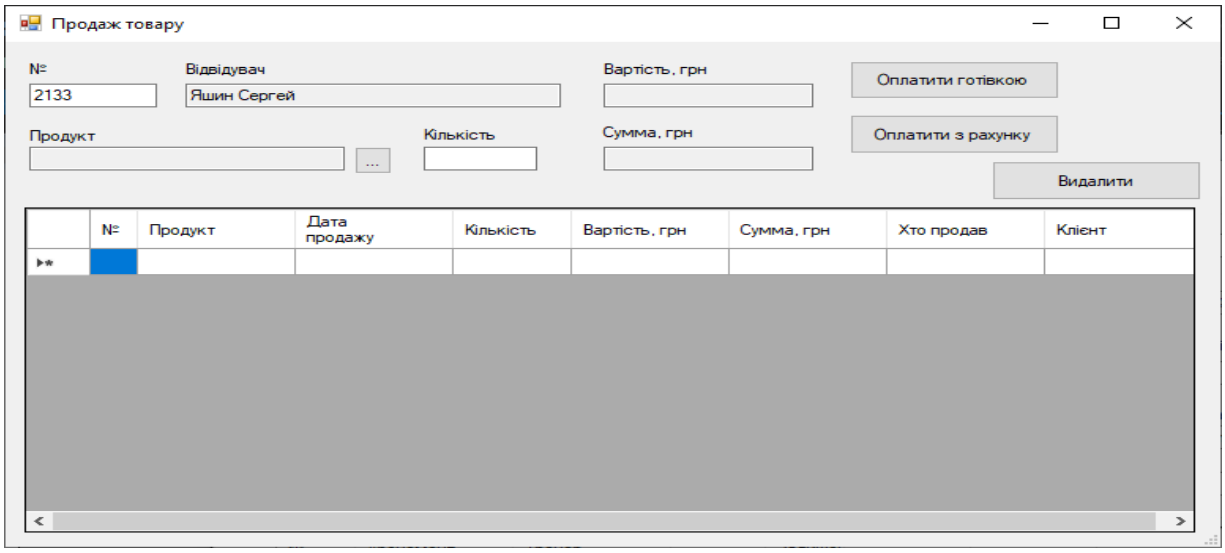


Рисунок 2.2.10 – Вікно «Продаж товару»

Після вибору товару необхідно натиснути кнопку для оплати товару, тоді у таблиці з’явиться новий запис. Якщо поля «Продукт» або «Вартість» будуть пустими, тоді програма відкриє вікно з повідомленням про помилку, яке зображено на рисунку 2.2.11.

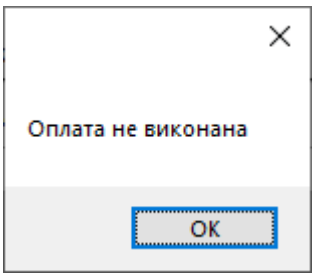


Рисунок 2.2.11 – Вікно з повідомленням про помилку

Наступним кроком було протестовано можливість «заморозки» послуг, які було попередньо продано клієнтові. Цю можливість реалізовано у вікні «Заморозити/ Розморозити», яке зображено на рисунку 2.2.12. На вікні «Заморозити/ Розморозити» розташовано наступні поля:

– «Абонемент», «Тренування» та «Дата закінчення (заявлена при покупці)», які заповнюються адміністратором з таблиці;
– «Кількість днів», які заповнює адміністратор;
– «Дата закінчення», яке заповнюється автоматично після заповнення поля «Кількість днів».

Також на цьому вікні знаходяться кнопки «Заморозити» та «Розморозити» для зміни статусу та подовження обраних послуг і таблиця для відображення дійсних послуг клієнта, прізвище та ім’я якого знаходиться в назві вікна.

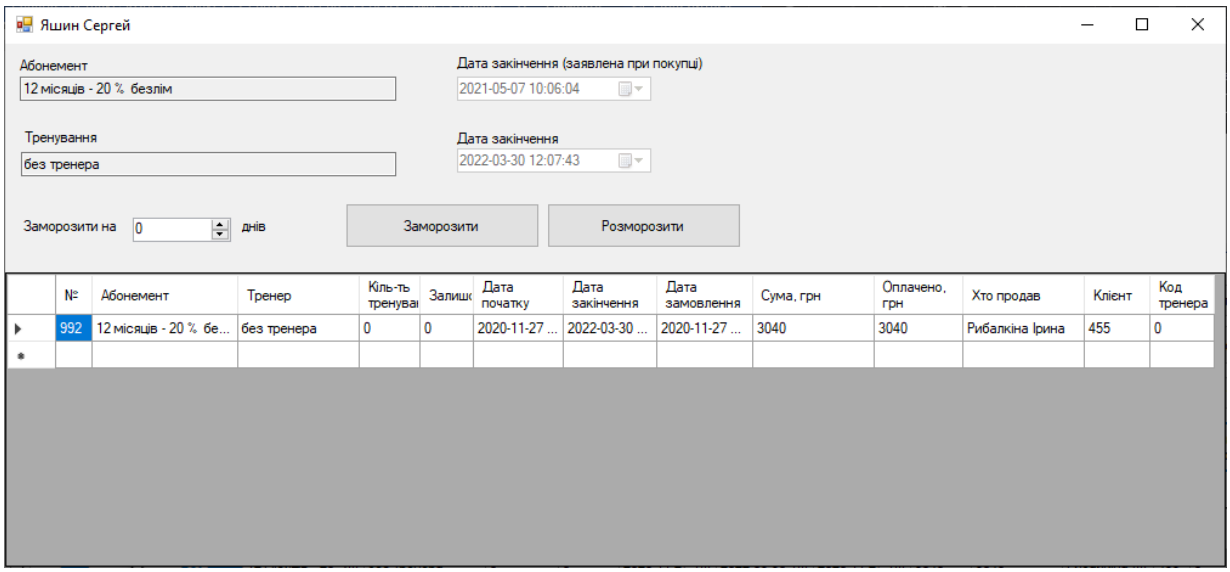


Рисунок 2.2.12 – Вікно «Заморозити/ Розморозити»

Наступним кроком було протестовано можливість додавання продукту до переліку продуктів, які пропонує клієнтам фітнес клуб. Для того, щоб додати продукт потрібно на головній формі викликати пункт «Довідники» - «Продукти». У вікні, що відкриється, яке зображено на рисунку 2.2.13, необхідно заповнити такі поля: «Код», «Назва» та «Ціна». У полі з підписом «Код» потрібно записати код нового продукту, який необхідно додати, у полі з підписом «Назва» необхідно

вписати назву продукту, у полі з підписом «Ціна» необхідно вписати ціну продукту.

Після заповнення полів натиснути кнопку «Додати», після чого дані в таблиці оновляться та з’явиться новий запис, в іншому випадку з’явиться повідомлення про помилку роботи системи. Помилка може бути у випадку, якщо заповнені не всі поля або вказано код продукту, який вже зарезервований для іншого продукту.

Для того, щоб оновити, тобто редагувати, певний рядок таблиці потрібно натиснути на нього. Потім всі поля заповняться відповідними значеннями з таблиці, яка знаходиться на формі. Потрібно відредагувати значення, що потребує зміни у відповідному полі на формі та натиснути кнопку «Редагувати». Після цього дані в таблиці оновляться, в іншому випадку з’явиться повідомлення про помилку та дані в таблиці залишаться незмінними.

Помилка може бути у випадку, якщо відредаговано код продукту, значення якого немає в таблиці.

Для того, щоб видалити певний рядок таблиці потрібно ввести код продукту в поле «Код». Або ж обрати його з таблиці, натиснувши на значення рядка таблиці, тоді значення в полі «Код» заповниться автоматично та натиснути кнопку «Видалити». Після цього дані в таблиці оновляться, в іншому випадку з’явиться повідомлення про помилку та дані в таблиці залишаться незмінними.

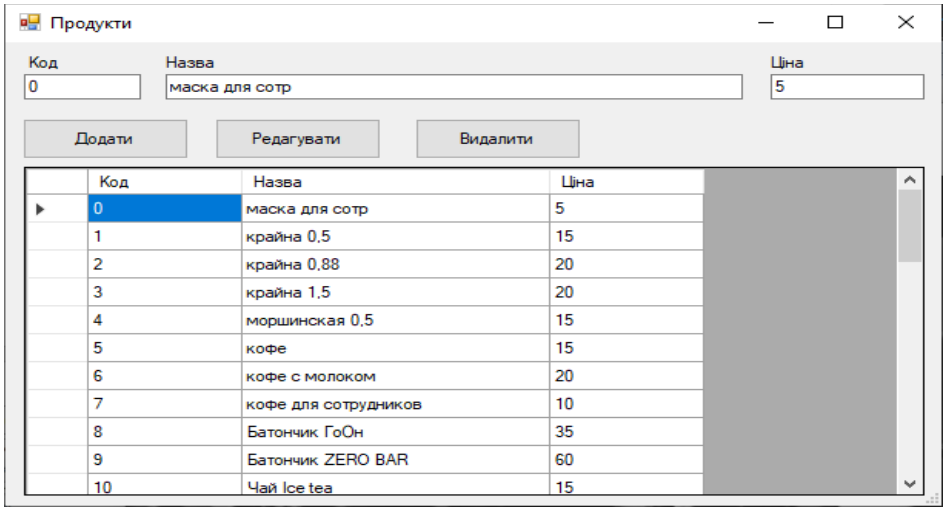


Рисунок 2.2.13 – Вікно «Продукти»

Наступним кроком було протестовано можливість додавання нового тренера до переліку тренерів, які є працівниками фітнес клубу. Для того, щоб додати нового тренера потрібно на головній формі викликати пункт «Довідники» - «Тренери». У вікні, що відкриється, яке зображено на рисунку 2.2.14, необхідно заповнити такі поля: «Прізвище Ім’я По-батькові», «Номер телефону», «Штрих-код», «Ставка» та «Розряд». У полі з підписом «Прізвище Ім’я По-батькові» потрібно записати прізвище, ім’я та по-батькові нового тренера, у полі з підписом «Номер телефону» необхідно вписати номер телефону тренера, у полі з підписом «Штрих-код» написати номер штрих-коду особистої карти тренера, у полі з підписом «Розряд» необхідно написати розряд тренера та у полі з підписом «Ставка» необхідно написати ціну за одне тренування з тренером, яка залежить від досвіду та розряду тренера.

Після заповнення полів натиснути кнопку «Додати», після чого дані в таблиці оновляться та з’явиться новий запис, в іншому випадку з’явиться повідомлення про помилку роботи системи. Помилка може бути у випадку, якщо заповнені не всі поля або вказано дані тренера, які вже зарезервовані для іншого тренера.

Для того, щоб оновити, тобто редагувати, певний рядок таблиці потрібно натиснути на нього. Потім всі поля заповняться відповідними значеннями з таблиці, яка знаходиться на формі. Потрібно відредагувати значення, що потребує зміни у відповідному полі на формі та натиснути кнопку «Редагувати». Після цього дані в таблиці оновляться, в іншому випадку з’явиться повідомлення про помилку та дані в таблиці залишаться незмінними.

Помилка може бути у випадку, якщо відредаговано дані тренера, значення якого немає в таблиці.

Для того, щоб видалити певний рядок таблиці потрібно обрати його з таблиці, натиснувши на значення рядка таблиці, тоді значення в полі «Прізвище Ім’я По-батькові» заповниться автоматично та натиснути кнопку «Видалити». Після цього дані в таблиці оновляться, в іншому випадку з’явиться повідомлення про помилку та дані в таблиці залишаться незмінними.

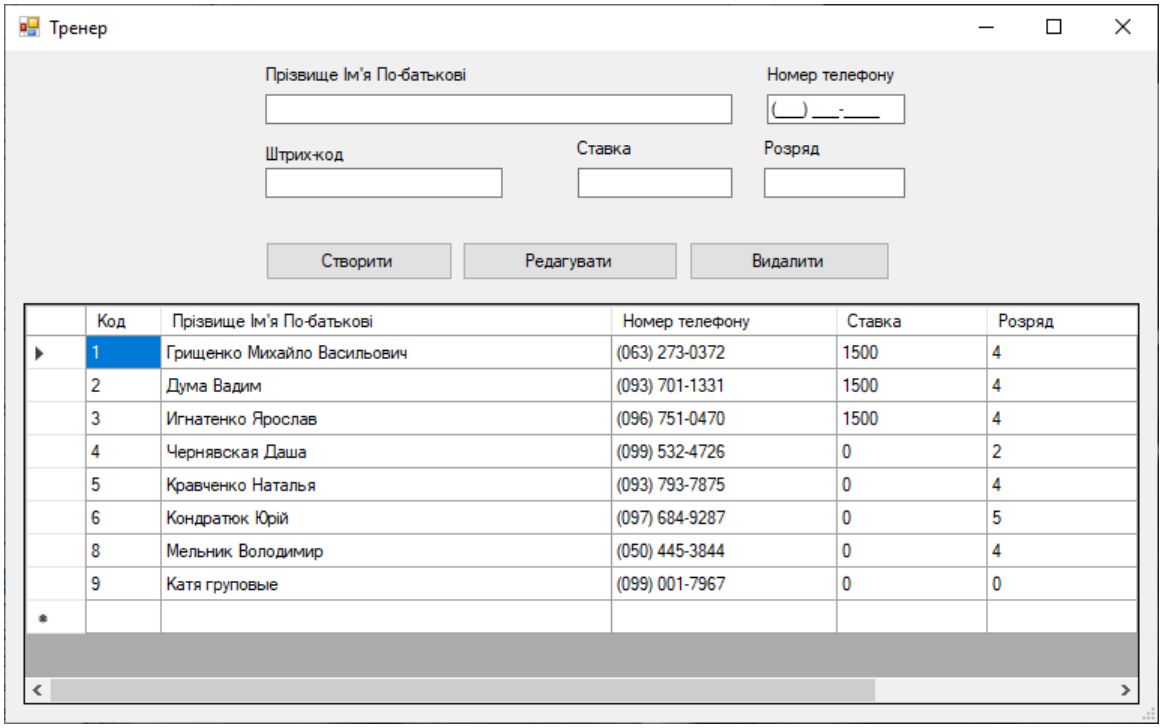


Рисунок 2.2.14 – Вікно «Тренер»

Наступним кроком було протестовано можливість додавання нового абонемента до переліку абонементів, які продає клієнтам фітнес клуб. Для того, щоб додати абонемент потрібно на головній формі викликати пункт «Довідники» - «Абонементи». У вікні, що відкриється, яке зображено на рисунку 2.2.15, необхідно заповнити такі поля: «Код», «Назва», «Період», «Ціна» та «Кількість тренувань». У полі з підписом «Код» необхідно написати код нового абонемента, у полі з підписом «Назва» потрібно написати назву абонемента, у полі з підписом «Період» необхідно вписати період абонемента у днях, у полі з підписом «Ціна» необхідно написати ціну абонемента у гривнях, у полі з підписом «Кількість тренувань» необхідно написати кількість тренувань, на яку розрахований абонемент.

Після заповнення полів натиснути кнопку «Додати», після чого дані в таблиці оновляться та з’явиться новий запис, в іншому випадку з’явиться повідомлення про помилку роботи системи. Помилка може бути у випадку, якщо заповнені не всі поля або вказано код абонемента, який вже зарезервований для іншого абонемента.

Для того, щоб оновити, тобто редагувати, певний рядок таблиці потрібно натиснути на нього. Потім всі поля заповняться відповідними значеннями з таблиці, яка знаходиться на формі. Потрібно відредагувати значення, що потребує зміни у відповідному полі на формі та натиснути кнопку «Редагувати». Після цього дані в таблиці оновляться, в іншому випадку з’явиться повідомлення про помилку та дані в таблиці залишаться незмінними.

Помилка може бути у випадку, якщо відредаговано абонемент, значення якого немає в таблиці.

Для того, щоб видалити певний рядок таблиці потрібно ввести код абонемента в поле «Код». Або ж обрати його з таблиці, натиснувши на значення рядка таблиці, тоді значення в полі «Код» заповниться автоматично та натиснути кнопку «Видалити». Після цього дані в таблиці оновляться, в іншому випадку з’явиться повідомлення про помилку та дані в таблиці залишаться незмінними.

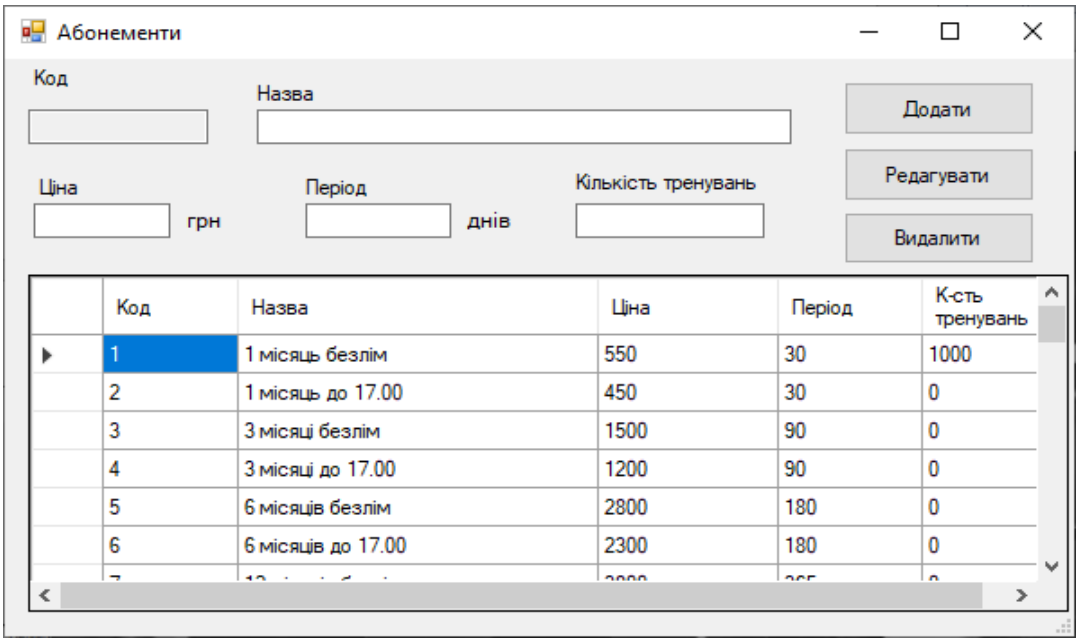


Рисунок 2.2.15 – Вікно «Абонементи»

Наступним кроком було протестовано можливість додавання нового адміністратора до переліку адміністраторів, які є працівниками фітнес клубу. Для того, щоб додати адміністратора потрібно на головній формі викликати пункт «Довідники» - «Працівники». У вікні, що відкриється, яке зображено на рисунку

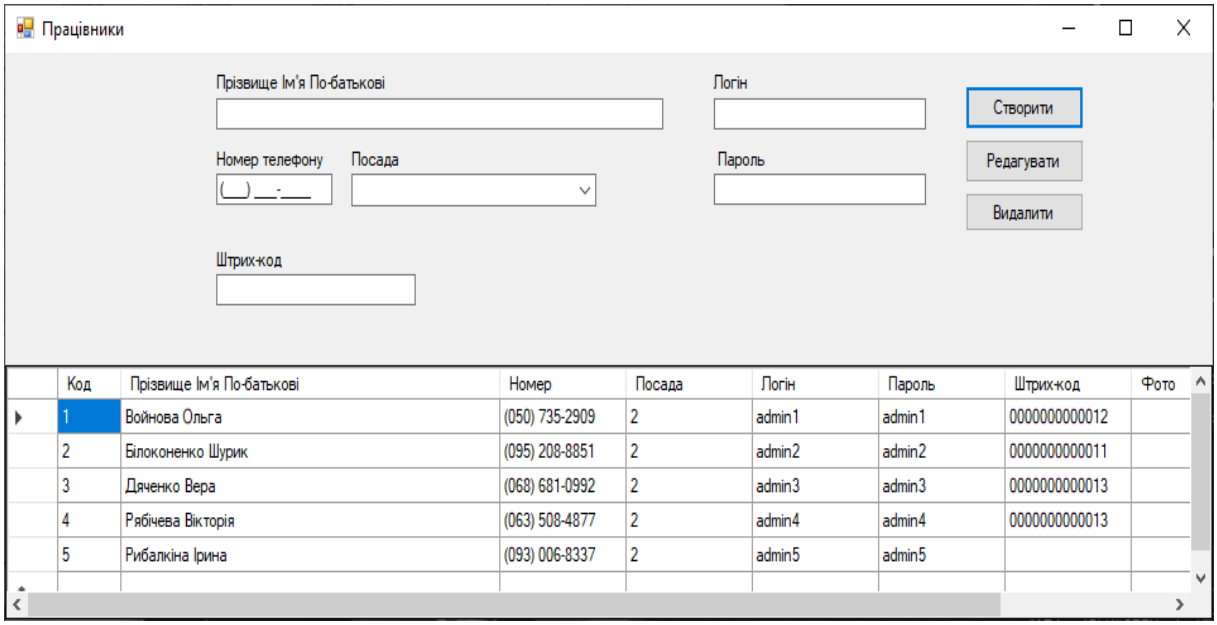
2.2.16, необхідно заповнити такі поля: «Прізвище Ім’я По-батькові», «Номер телефону», «Штрих-код», «Посада», «Логін» та «Пароль». У полі з підписом «Прізвище Ім’я По-батькові» потрібно записати прізвище, ім’я та по-батькові нового адміністратора, у полі з підписом «Номер телефону» необхідно вписати номер телефону адміністратора, у полі з підписом «Штрих-код» написати номер штрих-коду особистої карти адміністратора, у полі з підписом «Посада» необхідно обрати запис адміністратор з випадаючого списку, у полі з надписом «Логін» необхідно ввести логін для адміністратора та у полі з підписом «Пароль» необхідно написати пароль для адміністратора, який він буде використовувати для входу до системи.

Після заповнення всіх полів натиснути кнопку «Додати», після чого дані в таблиці оновляться та з’явиться новий запис, в іншому випадку з’явиться повідомлення про помилку роботи системи. Помилка може бути у випадку, якщо заповнені не всі поля або вказано штрих-код адміністратора, який вже зарезервований для іншого адміністратора.

Для того, щоб оновити, тобто редагувати, певний рядок таблиці потрібно натиснути на нього. Потім всі поля заповняться відповідними значеннями з таблиці, яка знаходиться на формі. Потрібно відредагувати значення, що потребує зміни у відповідному полі на формі та натиснути кнопку «Редагувати». Після цього дані в таблиці оновляться, в іншому випадку з’явиться повідомлення про помилку та дані в таблиці залишаться незмінними.

Помилка може бути у випадку, якщо відредаговано дані адміністратора, значення якого немає в таблиці.

Для того, щоб видалити певний рядок таблиці потрібно обрати його з таблиці, натиснувши на значення рядка таблиці, тоді значення в полі «Прізвище Ім’я По-батькові» заповниться автоматично та натиснути кнопку «Видалити». Після цього дані в таблиці оновляться, в іншому випадку з’явиться повідомлення про помилку та дані в таблиці залишаться незмінними.



Код	Прізвище Ім'я По-батькові	Номер	Посада	Логін	Пароль	Штрих-код	Фото
1	Войнова Ольга	(050) 735-2909	2	admin1	admin1	0000000000012	
2	Білоконенко Шурик	(095) 208-8851	2	admin2	admin2	0000000000011	
3	Дяченко Вера	(068) 681-0992	2	admin3	admin3	0000000000013	
4	Рябічева Вікторія	(063) 508-4877	2	admin4	admin4	0000000000013	
5	Рибалкіна Ірина	(093) 006-8337	2	admin5	admin5		

Рисунок 2.2.16 – Вікно «Працівники»

Наступним кроком було протестовано можливість додавання нового запису до таблиці вартості тренування з тренером. Для того, щоб додати запис потрібно на головній формі викликати пункт «Довідники» - «Вартість послуг тренера». У вікні, що відкриється, яке зображено на рисунку 2.2.17, необхідно заповнити такі поля: «Категорія», «Кількість тренувань», «Опис», «Ціна» та «Період». У полі з підписом «Категорія» потрібно записати категорію тренера, у полі з підписом «Кількість тренувань» необхідно написати кількість тренувань з тренером, у полі з підписом «Опис» описати вид тренувань з тренером, у полі з підписом «Ціна» необхідно написати ціну у гривнях, у полі з підписом «Період» необхідно написати кількість днів, тобто строк дії цієї послуги.

Після заповнення полів натиснути кнопку «Додати», після чого дані в таблиці оновляться та з’явиться новий запис, в іншому випадку з’явиться повідомлення про помилку роботи системи. Помилка може бути у випадку, якщо заповнені не всі поля або вказано категорію та кількість тренувань, які вже зарезервовані для іншого запису в таблиці.

Для того, щоб оновити, тобто редагувати, певний рядок таблиці потрібно натиснути на нього. Потім всі поля заповняться відповідними значеннями з таблиці, яка знаходиться на формі. Потрібно відредагувати значення, що потребує

зміни у відповідному полі на формі та натиснути кнопку «Редагувати». Після цього дані в таблиці оновляться, в іншому випадку з’явиться повідомлення про помилку та дані в таблиці залишаться незмінними.

Помилка може бути у випадку, якщо відредаговано код тренера, значення якого немає в таблиці.

Для того, щоб видалити певний рядок таблиці потрібно обрати його з таблиці, натиснувши на значення рядка таблиці, тоді значення в полях «Категорія» та «Кількість тренувань» заповняться автоматично та натиснути кнопку «Видалити». Після цього дані в таблиці оновляться, в іншому випадку з’явиться повідомлення про помилку та дані в таблиці залишаться незмінними.

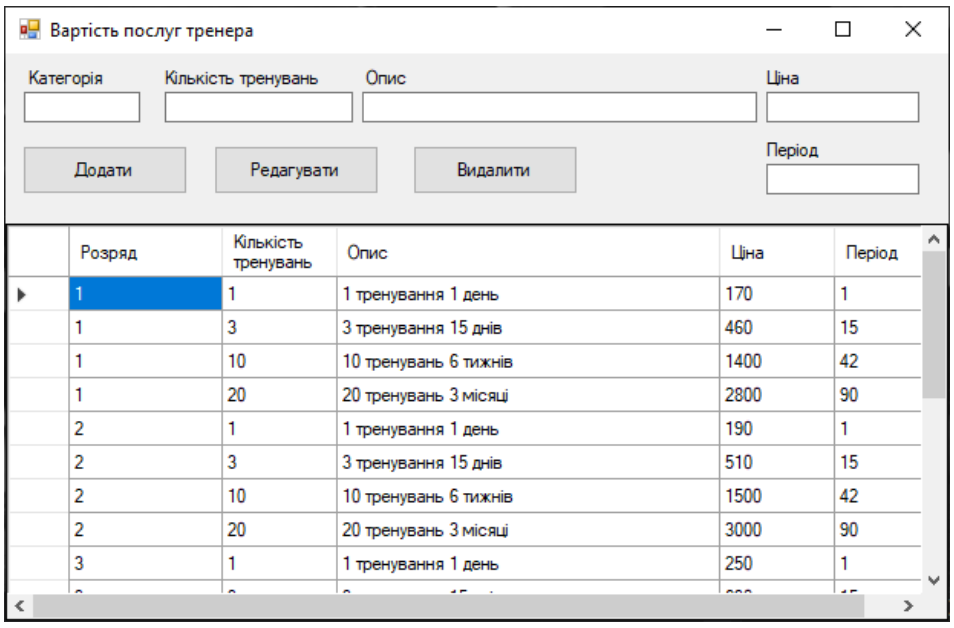


Рисунок 2.2.17 – Вікно «Вартість послуг тренера»

Наступним кроком було протестовано можливість додавання нового клієнта до переліку клієнтів, які є відвідувачами фітнес клубу. Для того, щоб додати клієнта потрібно на головній формі викликати пункт «Довідники» - «Відвідувачі». У вікні, що відкриється, яке зображено на рисунку 2.2.18 та рисунку 2.2.19 відображене вікно продаж товарів, необхідно заповнити такі поля: «Прізвище Ім’я По-батькові», «Номер телефону», «Штрих-код», «Кошти на рахунку», «Примітка» та «Дата реєстрації». У полі з підписом «Прізвище Ім’я По-батькові» потрібно

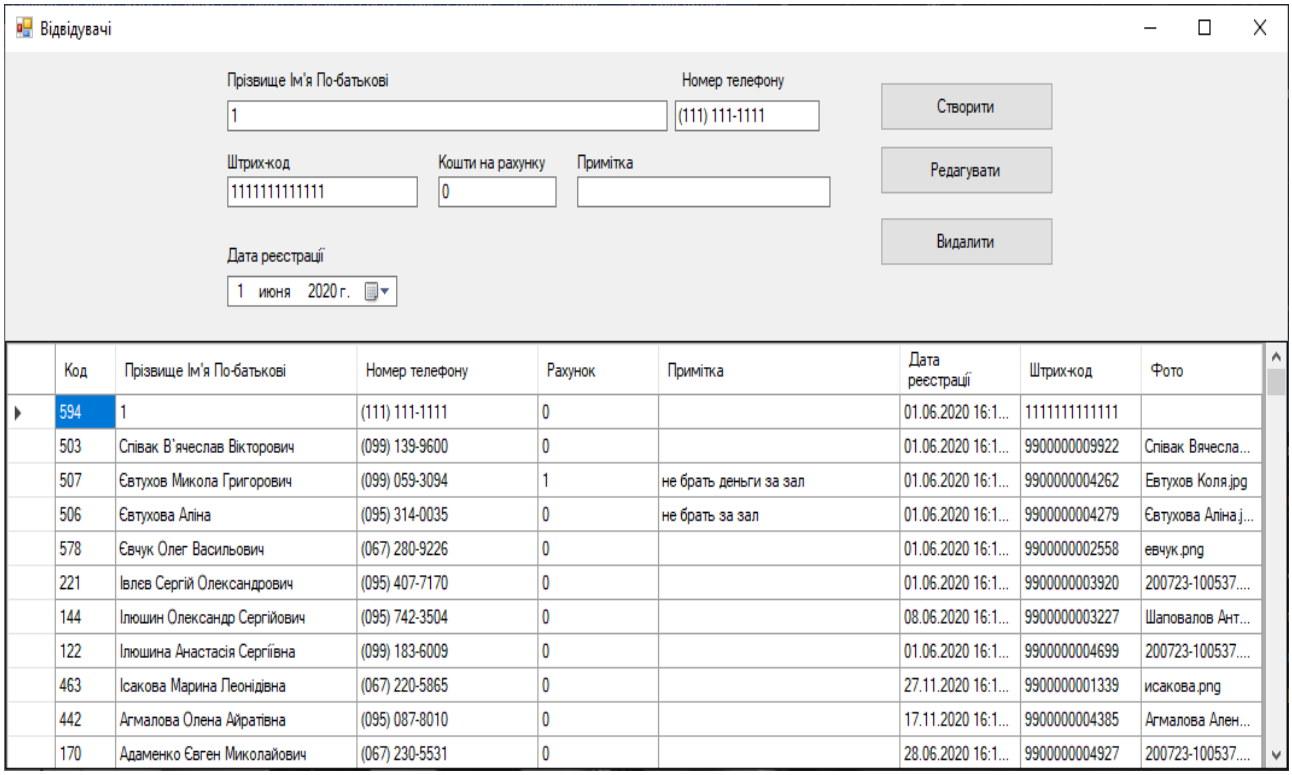
записати прізвище, ім’я та по-батькові нового клієнта, у полі з підписом «Номер телефону» необхідно вписати номер телефону клієнта, у полі з підписом «Штрих-код» написати номер штрих-коду особистої карти клієнта, у полі з підписом «Кошти на рахунку» необхідно написати суму коштів, у полі з підписом «Примітка» написати примітку для клієнта та у полі з підписом «Дата реєстрації» необхідно написати дату реєстрації, заповнюється автоматично.

Після заповнення полів натиснути кнопку «Додати», після чого дані в таблиці оновляться та з’явиться новий запис, в іншому випадку з’явиться повідомлення про помилку роботи системи. Помилка може бути у випадку, якщо заповнені не всі поля або вказано дані клієнта, які вже зарезервовані для іншого клієнта.

Для того, щоб оновити, тобто редагувати, певний рядок таблиці потрібно натиснути на нього. Потім всі поля заповняться відповідними значеннями з таблиці, яка знаходиться на формі. Потрібно відредагувати значення, що потребує зміни у відповідному полі на формі та натиснути кнопку «Редагувати». Після цього дані в таблиці оновляться, в іншому випадку з’явиться повідомлення про помилку та дані в таблиці залишаться незмінними.

Помилка може бути у випадку, якщо відредаговано дані клієнта, значення якого немає в таблиці.

Для того, щоб видалити певний рядок таблиці потрібно обрати його з таблиці, натиснувши на значення рядка таблиці, тоді значення в полі «Прізвище Ім’я По-батькові» заповниться автоматично та натиснути кнопку «Видалити». Після цього дані в таблиці оновляться, в іншому випадку з’явиться повідомлення про помилку та дані в таблиці залишаться незмінними.



Код	Прізвище Ім'я По-батькові	Номер телефону	Рахунок	Примітка	Дата реєстрації	Штрих-код	Фото
594	1	(111) 111-1111	0		01.06.2020 16:1...	1111111111111	
503	Співак В'ячеслав Вікторович	(099) 139-9600	0		01.06.2020 16:1...	9900000009922	Співак Вячесла...
507	Євтухов Микола Григорович	(099) 059-3094	1	не брать деньги за зал	01.06.2020 16:1...	9900000004262	Евтухов Коля.jpg
506	Євтухова Аліна	(095) 314-0035	0	не брать за зал	01.06.2020 16:1...	9900000004279	Євтухова Аліна.j...
578	Євчук Олег Васильович	(067) 280-9226	0		01.06.2020 16:1...	9900000002558	евчук.png
221	Івлєв Сергій Олександрович	(095) 407-7170	0		01.06.2020 16:1...	9900000003920	200723-100537....
144	Ілюшин Олександр Сергійович	(095) 742-3504	0		08.06.2020 16:1...	9900000003227	Шаповалов Ант...
122	Ілюшина Анастасія Сергіївна	(099) 183-6009	0		01.06.2020 16:1...	9900000004699	200723-100537....
463	Ісакова Марина Леонідівна	(067) 220-5865	0		27.11.2020 16:1...	9900000001339	исакова.png
442	Агмалова Олена Айратівна	(095) 087-8010	0		17.11.2020 16:1...	9900000004385	Агмалова Ален...
170	Адаменко Євген Миколайович	(067) 230-5531	0		28.06.2020 16:1...	9900000004927	200723-100537....

Рисунок 2.2.18 – Вікно «Відвідувачі»

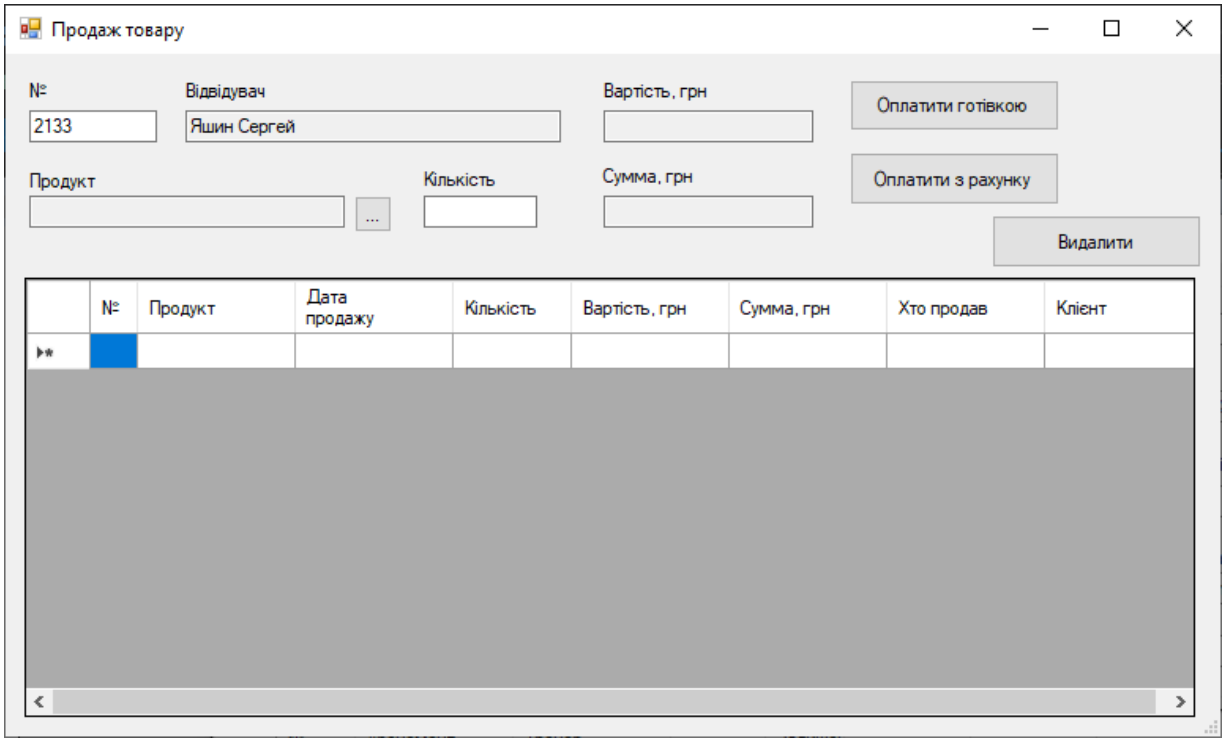


Рисунок 2.2.19 – Вікно «Продаж товару»

3. ОРГАНІЗАЦІЙНО - ЕКОНОМІЧНА ЧАСТИНА

3.1 Оцінка вартості програмного продукту

Прогрес в економіці, промисловості, науці і техніці, у сфері програмування в даний час багато в чому залежить від масового впровадження обчислювальної техніки та потужних пристроїв. Будь-який комп'ютер в процесі роботи використовує засоби програмного забезпечення.

Розробка програмних засобів вимагає певних інтелектуальних і трудових витрат, а також обов'язкового використання комп'ютерної техніки, що визначає особливості розрахунку собівартості програмного продукту.

Оцінка вартості програмування розглядається як оцінка затрат коштів і часу, необхідних для виконання етапів проекту.

Вартість програмного продукту включає:

1. собівартість програмного продукту;
2. плановий прибуток.

3.1.1 Розрахунок часу

Загальний час на створення програми складається з різних компонентів. Структура загального часу на створення програмного продукту представлена в таблиці 3.1.1

Таблиця 3.1.1 – Структура загального часу на створення програмного продукту.

№ етапу	Позначення часу даного етапу	Зміст етапу
1	$T_{ПО}$	Підготовка опису завдання
2	$T_{О}$	Опис завдання
3	$T_{А}$	Розробка алгоритмів

№ етапу	Позначення часу даного етапу	Зміст етапу
4	$T_{БС}$	Розробка блок-схеми алгоритмів
5	$T_{ПІ}$	Проектування інтерфейсу
6	$T_{Н}$	Написання програми (кодування)
7	$T_{ВТ}$	Відладка і тестування програми
8	$T_{Д}$	Оформлення документації, інструкції користувачеві, записки пояснення

Час розраховується в людино-годинах, причому $T_{ПО}$ та$T_{Д}$ береться по фактично відпрацьованому часу, а час останніх етапів визначається розрахунковий по умовному числу команд Q.

Умовне число команд Q визначається за формулою

$$Q= q \cdot з, \qquad (3.1)$$

де q - коефіцієнт, що враховує умовне число команд залежно від типу завдання. Значення коефіцієнта q вибирається з таблиці 3.2.

Значення коефіцієнта з вибирається з таблиці 3.3.

Таблиця 3.1.2 – Значення коефіцієнта q.

Тип завдання	Значення коефіцієнта q
Завдання обліку	1400…1500
Завдання оперативного управління	1500…1700
Завдання планування	3000…3500
Багатоваріантні завдання	4500…5000
Комплексні завдання	5000…5500

Для даного завдання приймається коефіцієнт q = 4500.

Програмні продукти за мірою новизни можуть бути віднесені до однієї з чотирьох груп:

- група А - розробка принципово нових завдань;
- група Б - розробка оригінальних програм;
- група В - розробка програм з використанням типових рішень;
- група Г - разове типове завдання.

Для даного завдання міра новизни: В

За складністю програмні продукти можуть бути віднесені до однієї з трьох груп:

1 алгоритми оптимізації і моделювання систем;

2 завдання обліку, звітності і статистики;

3 стандартні алгоритми.

Дане завдання може бути віднесено до третьої групи складності високого рівня.

Коефіцієнт **(** з **)** визначається з таблиці 3.3 в залежності від складності і міри новизни.

Таблиця 3.1.3 – Значення коефіцієнта з.

Мова програмування	Група складності	Міра новизни			
		А	Б	В	Г
Високого рівня	1	1,38	1,26	1,15	0,69
	2	1,30	1,19	1,08	0,65
	3	1,20	1,10	1,00	0,60
Низького рівня	1	1,58	1,45	1,32	0,79
	2	1,49	1,37	1,24	0,74
	3	1,38	1,26	1,15	0,69

Для даного завдання коефіцієнт з = 1,00.

За формулою 3.1 визначається умовне число команд Q.

$$Q = 4500 \cdot 1{,}00 = 4500 \text{ чол / годин.}$$

Визначається час, витрачений на кожен етап створення програмного продукту:

- $T_{ПО}$ (час на підготовку опису завдання), береться по факту і для даного завдання

$$T_{ПО} = 40 \text{ чол / годин.}$$

- $T_О$ (час на опис завдання) визначається за формулою

$$T_О = Q \cdot B / (50 \cdot K), \qquad (3.2)$$

де В - коефіцієнт обліку змін завдання. Коефіцієнт В залежно від складності завдання і числа змін вибирається в інтервалі від 1,2 до 1,5. Для даного завдання В = 1,3

К - коефіцієнт, що враховує кваліфікацію програміста. Значення коефіцієнта вибирається з таблиці 3.4.

Таблиця 3.1.4 – Значення коефіцієнта К.

Стаж програміста	Значення коефіцієнта К
до 2-х років	0,8
від 2 до 3 років	1,0
від 3 до 5 років	1,1…1,2
від 5 до 10 років	1,2…1,3
понад 10 років	1,3…1,5

В даному випадку коефіцієнт К = 0,8.

За формулою 3.2 підраховується час на опис завдання.

$$T_О = 4500 \cdot 1{,}3 / (50 \cdot 0{,}8) = 146{,}25 \text{ [чол. / годин].}$$

- T_A (час на розробку алгоритмів) розраховується за формулою

$$T_A = Q / (50 * K), \quad (3.3)$$

За формулою 3.3 підраховується час на розробку алгоритмів.

$$T_A = 4500 / (50 \cdot 0,8) = 112,5$$

[чол. / годин].

- $T_{БС}$ (час на розробку блок-схеми) визначається аналогічно ТА за формулою 3.3 і складає

$$T_{БС} = 4500 / (50 \cdot 0,8) = 112,5 \text{ [чол. / годин]}.$$

- T_H (час написання програми на мові програмування) визначається за формулою

$$T_H = Q \cdot 1,5 / (50 \cdot K), \quad (3.4)$$

За формулою 3.4 підраховується час написання програми на мові програмування.

$$T_H = 4500 \cdot 1,5 / (50 \cdot 0,8) = 168,75 \text{ [чол. / годин]}.$$

- $T_{ПІ}$ (час на проектування інтерфейсу) визначається за формулою

$$T_{ПІ} = Q / 50, \quad (3.5)$$

За формулою 3.5 підраховується час на проектування інтерфейсу програми.

$$T_{ПІ} = 4500 / 50 = 90 \text{ [чол. / годин]}.$$

- $T_{ВТ}$ (час відладки і тестування програми) визначається за формулою

$$T_{ВТ} = Q \cdot 4,2 / (50 \cdot K), \quad (3.6)$$

За формулою 3.6 підраховується час відладки і тестування програми.

$$T_{ВТ} = 4500 \cdot 4,2 / (50 \cdot 0,8) = 472,5 \text{ [чол. / годин]}.$$

- $T_Д$ (час на оформлення документації), береться по факту і для даного завдання

$$T_Д = 40 \text{ чол / годин}.$$

Підраховується загальний час на створення програмного продукту за формулою:

$$Т = Т_{ПО} + Т_{О} + Т_{А} + Т_{БС} + Т_{Н} + Т_{ПІ} + Т_{ВТ} + Т_{Д}, \quad (3.7)$$

Т = 40+146,25+112,5+112,5+168,75+90+472,5+40=1182,5 [чол. / годин].

3.1.2 Розрахунок заробітної плати виконавця робіт із створення програмного продукту

Основна ЗП визначається за формулою:

$$ЗПосн.=(ЗП\ 1р \cdot Кт \cdot Т)/(Чр \cdot tр.д.)\cdot(1 + П/100), \quad (3.8)$$

де ЗП 1р - місячна зарплата 1-го розряду, грн.;

Кт - тарифний коефіцієнт, відповідний розряду тарифної сітки, за яким працює виконавець;

Т - загальний час на створення програмного продукту, чол. / годин;

Чр - кількість робочих днів в місяць;

tр.д. - тривалість робочого дня в годинах;

П - відсоток премії.

Для даного завдання:

- ЗП 1р = 200,00 грн.;
- виконавець має 7 розряд;
- для 7 розряду тарифний коефіцієнт Кт = 1,54;
- загальний час створення програмного продукту Т = 1182,5 чол. / годин;
- кількість робочих днів Чр = 14;
- тривалість робочого дня tр.д. = 8 годин;
- відсоток премії П=15%.

Таким чином, визначаємо основну заробітну плату виконавця робіт із створення програмного продукту.

ЗП осн. = (200,00·1,54·1182,5)/(14·8)·(1+15/100)=2827,70 [грн].

Додаткова заробітна плата береться у розмірі 15 % від основної і розраховується за формулою:

$$ЗП\ дод. = ЗП\ осн. \cdot 15 / 100, \quad (3.9)$$

$$ЗП\ дод. = 2827{,}70 \cdot 15 / 100 = 424{,}00\ [грн].$$

Загальна заробітна плата розраховується за формулою:

$$ЗП\ загальна = ЗП\ осн + ЗП\ дод., \quad (3.10)$$

$$ЗП\ загальна = 2827{,}70 + 424{,}00 = 3251{,}70\ [грн].$$

3.1.3 Розрахунок нарахувань на заробітну плату (єдиного соціального внеску)

Єдиний соціальний внесок нараховується на заробітну плату за формулою 3.11 і складає 35,6 % від загальної заробітної плати

$$ЄСВ = ЗП\ загальна \cdot 22 / 100, \quad (3.11)$$

$$ЄСВ = 3251{,}70 \cdot 22 / 100 = 715{,}35\ [грн].$$

3.1.4 Розрахунок витрат на збереження та експлуатацію ПЕОМ, що відносяться до даного програмного продукту

Витрати на збереження та експлуатацію ПЕОМ визначаються у розмірі 130% від основної заробітної плати працівників, що забезпечують функціонування ПЕОМ.

До таких витрат відносяться витрати на:

- основну заробітну плату працівників, що забезпечують функціонування ПЕОМ (інженер-електронник; системний програміст; оператор);
- основна ЗП адміністративного і допоміжного персоналу;
- амортизаційні відрахування;
- витрати на електричну енергію;
- витрати на матеріали (до їх числа входять диски, картриджі і папір для принтерів та інше);
- витрати на профілактику ПЕОМ з периферією;
- витрати на опалювання виробничих площ;
- витрати на обслуговування виробничих площ;

- інші виробничі витрати.

Розрахунок витрат на збереження та експлуатацію ПЕОМ, що відносяться до даного програмного продукту за формулою

$$В\ зб.\ експл. = ЗП\ осн. \cdot 130 / 100, \quad (3.12)$$

$$В\ зб.\ експл. = 3521{,}70 \cdot 130 / 100 = 4578{,}20\ [грн].$$

3.2 Розрахунок собівартості програмного продукту

У собівартість програмного продукту входять наступні елементи:

- основна заробітна плата виконавця робіт із створення програмного продукту;
- додаткова заробітна плата виконавця робіт із створення програмного продукту;
- нарахування на заробітну плату (єдиний соціальний внесок);
- витрати на збереження та експлуатацію ПЕОМ, що відносяться до даного програмного продукту;
- інші витрати.

Інші витрати складають 10% від суми перших чотирьох елементів і розраховуються за формулою

$$І.вит.=(ЗП\ осн. + ЗП\ дод. + ЄСВ + В\ зб.\ експл.) \cdot 10 / 100 \quad (3.13)$$

$$І.вит.= (2827{,}70 + 424{,}00 + 715{,}35 + 4578{,}20) \cdot 10 / 100 = 854{,}50\ [грн].$$

Визначається собівартість програмного продукту за формулою

$$Сп.п. = ЗП\ осн. + ЗП\ дод. + ЄСВ + В\ зб.експл. + І.вит. \quad (3.14)$$

$$Сп.п. = 2827{,}70 + 424{,}00 + 715{,}35 + 4578{,}20 + 854{,}50 = 9399{,}75\ [грн].$$

Структура собівартості програмного продукту відображається в таблиці 3.2.1.

Таблиця 3.2.1 – Структура собівартості програмного продукту

Елементи структури	Сума грн.	Питома вага %
1 Основна заробітна плата виконавця	2827,70	30
2 Додаткова заробітна плата виконавця	424,00	4
3 Нарахування на заробітну плату (ЄСВ)	715,35	8
4 Витрати збереження та експлуатацію ПП	4578,20	49
5 Інші витрати	854,50	9
Собівартість програмного продукту	9399,75	100

3.3 Розрахунок ціни програмного продукту

Ціна програмного продукту складається з декількох компонентів і розраховуються за формулою

$$Ц = Сп.п. + П + ПДВ, \quad (3.15)$$

де Сп.п.- собівартість програмного продукту, грн.;

П - плановий прибуток, грн.;

ПДВ - податок на додану вартість, грн.

П – прибуток складає 40% від повної собівартості програмного продукту і розраховуються за формулою

$$П = Сп.п. \cdot 40 / 100 \quad (3.16)$$

$$П = 9399,75 \cdot 40 / 100 = 3759,90 \text{ [грн]}.$$

ПДВ - податок на додану вартість, який береться у розмірі 20% від суми собівартості і прибутку розраховуються за формулою

$$ПДВ = (Сп.п. + П) \cdot 20 / 100, \quad (3.17)$$

$$ПДВ = (9399,75+ 3759,90) \cdot 20 /100 = 939,90[грн].$$

За формулою 3.15 визначається ціна програмного продукту.

$$Ц = 9399,75+3759,90+939,90= 14\ 099,55[грн].$$

3.4 Зведена таблиця показників

Результати розрахунків відображаються в підсумкової таблиці 3.4.1

Таблиця 3.4.1 – Результати розрахунків

Найменування показника	Сума, грн.
Собівартість програмного продукту	9 399,75
Прибуток	3 759,90
ПДВ	939,90
Ціна програмного продукту	14 099,55

За результатами розрахунків ми можемо побачити, що для собівартість програмного продукту складає 9 399,75грн. Якщо ми очікуємо прибуток у 40% собівартості, то він має складати 3 759,90грн, а ПДВ на суму прибутку та собівартості буде нараховано у кількості 939,90грн. За сумою цих трьох результатів ми можемо вивести кінцеву ціну програмного продукту, що складає 14 099,55грн

4 БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ

4.1 Вимоги до приміщень і організації робочих місць

Загальні вимоги. Облаштування робочих місць, обладнаних відео терміналами, повинно забезпечувати:

- належні умови освітлення приміщення та робочого місця, відсутність відблисків від робочої поверхні;

- оптимальні параметри мікроклімату (температура, відносна вологість, швидкість руху, рівень іонізації повітря);

- належні ергономічні характеристики основних елементів робочого місця;

- також потрібно враховувати такі небезпечні та шкідливі фактори:

- наявність шуму та вібрації;

- м’яке рентгенівське випромінювання;

- електромагнітне випромінювання;

- ультрафіолетове та ультрачервоне випромінювання;

- електростатичне поле між екраном та оператором;

- наявність пилу, озону, оксидів азоту й аероіонізації.

Будівлі та приміщення, в яких експлуатують ЕОМ та виконують їх обслуговування, налагодження і ремонт, повинні відповідати основним вимогам:

– НиП 2.09.02-85 „Производственные здания”;

– СНип 2.09.04-87 „Административные и бытовые здания”;

– „Правил устройства электроустановок”, затверджених Головдерженергонаглядом СРСР 1984 р. (ПВЕ);

– „Правил технической эксплуатации электроустановок потребителей”, затверджених Головдерженергонаглядом СРСР 21.12.84 (ПТЕ);

– Правил безпечної експлуатації електроустановок споживачів, затверджених наказом Держнаглядохоронпраці 09.01.98 №4, зареєстрованих у Мін’юсті України 10.02.98 № 93/2533 (ПБЕ);

– СНиП 2.01.02-85 „Противопожарные нормы”;

– ГОСТ 12.1.004 „ССБТ. Пожарная безопасность. Общин требования безопасности”;

– Правил пожежної безпеки в Україні, затверджених наказом Управління Державної пожежної охорони МВС України від 14.06.95 № 400, зареєстрованих в Міністерстві юстиції України 14.07.95 за № 219/755;

– СНиП 2.08.02-89 „Общественные здания и сооружения” з доповненнями, затвердженими наказом Держкоммістобудування України від 29.12.94 № 106;

– Сн 512-78 „Инструкция по проэктированию зданий и помещений для электронно-вычислительных машин”, затверджених Держбудом СРСР;

– ДСанПіН 3.3.3.-007-98 „Державні санітарні правила і норми роботи з візуальними дисплеями, терміналами електронно-обчислювальних машин”, затверджених МОЗ України 10.12.98, а також вимогами нормативно-технічної та експлуатаційної документації заводу-виробника ЕОМ, чинних санітарних норм, санітарних норм і правил, правил у сфері охорони праці.

Для всіх споруд і приміщень, в яких експлуатуються відеотермінали та ЕОМ, повинна бути визначена категорія з вибухопожежної та пожежної безпеки відповідно до ОНТП 24-86 „Определение категорий помещений и зданий по взрывопожарной и пожарной опасности”, затверджених МВС СРСР 27.02.86, та клас зони згідно з ПВЕ. Відповідні позначення повинні бути нанесені на вхідні двері приміщення.

Будівлі і ті їх частини, в яких розташовуються ЕОМ, повинні мати не нижче II ступеня вогнестійкості. Приміщення для обслуговування, ремонту та налагодження ЕОМ повинні належати за пожежовибухобезпекою до категорії В відповідно до ОНТП 24-86, а за класом приміщення – до П-IIа за ПВЕ. Якщо відповідно до СНиП 2.09.02-85 ці приміщення повинні бути відокремленими від приміщень іншого призначення протипожежними стінами, то межа їх вогнестійкості визначається відповідно до СНиП 2.01.02 -85.

Неприпустимим є розташування приміщень категорій А і Б (ОНТП 24-86), а також виробництва з мокрими технологічними процесами поряд з приміщеннями, де розташовуються ЕОМ, виконується їх обслуговування, налагодження і ремонт,

а також над такими приміщеннями або під ними. Виробничі приміщення, в яких розташовані ЕОМ, не повинні межувати з приміщеннями, де рівні шуму та вібрації перевищують норму (механічні цехи, майстерні тощо).

Робочі місця з відео-терміналами або персональними ЕОМ у приміщеннях з джерелами шкідливих виробничих факторів, повинні розміщуватися в ізольованих кабінах з обладнаним повітрообміном. Стіни кабін виготовляються з негорючих матеріалів. Дозволяється виготовляти їх зі скла та металевих конструкцій. У кабіні мусить бути оглядове вікно (вікна). Висота оглядового вікна повинно бути не менше 1,5 м, а відстань віл підлоги не більше як 0,8 м.

Відповідно до ДСанПіН 3.3.3-007-98 „Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин", затверджених МОЗ України 10.12.98, є неприпустимим розташування приміщень для роботи з відео-терміналами та ЕОМ у підвалах та цокольних поверхах.

Площу приміщень в яких розташовують відеотермінали, визначають згідно з чинними нормативними документами, з розрахунку на одне робоче місце, обладнане відеотерміналом: площа – не менше ніж 6,0 м2, обсяг – не менше 20,0 м3, з урахуванням максимальної кількості осіб, які одночасно працюють у зміні.

Стіни, стеля та підлога приміщень, де розміщені ЕОМ, повинні виготовлятися з матеріалів, дозволених для оздоблення приміщень органами державного санітарно-епідеміологічного нагляду.

Обслуговування, налагодження і ремонт ЕОМ, вузлів та блоків ЕОМ слід виконувати в окремому приміщені (майстерні).

При цьому робочі місця електромеханіків повинні бути оснащені спеціальним обладнанням та захисними засобами відповідно до вимог до організації робочого місця з обслуговування, ремонту та налагодження ЕОМ.

У приміщеннях для обслуговування, ремонту та налагодження ЕОМ слід передбачити можливість вологого очищення поверхонь комунікації та опалювальних приладів.

Підлога всієї зони обслуговування, ремонту та налагодження ЕОМ, вузлів та

блоків ЕОМ має бути вкрита діелектричними килимками, термін використання яких після їх випробування на електричну міцність не закінчився, або викладена ізольованими підстилками (шириною не менше ніж 0,75-0,8 м) для ніг.

Приміщення, в яких проводиться паяння, крім того, повинні відповідати вимогам СП 952-72 „Санитарные правила организации процессов пайки мелких изделий сплавами, содержащими свинец", затверджених головним санітарним лікарем СРСР 20.03.72.

Приміщення комп'ютерних класів (залів), в яких проводиться навчання на ЕОМ, крім зазначених у пункті 1.2, повинні мати суміжне приміщення (лаборантську) площею не менше 18 кв. м. з двома входами: в учбове приміщення та в коридор (на сходову клітку).

Заземлені конструкції, що знаходяться в приміщеннях (батареї опалення, водопровідні труби, кабелі із заземленим відкритим екраном тощо), мають бути надійно захищені діелектричними щитками або сітками від випадкового дотику.

У приміщеннях з ЕОМ слід щоденно проводити вологе прибирання.

У приміщеннях з ЕОМ повинні бути медичні аптечки першої допомоги.

Приміщення з ЕОМ, крім приміщень, в яких розміщуються ЕОМ типу ЕС, СМ та інші великі ЕОМ загального призначення, повинні бути оснащені системою автоматичної пожежної сигналізації відповідно до вимог Переліку однотипних за призначенням об'єктів, які підлягають обладнанню

автоматичними установками пожежогасіння та пожежної сигналізації, затвердженого наказом Міністерства внутрішніх справ України від 20.11.97 №779 і зареєстрованого в Міністерстві юстиції України 28.11.97 за №567/2371, та СНиП 2.04.09-84 „Пожарная автоматика зданий и сооружений" з димовими пожежними сповіщувачами та переносними вуглекислотними вогнегасниками з розрахунку 2 шт. На кожні 20 м2 площі приміщення з урахуванням гранично допустимих концентрацій вогнегасної рідини відповідно до вимог Правил пожежної безпеки в Україні. В інших приміщеннях допускається встановлювати теплові пожежні сповіщувачі.

Приміщення, в яких розміщуються ЕОМ типу ЄС, СМ та інші великі ЕОМ

загального призначення, обладнуються системою автоматичної пожежної сигналізації та засобами пожежогасіння відповідно до вимог Переліку однотипних за призначенням об'єктів, які підлягають обладнанню автоматичними установками пожежогасіння та пожежної сигналізації, СНиП 2.04.09-84, СН 512-78, Правил пожежної безпеки в Україні та вимог нормативно-технічної та експлуатаційної документації заводу-виробника.

Підходи до засобів пожежогасіння повинні бути вільними.

Приміщення для відпочинку осіб, які працюють з ЕОМ, призначені для приймання їжі, психологічного розвантаження, та інші побутові приміщення повинні обладнуватись відповідно до вимог СНиП 2.09.04-87 „Административные и бытовые здания", з урахуванням максимальної кількості працівників, що одночасно працюють у зміні.

Власник організовує проведення досліджень шкідливих та небезпечних факторів виробничого середовища і трудового процесу на робочих місцях осіб, які працюють з ЕОМ, відповідно до чинних законодавчих та інших нормативно-правових актів і Порядку проведення атестації робочих місць за умовами праці, затвердженого постановою Кабінету Міністрів України від 01.08.92 №442.

Санітарно-гігієнічні вимоги. Умови праці осіб, які працюють з ЕОМ, повинні відповідати I або II класу згідно з Гігієнічною класифікацією праці за показниками шкідливості та небезпечності факторів виробничого середовища, важкості та напруженості трудового процесу №4137-86, затвердженою МОЗ СРСР 12.08.86.

Вимоги до освітлення

Приміщення з ЕОМ повинні мати природне і штучне освітлення відповідно до СПиП II-4-79 „Естественное и искусственное освещение".

Природне світло повинно проникати через бічні світлопрорізи, зорієнтовані, як правило, на північ чи північний схід, і забезпечувати коефіцієнт природної освітленості (КПО) не нижче 1,5%. Розрахунки КПО проводяться відповідно до СНиП II-4-79.

При виробничій потребі дозволяється експлуатувати ЕОМ у приміщеннях

без природного освітлення за узгодженням з органами державного нагляду за охороною праці ц органами та установами санітарно-епідеміологічної служби.

Вікна приміщень з відеотерміналами повинні мати регулювальні пристрої для відкривання, а також жалюзі, штори, зовнішні козирки тощо.

Штучне освітлення приміщення з робочими місцями, обладнаними відеотерміналами ЕОМ загального та персонального користування, має бути обладнане системою загального рівномірного освітлення. У виробничих та адміністративно-громадських приміщеннях, де переважають роботи з документами, допускається вживати систему комбінованого освітлення (додатково до загального освітлення встановлюються світильники місцевого освітлення).

Загальне освітлення має бути виконане у вигляді суцільних або переривчастих ліній світильників, що розміщуються збоку від робочих місць (переважно зліва) паралельно лінії зору працівників. Допускається застосувати світильники таких класів світлорозподілу:

- світильники прямого світла – П.

4.2 Вимоги охорони праці при експлуатації комп’ютерних систем

Користувачі комп’ютерних систем повинні слідкувати за тим, щоб відеотермінали, комп’ютерних систем, периферійні пристрої комп’ютерних систем та устаткування для обслуговування, ремонту та налагодження комп’ютерних систем були справними і випробуваними відповідно до чинних нормативних документів.

Щоденно перед початком роботи необхідно проводити очищення екрану відеотерміналу від пилу та інших забруднень.

Після закінчення роботи відеотермінал та персональна комп’ютерних систем повинні бути відключені від електричної мережі.

У разі виникнення аварійної ситуації необхідно негайно відключити відеотермінал та комп’ютерні системи від електричної мережі.

При використанні комп’ютерних систем та відеотерміналами лазерних принтерів потрібно дотримуватись вимог Санітарних норм та правил устрою та експлуатації лазерів №5804-91, затверджених Міністерством охорони здоров'я СРСР в 1991р.

При потребі, для захисту від електромагнітних, електростатичних та інших полів можуть застосовуватися спеціальні технічні засоби, що мають відповідний сертифікат або санітарно-гігієнічний висновок акредитованих органів щодо їх захисних властивостей.

Є неприпустимими такі дії:

1. виконання обслуговування, ремонту.

2. налагодження комп’ютерних систем безпосередньо на робочому місці користувача комп’ютерної системи.

3. зберігання біля відеотермінала та комп’ютерних систем паперу, дискет, інших носіїв інформації, запасних блоків, деталей тощо, якщо вони не використовуються для поточної роботи.

4. відключення захисних пристроїв, самочинне проведення змін у конструкції та складі комп’ютерних систем, устаткування або їх технічне налагодження.

5. робота з відеотерміналами, в яких під час роботи з'являються нехарактерні сигнали, нестабільне зображення на екрані тощо.

6. праця на матричному принтері зі знятою (трохи піднятою) верхньою кришкою.

Відомо, що шум він несприятливо діє на слуховий аналізатор та інші органи та системи організму людини. Визначальне значення щодо такої дії має інтенсивність шуму, його частотний склад, тривалість щоденного впливу, індивідуальні особливості людини, а також специфіка виробничої діяльності. Ті види діяльності, у яких поєднується напружена розумова робота та інтенсивне використання комп'ютера (редагування тексту, верстка оригіналу, "запуск" та відлагодження програм тощо) характеризується відчутним впливом навіть незначних рівнів шуму. Цей вплив виражається у зниженні розумової

працездатності, швидкій втомлюваності, послабленні уваги, появі головного болю та ін.

Основними заходами та засобами боротьби з шумом є:

- зниження рівнів шуму в джерелі його утворення (застосовується, як правило, в процесі проектування);

- використання звукопоглинаючих та звукоізолюючих засобів;

- раціональне планування виробничих приміщень та робочих місць.

На комп'ютеризованих робочих місцях основними джерелами шуму є вентилятори системного блоку, накопичувачі, принтери ударної дії. Для зниження рівнів шуму на робочих місцях рекомендується розмістити друкувальні пристрої ударної дії (матричні, шрифтові принтери тощо) в іншому приміщенні, або огородити їх звукоізолюючими екранами.

Під час виконання робіт з ВДТ і ПК у виробничих приміщеннях значення характеристик вібрації на робочих місцях не повинні перевищувати допустимих значень, визначених СН 3044-84 та ГОСТ 12.1.012-90. Керівники структурних підрозділів зобов’язані враховувати такі основні вимоги до організації робіт, пов’язаних з використанням ВДТ ЕОМ і ПП (відповідно до вимог ДСанПіН 3.3.2.007-98):

Розміщення робочих місць у підвальних приміщеннях і на цокольних поверхах заборонено;

Площа приміщення на одне робоче місце має становити не менше ніж 6,0 кв. м, а об’єм – не менше ніж 20,0 кв. м;

Приміщення повинні мати природне та штучне освітлення;

Приміщення мають бути обладнані системами опалення, кондиціювання повітря або припливно-витяжною вентиляцією, які б забезпечували дотримання в приміщеннях нормованих параметрів мікроклімату ;

Віконні прорізи приміщень мають бути обладнані регульованими пристроями (жалюзі, завіси тощо);

у приміщеннях повинно щоденно проводитися вологе прибирання;

Приміщення мають бути оснащені аптечками першої медичної допомоги,

вогнегасниками та системою автоматичної пожежної сигналізації;

При розміщенні робочих столів з ВДТ відстань між бічними поверхнями ВДТ має бути не меншою 1,2 м, а відстань між тильними поверхнями екранів ВДТ – не меншою 2,5 м;

Висота робочої поверхні робочого столу з ВДТ має бути в межах 680-800 мм, а ширина і глибина – забезпечувати можливість виконання операцій у зоні досяжності моторного поля (рекомендовані розміри: 600-1400 мм, глибина – 800-1000 мм);

Передбачати для операторів ЕОМ залежно від характеру праці такі внутрішньозмінні режими праці та відпочинку (для розробників програм із застосуванням ЕОМ – регламентовані перерви у роботі тривалістю 15 хв. після кожної години роботи за ВДТ, для операторів комп'ютерного набору – 10 хв. після кожної години роботи за ВДТ, для операторів ЕОМ з іншим характером праці –15 хв. через кожні дві години роботи з використанням ВДТ);

Для зниження вібрації обладнання, пристрої, пристосування необхідно встановлювати на спеціальні амортизуючі прокладки, передбачені нормативними документами.

Головним небезпечним фактором небезпеки при експлуатації та ремонті пристрою є електробезпека.

Електрика — це сукупність явищ, зумовлених існуванням, рухом взаємодією електрична заряджених тіл або часток. Електричний струм — це упорядкований (спрямований) рух електрична заряджених часток. Струм у металах зумовлений наявністю вільних електронів, у електролітах — іонів. Звичайно силою, яка викликає такий рух, є сила з боку електричного поля усередині провідника, яке визначається електричною напругою на кінцях провідника.

Наявність електричного струму в провідниках призводить до їх нагрівання, зміни хімічного складу, створення магнітного поля.

Електричні прилади, обладнання та установки, з яким людина має справу, становлять для неї велику потенційну небезпеку, яка посилюється тим, що органи

чуття людини не можуть на відстані виявити наявність електричної напруги, як, наприклад, світлову, теплову чи механічну енергію. Тому захисна реакція організму виявляється тільки після потрапляння безпосередньо під дію електричного струму. Другою особливістю дії електричного струму на організм людини є те, що струм, проходячи через людину, діє не тільки в місцях контактів і на шляху протікання через організм, а й викликає рефлекторні порушення нормальної діяльності окремих органів (серцево-судинної системи, системи дихання). Третя особливість — це можливість одержання електротравм без безпосереднього контакту із струмопровідними частинами — при переміщенні по землі поблизу ушкодженої електроустановки (у випадку замикання на землю), ураження через електричну дугу.

Електричний струм, проходячи через тіло людини, зумовлює перетворення поглинутої організмом електричної енергії в інші види і

спричиняє термічну, електролітичну, механічну і біологічну дію. Найбільш складною є біологічна дія, яка притаманна тільки живим організмам.

Статистика свідчить, що більше половини всіх електротравм становлять опіки. Вони важко піддаються лікуванню, тому що глибоко проникають у тканини організму. В електроустановках напругою до 1 кВ найчастіше спостерігаються опіки контактного виду при дотиканні тіла до струмопровідних частин.

Опіки можливі при проходженні через тіло людини струму більше 1 А. Тільки при великому струмі тканини, які уражаються, нагріваються до температури 60—70°С і вище, при якій згортається білок і з'являються опіки.

В електроустановках напругою вище 1 кВ опіки можуть виникнути при випадковому наближенні частин тіла людини до струмопровідних частин на небезпечну відстань; при цьому збільшується напруга електричного поля і внаслідок ударної іонізації діелектрика (повітряного проміжку) опір цього проміжку зменшується, його «пробиває» електричний розряд — електрична дуга, температура якої досягає приблизно 4000'С. Електричний струм протікає через дугу і тіло людини. За такої високої температури і великої кількості тепла, яка виділяється при проходженні струму через тіло, потерпілий одержує тяжкі опіки,

його м'язи скорочуються, дуга і ланцюг струму розриваються.

Майже у всіх випадках включення людини в електричний ланцюг на її тілі і в місцях дотикання спостерігаються «електричні знаки» сіро-жовтого кольору круглої або овальної форми.

При опіках від впливу електричної дуги можлива металізація шкіри частками металу дугової плазми. Уражена ділянка шкіри стає твердою, набуває кольору солей металу, які потрапили в шкіру.

Електролітична дія струму виявляється у розкладанні органічної рідини, в тому числі крові, яка є електролітом, та в порушенні її фізико-хімічного складу.

Біологічна дія струму виявляється через подразнення і збудження живих тканин організму, а також порушення внутрішніх біологічних процесів.

Механічна дія струму призводить до розриву тканин організму внаслідок електродинамічного ефекту, а також миттєвого вибухоподібного утворення пари з тканинної рідини і крові.

Внаслідок дії електричного струму або електричної дуги виникає електротравма. Електротравми за значенням поділяють на загальні і місцеві. До місцевих травм належать опіки, електричні знаки, електрометалізація шкіри, механічні пошкодження, а також електрофтальмія, запалення очей внаслідок впливу ультрафіолетових променів електричної дуги). Загальні електротравми називають також електричними ударами. Вони є найбільш небезпечним видом електротравм. При електричних ударах виникає збудження живих тканин, судомне скорочення м'язів, параліч м'язів опорно-рухового апарату, м'язів грудної клітки (дихальних), м'язів шлуночків серця. У першому випадку судомне скорочення м'язів не дозволяє людині самостійно уникнути дотикання з електроустановкою. При паралічі дихання припиняється газообмін і постачання організму киснем, внаслідок чого наступає задуха. При паралічі серця його функції або припиняються повністю, або деякий час продовжуються в режимі тріпотіння (фібриляції). Фібриляція — це мимовільні скорочення серцевих м'язів, черезя яке серце не може гнати кров судинами. При цьому порушується кровообмін, що також може спричинити смерть. Медичною практикою

встановлено, що після припинення роботи серця і дихання внаслідок кисневого голодування через 5—б хвилин гинуть клітини центральної нервової системи, відбувається втрата свідомості і припиняється управління функціями всіх органів тіла. Саме цей стан отримав назву «клінічна смерть», оскільки клітини інших органів тіла ще живі.

4.3 Висновки щодо безпеки життєдіяльності при розробці програмного забезпечення для автоматизації роботи фітнес клубу

Неприпустимим є знаходження приміщень категорій А і Б (ОНТП 24-86), а також виробництва з мокрими технологічними процесами поряд з приміщеннями, де розташовуються ЕОМ, виконується їх обслуговування, налагодження і ремонт, а також над такими приміщеннями або під ними. Виробничі приміщення, в яких розташовані ЕОМ, не мають межувати з приміщеннями, де рівні шуму та вібрації перевищують норму (механічні цехи, майстерні тощо).

Робочі місця з відеотерміналами або персональними ЕОМ у приміщеннях з джерелами шкідливих виробничих факторів завжди мають розміщуватися в ізольованих кабінах з обладнаним повітрообміном. Стіни таких кабін виготовляються з виключно негорючих матеріалів. Дозволяється виготовляти їх зі скла та металевих конструкцій. У кабіні мусить бути оглядове вікно (вікна). Висота оглядового вікна повинна бути не менше 1,5 м, а відстань від підлоги не більше 0,8 м.

Користувачі комп’ютерних систем завжди повинні слідкувати за тим, щоб відеотермінали комп’ютерних систем, периферійні пристрої комп’ютерних систем та устаткування для обслуговування, ремонту та налагодження комп’ютерних систем були постійно справними і випробуваними відповідно до чинних нормативних документів.

Після закінчення роботи відеотермінал та персональна комп’ютерна система повинні бути відключені від електричної мережі.

Висновки

Робота містить висловлювання про результати обраних методів, що використовуються при розробці програмного забезпечення, метою яких є розробка програмного забезпечення для автоматизації фітнес клубу.

Розроблена програма повністю відповідає описаній темі та виконує всі необхідні завдання.

Програмне забезпечення ретельно перевірено простими тестами, а також реальними людьми.

Програмне забезпечення написано мовою C# та використовує реляційну базу даних MySQL, яка була обрана як сховище даних, а режим комфортного відображення зв'язку між сутностями використовувався в режимі тривалого сну.

Microsoft Visual Studio було обрано як середовище розробки - комерційне інтегроване середовище розробки для різних мов програмування (C++, C#, F#, Javascript тощо) від Microsoft.

Головним завданням даної програми є допомога в розвитку малого бізнесу, а саме поодиноких фітнес-клубів, яким не потрібно багато функціоналу, а лише автоматизація роботи та здобуття бази нових та постійних клієнтів .

В майбутньому можна буде розширити цей проект в особистих і комерційних цілях. Архітектура добре спланована, тому її розширення не займе багато часу. Подальші розробки дозволять покращити інтерфейс користувача, розширити базу даних та додати багато корисних можливостей та функцій для зручності користувача.

Список використаних джерел

1. Офіційний сайт розробників 1С: Фітнес клуб [Електронний ресурс] : [Інтернет портал] – Режим доступу: https://1c.fitness/ - Дата доступу 30.04.2021

2. Презентація опису функціональних можливостей на офіційному сайті розробників Mobifitness [Електронний ресурс] : [Інтернет портал] – Режим доступу: https://new.mobifitness.ru/Mobifitness_Presentation.pdf - Дата доступу 30.04.2021

3. Офіційний сайт розробників Mobifitness [Електронний ресурс] : [Інтернет портал] – Режим доступу: https://new.mobifitness.ru/ - Дата доступу 30.04.2021

4. Офіційний сайт розробників FitBase [Електронний ресурс] : [Інтернет портал] – Режим доступу: https://fitbase.io/ - Дата доступу 30.04.2021

5. Бойко В., Савинков В. Проектирование баз данных информационных систем. -М.: Финансы и статистика, 1989.

6. Дейт К. Введение в систему баз данных.- М.:Мир, 1998.

7. Джексон Г. Проектирование реляционных баз данных. -М.: Мир, 1991.

8. Джудит С. Боуман, Сандра Л. Эмерсон, Марси Дарновски. Практическое руководство по SQL. – М.: Вильямс, 2001. – 336 с.

9. Константайн Л., Локвуд Л. Разработка программного обеспечения: Пер. с англ. - Питер, 2004, -470 с.

10. Мандел Т.: Разработка пользовательского интерфейса Пер. с англ. – М. : ДМК Пресс, 2001, - 416 с.

11. Липаев В.В. Тестирование программ [Текст] / В.В. Липаев. – М.: Радио и связь, 1986. – 296 с.

12. Дейкстра Дисциплина программирования [Текст] / Э.Дейкстра; пер. с англ. И. Х. Зусман ; ред. Э. З. Любимский. – М.: Мир, 1978. – 275 с. – (Математическое обеспечение).

13. ДСТУ 2844-94. Програмні засоби ЕОМ. Забезпечення якості. Терміни та визначення. – Введ. 1.08.1995.

Додаток А

Код програми

fAutorization.cs

```
using Data;
using System;
using System.Collections.Generic;
using System.Collections.Specialized;
using System.Security.Cryptography.X509Certificates;
using System.Windows.Forms;

namespace Fitnesclub
{
   public partial class fAutorization : Form
   {
      public fAutorization()
      {
         InitializeComponent();
      }

      public int userId = 0;
      public int userRol = 0;
      private DBManager db = new DBManager();

      private void label1_Click(object sender, EventArgs e){       }

      private void bEnter_Click(object sender, EventArgs e)
      {
         string login = "'" + tbLogin.Text + "'";
         string pass = "'" + tbPass.Text + "'";
         if (tbLogin.Text == "" || tbPass.Text == "")
         {
            MessageBox.Show("Заповніть поле логіну та паролю");
         }
         else
         {
            List<List<Object>> user = new List<List<Object>>();
            try
            {
               user = db.GetRows("admins", "*", "login=" + login +
                  " AND parol=" + pass);

            }
            catch (Exception ex)
            {
               MessageBox.Show(ex.Message);
               MessageBox.Show("Помилка, перевірте правильність вводу");
            }
            if (user.Count > 0)
```

```
                {
                    userId = Int32.Parse(user[0][0].ToString());
                    userRol = Int32.Parse(user[0][6].ToString());
                    DialogResult = DialogResult.OK;
                }
                else
                {
                    MessageBox.Show("Помилка, перевірте правильність вводу");
                    DialogResult = DialogResult.Cancel;
                }
            }
        }

        private void fAutorization_Load(object sender, EventArgs e)
        {
            tbLogin.Focus();

        }

        private void bReestr_Click(object sender, EventArgs e)
        {
            fReestr reestr = new fReestr();
            reestr.ShowDialog();

            if (reestr.DialogResult == DialogResult.OK)
            {
                MessageBox.Show("Реєстрація пройшла успішно! Введіть логін та пароль у відповідні поля");
            }
        }
    }
}
```

fMeetings.cs

```
using System;
using System.Collections.Generic;
using Data;
using System.Windows.Forms;

namespace Fitnesclub
{
    public partial class fMeeting : Form
    {
        public fMeeting()
        {
            InitializeComponent();
            string admin = db.GetValue("admins", "fio", "id=" + userId.ToString()).ToString();
            tbAdmin.Text = admin;
            tbCassa.Text = "0";
        }

        DBManager db = new DBManager();
```

```
        public int userId;
        public int userRol;
        string klient = "";
        string trener = "";
        string abon = "";
        int r = 0, trens=0;
        double sum_of_cassa = 0, schet;

        private void tb_clear()
        {
            tbAbon.Text = "";
            tbMetka.Text = "";
            tbMetkameet.Text = "";
            tbNomer.Text = "";
            tbSchet.Text = "";
            tbTrener.Text = "";
            tbUser.Text = "";
            klient = "";
            pbFoto.Image = null;
            cbTrener.Checked = false;
            tbKtren.Text = "";
        }

        private void filltextbox(int row)
        {
            if (row < dataGridView1.RowCount - 1 && row >= 0)
            {
                tbTrener.Text = dataGridView1.Rows[row].Cells[2].Value.ToString();
                tbKtren.Text = dataGridView1.Rows[row].Cells[4].Value.ToString();
                tbAbon.Text = dataGridView1.Rows[row].Cells[1].Value.ToString();
                trener = dataGridView1.Rows[row].Cells[12].Value.ToString();
                abon = dataGridView1.Rows[row].Cells[0].Value.ToString();
            }
        }

        private void fillGrid()
        {
            //SELECT * FROM `prodabon` WHERE cast(substring(`date_end`,1,10) AS
datetime)< now()
            if (klient != "")
            {
                dataGridView1.Rows.Clear();
                var list = db.GetRows("prodabon, treners, abons, admins",
                    "prodabon.id, abons.name, treners.fio , count_train, count_train_rest,
prodabon.date_begin," +
                    "prodabon.date_end, prodabon.date_zakaz, sum, pay, admins.fio, id_user,
treners.id",
                    "prodabon.trainer = treners.id and prodabon.abon = abons.id and
prodabon.id_admin = admins.id and " +
                    "id_user = " + klient + " and trainer = treners.id and cast(prodabon.date_end as
datetime) >= now() order by trainer, prodabon.id desc");
                for (int i = 0; i < list.Count; i++)
```

```
                {
                    dataGridView1.Rows.Add(list[i][0].ToString(),  list[i][1].ToString(),
list[i][2].ToString(), list[i][3].ToString(),
                        list[i][4].ToString(),  list[i][5].ToString(),  list[i][6].ToString(),
list[i][7].ToString(), list[i][8].ToString(),
                        list[i][9].ToString(),  list[i][10].ToString(),  list[i][11].ToString(),
list[i][12].ToString());
                }
                if (list.Count > 0)
                {
                    tbTrener.Text = list[0][2].ToString();
                    tbKtren.Text = list[0][4].ToString();
                    tbAbon.Text = list[0][1].ToString();
                    trener = list[0][12].ToString();
                    abon = list[0][0].ToString();
                    trens = Convert.ToInt32(list[0][4]);
                }
            }
        }
        private void bHUser_Click(object sender, EventArgs e)
        {

            var list_abon = db.GetRows("vidviduvachi", "fio, metka, schet, nomer, kode, id, foto",
"nomer= '"+tbNomer.Text+"'");
            show_vidvid(list_abon);
        }

        private void label10_Click(object sender, EventArgs e)
        {
            fSelect f = new fSelect();

            f.dataGridView1.Columns.Clear();
            f.dataGridView1.Columns.Add("col1", "Відвідувач");
            f.dataGridView1.Columns[0].Resizable = DataGridViewTriState.False;
            f.dataGridView1.Columns[0].Width = 200;
            f.dataGridView1.Columns.Add("col2", "Тренер");
            f.dataGridView1.Columns[1].Resizable = DataGridViewTriState.False;
            f.dataGridView1.Columns[1].Width = 200;
            f.dataGridView1.Columns.Add("col3", "Дата та час входу");
            f.dataGridView1.Columns[2].Resizable = DataGridViewTriState.False;
            f.dataGridView1.Columns[2].Width = 150;
            f.dataGridView1.Columns.Add("col4", "Примітка");
            f.dataGridView1.Columns[3].Resizable = DataGridViewTriState.False;
            f.dataGridView1.Columns[3].Width = 100;
            f.dataGridView1.Columns.Add("col5", "Код");
            f.dataGridView1.Columns[4].Resizable = DataGridViewTriState.False;
            f.dataGridView1.Columns[4].Width = 100;
            var list = db.GetRows("meetings, treners, vidviduvachi",
                "vidviduvachi.fio, treners.fio, meetings.date_in,  meetings.metka, meetings.klient,
meetings.vhod, meetings.trener ",
                "vhod=1 and meetings.trener = treners.id and meetings.klient = vidviduvachi.id");
            if (list.Count > 0)
```

```
                for (int i = 0; i < list.Count; i++)
                  f.dataGridView1.Rows.Add(list[i][0].ToString(), list[i][1].ToString(),
list[i][2].ToString(), list[i][3].ToString(), list[i][4].ToString());

              if (f.ShowDialog() == DialogResult.OK)
              {
                 tbUser.Text = list[f.row][0].ToString();
                 button1_Click(bspib, e);
              };

           }

           //1 voshel
           private void button7_Click(object sender, EventArgs e)
           {
             if (tbUser.Text != "")
              {
                 var enter = db.GetRows("meetings", "*", "klient ="+klient+" and vhod=1");
                 string kod = "0";
                 if (enter.Count == 0)
                 {
                    try
                    {
                       kod = Convert.ToString(Convert.ToInt32(db.GetValue("meetings", "max(id)",
"")) + 1);
                    }
                    catch
                    {
                       kod = "1";
                    }
                   // var freeze = db.GetRows("meetings", "*", "klient =" + klient + " and vhod=1");
                    if (cbTrener.Checked)//with trener
                    {
                       if (trener != "0")
                       {
                          string[] list_field = { "id", "admin", "trener", "klient", "date_in", "vhod",
"metka", "abon" };
                          string[] list_val = { kod, userId.ToString(), trener, klient,
                          """+DateTime.Now.ToString("yyyy-MM-dd HH:mm:ss")+"", "1", """ +
tbMetkameet.Text + """ , abon};
                          db.InsertToBD("meetings", list_field, list_val);
                          string[] list_field1 = { "id", "count_train_rest" };
                          string[] list_val1 = { abon, "count_train_rest-1" };
                          db.UpdateRecord("prodabon", list_field1, list_val1);
                          MessageBox.Show("Вхід виконано!");
                       }
                       else
                       {
                          MessageBox.Show("Оберіть замовлення з тренером!");
                       }
                    }
                    else//without trener
```

```
                {
                    if (trens>0)
                    {
                        string[] list_field = { "id", "admin", "klient", "date_in", "vhod", "metka" };
                        string[] list_val = { kod, userId.ToString(), klient, "'" +
DateTime.Now.ToString("yyyy-MM-dd HH:mm:ss") + "', "1", "'" + tbMetkameet.Text + "'" };
                        db.InsertToBD("meetings", list_field, list_val);
                        string[] list_field1 = { "id", "count_train_rest" };
                        string[] list_val1 = { abon, "count_train_rest-1" };
                        db.UpdateRecord("prodabon", list_field1, list_val1);
                        MessageBox.Show("Вхід виконано!");
                    }
                    else
                    {
                        string[] list_field = { "id", "admin", "klient", "date_in", "vhod", "metka" };
                        string[] list_val = { kod, userId.ToString(), klient, "'" +
DateTime.Now.ToString("yyyy-MM-dd HH:mm:ss") + "', "1", "'" + tbMetkameet.Text + "'" };
                        db.InsertToBD("meetings", list_field, list_val);
                        MessageBox.Show("Вхід виконано!");
                    }
                }
                else
                {
                    MessageBox.Show("Клієнт вже у залі!");
                }
            }
            else
            {
                MessageBox.Show("Вхід не виконано!");
            }
            cbTrener.Checked = false;
            fillGrid();
        }

        private void bProda_Click(object sender, EventArgs e)
        {
            if (klient!="")
            {
                fProdAbon prodAbon = new fProdAbon();
                prodAbon.userId = userId;
                prodAbon.userRol = userRol;
                prodAbon.klient = klient;
                prodAbon.tbUser.Text = tbUser.Text;
                prodAbon.ShowDialog();
                sum_of_cassa += prodAbon.sum_of_cassa;
                tbCassa.Text = Convert.ToString(sum_of_cassa);
                tbSchet.Text = Convert.ToString(prodAbon.schet);
                fillGrid();
                filltextbox(0);
            }
            else
```

```
            {
                MessageBox.Show("Оберіть клієнта!");
            }
        }

        private void bProdt_Click(object sender, EventArgs e)
        {

            if (klient != "")
            {
                fProdTovar prodTovar = new fProdTovar();
                prodTovar.userId = userId;
                prodTovar.userRol = userRol;
                prodTovar.klient = klient;
                prodTovar.tbUser.Text = tbUser.Text;
                prodTovar.ShowDialog();
                tbSchet.Text = Convert.ToString(prodTovar.schet);
                sum_of_cassa += prodTovar.sum_of_cassa;
                tbCassa.Text = Convert.ToString(sum_of_cassa);
                fillGrid();
                filltextbox(0);
            }
            else
            {
                MessageBox.Show("Оберіть клієнта!");
            }
        }

        private void bExit_Click(object sender, EventArgs e)
        {// 0 vyhod

            if (tbUser.Text != "")
                {
                string kod = Convert.ToString(db.GetValue("meetings", "id", "klient="+klient+" and
vhod=1"));

                string[] list_field = { "id", "date_out", "vhod"};
                string[] list_val = { kod, "'" + DateTime.Now.ToString("yyyy-MM-dd HH:mm:ss") +
"'", "0" };
                db.UpdateRecord("meetings", list_field, list_val);
                MessageBox.Show("Вихід успішний!");
                }
                else
                {
                MessageBox.Show("Вихід не виконано!");
                }
        }

        private void bNew_Click(object sender, EventArgs e)
        {
            fPosetit posetit = new fPosetit();
            posetit.ShowDialog();
```

```
        }

        void show_vidvid(List<List<Object>> list_abon)
        {
            tb_clear();
            if (list_abon.Count > 0)
            {
                tbUser.Text = list_abon[0][0].ToString();
                tbMetka.Text = list_abon[0][1].ToString();
                tbSchet.Text = list_abon[0][2].ToString();
                tbNomer.Text = list_abon[0][3].ToString();
                tbKode.Text = list_abon[0][4].ToString();
                klient = list_abon[0][5].ToString();
                if (list_abon[0][6].ToString() != "")
                {
                    pbFoto.Image = db.take_ftp(list_abon[0][6].ToString());
                }
                else
                    pbFoto.Image = null;
                fillGrid();
            }
            else
            {
                MessageBox.Show("Відвідувача не знайдено!");
                tb_clear();
            }
        }
        private void button1_Click(object sender, EventArgs e)
        {
            var list_abon = db.GetRows("vidviduvachi", "fio, metka, schet, nomer, kode, id, foto",
                "fio like '" + tbUser.Text + "%' ");

            show_vidvid(list_abon);
        }

        private void bskode_Click(object sender, EventArgs e)
        {
            var list_abon = db.GetRows("vidviduvachi", "fio, metka, schet, nomer, kode, id, foto",
"kode = '" + tbKode.Text + "'");
            show_vidvid(list_abon);
        }

        private void cbTrener_CheckedChanged(object sender, EventArgs e)
        {
            int i = 0;
            while                                                    (i<dataGridView1.RowCount-
1&&dataGridView1.Rows[i].Cells[12].Value.ToString() == "0")
                i++;
            if (i < dataGridView1.RowCount)
            {
                filltextbox(i);
```

```
            dataGridView1.Rows[i].Selected = true;
        }
    }

    private void dataGridView1_CellClick(object sender, DataGridViewCellEventArgs e)
    {
        r = e.RowIndex;
        filltextbox(r);
    }

    private void dataGridView1_KeyUp(object sender, KeyEventArgs e)
    {
        if (e.KeyCode == Keys.Down)
        {
            if (r < dataGridView1.RowCount - 1) r++;
            filltextbox(r);
        }
        if (e.KeyCode == Keys.Up)
        {
            if (r > 0) r--;
            filltextbox(r);
        }
    }

    private void bMoroz_Click(object sender, EventArgs e)
    {
        if (klient != ""&& tbAbon.Text!=""&& tbTrener.Text!="")
        {
            fMoroz moroz = new fMoroz();
            moroz.klient = klient;
            moroz.Text = tbUser.Text;
            moroz.abon = abon;
            moroz.tbAbon.Text = tbAbon.Text;
            moroz.tbTren.Text = tbTrener.Text;

            moroz.ShowDialog();
        }
        else
        {
            MessageBox.Show("Оберіть клієнта та обонемент, що необхідно заморозити!");
        }

    }

    private void bZvit_Click(object sender, EventArgs e)
    {
        if (klient != "")
        {
            fZvitKlient zvitklient = new fZvitKlient();
            zvitklient.tbUser.Text = tbUser.Text;
            zvitklient.klient = klient;
            zvitklient.Show();
```

```
        }
    }

    private void bskode_Click_1(object sender, EventArgs e)
    {
        var list_abon = db.GetRows("vidviduvachi", "fio, metka, schet, nomer, kode, id, foto",
"kode = '" + tbKode.Text + "'");
        show_vidvid(list_abon);
    }

    private void tbKode_TextChanged(object sender, EventArgs e)
    {
        if (tbKode.Text.Length == 13)
        {
            var list_abon = db.GetRows("vidviduvachi", "fio, metka, schet, nomer, kode, id,
foto", "kode = '" + tbKode.Text + "'");
            show_vidvid(list_abon);
        }
    }

    private void bpolozh_Click(object sender, EventArgs e)
    {
        if (tbUser.Text != "")
        {
            schet = Convert.ToDouble(db.GetValue("vidviduvachi", "schet", "id = " + klient));
            schet += Convert.ToDouble(tbSchet.Text);

            string[] list_field = { "id", "schet" };
            string[] list_val = { klient, schet.ToString() };
            db.UpdateRecord("vidviduvachi", list_field, list_val);

            tbSchet.Text = schet.ToString();
            MessageBox.Show("Гроші покладено!");
        }
        else
        {
            MessageBox.Show("Гроші не покладено!");
        }
    }

    private void tbSchet_KeyPress(object sender, KeyPressEventArgs e)
    {

        // char number = e.KeyChar;
        if ( e.KeyChar >= 58)
        {
            e.Handled = true;
        }
    }

    private void button5_Click_1(object sender, EventArgs e)
    {       //open&close smena
```

```
            if (bSmena.Text != "Закрити зміну")
            {
               //open
                int id =Convert.ToInt32(db.GetValue("ad_work", "max(id)", "")) + 1;

               string[] list_field = {"id","id_admin","dvhod"};
               // String date_ = DateTime.Now.Year.ToString()
                string[] list_val = {  id.ToString(), userId.ToString(), "now()"};
                db.InsertToBD("ad_work", list_field, list_val);
               bSmena.Text = "Закрити зміну";
            }
            else
            {
               //close
               string[] list_field = { "dvyhod", "cassa" };
               //String date_ = DateTime.Now.Year.ToString();
               string[] list_val = { "now()", sum_of_cassa.ToString() };
               db.UpdateRecord("ad_work", list_field, list_val, "dvyhod is null");
               bSmena.Text = "Відкрити зміну";
            }
         }
      }
   }
```

DBManager.cs

```
using MySql.Data.MySqlClient;
using System;
using System.Collections.Generic;
using System.Drawing;
using System.IO;
using System.Net;
using System.Windows.Forms;

namespace Data
{
   public class DBManager
   {
      private MySqlConnection connection;
//Pwd=1337      !9Plg2019
      public String connectionString=
"Server=localhost;Database=fitnesclub;Uid=root;Pwd=11111111;"; // TODO
Default connection string

      private MySqlTransaction currentTransaction;


      public DBManager()
      {
         connectionString =
```

```
"Server=localhost;Database=fitnesclub;Uid=root;Pwd=11111111;";
            connection = new MySqlConnection(connectionString);
            Connect();
        }

        public DBManager(String connectionString)
        {
            this.connectionString = connectionString;
            connection = new MySqlConnection(connectionString);
            Connect();
        }

        ~DBManager()
        {
            Disconnect();
        }

        public void Connect()
        {
            try
            {
                connection.Open();
            }
            catch (Exception e)
            {
                Console.Write(e.StackTrace);
            }
        }

        public void Disconnect()
        {
            try
            {
                connection.Close();
            }
            catch (Exception e)
            {
                Console.Write(e.StackTrace);
            }
        }

        public void StartTransaction()
        {
            currentTransaction = connection.BeginTransaction();
        }

        public void CommitTransaction()
        {
            if (currentTransaction != null)
            {
                currentTransaction.Commit();
            }
```

```
        }

        public void RollbackTransaction()
        {
            if (currentTransaction != null)
            {
                currentTransaction.Rollback();
            }
        }

        // Returns single first value according to parameters
        public Object GetValue(String tableName, String fields, String cond)
        {
            //try catch
            try
            {
                MySqlCommand command = new
MySqlCommand(GetSelectStatement(tableName, fields, cond), connection);

                return command.ExecuteScalar();
            }
            catch (MySqlException ex)
            {
                throw new Exception(ex.Message);
            }
        }

        private String GetSelectStatement(String tableName, String fields, String
cond)
        {
            String res = "SELECT " + fields + " FROM " + tableName;
            if (cond != "")
            {
                res += " WHERE " + cond;
            }
            res += ";";

            return res;
        }

        // Returns list of rows
        public List<List<Object>> GetRows(String tableName, String fields, String
cond)
        {
            //try catch
            var res = new List<List<Object>>();

            MySqlCommand command = new
MySqlCommand(GetSelectStatement(tableName, fields, cond), connection);

            using (var reader = command.ExecuteReader())
            {
```

```
            while (reader.Read())
            {
                List<Object> row = new List<object>();
                for (int i = 0; i < reader.FieldCount; i++)
                {
                    row.Add(reader[i]);
                }
                res.Add(row);
            }
        }
        return res;
    }

    // Update table and return number of updated rows
    public int SetValue(String tableName, String field, String value, String cond)
    {
        //try catch
        value = ValidateString(value);
        MySqlCommand command = new
MySqlCommand(GetUpdateStatement(tableName, field, value, cond),
connection);
        return command.ExecuteNonQuery();
    }

    private string GetUpdateStatement(string tableName, string field, string
value, string cond)
    {
        String res = "UPDATE " + tableName + " SET " + field + " = " + value + "
;";
        return res;
    }

    /// --IVAN

    public void DeleteFromDB(string table, string colName, string colValue)
    {
        string sqlCommand = "DELETE FROM " + table + " WHERE " +
colName + " = " + colValue + " ";
        MySqlCommand deleteCmd = new MySqlCommand(sqlCommand,
connection);
        deleteCmd.ExecuteNonQuery();
    }

    public void DeleteFromDB(string table, string[] colName, string[] colValue)
    {
        if (colName.Length == colValue.Length)
        {
            if (colName.Length > 1)
            {
                string sqlCommand = "";
                string res = "Deleted";
                try
```

```
                {
                    sqlCommand = "DELETE FROM " + table + " WHERE ";
                    for (int i = 0; i < colName.Length - 1; i++)
                    {
                        sqlCommand += colName[i] + " = " + colValue[i] + " AND ";
                    }
                    sqlCommand += "" + colName[colName.Length - 1] + " = " +
colValue[colValue.Length - 1] + ";";
                    MySqlCommand deleteCmd = new MySqlCommand(sqlCommand,
connection);
                    deleteCmd.ExecuteNonQuery();
                }
                catch (Exception ex) { res = ex.ToString() + "\n" + sqlCommand; }
            }
            else
            {
                string colname = colName[0];
                string colvalue = colValue[0];
            }
        }
        else { throw new ArgumentException("Field and Value list dont match.");
}
    }

    public int InsertToBD(string table, string list)
    {
        string sqlCommand = "INSERT INTO " + table + " VALUES(" + list +
");";
        sqlCommand += "select last_insert_id();";
        MySqlCommand insertCmd = new MySqlCommand(sqlCommand,
connection);
        return Int32.Parse(insertCmd.ExecuteScalar().ToString());
    }

    public int InsertToBD(string table, string[] fieldNames, string[] fieldValues)
    {
        if (fieldNames.Length == fieldValues.Length)
        {
            fieldValues = ValidateStrings(fieldValues);
            string sqlCommand = "INSERT INTO " + table + "(";
            for (int i = 0; i < fieldNames.Length - 1; i++)
            {
                sqlCommand += " " + fieldNames[i] + ",";
            }
            sqlCommand += fieldNames[fieldNames.Length - 1];
            sqlCommand += ") VALUES(";
            for (int i = 0; i < fieldValues.Length - 1; i++)
            {
                sqlCommand += " " + fieldValues[i] + ",";
            }
            sqlCommand += fieldValues[fieldNames.Length - 1];
            sqlCommand += ");";
```

```
            sqlCommand += "select last_insert_id();";
            MySqlCommand insertCmd = new MySqlCommand(sqlCommand,
connection);
            int id = Int32.Parse(insertCmd.ExecuteScalar().ToString());
            return id;
         }
         else
         {
            throw new ArgumentException("Field and Value list dont match.");
         }
      }

      public void InsertToBDWithoutId(string table, string[] fieldNames, string[]
fieldValues)
      {
         if (fieldNames.Length == fieldValues.Length)
         {
            try
            {
               fieldValues = ValidateStrings(fieldValues);
               string sqlCommand = "INSERT INTO " + table + "(";
               for (int i = 0; i < fieldNames.Length - 1; i++)
               {
                  sqlCommand += " " + fieldNames[i] + ",";
               }
               sqlCommand += fieldNames[fieldNames.Length - 1];
               sqlCommand += ") VALUES(";
               for (int i = 0; i < fieldValues.Length - 1; i++)
               {
                  sqlCommand += " " + fieldValues[i] + ",";
               }
               sqlCommand += fieldValues[fieldNames.Length - 1];
               sqlCommand += ");";
               new MySqlCommand(sqlCommand, connection).ExecuteNonQuery();
            }
            catch(Exception ex)
            {
               throw new Exception(ex.Message);
            }
         }
         else
         {
            throw new ArgumentException("Field and Value list dont match.");
         }
      }

      //"INSERT INTO " + table + "(" + fieldNames[i] + "

      public int UpdateRecord(string tableName, string[] colNames, string[]
colValues)
      {
         if (colNames.Length == colValues.Length)
```

```
        {
            colValues = ValidateStrings(colValues);

            string sqlCommand = "UPDATE " + tableName + " SET ";

            for (int i = 1; i < colValues.Length - 1; i++)
            {
                sqlCommand += colNames[i] + "=" + colValues[i] + ", ";
            }
            sqlCommand += colNames[colValues.Length - 1] + "=" +
colValues[colValues.Length - 1] + "";
            sqlCommand += " where " + colNames[0] + "=" + colValues[0] + "";
            MySqlCommand insertCmd = new MySqlCommand(sqlCommand,
connection);
            return insertCmd.ExecuteNonQuery();
        }
        else
        {
            throw new ArgumentException("Field and Value list dont match.");
        }
    }
    public int UpdateRecord(string tableName, string[] colNames, string[]
colValues, string cond)
    {
        if (colNames.Length == colValues.Length)
        {
            colValues = ValidateStrings(colValues);

            string sqlCommand = "UPDATE " + tableName + " SET ";

            for (int i = 1; i < colValues.Length - 1; i++)
            {
                sqlCommand += colNames[i] + "=" + colValues[i] + ", ";
            }
            sqlCommand += colNames[colValues.Length - 1] + "=" +
colValues[colValues.Length - 1] + "";
            sqlCommand += " where " + cond;
            MySqlCommand insertCmd = new MySqlCommand(sqlCommand,
connection);
            return insertCmd.ExecuteNonQuery();
        }
        else
        {
            throw new ArgumentException("Field and Value list dont match.");
        }
    }
    private string ValidateString(String str)
    {
        if (str[0] == '\'')
            return '\'' + str.Trim('\'').Replace('\'', '`') + '\'';
        return str.Replace('\'', '`');
    }
```

```
        private string[] ValidateStrings(string[] strs)
        {
            string[] res = new string[strs.Length];

            for (int i = 0; i < strs.Length; i++)
            {
                res[i] = ValidateString(strs[i]);
            }

            return res;
        }

        // take file from ftp server
        //------
        public Image take_ftp(string file_take)
        {
            try
            {
                // Get the object used to communicate with the server.
                FtpWebRequest request =
(FtpWebRequest)WebRequest.Create("ftp://fit-hub.kiev.ua/www/fit-
hub.kiev.ua/photo/" + file_take);
                request.Method = WebRequestMethods.Ftp.DownloadFile;

                // This example assumes the FTP site uses anonymous logon.
                request.Credentials = new NetworkCredential("h33196i",
"2@SiPeOYk1*12m");//set our login data

                FtpWebResponse response = (FtpWebResponse)request.GetResponse();

                Stream responseStream = response.GetResponseStream();

                var image = Image.FromStream(responseStream);
                return image;
                //pbFoto.Image = image;

                //StreamReader reader = new StreamReader(responseStream);
                // Console.WriteLine(reader.ReadToEnd());
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
                return null;
            }

        }
        //send files to ftp server
        public void send_ftp(string file_send, string file_sourse, PictureBox pbFoto)
        {
            // Get the object used to communicate with the server.
            //----------set out data
```

```
        FtpWebRequest request = (FtpWebRequest)WebRequest.Create("ftp://fit-
hub.kiev.ua/www/fit-hub.kiev.ua/photo/" + file_send);
        //request.Method = WebRequestMethods.Ftp.DownloadFile;
        request.Method = WebRequestMethods.Ftp.UploadFile;
        // request.Method = WebRequestMethods.Ftp.ListDirectory;

        // This example assumes the FTP site uses anonymous logon.
        request.Credentials = new NetworkCredential("h33196i",
"2@SiPeOYk1*12m");//set our login data

        // Copy the contents of the file to the request stream.
        //---------
        //Image img = Image.FromFile(@"C:\Lenna.jpg");
        byte[] fileContents;
        using (MemoryStream ms = new MemoryStream())
        {
           pbFoto.Image.Save(ms, System.Drawing.Imaging.ImageFormat.Jpeg);
           fileContents = ms.ToArray();
        }

        //---
        /*--decoding file text
        byte[] fileContents;
        using (StreamReader sourceStream = new StreamReader(file_sourse))
        {
           fileContents = Encoding.UTF8.GetBytes(sourceStream.ReadToEnd());
        }
        end of decoding*/

        // FtpWebResponse response = (FtpWebResponse)request.GetResponse();
        request.ContentLength = fileContents.Length;

        using (Stream requestStream = request.GetRequestStream())
        {
           requestStream.Write(fileContents, 0, fileContents.Length);
        }
     }

  }
}
```

Додаток Б

Демонстраційний матеріал

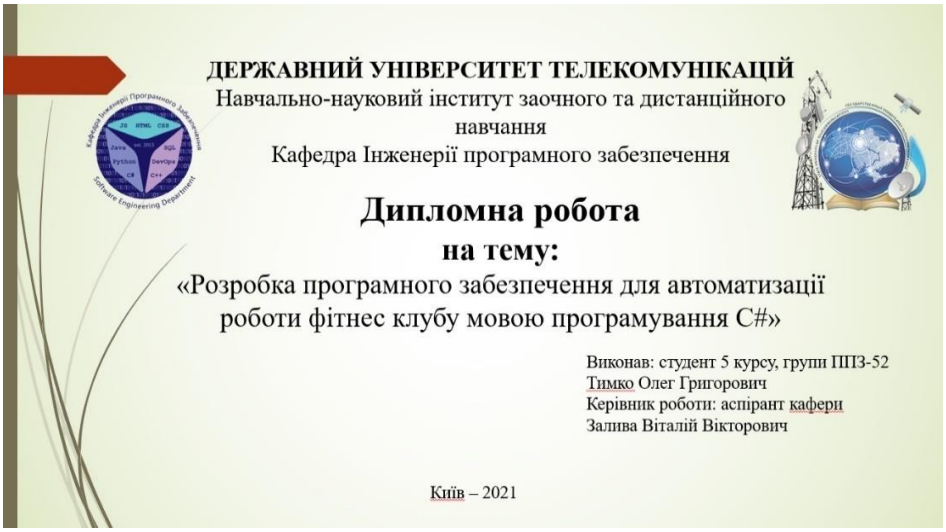


Рисунок Б.1 – Титульний слайд

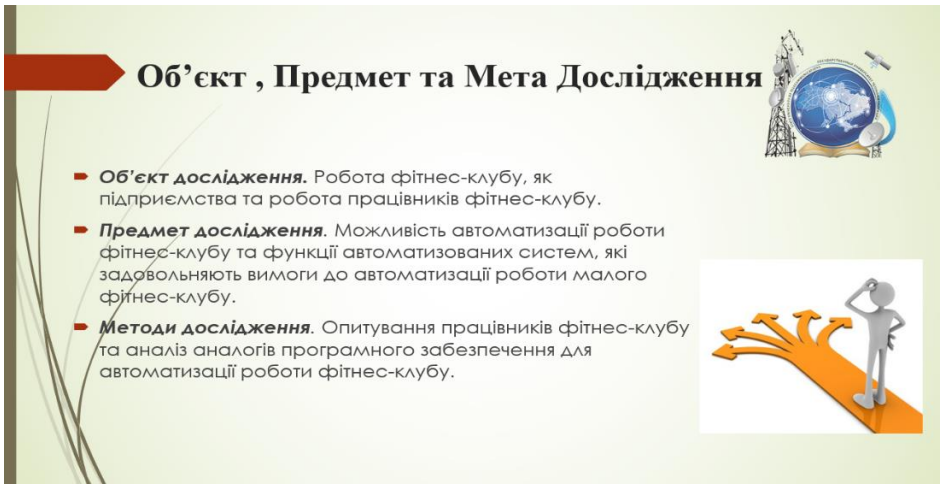


Рисунок Б.2 – Об’єкт, предмет та мета дослідження

Рисунок Б.3 – Актуальність вибраної теми

Рисунок Б.4 – Автоматизація роботи фітнес-клубу

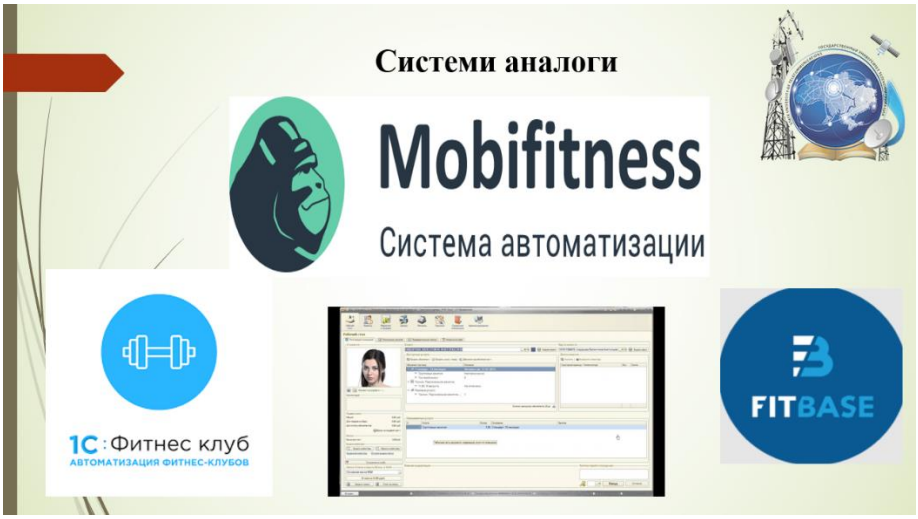


Рисунок Б.5 – Системи аналоги

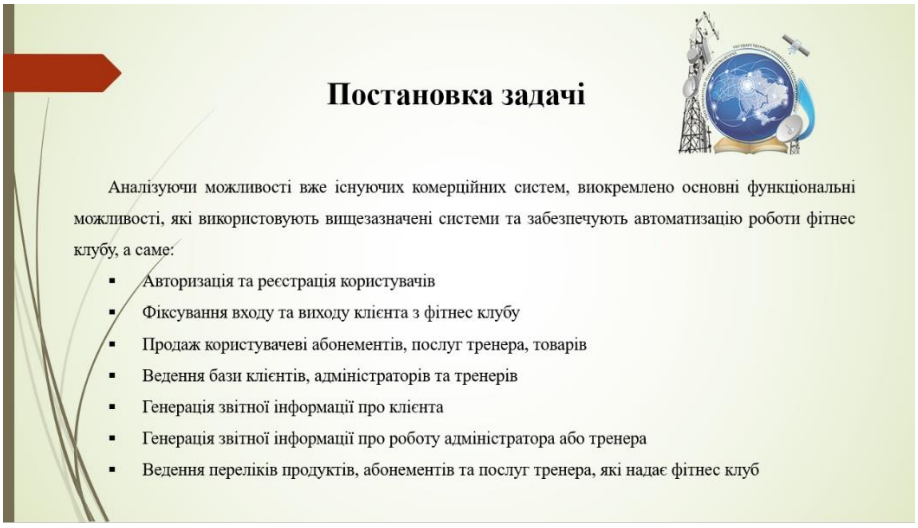


Рисунок Б.6 – Постановка задачі

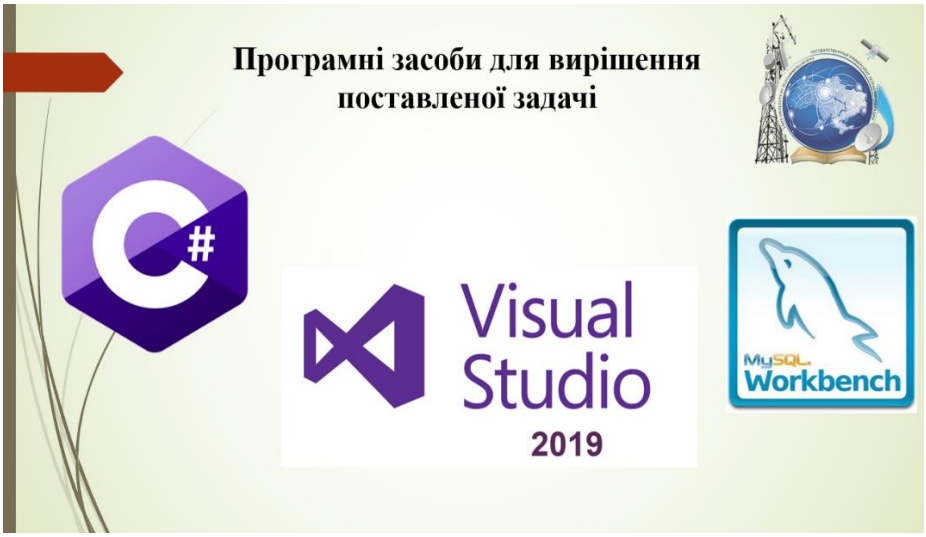


Рисунок Б.7 – Програмні засоби для вирішення поставленої задачі

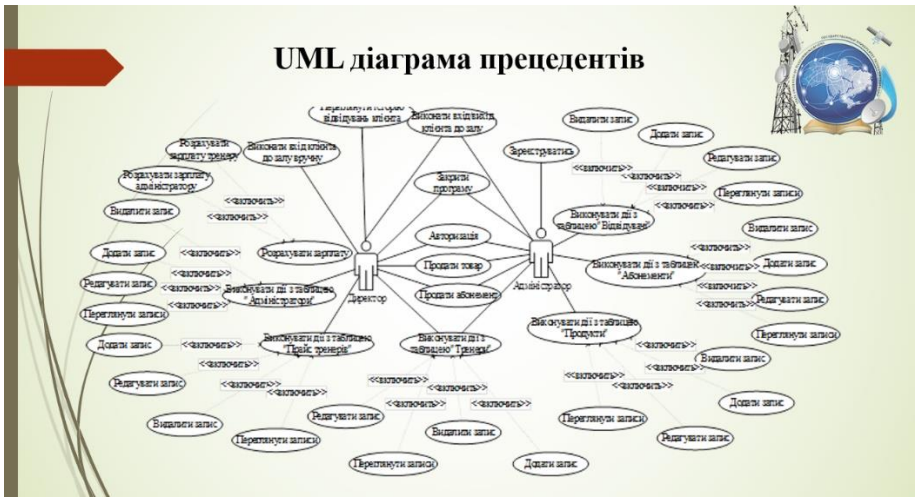


Рисунок Б.8 – UML діаграма прецедентів

Рисунок Б.9 – Алгоритм роботи програми

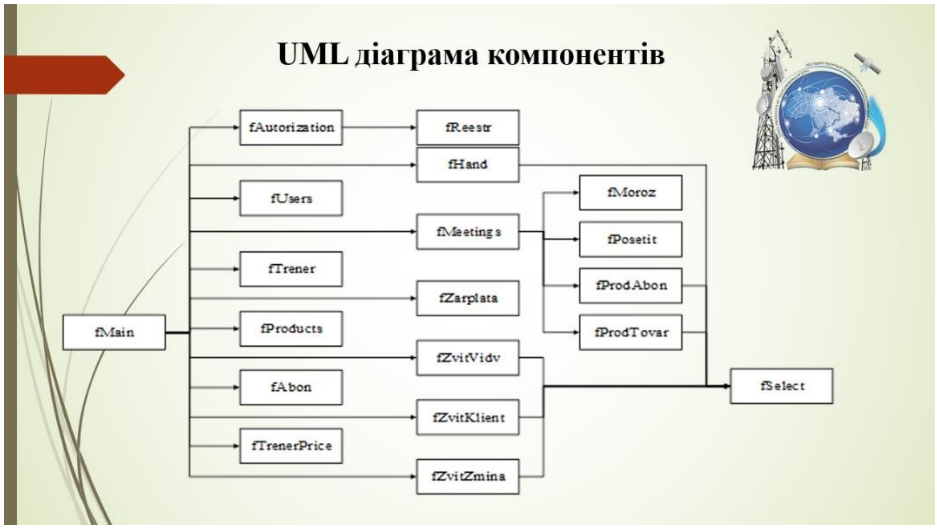


Рисунок Б.10 – UML діаграма компонентів

Рисунок Б.11 – Схема бази даних

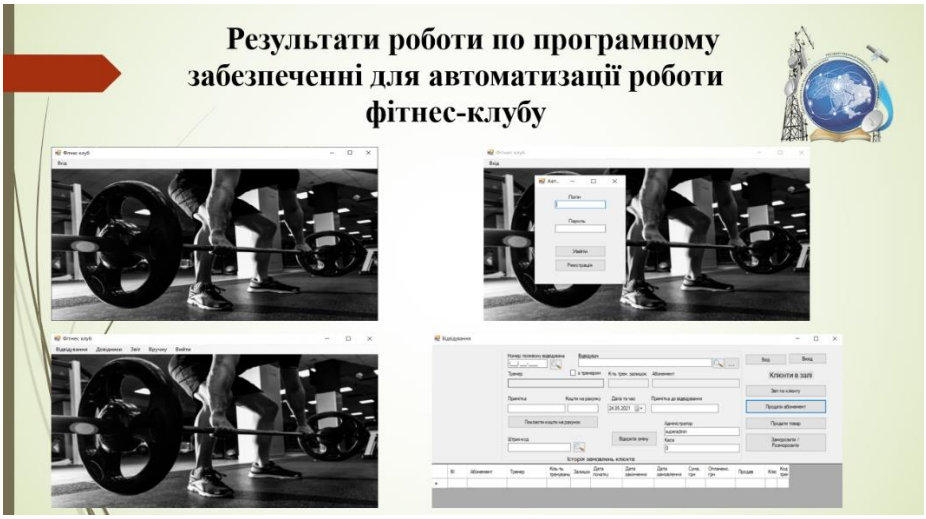


Рисунок Б.12 – Результати роботи по програмному забезпеченні для автоматизації роботи фітнес-клубу

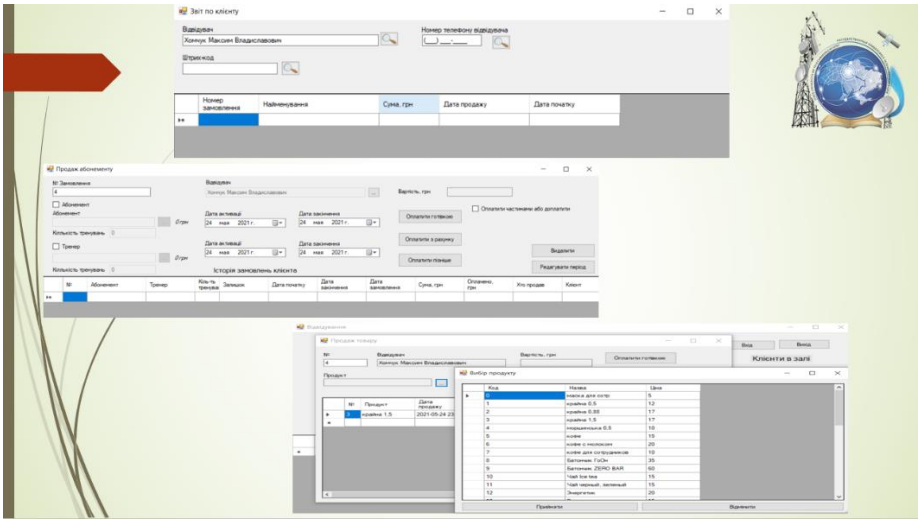


Рисунок Б.13 – Результати роботи по програмному забезпеченні для автоматизації роботи фітнес-клубу

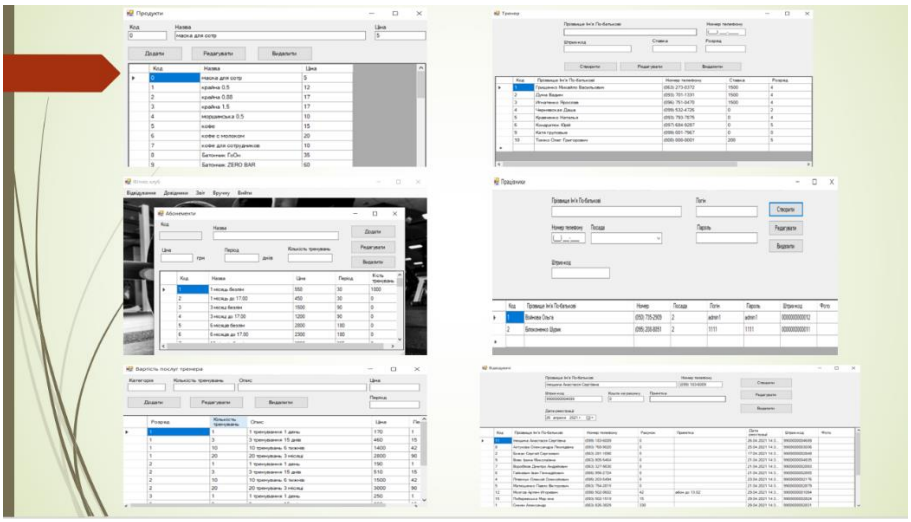


Рисунок Б.14 – Результати роботи по програмному забезпеченні для автоматизації роботи фітнес-клубу

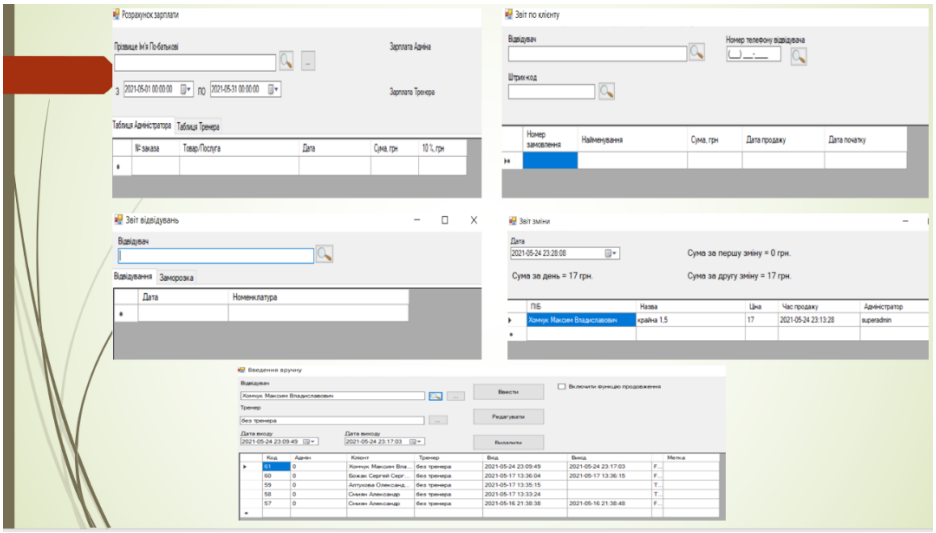


Рисунок Б.15 – Результати роботи по програмному забезпеченні для автоматизації роботи фітнес-клубу

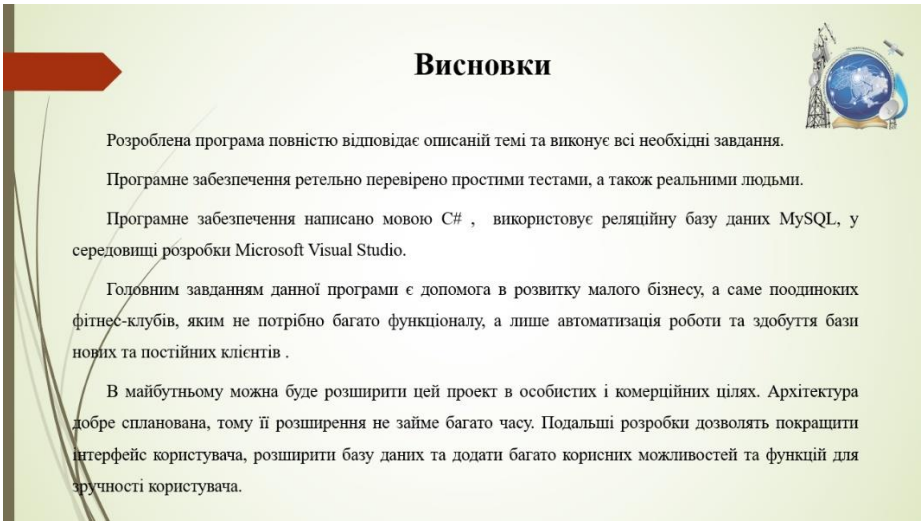


Рисунок Б.16 – Висновки

Рисунок Б.17 – Дякую за увагу