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 КОНФІГУРАЦІЇ ПЕРСОНАЛЬНОГО КОМП'ЮТЕРА МОВОЮ C#»

Виконав: студент 4 курсу, групи ПД-42
спеціальності
121 Інженерія програмного забезпечення
(шифр і назва спеціальності/спеціалізації)

Тригуб В. Ю.
(прізвище та ініціали)
Керівник Коваленко Д.С
(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»
Спеціальність підготовки – 121 «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії програмного забезпечення
_____Негоденко О.В.
“____” _____2021 року

З А В Д А Н Н Я

НА БАКАЛАВРСЬКУ РОБОТУ СТУДЕНТА

ТРИГУБУ ВЛАДИСЛАВУ ЮРІЙОВИЧУ

(прізвище, ім’я, по батькові)

1. Тема роботи: «Розробка програмного забезпечення для конфігурації персонального комп’ютера мовою C#»

Керівник роботи: Коваленко Д.С., асистент кафедри ІПЗ
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом вищого навчального закладу від «12» березня 2021 року № 65.
2. Строк подання студентам роботи «01» червня 2021 року

3. Вхідні дані до роботи

Теоретичні відомості відносно розробки програмного забезпечення на мові програмування C#;
Реляційна система керування базами даних SQLite;
Середовище розробки програмного продуку Visual Studio 2019

4. Зміст розрахунково-пояснювальної записки(перелік питань, які потрібно розробити).

4.1 Актуальність обраної теми.

4.2 Опис використаних інструментів для розробки.
4.3 Порівняння аналогів програмного продукту.
4.4 Унікальність продукту.
4.5 Розробка.
4.6 Тестування.

5. Перелік демонстраційного матеріалу (назва основних слайдів)

1. Мета, об’єкт та предмет дослідження;
2. Огляд аналогів;
3. Засоби реалізації програмного продукту;
4. Діаграма використання;
5. Висновки.

6. Дата видачі завдання «19» жовтня 2021

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврської роботи	Строк виконання етапів	Примітка
1.	Підбір науково-технічної літератури	19.04 – 22.04	
2.	Дослідження середовища розробки	26.04 – 29.04	
3.	Дослідження та аналіз аналогів	30.04 – 04.05	
4.	Розробка програмного продукту	05.05 – 21.05	
5.	Створення документації	24.05 – 26.05	
6.	Здача роботи	01.06.2021	

Студент _____ Тригуб В. Ю.
(підпис) (прізвище та ініціали)
Керівник роботи _____ Коваленко Д.С.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Текстова частина дипломної роботи містить 32с., 2 табл., 8 рис., 2 дод., 14 джерел.

Об'єкт дослідження – використання навичок та знань в розробці програмного забезпечення для вивчення апаратної складової персонального комп'ютера.

Предмет дослідження – програмний продукт, що дає можливість підібрати комплектуючі персонального комп'ютера та зрозуміти їхню взаємодію.

Мета дослідження – вивчення сумісності комплектуючих ПК, а також створення ПЗ для ознайомлення з апаратною частиною ПК іншими користувачами.

Під час виконання дипломної роботи було проведено дослідження комп'ютерних технологій та ролі персональних комп'ютерів в суспільстві. Проаналізовано ринок персональних комп'ютерів та їх апаратної складової, порівняно існуючі аналоги програмного продукту, виявлені їх переваги та недоліки.

Враховуючи всі переваги та недоліки програмного продукту було виявлено, що головною проблемою аналогів є необхідність у постійному з'єднанні з мережею інтернет для конфігурації персонального комп'ютеру, водночас даний дипломний проект має власну базу даних, яка містить інформацію по комплектуючим ПК незалежно від того, чи присутній інтернет зв'язок.

В якості зберігання інформації в базі даних було використано реляційну систему керування базами даних SQLite.

Крім цього, під час виконання дипломної роботи було спроектовано діаграму варіантів використання, яка надає змогу зрозуміти взаємодію користувача та системи.

В результаті проведених досліджень було вирішено розробити програмний продукт використовуючи об'єктно-орієнтовану мову програмування «C#» та середовище розробки Microsoft Visual Studio.

Ключові слова: C#, Microsoft Visual Studio, SQLite, ПК, ПЗ.

Зміст

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

IT – Information Technology. Обширна галузь, яка являє собою набір процесів для обробки, передачі та аналізу інформації.

ПЗ – програмне забезпечення.

ПК – персональний комп’ютер.

IDC – International Data Corporation (міжнародна компанія маркетингових досліджень)

.NET – програмна технологія, як платформа для створення як звичайних програм, так і веб-застосунків.

API — програмний інтерфейс додатку.

БД – база даних.

СКБД – система керування базами даних.

SQL – декларативна мова програмування для взаємодії користувача з базами даних.

SQLite – полегшена реляційна система керування базами даних.

ПКМ – права кнопка миші

ВСТУП

Об’єкт дослідження – використання навичок та знань в розробці програмного забезпечення для вивчення апаратної складової персонального комп’ютера.

Предмет дослідження – програмний продукт, що дає можливість підібрати комплектуючі персонального комп’ютера та зрозуміти їхню взаємодію.

Мета дослідження – вивчення сумісності комплектуючих ПК, а також створення ПЗ для ознайомлення з апаратною частиною ПК іншими користувачами.

В сучасному світі комп'ютери проникли в усі сфери діяльності людини, починаючи з початкової освіти і закінчуючи вивченням новітніх технологій, вивчення нових видів матерії. Застосування комп'ютерних технологій полегшує процес освіти в середніх і вищих навчальних закладах як самих учнів, студентів, так і робочого персоналу.

Персональні комп’ютери для жителів багатьох країн світу стали звичними домашніми апаратами, як, скажімо, телефон або телевізор. Це записник і довідник, бухгалтер і перекладач, домашній учитель і екскурсовод, кінотеатр, та засіб зв’язку. Для людей з обмеженими можливостями часом це єдиний спосіб не тільки спілкування, а й завдяки сучасним комп'ютерним технологіям такі люди можуть себе творчо реалізувати або отримати роботу.

В зв’язку з цим було вирішено створити програмне забезпечення, яке надає змогу користувачу підібрати потрібні комплектуючі згідної їхньої відповідності та отримати орієнтовну ціну за ПК.

В процесі дослідження будуть розкриті наступні завдання:

- Актуальність теми дипломної роботи;
- Аналіз аналогів та пошук їх недоліків;
- Визначення вимог до програмного продукту;
- Вибір технологій для розробки;
- Опис функціоналу ПЗ.

Актуальність теми дипломного проекту полягає в тому, що більша частина користувачів ПК не розуміється на апаратній частині ПК та не має уявлення, що кожна складова комп’ютеру є окремим елементом, який є унікальним в своєму роді та при взаємодії з іншими частинами може як працювати так і ні.

Даним програмним забезпеченням буде надано можливість користувачу підібрати комплектуючі ПК, які будуть пов’язані між собою як технічно, так і функціонально.

Наукова новизна полягає в розробці програмного забезпечення для конфігурації ПК та взаємодії комплектуючих між собою.

Практична значущість полягає у можливості використання програмного продукту, як фізичним особам для власного підбору ПК так і підприємствам чи магазинам для комплектації та подальшого продажу ПК.

В якості середовища розробки програмного продукту для написання коду було обрано «Microsoft Visual Studio 2019». Програмний продукт буде написаний на мові програмування «C#» з використанням реляційної системи керування базами даних SQLite.

1 РОЗРОБКА ПРОГРАМНОГО ПРОДУКТУ. ЙОГО АКТУАЛЬНІСТЬ

1.1 Розвиток програмного забезпечення

Сучасний світ неможливо уявити без програмного забезпечення (ПЗ). Розробка ПЗ перетворилася на окрему галузь, що займає чільне місце у світовій економіці, в тому числі в Україні. У 2020 році світові витрати на IT послуги склали 1849 млрд дол. США, і ця галузь зростає щороку, в тому числі очікується зростання на 4,2% у 2021 році. При цьому із розвитком технологій, з новою технологічною революцією, значення розробки програмного продукту лише зростатиме.

Сьогодні Україна не залишається осторонь глобальних процесів. Наша держава одна з найкращих для розміщення замовлень у сфері аутсорсингу бізнес-процесів та IT, що підтверджує 24 місце у рейтингу привабливості.

Ринок розробки ПЗ сьогодні є здебільшого експортно-орієнтованим і займає третє місце за обсягами виручки, тож є одним із головних джерел надходження валютних коштів в Україну. Експортно-орієнтований сектор розробки ПЗ у 2016 році склав 3,2 млрд дол. США, що становить близько 26% експорту послуг за відповідний період і 7% сукупного експорту товарів і послуг. Таким чином ринок позитивно впливає на стабільність курсу національної валюти і цін в державі[4].

Враховуючи політику світового ринку, розробка ПЗ є актуальним та прибутковим методом заробітку, а також спосіб втілення креативних ідей в сучасне життя.

Під час розробки програмного забезпечення важливо правильно сформувати етапи проектування, а також підібрати інструменти які будуть використовуватись в процесі розробки бажаного проекту. Для розробки програмного забезпечення необхідно обрати мову програмування, яка дає змогу реалізувати даний проект в реальність маючи базові знання та навички в створенні програмного забезпечення.

Станом на сьогодні «C#» визначено флагманською мовою програмування

корпорації Microsoft, бо вона найповніше використовує нові можливості .NET, а також є однією із найпоширеніших мов програмування.

Враховуючи ці дані було прийнято рішення реалізувати ідеї проекту на мові програмування «C#».

1.2 Актуальність стаціонарних комп'ютерів

Згідно статистики компанії Canalys, в 2020 році (рис 1.1), поставки ПК досягли 297 мільйона штук, збільшившись в річному вираженні на 11%. Їх колеги з IDC наводять ще більш високу оцінку - 302 мільйони (+ 13,1%). У той же час Gartner також погоджуються з колегами, що 2020 рік став вдалим роком для ринку ПК, ознаменувався великими зростанням з 2010 року[2].

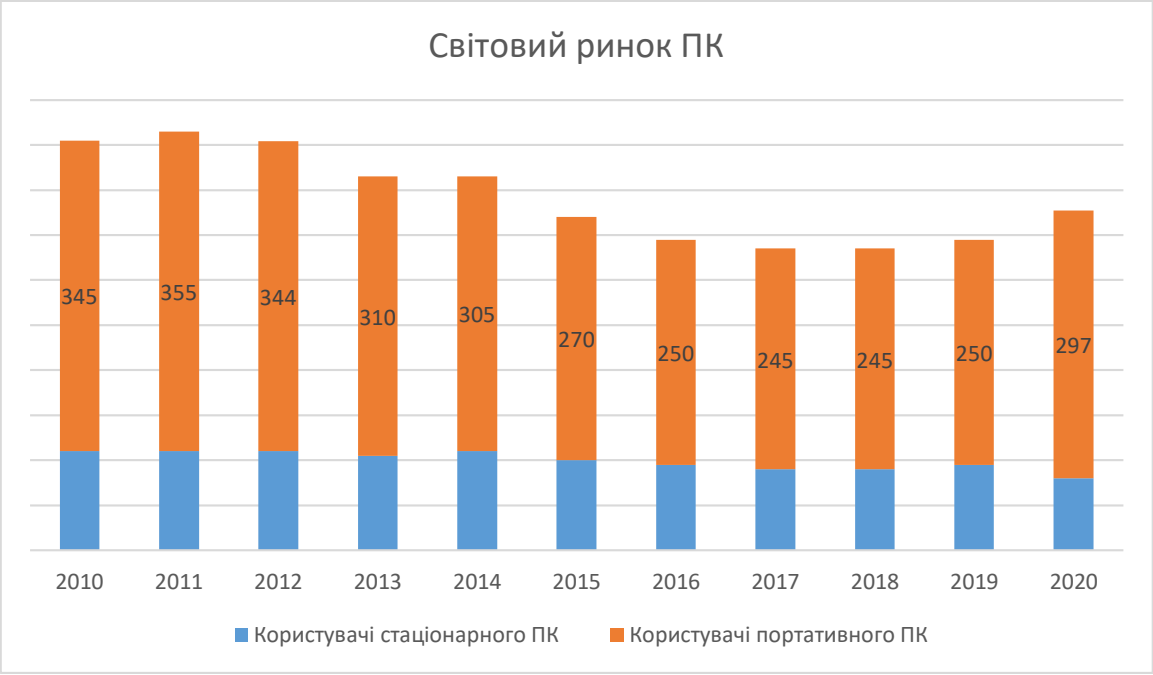


Рисунок 1.1 – Популярність персональних комп'ютерів

Не зважаючи на те, що більшість людей надають перевагу портативним комп'ютерам, беручи до уваги мобільність та компактність є велика кількість

користувачів, які використовують стаціонарні комп’ютери, звертаючи до уваги надійність та продуктивність роботи приладу.

Варто взяти до уваги також статистику використання операційних систем(рис. 1.2), які використовуються в ПК для взаємодії користувача.

Рисунок 1.2 – Статистика популярності ОС

Якщо брати окремо операційні системи які встановлені на комп’ютерах, то тут лідирує Windows – 76,58% всіх комп’ютерів. OS X – 18,93 і Linux – 1,62%.[7]

Виходячи з даної статистики було вирішено розробити програмний продукт, який буде адаптований під операційну систему Windows.

Більша частина користувачів стаціонарних ПК не мають уяви стосовно того, яким чином влаштований комп’ютер, з яких комплектуючих він складається, та яка роль деталей, з яких складається ПК.

Для початку бажано наголосити, що ПК це не лише процесор, материнська плата, відеокарта, тощо, а також і монітор, комп’ютерна миша, а також клавіатура, тому в темі дипломного проекту буде більш детальніше відкрито питання

системного блоку, оскільки конфігурація ПК буде відбуватися на основі збірки системного блоку.

Отже ми вже можемо зрозуміти, що головними комплектуючими в системному блоці виступають материнська плата, процесор та відеокарта. Це є одними із цінніших деталей системного блоку, проте варто розуміти, що крім цих деталей системний блок також складається з не менш важливої деталі, а саме блоку живлення. Він відіграє роль джерела живлення, призначене для забезпечення енергії в електроприладах. Також системний в системному блоці присутня: оперативна пам’ять, жорсткий диск, додаткові накопичувачі, системи охолодження, а також сам корпус системного блоку.

Саме з такими даними ми будемо працювати в рамках даного дипломного проекту, та спробуємо донести до користувача відповідність комплектуючих між собою та їхня взаємодія.

1.3 Середовище розробки Microsoft Visual Studio

Microsoft Visual Studio – серія продуктів фірми Майкрософт, які містять інтегроване середовище розробки програмного забезпечення та низку інших інструментальних засобів. Ці продукти дають змогу розробляти як консольні програми, так і програми з графічним інтерфейсом, включно 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.

Середовище Visual Studio дозволяє розробляти додатки, використовуючи різні мови програмування: «Visual C#», «Visual Basic», «Java», «Visual C++», «Python». Також існує можливість розробляти додатки не тільки під Windows, а і під інші популярні платформи: Android, iOS.

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

1.4 Мова програмування C#

Мова програмування C# – об'єктно-орієнтована мова програмування з безпечною системою типізації для платформи .NET, являється простою і в той же час потужною мовою програмування, що дозволяє програмістам створювати багатофункціональні програми. Розроблена Андерсом Гейлсбергом, Скотом Вілтамутом та Пітером Гольде під егідою Microsoft Research [6].

Синтаксис мови програмування C# близький до C++ і Java. Мова має сувору статичну типізацію, підтримує поліморфізм, перевантаження операторів, вказівники на функції-члени класів, атрибути, події, властивості, винятки, коментарі у форматі XML. Перейнявши багато що від своїх попередників – мов C++, Delphi, Модула і Smalltalk – C#, спираючись на практику їхнього використання, виключає деякі моделі, що зарекомендували себе як проблематичні при розробці програмних систем, наприклад, множинне спадкування класів (на відміну від C++).

Система Common Language Runtime (CLR) – «загальне середовище виконання мов» – це компонент пакету Microsoft .NET Framework, віртуальна машина, на якій виконуються всі мови платформи .NET Framework [4].

CLR транслюється початковий код в байт-код на мові IL, реалізація компіляції якого компанією Microsoft називається MSIL, а також надає MSIL-програмам (а отже, і програмам, написаним на мовах високого рівня, що підтримують .NET Framework) доступ до бібліотеки класів .NET Framework, або так званою .NET Framework Class Library(FCL).

Мова C# розроблялась спеціально, як мова програмування прикладного рівня для CLR і тому вона залежить, перш за все, від можливостей самої CLR. Це стосується, перш за все, системи типів C#. Присутність або відсутність тих або інших виразних

особливостей мови диктується тим, чи може конкретна мовна особливість бути трансльована у відповідні конструкції CLR. Так, з розвитком CLR від версії 1.1 до 2.0 значно збагатився і сам C#; подібної взаємодії слід чекати і надалі. (Проте ця закономірність буде порушена з виходом C# 3.0, що є розширеннями мови, що не спираються на розширення платформи .NET.) CLR надає C#, як і всім іншим .NET орієнтованим мовам, багато можливостей, яких позбавлені «класичні» мови програмування. Наприклад, збірка сміття не реалізована в самому C#, а проводиться CLR для програм, написаних на C# точно так, як і це робиться для програм на VB.NET, J#, тощо[4].

До основних переваг мови C# можна віднести:

- C# – це об’єктно-орієнтована мова програмування, яка користується великим попитом. Нею володіють близько 60 % усіх програмістів;
- повний і добре визначений набір основних типів;
- вбудована підтримка автоматичної генерації XML — документації;
- автоматичне звільнення динамічної розподіленої пам’яті;
- повний доступ до бібліотеки базових класів .NET, а також легкий доступ до Windows API;
- просте зміна ключів компіляції. Дозволяє отримувати виконувані файли або бібліотеки компонентів .NET, які можуть бути викликані іншим кодом так само, як елементи управління ActiveX (компоненти COM);
- має 100 000 оптимізованих методів. при чому в мові «C#» є функції Delphi, Ассемблера, C++ та інших мов.

1.5 Порівняння аналогів ПЗ

На етапі проектування програмного продукту було проведено аналіз вже існуючого програмного забезпечення, яке схоже за функціоналом теми дипломного проекту, тобто здійснено пошук аналогів ПЗ. Існуючі аналоги представлені

виключно в вигляді інтернет ресурсів для конфігурації ПК та є в вільному доступі користувачам, проте не всі інтернет ресурси здійснюють підбір комплектуючих, які будуть взаємодіяти між собою.

Більшість аналогів використовують підбір вже готових комплектацій та не надають змоги користувачу самостійно підібрати комплектуючі, які його цікавлять, проте все ж таки існує інтернет магазин під назвою «Telemart», на сайті якого інтегрована система пошуку та взаємодії комплектуючих між собою для подальшого придбання.

Після проведеного огляду було складено порівняльну характеристику:

Таблиця 1.1 Порівняння ПЗ з аналогами

Назва	Фільтри пошуку	Кросплатформеність	Широкий асортимент	Можливість придбання комплектуючих	Можливість взаємодії локально
Telemart	Так	Так	Так	Так	Ні
AutoPC	Так	Ні	Ні	Ні	Так

Беручи до уваги інформацію з даної таблиці до переваги розроблюваного програмного продукту можливо віднести можливість роботи застосунку локально, тобто без використання інтернету, що дає перевагу перед інтернет магазином.

1.6 Висновки

Перед початком роботи над програмним продуктом, було проведено ринку персональних комп’ютерів та було сформовано бачення майбутнього в даному напрямку. Відповідно до статистики, упродовж з 2010 року в світі спостерігалось падіння поставок ПК на незначну частину, проте кількість активних користувачів ПК залишалася на високому рівні. Враховуючи статистику використання операційної системи для ПК стало зрозуміло те, що більша частина населення світу працює з ОС

Windows. Було сформовано бачення проекту та описано інструменти для розробки, виявлені їх плюси та мінуси. Також було наведено приклад аналогу програмного продукту та складено порівняльну таблицю, в якій показані переваги продукту перед його аналогом, а також недоліки, в яких можливо сформувати цілі на майбутні вдосконалення програмного продукту.

В результаті дослідження в якості розробки програмного продукту буде виступати мова програмування «C#», через її популярність, доступність та зручність в розробці для різних платформ, як стаціонарних так і мобільних, а також через офіційну підтримку та можливість використання повного спектру функцій та методів програмного продукту.

Середовищем розробки буде виступати «Microsoft Visual Studio» , тому що воно має низку переваг під час роботи на «C#» зокрема зручні функції опрацювання з базою даних, а також API. Також це середовище має безкоштовну версію, якої достатньо для комфортної роботи з мовою «C#».

За елемент наповнення продукту будуть відповідати дані комплектуючих, які були взяті з мережі інтернет та пов’язані між собою взаємодією інтерфейсів підключення.

Оскільки спектр комплектуючих ПК є доволі великий, в програмному продукті буде представлено декілька варіантів взаємодії виробників сімейства процесорів AMD та Intel, а також інших комплектуючих пов’язаних інтерфейсом підключення між собою.

Після аналітичної частини роботи будуть складені специфікації до системи програмного продукту з урахування недоліків та переваг аналогів.

2 ВИМОГИ ТА ДОСЛІДЖЕННЯ ПРОГРАМНОГО ПРОДУКТУ

2.1 Вимоги до програмного забезпечення

Вимоги до програмного забезпечення – це набір правил та специфікацій, які повинні бути реалізовані в програмному продукті, для його розуміння. Класифікація вимог до програмного забезпечення поділяється на декілька типів.

Ґрунтуючись на характерних ознаках, вимоги до ПЗ класифікуються за наступними категоріями:

I. Функціональні, ті що відносяться до системної поведінки:

- Бізнес – визначають основне призначення продукту;
- Користувацькі – дозволяють визначити завдання, покладені на програмне рішення;
- Системні – специфікація, що охоплює дії, які буде виконувати ПЗ.

II. Нефункціональні, ті що визначають характер системної поведінки, які включають вимоги до:

- Документування;
- дизайну;
- надійності;
- простоти використання;
- безпеки;
- показниками призначення;
- експлуатації;
- персоналу;
- мобільності;
- автономності;
- зовнішніх впливів;

– бізнес-правилами;

Якщо розглядати функціональні вимоги, можливо скласти короткий опис, а саме те, що бізнес вимога описує економічну складову проекту, спосіб отримання прибутку та відіграє одну з головних ролей в розробці програмного продукту. Не менш важливою вимогою є користувацька, оскільки вона визначає способи взаємодії програмного продукту з користувачем, а також системна – описує роботу та поведінку системи, основні її дії та функції.

Розглядаючи нефункціональні вимоги, можливо сформувати думку, що вони являють собою опис того, як повинен працювати програмний продукт, вимоги до його безпеки, продуктивності, зовнішнього вигляду та обмеження, які описують можливі варіанти реалізації цих вимог.

2.2 Варіанти використання програмного продукту

Для того, щоб більш точно зрозуміти, як повинна працювати система, все частіше використовується опис функціональності системи через варіанти використання (Use Case або прецеденти).

Варіанти використання – це опис послідовності дій, які може здійснювати система у відповідь на зовнішні впливи користувачів або інших програмних систем. Варіанти використання відображають функціональність системи з точки зору отримання відчутного результату для користувача, тому вони точніше дозволяють трактувати функції за значимістю одержуваного результату.

Варіанти використання призначені в першу чергу для визначення функціональних вимог до системи і керують усім процесом розробки. Всі основні види діяльності такі як аналіз, проектування, тестування виконуються на основі варіантів використання.

Під час аналізу і проектування варіанти використання дозволяють зрозуміти як результати, які хоче отримати користувач впливають на архітектуру системи і як

повинні поводитися компоненти системи, для того щоб реалізувати потрібну для користувача функціональність.

Діаграма варіантів використання складається з акторів, для яких система виробляє дію і власне дії Use Case, які описують те, що актор хоче отримати від системи. Актор позначається чоловічка, а Use Case - овалом.

Діаграма використання програмного продукту продемонстрована на (рис. 2.1)

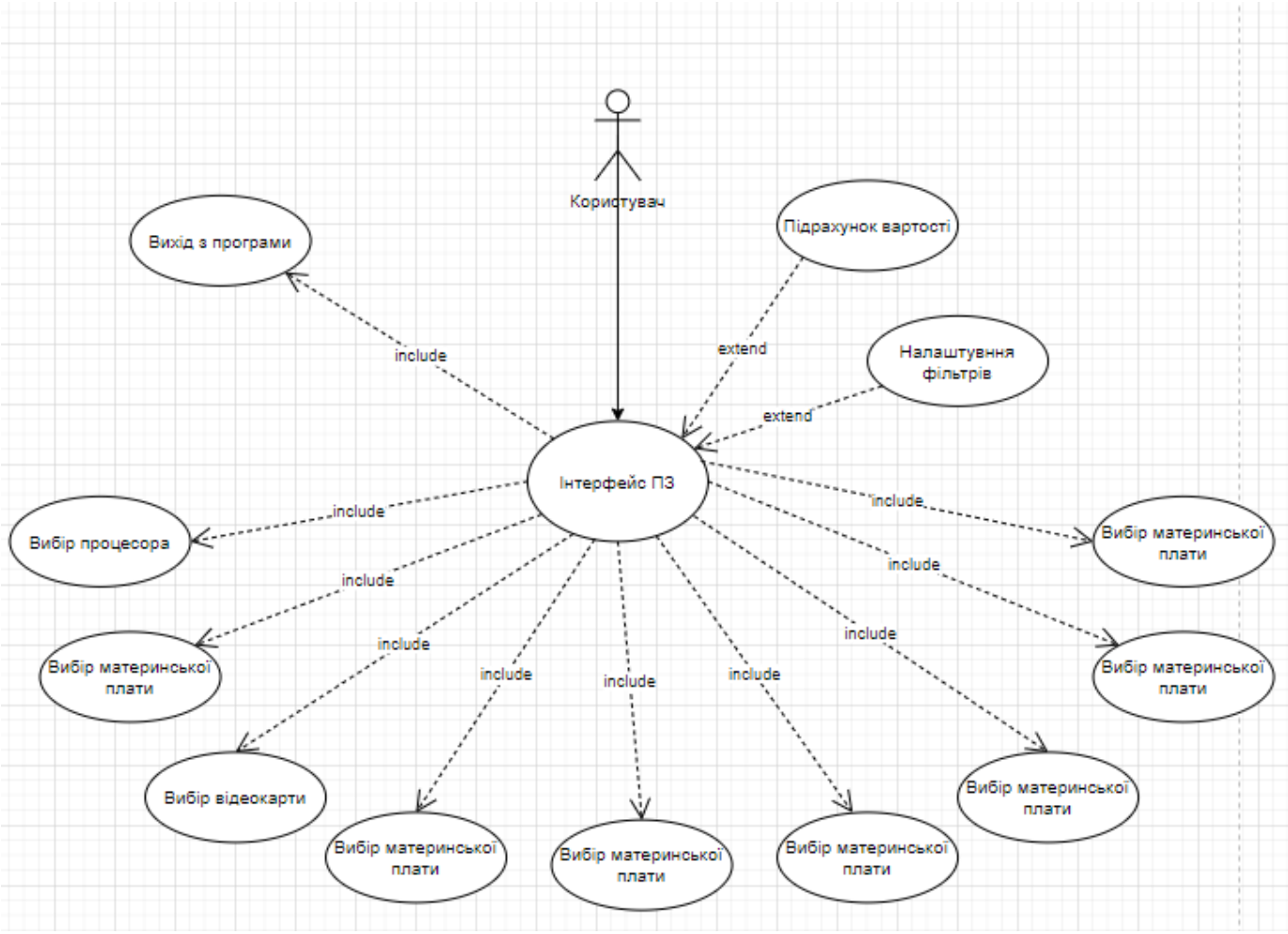


Рисунок 2.1 Діаграма використання (Use case)

2.3 Опис структури бази даних

База даних (БД) - упорядкований набір логічно взаємопов’язаних даних, що використовується спільно, та призначений для задоволення інформаційних потреб

користувачів. Вхідні та вихідні дані, які опрацьовуються сайтом надходять саме з БД.

SQLite – реляційна СКБД, міститься в бібліотеці мови С. Вільна, серверна та функціональна, без проблем працює з веб-додатками. Спроектована у 2000 році. Мета дизайну SQLite полягала в тому, щоб дозволити програмі працювати без встановлення СКБД чи вимагати адміністратора БД.

На відміну від багатьох інших СКБД, SQLite не є механізмом бази даних клієнт-сервер. Скоріш вона вмонтована в кінцеву програму. SQLite є сумісною з ACID і за стандарту SQL реалізує більшу його частину. Проте вона використовує синтаксис SQL, що динамічно і слабо типізований, з цього випливає, що немає гарантії цілісності домену.

SQLite є досить популярним вибором ПЗ вбудованої БД як для локального, так і для клієнтського зберігання в прикладних програмах, таких як, наприклад, браузери. Це, мабуть, найбільш широко розгорнутий двигун БД, так як на сьогоднішній день, він використовується у більшості браузерів, операційних системах, та вбудованими системах (наприклад, мобільними телефонами). SQLite має прив'язки до багатьох мов програмування .

Розглянемо на прикладі дипломного проекту його структуру БД, яка зображена на (рис 2.2)

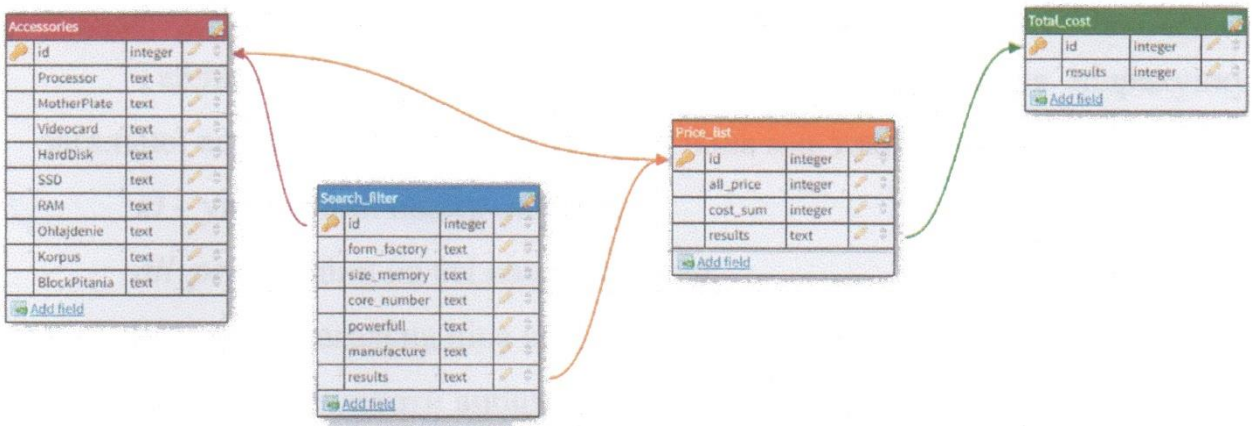


Рисунок 2.2 База даних в SQLite

В даній базі даних присутні 4 таблиці взаємодії користувача та системи.

Таблиця *Accessories* використовується в програмному продукті, як перелік всіх комплектуючих та має наступні поля вводу:

- *id* – унікальний номер об’єктів;
- *Processor* – перелік процесорів;
- *MotherPlate* – перелік материнський плат;
- *Videocard* – перелік відеокарт;
- *HardDisk* – перелік жорстких дисків;
- *SSD* – перелік твердотільних накопичувачів;
- *RAM* – перелік оперативної пам’яті;
- *Ohlajdenie* – перелік систем охолодження;
- *Korpus* – перелік корпусів;
- *BlockPitania* – перелік блоків живлення.

Таблиця *Search_filter* використовується в програмному продукті, як параметр швидкого пошуку за певною характеристикою, структура таблиці складається з:

- *id* – унікальний номер об’єктів;
- *form_factory* – тип об’єкту;
- *size_memory* – розмір пам’яті;
- *core_number* – кількість ядер;
- *powerful* – потужність;
- *manufacture* – тип процесора;
- *results* – результати фільтру.

Таблиця *Price_list* використовується в програмному продукті, як таблиця підрахунку загальної вартості всіх комплектуючих персонального комп’ютеру та містить в собі наступні значення:

- *id* – унікальний номер об’єктів;
- *all_price* – ціни на всі комплектуючі;
- *cost_sum* – загальна ціна обраних комплектуючих;

- *results* – результат вибірки.

Таблиця *Total_cost* використовується для підрахування загальної вартості всіх обраних комплектуючих персонального комп’ютера та складається з наступних полів:

- *id* – унікальний номер об’єктів;
- *results* – результат вибірки.

2.4 Висновки

В даному розділі були визначені основні вимоги до програмного забезпечення та їх класифікацію для подальшої реалізації у вигляді програмного продукту. Було продемонстровано діаграму використання програмного продукту для розуміння взаємодії користувача із системою.

Також було описано структуру бази даних, яка буде використовуватись в якості збереження інформації для подальшого використання користувачем.

Під час виконання цього етапу розробки були покращені теоретичні та практичні знання з аналізу вимог до програмного забезпечення, а також сформовано бачення подальшої розробки дипломного проекту.

3 РОЗРОБКА ТА ТЕСТУВАННЯ ПРОГРАМНОГО ПРОДУКТУ

3.1 Середовище розробки. Робота з матеріалами продукту

Для початку роботи необхідно встановити середовище розробки програмного забезпечення Microsoft Visual Studio, зробити це можливо на офіційному сайті корпорації Microsoft[5]. Під час встановлення середовища розробки бажано обрати версію «Community 2019» (рис 3.1), оскільки вона є безкоштовною, проте в версіях «Professional 2019» та «Enteprise 2019» присутній пробний період, який надає змогу протягом 7 днів отримати повний функціонал версії середовища розробки для розробки з використанням мови програмування C#.

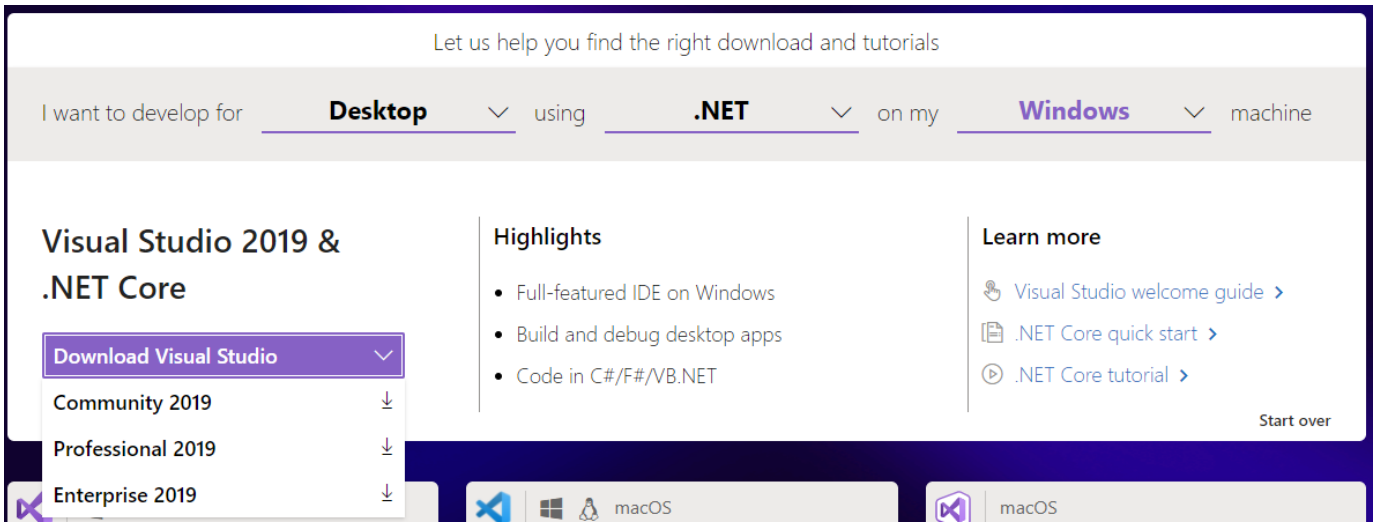


Рисунок 3.1 Встановлення середовища розробки

Після встановлення потрібно визначитись з типом проекту, в якому буде розроблюватись програмний продукт. Для реалізації теми дипломного проекту я вирішив створити проект Windows Forms (рис 3.2), оскільки в ньому вже присутня API для зручної взаємодії системи та користувача.

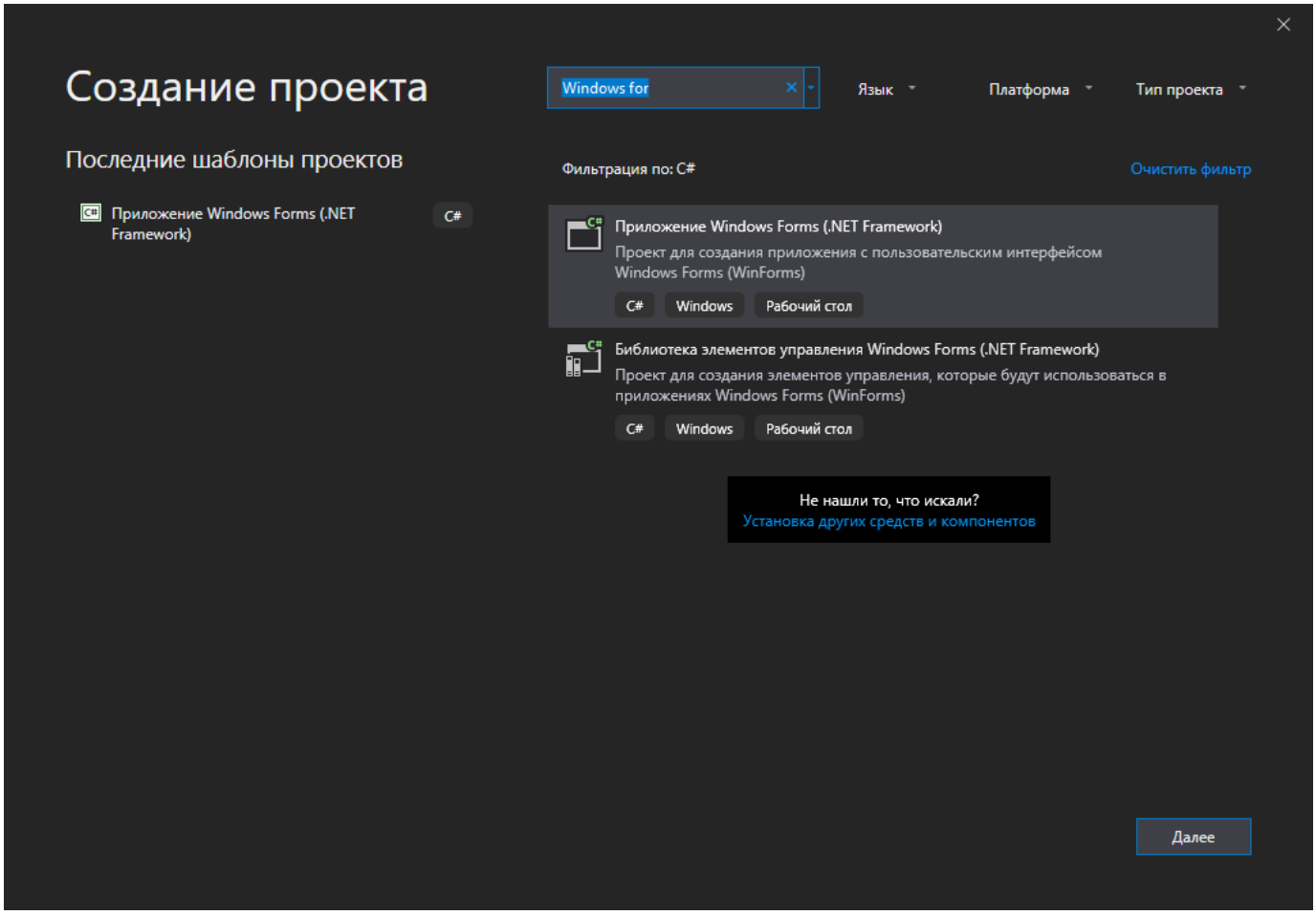


Рисунок 3.2 Створення проекту

Отже після створення проекту ми потрапляємо на форму взаємодії користувана, на якій ми можемо розмістити необхідні вхідні та вихідні дані, з якими буде взаємодіяти користувач.

Далі нам необхідно підключити реляційну систему керування базою даних SQLite, з якою ми будемо взаємодіяти для створення програмного продукту. Для цього потрібно перейти до оглядача рішень в правому куртку на натиснувши ПКМ на розділ «Посилання» нам потрібно обрати розділ «Управління пакетами Nu Get…» де в пошуку необхідно зазначити SQLite та підключити базу даних до проекту(рис 3.3).

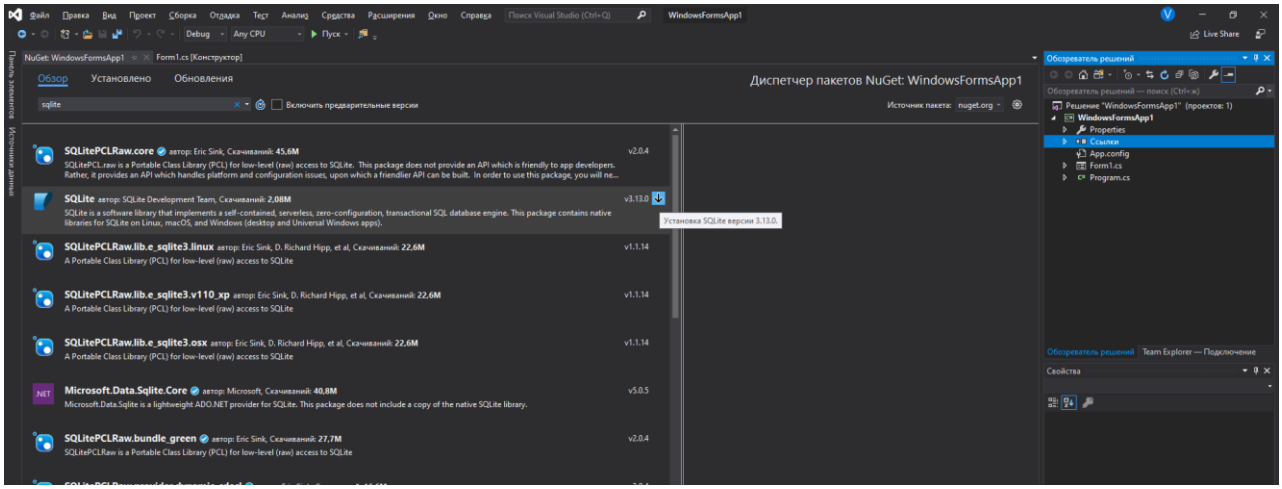


Рисунок 3.3 Підключення SQLite

Технічна частина підготовки до створення проекту завершена і вже можливо приступати до створення інтерфейсу користувача, а також написання функціоналу для виконання дипломного проекту.

3.2 Опис основних функцій

Розглянемо основні функції, які використовувались в розробці програмного продукту представлені у вигляді таблиці

Таблиця 3.1 Опис головного функціоналу ПЗ

Функціонал	Приклад коду
Додавання та видалення комплектуючих	`public void DeletedBlock(UserControl control)` `{` `  foreach (UserControl item in flowLayoutPanel1.Controls)` `  {` `  if (item.GetType() == control.GetType()) flowLayoutPanel1.Controls.Remove(item);` `  }` `  flowLayoutPanel1.Controls.Remove(control);` `}` `public void AddToBasketProc(UserControl contro)` `{` `  foreach (UserControl item in flowLayoutPanel1.Controls)` `  {` `  if (item.GetType() == contro.GetType()) flowLayoutPanel1.Controls.Remove(item);` `  }` `  flowLayoutPanel1.Controls.Add(contro);` `}` Приклад, як за допомогою методу *flowLayout* можливо контролювати додані поля або їх видалення

Продовження таблиці 3.1

Фільтр пошуку комплектуючих	 Для початку виконується перевірка на натиснення відповідних фільтрів, після чого перед користувачем з’являється результати фільтру по потрібному розділу.
Розрахунок вартості обраних комплектуючих	

```
private void button9_Click(object sender, EventArgs e)
{
    string filter = "";
    List<int> proizvosList = new List<int>();
    List<int> formfactor = new List<int>();
    int moshmin;
    flowLayoutPanel10.Controls.Clear();
    if (checkBox119.Checked)
    {
        proizvosList.Add(27);
    }
    if (checkBox120.Checked)
    {
        proizvosList.Add(33);
    }
    if (checkBox121.Checked)
    {
        proizvosList.Add(28);
    }
    if (checkBox122.Checked)
    {
        proizvosList.Add(34);
    }
    if (proizvosList.Count != 0)
    {
        filter += " and kp.Proizvod in(" +
                  string.Join(",", proizvosList.Select(n => n.ToString()).ToArray()) +
                  ")";
    }
    if (checkBox123.Checked)
    {
        formfactor.Add(1);
    }
    if (checkBox124.Checked)
    {
        formfactor.Add(2);
    }
    if (checkBox125.Checked)
    {
        formfactor.Add(3);
    }
    if (formfactor.Count != 0)
    {
        filter += " and kf.IdFormFactor in(" + string.Join(",", formfactor.Select(n => n.ToString()).ToArray()) + ")";
    }
    if (textBox3.Text.Length != 0)
    {
        if (Int32.TryParse(textBox3.Text, out moshmin)) filter += " and kp.Pitanie >= " + moshmin;
    }
    SortoutKorpus(filter);
```

```
private void button10_Click(object sender, EventArgs e)
{
    ItemAddCheckHelper ds = new ItemAddCheckHelper() ;
    ds.AllComlect(flowLayoutPanel1.Controls);
    float AALLPrice = 0;

    float motherprice = 0;
    foreach (UserControl item in flowLayoutPanel1.Controls)
    {
        if (item.GetType() == typeof(LeftPanelMotherView))
        {
            motherprice = ((LeftPanelMotherView)item).leftMother.Price;
        }
    }
    float procprice = 0;
    foreach (UserControl item in flowLayoutPanel1.Controls)
    {
        if (item.GetType() == typeof(LeftPanelProcessorMiniView))
        {
            procprice = ((LeftPanelProcessorMiniView)item).LeftProc.Price;
        }
    }
    float hardprice = 0;
    foreach (UserControl item in flowLayoutPanel1.Controls)
    {
        if (item.GetType() == typeof(LeftPanelHardDisk))
        {
            hardprice = ((LeftPanelHardDisk)item).HardDisk.Price;
        }
    }
    float Videocardprice = 0;
    foreach (UserControl item in flowLayoutPanel1.Controls)
    {
        if (item.GetType() == typeof(LeftPanelVideocardView))
        {
            Videocardprice = ((LeftPanelVideocardView)item).videocard.Price;
        }
    }
```

Продовження таблиці 3.1

	`` float blockprice = 0; foreach (UserControl item in flowLayoutPanel1.Controls) { if (item.GetType() == typeof(LeftPanelBlockPitaniaView)) { blockprice = ((LeftPanelBlockPitaniaView)item).BlockPower.Price; } } AALLPrice += hardprice + procprice + motherprice + SSDprice + Ohladprice + Videocardprice + ramprice + Korpusprice + blockprice; ALLprice.Text = AALLPrice + " Грн"; ``` В даному коді зазначено етап перевірки обраних деталей, де по завершенню в текстовому полі «Загальна вартість» користувачу буде виведена сума всіх обраних комплектуючих

Після обраних комплектуючих користувач зможе отримати перед собою як список деталей, які були додані, так і їхню загальну вартість(рис 3.4).

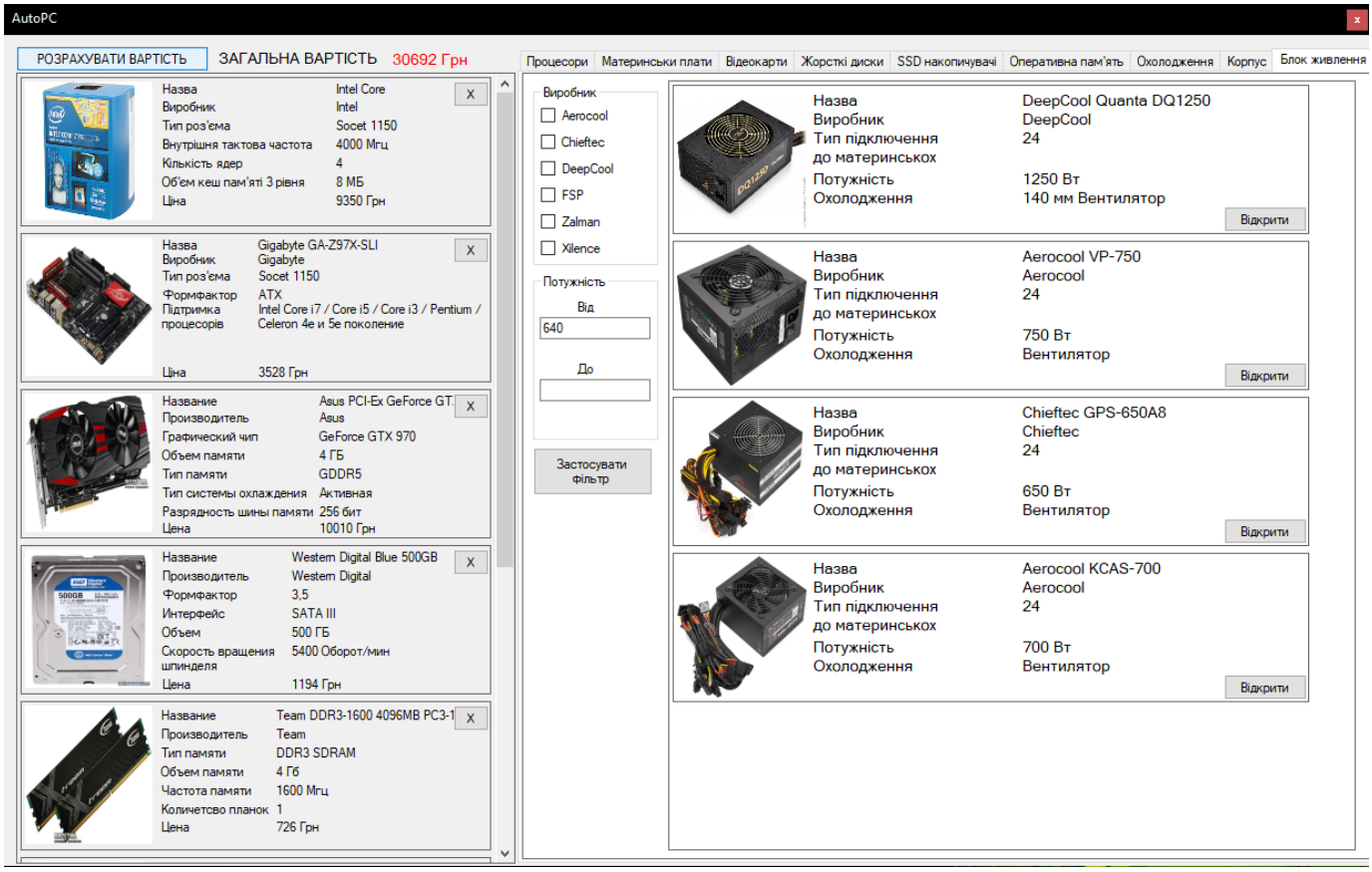


Рисунок 3.4 Фінальна сторінка

3.3 Тестування програмного продукту

Тестування програмного забезпечення – це процес технічного дослідження, призначений для виявлення інформації про якість продукту відносно контексту, в якому він має використовуватись. Техніка тестування також включає, як процес пошуку помилок або інших дефектів, так і випробування програмних складових із метою оцінки.

Програмний продукт буде перевірено на відповідність вимогам, які були сформовані на етапі проектування

Розглянемо на прикладі декілька тест кейсів, які зустрічаються кожному користувачу програмного продукту.

I. Тест-кейс 1:
   - Відкрити програму Diplom.exe;
   - В розділі «Процесори» відкрити будь-який з представлених та додати до бажаного;
   - Після додавання спробувати додати будь-який інший процесор та перевірити чи буде додано ще один;
   - Очікуваний результат – процесор буде замінено на новий;
   - Фактичний результат – процесор було замінено на новий.

II. Тест-кейс 2:
   - Відкрити програму Diplom.exe;
   - Обрати в кожному розділі по одній деталі із комплектуючих;
   - Після додавання спробувати видалити в списку бажаних будь-яку деталь;
   - Очікуваний результат – з’явиться помилка з проханням додати відсутню деталь.
   - Фактичний результат – програма виконає підрахунок доданих деталей без виконання перевірки на відсутню деталь.

3.4 Висновки

В даному розділі було обговорено питання роботи в середовищі розробки Visual Studio, та наведено приклад створення простого проекту з елементом підключення СКБД – SQLite.

Також у вигляді таблиці було наведені основні функції, які виконує програмний продукт та кінцевий результат, до якого може прийти користувач програми.

За результатами проведеного тестування можна зробити висновок, що під час останніх поправок в програмному продукті було залишено зміну, яка вплинула на перевірку всіх обраних деталей, з яких складається системний блок для подальшого підрахування вартості.

Критичних помилок, які порушують роботу програми або перешкоджають її роботі не виявлено, проте дана помилка обов’язково буде виправлена в наступних оновленнях.

ВИСНОВКИ

Результатом виконання дипломного проекту стала розробка програмного продукту для конфігурації персонального комп'ютеру. Програма надає можливість користувачу ознайомитися з представленими комплектуючими персонального комп'ютеру, їхні характеристики та взаємодії між іншими комплектуючими складової системного блоку. Було проведено дослідження аналогів програмного продукту, виявлено актуальність теми дипломного проекту, оскільки було знайдено лише 1 систему, а саме інтернет ресурс, який надав змогу виконати підбір комплектуючих, можливість ознайомлення з характеристиками деталі, а також формування системного блоку в цілому.

Під час створення дипломного проекту було освоєно нові технології взаємодії з API, покращені теоретичні та практичні знання протягом періоду створення програмного продукту

Програмний продукт розроблений об'єктно-орієнтованою мовою програмування C#, з використанням полегшеної реляційної системи керування базами даних SQLite. В якості середовища розробки використано програмне забезпечення Visual Studio 2019.

Програмний продукт, який було розроблено в дипломному проекті може покращуватись та оновлюватись в процесі підтримки програмного забезпечення.

ПЕРЕЛІК ПОСИЛАНЬ

1. Світовий ринок ПК [Електронний ресурс] Режим доступу до ресурсу: https://itc.ua/news/v-2020-godu-rynok-pk-vpervye-za-poslednee-desyatiletie-vyros-bolee-chem-na-10/
2. Роль комп'юрету в житті людини [Електронний ресурс] Режим доступу до ресурсу: https://computertechnology4656.blogspot.com/2019/03/blog-post_11.html
3. Сучасний ринок IT технологій [Електронний ресурс] Режим доступу до ресурсу: https://regulation.gov.ua/dialogue/it-i-telekom/14-rinok-rozrobki-programnogo-zabezpecenna
4. C# development with Visual Studio [Електронний ресурс] // Microsoft – Режим доступу до ресурсу: https://docs.microsoft.com/en-us/visualstudio/get-started/csharp/?view=vs-2019.
5. Visual Studio Download [Електронний ресурс] // Microsoft – Режим доступу до ресурсу: https://visualstudio.microsoft.com/
6. Visual C# 2008: базовый курс / [К. Уотсон, К. Нейгел, Я. Педерсен та ін.]. – М.: И.Д. Вильямс, 2009. – 1216 с.
7. Найпопулярніші операційні системи світу 2020 [Електронний ресурс] – Режим доступу до ресурсу: https://marketer.ua/ua/stats-operating-system-2020/
8. Документація по SQLite [Електронний ресурс] – Режим доступу до ресурсу: https://www.sqlite.org/docs.html
9. Windows Forms overview» [Електронний ресурс] – Режим доступу до ресурсу: https://docs.microsoft.com/enus/dotnet/framework/winforms/windows-forms-overview
10. C# 6.0 and the .NET 4.6 Framework (Язык программирования) / [Ендрю Троелсен і Філіп Джапіксе] 2015
11. Windows Form Tutorial [Електронний ресурс] – Режим доступу до ресурсу: https://metanit.com/sharp/windowsforms/1.1.php

12. Тестування програмного забезпечення [Електронний ресурс] – Режим доступу до ресурсу: https://qalearning.com.ua/theory/lectures/material/testing-intro/
13. SQLite on C# [Електронний ресурс] – Режим доступу до ресурсу: https://zetcode.com/csharp/sqlite/
14. Статистика стаціонарних та портативних ПК [Електронний ресурс] – Режим доступу до ресурсу: https://vawilon.ru/statistika-noutbukov/

ДОДАТОК А

```
using System;
using System.Collections.Generic;
using System.Drawing;
using System.Linq;
using System.Windows.Forms;
using Diplom.DatabaseClass;
using Diplom.Patterns;
using SQLite;

namespace Diplom
{
    public partial class Form1 : Form
    {
        private int locationx = 135;
        private int locationy = 0;

        private SQLiteConnection sqlcon;

        private UserControl _ohladControl;
        private UserControl _ramControl;
        private UserControl _ssdControl;
        private UserControl _harddiskControl;
        private UserControl _videocardControl;
        private UserControl _blockpitaniaControl;
        private UserControl _procControl;
        private UserControl _motherControl;

        public Form1()
        {
            InitializeComponent();
            sqlcon = new SQLiteConnection(@"Alda");
            SortoutProcessor("");

        }

        public void DeletedMiniview(UserControl control)
        {
            foreach (UserControl item in flowLayoutPanel1.Controls)
            {
                if (item.GetType() == control.GetType()) flowLayoutPanel1.Controls.Remove(item);
            }
            flowLayoutPanel1.Controls.Remove(control);
        }

        public void DeletedMotherview(UserControl control)
        {
            foreach (UserControl item in flowLayoutPanel1.Controls)
            {
                if (item.GetType() == control.GetType()) flowLayoutPanel1.Controls.Remove(item);
            }
            flowLayoutPanel1.Controls.Remove(control);
        }
        public void DeletedVideocardview(UserControl control)
        {
            foreach (UserControl item in flowLayoutPanel1.Controls)
            {
                if (item.GetType() == control.GetType()) flowLayoutPanel1.Controls.Remove(item);
            }
            flowLayoutPanel1.Controls.Remove(control);
        }
```

```
private void SortoutKorpus(string filter)
{
    SQLiteCommand korpusesCommand =
        sqlcon.CreateCommand(
            "select kp.* ,group_concat(fl.Name) as FName,pl.Name as PName from Korpus kp, ProizvodList pl, FormFactorList fl,KorpusFormFactor kf where pl.Id = kp.Proizvod and kf.IdFormFactor = fl.Id and kf.IdKorpus = kp.Id " + filter

    List<Korpus> korpuses = korpusesCommand.ExecuteQuery<Korpus>();
    locationy = 0;
    foreach (var korpuse in korpuses)
    {
        UserControl korpusesControl = new Korpusminiview(korpuse, AddToBascketKorpus, DeletedKorpus)
        {
            Location = new Point(locationx, locationy)
        };
        flowLayoutPanel10.Controls.Add(korpusesControl);
        locationy += 135;
    }
}

private void SortoutProcessor(string filter)
{

    List<Processor> Processors;
    SQLiteCommand processorsCommand;
    processorsCommand =
        sqlcon.CreateCommand("Select ProizvodList.Name as PLN,SocetList.Socet as SLN, Processor.* " +
                             "From ProizvodList,Processor,SocetList " +
                             "where Processor.ProizvoditelId = ProizvodList.Id and Processor.Socet = SocetList.Id" + filter
                             );
    Processors = processorsCommand.ExecuteQuery<Processor>();

    foreach (var processor in Processors)
    {

        _procControl = new Procminiview(processor, AddToBasketProc, DeletedMiniview)
        {
            Location = new Point(locationx, locationy)
        };
        flowLayoutPanel2.Controls.Add(_procControl);

    }

    processorsCommand.CommandText = "Select ProizvodList.Name as PLN,SocetList.Socet as SLN, Processor.* " +
                                    "From ProizvodList,Processor,SocetList " +
                                    "where Processor.ProizvoditelId = ProizvodList.Id and Processor.Socet = SocetList.Id";

}
```

```
private void SortoutMotherPlate(string filter)
{
    SQLiteCommand motherviewCommand;
    List<MotherPlate> MotherPlatess;
    motherviewCommand =
       sqlcon.CreateCommand(
           "select mp.*,group_concat(tm.Kolichestvo||' x '|| tl.Name) as SupMemory, sl.Socet as SName, fl.Name as FName, pl.Name as PName " +
           "from MotherPlate mp, TypeRamListMother tm , TypeRamList tl,  SocetList sl,  FormFactorList fl, ProizvodList pl " +
           "where pl.Id = mp.ProizvoditelId and sl.Id = mp.Socet and fl.Id = mp.Formfactor and tm.IdMother = mp.Id and tl.Id = tm.IdRam " + filter +
           "group by mp.Id ");
    MotherPlatess = motherviewCommand.ExecuteQuery<MotherPlate>();
    locationy = 0;
    foreach (var motherPlate in MotherPlatess)
    {
        _motherControl = new MotherPlateminiview(motherPlate, AddToBasketMother, DeletedMotherview)
        {
            Location = new Point(locationx, locationy)
        };
        flowLayoutPanel3.Controls.Add(_motherControl);
        locationy += 135;
    }
    motherviewCommand.CommandText =
        "select mp.*,group_concat(tm.Kolichestvo||' x '|| tl.Name) as SupMemory, sl.Socet as SName, fl.Name as FName, pl.Name as PName " +
        "from MotherPlate mp, TypeRamListMother tm , TypeRamList tl,  SocetList sl,  FormFactorList fl, ProizvodList pl " +
        "where pl.Id = mp.ProizvoditelId and sl.Id = mp.Socet and fl.Id = mp.Formfactor and tm.IdMother = mp.Id and tl.Id = tm.IdRam " +
        "group by mp.Id";
}

private void SortoutVideocard(string filter)
{
    SQLiteCommand videocardviewCommand;
    List<Videocard> videocards;
    videocardviewCommand =
        sqlcon.CreateCommand(
            "Select ProizvodList.Name as PName, vc.*,gl.Name as SName from Videocard vc ,ProizvodList,GraphProcesorsList gl where  vc.Proizvoditel = ProizvodList.Id and vc.SemiaGraphProc = gl.Id  " + filter);
    videocards = videocardviewCommand.ExecuteQuery<Videocard>();
    locationy = 0;
    foreach (var videocard in videocards)
    {
        _videocardControl = new Videocardminiview(videocard, AddToBascetVideocard, DeletedVideocardview)
        {
            Location = new Point(locationx, locationy)
        };
        flowLayoutPanel4.Controls.Add(_videocardControl);
        locationy += 135;
    }
    videocardviewCommand.CommandText =
        "Select ProizvodList.Name as PName, vc.*,gl.Name as SName from Videocard vc ,ProizvodList,GraphProcesorsList gl where  vc.Proizvoditel = ProizvodList.Id and gl.Id = vc.SemiaGraphProc";
}
```

```
private void SortoutOhlad(string filter)
{
    SQLiteCommand ohladCommand;
    List<Ohlajdenie> Ohlajdenies;
    ohladCommand =
        sqlcon.CreateCommand(
            "Select oh.*, pl.Name as PName, cl.TypeKuller as TName from Ohlajdenie oh,ProizvodList pl, CollerTypeList cl where cl.Id = oh.TypeKuller and pl.Id = oh.Proizvoditel " + filter);
    Ohlajdenies = ohladCommand.ExecuteQuery<Ohlajdenie>();
    locationy = 0;
    foreach (var ohlajdeny in Ohlajdenies)
    {
        _ohladControl = new Ohlajdenieminivjew(ohlajdeny, AddToBascetOhlad, DeletedOhlad)
        {
            Location = new Point(locationx, locationy)
        };
        flowLayoutPanel8.Controls.Add(_ohladControl);

    }
}

private void SortoutBlockpitania(string fitler)
{
    SQLiteCommand blockpitanizCommand;
    List<BlockPitania> BlockPitanias;
    blockpitanizCommand =
        sqlcon.CreateCommand(
            "Select bp.*, pl.Name as PName from BlockPitania bp, ProizvodList pl where pl.Id = bp.Proizvod" + fitler);
    BlockPitanias = blockpitanizCommand.ExecuteQuery<BlockPitania>();
    locationy = 0;
    foreach (var blockPitania in BlockPitanias)
    {
        _blockpitaniaControl = new Blockpitanieminiview(blockPitania, AddToBascetBlockPit, DeletedBlock)
        {
            Location = new Point(locationx, locationy)
        };
        flowLayoutPanel9.Controls.Add(_blockpitaniaControl);


    }
    blockpitanizCommand.CommandText =
          "Select bp.*, pl.Name as PName from BlockPitania bp, ProizvodList pl where pl.Id = bp.Proizvod";
}
```

```
        if (ProizvodProc.Count != 0)
        {
            Filter += " and Processor.ProizvoditelId in(" +
                                            string.Join(",", ProizvodProc.Select(n => n.ToString()).ToArray()) +
                                            ")";
        }
        if (SocetProc.Count != 0)
        {
            Filter += " and Processor.Socet in(" +
            string.Join(",", SocetProc.Select(n => n.ToString()).ToArray()) +
            ")";
        }
        if (CoreProc.Count != 0)
        {
            Filter += " and Processor.ManyCore in(" +
            string.Join(",", CoreProc.Select(n => n.ToString()).ToArray()) +
            ")";
        }
        SortoutProcessor(Filter);
    }

    private void tabPage1_Click(object sender, EventArgs e)
    {

    }

    private void button2_Click(object sender, EventArgs e)
    {
        string Filter = "";
        List<int> ProizvodMother = new List<int>();
        List<int> SocetMother = new List<int>();
        List<int> FormFactor = new List<int>();
        flowLayoutPanel3.Controls.Clear();

        if (checkBox19.Checked)
        {
            ProizvodMother.Add(13);
        }
        if (checkBox20.Checked)
        {
            ProizvodMother.Add(14);
        }
        if (checkBox21.Checked)
        {
            ProizvodMother.Add(15);
        }
        if (checkBox22.Checked)
        {
            ProizvodMother.Add(16);
        }
        if (checkBox23.Checked)
```

```
using System.Collections.Generic;
using System.Diagnostics.Eventing.Reader;
using System.Windows.Forms;
using Diplom.DatabaseClass;
using Diplom.Patterns;

namespace Diplom
{
    public class ItemAddCheckHelper
    {
        public void AllComlect(Control.ControlCollection item)
        {
            bool procbool = false;
            bool motherboll = false;
            bool holadbool = false;
            bool rambool = false;
            bool videocardbool = false;
            bool blockbool = false;
            bool korpusbool = false;
            bool memorybool = false;
            foreach (var userControl in item)
            {
                if (userControl.GetType() == typeof(LeftPanelProcessorMiniView))
                {
                    procbool = true;

                }
                if (userControl.GetType() == typeof(LeftPanelMotherView))
                {
                    motherboll = true;

                }
                if (userControl.GetType() == typeof(LeftPanelVideocardView))
                {
                    videocardbool = true;

                }
                if (userControl.GetType() == typeof(LeftPanelRamView))
                {
                    rambool = true;

                }
                if (userControl.GetType() == typeof(LeftPanelBlockPitaniaView))
                {
                    blockbool = true;

                }
                if (userControl.GetType() == typeof(LeftPanelKorpusview))
                {
                    korpusbool = true;

                }
                if (userControl.GetType() == typeof(LeftPanelOhladView))
                {
                    holadbool = true;

                }
                if ((userControl.GetType() == typeof(LeftPanelHardDisk)) && (item.GetType() != typeof(LeftPanelSSDView)))
                {
                    memorybool = true;
```

```
60                          memorybool = true;
61
62                      }
63                  }
64                  if (procbool == false)
65                  {
66                      MessageBox.Show("Ви не вибрали процесор", "Увага");
67                  }
68                  if (motherboll == false)
69                  {
70                      MessageBox.Show("Ви не вибрали материнську плату", "Увага");
71                  }
72                  if (videocardbool == false)
73                  {
74                      MessageBox.Show("Ви не вибрали відеокарту", "Увага");
75                  }
76                  if (rambool == false)
77                  {
78                      MessageBox.Show("Ви не вибрали оперативну пам'ять", "Увага");
79                  }
80                  if (blockbool == false)
81                  {
82                      MessageBox.Show("Ви не вибрали блок живлення", "Увага");
83                  }
84                  if (korpusbool == false)
85                  {
86                      MessageBox.Show("Ви не вибрали корпус", "Увага");
87                  }
88                  if (holadbool == false)
89                  {
90                      MessageBox.Show("Ви не вибрали охолодження", "Увага");
91                  }
92                  if (memorybool == false)
93                  {
94                      MessageBox.Show("Ви не вибрали накопичувач пам'яті", "Увага");
95                  }
96              }
97
98              public void RightComlect(Control.ControlCollection item)
99              {
100                 int procSocet = 0;
101                 foreach (var userControl in item)
102                 {
103
104                 }
105
106             }
107         }
108     }
```

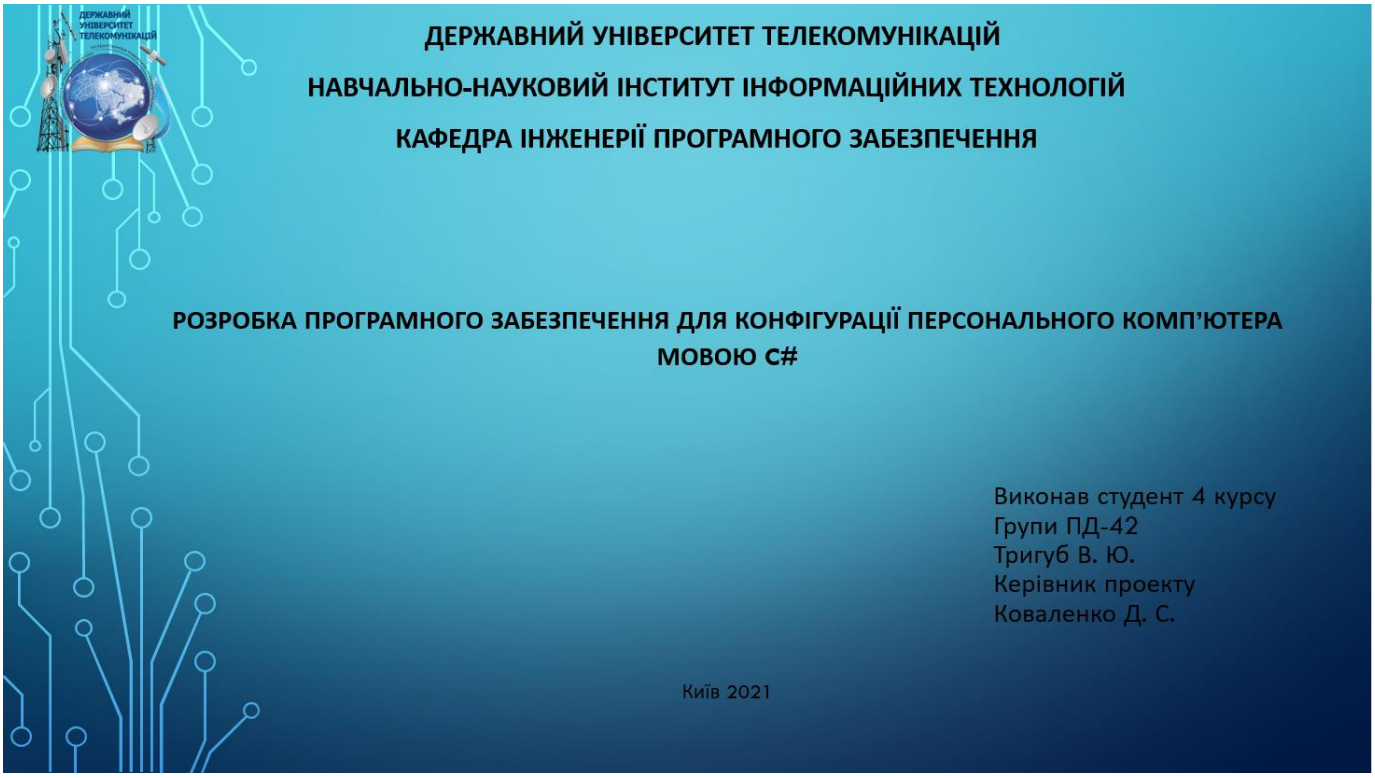
```
LeftPanelMotherView mother = null;
LeftPanelMotherView kp = null;
LeftPanelProcessorMiniView sd = null;
if (tabControl1.SelectedIndex == 0)
{
    foreach (UserControl item in flowLayoutPanel1.Controls)
    {
        if (item.GetType() == typeof(LeftPanelMotherView))
        {
            mother = (LeftPanelMotherView)item;
        }
    }
    if (mother == null)
    {
        Procminiview proc = null;
        foreach (UserControl item in flowLayoutPanel2.Controls)
        {
            if (item.GetType() == typeof(Procminiview))
            {
                proc = (Procminiview)item;
            }
        }
        if (proc == null)
        {
            SortoutProcessor("");
        }
    }
    else
    {
        checkBox3.Checked =
            checkBox4.Checked =
                checkBox5.Checked =
                    checkBox6.Checked =
                        checkBox7.Checked =
                            checkBox8.Checked = checkBox9.Checked = checkBox10.Checked = false;
        switch (mother.leftMother.Socet)
        {
            case 1:
                checkBox4.Checked = true;
                break;
            case 2:
                checkBox3.Checked = true;
                break;
            case 3:
                checkBox5.Checked = true;
                break;
            case 4:
                checkBox6.Checked = true;
                break;
            case 5:
                checkBox7.Checked = true;
                break;
            case 6:
                checkBox8.Checked = true;
                break;
            case 7:
                checkBox9.Checked = true;
                break;
            case 9:
                checkBox10.Checked = true;
                break;
```

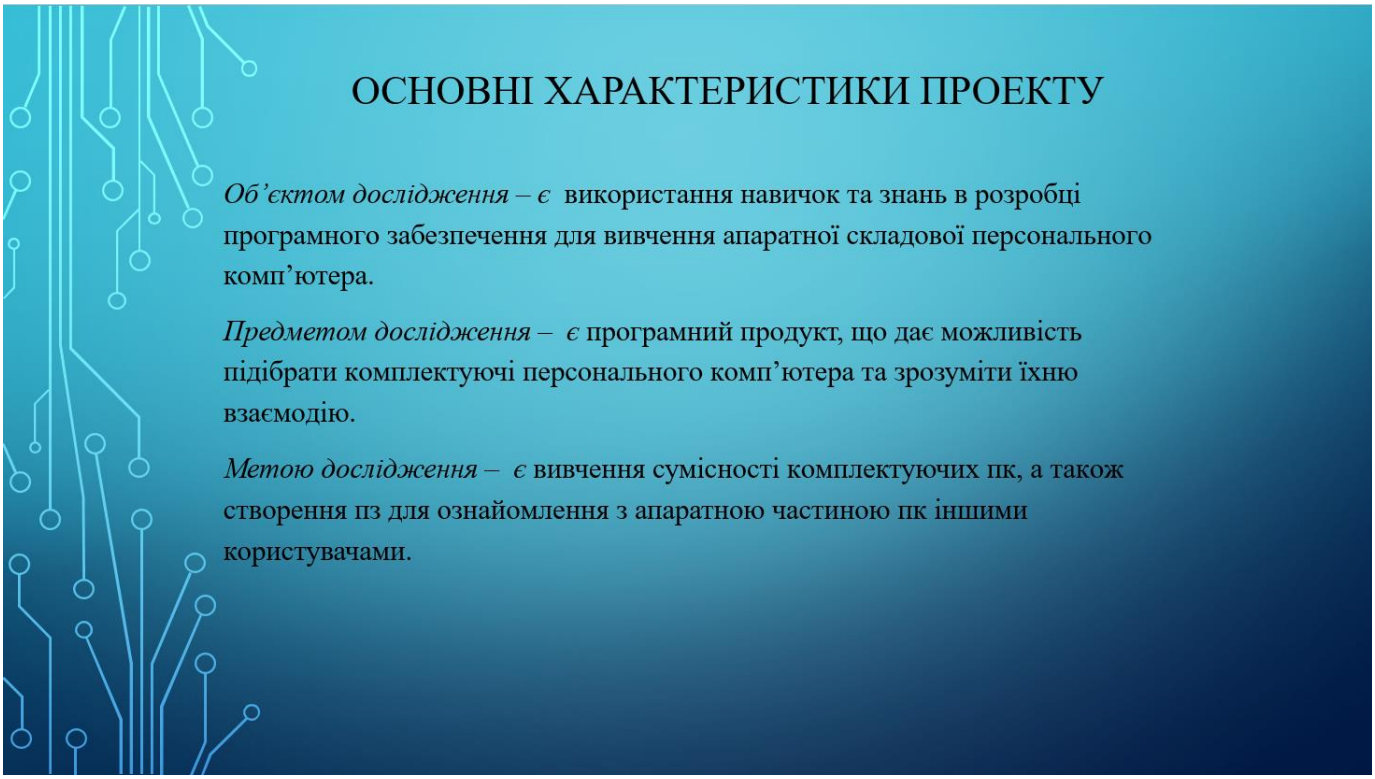
```
    LeftPanelHardDisk hr = null;
    LeftPanelRamView ram = null;
    LeftPanelVideocardView video = null;
    if (tabControl1.SelectedIndex == 8)
    {

        foreach (UserControl item in flowLayoutPanel1.Controls)
        {
            if (item.GetType() == typeof(LeftPanelProcessorMiniView))
            {
                sd = (LeftPanelProcessorMiniView)item;
            }
        }
        foreach (UserControl item in flowLayoutPanel1.Controls)
        {
            if (item.GetType() == typeof(LeftPanelHardDisk))
            {
                hr = (LeftPanelHardDisk)item;
            }
        }
        foreach (UserControl item in flowLayoutPanel1.Controls)
        {
            if (item.GetType() == typeof(LeftPanelRamView))
            {
                ram = (LeftPanelRamView)item;
            }
        }
        foreach (UserControl item in flowLayoutPanel1.Controls)
        {
            if (item.GetType() == typeof(LeftPanelVideocardView))
            {
                video = (LeftPanelVideocardView)item;
            }
        }
        if (video == null) return;
        if (ram == null) return;
        if (hr == null) return;
        if (sd == null) return;
        int mintpd = 0;
        mintpd += (int)hr.HardDisk.MaxTpd + (int)sd.LeftProc.TDP + (int)ram.RAM.Pitanie + video.videocard.MinTPD + 50;
        textBox1.Text = mintpd.ToString();
        button8_Click(null, null);
    }

}
```

ДОДАТОК Б

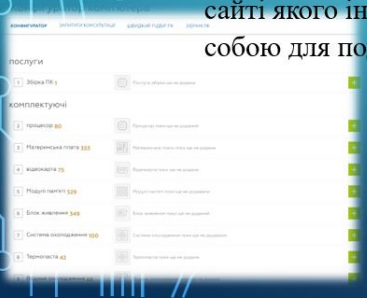

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ
РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ КОНФІГУРАЦІЇ ПЕРСОНАЛЬНОГО КОМП'ЮТЕРА МОВОЮ C#
Виконав студент 4 курсу
Групи ПД-42
Тригуб В. Ю.
Керівник проекту
Коваленко Д. С.
Київ 2021

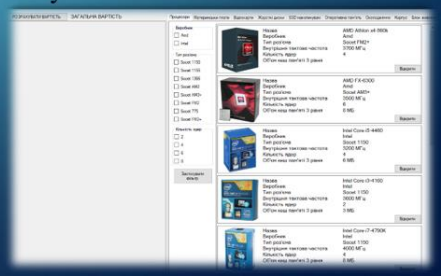

ОСНОВНІ ХАРАКТЕРИСТИКИ ПРОЕКТУ
Об'єктом дослідження – є використання навичок та знань в розробці програмного забезпечення для вивчення апаратної складової персонального комп'ютера.
Предметом дослідження – є програмний продукт, що дає можливість підібрати комплектуючі персонального комп'ютера та зрозуміти їхню взаємодію.
Метою дослідження – є вивчення сумісності комплектуючих пк, а також створення пз для ознайомлення з апаратною частиною пк іншими користувачами.

ОГЛЯД АНАЛОГІВ

На етапі проектування програмного продукту було проведено аналіз вже існуючого програмного забезпечення, яке схоже за функціоналом теми дипломного проекту, тобто здійснено пошук аналогів ПЗ. Існуючі аналоги представлені виключно в вигляді інтернет ресурсів для конфігурації ПК та є в вільному доступі користувачам, проте не всі інтернет ресурси здійснюють підбір комплектуючих, які будуть взаємодіяти між собою.

Більшість аналогів використовують підбір вже готових комплектацій та не надають змоги користувачу самостійно підібрати комплектуючі, які його цікавлять, проте все ж таки існує інтернет магазин під назвою «Telemart», на сайті якого інтегрована система пошуку та взаємодії комплектуючих між собою для подальшого придбання.





Порівняльна таблиця аналогу та розроблюваного проекту

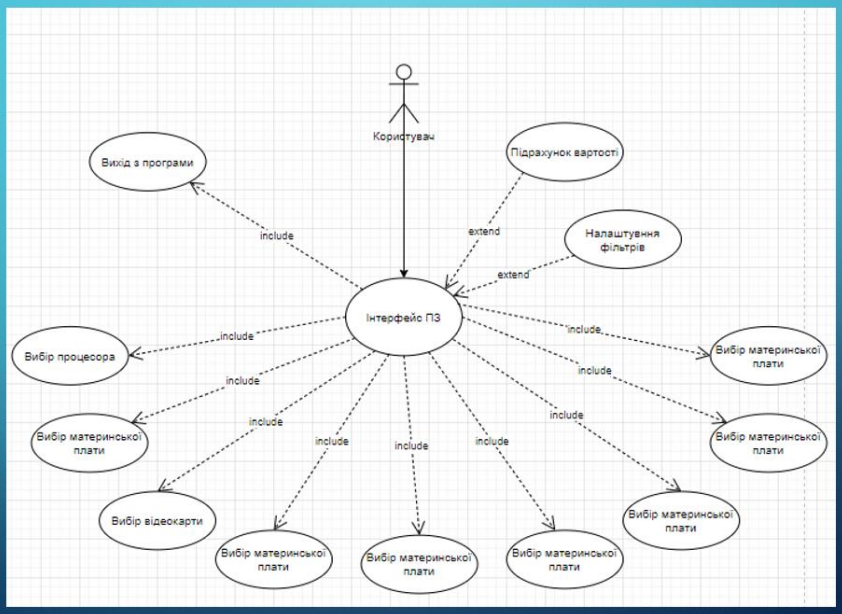
Назва	Фільтри пошуку	Кросплатформеність	Широкий асортимент	Можливість придбання комплектуючих	Можливість взаємодії локально
Telemart	Так	Так	Так	Так	Ні
AutoPC	Так	Ні	Ні	Ні	Так

ЗАСОБИ РЕАЛІЗАЦІЇ ПРОГРАМНОГО ПРОДУКТУ

При створенні дипломного проекту було використано наступні програмні засоби реалізації:

- Середовище розробки «Microsoft Visual Studio»;
- Мова програмування «C#»;
- Реляційня СУБД SQLite.

ДІАГРАМА ВИКОРИСТАННЯ



ВИСНОВКИ

Результатом дипломного проекту є програмний продукту , який дає змогу користувачу виконати конфігурацію програмного комп’ютеру та дізнатися щось нове в напрямку комплектуючих персонального комп’ютеру.

Було проведено дослідження аналогів програмного продукту, виявлено актуальність теми дипломного проекту, оскільки було знайдено лише 1 систему, а саме інтернет ресурс, який надав змогу виконати підбір комплектуючих, можливість ознайомлення з характеристиками деталі, а також формування системного блоку в цілому.

Програмний продукт, який було розроблено в дипломному проекті може покращуватись та оновлюватись в процесі підтримки програмного забезпечення.