Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Кафедра програмної інженерії

КУРСОВА РОБОТА

ПОЯСНЮВАЛЬНА ЗАПИСКА

З дисципліни “Об’єктно - орієнтоване програмування”

Керівник , Професор кафедри програмної інженерії Бондарєв В.М.

Студент гр.ПЗПІ-22-1 Токарчук М.В

Комісія : Проф. Бондарєв В.М.,

Ст.викл. Черепанова Ю.Ю.,

Ст.викл. Ляпота В.М.

Харків 2023

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

РАДІОЕЛЕКТРОНІКИ

Кафедра *програмної інженерії*

Рівень вищої освіти *перший (бакалаврський)*

Дисципліна  *Об’єктно - орієнтоване програмування*

Спеціальність *121 Інженерія програмного забезпечення*

Освітня програма : *Програмна інженерія*

Курс 1 . Група *ПЗПІ-22-1.* Семестр  *2 .*

**ЗАВДАННЯ**

**На курсовий проект студента**

Токарчука Максима Валерійовича

1. Тема проекту : Біржа товарів
2. Термін здачі студентом закінченого проекту: ***“23”*** *-* ***червня - 2023р.***
3. Вхідні дані до проекту:

*Специфікація програми, методичні вказівки до*

*виконання курсової роботи*  .

1. Зміст розрахунково-пояснювальної записки:

*Вступ, специфікація програми, проектування програми,*

*Інструкція користувача, висновки .*

КАЛЕНДАРНИЙ ПЛАН

|  |  |  |
| --- | --- | --- |
| *№* | *Назва етапу* | *Термін виконання* |
| 1 | Видача теми, узгодження і затвердження  теми | 13.02.2023 - 14.03.2023 р. |
| 2 | Формулювання вимог до програми | 07.06.2023 - 08.06.2023 р. |
| 3 | Розробка підсистеми зберігання та пошуку даних. | 08.06.2023 - 09.06.2023 р. |
| 4 | Розробка функцій ….. | 09.06.2023 - 18.06.2023 р. |
| 5 | Розробка функцій зберігання та  завантаження даних | 18.06.2023 - 19.06.2023 р. |
| 6 | Тестування і доопрацювання розробленої  програмної системи. | 19.06.2023 – 20.06.2023 р. |
| 7 | Оформлення пояснювальної записки,  додатків, графічного матеріалу | 20.06.2023 – 22.06.2023 р. |
| 8 | Захист | 23.06.2023 – 27.06.2023 р. |

Студент Токарчук Максим Валерійович

Керівник Бондарєв Володимир Михайлович

«23» червня 2023 р.

РЕФЕРАТ

Пояснювальна записка до курсової роботи: 35 с., 16 рис., 3 додатки, 5 джерел.

ПОКУПЕЦЬ, МАГАЗИН, ЗВІТ, ООП, .NET, МОВА C#

Метою роботи є розробка програми «Біржа товарів», яка буде опрацьовувати заявки від клієнтів, які надходять до магазину.

В результаті отримана програма, що дозволяє зберігати список магазинів, характеристики кожного магазину, такі як: назва, спеціалізація, адреса, телефон, час роботи, вид власності. Є можливість утворювати нові списки магазинів, додавати, видаляти та редагувати магазини. Також, є функція формування звіту у Microsoft Excel.

В результаті отримана програма буде мати:

* База продавців: найменування товару, обсяг партії при оптовому продажі, ціна, умови продажу-відвантаження, форма оплати, контактна адреса або телефон, примітка (наприклад, «посередників прошу не турбуватися»).
* База покупців: найменування товару, обсяг покупки, прийнятна ціна та форма оплати, контактна адреса або телефон, примітка.
* Пошук і реєстрація варіантів з того чи іншого боку;
* Формування оголошень для друку, видалення в архів після купівлі-продажу (можливо, один з клієнтів залишається незадоволеним), повне видалення при відмові від послуг.

В процесі розробки використано середовища Microsoft Visual Studio 2022, фреймворк Windows Forms, платформи .NET 6.0, мова програмування C#

ЗМІСТ

[ВСТУП 6](#_Toc138166057)

[РОЗДІЛ 1 ТЕОРЕТИЧНІ ВІДОМОСТІ 7](#_Toc138166058)

[1.1 Дослідження поняття 7](#_Toc138166059)

[1.1.1 Товарна біржа 7](#_Toc138166060)

[1.1.2 Мова програмування 8](#_Toc138166061)

[1.2 Аналіз предметної області 15](#_Toc138166062)

[1.2.1 Принципи розробки програмного забезпечення 15](#_Toc138166063)

[1.2.2 Особливості інсталяції програмного забезпечення 16](#_Toc138166064)

[1.3 Коротка характеристика об'єкту управління 17](#_Toc138166065)

[1.4 Опис предметної області 18](#_Toc138166066)

[1.6 Аналоги біржі товарів 20](#_Toc138166067)

[1.6 Висновок до розділу 1 22](#_Toc138166068)

[РОЗДІЛ 2 ЗАСОБИ РОЗРОБКИ 23](#_Toc138166069)

[2.1 Вибір мови програмування та середовища розробки 23](#_Toc138166070)

[2.1.1 Мова програмування C# 23](#_Toc138166071)

[2.1.2 Microsoft Visual Studio 25](#_Toc138166072)

[2.2 Принципи створення дизайну програмного забезпечення 27](#_Toc138166073)

[2.3 UML 28](#_Toc138166074)

[2.4 Висновок до розділу 2 29](#_Toc138166075)

[РОЗДІЛ 3 РОЗРОБКА ДОДАТКУ 30](#_Toc138166076)

[3.1 Реалізація додатку 30](#_Toc138166077)

[3.2 Тестування додатку 32](#_Toc138166078)

[3.3 Висновок до розділу 3 32](#_Toc138166079)

[ВИСНОВОК 33](#_Toc138166080)

[СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 34](#_Toc138166081)

[ДОДАТОК 35](#_Toc138166082)

[ДОДАТОК А 36](#_Toc138166083)

[ДОДАТОК Б 37](#_Toc138166084)

[ДОДАТОК В 38](#_Toc138166085)

ВСТУП

Сьогодні використання комп'ютера для купівлі та продажу товарів уже не є буденним явищем. Дуже зручно мати можливість замовити все, що хочеш, коли і де хочеш. Позбавлення від старих речей або створення власного малого бізнесу - теж дуже важлива частина життя, але без спеціалізованої платформи це може бути важко.

Розроблений додаток вирішує ці проблеми. Користувачі можуть виставити на продаж те, що вони хочуть, або купити все, що є в каталозі. Авторизовані користувачі мають повний доступ до його функцій. Доступ до облікового запису диспетчера здійснюється за допомогою пароля.

Крім розроблення описаних вище програмних систем, цілями цієї курсової роботи є вивчення мови програмування C#, освоєння об'єктно-орієнтованого програмування та вдосконалення навичок роботи з Windows Forms.

Цілі роботи:

* Розробити систему для читання та запису даних;
* Створення логіки системи;
* Робота з користувачами;
* Створення логіки для додавання, вилучення та видалення інформації;
* Створення стабільності додатка.

РОЗДІЛ 1 ТЕОРЕТИЧНІ ВІДОМОСТІ

* 1. Дослідження поняття

1.1.1 Товарна біржа

Товарна біржа - це організація учасників ринку - продавців, покупців і посередників, які встановлюють ціни, обсяги торгівлі, укладають угоди, сприяють обігу капіталу, визначають ціни, аналізують попит і пропозицію та створюють найсприятливіші умови для інших торговельних операцій. Торгівля на товарних біржах ведеться не у звичайний спосіб, а на основі зразків, стандартів і документів відповідного формату, в яких зазначено ціни, товарні кількості, номенклатура, способи поставки, умови тощо.

Особливості товарної біржі

Товарні біржі є юридичними особами і діють на засадах економічної незалежності та автономії. Вона має власні банківські рахунки (валютний, розрахунковий та інші), самостійне майно, баланс і печатку зі своїм найменуванням.

Біржі не працюють з метою отримання прибутку і не є комерційними посередниками.

Типи товарних бірж

Існує дві класифікації товарних бірж:

1. Залежно від кількості країн, що беруть участь у торгівлі, біржі поділяють на дві категорії:

* Міжнародні біржі (понад 90% світової біржової торгівлі зосереджено в США, Японії та Великій Британії);
* Національні біржі (наприклад, онлайн-торгівля тільки на українському ринку).

2. За видами товарів:

* Універсальні - продаються промислові та сільськогосподарські товари;
* Спеціалізовані - спеціалізуються на продажу певних видів товарів (наприклад, біржі какао, цукру, кави).

Члени товарних бірж

Членами товарних бірж можуть бути як юридичні, так і фізичні особи, які займаються комерційною та виробничою діяльністю:

* Приватні громадяни;
* Посередники (наприклад, брокерські фірми, торгові компанії);
* Виробничі компанії;
* Банківські установи;
* Інвестиційні компанії.

Переваги торгівлі на біржі

Товарні біржі пропонують учасникам такі переваги:

* Ціни встановлюються на основі аналізу попиту та пропозиції;
* Торгові партнери обирають один одного на власний розсуд;
* Торгівля ведеться за єдиними правилами, встановленими конкретною біржею, і за певним алгоритмом;
* Існує встановлена форма контракту для спрощення торгівлі.
  + 1. Мова програмування

Мова програмування - це набір інструкцій, які використовують розробники для створення програмного забезпечення, за допомогою якого користувачі взаємодіють із комп'ютером.

Мови програмування бувають різних видів. Деякі з них складні у використанні. Деякі дають більше можливостей для контролю над поведінкою комп'ютера. Деякі програми кращі за інші.

Але в основі своїй усі мови програмування мають спільні риси, які полегшують створення програмного забезпечення.

Хоча мови програмування принципово не змінилися за десятиліття, вони залишаються одним з основних інструментів для розробників та інженерів-програмістів. Тому дуже важливо розуміти, як працюють мови програмування, основні відмінності між ними та яка мова програмування підходить для конкретного випадку використання.

Робота мови програмування

Мови програмування дають змогу розробникам писати читабельний код, який потім перетворюється у форму, доступну для виконання комп'ютером. Цей процес перетворення зазвичай виконується так званими компіляторами.

Однак у деяких мовах, які називаються інтерпретованими, код автоматично компілюється і виконується на льоту, тож програмістові не потрібно компілювати код у явному вигляді.

Мови програмування також містять синтаксис і граматику, які визначають, як розробники мають структурувати свій код. Наприклад, деякі мови програмування використовують круглі дужки або перенесення рядків для позначення певних функцій або підпрограм у файлах, що містять великий обсяг коду.

Це може бути використано розробниками для вказівки комп'ютеру виконати певні дії за допомогою спеціальних термінів, таких як "if", "for" і "import".

Переваги мови програмування

З технічної точки зору мови програмування - не єдиний спосіб взаємодії з комп'ютером. Можна написати так званий машинний код, який може бути виконаний безпосередньо процесором комп'ютера.

Однак основна проблема машинного коду полягає в тому, що його дуже важко читати і ще важче писати.

Мови програмування розв'язують цю проблему, дозволяючи розробникам писати інструкції для комп'ютера за допомогою коду, який легко читати та легко змінювати.

Щоб зрозуміти, що це означає на практиці, розглянемо такий код(рис….), написаний мовою C.

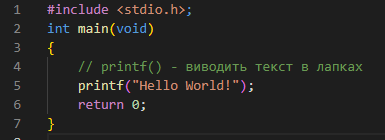


Рис. 1.1 – «Hello World!»

Будь-який програміст, який розуміє мову C, може прочитати цей код, що являє собою просту програму, яка виводить рядок "Hello World!" у командному рядку.

Код також можна легко переписати, якщо ви захочете змінити поведінку програми. Наприклад, якщо ви хочете вивести на екран рядок "Hi World!" замість "Hello World!", ви можете змінити програму таким чином. Ви зміните свою програму таким чином:

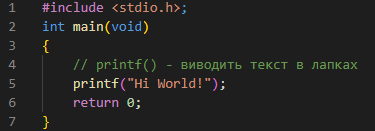


Рис. 1.2 – «Hi World!»

Другою важливою перевагою мов програмування є те, що вони полегшують написання програмного забезпечення, яке працює на різних платформах з мінімальними змінами або взагалі без них.

Це важливо, оскільки машинна мова зазвичай може працювати лише на певній архітектурі процесора комп'ютера, для якого вона призначена.

Однак, коли ви пишете програму мовою програмування, ви повідомляєте компілятору, на якому комп'ютері ви хочете її запустити, і компілятор перекладає її в машинний код для відповідного комп'ютера.

Таким чином, мови програмування дають змогу розробникам написати код один раз і створити програму, яка працюватиме де завгодно.

Недоліки мов програмування

Кожна мова програмування має свої переваги та недоліки, але загалом існує лише два реальних недоліки під час використання мов програмування:

* Продуктивність: машинний код, створюваний мовами програмування та компіляторами, не завжди оптимізований для найшвидшої роботи. Загалом, переваги продуктивності при написанні машинного коду вручну значно переважують складність такого підходу. Однак у ситуаціях, що вимагають надвисокої продуктивності та високої ефективності, мови програмування не завжди є найкращим способом написання програмного забезпечення;
* Контроль: мови програмування зазвичай накладають обмеження на рівень контролю розробника над поведінкою комп'ютера. Контроль: мови програмування накладають обмеження на рівень контролю розробника над поведінкою комп'ютера. Але навіть тут незначні переваги, одержувані під час генерації машинного коду вручну, значно переважують складність написання машинного коду вручну.

Незважаючи на ці недоліки, більшість виробленого сьогодні програмного забезпечення написано мовами програмування.

Популярні мови програмування



Рис. 1.3 – Індекс мов програмування

Розглянемо декілька мов програмування.

Python

Python зародився в 1980-х роках як мова сценаріїв для допомоги в розв'язанні завдань системного адміністрування, але зараз став популярною мовою загального призначення, яку можна використовувати для створення всіляких застосунків на всіх основних платформах.

Основна привабливість Python полягає в тому, що, будучи інтерпретованою мовою, вона не вимагає від розробників окремої компіляції коду.

Вони можуть просто написати код Python і запустити його безпосередньо.

C

Мова Сі, розроблена 1972 року, є однією з найстаріших мов програмування і широко використовується досі.

Під час компіляції коду мовою Сі виходять програми, що забезпечують відносно високу продуктивність. Однак мова Сі не має вбудованих функцій, які роблять програмування зручнішим, таких як збирання сміття та обробка винятків.

C++

Створений 1985 року як розширення мови C, C++ ставить на чільне місце ефективність і продуктивність порівняно зі своєю батьківською мовою.

Java

Оскільки код Java відносно "багатослівний", для реалізації Java-додатка зазвичай потрібно більше рядків коду, ніж для інших мов.

Проте Java вже давно є однією з найпопулярніших мов завдяки своїй простоті та легкості, з якою Java-додатки можна запускати на різних операційних системах із мінімальними змінами коду.

C#

C# - це дещо зручніша альтернатива мові C.

C# пропонує такі можливості, як вбудоване збирання сміття, і, на відміну від C, є об'єктно-орієнтованим, що полегшує структурування коду.

Основними недоліками C# є його менша переносимість і загальна продуктивність порівняно з C, але це не завадило C# довгий час залишатися однією з найпопулярніших мов.

Visual Basic

Мова Visual Basic настільки проста у вивченні, що її стали називати мовою для несерйозних програмістів або навіть "слабаків", цитуючи одне повідомлення на форумі про мову, написане кілька десятиліть тому.

Однак, незважаючи на те, що код Visual Basic не дуже ефективний і дає мало контролю над комп'ютером, Visual Basic, як і раніше, широко використовується.

JavaScript

Спочатку розроблений для створення динамічних веб-сайтів, JavaScript залишається однією з основних мов програмування для Інтернету.

Сьогодні він також використовується як мова серверної частини за допомогою таких фреймворків, як Node.js.

1.2 Аналіз предметної області

1.2.1 Принципи розробки програмного забезпечення

Під час розроблення програмного забезпечення використовують різні принципи, спрямовані на досягнення високої якості, ефективності та надійності програмних продуктів. Основними принципами розроблення програмного забезпечення є:

* модульність: декомпозиція програмного продукту на окремі модулі або компоненти, які можна розробляти, тестувати та підтримувати незалежно один від одного. Це спрощує розробку, розширення та модифікацію програмного коду і полегшує виявлення та виправлення помилок.
* розширюваність: забезпечення того, що програмний продукт можна легко модифікувати та розширити для впровадження нової функціональності або внесення змін до наявної функціональності. Це дає змогу продукту реагувати на мінливі вимоги та потреби користувачів.
* можливість повторного використання: економія часу і зусиль на розробку завдяки використанню готових компонентів, модулів, бібліотек тощо. Це дає змогу створювати більш стабільні та надійні програмні продукти, а також полегшує їхній супровід і підтримку.
* тестування Систематичне тестування програмного коду з метою виявлення помилок, уразливостей, некоректної роботи тощо. Це забезпечує якість і надійність програмного продукту, а також його стабільність під час змін і вдосконалень.
* контроль версій: використання систем контролю версій для відстеження та управління змінами в програмному коді, спільної роботи між командами розробників і відновлення попередніх версій. Це може полегшити співпрацю, підвищити безпеку коду та забезпечити управління змінами.
* чистий код: Створення програмного коду, який зрозумілий, простий для розуміння і супроводу. Це охоплює використання чітких імен змінних і функцій, правильне форматування коду, усунення непотрібного дублювання та підтримання загальних стандартів програмування.

Ці принципи допомагають забезпечити успішну розробку програмного забезпечення, знизити витрати на супровід і підтримку та підвищити задоволеність користувачів програмним продуктом.

1.2.2 Особливості інсталяції програмного забезпечення

Існує кілька особливостей встановлення програмного забезпечення, які варто розглянути:

* Вибір пакета інсталяції: існують різні типи пакетів інсталяції, включно з програмами інсталяції, менеджерами пакетів і веб-інсталяторами. Під час вибору відповідного пакета важливо враховувати потреби проєкту, платформу, на якій він працюватиме, і вимоги до процесу встановлення.
* Вимоги до попереднього встановлення Деякі програми можуть вимагати наявності певного програмного забезпечення або конфігураційного середовища в системі користувача до початку встановлення. Важливо передбачити ці вимоги і надати користувачеві відповідні інструкції з налаштування його системи до початку встановлення.
* Параметри конфігурації: під час інсталяції може виникнути необхідність у конфігурації програмного забезпечення для виконання певних функцій системи або вимог користувача. Розробники повинні забезпечити доступність необхідної конфігурації та інтерфейсу для її зміни.
* Ліцензування та активація: деякі додатки вимагають ліцензію та активацію для нормального функціонування. Розробники повинні забезпечити механізм управління ліцензіями та гарантувати, що процес активації буде зручним і зрозумілим для користувачів.
* Розміщення файлів: важливо, щоб усі необхідні файли програмного забезпечення були розміщені в правильній структурі каталогів під час встановлення. Це включає в себе.
* може включати розміщення файлів у відповідних каталогах, створення посилань на виконувані файли та налаштування середовища.
* Призначений для користувача інтерфейс: якщо в процесі встановлення потрібна взаємодія з користувачем, важливо забезпечити чіткий та інтуїтивно зрозумілий інтерфейс. Він може містити введення параметрів, вибір компонентів для встановлення і надання інформації про хід встановлення.
* Валідація і тестування Перед випуском програмного забезпечення важливо, щоб процес встановлення був належним чином перевірений і протестований. Це допоможе виявити й усунути будь-які помилки або проблеми, які можуть виникнути в процесі встановлення.

Загалом, під час розроблення програмного забезпечення процес інсталяції має бути ретельно спланований і реалізований, щоб забезпечити його надійність, зручність і відповідність вимогам користувача.

1.3 Коротка характеристика об'єкту управління

Товарні біржі - це контрольовані організації, які надають платформу для торгівлі різними товарами. Товарні біржі об'єднують покупців і продавців для обміну товарами за встановленими цінами. Товарні біржі відіграють ключову роль у формуванні цін на товари, забезпеченні прозорості ринку та ефективності торгівлі учасників. Вони можуть укладати угоди з широким спектром товарів, включно із сировиною, продуктами харчування та фінансовими інструментами. Товарні біржі потребують ефективного управління та регулювання для забезпечення справедливої, стабільної та безпечної торгівлі для всіх учасників.

Товарні біржі - це складні системи, що об'єднують різні групи учасників, включаючи виробників, посередників, інвесторів і трейдерів. Їхні основні функції полягають в управлінні процесом торгівлі, забезпеченні ліквідності та створенні сприятливих умов для торгівлі. Біржі стежать за дотриманням правил торгівлі, створюють механізми ціноутворення, відстежують рух товарів і надають інформаційну підтримку учасникам ринку. Для товарних бірж характерно те, що торгівля завжди йде динамічно і швидко, а також високий ризик і потенційна схильність до впливу зовнішніх факторів на ціни товарів. Ефективна робота товарних бірж вимагає компетентності у фінансових аспектах, регулюванні ринку, технічних рішеннях і взаємодії з учасниками ринку.

1.4 Опис предметної області

Предметна область програмного забезпечення "Товарна біржа" - це система керування базою даних продавців і покупців. У базі даних продавця зберігаються такі дані: найменування товару, кількість партії для оптових продажів, ціна, умови продажу і відвантаження, форма оплати, контактна адреса або телефон, а також примітки, які можуть містити додаткову інформацію, наприклад, про посередників.

У базі даних клієнтів зберігається назва товару, кількість закупівель, допустима ціна і форма оплати, контактна адреса або телефон, а також примітки.

Вона також має функції пошуку для обох сторін (продавця і покупця), реєстрації варіантів, створення друкованих оголошень, видалення записів, за якими вже було здійснено угоди (наприклад, після продажу або купівлі), і повного видалення при відмові від послуг. Також можливе архівування записів у разі незадоволеності клієнта.

Таким чином, програмне забезпечення "Товарна біржа" забезпечує ефективну організацію та опрацювання даних продавців і покупців, пошук і обмін між ними, а також створення оголошень і управління угодами, з відповідними функціями видалення та архівування.

Розділити проект, на два окремі проекти не є раціональним рішенням, оскільки функції продавця та покупця перетинаються. Для забезпечення належної перевірки програми на етапі програмування розумним буде створити декілька наборів тестових даних, щоб переконатись у правильності роботи програми.

На основі цього була створена структура на рис. 1.4.



Рис. 1.4 – Схема роботи додатку

Опис об’єктної моделі

На товарній біржі є наявні товари.

Користувач додає нові товари та оформляє замовлення.

Про товар відомо: ідентифікатор, назва, одиниця вимірювання, роздрібна ціна, оптова ціна, мінімальне замовлення для опту, на складі, продавець.

Про користувача відомо: ідентифікатор, ім'я, пароль, номер телефону, електронна пошта.

Про замовлення відомо: номер замовлення, покупець, продавець, товар, кількість, дата та час замовлення, адреса доставки, спосіб оплати.

Товар містить лише ідентифікатор продавця, а не тип даних "Користувач". Замовлення містить лише ідентифікатори товару, продавця та покупця, а не типи даних "Товар" та "Користувач" відповідно.

До простих типів не відносяться:

База товарів, товар, список користувачів, користувач, база замовлень, замовлення.

З них не є колекціями і не містять колекцій:

Товар, користувач, замовлення.

База товарів - це колекція товарів.

База замовлень - це колекція замовлень.

Список користувачів - це колекція користувачів.

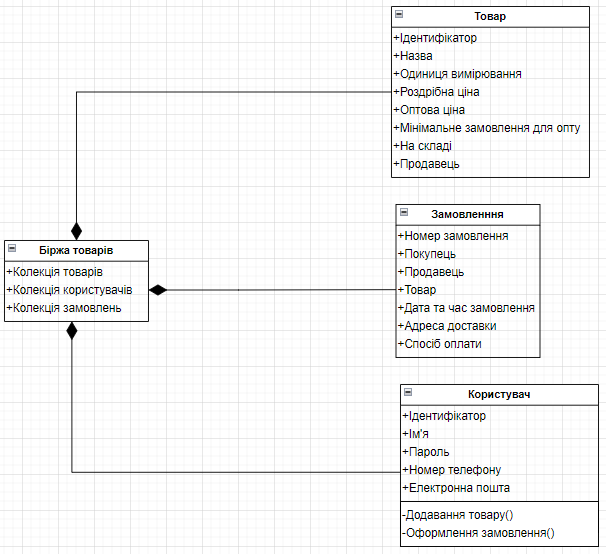


Рис. 1.5 – Об’єктна модель

1.6 Аналоги біржі товарів

Розглянемо декілька аналогів біржі товарів:

1. Аmazon seller



Рис. 1.6 – Логотип Amazon

Amazon Seller - це додаток, розроблений компанією Amazon для індивідуальних підприємців для продажу товарів на платформі Amazon. Додаток дає змогу продавцям створювати списки товарів, керувати запасами, встановлювати ціни, відстежувати замовлення, зв'язуватися з покупцями та багато іншого.

Використовуючи потужну інфраструктуру і ресурси Amazon, Amazon Seller надає продавцям доступ до мільйонів покупців по всьому світу. Додаток надає зручний і простий інтерфейс для управління бізнесом, що продає товари на Amazon.

Деякі з функцій Amazon Seller включають:

* Створення продуктових списків: Продавці можуть створювати докладні описи продуктів, додавати фотографії та інші важливі деталі.
* Управління запасами: Продавці можуть відстежувати кількість товарів у наявності, отримувати повідомлення про запаси та керувати перезавантаженням.
* Обробка замовлень: Продавці можуть відстежувати замовлення, надсилати повідомлення покупцям, вирішувати суперечки та здійснювати повернення товарів.
* Аналітика: Додаток надає різноманітні аналітичні звіти та статистику щодо продажів, що допомагає продавцям зрозуміти попит на їхні товари та приймати виважені рішення.
* Клієнтська підтримка: Продавці мають доступ до клієнтської підтримки Amazon, яка надає допомогу та відповідає на запитання стосовно платформи та бізнесу продавців.

1. eBay



Рис. 1.7 – Логотип eBay

eBay - це глобальний торговельний онлайн-майданчик, що дає змогу користувачам купувати і продавати широкий спектр товарів. eBay було засновано 1995 року, і він дає змогу мільйонам людей по всьому світу купувати і продавати товари через Інтернет.

Основні особливості eBay полягають у такому:

* Міжнародний ринок: eBay дає змогу користувачам з різних країн продавати і купувати товари. Це створює можливості для глобальної торгівлі та збільшує потенційну аудиторію.
* Аукціони та фіксовані ціни: користувачі можуть виставляти товари на аукціон, а покупці можуть робити конкурентні ставки. Крім того, користувачі можуть продавати товари за фіксованою ціною, не виставляючи їх на аукціон.
* Різноманітність товарів: на eBay можна знайти широкий асортимент товарів, зокрема електроніку, модний одяг, книги, предмети колекціонування, автомобілі та багато іншого.
* Захист покупців і продавців: eBay пропонує систему захисту для вирішення спорів і забезпечення безпеки покупців і продавців.
* Рейтинги та відгуки: користувачі можуть залишати відгуки про угоди, що допомагає зміцнити довіру й оцінити надійність продавця.

1.6 Висновок до розділу 1

Побудовані діаграми програмного продукту, для наочного зображення основних процесів та дії, які виконуватиме система. Проведено огляд основних понять, а також проведений огляд аналогів додатку.

РОЗДІЛ 2 ЗАСОБИ РОЗРОБКИ

2.1 Вибір мови програмування та середовища розробки

Мною була обрана мова програмування С#, яка дуже добре підходила для створення мого ПЗ. Середовище розробки та для редагуванню коду використовувалося Microsoft Visual Studio.

2.1.1 Мова програмування C#

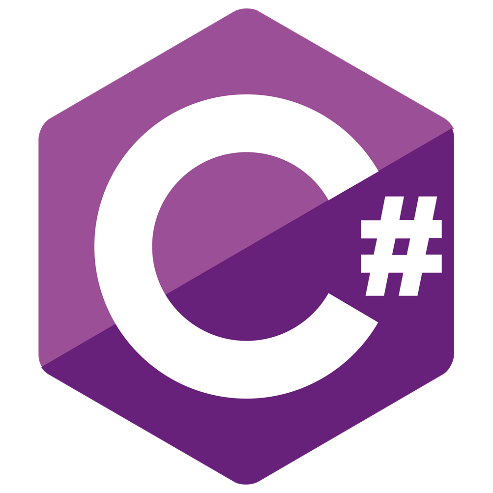


Рис. 2.1 – Логотип С#

Мова програмування C# (C Sharp) була розроблена компанією Microsoft в кінці 1990-х років. Робота над створенням мови розпочалася у 1999 році під керівництвом архітектора Майкла Хіллерсбергера та його команди в рамках проекту Microsoft .NET Framework.

Цілью створення C# було створення мови програмування, яка б поєднувала простоту і зручність використання з потужністю і гнучкістю. Вона мала бути частиною нового фреймворку .NET, який надавав інструменти для розробки програмного забезпечення різних типів і застосувань.

Перший випуск C# відбувся у 2000 році разом з випуском платформи .NET Framework. З того часу мова постійно розвивалася і отримувала оновлення, додаткові функціональні можливості та покращення продуктивності.

C# став основною мовою для розробки програмного забезпечення на платформі .NET, а також отримав підтримку в інших середовищах, таких як Xamarin для розробки мобільних додатків та .NET Core для крос-платформеного програмування.

Сьогодні C# є однією з найпопулярніших мов програмування, особливо в середовищі Microsoft, і вона використовується для розробки різноманітних програмних застосунків, включаючи десктопні додатки, веб-програми, мобільні додатки, хмарні рішення і багато іншого.

Основні принципи мови C# включають:

* Об'єктно-орієнтований підхід: C# призначений для побудови програм з використанням класів і об'єктів. Це дає змогу організувати код у логічні блоки, що полегшує розробку, супровід і розширення додатків.
* Типізація: C# є суворо типізованою мовою, в якій типи даних визначені для всіх змінних. Це полегшує виявлення помилок під час компіляції та роботу над великими проектами.
* Керування пам'яттю: У C# використовується автоматичне керування пам'яттю, тож програмістам не потрібно вручну виконувати операції виділення та деаллокації пам'яті. Крім того, система збирання сміття автоматично виявляє і видаляє непотрібні об'єкти.
* Розширюваність C# підтримує розширення функціональності за допомогою класів розширення, делегатів, подій та інтерфейсів.
* Інтеграція з платформою .NET: C# є членом сімейства мов .NET. Це дозволяє розробляти швидкі додатки, використовуючи багатий набір бібліотек, фреймворків та інструментів.
* Підтримка багатоплатформеності: за допомогою .NET Core можна розробляти кросплатформні додатки, що працюють на різних операційних системах, таких як Windows, macOS і Linux.

2.1.2 Microsoft Visual Studio

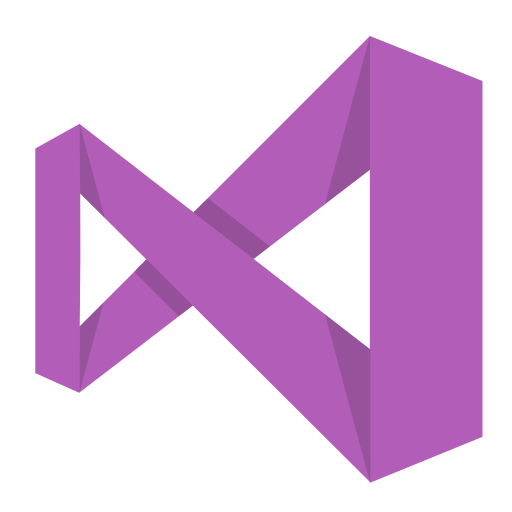


Рис. 2.2 – Логотип Microsoft Visual Studio

Історія Microsoft Visual Studio починається у 1997 році, коли була випущена перша версія цього інтегрованого середовища розробки. Протягом років Visual Studio пройшла великий шлях розвитку та стала одним з найпопулярніших інструментів для програмування.

Початково Visual Studio була спрямована на розробку програм для операційних систем Windows. Згодом було додано підтримку різних мов програмування, таких як C++, Visual Basic, C#, F# та інші, що робило її більш універсальною для розробки різноманітних додатків.

У 2002 році була випущена Visual Studio .NET, що внесла великі зміни у розробку програмного забезпечення. Вона впровадила підтримку платформи .NET Framework, що дозволяє розробляти додатки для різних платформ, включаючи Windows, веб-сайти та мобільні пристрої.

З часом Microsoft продовжувала випускати нові версії Visual Studio з розширеними можливостями та покращеннями продуктивності. Були додані інструменти для розробки веб-додатків, хмарних рішень, мобільних додатків та штучного інтелекту.

На сьогоднішній день Microsoft Visual Studio є одним з найпопулярніших та найвикористовуваніших інструментів розробки програмного забезпечення. Вона продовжує розвиватися та надавати розробникам потужні інструменти для ефективної та продуктивної розробки програм.

Особливості Microsoft Visual Studio включають:

* Підтримка мов: Visual Studio підтримує широкий спектр мов програмування, включно з C#, C++, Visual Basic, JavaScript і Python. Це дає змогу розробникам використовувати ці мови у своїх проектах за допомогою корисних інструментів редагування коду, підказок, налагодження та інших можливостей.
* Візуальний редактор: Visual Studio надає потужний візуальний редактор для швидкого і простого створення користувацьких інтерфейсів. Він має набір графічних інструментів, бібліотек компонентів і функцій швидкої побудови користувацького інтерфейсу.
* Інструменти налагодження: Visual Studio надає розширені інструменти налагодження коду, такі як покроковий перегляд коду, відображення значень змінних і аналіз стеків викликів. Це дає змогу виявляти та виправляти помилки та проблеми у вашій програмі.
* Інтеграція з платформами та сервісами Visual Studio підтримує широкий спектр платформ, включно з Windows, Android та iOS. Вона також інтегрується з низкою сервісів, включно з Azure для хмарної розробки, GitHub для контролю версій і Team Foundation Server для спільної роботи над проектами.
* Розширення та спільноти Visual Studio має низку розширень, які розробники можуть встановити для розширення функціональності IDE. Крім того, існує активна спільнота розробників, які діляться досвідом, публікують корисні матеріали та надають підтримку.

Ці можливості роблять Microsoft Visual Studio потужним інструментом розробки програмного забезпечення, що дає змогу розробникам писати високоякісний код, підтримувати проєкти, налагоджувати додатки та співпрацювати з командами розробників.

2.2 Принципи створення дизайну програмного забезпечення

Основні принципи створення дизайну для програмного забезпечення включають:

* Принцип користувацької центричності: Дизайн повинен бути спрямований на потреби та зручність користувачів. Важливо враховувати їхні очікування, навички та контекст використання програми.
* Принцип простоти та зрозумілості: Дизайн повинен бути простим, зрозумілим і легким у використанні. Мінімізація складності та зайвих елементів дозволяє користувачам швидше орієнтуватися та використовувати програму.
* Принцип послідовності: Дизайн повинен мати послідовність та логіку в усіх його елементах. Це допомагає користувачам передбачувати, які дії та результати очікувати від програми.
* Принцип ефективності та продуктивності: Дизайн повинен допомагати користувачам виконувати завдання швидко та ефективно. Він має спрощувати робочий процес, зменшувати кількість кроків та зусиль, необхідних для досягнення результату.
* Принцип доступності: Дизайн повинен бути доступним для різних категорій користувачів, включаючи людей з обмеженими можливостями. Розробка зрозумілих інтерфейсів, підтримка клавіатурних та голосових команд, врахування конфігураційних налаштувань допомагають забезпечити доступність програми для всіх.
* Принцип гнучкості та розширюваності: Дизайн повинен бути гнучким і легко розширюватися, щоб враховувати змінні потреби та вимоги користувачів. Модульна архітектура, використання стандартних інтерфейсів та розширюваність дозволяють легко внести зміни та покращення в програмне забезпечення.
* Принцип консистентності: Дизайн повинен дотримуватися стандартів та конвенцій, які використовуються в даній платформі або галузі. Це допомагає забезпечити однорідність і зрозумілість для користувачів.
* Принцип естетики: Дизайн повинен бути привабливим та приємним для очей користувачів. Гармонійні кольорові схеми, зручні шрифти та розміщення елементів створюють позитивне враження та сприяють задоволенню користувачів.

Застосування цих принципів у розробці дизайну програмного забезпечення допомагає створити ефективні, зрозумілі та користувачам зручні продукти.

2.3 UML



Рис. 2.3 – Логотип сервісу draw.io

Draw.io - це онлайн-сервіс для створення графіків, діаграм, креслень та інших візуалізацій. Він надає потужні інструменти для створення, редагування та спільної роботи над різними графічними візуалізаціями.

Основні особливості сервісу

Draw.io надає багатий набір інструментів для створення різних видів діаграм, таких як блок-схеми, UML-діаграми, потокові діаграми, схеми баз даних, мережеві діаграми та багато інших.

Сервіс має інтуїтивно зрозумілий інтерфейс, що дозволяє легко створювати та редагувати графічні елементи. Він пропонує різні інструменти креслення, включаючи фігури, стрілки, текстові блоки, зображення та багато інших.

Draw.io дозволяє змінювати розмір, кольори, шрифти та інші атрибути елементів. Ви можете групувати, розгруповувати, копіювати та переміщати елементи для досягнення потрібного вигляду вашої діаграми.

Ви можете зберігати ваші роботи в хмарному сховищі, такому як Google Drive, Dropbox або GitHub. Також є можливість експортувати діаграми у різних форматах, таких як PNG, JPEG, SVG, PDF і багато інших.

Draw.io популярний завдяки своїй багатій функціональності, простоті використання і спільної роботи. Він широко використовується в бізнесі, освіті та інших галузях для створення професійних діаграм, презентацій, технічних схем і багатьох інших видів візуалізації.

2.4 Висновок до розділу 2

Виконаний огляд середовища розробки та мов програмування.

РОЗДІЛ 3 РОЗРОБКА ДОДАТКУ

3.1 Реалізація додатку

На рисунку 3.1, зображений головний екран додатку, після того як ви увійшли в акаунт.

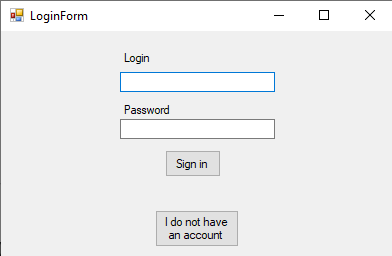


Рис. 3.1 – Вікно «LoginForm»

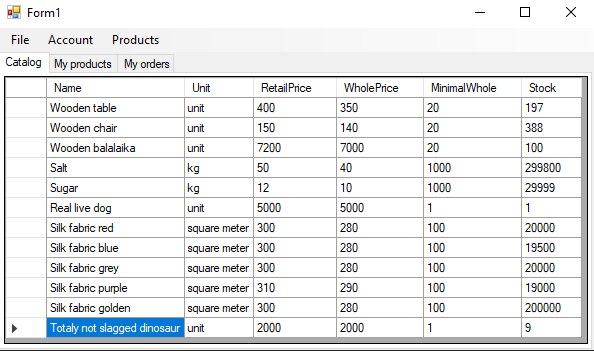


Рис. 3.2 – Головний екран (Catalog)

На головному екрані можна побачити:

* Каталог продуктів (назва, одиниця виміру, роздрібна ціна, оптова ціна, мінімальний опт, на складі)
* Мої продукти (назва, одиниця виміру, роздрібна ціна, оптова ціна, мінімальний опт, на складі)
* Мої замовлення (продавець, покупець, продукт, кількість, ціна, дата та час, спосіб оплати)

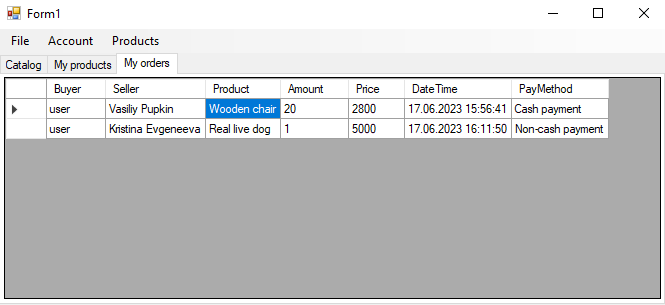


Рис. 3.3 – Вікно «My order»

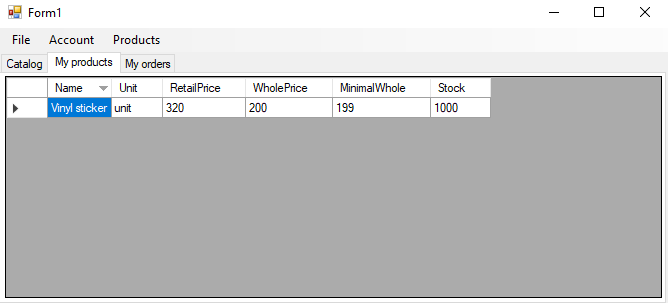


Рис. 3.4 – Вікно «My products»

Так виглядає вікно опису товару(рис. 3.4)

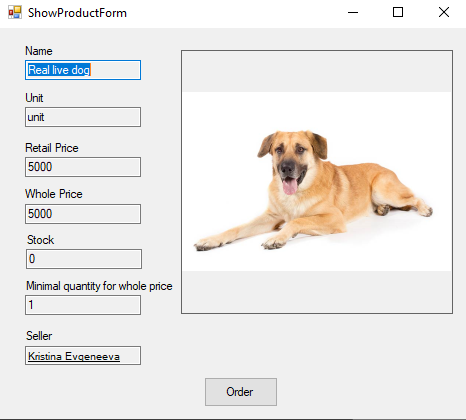


Рис. 3.5 – Вікно «ShowProductForm»

Вікно огляду товару, який ви додаєте до каталогу (рис 3.6). Ви можете редагувати дані про товар в будь-який час.

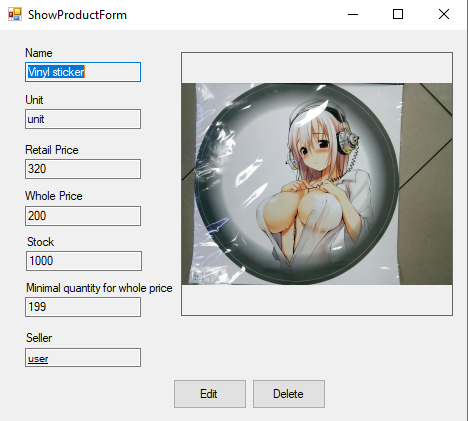


Рис. 3.6 – Вікно вашого товару

Ще на головному екрані, ми можемо бачити такі пункти як:

* File (збереження інформації)
* Account (вихід з аккаунту)
* Product (Додавання та огляд товару)

3.2 Тестування додатку

Щоб протестувати функціонал додатку було використано ПК з такими характеристиками:

* ОЗП: 16gb
* ГП: 1060 3gb
* ЦП: i7-4790k

Тестування проходило таким чином. Користувач відкривав додаток, заходив до свого аккаунту та обирав необхідний йому товар, та робив замовлення.

Для того щоб користувач міг вільно працювати з додатком було вирішено заповнити його випадковим контентом.

Під час тестування було виявлено деякі помилки, які буде усунуто пізніше.

3.3 Висновок до розділу 3

У цьому розділі було створено додаток, а також було проведено тестування на працездатність.

ВИСНОВОК

Під час виконання курсової роботи навчився створювати програмні системи на основі об'єктно-орієнтованого програмування та розвивав навички маніпулювання програмними системами і створення зв'язків між об'єктами.

У результаті було розроблено програму "Біржа товарів", яка дає змогу звичайним користувачам переглядати, замовляти, відображати та редагувати товари.

Додаток написано мовою програмування C# на платформі .NET Framework 6.0 з використанням інтерфейсу програмування Windows Forms.

Усі дані для додатка зберігаються локально у файлі exchange.bin. Додаток можна вдосконалити за рахунок використання реальної бази даних, яка синхронізується з локальними даними біржі через мережу.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Створення UML-діаграм [Електронний ресурс]. URL: https://app.diagrams.net
2. C Sharp [Електронний ресурс]. URL: https://ru.wikipedia.org/wiki/C\_Sharp.
3. Початок роботи. Visual Studio [Електронний ресурс]. URL: https://metanit.com/sharp/tutorial/1.2.php
4. C# Guide [Електронний ресурс]. UML: https://docs.microsoft.com/en-us/dotnet/csharp/
5. Відомості про проекти і рішення на мові C# [Електронний ресурс]. URL: https://docs.microsoft.com/ru-ru/visualstudio/get-started/tutorial projectssolutions?view=vs-2019.

ДОДАТОК

ДОДАТОК А

Лістинг класу Product:

using System;

using System.Drawing;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace CommodityExchange.Models

{

    [Serializable]

    public class Product

    {

        public Guid Id { set; get; }

        public string Name { set; get; }

        public string Unit { set; get; }

        public double RetailPrice { set; get; }

        public double WholePrice { set; get; }

        public double MinimalWhole { set; get; }

        public double Stock { set; get; }

        public User Seller { set; get; }

        public Image Image { set; get; }

        public Product(string name, string unit, double retailPrice, double wholePrice, double minimalWhole, double stock, User seller, Image image)

        {

            Id = Guid.NewGuid();

            Name = name;

            Unit = unit;

            RetailPrice = retailPrice;

            WholePrice = wholePrice;

            MinimalWhole = minimalWhole;

            Stock = stock;

            Seller = seller;

            Image = image;

        }

    }

}

ДОДАТОК Б

Лістинг класу User:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace CommodityExchange.Models

{

    [Serializable]

    public class User

    {

        public Guid Id { set; get; }

        public string Name { set; get; }

        public string Password { set; get; }

        public string Phone { set; get; }

        public string Email { set; get; }

        public User() { }

        public User(string name, string password, string phone, string email)

        {

            Id = Guid.NewGuid();

            Name = name;

            Password = password;

            Phone = phone;

            Email = email;

        }

    }

}

ДОДАТОК В

Лістинг класу Order:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace CommodityExchange.Models

{

    [Serializable]

    public class Order

    {

        public Guid Id { set; get; }

        public User Buyer { set; get; }

        public User Seller { set; get; }

        public Product Product { set; get; }

        public double Amount { set; get; }

        public double Price { set; get; }

        public DateTime DateTime { set; get; }

        public string PayMethod { set; get; }

        public Order(User buyer, User seller, Product product, double amount, double price, string paymethod, DateTime? t = null)

        {

            Id = Guid.NewGuid();

            Buyer = buyer;

            Seller = seller;

            Product = product;

            Amount = amount;

            Price = price;

            PayMethod = paymethod;

            if (t == null)

                DateTime = DateTime.Now;

            else

                DateTime = (DateTime)t;

        }

    }

}