АНОТАЦІЯ

У результаті виконання даної дипломної роботи розроблено автоматизовану інформаційну систему по обслуговуванню клієнтів підприємством, що надає послуги по ремонту мобільних пристроїв.

Для програмної реалізації обрано мову програмування C# , для створення користувальницького інтерфейсу – технологію WPF (Windows Presentation Foundation).

Робота має практичне значення для забезпечення підвищення продуктивності та полегшення управління підприємством, що займається ремонтом мобільних пристроїв.

У першому розділі дипломної роботи досліджена предметна область - організація ремонту мобільних пристроїв підприємством. Визначені цілі, які повинна реалізовувати інформаційна система, та методи її досягнення. Визначені інструментальні засоби розробки системи: СКБД, середовище розробки, платформа розробки та мова програмування.

У другому розділі описується розроблена інформаційна система для організація ремонту мобільних пристроїв підприємством. Представлена схема та структура бази даних. Розроблено UML діаграму варіантів використання та класів. Описаний функціонал та призначення окремих частини готового програмного продукту демонструється на тестовому прикладі.

ANNOTATION

As a result of this thesis developed an automated information system customer service company that provides mobile repair.

For software implementation chosen programming language C #, to create the user interface - technology WPF (Windows Presentation Foundation).

The work has practical value for improving productivity and facilitate management company, engaged in the repair of mobile devices.

In the first chapter of the thesis subject area studied - the organization of mobile repair business. Defined goals that should implement information systems, and methods to achieve it. Identified tools of the system, database, development environment, development platform and programming language.

The second section describes the information system designed for mobile repair organization now. The scheme and structure of the database. Developed UML use case diagram and class. The described functionality and purpose of some of the finished software demonstrated in a test sample.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

АІС – автоматизована інформаційна система;

АС – автоматизовані системи;

АСК – автоматизована система керування;

ІС – інформаційна система;

# ЗМІСТ

[ЗМІСТ 4](#_Toc484480108)

[ВСТУП 4](#_Toc484480109)

[Розділ 1. ОПИС ПРЕДМЕТНОЇ ОБЛАСТІ 4](#_Toc484480110)

[1.1. Актуальність предметної області 4](#_Toc484480111)

[1.2. Класифікація інформаційних систем 4](#_Toc484480112)

[1.3. Основні вимоги до інформаційної системи 4](#_Toc484480113)

[1.4. Опис підприємства «СервісФон». 4](#_Toc484480114)

[1.5. Інструментальні засоби розробки автоматизованої інформаційної системи 4](#_Toc484480115)

[1.5.1 Комп`ютерна платформа .Net Framework 4](#_Toc484480116)

[1.5.2 Мова програмування С# 4](#_Toc484480117)

[1.5.3 Технологія WPF 4](#_Toc484480118)

[ВИСНОВОК 4](#_Toc484480119)

[Розділ 2. ОПИС ОБ`ЄКТУ РОЗРОБКИ 4](#_Toc484480120)

[2.1. Вимоги до програмного та технічного забезпечення 4](#_Toc484480121)

[2.2. Діаграма варіантів використання автоматизованої інформаційної системи 4](#_Toc484480122)

[2.3. Схема та структура бази даних 4](#_Toc484480123)

[2.4. Діаграми класів автоматизованої інформаційної системи 4](#_Toc484480124)

[2.4.1 Опис головного класу для роботи з вікнами 4](#_Toc484480125)

[2.4.2 Опис класу для роботи з базою даних 4](#_Toc484480126)

[2.4.3 Опис класу для обчислення даних, що використовуються при побудові графіків 4](#_Toc484480127)

[2.4.4 Опис класу що забезпечує використання команд 4](#_Toc484480128)

[2.4.5 Опис класу для роботи з головним вікном 4](#_Toc484480129)

[2.4.6 Опис класу для роботи з вікном входу в систему 4](#_Toc484480130)

[2.5. Проектна реалізація автоматизованої інформаційної системи 4](#_Toc484480131)

[ВИСНОВОК 4](#_Toc484480132)

[ВИСНОВКИ 4](#_Toc484480133)

[СПИСОК ВИКОРИСТАНИХ ЛІТЕРАТУРНИХ ДЖЕРЕЛ 4](#_Toc484480134)

# ВСТУП

Сучасне життя неможливе без ефективного керування. Важливою категорією керування є системи обробки інформації, від яких багато в чому залежить ефективність роботи будь-якого підприємства. Дана система повинна забезпечувати одержання загальних звітів за підсумками роботи, дозволяти легко визначати тенденції зміни найважливіших показників, забезпечувати одержання інформації, критичної за часом, без істотних затримок, виконувати точний і повний аналіз даних.

Якщо сучасна людина залишається без мобільного зв`язку, у більшості випадків вона відчуває занепокоєння або паніку. Кожен абонент зробить усе можливе, щоб якнайшвидше підключитися до мережі. Тому не дивно, що у разі пошкодження пристрою ми відразу почнемо пригадувати, де поблизу роблять ремонт мобільних телефонів у Львові.

Мета і задачі дослідження. Метою дипломної роботи є розробка інформаційної системи для підвищення продуктивності та полегшення управління підприємством, що займається ремонтом мобільних пристроїв. Оскільки підвищення ефективності та організації обліку є пріоритетним завдання для цієї системи.

У ході виконання дипломної роботи було виконано наступні завдання:

* досліджено предметну область;
* досліджено принципи роботи підприємства, що займається ремонтом мобільних пристроїв;
* сформовано вимоги для інформаційної системи;
* розроблено схему бази даних;
* розроблено UML діаграму варіантів використання даної АІС ;
* розроблено зручний користувальницький інтерфейс для роботи з програмним продуктом;
* написано код даної АІС.

Об`єктом дослідження єсервісний центр «СервісФон»

Результатом виконання дипломної роботи є завершена та готова до використання інформаційна система для підприємства що займається ремонтом мобільних пристроїв.

# ОПИС ПРЕДМЕТНОЇ ОБЛАСТІ

## Актуальність предметної області

Наявність мобільного телефону на сьогоднішній день є нормою для кожної цивілізованої людини. Завдяки широким функціональним можливостям даний пристрій вже давно витіснив стаціонарні телефони і став незамінним помічником в повсякденному житті кожного. Багато користувачів часто стикаються з проблемою вибору пристрою, від якого залежить не тільки продуктивність і функціональність телефону, але і його надійність. Якщо вибрати неякісний пристрій, то наступною проблемою може стати його ремонт, а також пошук відповідного фахівця.

Очевидним є те, що вартість пристрою в першу чергу обумовлена його функціональністю. Також при купівлі мобільного пристрою потрібно орієнтуватись на особу, що безпоседньо буде ним користуватися. Наприклад, якщо телефон призначений для дитини або літньої людини, перевагу слід віддавати бюджетним моделям, так як дана категорія користувачів не дуже вимоглива до оснащення пристрою, та ймовірність поломки телефона у них доволі мала. За статистикою ремонт мобільних телефонів, які належать дітям і старшим людям відбувається значно рідше, оскільки часто поломки відбуваються через механічні пошкодження. Для людей, які займаються бізнесом, більш придатними є телефони, оснащені широкими комунікаційними можливостями, а також ємним телефонним довідником. Основна споживча група мультимедійних телефонів складається переважно з молоді, для якої головним є наявність великого дисплея, ємною батареї, камери з високою роздільною здатністю, і інших просунутих функцій.

На сьогоднішній день найбільш затребуваним є такий вид мобільних телефонів, як смартфони. За своєю функціональністю смартфони можна сміливо порівняти з портативними кишеньковими комп`ютерами, так як вони здатні виконувати складні обчислювальні операції. А наявність сенсорного екрану дозволило в істотній мірі збільшити екран телефону, що зробило його більш зручним для перегляду фотографій, відео та користування інтернетом. Окремої уваги заслуговує можливість миттєвого виходу через смартфон в інтернет, оскільки з його допомогою можна мати доступ до численних інформаційних ресурсів, без необхідності користуватися стаціонарним комп`ютером.

Ще одним надзвичайно популярним мобільним пристроєм у нас час став планшет. Для багатьох цей клас гаджетів є всього лише іграшкою, яка призначена виключно для розваг. Проте є велика кількість продвинутих користувачів, для яких планшет інструмент, що допомагає працювати з офісними документами, дозволяє з легкістю демонструвати презентації, а також можна читати книги піклуючись про природу, що доволі актуально у нас час. На даний момент лідируючі позиції по продажам займають планшети на операційній системі «Android». На сьогоднішній день у планшетів досить апаратної продуктивності і готових додатків, щоб редагувати фото, відео, аудіо, здійснювати віддалене адміністрування і навіть займатися програмуванням.

Основний мінус планшетів - відсутність клавіатури. Це накладає певні обмеження та незручності у використанні пристрію. Звичайно одна справа - ввести з сенсорного екрану пошуковий запит в адресному рядку, інша - написати статтю на кілька сторінок. У другому випадку ноутбук виявиться куди практичніше. Правда варто відзначити, що на допомогу планшетів можуть прийти Bluetooth-клавіатури і різноманітні док-станції з повнорозмірною клавіатурою.

Говорячи про будь-який пристрій можна з великою достовірністю стверджувати, що чим більше функцій він виконує тим більша ймовірність його скорішого виходу з ладу. Сучасні мобільні пристрої по своїй конструкції і функціональності набагато потужніші ніж багато комп`ютерів п`ять років тому назад, а значить причин поломок може бути безліч.

Буквально декілька років назад при поломці телефону розумілася заміна батареї або шлейфа підключення екрану і клавіатури. Сьогоднішня внутрішня начинка смартфонів і планшетів значно складніша, тому окрім вище вказаних причин з ладу може вийти:

* динаміки;
* процесор;
* модулі безпровідного зв`язку(Bluetooth і Wifi);
* мікрофон;
* мікросхема;
* слот для карти пам`яті;
* камера.

Також значне число поломок смартфонів відбувається через збій у програмному забезпеченні. Це зазвичай призводить до того, що смартфон не вмикається, зависає, може самовільно вимикатися. За статистикою телефони були, і залишаються одним з найбільш уразливих пристроїв, тому ремонт телефонів займає лідируючі позиції серед інших сервісних послуг.

## Класифікація інформаційних систем

Інформаційна система — сукупність організаційних і технічних засобів для збереження та обробки інформації з метою забезпечення інформаційних потреб користувачів.

Інформаційні системи можуть значно різнитися за типами об`єктів управління, характером та обсягом розв`язуваних завдань і рядом інших ознак:

* за рівнем або сферою діяльності — державні, територіальні (регіональні), галузеві, об`єднань, підприємств або установ, технологічних процесів;
* за рівнем автоматизації процесів управління — інформаційно-пошукові, інформаційно-довідкові, інформаційно-керівні, системи підтримки прийняття рішень, інтелектуальні АС;
* за ступенем централізації обробки інформації — централізовані АС, децентралізовані АС, інформаційні системи колективного використання;
* за ступенем інтеграції функцій — багаторівневі АС з інтеграцією за рівнями управління (підприємство — об`єднання, об`єднання — галузь і т. ін.), багаторівневі АС з інтеграцією за рівнями планування.

Державні АС призначені для вирішення найважливіших народногосподарських проблем країни. На базі використання обчислювальних комплексів та економіко-математичних методів у них складають перспективні та поточні плани розвитку країни, ведуть облік результатів та регулюють діяльність окремих ланцюгів народного господарства, розробляють державний бюджет та контролюють його виконання і т. ін.

Центральне місце в мережі державних АС належить автоматизованій системі державної статистики (АСДС). Роль та місце АСДС в ієрархії управління визначається тим, що вона є основним джерелом статистичної інформації, дуже потрібної для функціонування усіх державних та регіональних АС.

Серед АС, з якими взаємодіє АСДС, важливе місце належить автоматизованій системі планових розрахунків (АСПР). АСПР функціонує при Міністерстві економіки України і являє собою інформаційну систему, призначену для розробки народногосподарських планів та контролю за їх виконанням в умовах застосування засобів обчислювальної техніки для збору та обробки інформації.

Процес взаємодії АСДС з АСПР має взаємний характер: статистична інформація, джерелом якої є АСДС, необхідна на всіх етапах складання перспективних і поточних планів розвитку господарства країни. У свою чергу, планова інформація надходить до АСДС і є основою для обліку та аналізу виконання планів і завдань. Взаємодія АСДС та АСПР передбачає також спільний аналіз соціально-економічних проблем розвитку народного господарства. Тому АСДС має повністю задовольнити потреби оптимального планування, проводити економіко-математичний аналіз демографічних процесів у суспільстві, міжгалузевих зв`язків, споживання та прибутків населення, показників діяльності підприємств.

АСДС взаємодіє також з державною інформаційною системою фінансових розрахунків (АСФР) при Міністерстві фінансів України.

АСФР призначена для автоматизації фінансових розрахунків на базі сучасної обчислювальної техніки з формування державного бюджету країни та контролю за його виконанням. При цьому вона використовує статистичну інформацію про випуск і реалізацію продукції, фонди споживання, запаси та витрати фінансових ресурсів і т. ін.

Відомі й інші державні АС, система обробки інформації з цін (АСОІ цін), система управління національним банком (АСУ банк), система обробки науково-технічної інформації (АСО НТІ) і т. ін.

Територіальні (регіональні) АС призначені для управління адміністративно-територіальним регіоном. Сюди належать АС області, міста, району. Ці системи виконують роботи з обробки інформації, яка необхідна для реалізації функцій управління регіоном, формування звітності й видачі оперативних даних місцевим і керівним державним та господарським органам.

Галузеві інформаційні системи управління призначені для управління підвідомчими підприємствами та організаціями. Галузеві, АС діють у промисловості та в сільському господарстві, будівництві на транспорті і т. ін. У них розв`язуються задачі інформаційного обслуговування апарату управління галузевих міністерств і їх підрозділів.

Інформаційні системи управління підприємствами (АСУП) або виробничими об`єднаннями (АСУ В) — це системи із застосуванням сучасних засобів автоматизованої обробки даних, економіко-математичних та інших методів для регулярного розв`язування завдань управління виробничо-господарською діяльністю підприємства.

Інформаційні системи управління технологічними процесами (АСУ ТП) керують станом технологічних процесів виробництва. Перша й головна відмінність цих систем від розглянутих раніше полягає передусім у характері об`єкта управління — це різноманітні машини, прилади, обладнання. Друга відмінність полягає у формі передачі інформації. Для АСУ ТП основною формою передачі інформації є сигнал, а в інших АСУ — документи.

Залежно від мети функціонування та завдань, які покладені на АС на етапах збору та змістової обробки даних, розрізняють такі типи АС:

* інформаційно-пошукові;
* інформаційно-довідкові;
* інформаційно-управляючі (управлінські);
* інтелектуальні інформаційні системи та системи підтримки прийняття рішень;
* інформаційно-пошукові системи (ІСП) орієнтовані на розв`язування завдань пошуку інформації. Змістова обробка інформації в таких системах відсутня.

В інформаційно-довідкових системах (ІДС) за результатами пошуку обчислюють значення арифметичних функцій.

Інформаційно-управляючі, або управлінські, системи являють собою організаційно-технічні системи, які забезпечують вироблення рішення на основі автоматизації інформаційних процесів у сфері управління. Отже, ці системи призначені для автоматизованого розв`язування широкого кола завдань управління.

До інформаційних систем нового покоління належать системи підтримки прийняття рішень (СППР) та інформаційні системи, побудовані на штучному інтелекті (інтелектуальні АС).

СППР — це інтерактивна комп`ютерна система, яка призначена для підтримки різних видів діяльності при прийнятті рішень із слабо структурованих або неструктурованих проблем. Інтерес до СППР, як перспективної галузі використання обчислювальної техніки та інструментарію підвищення ефективності праці в сфері управління економікою, постійно зростає.

Штучний інтелект — це штучні системи, створені людиною на базі комп`ютерної техніки, що імітують розв`язування людиною складаних творчих завдань. Створенню інтелектуальних інформаційних систем сприяла розробка в теорії штучного інтелекту логіко-лінгвістичних моделей. Ці моделі дають змогу формалізувати конкретні змістовні знання про об`єкти управління та процеси, що відбуваються в них, тобто ввести в ЕОМ логіко-лінгвістичні моделі поряд з математичними. Логіко лінгвістичні моделі — це семантичні мережі, фрейми, продукувальні системи — іноді об`єднуються терміном «програмно-апаратні засоби в системах штучного інтелекту».

Розрізняють три види інтелектуальних АС:

* інтелектуальні інформаційно-пошукові системи ( системи типу «запитання — відповідь»), які в процесі діалогу забезпечують взаємодію кінцевих користувачів — непрограмістів з базами даних та знань професійними мовами користувачів, близьких до природних;
* розрахунково-логічні системи, які дають змогу кінцевим користувачам, що не є програмістами та спеціалістами в галузі прикладної математики, розв`язувати в режимі діалогу з ЕОМ свої задачі з використанням складаних методів і відповідних прикладних програм;
* експертні системи, які дають змогу провадити ефективну комп`ютеризацію областей, у яких знання можуть бути подані в експертній описовій формі, але використання математичних моделей утруднене або неможливе.

В економіці України найпоширенішими є експертні системи. Це системи, які дають змогу на базі сучасних персональних комп`ютерів виявляти, нагромаджувати та коригувати знання з різних галузей народного господарства (предметних областей).

## Основні вимоги до інформаційної системи

Сучасні інформаційні системи не залежно від їхнього масштабу, програмно-апаратної платформи і вартості повинні забезпечувати якісне ведення обліку, бути надійними і зручними в експлуатації.

У функціональному аспекті вимогами до даної інформаційної систем є:

* безпомилкові арифметичні розрахунки;
* забезпечення підготовки , заповнення, роздруківки первинних і звітних документів;
* полегшення доступу до бази даних товарів для прийняття рішення, щодо замовлення;
* спрощення та прискорення обробки даних
* забезпечення звертання до даних і звітів за минулі періоди (вести архів).

На підприємстві інформаційна система дає можливість уникати ручної технічної праці при оформленні ремонту, обстеження чи продажу/купівлі запчастин, та швидкого доступу до залишків товарів та перевірки стану виконання роботи. Подання статистики підприємства в електронному вигляді теж являється перевагою інформаційної системи .

Для того щоб забезпечити зазначені можливості, інформаційна система повинна мати єдину базу даних по поточному стану бухгалтерського обліку на підприємстві та запчастин, будь-які відомості з яких можуть бути легко отримані по запиті користувача. У залежності від особливостей обліку на підприємстві бази даних можуть мати різну структуру, але в обов`язковому порядку повинні відповідати структурі прийнятого плану рахунків, що задає основні параметри настроювання системи на конкретну облікову діяльність .

Надійність інформаційної системи в комп`ютерному плані означає захищеність її від випадкових збоїв і в деяких випадках від навмисного псування даних. Як відомо, сучасні персональні комп`ютери є досить відкритими, тому не можна вірогідно гарантувати захист чисто на фізичному рівні. Важливо, щоб після збою зруйновану базу даних можна було легко відновити, а роботу системи відновити в найкоротший термін.

Підприємство створює інформаційну систему, щоб стати більш ефективним й зберігати гроші. На підприємстві з економічної точки зору інформаційна система може розглядатися як засіб, який може вільно замінювати робочу силу.

Сучасна технологія передачі даних дозволяє організовувати роботу більш гнучкими способами, підвищуючи здатність реагувати на зміни в ринку. Інформаційна система надає підприємству додаткову гнучкість.

Таким чином, впровадження системи управління на підприємстві створить передумови для якісного поліпшення процесу управлінського планування й контролю діяльності з боку керівництва.

Також клієнт через впровадження на підприємстві автоматизованої інформаційної системи зможе отримати більш зручніше та якісне обслуговування, що в свою чергу дозволить з економити його час та нерви під час ремонту свого мобільного пристрою.

Дана інформаційна система повинна підтримувати три типи користувачів:

* Адміністратор
* Робітник
* Оператор

Адміністратор - це користувач, що має найбільше прав вданій системі він може додавати нових робітників чи операторів і отримувати статистику про роботу підприємства.

Оператор – це користувач, що безпосередньо взаємодіє з клієнтами та приймає замовлення.

Робітник – це користувач, що займається ремонтом пристроїв та за допомогою інформаційної системи оперативно повідомляє про його стан.

## Опис підприємства «СервісФон».

СервісФон підприємство, що займається ремонтом мобільних телефонів та планшетів, як на гарантійній основі так і за кошти клієнтів, та продажем запасних частин. Структура підприємства зображена на рис1.1.

Дане підприємство складається з 3-х основних відділень:

* Відділення роботи з клієнтом
* Віділення проведення ремонту
* Склад запчаснин



Рис.1.1. (Структура підприємства «СервісФон»)

Також підприємство містить відділ кадрів, що займається набором професійних робітників та менеджерів для спілкування з клієнтами. Для здійснення бухгалтерського обліку господарської діяльності на підприємстві є окреме відділення бухгалтерія, яка тісно співпрацює з відділом матеріально-технічного постачання та відділом ремонту. Бухгалтерія отримує від них необхідні для обліку і контролю документи і надає їм обліково-економічну інформацію.

## Інструментальні засоби розробки автоматизованої інформаційної системи

### Комп`ютерна платформа .Net Framework

.NET Framework - це платформа програмування. Взагалі, комп`ютерна платформа - це апаратний або програмний комплекс, що служить основою для різних обчислювальних систем. Прикладом платформи програмування може бути операційна система комп`ютера. Алгоритмічна мова C# якраз і створена для роботи на платформі .NET.

Розробка програмного забезпечення (ПЗ) на платформах операційних систем (ОС) сімейства Windows мала на меті використання мови програмування C у поєднанні із спеціальними засобами ОС Windows, які називаються скорочено API. Це абревіатура від Application Programming Interface - інтерфейс прикладного програмування. У цьому інтерфейсі зосереджені великі програмні структури, що дозволяють шляхом їх налаштування на конкретне застосування автоматизувати процес трудомісткого програмування на С.

Все заново створене зазвичай дуже недосконале і доробляється в процесі тривалої експлуатації. Потреба відійти від використання безпосередньо в програмуванні засобів API привела до створення досконаліших систем програмування типу, наприклад, Borland C++ Builder, які значно полегшили та зробили важку працю програміста витонченою. Проте життя не стоїть на місці, і мова C на певному етапі перестала забезпечувати потреби програмування. На горизонті з`явилася концепція так званого об`єктно-орієнтованого програмування (ООП), яка дозволяла подивитися на сам процес створення програмного продукту зовсім з іншого боку, даючи програмісту ширші можливості для автоматизації його праці і створення якіснішої програмної продукції.

Основою ООП стали поняття класу та об`єкта. Розробники мови C пішли шляхом додавання до C структури "клас". Вийшла мова C++. Цей процес виявився настільки непростим, що думаю, свого часу самі розробники дуже пошкодували, що прийняли саме таку концепцію бути на рівні сучасних вимог до процесу створення програмного продукту. У гонитві за швидкістю обробки застосуваннями даних і за потрібною надійністю і безпекою роботи застосувань розробникам довелося організовувати два види пам`яті при обробці даних: некеровану (у C пам`яттю доводиться управляти вручну) і керовану (в C++ цю функцію бере на себе спеціальне середовище, так звана керована куча, тому управління пам`яттю - автоматичне), організувати спеціальний і досить неприємний апарат покажчиків.

Але ми знаємо, що чим далі в ліс, тим більше дров. Розробникам довелося будувати апарат переходу між даними з керованої пам`яті в некеровану і навпаки. Легше було поховати C і створити наново іншу мову на новій концепції. Але розробники були зв`язані по руках: дуже багато програмного продукту на C вже працювало в світі, і поставити на ньому хрест означало підірвати виробничий процес багатьох підприємств та організацій. Тому доводилося не тільки піклуватися про збереження C, але і дотримуватися сучасних вимог (створення C++), підтримувати сумісність старих програм при роботі в нових середовищах. Тобто потрібно було тягнути за собою хвости C в нову мову C++, які тільки заважали новій мові та ускладнювали процес розробки програм на цій мові.

Врешті решт, мабуть, у розробників терпець увірвався, і вони створили нову мову під назвою C#, яка враховує нові віяння в програмуванні (ООП) і вільну від недоліків C++. Проте і C++ не виявилася покинутою з причини, відміченої раніше (сумісність і підтримка вже працюючих у світі програм).

Якщо встановити безкоштовний продукт фірми Microsoft .NET 4.0 Framework Software Development Kit(SDK) або середовище Visual Studio 2010, то для програмування на основі платформи .NET стають доступними мови C#, F#, JScript .NET, Visual Basic, C++/CLI. Тут CLI (Common Language Infrastructure, спільномовна інфраструктура) - прив`язка C++ до платформи .NET. Повернемося все таки до платформи .NET, на базі якої функціонує C#.

Ця платформа є програмною платформою для створення застосувань не лише на базі ОС сімейства Windows, але й інших ОС, які створювалися не фірмою Microsoft, як Windows. Це системи Mac OS X, UNIX, Linux. Платформа забезпечує взаємодію із вже існуючим програмним забезпеченням. Додатки на платформі .NET можна створювати за допомогою багатьох мов програмування, таких як C#, F#, S#, Visual Basic та ін.

Сьогодні фірма Microsoft випускає продукт під назвою Visual Studio (2017), який дає можливість створювати додатки різними мовами на платформі .NET. Усі мови, підтримувані .NET мають спільний виконуючий механізм. Платформа містить в собі велику і, що важливо, спільну для усіх підтримуваних мов бібліотеку базових класів, які забезпечують, наприклад, введення-виведення даних, роботу додатків з графічними об`єктами, створення не лише веб-інтерфейсів, але і звичайних (настільних) і консольних (без графіки) застосувань, роботу з базами даних, дають можливість створювати інтерфейси для роботи з віддаленими об`єктами.

Зокрема, платформа .NET Framework – це кероване середовище виконання, що надає різноманітні служби працюючим в ньому застосуванням. Вона складається з двох основних компонентів: виконавчого середовища спільної мови (Common Language Runtime, CLR), яка є механізмом, управляючим застосуванням, яке виконуються, і бібліотеки класів .NET Framework, яка представляє бібліотеку перевіреного коду, призначену для повторного використання, який розробники можуть викликати зі своїх застосувань. Служби (точніше - сервіси, а ще точніше - послуги), які платформа .NET Framework надає працюючим застосуванням:

* управління пам`яттю. У багатьох мовах програмування розробники самостійно призначають і виділяють ресурси пам`яті і вирішують питання, пов`язані з часом життя об`єктів. У застосуваннях платформи .NET Framework середовище CLR надає ці сервіси автоматично;
* система спільного типу. У традиційних мовах програмування базові типи визначаються компілятором, що ускладнює взаємодію між мовами. У платформі .NET Framework базові типи визначаються єдиною системою типу .NET Framework, яка називається CTS (Common Type System). При цьому використовуються одні і ті ж базові типи для усіх мов .NET Framework;
* розширена бібліотека класів. Замість того щоб писати багато коду для виконання стандартних низькорівневих операцій програмування, розробники можуть використати легкодоступну бібліотеку типів і членів з бібліотеки класів .NET Framework;
* платформи і технології розробки. Платформа .NET Framework включає бібліотеки для конкретних областей розробки застосувань, наприклад ASP.NET для веб-застосувань, ADO.NET для доступу до даних і Windows Communication Foundation для застосувань, орієнтованих на служби (сервіси);
* взаємодія мов. Мовні компілятори на платформі .NET Framework компілюють додаток не у виконуваний код відразу, а в проміжний код, що називається мовою CIL (Common Intermediate Language), який згодом компілюється під час виконання застосування середовищем CLR. Такий підхід приводить до того, що програми, написані на одній мові, доступні в інших мовах, а розробники можуть зосередитися на створенні додатків на мові, якій віддається перевага, або мовах;
* сумісність версій. За рідкісними виключеннями, додатки, які розробляються за допомогою платформи .NET Framework певної версії, можуть виконуватися без змін на більш пізній версії;
* паралельне виконання. Платформа .NET Framework допомагає у вирішенні конфліктів версій, дозволяючи встановлення декількох версій середовища CLR на одному комп`ютері. Це означає, що декілька версій додатків також можуть співіснувати, і що застосування може виконуватися на версії платформи .NET Framework, для якої воно було створене;
* налаштування для різних версій. Орієнтуючись на переносність бібліотеки класів платформи .NET Framework, розробники можуть створювати збірки (exe - або dll- файли, призначені для виконання), які працюють на декількох платформах .NET Framework. Наприклад, на .NET Framework, Silverlight, Windows, Phone 7 або Xbox 360.

Якщо ви не розробляєте застосування.NET Framework, але використовуєте їх, ви не повинні володіти якимись спеціальними знаннями про платформу .NET Framework або її роботу.

Якщо використовується ОС Windows, платформа .NET Framework може бути вже встановлена на комп`ютері. Крім того якщо встановлюється додаток, що вимагає платформу .NET Framework, програма встановлення застосування може інсталювати конкретну версію .NET Framework на вашому комп`ютері. Інколи можна побачити діалогове вікно, яке просить встановити платформу .NET Framework.

Зазвичай, не вимагається видаляти які-небудь версії .NET Framework вже встановлені на вашому комп`ютері, тому що використовуваний додаток може залежати від конкретної версії. У разі видалення якої-небудь версії його виконання може завершитися помилкою. Зверніть увагу, що на одному комп`ютері може бути одночасно завантажено декілька версій платформи .NET Framework. Це означає, що не треба видаляти попередні версії для встановлення пізнішої версії.

Розробник може вибрати будь-яку мову програмування, яка підтримує платформу .NET Framework, для створення застосування. Через те, що платформа .NET Framework забезпечує незалежність і взаємодію мов, можна взаємодіяти з іншими застосуваннями компонентами платформи .NET Framework незалежно від мови, за допомогою якої вони були розроблені.

### Мова програмування С#

Мова С# (вимовляеться "Сі-шарп") — це багатопарадигмова о6ектнo- оріентована та компонентно-оріентована мова програмування зі строгою типизаціею, розроблена для платформи .NET Framework. Використання мови визначене стандартами ЕСМА-3344 [4] та ISO/ІЕС 23270:20065 [5]. Розроблена командою Microsoft Reseazch під керівництвом Андерса Гейлсберга. Синтаксис мови близький до мов С++ та Java.

Деякі риси (нanриклад, сгрога статична типизація), наближують їі струкryру до Delphi (Object Pascal).Цілі, поставлені при розробці мови С#, були такими;

* С# має бути простою, сучасною, об`єктно-орієнтованою мовою програмування;
* мова має підтримувати безпечні принципи програмування, такі як строга перевірка типів, перевірка меж масиву, виявлення спроб використання неініціалізованих змінних, і автоматичне прибирання сміття;
* можливість розробки програмних компонентів для розподілених систем;
* мова має підтримувати переносимість коду;
* підтримка національних мовних та інших особливостей має бути простою.

Базовий синтаксис С# схожий до інших С-подібних мов, таких як С, С++ та Java, зокрема:

* крапку з комою використовують для позначення кінця інструкції;
* фігурні дужки використовують для формування програмних блоків;
* значення змінним присвоюють за допомогою знаку "=", а для їх порівняння використовують "= =";
* квадратні дужки використовують для індексації масивів.

Проте, С# має суттєві відмінності від розглянутих мов- попередників, у тому числі:

* краща переносимість між різними платформами, пов`язана з тим, що мова відповідаї специфікації CLI;
* строга типизація робить мову безпечнішою. С# підтримує логічний тип Boolean;
* мова забезпечуе використання властивостей у класах, роблячи роботу з об`єктами зручнішою та безпечнішою;
* програма на С# краще структурована завдяки групуванню коду в простори імен;
* обмежене використання вказівників робить безпечнішою роботу з пам`яттю.

Перша версія С# 1.0 вийшла разом з Microsoft Visual Studio .NET y лютому 2002 року. В липні 2015 року випущено версію С# 6.0. Порівняно із першою версією, мову С# суттєво розвинули, доповнивши ії рядом вагомих рис, які роблять їі сучасною, гнучкою та перспективною для вирішення широкого кола завдань. Існує кілька середовищ програмування, які використовують мову С#, зокрема: Microsoft Visual Studio, MonoDevelop, SharpDevelop тошо.

Найвживанішим з них є інтегроване середовище розробки Microsoft Visual Studio, яке, крім С#, підтримує і ряд інших мов програмування, а також забезпечуе великий набір класів для розв`язування різних завдань.

### Технологія WPF

Технологія WPF (Windows Presentation Foundation) є частина платформи .NET і являє собою підсистему для побудови графічних інтерфейсів для користувачів.

Якщо при створенні традиційних додатків на основі WinForms за малювання елементів управління і графіки відповідали такі частини операційної системи Windows, як User32 і GDI +, то додатки WPF засновані на DirectX. У цьому полягає ключова особливість рендеринга графіки в WPF: використовуючи WPF, значна частина роботи по відображенні графіки, як найпростіших кнопочок, так і складних 3D-моделей, лягає на графічний процесор на відеокарті, що також дозволяє скористатися апаратним прискоренням графіки.

Однією з важливих особливостей є використання мови декларативної розмітки інтерфейсу XAML, заснованого на XML: ви можете створювати насичений графічний інтерфейс, використовуючи або декларативне оголошення інтерфейсу, або код на керованих мовах C # і VB.NET, або поєднувати і те, і інше.

Переваги використання WPF:

* використання традиційних мов .NET-платформи - C # і VB.NET для створення логіки додатка;
* можливість декларативного визначення графічного інтерфейсу за допомогою спеціальної мови розмітки XAML, заснованому на xml і представляє альтернативу програмному створення графіки та елементів управління, а також можливість комбінувати XAML і C # / VB.NET;
* незалежність від дозволу екрану: оскільки в WPF всі елементи вимірюються в незалежних від пристрою одиницях, додатки на WPF легко масштабуються під різні екрани з різним розширенням;
* нові можливості, яких складно було досягти в WinForms, наприклад, створення тривимірних моделей, прив`язка даних, використання таких елементів, як стилі, шаблони, теми і ін;
* хорошу взаємодію з WinForms, завдяки чому, наприклад, в додатках WPF можна використовувати традиційні елементи управління з WinForms.
* багаті можливості по створенню різних додатків: це і мультимедіа, і двомірна і тривимірна графіка, і багатий набір вбудованих елементів управління, а також можливість самим створювати нові елементи, створення анімацій, прив`язка даних, стилі, шаблони, теми і багато іншого;
* апаратне прискорення графіки - незалежно від того, чи працюєте ви з 2D або 3D, графікою або текстом, все компоненти програми транслюються в об`єкти, зрозумілі Direct3D, і потім візуалізуються за допомогою процесора на відеокарті, що підвищує продуктивність, робить графіком більш плавною;
* створення додатків під безліч ОС сімейства Windows - від Windows XP до Windows 10

У той же час WPF має певні обмеження. Незважаючи на підтримку тривимірної візуалізації, для створення додатків з великою кількістю тривимірних зображень, перш за все ігор, краще використовувати інші засоби - DirectX або спеціальні фреймворки, такі як Monogame або Unity.

Також варто враховувати, що в порівнянні з додатками на Windows Forms обсяг програм на WPF і споживання ними пам`яті в процесі роботи в середньому трохи вище. Але це з лишком компенсується більш широкими графічними можливостями і провишенной продуктивністю при відображенні

Схематично архітектура WPF зображена на рис.1.2.



Рис.1.2.(Архітектура WPF)

Як видно на схемі, WPF розбивається на два рівня: managed API і unmanaged API (рівень інтеграції з DirectX). Managed API (керований API-інтерфейс) містить код, що виконується під управлінням загальномовного середовища виконання .NET - Common Language Runtime. Цей API описує основний функціонал платформи WPF і складається з наступних компонентів:

* PresentationFramework.dll: містить всі основні реалізації компонентів і елементів управління, які можна використовуються при побудові графічного інтерфейсу;
* PresentationCore.dll: містить всі базові типи для більшості класів з PresentationFramework.dll;
* WindowsBase.dll: містить ряд допоміжних класів, які застосовуються в WPF, але можуть також використовуватися і поза даної платформи

Unmanaged API використовується для інтеграції вищого рівня зDirectX;

* milcore.dll власне забезпечує інтеграцію компонентів WPF з DirectX. Даний компонент написаний на некерованому коді (С / С ++) для взаємодії з DirectX;
* WindowsCodecs.dll: бібліотека, яка надає низкоуровневую підтримку для зображень вWPF.

Ще нижче власне знаходяться компоненти операційної системи і DirectX, які проізвоітся візуалізацію компонентів програми, або виконують іншу низкоуровневую обробку. Зокрема, за допомогою низкоуровневого інтерфейсу Direct3D, який входить до складу DirectX, відбувається трансляція.

Тут також на одному рівні знаходиться бібліотека user32.dll. І хоча вище говорилося, що WPF не використовує цю бібліотеку для рендеринга і візуалізації, проте для ряду обчислювальних задач (що не включають візуалізацію) дана бібліотека продовжує використовуватися.

WPF є частиною .NET і розвивається разом з фреймворком .NET і має ті ж версії. Перша версія WPF 3.0 вийшла разом з .NET 3.0 і операційною системою Windows Vista в 2006 році. З тих пір платформа послідовно розвивається. Остання версія WPF 4.6 вийшла паралельно з .NET 4.6 в липні 2015 року, ознаменувавши дев`ятиріччя даної платформи.

# ВИСНОВОК

В даному розділі досліджено предметну область обслуговування клієнтів підприємством, що надає послуги по ремонту мобільних пристроїв, представлені основні вимоги до системи. Розглянуто інструментальні засоби для розробки автоматизованої інформаційної системи.

# ОПИС ОБ`ЄКТУ РОЗРОБКИ

## Вимоги до програмного та технічного забезпечення

## Діаграма варіантів використання автоматизованої інформаційної системи

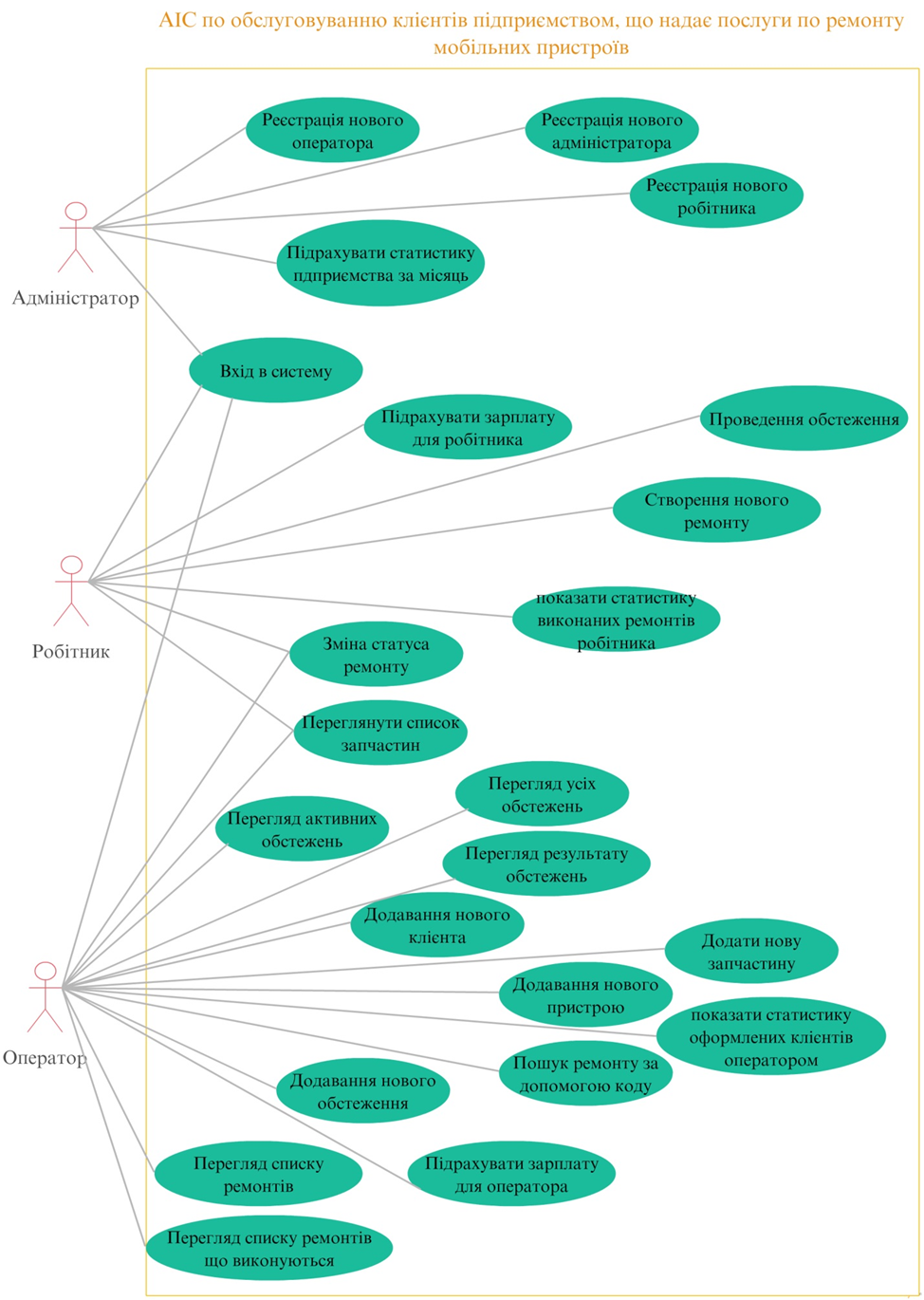


Рис.2.1.(Діаграма варіантів використання АІС)

Діаграма прецедентів є графом, що складається з множини акторів, прецедентів (варіантів використання) обмежених границею системи (прямокутник), асоціацій між акторами та прецедентами, відношень серед прецедентів, та відношень узагальнення між акторами. Діаграми прецедентів відображають елементи моделі варіантів використання.

Для підприємста, що займається ремонтом мобільних пристроїв розроблена діаграма варіантів використання зображена на рис.2.1.

## Схема та структура бази даних

Для створення АІС підприємства, що надає послуги по ремонту мобільних пристроїв була розроблена база даних, схема якої представлена на рис.2.2 за допомогою якої можна досягти поставлених цілей.

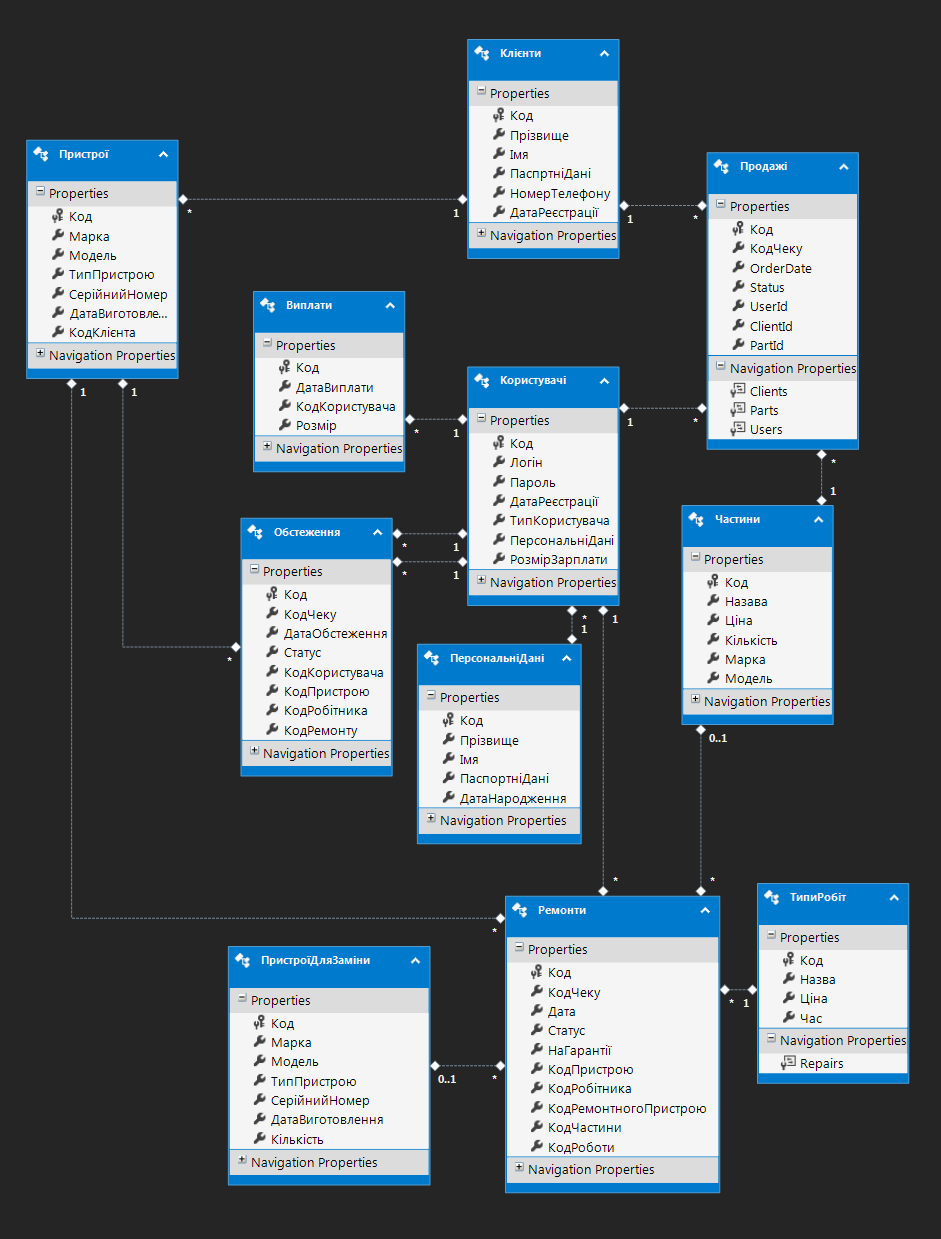


Рис.2.2(Схема зв’язків між таблицями бази даних)

База даних складається з 11 таблиць:

* клієнти;+
* пристрої;+
* частини;+
* користувачі;+
* виплати;+
* персональні дані;+
* продажі;
* типи робіт;+
* ремонти;+
* обстеження;
* пристрої на час ремонту.

Таблиця 2.1

Структура таблиці Клієнти в базі даних

|  |  |  |
| --- | --- | --- |
| Назва | Тип | Опис |
| Код | int | Ключове поле |
| Прізвище | nvarchar(15) | Прізвище клієнта |
| Ім`я | nvarchar(15) | Ім`я клієнта |
| Паспортні дані | nvarchar(15) | Серія і код паспорта |
| Номер телефону | nvarchar(15) | Мобільний номер клієнта |
| Дата реєстрації | date | Дата реєстрація клієнта |

У табл. 2.1 наведено перелік назв полів, їх тип даних та опис їх призначення для таблиці «Клієнти». В даній таблиці зберігається інформація про кожного клієнта, що обслуговується підприємством.

Таблиця 2.2

Структура таблиці Пристрої в базі даних

|  |  |  |
| --- | --- | --- |
| Назва | Тип | Опис |
| Код | int | Ключове поле |
| Марка | nvarchar(50) | Марка виробника |
| Модель | nvarchar(50) | Модель даного пристрою |
| Тип пристрою | int | Тип пристрою може бути телефон або планшет |
| Серійний номер | nvarchar(50) | Унікальний код пристрою |
| Дата виготовлення | date | Дата виготовлення пристрою |
| Код клієнта | int | Код власника пристрою |

У табл. 2.2 наведено перелік назв полів, їх тип даних та опис їх призначення для таблиці «Пристрої». В даній таблиці міститься інформація про мобільні пристрої, що були на ремонті і що зараз перебувають на стадії ремонту

Таблиця 2.3

Структура таблиці Частини в базі даних

|  |  |  |
| --- | --- | --- |
| Назва | Тип | Опис |
| Код | int | Ключове поле |
| Назва частини | nvarchar(500) | Назва запчастини її характеристика |
| Ціна | float | Ціна частини |
| Кількість | int | Кількість частин що є в наявності |
| Марка | nchar(50) | Марка для якої підходить запчастина |
| Модель | nchar(50) | Модель для якої підходить частина |

У табл. 2.3 наведено перелік назв полів, їх тип даних та опис їх призначення для таблиці «Частини». В даній таблиці міститься інформації про частини, що використовуються під час ремонту телефонів або планшетів та частини для продажу.

Таблиця 2.4

Структура таблиці Користувачі в базі даних

|  |  |  |
| --- | --- | --- |
| Назва | Тип | Опис |
| Код | int | Ключове поле |
| Логін | nvarchar(15) | Логін для входу в систему |
| Пароль | nvarchar(15) | Пароль для входу в систему |
| Дата реєстрації | date | Дата реєстрація користувача |
| Тип користувача | nvarchar(15) | Тип користувача може бути адміністратор, оператор або робітник |
| Персональні дані | int | Код персональних даних про користувача |
| Розмір зарплати | float | Розмір зарплати для користувача |

У табл. 2.4 наведено перелік назв полів, їх тип даних та опис їх призначення для таблиці «Користувачі». В даній таблиці міститься інформації про користувачів, що можуть здійснювати вхід в систему, та визначає права та можливості використання певних функцій системи.

Таблиця 2.5

Структура таблиці Виплати в базі даних

|  |  |  |
| --- | --- | --- |
| Назва | Тип | Опис |
| Код | int | Ключове поле |
| Дата | date | Дата здійснення виплати |
| Розмір | float | Розмір здійсненої виплати |
| Код користувача | int | Код користувача для якого була здійснена виплата |

У табл. 2.5 наведено перелік назв полів, їх тип даних та опис їх призначення для таблиці «Виплати». В даній таблиці міститься інформації про витрати, що підприємство виплачує своїм операторам та робітникам.

Таблиця 2.6

Структура таблиці Персональні дані в базі даних

|  |  |  |
| --- | --- | --- |
| Назва | Тип | Опис |
| Код | int | Ключове поле |
| Прізвище | nvarchar(15) | Прізвище користувача |
| Ім`я | nvarchar(15) | Ім`я користувача |
| Паспортні дані | nvarchar(15) | Серія і код паспорта користувача |
| Дата народження | date | Дата народження користувача |

У табл. 2.6 наведено перелік назв полів, їх тип даних та опис їх призначення для таблиці «Персональні дані». В даній таблиці міститься інформації про персональні дані користувачів, що можуть здійснювати вхід в систему.

Таблиця 2.7

Структура таблиці Типи робіт в базі даних

|  |  |  |
| --- | --- | --- |
| Назва | Тип | Опис |
| Код | int | Ключове поле |
| Назва | nvarchar(15) | Назва роботи |
| Ціна | float | Ціна виконання роботи |
| Час | time | Час виконання роботи |

У табл. 2.7 наведено перелік назв полів, їх тип даних та опис їх призначення для таблиці «Типи робіт». В даній таблиці міститься інформації про типи робіт, які може проводити підприємство.

Таблиця 2.8

Структура таблиці Ремонти в базі даних

|  |  |  |
| --- | --- | --- |
| Назва | Тип | Опис |
| Код | int | Ключове поле |
| Код Чеку | int | Код чеку ремонту |
| Дата | date | Дата оформлення ремонту |
| Статус | int | Статус ремонту |
| Гарантійний ремонт | bit | Телефон знаходиться на гарантії |
| Код пристрою | int | Код пристрою над яким відбувається ремонт |
| Код робітника | int | Код робітника що здійснює ремонт |
| Код запасного пристрою | int | Код запасного пристрою на час ремонту |
| Код частини | int | Код частини, що використовується для ремонту |
| Код роботи | int | Код роботи,що здійснюється |

У табл. 2.8 наведено перелік назв полів, їх тип даних та опис їх призначення для таблиці «Ремонти». В даній таблиці міститься інформація про ремонти, що проводились та ті що зараз виконуються.

Таблиця 2.2

Структура таблиці Пристрої на час ремонту в базі даних

|  |  |  |
| --- | --- | --- |
| Назва | Тип | Опис |
| Код | int | Ключове поле |
| Марка | nvarchar(50) | Марка виробника |

Продовження таблиці 2.9

|  |  |  |
| --- | --- | --- |
| Модель | nvarchar(50) | Модель даного пристрою |
| Тип пристрою | int | Тип пристрою може бути телефон або планшет |
| Серійний номер | nvarchar(50) | Унікальний код пристрою |
| Дата виготовлення | date | Дата виготовлення пристрою |
| Кількість | int | Кількість штук в наявності |

У табл. 2.9 наведено перелік назв полів, їх тип даних та опис їх призначення для таблиці «Пристрої на час ремонту». В даній таблиці міститься інформація про мобільні пристрої, що видаються клієнтам на час ремонту.

Таблиця 2.10

Структура таблиці Продажі в базі даних

|  |  |  |
| --- | --- | --- |
| Назва | Тип | Опис |
| Код | int | Ключове поле |
| Код Чеку | int | Код чеку продажі |
| Дата | date | Дата оформлення продажу |
| Статус | int | Статус продажі |
| Код користувача | int | Код користувача, що здійснює продаж |
| Код частини | int | Код часІтини для продажі |
| Кількість частин | int | Кількість частин |
| Код клієнта | int | Код клієнта |

У табл. 2.8 наведено перелік назв полів, їх тип даних та опис їх призначення для таблиці «Продажі». В даній таблиці міститься інформація про продані частини.

Таблиця 2.11

Структура таблиці Обстеження в базі даних

|  |  |  |
| --- | --- | --- |
| Назва | Тип | Опис |
| Код | int | Ключове поле |
| Код Чеку | int | Код чеку обстеження |
| Дата | date | Дата оформлення обстеження |
| Статус | int | Статус обстеження |
| Код оператора | int | Код оператора, що оформляє обстеження |
| Код пристрою | int | Код пристрою |
| Код робітника | int | Код робітника, що здійснюватиме обстеження |
| Код ремонту | int | Код ремонту |

У табл. 2.8 наведено перелік назв полів, їх тип даних та опис їх призначення для таблиці «Продажі». В даній таблиці міститься інформація про активні обстеження та ті що були здійснені раніше.

## Діаграми класів автоматизованої інформаційної системи

### Опис головного класу для роботи з вікнами

Клас WindowsFactory призначений для відкривання нових вікон у програмі. Він визначає методи відкриття вікон. За його допомогою синхронізується інформація у всіх вікнах. На рис.2.3 представлена UML діаграма класу.

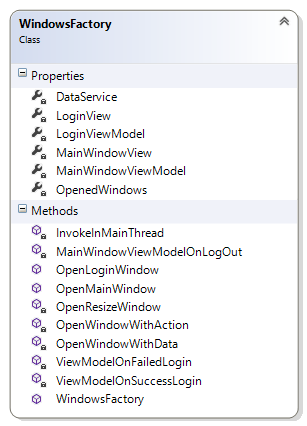


Рис.2.3.(UML діаграма класу для роботи з вікнами)

Цей клас виконує функцію ядра програми, при запуску програми створюється спочатку екземпляр класу WindowsFactory після чого він відкриває вікно входу в систему. Кожне нове вікно додається до списку відкритих вікон за допомогою чого можна дізнатись, яке вікна зараз є відкрити і оперативно закрити усі відкриті вікна.

### Опис класу для роботи з базою даних

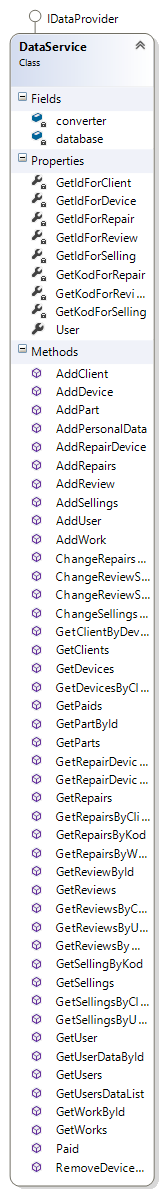


Рис.2.4.( UML діаграма класу для роботи з базою даних)

Клас DataService призначений для взаємодії з сховищем даних Він реалізує інтерфейс IDataProvider, що забезпечує асинхронну роботу з базою даних. На рис.2.4 представлена UML діаграма класу.

### Опис класу для обчислення даних, що використовуються при побудові графіків

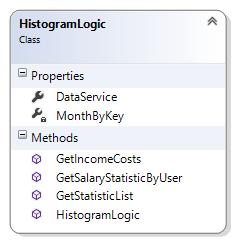


Рис.2.5.( UML діаграма класу для обчислення даних, що використовуються при побудові графіків)

Клас HistogramLogic призначений для опрацювання даних, що містяться в базі даних і їх використання для побудови гістограм та кругових діаграм. На рис.2.5 представлена UML діаграма класу.

### Опис класу що забезпечує використання команд

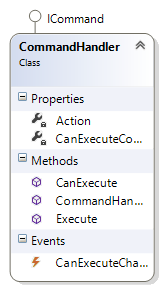


Рис.2.6.( UML діаграма класу використання команд)

Клас CommandHandler призначений для взаємодії користувача системи з самою системою. У технології WPF усі дії з графічним інтерфейсом відбуваються за рахунок інтерфейсу ICommand, що і реалізує даний клас. На рис.2.6 представлена UML діаграма класу.

### Опис класу для роботи з головним вікном

Клас MainWindowViewModel призначений для роботи з головним вікном програми. Він містить команди, що дозволяють відкривати вікна та доступатися до бази даних. На рис.2.7 представлена UML діаграма класу.

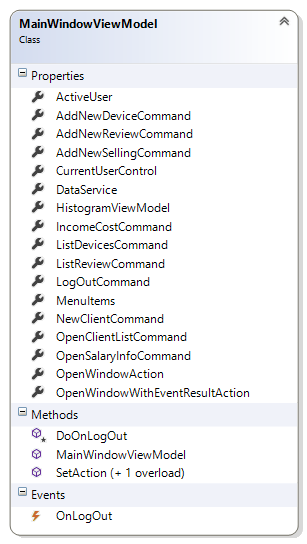


Рис.2.7.( UML діаграма класу для роботи з головним вікном)

### Опис класу для роботи з вікном входу в систему

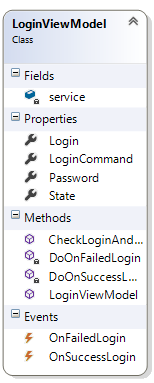


Рис.2.8.( UML діаграма класу для роботи з вікном входу в систему)

Клас LoginViewModel призначений для роботи з вікном входу в систему. Він містить команду, що доступається до бази даних та перевіряє чи існує такий користувач. У разі знаходження такого користувача спрацьовує подія успішного входу в систему, що відкриває головне вікно програми. В іншому разі спрацьовує подія помилкового входу в систему. На рис.2.8 представлена UML діаграма класу.

## Проектна реалізація автоматизованої інформаційної системи

Дана автоматизована інформаційна система підтримує три типи користувачів і для того щоб зайти у систему потрібно знати логін і пароль.

Для того щоб додати нового користувача потрібно мати права адміністратора. Першого користувача системи потрібно додати в базу даних за допомогою допоміжних програм. На рис.2.9 зображено інтерфейс додавання нового користувача.

Рис.2.9.(Вікно додавання нового користувача системи)

Після того коли ви отримали свій логін і пароль ви можете за їх допомогою увійти в систему ввівши їх у відповідні текстові поля вікна входу, що зображено на рис.2.10.

Рис.2.10.(Вікно входу в систему)

Якщо ж ви ввели не коректний логін або пароль то вам вискочить попередження, зображене на рис.2.11.

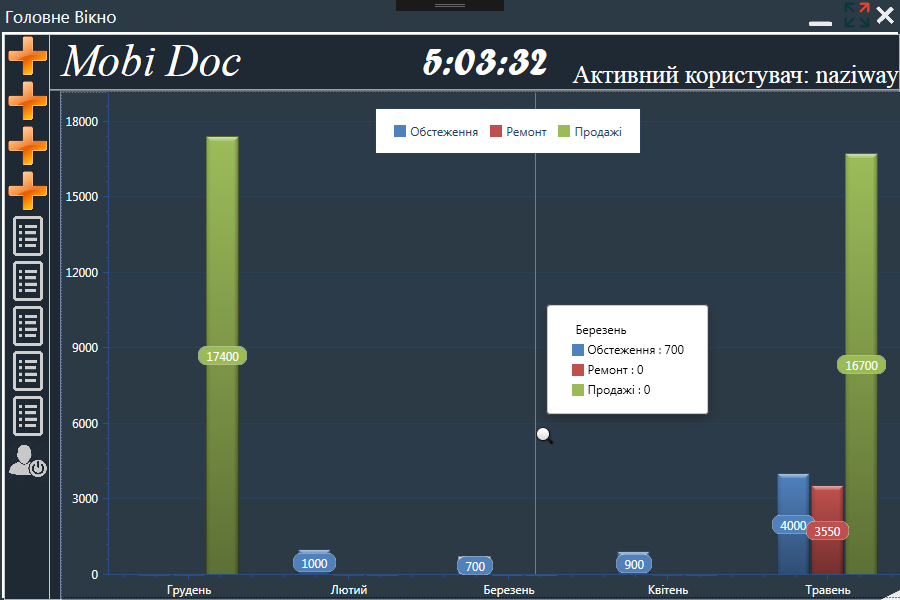
Рис.2.10.(Вікно не коректного входу в систему)

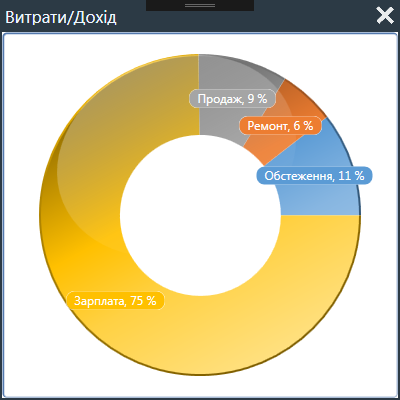
При вході в систему для користувача відкривається головне вікно програми і в залежності від прав доступу йому будуть доступні різні функції системи. На рис.2.11 представлено вигляд головного вікна програми, а на рис.2.12 зображено набори команд для різних типів користувачів.

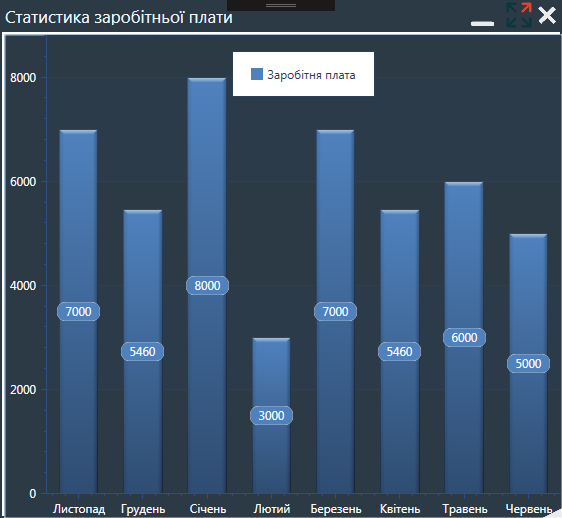
Рис.2.11(Головне вікно програми)

Рис.2.12(Панелі команд)

Головними об







# ВИСНОВОК

# ВИСНОВКИ

# СПИСОК ВИКОРИСТАНИХ ЛІТЕРАТУРНИХ ДЖЕРЕЛ